Task:

A white background with black text

Description automatically generated

Initial Data Exploration:

At a glance:

By simply taking a look at some basic summary information about the data, we can identify various issues that need to be addressed about the data, before it is ready to be trained using a model.

**A screenshot of a computer screen

Description automatically generated**

As highlighted, there are various issues about the data at hand such as:

1. Many columns have certain missing values. The dataset has 1,00,000 total rows but the number of non-null counts in columns like Name, Monthly salary, Loan Type, etc. are less than that. We need to address that.
2. The datatype for numerical columns like ‘Age’, ‘Annual Income’ should not be *object*. We need to convert it to a numerical datatype like floar or int, whichever is more suitable.

There are various issues that could arise if we don’t do this. For instance, if we calculate the summary statistics of the dataframe using the ‘describe’ function in pandas, we will not be able to see the stats for columns like Age, Income etc, despite the fact that they are numerical. We will not know then if there are unusual values within these columns or not. In the graph below (drawn using Tableau), we can see that the average age for all three classes of credit score is well over 100 years. This is certainly not consistent with expected values and indicates that outliers exist within the ‘Age’ column.

***Note: This graph was made possible by Tableau which could identify Age as a numerical column and ignored the ‘\_’ (underscore) actually present in some instances in the data.***

**A graph of a number of people

Description automatically generated with medium confidence**

**Age Column**

Upon a brief observation of the data, we notice that there are misplaced underscores ‘\_’ occasionally, the datatype is ‘object’ and some absurdly high values of age (5000) are present. Hence, we will fix this by

1. Eliminating the underscore
2. Converting the datatype to integer
3. Replacing ‘abnormal’ age values (here assumed to be under 10 and over 70) by corresponding age values of other columns with same customer ID and a ‘normal’ age value.

Before Handling Age Column:

A screenshot of a table

Description automatically generated

After:

A screenshot of a computer

Description automatically generated

**Name Column**

Before Handling the Name missing value:

A screenshot of a computer

Description automatically generated

After Handling the Name missing value:

A screenshot of a computer code

Description automatically generated

Note: There’s actually no need to handle missing values in columns such as ‘Name’ since we will eventually drop it during the model training stages. However, this data presents us with a unique opportunity to handle these values by using the corresponding *customer-id* associated with every name, and hence we have implemented it. Using such techniques can help us learn different ways in which missing values can be handled depending on the data - apart from traditional methods like dropping values of interpolation.

**Occupation:**

Upon observing various plots in Tableau, we noticed that in the occupation vs target column (credit score) graph, there was an anomaly. An occupation that just had ‘\_\_\_\_\_\_\_\_\_’ mentioned in it.

A blue and white bar graph

Description automatically generated

This was evidenced by the code as well.

A screenshot of a computer

Description automatically generated

Similar to above techniques, we can simply replace occupation of '\_\_\_\_\_\_' with the correct occupation of the applicant by looking up the corresponding customer\_id

A screenshot of a computer

Description automatically generated

Important thing to note here is that: even though all seemed fine about the occupation column in the initial *data.info* part (no missing values, datatype was categorical as expected), there were certain issues with the data. Unlike the Name column which couldn’t have any influence on our target column, the occupation might be an important feature and thereby needs to be cleaned properly. This goes to show that data cleaning should be done carefully and scrupulously.

**Annual Income:**

Similar to previous cases, the annual income column has two issues. A. ‘\_’ present on certain intervals, B. Object datatype, C. Presence of outliers.

After handling underscores and converting the datatype, we handled the outliers by using the correct annual income from the corresponding customer\_id, as shown in previous cases.

Before:

A table with numbers and letters

Description automatically generated

After:

A screenshot of a computer

Description automatically generated

Box Plot of the Annual Income now looks like this:

A screenshot of a graph

Description automatically generated

**Monthly In hand Salary:**

As we saw in the data.info( ) section, there are missing values in this columns. Precisely, 15002 values are missing. We first assign a dummy value, ‘-1’ to all these missing values and then replace it with corresponding salary values of its customer id. The same process has been shown below:

A screenshot of a computer code

Description automatically generated

Before: A table with numbers and letters

Description automatically generated

After:

A screenshot of a computer code

Description automatically generated

A close-up of a credit card

Description automatically generated

**Number of bank Accounts:**

Entries in this dataset consist of people with an usual amount of bank accounts.

For example, a person in the list below holds 1414 according to one entry and 8 according to all ither entries. This is clearly an error which would massively impact the model if not treated.

A table with numbers and letters

Description automatically generated

We will thus adjust all entries where bank accounts are more than 11 or less than 0 (summary data shows there are negative values too).

A screenshot of a computer code

Description automatically generated

After this, the values of the previous customer have now been adjusted as shown:

A screenshot of a computer

Description automatically generated

Similar processing techniques were applied to remaining columns like Number of Credit cards, Interest rates, Number of loans, etc.

Since our dataset had multiple entries for each customer ID and almost the same demographic and income information across entries for customers, we used a custom splitter to group data by ‘Customer\_ID’ and did a stratified split based on the target class.

We have applied a standard scaler to the dataset as models like Logistic Regression and SVM require this preprocessing step

Multiclass One-vs-One Logistic Regression model with default parameters:

A screenshot of a computer screen

Description automatically generated

Multiclass One-vs-One SVM Primal model:

A screenshot of a black screen

Description automatically generated

Multiclass One-vs-One SVM Random Forest model:

A screenshot of a computer screen

Description automatically generated