# Assignment Submission

- Sathish Madhiyalagan

## Review Column Descriptions :

Before proceeding with data cleaning, it's essential to thoroughly review the column descriptions provided in the dataset documentation or data dictionary. Initially, we aim to gain a clear understanding of all columns. To achieve this, I spent considerable time reviewing the 'columns_description.csv' file. Finally, I obtained a clear understanding of the data.

## Importing Essential Libraries :

**import pandas as pd**: Import pandas library for data manipulation, assigning alias pd.

**import numpy as np**: Import NumPy library for numerical computing, assigning alias np.

**import matplotlib.pyplot as plt**: Import matplotlib's pyplot module for plotting, assigning alias plt.

**import seaborn as sns**: Import seaborn library for statistical visualization, assigning alias sns.

**%matplotlib inline**: Enable inline plotting for matplotlib in Jupyter Notebook or JupyterLab.

## Objective :

The objective of this data cleaning and analysis project is to improve the accuracy of predicting loan defaulters while minimizing the risk of granting loans to non-defaulters.

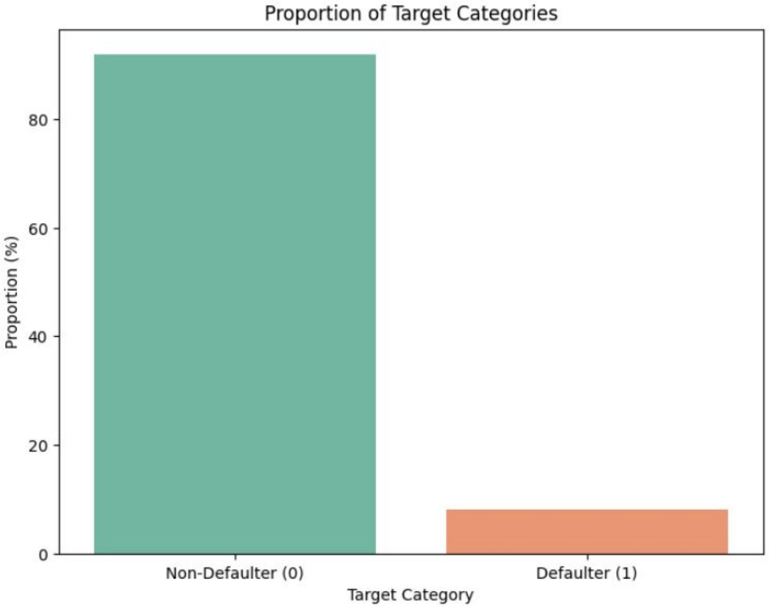
In our dataset, the target variable (TARGET) represents whether a loan applicant defaulted on their loan (1) or not (0).

**Initially :**

So this is goal

We want to improve

Defaulter



## Initiating Data Exploration :

After obtaining a clear understanding of the column descriptions, the next step is to examine the **shape** of our dataset to understand its dimensions. We then utilize the **'columns'** attribute to list all available columns, followed by **'info'** to gather essential information about the dataset, such as data types and missing values. Employing **'describe'** helps us gain statistical insights into numerical columns, while **'head'** and **'tail'** enable us to preview the initial and final rows of our dataset, respectively. These preliminary steps lay the groundwork for further data exploration and cleaning processes.

Continuing from our initial exploration, we proceed to assess data clarity using '**.value_counts**()', '**.isnull**()**.sum**()', and '**.dtype**()'. These functions provide deeper insights into variable distributions, missing values, and data types, facilitating further data refinement and preparation.

## Missing Values :

Identifying missing values is crucial for data quality. To address this, we plan to drop columns with missing values exceeding 40%. This strategic approach ensures the preservation of data integrity while mitigating the impact of incomplete information on our analysis.( both dat set application and previous application )

## Before :

Application Data Size : (307511, 122)

Previous Application Data Size : (1670214, 37)

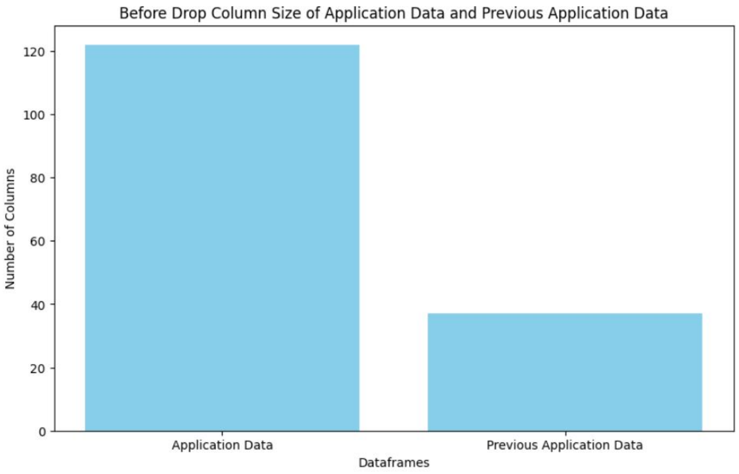
Columns Description : (160, 5)

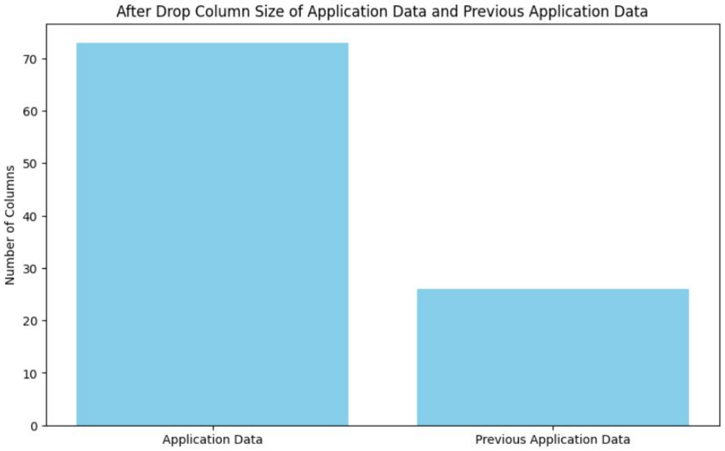
## After Drop:

Application Data Size : (307511, 73)

Previous Application Data Size : (1670214,26)

Columns Description : (160, 5)



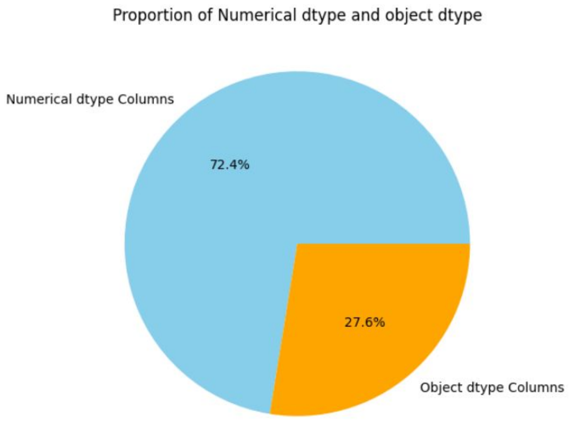


## Data Merge :

We performed a left join operation using the pandas pd.merge() function to merge the 'Application Data' dataframe (app_DF) with the 'Previous Application Data' dataframe (pre_DF) based on the common column 'SK_ID_CURR', resulting in a merged dataframe (merged_df) that retains all records from the 'Application Data' dataframe while incorporating additional information from the 'Previous Application Data' dataframe where available.

So now we have more row like (**1430155, 98**) shape.

After applying the describe() function to the merged dataframe, we identified the presence of **71 numerical** columns in the dataset.so balance 27 columns Object Dtype in this merged dataframe.



## Data Correction :

- ### Negative Value :

  - It iterates through these numeric columns to find those containing negative values.

  - It converts the negative values to positive for the identified columns (pyhton have **abs()** method ).

  - It confirms whether all negative values have been successfully converted to positive.

### Before :

merged_df["DAYS_BIRTH"]

```
0        -9461
1       -16765
2       -16765
3       -16765
4       -19046….
```

### After:

```
0        9461
1        16765
2        16765
3        16765
4       -19046 ….
```

This columns have negative value finally converted.

['DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH', 'DAYS_LAST_PHONE_CHANGE', 'DAYS_DECISION', 'SELLERPLACE_AREA']

## Categorical Columns to Integer Values :

In this dataframe, whenever categorical columns are identified using the value_counts() method and their count is below 15, it is categorized as a categorical variable. Once identified, these categorical columns create new columns and are replaced with unique numeric values.

In this **merged_df["NAME_CONTRACT_TYPE_x"].value_counts()**

### Before :

```
NAME_CONTRACT_TYPE_x
Cash loans        1320679
Revolving loans    109476
Name: count, dtype: int64
```

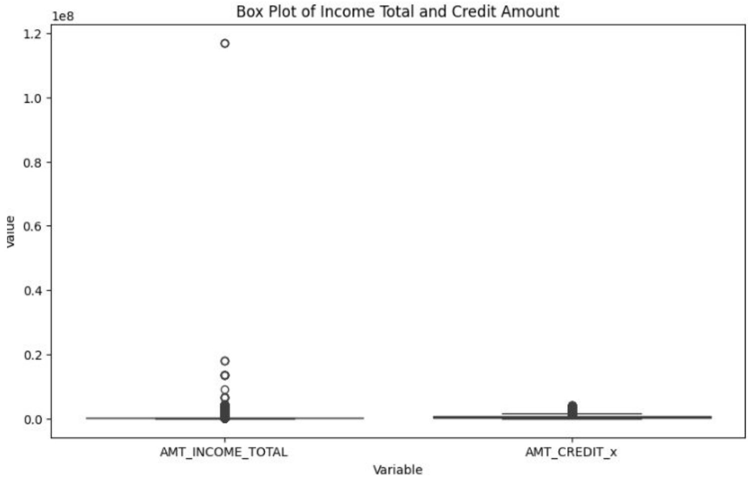
### After (New Columns):

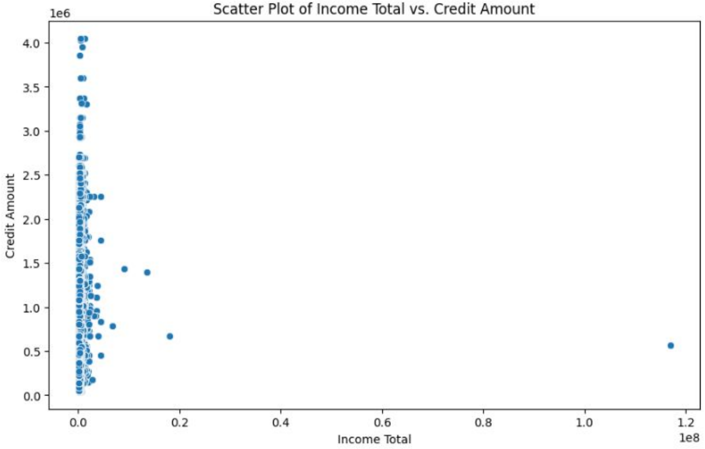
```
NAME_CONTRACT_TYPE_xNumerical
0    1320679
1     109476
Name: count, dtype: int64
```

## Handling Outliers :

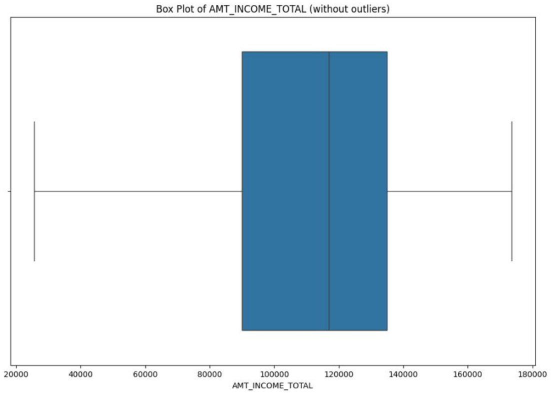
Outliers are observations in data that significantly deviate from the rest of the dataset. In financial analysis, outliers can distort statistical measures and affect the performance of predictive models. Therefore, it's crucial to identify and handle outliers appropriately to ensure the integrity and accuracy of the analysis.

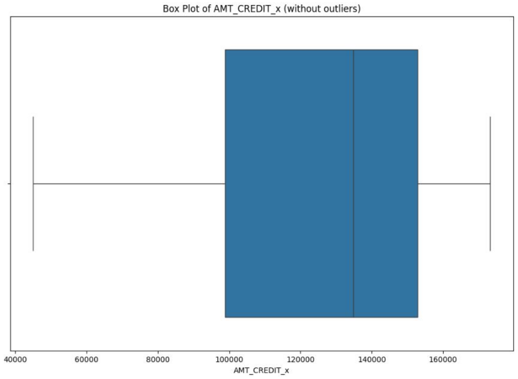
Before Removing Outliers :



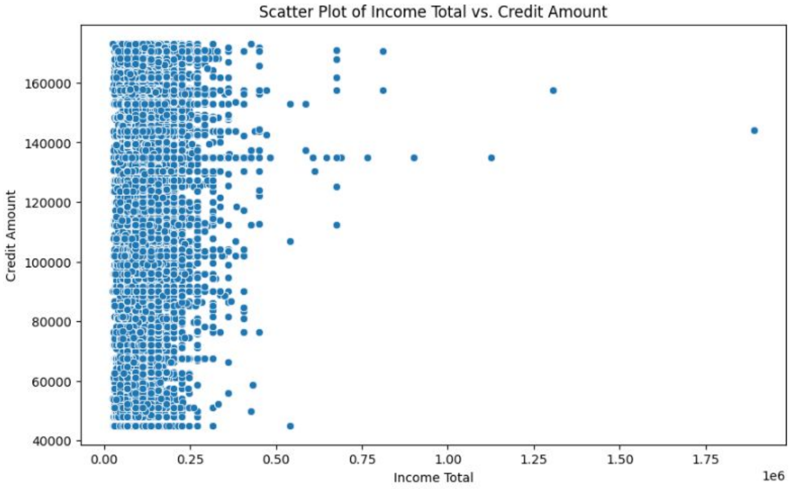


## After Removing Outliers :



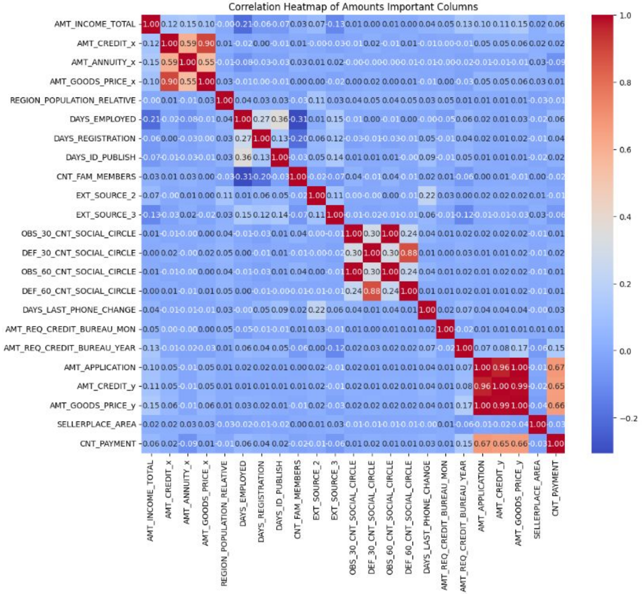


Here we see Outliers  are removed.



# Exploring Correlations :

The final dataframe depicting correlations among financial columns provides a comprehensive overview of the dataset, offering valuable insights into the interrelationships among various financial metrics. Based on this analysis, the decision to proceed with the default loan option appears to be well-founded and supported by the data.

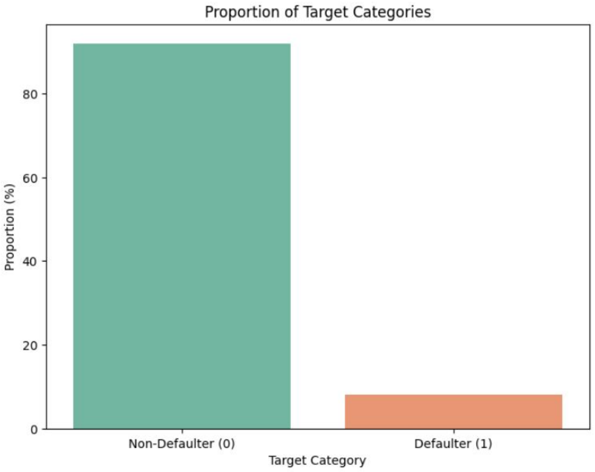


In this dataset, the majority of columns exhibit a positive correlation.

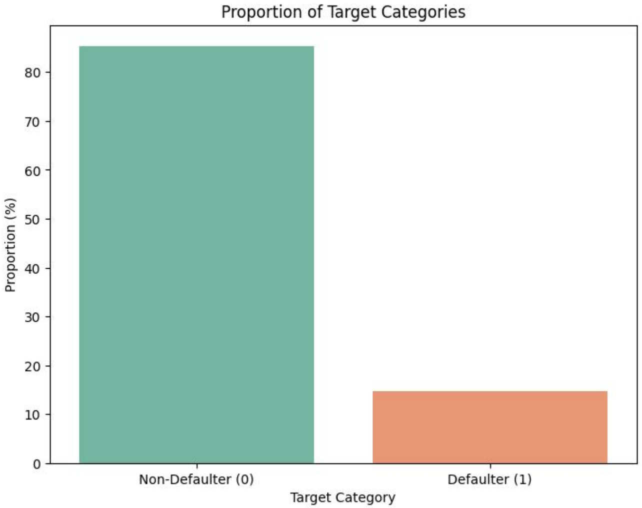
## Conclustion:

After cleaning the "TARGET" column in app_DF, the proportion of class 0 decreased from 91.93% to 85.35%, while the proportion of class 1 increased from 8.07% to 14.65%. This change indicates a more balanced distribution between the two classes, enhancing the dataset's representativeness. The cleaning process likely involved addressing missing values, outliers, or employing class rebalancing techniques.

Before Cleaning :



After Cleaning :


Proportion of Target Categories
Proportion (%)
Non-Defaulter (0)
Defaulter (1)
Target Category