Set up the environment

In [ ]:

```
# install pycaret
! pip install pycaret

#a1
# from pycaret.classification import *
# import pycaret
```

In [ ]:

```
# import libraries

import pandas as pd
import numpy as np
# eda
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
#pre
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.preprocessing import LabelEncoder, StandardScaler
import scipy.stats as stats
from sklearn.impute import KNNImputer
from imblearn.over_sampling import SMOTE

#a2
from sklearn.ensemble import RandomForestClassifier

#  evaluate
from sklearn import metrics
import pickle
#

#
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:

```
# Path in Colab
from google.colab import files
from google.colab import drive

drive.mount('/content/drive')

PATH_TRAIN = r'/content/drive/MyDrive/ds/train.csv'
```

```
Mounted at /content/drive
```

In [ ]:

```
# Path in My PC
# PATH_TRAIN = r''
```

In [ ]:

```
### MY FUNCTIONS ###

# remove _ from _8 | 8_
def Remove_value(datafram, col):
    datafram[col] = [str(s).replace('_', '') for s in datafram[col]]

# convert to dt

def ConvertDataType(datafram, col, dt):
    datafram[col] = datafram[col].astype(dt)
```

Introduction

In [ ]:

```
# Read Data
df_train = pd.read_csv(PATH_TRAIN, encoding='utf-8', sep=',')
```

In [ ]:

```
df_train.head(5)
```

Out[6]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly
0	0x1602	CUS_0xd40	January	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	
1	0x1603	CUS_0xd40	February	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	
2	0x1604	CUS_0xd40	March	Aaron Maashoh	-500	821-00-0265	Scientist	19114.12	
3	0x1605	CUS_0xd40	April	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	
4	0x1606	CUS_0xd40	May	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	

5 rows × 28 columns

In [ ]:

```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 28 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   ID                        100000 non-null  object 
 1   Customer_ID               100000 non-null  object 
 2   Month                     100000 non-null  object 
 3   Name                      90015 non-null   object 
 4   Age                       100000 non-null  object 
 5   SSN                       100000 non-null  object 
 6   Occupation                100000 non-null  object 
 7   Annual_Income             100000 non-null  object 
 8   Monthly_Inhand_Salary     84998 non-null   float64
 9   Num_Bank_Accounts         100000 non-null  int64  
 10  Num_Credit_Card           100000 non-null  int64  
 11  Interest_Rate             100000 non-null  int64  
 12  Num_of_Loan               100000 non-null  object 
 13  Type_of_Loan              88592 non-null   object 
 14  Delay_from_due_date       100000 non-null  int64  
 15  Num_of_Delayed_Payment    92998 non-null   object 
 16  Changed_Credit_Limit      100000 non-null  object 
 17  Num_Credit_Inquiries      98035 non-null   float64
 18  Credit_Mix                100000 non-null  object 
 19  Outstanding_Debt          100000 non-null  object 
 20  Credit_Utilization_Ratio  100000 non-null  float64
 21  Credit_History_Age        90970 non-null   object 
 22  Payment_of_Min_Amount     100000 non-null  object 
 23  Total_EMI_per_month       100000 non-null  float64
 24  Amount_invested_monthly   95521 non-null   object 
 25  Payment_Behaviour         100000 non-null  object 
 26  Monthly_Balance           98800 non-null   object 
 27  Credit_Score              100000 non-null  object 
dtypes: float64(4), int64(4), object(20)
memory usage: 21.4+ MB
```

In [ ]:

```
df_train.describe(include='object').T
```

Out[8]:

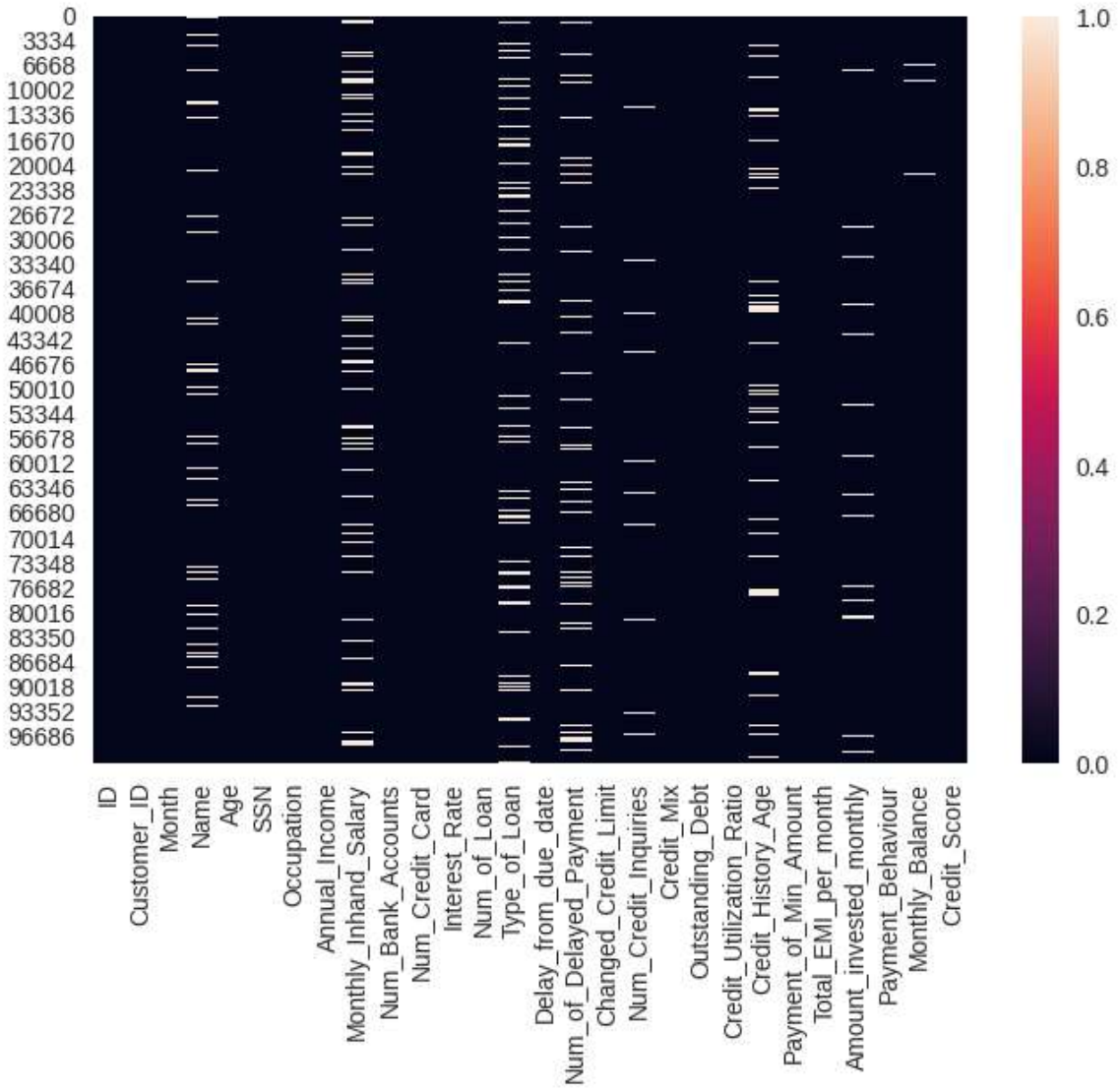
	count	unique	top	freq
ID	100000	100000	0x1602	1
Customer_ID	100000	12500	CUS_0xd40	8
Month	100000	8	January	12500
Name	90015	10139	Langep	44
Age	100000	1788	38	2833
SSN	100000	12501	#F%$D@*&8	5572
Occupation	100000	16	_______	7062
Annual_Income	100000	18940	36585.12	16
Num_of_Loan	100000	434	3	14386
Type_of_Loan	88592	6260	Not Specified	1408
Num_of_Delayed_Payment	92998	749	19	5327
Changed_Credit_Limit	100000	4384	_	2091
Credit_Mix	100000	4	Standard	36479
Outstanding_Debt	100000	13178	1360.45	24
Credit_History_Age	90970	404	15 Years and 11 Months	446
Payment_of_Min_Amount	100000	3	Yes	52326
Amount_invested_monthly	95521	91049	__10000__	4305
Payment_Behaviour	100000	7	Low_spent_Small_value_payments	25513
Monthly_Balance	98800	98792	__-333333333333333333333333333__	9
Credit_Score	100000	3	Standard	53174

In [ ]:

```
sns.heatmap(df_train.isnull())
```

Out[9]:

<Axes: >



We have wrong values and missing values -- We will detect the wrong values and process these values.

Preprocessing - Missing values

Handling Missing and Wrong Values - and outliers

[03] Name

- missing :
  - NaN Values

In [ ]:

```
print(df_train['Name'].value_counts())

print('#'*15)

print('Missing = ',df_train['Name'].isnull().sum())
```

```
Langep            44
Stevex            44
Vaughanl          39
Jessicad          39
Raymondr          38
                  ..
Alina Selyukhg     4
Habboushg          4
Mortimerq          4
Ronaldf            4
Timothyl           3
Name: Name, Length: 10139, dtype: int64
###############
Missing =  9985
```

In [ ]:

```
df_train[df_train['Name'].isnull()]
```

Out[11]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Inc
7	0x1609	CUS_0xd40	August	NaN	23	#F%$D@*&8	Scientist	1911
17	0x161b	CUS_0x2dbc	February	NaN	34	486-85-3974	Engineer	14316
22	0x1620	CUS_0x2dbc	July	NaN	34	486-85-3974	Engineer	14316
64	0x1662	CUS_0x4157	January	NaN	23	070-19-1622	Doctor	11483
80	0x167a	CUS_0xa66b	January	NaN	40	221-30-8554	Teacher	3375
...	...	...	...	...	...	...	...	
99964	0x25fba	CUS_0x372c	May	NaN	18	340-85-7301	Lawyer	4290
99965	0x25fbb	CUS_0x372c	June	NaN	19	340-85-7301	Lawyer	4290
99969	0x25fc3	CUS_0xf16	February	NaN	45	868-70-2218	Media_Manager	1668
99973	0x25fc7	CUS_0xf16	June	NaN	45	868-70-2218	Media_Manager	1668
99986	0x25fdc	CUS_0x8600	March	NaN	28	031-35-0942	Architect	2000

9985 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x8600']['Name']
```

Out[12]:

```
99984    Sarah McBridec
99985    Sarah McBridec
99986               NaN
99987    Sarah McBridec
99988    Sarah McBridec
99989    Sarah McBridec
99990    Sarah McBridec
99991    Sarah McBridec
Name: Name, dtype: object
```

In [ ]:

```
# get customer id for nan values in Name Column
Customer_IDs = df_train[df_train['Name'].isnull()]['Customer_ID'].values

# fill missing values
for id in Customer_IDs:
    # get real name by customer id
    realName = ''
    realName = df_train.loc[(df_train['Customer_ID'] == id) & (df_train['Name'].notna())]
    # fill missing value
    df_train.loc[(df_train['Customer_ID'] == id) & (df_train['Name'].isna()), ['Name']] =
```

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x8600']['Name']
```

Out[14]:

```
99984    Sarah McBridec
99985    Sarah McBridec
99986    Sarah McBridec
99987    Sarah McBridec
99988    Sarah McBridec
99989    Sarah McBridec
99990    Sarah McBridec
99991    Sarah McBridec
Name: Name, dtype: object
```

In [ ]:

```
df_train[df_train['Name'].isnull()] # Check
```

Out[15]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[04] Age

- Missing :

- values Such as (_28)
- values < 0
- values > 60

In [ ]:

```
# We are have values = 28_
Remove_value(df_train, 'Age')

# Convert to int
ConvertDataType(df_train, 'Age',int)
```

In [ ]:

```
df_train[(df_train['Age'] < 0) | (df_train['Age'] > 60)]
```

Out[18]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Incom
2	0x1604	CUS_0xd40	March	Aaron Maashoh	-500	821-00-0265	Scientist	19114.1
56	0x1656	CUS_0x5407	January	Annk	7580	500-92-6408	Media_Manager	34081.38
113	0x16ab	CUS_0xff4	February	Poornimaf	-500	655-05-7666	Entrepreneur	25546.2
122	0x16b8	CUS_0x33d2	March	Chalmersa	181	965-46-2491	Scientist	31993.7
219	0x1749	CUS_0x3edc	April	Williamso	995	663-16-3845	Accountant	43070.2
...	...	...	...	...	...	...	...	.
99913	0x25f6f	CUS_0x1619	February	Phil Wahbao	2263	683-59-7399	Media_Manager	20059.9
99937	0x25f93	CUS_0xad4f	February	Sabina Zawadzkig	-500	226-45-0652	_____	22620.7
99950	0x25fa4	CUS_0x51b3	July	Ryana	1342	837-85-9800	Media_Manager	59146.3
99963	0x25fb9	CUS_0x372c	April	Lucia Mutikanik	-500	340-85-7301	Lawyer	42903.7
99972	0x25fc6	CUS_0xf16	May	Maria Sheahanb	1753	868-70-2218	Media_Manager	16680.3

2781 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x33d2']['Age']
```

Out[19]:

```
120     30
121     30
122    181
123     30
124     30
125     31
126     31
127     31
Name: Age, dtype: int64
```

In [ ]:

```
fig = px.box(df_train, y="Age")
fig.show()
```

In [ ]:

```
Customer_IDs= df_train[(df_train['Age'] < 0) | (df_train['Age'] > 60)]['Customer_ID'].val

for id in Customer_IDs:
    realAge = df_train[df_train['Customer_ID'] ==id]['Age'].drop_duplicates().values[0]
    if realAge < 0 or realAge > 60:
        realAge = df_train[df_train['Customer_ID'] ==id]['Age'].drop_duplicates().values[

    df_train.loc[(df_train['Customer_ID'] == id) & (
        (df_train['Age'] < 0) | (df_train['Age'] > 60)), ['Age']] = realAge
```

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x33d2']['Age']
```

Out[22]:

```
120    30
121    30
122    30
123    30
124    30
125    31
126    31
127    31
Name: Age, dtype: int64
```

In [ ]:

```
fig = px.box(df_train, y="Age")
fig.show()
```

[05] SSN

- missing :
  - values = #F%$D@*&8

In [ ]:

```
df_train['SSN'].value_counts()
```

Out[24]:

```
#F%$D@*&8      5572
078-73-5990       8
486-78-3816       8
750-67-7525       8
903-50-0305       8
                ...
856-06-6147       4
753-72-2651       4
331-28-1921       4
604-62-6133       4
286-44-9634       4
Name: SSN, Length: 12501, dtype: int64
```

In [ ]:

```
df_train[df_train['SSN']=='#F%$D@*&8']
```

Out[25]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_
7	0x1609	CUS_0xd40	August	Aaron Maashoh	23	#F%$D@*&8	Scientist	1
29	0x162b	CUS_0xb891	June	Jasond	55	#F%$D@*&8	_____	3
51	0x164d	CUS_0x284a	April	Nadiaq	34	#F%$D@*&8	Lawyer	1
54	0x1650	CUS_0x284a	July	Nadiaq	34	#F%$D@*&8	Lawyer	109
98	0x1694	CUS_0x3e45	March	Harriet McLeodd	35	#F%$D@*&8	Entrepreneur	54
...	...	...	...	...	...	...	...	
99914	0x25f70	CUS_0x1619	March	Phil Wahbao	54	#F%$D@*&8	Media_Manager	2
99942	0x25f98	CUS_0xad4f	July	Sabina Zawadzkig	48	#F%$D@*&8	Developer	2
99946	0x25fa0	CUS_0x51b3	March	Ryana	33	#F%$D@*&8	Media_Manager	5
99968	0x25fc2	CUS_0xf16	January	Maria Sheahanb	44	#F%$D@*&8	Media_Manager	1
99988	0x25fde	CUS_0x8600	May	Sarah McBridec	28	#F%$D@*&8	Architect	2

5572 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x51b3']['SSN']
```

Out[26]:

```
99944    837-85-9800
99945    837-85-9800
99946      #F%$D@*&8
99947    837-85-9800
99948    837-85-9800
99949    837-85-9800
99950    837-85-9800
99951    837-85-9800
Name: SSN, dtype: object
```

In [ ]:

```
Customer_IDs = df_train[df_train['SSN']== '#F%$D@*&8']['Customer_ID'].values
Customer_IDs

for id in Customer_IDs:
    realSSN = df_train[df_train['Customer_ID'] ==id]['SSN'].drop_duplicates().values[0]
    if realSSN == '#F%$D@*&8':
        realSSN = df_train[df_train['Customer_ID'] ==id]['SSN'].drop_duplicates().values[

    df_train.loc[(df_train['Customer_ID'] == id) & (
        df_train['SSN'] == '#F%$D@*&8'), ['SSN']] = realSSN
```

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x51b3']['SSN']
```

Out[28]:

```
99944    837-85-9800
99945    837-85-9800
99946    837-85-9800
99947    837-85-9800
99948    837-85-9800
99949    837-85-9800
99950    837-85-9800
99951    837-85-9800
Name: SSN, dtype: object
```

In [ ]:

```
df_train[df_train['SSN'] == '#F%$D@*&8'] # Check
```

Out[29]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[06] Occupation

- Missing :

- values = ___

In [ ]:

```
print(df_train['Occupation'].value_counts())
print('#'*15)
print('Missing = ',df_train['Occupation'].isnull().sum())
```

```
_______           7062
Lawyer            6575
Architect         6355
Engineer          6350
Scientist         6299
Mechanic          6291
Accountant        6271
Developer         6235
Media_Manager     6232
Teacher           6215
Entrepreneur      6174
Doctor            6087
Journalist        6085
Manager           5973
Musician          5911
Writer            5885
Name: Occupation, dtype: int64
###############
Missing =  0
```

In [ ]:

```
df_train[df_train['Occupation'] == '_______']
```

Out[32]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	M
8	0x160e	CUS_0x21b1	January	Rick Rothackerj	28	004-07-5839	_______	34847.84	
16	0x161a	CUS_0x2dbc	January	Langep	34	486-85-3974	_______	143162.64	
18	0x161c	CUS_0x2dbc	March	Langep	34	486-85-3974	_______	143162.64	
20	0x161e	CUS_0x2dbc	May	Langep	34	486-85-3974	_______	143162.64	
29	0x162b	CUS_0xb891	June	Jasond	55	072-31-6145	_______	30689.89	
...	...	...	...	...	...	...	...	...	
99920	0x25f7a	CUS_0x2654	January	enj	37	647-67-8889	_______	139664.96	
99935	0x25f8d	CUS_0xb11c	August	Yinka Adegokej	38	546-94-4789	_______	15319.65	
99937	0x25f93	CUS_0xad4f	February	Sabina Zawadzkig	47	226-45-0652	_______	22620.79	
99943	0x25f99	CUS_0xad4f	August	Sabina Zawadzkig	48	226-45-0652	_______	22620.79	
99989	0x25fdf	CUS_0x8600	June	Sarah McBridec	28	031-35-0942	_______	20002.88	

7062 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0xad4f']['Occupation']
```

Out[33]:

```
99936    Developer
99937      _______
99938    Developer
99939    Developer
99940    Developer
99941    Developer
99942    Developer
99943      _______
Name: Occupation, dtype: object
```

In [ ]:

```
Customer_IDs = df_train[df_train['Occupation']== '_______']['Customer_ID'].values
Customer_IDs

for id in Customer_IDs:
    realOcc = df_train[df_train['Customer_ID'] ==id]['Occupation'].drop_duplicates().valu
    if realOcc == '_______':
        realOcc = df_train[df_train['Customer_ID'] == id]['Occupation'].drop_duplicates()

    df_train.loc[(df_train['Customer_ID'] == id) & (
        df_train['Occupation'] == '_______'), ['Occupation']] = realOcc
```

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0xad4f']['Occupation']
```

Out[35]:

```
99936    Developer
99937    Developer
99938    Developer
99939    Developer
99940    Developer
99941    Developer
99942    Developer
99943    Developer
Name: Occupation, dtype: object
```

In [ ]:

```
df_train[df_train['Occupation'] == '_______']
```

Out[36]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[07] Annual Income

- missing :
  - values such as 280.60_
  - wrong values
- outliers

In [ ]:

```
print(df_train['Annual_Income'].value_counts())
print('#'*15)
print('Missing = ',df_train['Annual_Income'].isnull().sum())
```

```
36585.12     16
20867.67     16
17273.83     16
9141.63      15
33029.66     15
             ..
20269.93_     1
15157.25_     1
44955.64_     1
76650.12_     1
4262933.0     1
Name: Annual_Income, Length: 18940, dtype: int64
###############
Missing =  0
```

In [ ]:

```
# We are have values = 280.60_
Remove_value(df_train, 'Annual_Income')

# convert to float
ConvertDataType(df_train, 'Annual_Income', float)
```

In [ ]:

```
fig = px.box(df_train, y="Annual_Income")
fig.show()
```

In [ ]:

```
df_train[df_train['Annual_Income'] > 153000]
```

Out[40]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_I
54	0x1650	CUS_0x284a	July	Nadiaq	34	411-51-0676	Lawyer	1090
231	0x1759	CUS_0xbffe	August	Dhanya Skariachano	40	311-13-7309	Architect	651
245	0x176f	CUS_0x9a71	June	Mukhopadhyayc	55	889-07-2357	Scientist	58
361	0x181f	CUS_0x8e9b	February	Rachelle Younglaic	15	925-51-5335	Entrepreneur	1833
368	0x182a	CUS_0x609d	January	Shupingu	27	911-47-6879	Architect	1971
...	...	...	...	...	...	...	...	
99664	0x25dfa	CUS_0xb09	January	Lianau	31	228-47-4867	Lawyer	579
99714	0x25e44	CUS_0xadbd	March	Scotts	26	864-24-3672	Doctor	1171
99721	0x25e4f	CUS_0x11c7	February	raden Reddallh	53	646-19-1493	Architect	854
99882	0x25f40	CUS_0x47fa	March	Yantoultra Nguif	31	291-51-7240	Mechanic	1688
99945	0x25f9f	CUS_0x51b3	February	Ryana	33	837-85-9800	Media_Manager	1015

2759 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x284a']['Annual_Income']
```

Out[41]:

```
48      131313.4
49      131313.4
50      131313.4
51      131313.4
52      131313.4
53      131313.4
54    10909427.0
55      131313.4
Name: Annual_Income, dtype: float64
```

In [ ]:

```
Customer_IDs = df_train[df_train['Annual_Income']> 153000]['Customer_ID'].values


for id in Customer_IDs:
    realIncome = df_train[df_train['Customer_ID'] ==id]['Annual_Income'].drop_duplicates(
    # if realIncome > 153000 :
    #     realIncome = df_train[df_train['Customer_ID'] == id]['Annual_Income'].drop_dupl

    df_train.loc[(df_train['Customer_ID'] == id) & (
        df_train['Annual_Income'] > 153000), ['Annual_Income']] = realIncome
```

In [ ]:

```
fig = px.box(df_train, y="Annual_Income")
fig.show()
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x2c31']['Annual_Income']
```

Out[44]:

```
1728    173196.32
1729    173196.32
1730    173196.32
1731    173196.32
1732    173196.32
1733    173196.32
1734    173196.32
1735    173196.32
Name: Annual_Income, dtype: float64
```

[08] Monthly Inhand Salary

In [ ]:

```
print(df_train['Monthly_Inhand_Salary'].value_counts())
print("#"*20)
print('Missing = ',df_train['Monthly_Inhand_Salary'].isnull().sum())
```

```
6769.130000    15
6358.956667    15
2295.058333    15
6082.187500    15
3080.555000    14
               ..
1087.546445     1
3189.212103     1
5640.117744     1
7727.560450     1
2443.654131     1
Name: Monthly_Inhand_Salary, Length: 13235, dtype: int64
####################
Missing =  15002
```

In [ ]:

```
# Convert nan to 0.0
df_train['Monthly_Inhand_Salary'].replace(np.nan, 0.0, inplace=True)
df_train[df_train['Monthly_Inhand_Salary']==0.0]
```

Out[46]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Incom
1	0x1603	CUS_0xd40	February	Aaron Maashoh	23	821-00-0265	Scientist	19114.1
2	0x1604	CUS_0xd40	March	Aaron Maashoh	23	821-00-0265	Scientist	19114.1
3	0x1605	CUS_0xd40	April	Aaron Maashoh	23	821-00-0265	Scientist	19114.1
5	0x1607	CUS_0xd40	June	Aaron Maashoh	23	821-00-0265	Scientist	19114.1
11	0x1611	CUS_0x21b1	April	Rick Rothackerj	28	004-07-5839	Teacher	34847.8
...	...	...	...	...	...	...	...	..
99944	0x25f9e	CUS_0x51b3	January	Ryana	33	837-85-9800	Media_Manager	59146.3
99955	0x25fad	CUS_0x2084	April	Ryanl	21	253-72-7758	Architect	38321.3
99963	0x25fb9	CUS_0x372c	April	Lucia Mutikanik	18	340-85-7301	Lawyer	42903.7
99975	0x25fc9	CUS_0xf16	August	Maria Sheahanb	45	868-70-2218	Media_Manager	16680.3
99978	0x25fd0	CUS_0xaf61	March	Chris Wickhamm	49	133-16-7738	Writer	37188.1

15002 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0xd40']['Monthly_Inhand_Salary']
```

Out[47]:

```
0    1824.843333
1       0.000000
2       0.000000
3       0.000000
4    1824.843333
5       0.000000
6    1824.843333
7    1824.843333
Name: Monthly_Inhand_Salary, dtype: float64
```

In [ ]:

```
Customer_IDs = df_train[df_train['Monthly_Inhand_Salary']==0.0]['Customer_ID'].values

for id in Customer_IDs :
      realMis=df_train[df_train['Customer_ID'] ==
                  id]['Monthly_Inhand_Salary'].drop_duplicates().sort_values().values[-1]


      df_train.loc[(df_train['Customer_ID'] == id) & (df_train['Monthly_Inhand_Salary']==
                  , ['Monthly_Inhand_Salary']] = realMis
```

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0xd40']['Monthly_Inhand_Salary']
```

Out[49]:

```
0    1824.843333
1    1824.843333
2    1824.843333
3    1824.843333
4    1824.843333
5    1824.843333
6    1824.843333
7    1824.843333
Name: Monthly_Inhand_Salary, dtype: float64
```

In [ ]:

```
df_train[df_train['Monthly_Inhand_Salary'] == 0.0]
```

Out[50]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[09] Num Bank Accounts

In [ ]:

```
print(df_train['Num_Bank_Accounts'].value_counts())
print("#"*20)
print('Missing = ',df_train['Num_Bank_Accounts'].isnull().sum())
```

```
6       13001
7       12823
8       12765
4       12186
5       12118
        ...
1626        1
1470        1
887         1
211         1
697         1
Name: Num_Bank_Accounts, Length: 943, dtype: int64
####################
Missing =  0
```

In [ ]:

```
px.box(df_train,y='Num_Bank_Accounts')
```

In [ ]:

```
df_train[(df_train['Num_Bank_Accounts'] > 11) |
        (df_train['Num_Bank_Accounts'] < 0)]
```

Out[53]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Incom
267	0x1791	CUS_0x4004	April	Carlosj	44	679-26-6464	Writer	58317.0
288	0x17b2	CUS_0x4080	January	ra Alperx	16	995-37-8920	Mechanic	29469.9
310	0x17d0	CUS_0x42ac	July	Lawrencea	37	700-60-3660	Musician	15566.0
339	0x17fd	CUS_0x9bc1	April	Jaisinghanij	42	445-18-4420	Architect	20574.4
356	0x1816	CUS_0xaedb	May	Olivia Oranr	19	272-47-1135	Musician	85554.0
...	...	...	...	...	...	...	...	.
99591	0x25d89	CUS_0x544	August	Jon Herskovitzu	29	163-45-1172	Mechanic	17013.2
99638	0x25dd0	CUS_0x296f	July	David Millikenh	25	972-13-0672	Developer	125271.9
99666	0x25dfc	CUS_0xb09	March	Lianau	31	228-47-4867	Lawyer	146310.6
99722	0x25e50	CUS_0x11c7	March	raden Reddallh	53	646-19-1493	Architect	36817.9
99916	0x25f72	CUS_0x1619	May	Phil Wahbao	54	683-59-7399	Media_Manager	20059.9

1336 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x1619']['Num_Bank_Accounts']
```

Out[54]:

```
99912      8
99913      8
99914      8
99915      8
99916    182
99917      8
99918      8
99919      8
Name: Num_Bank_Accounts, dtype: int64
```

In [ ]:

```
Customer_IDs=df_train[(df_train['Num_Bank_Accounts'] > 11) |(df_train['Num_Bank_Accounts'

for id in Customer_IDs:
    realNum = df_train[df_train['Customer_ID'] ==id]['Num_Bank_Accounts'].drop_duplicates
    if realNum < 0 or realNum > 11:
        realNum = df_train[df_train['Customer_ID'] ==id]['Num_Bank_Accounts'].drop_duplic

    df_train.loc[(df_train['Customer_ID'] == id) & (
        (df_train['Num_Bank_Accounts'] < 0) | (df_train['Num_Bank_Accounts'] > 11)), ['Nu
```

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x1619']['Num_Bank_Accounts']
```

Out[56]:

```
99912    8
99913    8
99914    8
99915    8
99916    8
99917    8
99918    8
99919    8
Name: Num_Bank_Accounts, dtype: int64
```

In [ ]:

```
px.box(df_train, y='Num_Bank_Accounts')
```

[12] Num of Loan

In [ ]:

```
print(df_train['Num_of_Loan'].value_counts())
print("#"*20)
print('Missing = ',df_train['Num_of_Loan'].isnull().sum())
```

```
3    15752
2    15712
4    15456
0    11408
1    11128
6     8144
7     7680
5     7528
9     3856
8     3336
Name: Num_of_Loan, dtype: int64
####################
Missing =  0
```

In [ ]:

```
# We are have values = 23058_
Remove_value(df_train, 'Num_of_Loan')
# convert to int
ConvertDataType(df_train, 'Num_of_Loan',int)
```

In [ ]:

```
px.box(df_train,y='Num_of_Loan')
```

In [ ]:

```
df_train[(df_train['Num_of_Loan'] > 9) |(df_train['Num_of_Loan'] <= 0)].T
```

Out[75]:

	32	33	
ID	0x1632	0x1633	
Customer_ID	CUS_0x1cdb	CUS_0x1cdb	
Month	January	February	
Name	Deepaa	Deepaa	
Age	21	21	
SSN	615-06-7821	615-06-7821	
Occupation	Developer	Developer	
Annual_Income	35547.71	35547.71	
Monthly_Inhand_Salary	2853.309167	2853.309167	
Num_Bank_Accounts	7	7	
Num_Credit_Card	5	5	
Interest_Rate	5	5	
Num_of_Loan	0	0	
Type_of_Loan	NaN	NaN	
Delay_from_due_date	5	9	
Num_of_Delayed_Payment	NaN	NaN	
Changed_Credit_Limit	2.58	2.58	
Num_Credit_Inquiries	4.0	4.0	
Credit_Mix	Standard	Standard	
Outstanding_Debt	943.86	943.86	
Credit_Utilization_Ratio	39.797764	27.02036	
Credit_History_Age	30 Years and 8 Months	30 Years and 9 Months	
Payment_of_Min_Amount	Yes	NM	
Total_EMI_per_month	0.0	0.0	
Amount_invested_monthly	276.72539431736266	74.44364104999623	
Payment_Behaviour	!@9#%8	High_spent_Medium_value_payments	Low_spent_
Monthly_Balance	288.60552234930395	460.88727561667037	
Credit_Score	Standard	Standard	

28 rows × 11408 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x3048']['Num_of_Loan']
```

Out[76]:

```
99848    7
99849    7
99850    7
99851    7
99852    7
99853    7
99854    7
99855    7
Name: Num_of_Loan, dtype: int64
```

In [ ]:

```
Customer_IDs = df_train[(df_train['Num_of_Loan'] > 9) | (
    df_train['Num_of_Loan'] <= 0)]['Customer_ID'].values

for id in Customer_IDs:
    realNum = df_train[df_train['Customer_ID'] ==
                       id]['Num_of_Loan'].drop_duplicates().values[0]
    if realNum < 0 or realNum > 9:
        realNum = df_train[df_train['Customer_ID'] ==
                           id]['Num_of_Loan'].drop_duplicates().values[1]

    df_train.loc[(df_train['Customer_ID'] == id) & (
        (df_train['Num_of_Loan'] < 0) | (df_train['Num_of_Loan'] > 9)), ['Num_of_Loan']]
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x3048']['Num_of_Loan']
```

Out[78]:

```
99848    7
99849    7
99850    7
99851    7
99852    7
99853    7
99854    7
99855    7
Name: Num_of_Loan, dtype: int64
```

In [ ]:

```
px.box(df_train,y='Num_of_Loan')
```

[13] Type of Loan

In [ ]:

```
print(df_train['Type_of_Loan'].value_counts())
print("#"*20)
print('Missing = ',df_train['Type_of_Loan'].isnull().sum())
```

```
Not Specified
1408
Credit-Builder Loan
1280
Personal Loan
1272
Debt Consolidation Loan
1264
Student Loan
1240

...
Not Specified, Mortgage Loan, Auto Loan, and Payday Loan
8
Payday Loan, Mortgage Loan, Debt Consolidation Loan, and Student Loan
8
Debt Consolidation Loan, Auto Loan, Personal Loan, Debt Consolidation Loa
n, Student Loan, and Credit-Builder Loan                          8
Student Loan, Auto Loan, Student Loan, Credit-Builder Loan, Home Equity Lo
an, Debt Consolidation Loan, and Debt Consolidation Loan       8
Personal Loan, Auto Loan, Mortgage Loan, Student Loan, and Student Loan
8
Name: Type_of_Loan, Length: 6260, dtype: int64
####################
Missing =  11408
```

In [ ]:

```
df_train[df_train['Type_of_Loan'].isna()]
```

Out[81]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	M
32	0x1632	CUS_0x1cdb	January	Deepaa	21	615-06-7821	Developer	35547.71	
33	0x1633	CUS_0x1cdb	February	Deepaa	21	615-06-7821	Developer	35547.71	
34	0x1634	CUS_0x1cdb	March	Deepaa	21	615-06-7821	Developer	35547.71	
35	0x1635	CUS_0x1cdb	April	Deepaa	21	615-06-7821	Developer	35547.71	
36	0x1636	CUS_0x1cdb	May	Deepaa	21	615-06-7821	Developer	35547.71	
...	...	...	...	...	...	...	...	...	
99939	0x25f95	CUS_0xad4f	April	Sabina Zawadzkig	47	226-45-0652	Developer	22620.79	
99940	0x25f96	CUS_0xad4f	May	Sabina Zawadzkig	47	226-45-0652	Developer	22620.79	
99941	0x25f97	CUS_0xad4f	June	Sabina Zawadzkig	47	226-45-0652	Developer	22620.79	
99942	0x25f98	CUS_0xad4f	July	Sabina Zawadzkig	48	226-45-0652	Developer	22620.79	
99943	0x25f99	CUS_0xad4f	August	Sabina Zawadzkig	48	226-45-0652	Developer	22620.79	

11408 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x1cdb']['Type_of_Loan']
```

Out[82]:

```
32    NaN
33    NaN
34    NaN
35    NaN
36    NaN
37    NaN
38    NaN
39    NaN
Name: Type_of_Loan, dtype: object
```

In [ ]:

```
df_train[df_train['Num_of_Loan']==1]['Type_of_Loan'].value_counts()
```

Out[85]:

```
Not Specified              1408
Credit-Builder Loan        1280
Personal Loan              1272
Debt Consolidation Loan    1264
Student Loan               1240
Payday Loan                1200
Mortgage Loan              1176
Auto Loan                  1152
Home Equity Loan           1136
Name: Type_of_Loan, dtype: int64
```

In [ ]:

```
# df_train['Type_of_Loan'].fillna(value=np.random.choice(df_train['Type_of_Loan'].dropna(

Customer_IDs = df_train[df_train['Type_of_Loan'].isna()]['Customer_ID']

# each itr choice random value to fill missing values
for id in Customer_IDs:
    df_train.loc[(df_train['Customer_ID'] == id) & (
        (df_train['Type_of_Loan'].isna())), ['Type_of_Loan']] = np.random.choice(df_trair
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x1cdb']['Type_of_Loan']
```

Out[87]:

```
32    Personal Loan, Payday Loan, Credit-Builder Loa...
33    Personal Loan, Payday Loan, Credit-Builder Loa...
34    Personal Loan, Payday Loan, Credit-Builder Loa...
35    Personal Loan, Payday Loan, Credit-Builder Loa...
36    Personal Loan, Payday Loan, Credit-Builder Loa...
37    Personal Loan, Payday Loan, Credit-Builder Loa...
38    Personal Loan, Payday Loan, Credit-Builder Loa...
39    Personal Loan, Payday Loan, Credit-Builder Loa...
Name: Type_of_Loan, dtype: object
```

In [ ]:

```
df_train[df_train['Type_of_Loan'].isna()]
```

Out[88]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[14] Delay from due date

In [ ]:

```
print(df_train['Delay_from_due_date'].value_counts())
print("#"*20)
print('Missing = ',df_train['Delay_from_due_date'].isnull().sum())
```

```
 15    3596
 13    3424
 8     3324
 14    3313
 10    3281
       ...
-4       62
 65      56
-5       33
 66      32
 67      22
Name: Delay_from_due_date, Length: 73, dtype: int64
####################
Missing =  0
```

In [ ]:

```
df_train[df_train['Delay_from_due_date'] < 0]
```

Out[90]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Incom
1	0x1603	CUS_0xd40	February	Aaron Maashoh	23	821-00-0265	Scientist	19114.1
49	0x164b	CUS_0x284a	February	Nadiaq	34	411-51-0676	Lawyer	131313.4
74	0x1670	CUS_0xba08	March	Jamesj	44	366-68-1681	Journalist	31370.8
78	0x1674	CUS_0xba08	July	Jamesj	45	366-68-1681	Journalist	31370.8
79	0x1675	CUS_0xba08	August	Jamesj	45	366-68-1681	Journalist	31370.8
...	...	...	...	...	...	...	...	.
99198	0x25b3c	CUS_0xb59b	July	Luciax	35	447-00-0042	Mechanic	96047.2
99358	0x25c2c	CUS_0x870c	July	Kaustubhn	44	310-73-7596	Manager	19631.2
99368	0x25c3e	CUS_0x163c	January	Julien Toyera	31	534-12-7952	Media_Manager	61465.7
99371	0x25c41	CUS_0x163c	April	Julien Toyera	31	534-12-7952	Media_Manager	61465.7
99667	0x25dfd	CUS_0xb09	April	Lianau	31	228-47-4867	Lawyer	146310.6

591 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x163c']['Delay_from_due_date']
```

Out[91]:

```
99368   -4
99369    0
99370    0
99371   -5
99372    0
99373    0
99374    0
99375    0
Name: Delay_from_due_date, dtype: int64
```

In [ ]:

```
Customer_IDs = df_train[df_train['Delay_from_due_date']< 0]['Customer_ID'].values

for id in Customer_IDs:
    realvalue = df_train[df_train['Customer_ID'] ==id]['Delay_from_due_date'].drop_duplic
    if realvalue < 0:
        realvalue = df_train[df_train['Customer_ID'] ==id]['Delay_from_due_date'].drop_du

    df_train.loc[(df_train['Customer_ID'] == id) & (df_train['Delay_from_due_date'] < 0),
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x163c']['Delay_from_due_date']
```

Out[93]:

```
99368    0
99369    0
99370    0
99371    0
99372    0
99373    0
99374    0
99375    0
Name: Delay_from_due_date, dtype: int64
```

In [ ]:

```
df_train[df_train['Delay_from_due_date'] < 0]
```

Out[137]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[15] Num of Delayed Payment

- NaN
- 8_
- < 0

In [ ]:

```
print(df_train['Num_of_Delayed_Payment'].value_counts())
print("#"*20)
print('Missing = ',df_train['Num_of_Delayed_Payment'].isnull().sum())
```

```
19      5327
17      5261
16      5173
10      5153
18      5083
        ...
848_       1
4134       1
1530       1
1502       1
2047       1
Name: Num_of_Delayed_Payment, Length: 749, dtype: int64
####################
Missing =  7002
```

In [ ]:

```
df_train[df_train['Num_of_Delayed_Payment'].isna()]
```

Out[96]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income
1	0x1603	CUS_0xd40	February	Aaron Maashoh	23	821-00-0265	Scientist	19114.12
4	0x1606	CUS_0xd40	May	Aaron Maashoh	23	821-00-0265	Scientist	19114.12
30	0x162c	CUS_0xb891	July	Jasond	55	072-31-6145	Entrepreneur	30689.89
32	0x1632	CUS_0x1cdb	January	Deepaa	21	615-06-7821	Developer	35547.71
33	0x1633	CUS_0x1cdb	February	Deepaa	21	615-06-7821	Developer	35547.71
...	...	...	...	...	...	...	...	...
99973	0x25fc7	CUS_0xf16	June	Maria Sheahanb	45	868-70-2218	Media_Manager	16680.35
99974	0x25fc8	CUS_0xf16	July	Maria Sheahanb	45	868-70-2218	Media_Manager	16680.35
99992	0x25fe6	CUS_0x942c	January	Nicks	24	078-73-5990	Mechanic	39628.99
99993	0x25fe7	CUS_0x942c	February	Nicks	25	078-73-5990	Mechanic	39628.99
99998	0x25fec	CUS_0x942c	July	Nicks	25	078-73-5990	Mechanic	39628.99

7002 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x942c']['Num_of_Delayed_Payment']
```

Out[97]:

```
99992     NaN
99993     NaN
99994       6
99995       7
99996       7
99997       6
99998     NaN
99999       6
Name: Num_of_Delayed_Payment, dtype: object
```

In [ ]:

```
#  [1] fill nan

Customer_IDs = df_train[df_train['Num_of_Delayed_Payment'].isna()]['Customer_ID'].values
for id in Customer_IDs:

    mode_v = df_train[df_train['Customer_ID'] == id]['Num_of_Delayed_Payment'].mode()[
        0]

    df_train.loc[(df_train['Customer_ID'] == id) & (
        df_train['Num_of_Delayed_Payment'].isna()), ['Num_of_Delayed_Payment']] = mode_v
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x942c']['Num_of_Delayed_Payment']
```

Out[99]:

```
99992    6
99993    6
99994    6
99995    7
99996    7
99997    6
99998    6
99999    6
Name: Num_of_Delayed_Payment, dtype: object
```

In [ ]:

```
# check
df_train[df_train['Num_of_Delayed_Payment'].isna()]
```

Out[100]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

In [ ]:

```
# we are have values = 8_
Remove_value(df_train, 'Num_of_Delayed_Payment')
```

In [ ]:

```
# convert to int
ConvertDataType(df_train, 'Num_of_Delayed_Payment',int)
```

In [ ]:

```
df_train[df_train['Num_of_Delayed_Payment']<0]
```

Out[103]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income
10	0x1610	CUS_0x21b1	March	Rick Rothackerj	28	004-07-5839	Teacher	34847.840
698	0x1a18	CUS_0x71e0	March	Aruna Viswanathau	28	330-96-0638	Mechanic	15398.950
1253	0x1d57	CUS_0x31d5	June	Mutikanit	49	325-07-8725	Doctor	15488.885
1561	0x1f27	CUS_0xb374	February	enu	31	946-08-9738	Architect	30922.280
2167	0x22b1	CUS_0x280b	August	orv	27	434-17-8443	Accountant	34841.870
...	...	...	...	...	...	...	...	...
99269	0x25ba7	CUS_0xa66c	June	Ernest Scheyderi	24	770-36-8261	Accountant	74349.180
99512	0x25d16	CUS_0x10f9	January	Gilles Guillaumev	53	746-79-6875	Lawyer	150131.680
99515	0x25d19	CUS_0x10f9	April	Gilles Guillaumev	54	746-79-6875	Lawyer	150131.680
99586	0x25d84	CUS_0x544	March	Jon Herskovitzu	28	163-45-1172	Mechanic	17013.290
99965	0x25fbb	CUS_0x372c	June	Lucia Mutikanik	19	340-85-7301	Lawyer	42903.790

653 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x10f9']['Num_of_Delayed_Payment']
```

Out[104]:

```
99512   -2
99513    2
99514    0
99515   -2
99516    0
99517    0
99518    0
99519    0
Name: Num_of_Delayed_Payment, dtype: int64
```

In [ ]:

```
# fill < 0
Customer_IDs = df_train[df_train['Num_of_Delayed_Payment']< 0]['Customer_ID'].values

for i in Customer_IDs:
    df_train.loc[(df_train['Customer_ID'] == i) & (
        df_train['Num_of_Delayed_Payment'] < 0), ['Num_of_Delayed_Payment']] = 0
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x10f9']['Num_of_Delayed_Payment']
```

Out[106]:

```
99512    0
99513    2
99514    0
99515    0
99516    0
99517    0
99518    0
99519    0
Name: Num_of_Delayed_Payment, dtype: int64
```

In [ ]:

```
df_train[df_train['Num_of_Delayed_Payment'] < 0]  # check
```

Out[136]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[16] Changed Credit Limit

- _
- < 0

In [ ]:

```
print(df_train['Changed_Credit_Limit'].value_counts())
print("#"*20)
print('Missing = ',df_train['Changed_Credit_Limit'].isnull().sum())
```

```
_                      2091
8.22                    133
11.5                    127
11.32                   126
7.35                    121
                       ...
-1.84                     1
0.8899999999999999        1
28.06                     1
1.5599999999999996        1
21.17                     1
Name: Changed_Credit_Limit, Length: 4384, dtype: int64
####################
Missing =  0
```

In [ ]:

```
df_train[(df_train['Changed_Credit_Limit'] == '_')]
```

Out[109]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	M
2	0x1604	CUS_0xd40	March	Aaron Maashoh	23	821-00-0265	Scientist	19114.120	
65	0x1663	CUS_0x4157	February	Charlie Zhur	23	070-19-1622	Doctor	114838.410	
66	0x1664	CUS_0x4157	March	Charlie Zhur	23	070-19-1622	Doctor	114838.410	
109	0x16a3	CUS_0x6c66	June	Sinead Carews	39	328-33-6328	Manager	8701.545	
110	0x16a4	CUS_0x6c66	July	Sinead Carews	39	328-33-6328	Manager	8701.545	
...	...	...	...	...	...	...	...	...	
99548	0x25d4a	CUS_0x2637	May	Diane Bartzj	28	272-33-1370	Writer	29090.500	
99618	0x25db4	CUS_0xae66	March	Jino	30	721-45-4479	Writer	142560.360	
99800	0x25ec6	CUS_0x1232	January	Lawderr	16	441-26-1297	Developer	14937.490	
99892	0x25f4e	CUS_0x89aa	May	Kwokw	38	018-63-7005	Manager	85744.120	
99931	0x25f89	CUS_0xb11c	April	Yinka Adegokej	38	546-94-4789	Manager	15319.650	

2091 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='']['Changed_Credit_Limit']
```

Out[110]:

```
Series([], Name: Changed_Credit_Limit, dtype: object)
```

In [ ]:

```
Customer_IDs= df_train[(df_train['Changed_Credit_Limit'] == '_')]['Customer_ID'].values

# _ values
for id in Customer_IDs:

    mode_v = df_train[df_train['Customer_ID'] == id]['Changed_Credit_Limit'].mode()[0]

    df_train.loc[(df_train['Customer_ID'] == id) & (
        df_train['Changed_Credit_Limit'] == '_'), ['Changed_Credit_Limit']] = mode_v
```

In [ ]:

```
df_train[df_train['Customer_ID']=='']['Changed_Credit_Limit']
```

Out[112]:

```
Series([], Name: Changed_Credit_Limit, dtype: object)
```

In [ ]:

```
df_train[df_train['Changed_Credit_Limit']=='_'] #Check
```

Out[113]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

In [ ]:

```
# convert to float
ConvertDataType(df_train,'Changed_Credit_Limit',float)
```

In [ ]:

```
df_train[(df_train['Changed_Credit_Limit'] < 0)]
```

Out[115]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	M
27	0x1629	CUS_0xb891	April	Jasond	55	072-31-6145	Entrepreneur	30689.89	
28	0x162a	CUS_0xb891	May	Jasond	55	072-31-6145	Entrepreneur	30689.89	
29	0x162b	CUS_0xb891	June	Jasond	55	072-31-6145	Entrepreneur	30689.89	
73	0x166f	CUS_0xba08	February	Jamesj	44	366-68-1681	Journalist	31370.80	
179	0x170d	CUS_0xac86	April	Nickb	20	028-16-4402	Entrepreneur	106733.13	
...	...	...	...	...	...	...	...	...	
99787	0x25eb1	CUS_0x62f5	April	Alexein	54	272-40-2510	Musician	99520.50	
99826	0x25eec	CUS_0x2c0a	March	Huww	44	280-99-6832	Doctor	32625.59	
99900	0x25f5a	CUS_0x4986	May	Charles Abbotta	33	971-61-8388	Entrepreneur	41329.56	
99958	0x25fb0	CUS_0x2084	July	Ryanl	21	253-72-7758	Architect	38321.39	
99961	0x25fb7	CUS_0x372c	February	Lucia Mutikanik	18	340-85-7301	Lawyer	42903.79	

1587 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x372c']['Changed_Credit_Limit']
```

Out[116]:

```
99960     4.1
99961    -0.9
99962    12.1
99963     5.1
99964     5.1
99965     5.1
99966     5.1
99967     5.1
Name: Changed_Credit_Limit, dtype: float64
```

In [ ]:

```
Customer_IDs = df_train[(df_train['Changed_Credit_Limit'] < 0)]['Customer_ID'].values

for id in Customer_IDs:

    max_v = df_train[df_train['Customer_ID'] == id]['Changed_Credit_Limit'].max()
    df_train.loc[(df_train['Customer_ID'] == id) & (
        df_train['Changed_Credit_Limit'] < 0), ['Changed_Credit_Limit']] = max_v
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x372c']['Changed_Credit_Limit']
```

Out[118]:

```
99960     4.1
99961    12.1
99962    12.1
99963     5.1
99964     5.1
99965     5.1
99966     5.1
99967     5.1
Name: Changed_Credit_Limit, dtype: float64
```

In [ ]:

```
# check
df_train[df_train['Changed_Credit_Limit'] < 0]
```

Out[119]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[17] Num Credit Inquiries

- Values > 17
- NaN

In [ ]:

```
print(df_train['Num_Credit_Inquiries'].value_counts())
print("#"*20)
print('Missing = ',df_train['Num_Credit_Inquiries'].isnull().sum())
```

```
4.0       11271
3.0        8890
6.0        8111
7.0        8058
2.0        8028
          ...
1721.0        1
1750.0        1
2397.0        1
621.0         1
74.0          1
Name: Num_Credit_Inquiries, Length: 1223, dtype: int64
####################
Missing =  1965
```

In [ ]:

```
px.box(df_train,y='Num_Credit_Inquiries')
```

In [ ]:

```
px.histogram(df_train, x='Num_Credit_Inquiries')
```

In [ ]:

```
Customer_IDs = df_train[(df_train['Num_Credit_Inquiries'].isna()) |
                        (df_train['Num_Credit_Inquiries'] > 17) ]['Customer_ID'].values

for id in Customer_IDs:

    mode_v = df_train[df_train['Customer_ID'] == id]['Num_Credit_Inquiries'].mode()[0]

    df_train.loc[(df_train['Customer_ID'] == id) &
                ((df_train['Num_Credit_Inquiries'].isna()) | (df_train['Num_Credit_Inquir
                ['Num_Credit_Inquiries']]=mode_v
```

In [ ]:

```
px.box(df_train, y='Num_Credit_Inquiries')
```

In [ ]:

```
px.histogram(df_train, x='Num_Credit_Inquiries')
```

[18] Credit Mix

- _ values

In [ ]:

```
print(df_train['Credit_Mix'].value_counts())
print("#"*20)
print('Missing = ',df_train['Credit_Mix'].isnull().sum())
```

```
Standard    36479
Good        24337
_           20195
Bad         18989
Name: Credit_Mix, dtype: int64
####################
Missing =  0
```

In [ ]:

```
df_train[df_train['Credit_Mix'] == '_']
```

Out[127]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	M
0	0x1602	CUS_0xd40	January	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	
10	0x1610	CUS_0x21b1	March	Rick Rothackerj	28	004-07-5839	Teacher	34847.84	
19	0x161d	CUS_0x2dbc	April	Langep	34	486-85-3974	Engineer	143162.64	
29	0x162b	CUS_0xb891	June	Jasond	55	072-31-6145	Entrepreneur	30689.89	
35	0x1635	CUS_0x1cdb	April	Deepaa	21	615-06-7821	Developer	35547.71	
...	...	...	...	...	...	...	...	...	
99988	0x25fde	CUS_0x8600	May	Sarah McBridec	28	031-35-0942	Architect	20002.88	
99992	0x25fe6	CUS_0x942c	January	Nicks	24	078-73-5990	Mechanic	39628.99	
99994	0x25fe8	CUS_0x942c	March	Nicks	25	078-73-5990	Mechanic	39628.99	
99995	0x25fe9	CUS_0x942c	April	Nicks	25	078-73-5990	Mechanic	39628.99	
99996	0x25fea	CUS_0x942c	May	Nicks	25	078-73-5990	Mechanic	39628.99	

20195 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x942c']['Credit_Mix']
```

Out[128]:

```
99992       _
99993    Good
99994       _
99995       _
99996       _
99997    Good
99998    Good
99999    Good
Name: Credit_Mix, dtype: object
```

In [ ]:

```
Customer_IDs = df_train[df_train['Credit_Mix'] == '_']['Customer_ID'].values


for id in Customer_IDs:
    realCreditMix = df_train[df_train['Customer_ID']==id]['Credit_Mix'].drop_duplicates()
    if realCreditMix =='_':
        realCreditMix = df_train[df_train['Customer_ID'] ==id]['Credit_Mix'].drop_duplica

    df_train.loc[(df_train['Customer_ID'] == id) & (
        df_train['Credit_Mix'] == '_'), ['Credit_Mix']] = realCreditMix
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x942c']['Credit_Mix']
```

Out[130]:

```
99992    Good
99993    Good
99994    Good
99995    Good
99996    Good
99997    Good
99998    Good
99999    Good
Name: Credit_Mix, dtype: object
```

In [ ]:

```
df_train[df_train['Credit_Mix'] == '_']
```

Out[131]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[19] Outstanding Debt

- _

In [ ]:

```
print(df_train['Outstanding_Debt'].value_counts())
print("#"*20)
print('Missing = ',df_train['Outstanding_Debt'].isnull().sum())
```

```
1360.45     24
460.46      23
1151.7      23
1109.03     23
467.7       16
            ..
245.46_      1
645.77_      1
174.79_      1
1181.13_     1
1013.53_     1
Name: Outstanding_Debt, Length: 13178, dtype: int64
####################
Missing =  0
```

In [ ]:

```
Remove_value(df_train, 'Outstanding_Debt')
df_train['Outstanding_Debt'].value_counts() # check
```

Out[133]:

```
1109.03    24
1151.7     24
1360.45    24
460.46     24
1058.13    16
           ..
4230.04     8
641.99      8
98.61       8
2614.48     8
502.38      8
Name: Outstanding_Debt, Length: 12203, dtype: int64
```

[21] Credit History Age

In [ ]:

```
print(df_train['Credit_History_Age'].value_counts())
print("#"*20)
print('Missing = ',df_train['Credit_History_Age'].isnull().sum())
```

```
15 Years and 11 Months    446
19 Years and 4 Months     445
19 Years and 5 Months     444
17 Years and 11 Months    443
19 Years and 3 Months     441
                         ...
0 Years and 3 Months       20
0 Years and 2 Months       15
33 Years and 7 Months      14
33 Years and 8 Months      12
0 Years and 1 Months        2
Name: Credit_History_Age, Length: 404, dtype: int64
####################
Missing =  9030
```

In [ ]:

```
df_train[df_train['Credit_History_Age'].isna()]
```

Out[135]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income
1	0x1603	CUS_0xd40	February	Aaron Maashoh	23	821-00-0265	Scientist	19114.12
7	0x1609	CUS_0xd40	August	Aaron Maashoh	23	821-00-0265	Scientist	19114.12
19	0x161d	CUS_0x2dbc	April	Langep	34	486-85-3974	Engineer	143162.64
40	0x163e	CUS_0x95ee	January	Np	31	612-70-8987	Lawyer	73928.46
42	0x1640	CUS_0x95ee	March	Np	31	612-70-8987	Lawyer	73928.46
...	...	...	...	...	...	...	...	...
99944	0x25f9e	CUS_0x51b3	January	Ryana	33	837-85-9800	Media_Manager	59146.36
99963	0x25fb9	CUS_0x372c	April	Lucia Mutikanik	18	340-85-7301	Lawyer	42903.79
99968	0x25fc2	CUS_0xf16	January	Maria Sheahanb	44	868-70-2218	Media_Manager	16680.35
99975	0x25fc9	CUS_0xf16	August	Maria Sheahanb	45	868-70-2218	Media_Manager	16680.35
99990	0x25fe0	CUS_0x8600	July	Sarah McBridec	28	031-35-0942	Architect	20002.88

9030 rows × 28 columns

In [ ]:

```
valuesNAN ='0 Years and 0 Months'
df_train['Credit_History_Age'].fillna(valuesNAN, inplace=True)
Customer_IDs = df_train[df_train['Credit_History_Age']== valuesNAN]['Customer_ID']
User_IDs = df_train[df_train['Credit_History_Age']==valuesNAN]['ID']
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0xf16']['Credit_History_Age']
```

Out[142]:

```
99968     0 Years and 0 Months
99969    21 Years and 2 Months
99970    21 Years and 3 Months
99971    21 Years and 4 Months
99972    21 Years and 5 Months
99973    21 Years and 6 Months
99974    21 Years and 7 Months
99975     0 Years and 0 Months
Name: Credit_History_Age, dtype: object
```

In [ ]:

```
for cust_id, user_id in zip(Customer_IDs, User_IDs):
    # init
    temp_hist_value = [0, 'Years', 'and', 0, 'Months']
    list_split = []
    new_month = 0
    new_year = 0
    myValues = []
    supporte_year = 0
    supporte_month = 0
    steps = 0

# values of history of this customer only
    myValues = df_train[df_train["Customer_ID"] ==
                        cust_id]['Credit_History_Age'].values.tolist()

# git index of nan (valuesNAN)
    indxNAN = myValues.index(valuesNAN)
# split  string into list
    for i in myValues:
        list_split.append(i.split(" "))

# new year and month

# case one
    if indxNAN != 0:
        supporte_year = int(list_split[indxNAN-1][0])
        supporte_month = int(list_split[indxNAN-1][3])

        if supporte_month == 11:
            new_month = 0
            new_year = supporte_year+1
        else:
            new_month = supporte_month+1
            new_year = supporte_year

# case two
    #if missing in index 0
    else:
        # new year and month
        supporte_year = int(list_split[indxNAN+1][0])
        supporte_month = int(list_split[indxNAN+1][3])

    # case  two [1] --> if missing in index 0 and 1 and ...
        if supporte_month == 0 and supporte_year == 0:
            for step in myValues:
                if step == valuesNAN:
                    steps = steps+1
                else:
                    break

            supporte_month = int(list_split[indxNAN+steps][3]) - steps

            if supporte_month <= 0:
                new_month = supporte_month - steps + 12
                new_year = int(list_split[indxNAN+steps][0])-1
            else:
                new_month = supporte_month - steps
                new_year = int(list_split[indxNAN+steps][0])
```

```
        else:
            if supporte_month == 0:
                new_month = 11
                new_year = supporte_year - 1
            else:
                new_month = supporte_month-1
                new_year = supporte_year
    # new value
    temp_hist_value[0] = str(new_year)
    temp_hist_value[3] = str(new_month)

# replace new valu

    df_train.loc[(df_train['Customer_ID'] == cust_id) & (
        (df_train['Credit_History_Age'] == valuesNAN) & (df_train['ID'] == user_id)), ['C
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0xf16']['Credit_History_Age']
```

Out[144]:

```
99968    21 Years and 1 Months
99969    21 Years and 2 Months
99970    21 Years and 3 Months
99971    21 Years and 4 Months
99972    21 Years and 5 Months
99973    21 Years and 6 Months
99974    21 Years and 7 Months
99975    21 Years and 8 Months
Name: Credit_History_Age, dtype: object
```

In [ ]:

```
df_train[df_train['Credit_History_Age']== valuesNAN]
```

Out[145]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[23] Total EMI per month

- missing = 0.000
- values > 360 --> outlier

In [ ]:

```
print(df_train['Total_EMI_per_month'].value_counts())
print("#"*20)
print('Missing = ',df_train['Total_EMI_per_month'].isnull().sum())
```

```
0.000000        10613
49.574949           8
73.533361           8
22.960835           8
38.661127           8
                ...
36408.000000        1
23760.000000        1
24612.000000        1
24325.000000        1
58638.000000        1
Name: Total_EMI_per_month, Length: 14950, dtype: int64
####################
Missing =  0
```

In [ ]:

```
px.box(df_train,y='Total_EMI_per_month')
```

In [ ]:

```
df_train[(df_train['Total_EMI_per_month'] == 0.00) | (
    df_train['Total_EMI_per_month'] > 360)]
```

Out[148]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income
32	0x1632	CUS_0x1cdb	January	Deepaa	21	615-06-7821	Developer	35547.71
33	0x1633	CUS_0x1cdb	February	Deepaa	21	615-06-7821	Developer	35547.71
34	0x1634	CUS_0x1cdb	March	Deepaa	21	615-06-7821	Developer	35547.71
35	0x1635	CUS_0x1cdb	April	Deepaa	21	615-06-7821	Developer	35547.71
36	0x1636	CUS_0x1cdb	May	Deepaa	21	615-06-7821	Developer	35547.71
...	...	...	...	...	...	...	...	...
99959	0x25fb1	CUS_0x2084	August	Ryanl	21	253-72-7758	Architect	38321.39
99960	0x25fb6	CUS_0x372c	January	Lucia Mutikanik	18	340-85-7301	Lawyer	42903.79
99970	0x25fc4	CUS_0xf16	March	Maria Sheahanb	45	868-70-2218	Media_Manager	16680.35
99985	0x25fdb	CUS_0x8600	February	Sarah McBridec	28	031-35-0942	Architect	20002.88
99993	0x25fe7	CUS_0x942c	February	Nicks	25	078-73-5990	Mechanic	39628.99

17361 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x8600']['Total_EMI_per_month']
```

Out[149]:

```
99984       60.964772
99985    12112.000000
99986       60.964772
99987       60.964772
99988       60.964772
99989       60.964772
99990       60.964772
99991       60.964772
Name: Total_EMI_per_month, dtype: float64
```

In [ ]:

```
Customer_IDs = df_train[(df_train['Total_EMI_per_month'] == 0.00) | (
    df_train['Total_EMI_per_month'] > 360)]['Customer_ID'].values

for i in Customer_IDs:
    total_values = df_train[df_train['Customer_ID'] ==
                            i]['Total_EMI_per_month'].drop_duplicates().values
    len_values = total_values.shape[0]
    realvalue = 0

    if len_values > 1:
        if total_values[0] != 0.00:
            realvalue = total_values[0]
        else:
            realvalue = total_values[1]

        df_train.loc[(df_train['Customer_ID'] == i) & ((df_train['Total_EMI_per_month'] =
                                                        (df_train['Total_EMI_per_month']
                     ['Total_EMI_per_month']] = realvalue
```

In [ ]:

```
print(df_train['Total_EMI_per_month'].value_counts())
print("#"*20)
print('Missing = ',df_train['Total_EMI_per_month'].isnull().sum())
```

```
0.000000        7992
50548.000000      10
72572.000000       9
66.681222          8
92.738644          8
                ...
54544.000000       1
35821.000000       1
78.772697          1
58509.000000       1
320.505265         1
Name: Total_EMI_per_month, Length: 12191, dtype: int64
####################
Missing =  0
```

In [ ]:

```
df_train['Total_EMI_per_month'].replace(0.00, np.NaN, inplace=True)
```

In [ ]:

```
# fill by knn imputer
imputer = KNNImputer(n_neighbors=8)
df_train['Total_EMI_per_month'] = imputer.fit_transform(
    df_train[['Total_EMI_per_month']])


fig = px.histogram(df_train, x="Total_EMI_per_month")
fig.show()
```

In [ ]:

```
df_train[df_train['Customer_ID'] == 'CUS_0x8600']['Total_EMI_per_month']
```

Out[154]:

```
99984    60.964772
99985    60.964772
99986    60.964772
99987    60.964772
99988    60.964772
99989    60.964772
99990    60.964772
99991    60.964772
Name: Total_EMI_per_month, dtype: float64
```

In [ ]:

```
print(df_train['Total_EMI_per_month'].value_counts())
print("#"*20)
print('Missing = ',df_train['Total_EMI_per_month'].isnull().sum())
```

```
1563.673641     7992
50548.000000      10
72572.000000       9
66.681222          8
92.738644          8
                ...
54544.000000       1
35821.000000       1
78.772697          1
58509.000000       1
320.505265         1
Name: Total_EMI_per_month, Length: 12191, dtype: int64
####################
Missing =  0
```

[24] Amount invested monthly

- NaN
- __ 10000 __
- 0.0

In [ ]:

```
print(df_train['Amount_invested_monthly'].value_counts())
print("#"*20)
print('Missing = ',df_train['Amount_invested_monthly'].isnull().sum())
```

```
__10000__             4305
0.0                    169
80.41529543900253        1
36.66235139442514        1
89.7384893604547         1
                       ...
36.541908593249026       1
93.45116318631192        1
140.80972223052834       1
38.73937670100975        1
167.1638651610451        1
Name: Amount_invested_monthly, Length: 91049, dtype: int64
####################
Missing =  4479
```

In [ ]:

```
df_train['Amount_invested_monthly'].replace('0.0',np.NaN,inplace=True)
df_train['Amount_invested_monthly'].replace('__10000__',np.NaN,inplace=True)
```

In [ ]:

```
ConvertDataType(df_train,'Amount_invested_monthly',float)
```

In [ ]:

```
fig = px.histogram(df_train, x="Amount_invested_monthly")
fig.show()
```

In [ ]:

```
df_train[df_train['Amount_invested_monthly'].isna()]
```

Out[162]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income
18	0x161c	CUS_0x2dbc	March	Langep	34	486-85-3974	Engineer	143162.64
23	0x1621	CUS_0x2dbc	August	Langep	34	486-85-3974	Engineer	143162.64
28	0x162a	CUS_0xb891	May	Jasond	55	072-31-6145	Entrepreneur	30689.89
60	0x165a	CUS_0x5407	May	Annk	30	500-92-6408	Media_Manager	34081.38
61	0x165b	CUS_0x5407	June	Annk	30	500-92-6408	Media_Manager	34081.38
...	...	...	...	...	...	...	...	...
99931	0x25f89	CUS_0xb11c	April	Yinka Adegokej	38	546-94-4789	Manager	15319.65
99951	0x25fa5	CUS_0x51b3	August	Ryana	33	837-85-9800	Media_Manager	59146.36
99961	0x25fb7	CUS_0x372c	February	Lucia Mutikanik	18	340-85-7301	Lawyer	42903.79
99973	0x25fc7	CUS_0xf16	June	Maria Sheahanb	45	868-70-2218	Media_Manager	16680.35
99974	0x25fc8	CUS_0xf16	July	Maria Sheahanb	45	868-70-2218	Media_Manager	16680.35

8953 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0xf16']['Amount_invested_monthly']
```

Out[163]:

```
99968     52.951978
99969    104.646237
99970     75.504972
99971    199.988581
99972     64.549748
99973           NaN
99974           NaN
99975     70.805550
Name: Amount_invested_monthly, dtype: float64
```

In [ ]:

```
Customer_IDs =df_train[df_train['Amount_invested_monthly'].isna()]['Customer_ID'].values

for i in Customer_IDs:
    min_value = df_train[df_train['Customer_ID'] ==i]['Amount_invested_monthly'].min()
    mean_value = df_train[df_train['Customer_ID'] ==i]['Amount_invested_monthly'].mean()

    df_train.loc[(df_train['Customer_ID'] == i) &(df_train['Amount_invested_monthly'].isr
                , ['Amount_invested_monthly']] = mean_value - min_value
```

In [ ]:

```
fig = px.histogram(df_train, x="Amount_invested_monthly")
fig.show()
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0xf16']['Amount_invested_monthly']
```

Out[166]:

```
99968     52.951978
99969    104.646237
99970     75.504972
99971    199.988581
99972     64.549748
99973     41.789200
99974     41.789200
99975     70.805550
Name: Amount_invested_monthly, dtype: float64
```

In [ ]:

```
df_train[df_train['Amount_invested_monthly'].isna()]
```

Out[167]:

ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Sa

0 rows × 28 columns

[25] Payment Behaviour

- !@9#%8 >> missing

In [ ]:

```
print(df_train['Payment_Behaviour'].value_counts())
print("#"*20)
print('Missing = ',df_train['Payment_Behaviour'].isnull().sum())
```

```
Low_spent_Small_value_payments      25513
High_spent_Medium_value_payments    17540
Low_spent_Medium_value_payments     13861
High_spent_Large_value_payments     13721
High_spent_Small_value_payments     11340
Low_spent_Large_value_payments      10425
!@9#%8                               7600
Name: Payment_Behaviour, dtype: int64
####################
Missing =  0
```

In [ ]:

```
df_train['Payment_Behaviour'].replace('!@9#%8',np.NaN,inplace=True)
df_train['Payment_Behaviour'].isnull().sum()
```

Out[169]:

```
7600
```

In [ ]:

```
df_train[df_train['Payment_Behaviour'].isna()]
```

Out[170]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income
5	0x1607	CUS_0xd40	June	Aaron Maashoh	23	821-00-0265	Scientist	19114.12
16	0x161a	CUS_0x2dbc	January	Langep	34	486-85-3974	Engineer	143162.64
32	0x1632	CUS_0x1cdb	January	Deepaa	21	615-06-7821	Developer	35547.71
47	0x1645	CUS_0x95ee	August	Np	31	612-70-8987	Lawyer	73928.46
54	0x1650	CUS_0x284a	July	Nadiaq	34	411-51-0676	Lawyer	131313.40
...	...	...	...	...	...	...	...	...
99947	0x25fa1	CUS_0x51b3	April	Ryana	33	837-85-9800	Media_Manager	59146.36
99980	0x25fd2	CUS_0xaf61	May	Chris Wickhamm	49	133-16-7738	Writer	37188.10
99982	0x25fd4	CUS_0xaf61	July	Chris Wickhamm	50	133-16-7738	Writer	37188.10
99989	0x25fdf	CUS_0x8600	June	Sarah McBridec	28	031-35-0942	Architect	20002.88
99999	0x25fed	CUS_0x942c	August	Nicks	25	078-73-5990	Mechanic	39628.99

7600 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0xaf61']['Payment_Behaviour']
```

Out[171]:

```
99976      Low_spent_Small_value_payments
99977     High_spent_Large_value_payments
99978     Low_spent_Medium_value_payments
99979      Low_spent_Small_value_payments
99980                                 NaN
99981      Low_spent_Small_value_payments
99982                                 NaN
99983    High_spent_Medium_value_payments
Name: Payment_Behaviour, dtype: object
```

In [ ]:

```
Customer_IDs = df_train[df_train['Payment_Behaviour'].isna()]['Customer_ID']

for i in Customer_IDs:
    mode_val = df_train[df_train['Customer_ID'] == i]['Payment_Behaviour'].mode()[0]
    if mode_val == np.NaN:
        mode_val = df_train[df_train['Customer_ID'] == i]['Payment_Behaviour'].mode()[1]

    df_train.loc[(df_train['Customer_ID'] == i) & (df_train)[
        'Payment_Behaviour'].isna(), ['Payment_Behaviour']] = mode_val
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0xaf61']['Payment_Behaviour']
```

Out[173]:

```
99976      Low_spent_Small_value_payments
99977     High_spent_Large_value_payments
99978     Low_spent_Medium_value_payments
99979      Low_spent_Small_value_payments
99980      Low_spent_Small_value_payments
99981      Low_spent_Small_value_payments
99982      Low_spent_Small_value_payments
99983    High_spent_Medium_value_payments
Name: Payment_Behaviour, dtype: object
```

In [ ]:

```
df_train['Payment_Behaviour'].isnull().sum()  # Check
```

Out[174]:

```
0
```

[26] Monthly Balance

In [ ]:

```
print(df_train['Monthly_Balance'].value_counts())
print("#"*20)
print('Missing = ',df_train['Monthly_Balance'].isnull().sum())
```

```
__-333333333333333333333333__    9
312.49408867943663                1
415.32532309844316                1
252.08489793906085                1
254.9709216273975                 1
                                 ..
366.2890379762706                 1
151.1882696261166                 1
306.75027851710234                1
278.8720257394474                 1
393.6736955618808                 1
Name: Monthly_Balance, Length: 98792, dtype: int64
####################
Missing =  1200
```

In [ ]:

```
df_train['Monthly_Balance'].replace('__-333333333333333333333333__',np.NaN,inplace=Tru
df_train['Monthly_Balance'].isnull().sum()
```

Out[176]:

```
1209
```

In [ ]:

```
# convert data
ConvertDataType(df_train,'Monthly_Balance',float)
```

In [ ]:

```
fig = px.histogram(df_train, x="Monthly_Balance")
fig.show()
```

In [ ]:

```
df_train[df_train['Monthly_Balance'].isna()]
```

Out[179]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Incom
197	0x1727	CUS_0xa5f9	June	Rickt	34	189-09-5267	Mechanic	57983.1
314	0x17d8	CUS_0x571f	March	Emily Flitterl	18	397-28-9675	Developer	66567.3
388	0x1846	CUS_0x9b3c	May	Scheydere	43	224-48-7837	Lawyer	81842.2
456	0x18ae	CUS_0x9d78	January	Tim Hepherp	20	423-77-6457	Scientist	69977.6
457	0x18af	CUS_0x9d78	February	Tim Hepherp	20	423-77-6457	Scientist	69977.6
...	...	...	...	...	...	...	...	
99820	0x25ee2	CUS_0x40ad	May	Dorisw	20	715-14-3312	Scientist	65858.4
99839	0x25efd	CUS_0x8788	August	Sophie Sassardv	25	138-38-7150	Musician	71025.4
99852	0x25f12	CUS_0x3048	May	Rick Rothackeru	40	375-64-6913	Scientist	81093.1
99854	0x25f14	CUS_0x3048	July	Rick Rothackeru	40	375-64-6913	Scientist	81093.1
99927	0x25f81	CUS_0x2654	August	enj	38	647-67-8889	Media_Manager	139664.9

1209 rows × 28 columns

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x9d78']['Monthly_Balance']
```

Out[180]:

```
456           NaN
457           NaN
458    214.190642
459    141.330065
460    229.365166
461     59.886193
462           NaN
463           NaN
Name: Monthly_Balance, dtype: float64
```

In [ ]:

```
Customer_IDs=df_train[df_train['Monthly_Balance'].isna()]['Customer_ID'].values

for i in Customer_IDs:
    mean_val=df_train[df_train['Customer_ID'] == i]['Monthly_Balance'].mean()
    df_train.loc[(df_train['Customer_ID'] == i) & (df_train['Monthly_Balance'].isna()),['
```

In [ ]:

```
fig = px.histogram(df_train, x="Monthly_Balance")
fig.show()
```

In [ ]:

```
df_train[df_train['Customer_ID']=='CUS_0x9d78']['Monthly_Balance']
```

Out[183]:

```
456    161.193017
457    161.193017
458    214.190642
459    141.330065
460    229.365166
461     59.886193
462    161.193017
463    161.193017
Name: Monthly_Balance, dtype: float64
```

[27] Credit_Score

In [ ]:

```
print(df_train['Credit_Score'].value_counts())
print("#"*20)
print('Missing = ',df_train['Credit_Score'].isnull().sum())
```

```
Standard    53174
Poor        28998
Good        17828
Name: Credit_Score, dtype: int64
####################
Missing =  0
```

In [ ]:

```
px.pie(df_train,names='Credit_Score')
```

Note :

> We will oversampling the target by SMOTE

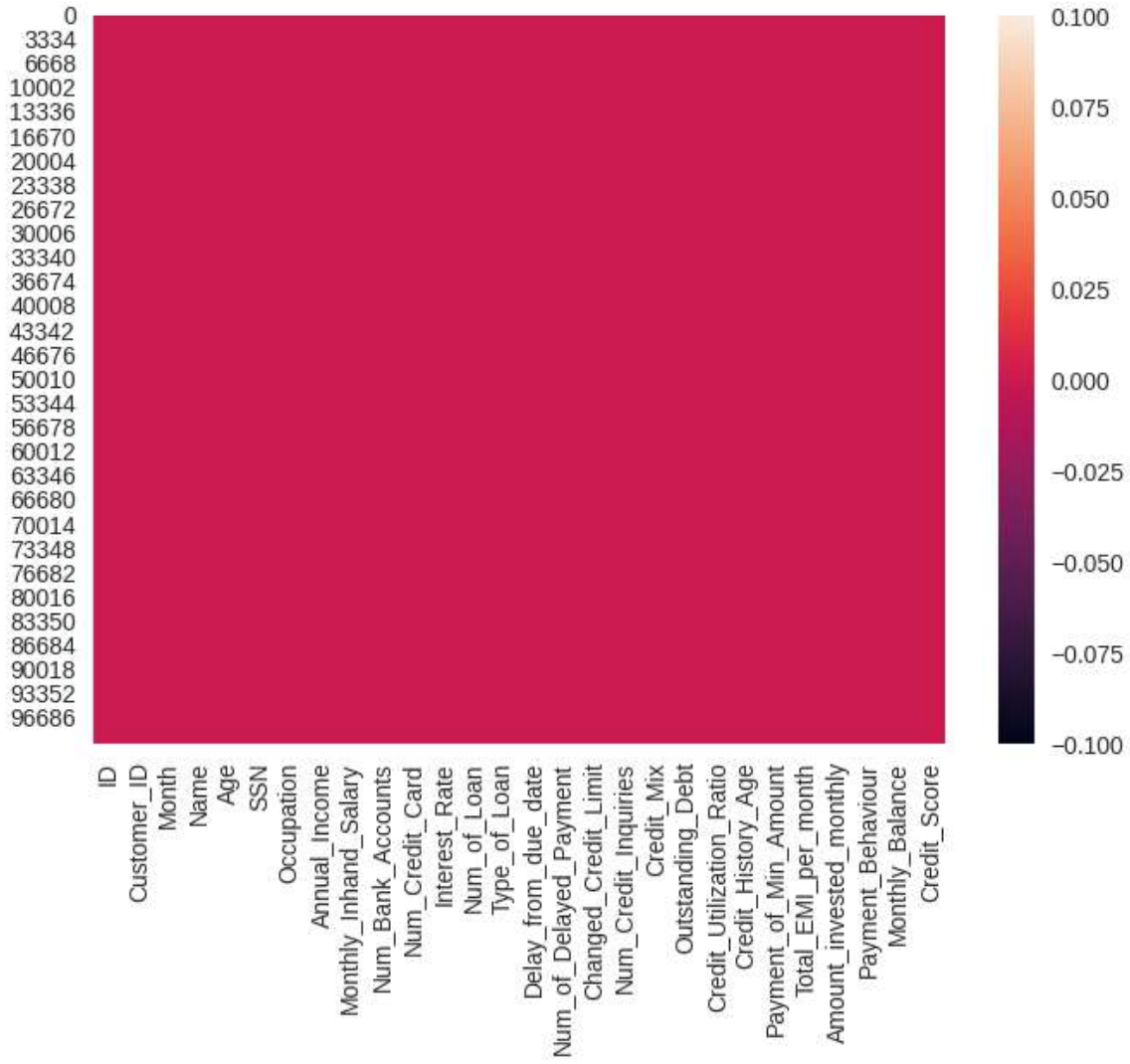
Check over Handling Missing Values

In [ ]:

```
sns.heatmap(df_train.isnull())
```

Out[186]:

<Axes: >



In [ ]:

```
df_train.isnull().sum()
```

Out[187]:

```
ID                          0
Customer_ID                 0
Month                       0
Name                        0
Age                         0
SSN                         0
Occupation                  0
Annual_Income               0
Monthly_Inhand_Salary       0
Num_Bank_Accounts           0
Num_Credit_Card             0
Interest_Rate               0
Num_of_Loan                 0
Type_of_Loan                0
Delay_from_due_date         0
Num_of_Delayed_Payment      0
Changed_Credit_Limit        0
Num_Credit_Inquiries        0
Credit_Mix                  0
Outstanding_Debt            0
Credit_Utilization_Ratio    0
Credit_History_Age          0
Payment_of_Min_Amount       0
Total_EMI_per_month         0
Amount_invested_monthly     0
Payment_Behaviour           0
Monthly_Balance             0
Credit_Score                0
dtype: int64
```

Save Data After Processing Missing Values

In [ ]:

```
df_train.to_csv("Preprocessed_Missing_dataset.csv", index=False)
```

In [ ]:

```
# # Colab
# files.download('Preprocessed_Missing_dataset.csv')
```

<IPython.core.display.Javascript object>

<IPython.core.display.Javascript object>

In [ ]:

```
# read  Preprocessed_Missing_dataset

# colab
PATH_PRE_MISSING = '/content/drive/MyDrive/ds/Preprocessed_Missing_dataset.csv'

# my pc
# PATH_PRE_MISSING ='ignoreFiles\Preprocessed_Missing_dataset.csv'

df_train = pd.read_csv(PATH_PRE_MISSING,encoding='utf-8',sep=',')
df_copy=df_train.copy()
```

EDA

Tableau public.tableau.com/app/profile/zeyad.usf/viz/CreditScoreEDA/CreditScoreEDA2

Preprocessing

Encoding

In [ ]:

```
le = LabelEncoder()

for i in df_train.select_dtypes('object').columns:
  df_train[i] = le.fit_transform(df_train[i])
```

Feature Select

- chi-square

In [ ]:

```
not_important_features =[]
```

In [ ]:

```
# Create a contingency table for each categorical column
for col in df_train.columns:
    contingency_table = pd.crosstab(df_train[col], df_train['Credit_Score'])
    # Apply the chi-square test
    chi2, p, dof, expected = stats.chi2_contingency(contingency_table)
    print(f"Chi-square test results for {col}:")
    print(f"Chi-square statistic: {chi2}")
    print(f"P-value: {p}")
    print(f"Degrees of freedom: {dof}")
    print('-'*15)
    if p != 0.0 :not_important_features.append(col)
    # print(f"Expected frequencies table:\n{expected}\n")
```

```
Chi-square test results for ID:
Chi-square statistic: 199999.99999999997
P-value: 0.4983179126799479
Degrees of freedom: 199998
---------------
Chi-square test results for Customer_ID:
Chi-square statistic: 133795.84086124436
P-value: 0.0
Degrees of freedom: 24998
---------------
Chi-square test results for Month:
Chi-square statistic: 201.80530639299855
P-value: 2.3494099136740425e-35
Degrees of freedom: 14
---------------
Chi-square test results for Name:
Chi-square statistic: 108816.79465970308
P-value: 0.0
Degrees of freedom: 20276
```

In [ ]:

```
X = df_train.drop(['Credit_Score','Monthly_Balance','Amount_invested_monthly','Credit_Uti
                   'Occupation','Month','ID'],axis=1)
Y=df_train['Credit_Score']
```

Scaling

In [ ]:

```
sc = StandardScaler()
X = sc.fit_transform(X)
```

Split Data (Train - Test)

In [ ]:

```
x_train , x_test , y_train , y_test =train_test_split(X,Y,test_size=0.2,random_state=42)
```

In [ ]:

```
print(X.shape, x_train.shape,x_test.shape,sep='\n')
print('-------\n',y_train.value_counts())
```

```
(100000, 21)
(80000, 21)
(20000, 21)
-------
 2    42575
1    23124
0    14301
Name: Credit_Score, dtype: int64
```

Balanced Data

In [ ]:

```
sm = SMOTE(k_neighbors=7)
x_train_sm,y_train_sm=sm.fit_resample(x_train,y_train)
```

In [ ]:

```
print(X.shape, x_train_sm.shape, x_train.shape, sep='\n')
print('-------\n', y_train_sm.value_counts())
```

```
(100000, 21)
(127725, 21)
(80000, 21)
-------
 2    42575
0    42575
1    42575
Name: Credit_Score, dtype: int64
```

Build Models

- First Approach : PyCaret
- Second Approach : Random Forest Model

First Approach : PyCaret

In [ ]:

```
s = setup(x_train_sm, target = y_train_sm,normalize=True)
```

	Description	Value
0	Session id	7152
1	Target	Credit_Score
2	Target type	Multiclass
3	Original data shape	(127725, 22)
4	Transformed data shape	(127725, 22)
5	Transformed train set shape	(89407, 22)
6	Transformed test set shape	(38318, 22)
7	Numeric features	21
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Normalize	True
13	Normalize method	zscore
14	Fold Generator	StratifiedKFold
15	Fold Number	10
16	CPU Jobs	-1
17	Use GPU	False
18	Log Experiment	False
19	Experiment Name	clf-default-name
20	USI	1984

In [ ]:

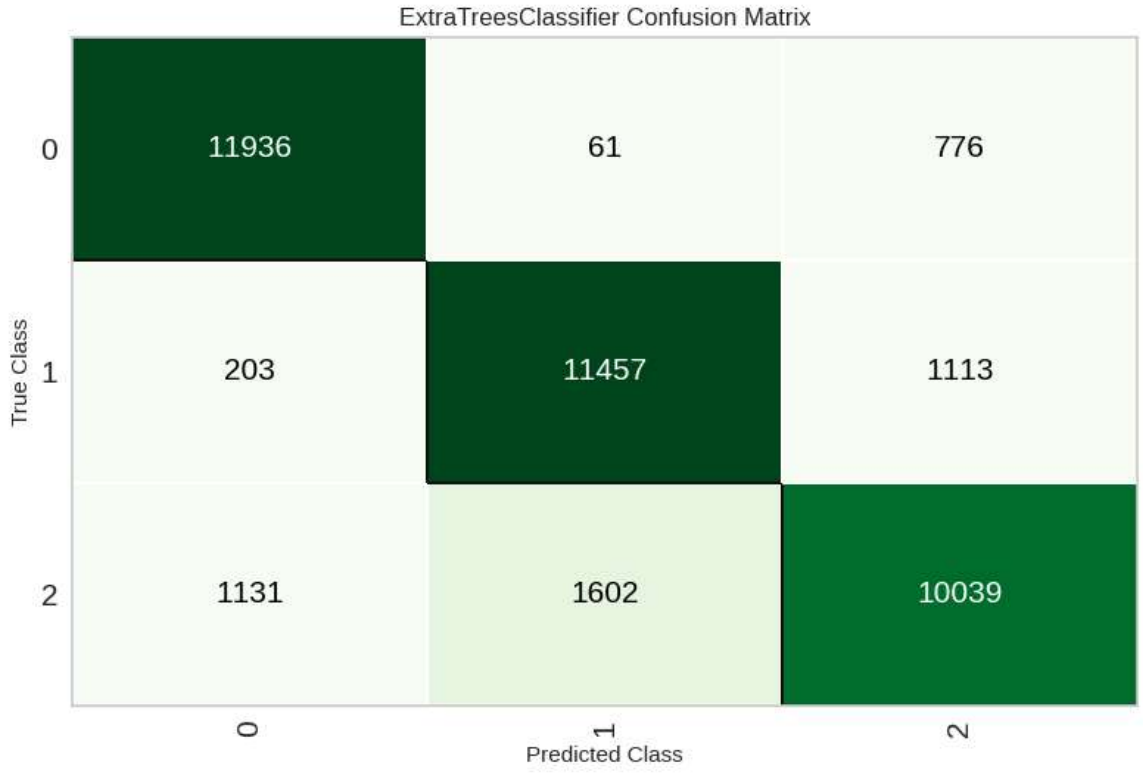
```
best_model = compare_models(include=['knn','et','rf','xgboost','dt'])
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
et	Extra Trees Classifier	0.8664	0.9506	0.8664	0.8654	0.8654	0.7997	0.8001	15.8010
rf	Random Forest Classifier	0.8641	0.9541	0.8641	0.8633	0.8630	0.7962	0.7969	36.8500
xgboost	Extreme Gradient Boosting	0.8303	0.9467	0.8303	0.8300	0.8294	0.7455	0.7463	73.2510
knn	K Neighbors Classifier	0.8200	0.9270	0.8200	0.8249	0.8119	0.7300	0.7385	9.0000
dt	Decision Tree Classifier	0.7891	0.8418	0.7891	0.7890	0.7890	0.6837	0.6837	2.2940

```
Processing:   0%|          | 0/25 [00:00<?, ?it/s]
```

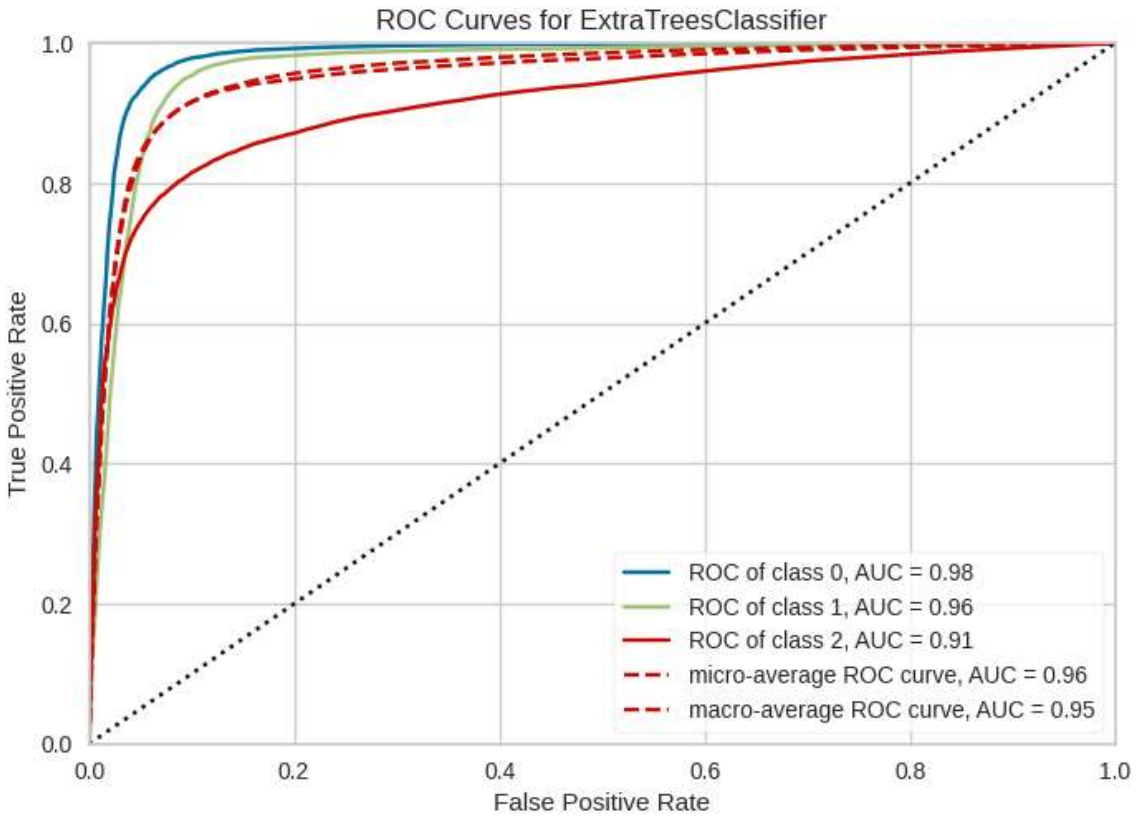
In [ ]:

```
plot_model(best_model, plot = 'confusion_matrix')
```



In [ ]:

```
plot_model(best_model, plot = 'auc')
```



In [ ]:

```
holdout_pred = predict_model(best_model)
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Extra Trees Classifier	0.8725	0.9534	0.8725	0.8715	0.8715	0.8087	0.8092

In [ ]:

```
holdout_pred.head()
```

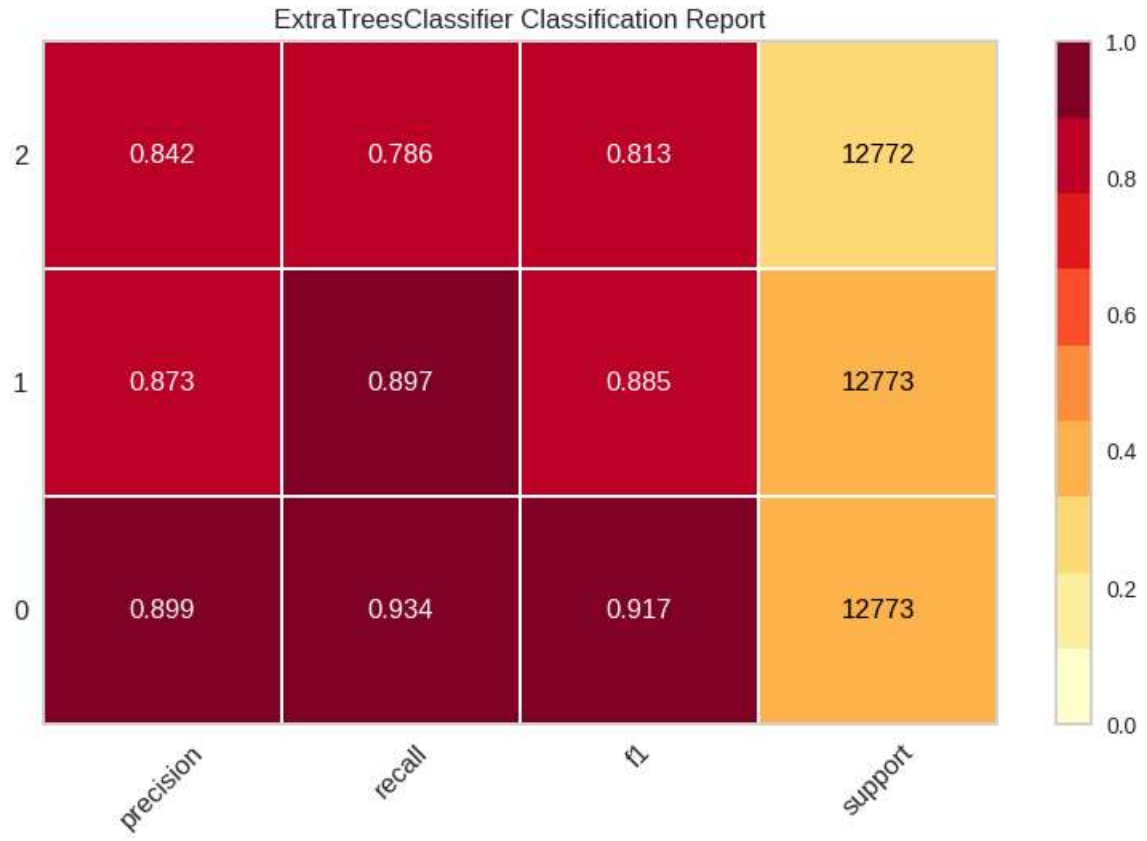
Out[51]:

	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8
60206	-1.269108	-0.049424	-0.771340	0.917156	2.126048	2.146917	-0.142216	-0.135401
11537	0.657209	0.112108	0.250503	1.722213	-0.587351	-0.667319	1.400259	-0.119904
34857	-1.055720	1.240429	0.064713	1.335619	-0.987628	-0.931874	0.629021	-0.150898
20709	-0.849813	-0.008955	-0.771340	1.346427	-1.119928	-1.026041	1.014640	-0.135401
118891	0.995583	1.685584	-1.151126	-0.855910	-1.059639	-1.080795	0.629021	-0.104407

5 rows × 24 columns

In [ ]:

```
plot_model(best_model, plot = 'class_report')
```



Second Approach : Random Forest Model

In [ ]:

```
rf = RandomForestClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                            max_depth=None, max_features=None, max_leaf_nodes=None,
                            min_impurity_decrease=0.0, min_samples_leaf=1,
                            min_samples_split=2, min_weight_fraction_leaf=0.0,
                            random_state=7152)
```

In [ ]:

```
rf.fit(x_train_sm, y_train_sm)
```

Out[88]:

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features=Non
e,
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_samples_leaf=1,
                       min_samples_split=2, min_weight_fraction_leaf=0.0,
                       n_estimators=100, n_jobs=None, oob_score=False,
                       random_state=7152, verbose=0, warm_start=False)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [ ]:

```
y_pred_rf = rf.predict(x_test)
```

In [ ]:

```
print(metrics.classification_report(y_test,y_pred_rf))
```

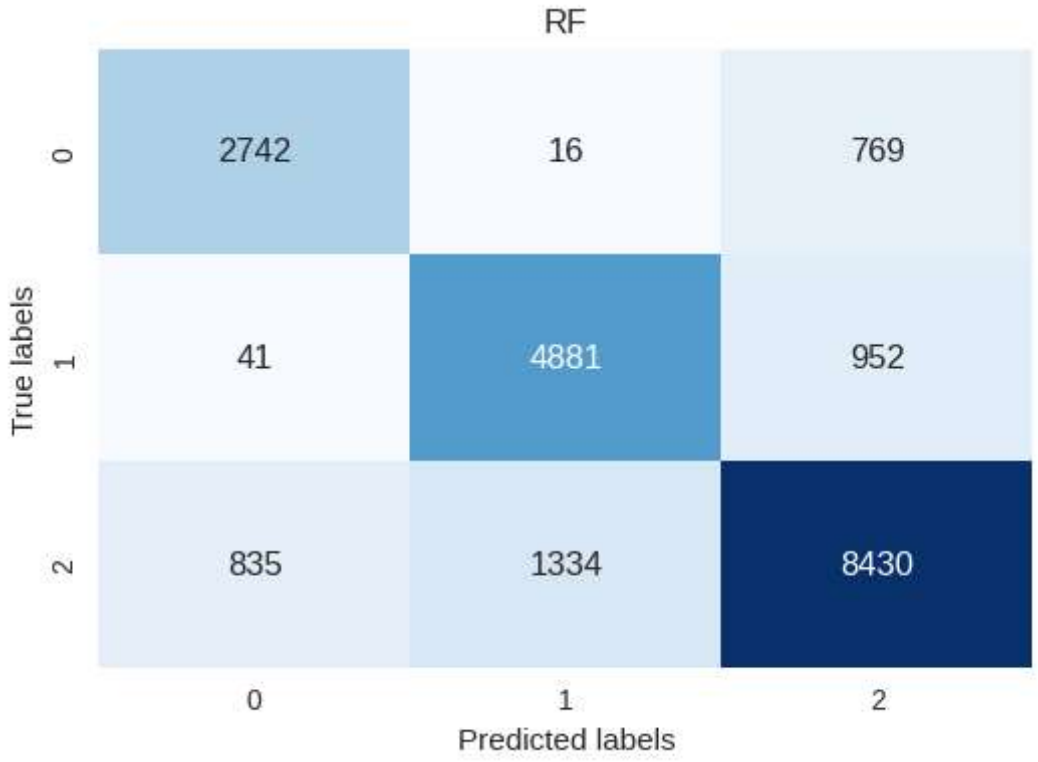
```
              precision    recall  f1-score   support

           0       0.76      0.78      0.77      3527
           1       0.78      0.83      0.81      5874
           2       0.83      0.80      0.81     10599

    accuracy                           0.80     20000
   macro avg       0.79      0.80      0.80     20000
weighted avg       0.80      0.80      0.80     20000
```

In [ ]:

```
fig, ax = plt.subplots(figsize=(6, 4))
ax.set_title("RF")
sns.heatmap(metrics.confusion_matrix(y_test, y_pred_rf),
            annot=True, cmap='Blues', fmt='g', cbar=False, ax=ax)
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.show()
```



Prediction for Models

In [ ]:

```
# read data
PATH_NEW_DF ='/content/drive/MyDrive/ds/test.csv'
new_test_df = pd.read_csv(PATH_NEW_DF, encoding='utf-8', sep=',')
```

In [ ]:

```
# encoding
for i in new_test_df.select_dtypes('object').columns:
    new_test_df[i] = le.fit_transform(new_test_df[i])

# selection
new_test_df.drop(['Monthly_Balance','Amount_invested_monthly','Credit_Utilization_Ratio',
                  'Occupation','Month','ID'], axis=1, inplace=True)
#
columns_pyCaret=get_config('X').columns
columns=new_test_df.columns


# scaling
new_test_df = sc.fit_transform(new_test_df)

# convert numpy array back to pandas dataframe
new_test_df_PyCaret = pd.DataFrame(new_test_df, columns=columns_pyCaret)
new_test_df= pd.DataFrame(new_test_df, columns=columns_cv)
```

Prediction By pyCaret

In [ ]:

```
predictions_pycaret = predict_model(best_model, data=new_test_df_PyCaret)
```

In [ ]:

```
predictions_pycaret.head()
```

Out[62]:

	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8	featu
0	1.682306	-1.745317	-0.994264	1.134907	-0.650926	-0.742630	-0.118890	-0.146323	-0.14
1	1.682306	-1.745317	-0.897783	1.134907	-0.650926	-0.742630	-0.118890	-0.146323	-0.14
2	1.682306	-1.745317	-0.897783	1.134907	-0.650926	-0.742630	-0.118890	-0.146323	-0.14
3	1.682306	-1.745317	-0.820598	1.134907	-0.650926	NaN	-0.118890	-0.146323	-0.14
4	-1.436494	0.602678	-0.550450	-1.542797	0.134881	-0.360426	-0.127481	-0.146323	-0.13

5 rows × 23 columns

In [ ]:

```
test_score_pyC=(predictions_pycaret[predictions_pycaret['prediction_score'] >0.49].shape[
```

In [ ]:

```
test_score_pyC
```

Out[118]:

```
52.292
```

In [ ]:

```
# Save Model
save_model(best_model, 'PyCaret_et_Model')
```

```
Transformation Pipeline and Model Successfully Saved
```

Out[67]:

```
(Pipeline(memory=FastMemory(location=/tmp/joblib),
          steps=[('numerical_imputer',
                  TransformerWrapper(exclude=None,
                                     include=['feature_1', 'feature_2',
                                              'feature_3', 'feature_4',
                                              'feature_5', 'feature_6',
                                              'feature_7', 'feature_8',
                                              'feature_9', 'feature_10',
                                              'feature_11', 'feature_12',
                                              'feature_13', 'feature_14',
                                              'feature_15', 'feature_16',
                                              'feature_17', 'feature_18',
                                              'featur...
                  ExtraTreesClassifier(bootstrap=False, ccp_alpha=0.0,
                                       class_weight=None, criterion='gin
i',
                                       max_depth=None, max_features='sqr
t',
                                       max_leaf_nodes=None, max_samples=No
ne,
                                       min_impurity_decrease=0.0,
                                       min_samples_leaf=1, min_samples_spl
it=2,
                                       min_weight_fraction_leaf=0.0,
                                       n_estimators=100, n_jobs=-1,
                                       oob_score=False, random_state=7152,
                                       verbose=0, warm_start=False))],
          verbose=False),
 'PyCaret_et_Model.pkl')
```

Type *Markdown* and LaTeX: α^2

Prediction By RandomForest

In [ ]:

```
pre_rf =rf.predict(new_test_df.dropna()) # without nan
```

In [ ]:

```
pre_rf.shape[0]
```

Out[107]:

```
41614
```

In [ ]:

```
pre_rf[pre_rf >0.49].shape[0]
```

Out[108]:

```
28046
```

In [ ]:

```
test_score_rf=(pre_rf[pre_rf >0.49].shape[0] /pre_rf.shape[0]) * 100
```

In [ ]:

```
test_score_rf
```

Out[121]:

```
67.3955880232614
```

In [ ]:

```
# Save Model
with open('RF_model.pkl', 'wb') as file:
    pickle.dump(rf, file)
```