Predicting Loan Defaulters - Rushikesh Kishor Khankar

Project 2

DESCRIPTION

Data Analysis is the process of creating a story using the data for easy and effective communication. It mostly utilizes visualization methods like plots, charts, and tables to convey what the data holds beyond the formal modeling or hypothesis testing task.

Domain: Finance

Read the information given below and also refer to the data dictionary provided separately in an excel file to build your understanding.

Problem Statement

Financial institutions incur significant losses due to the default of vehicle loans. This has led to the tightening up of vehicle loan underwriting and increased vehicle loan rejection rates. The need for a better credit risk scoring model is also raised by these institutions. This warrants a study to estimate the determinants of vehicle loan default.

There is 1 dataset data that have 41 attributes. You are required to determine and examine factors that affected the ratio of vehicle loan defaulters. Also, use the findings to create a model to predict the potential defaulters.

Approach:

1. Data Preliminary analysis:

Perform preliminary data inspection and report the findings as to the structure of the data, missing values, duplicates, etc.

Variable names in the data may not be in accordance with the identifier naming in Python. Change the variable names accordingly.

The presented data might also contain missing values, therefore exploration will also lead to devising strategies to fill in the missing values. Devise strategies to do so whilst exploring the data.

1. Performing EDA:

Provide the statistical description of the quantitative data variables

How is the target variable distributed overall?

Study the distribution of the target variable across the various categories such as branch, city, state, branch, supplier, manufacturer, etc. What are the different employment types given in the data? Can a strategy be developed to fill in the missing values (if any)? Use pie charts to express how different types of employment defines defaulter and non-defaulters.

Has age got something to do with defaulting? What is the distribution of age w.r.t. to defaulters and non-defaulters?

What type of ID was presented by most of the customers as proof?

Explain the factors in the data that may have an effect on ratings e.g. No. of cuisines, cost, delivery option, etc.

1. Performing EDA and Modelling:

Provide the statistical description of the quantitative data variables

How is the target variable distributed overall?

Study the distribution of the target variable across the various categories such as branch, city, state, branch, supplier, manufacturer, etc.

What are the different employment types given in the data? Can a strategy be developed to fill in the missing values (if any)? Use pie charts to express how different types of employment defines defaulter and non-defaulters.

Has age got something to do with defaulting? What is the distribution of age w.r.t. to defaulters and non-defaulters?

What type of ID was presented by most of the customers as proof?

Explain the factors in the data that may have an effect on ratings e.g. No. of cuisines, cost, delivery option, etc.

Project Task: Week 1

Importing, Understanding, and Inspecting Data :

Perform preliminary data inspection and report the findings as the structure of the data, missing values, duplicates, etc.

In [1]:

```
import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
data = pd.read_excel('data.xlsx')
```

In [3]:
```
data.head()
```

Out[3]:

	UniqueID	disbursed_amount	asset_cost	ltv	branch_id	supplier_id	manufacturer_ic
0	420825	50578	58400	89.55	67	22807	45
1	417566	53278	61360	89.63	67	22807	45
2	539055	52378	60300	88.39	67	22807	45
3	529269	46349	61500	76.42	67	22807	45
4	563215	43594	78256	57.50	67	22744	86

5 rows × 41 columns

In [4]:
```
data.shape
```

Out[4]:
```
(233154, 41)
```

In [5]:
```
data.isnull().sum()
```

Out[5]:
```
UniqueID                                  0
disbursed_amount                          0
asset_cost                                0
ltv                                       0
branch_id                                 0
supplier_id                               0
manufacturer_id                           0
Current_pincode_ID                        0
Date.of.Birth                             0
Employment.Type                        7661
DisbursalDate                             0
State_ID                                  0
Employee_code_ID                          0
MobileNo_Avl_Flag                         0
Aadhar_flag                               0
PAN_flag                                  0
VoterID_flag                              0
Driving_flag                              0
Passport_flag                             0
PERFORM_CNS.SCORE                         0
PERFORM_CNS.SCORE.DESCRIPTION             0
PRI.NO.OF.ACCTS                           0
PRI.ACTIVE.ACCTS                          0
PRI.OVERDUE.ACCTS                         0
PRI.CURRENT.BALANCE                       0
PRI.SANCTIONED.AMOUNT                     0
PRI.DISBURSED.AMOUNT                      0
SEC.NO.OF.ACCTS                           0
SEC.ACTIVE.ACCTS                          0
SEC.OVERDUE.ACCTS                         0
SEC.CURRENT.BALANCE                       0
SEC.SANCTIONED.AMOUNT                     0
SEC.DISBURSED.AMOUNT                      0
PRIMARY.INSTAL.AMT                        0
SEC.INSTAL.AMT                            0
NEW.ACCTS.IN.LAST.SIX.MONTHS              0
DELINQUENT.ACCTS.IN.LAST.SIX.MONTHS       0
AVERAGE.ACCT.AGE                          0
CREDIT.HISTORY.LENGTH                     0
NO.OF_INQUIRIES                           0
loan_default                              0
dtype: int64
```

In [6]:
```
data.duplicated().any()
```

Out[6]:
```
False
```

Variable names in the data may not be in accordance with the identifier naming in Python so, change the variable names accordingly

In [7]:
```
data.columns
```

Out[7]:
```
Index(['UniqueID', 'disbursed_amount', 'asset_cost', 'ltv', 'branch_id',
       'supplier_id', 'manufacturer_id', 'Current_pincode_ID', 'Date.of.Bir
th',
       'Employment.Type', 'DisbursalDate', 'State_ID', 'Employee_code_ID',
       'MobileNo_Avl_Flag', 'Aadhar_flag', 'PAN_flag', 'VoterID_flag',
       'Driving_flag', 'Passport_flag', 'PERFORM_CNS.SCORE',
       'PERFORM_CNS.SCORE.DESCRIPTION', 'PRI.NO.OF.ACCTS', 'PRI.ACTIVE.ACCT
S',
       'PRI.OVERDUE.ACCTS', 'PRI.CURRENT.BALANCE', 'PRI.SANCTIONED.AMOUNT',
       'PRI.DISBURSED.AMOUNT', 'SEC.NO.OF.ACCTS', 'SEC.ACTIVE.ACCTS',
       'SEC.OVERDUE.ACCTS', 'SEC.CURRENT.BALANCE', 'SEC.SANCTIONED.AMOUNT',
       'SEC.DISBURSED.AMOUNT', 'PRIMARY.INSTAL.AMT', 'SEC.INSTAL.AMT',
       'NEW.ACCTS.IN.LAST.SIX.MONTHS', 'DELINQUENT.ACCTS.IN.LAST.SIX.MONTHS
',
       'AVERAGE.ACCT.AGE', 'CREDIT.HISTORY.LENGTH', 'NO.OF_INQUIRIES',
       'loan_default'],
      dtype='object')
```

In [8]:
```
data.columns = ['Unique_ID', 'Disbursed_Amount', 'Asset_Cost', 'ltv', 'Bran
         'Supplier_ID', 'Manufacturer_ID', 'Current_Pincode_ID', 'Date_Of_Bi
         'Employment_Type', 'Disbursal_Date', 'State_ID', 'Employee_Code_ID'
         'MobileNo_Avl_Flag', 'Aadhar_Flag', 'PAN_Flag', 'VoterID_Flag',
         'Driving_Flag', 'Passport_Flag', 'Perform_CNS_Score',
         'Perform_CNS_Score_Description', 'PRI_NO_OF_ACCTS', 'PRI_ACTIVE_ACC
         'PRI_OVERDUE_ACCTS', 'PRI_CURRENT_BALANCE', 'PRI_SANCTIONED_AMOUNT'
         'PRI_DISBURSED_AMOUNT', 'SEC_NO_OF_ACCTS', 'SEC_ACTIVE_ACCTS',
         'SEC_OVERDUE_ACCTS', 'SEC_CURRENT_BALANCE', 'SEC_SANCTIONED_AMOUNT'
         'SEC_DISBURSED_AMOUNT', 'PRIMARY_INSTAL_AMT', 'SEC_INSTAL_AMT',
         'NEW_ACCTS_IN_LAST_SIX_MONTHS', 'DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS
         'AVERAGE_ACCT_AGE', 'CREDIT_HISTORY_LENGTH', 'NO_OF_INQUIRIES',
         'Loan_Default']
```

In [9]:
```
data.columns.astype('object')
```

Out[9]:
```
Index(['Unique_ID', 'Disbursed_Amount', 'Asset_Cost', 'ltv', 'Branch_ID',
       'Supplier_ID', 'Manufacturer_ID', 'Current_Pincode_ID', 'Date_Of_Bir
th',
       'Employment_Type', 'Disbursal_Date', 'State_ID', 'Employee_Code_ID',
       'MobileNo_Avl_Flag', 'Aadhar_Flag', 'PAN_Flag', 'VoterID_Flag',
       'Driving_Flag', 'Passport_Flag', 'Perform_CNS_Score',
       'Perform_CNS_Score_Description', 'PRI_NO_OF_ACCTS', 'PRI_ACTIVE_ACCT
S',
       'PRI_OVERDUE_ACCTS', 'PRI_CURRENT_BALANCE', 'PRI_SANCTIONED_AMOUNT',
       'PRI_DISBURSED_AMOUNT', 'SEC_NO_OF_ACCTS', 'SEC_ACTIVE_ACCTS',
       'SEC_OVERDUE_ACCTS', 'SEC_CURRENT_BALANCE', 'SEC_SANCTIONED_AMOUNT',
       'SEC_DISBURSED_AMOUNT', 'PRIMARY_INSTAL_AMT', 'SEC_INSTAL_AMT',
       'NEW_ACCTS_IN_LAST_SIX_MONTHS', 'DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS
',
       'AVERAGE_ACCT_AGE', 'CREDIT_HISTORY_LENGTH', 'NO_OF_INQUIRIES',
       'Loan_Default'],
      dtype='object')
```

In [10]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 233154 entries, 0 to 233153
Data columns (total 41 columns):
 #   Column                               Non-Null Count   Dtype
---  ------                               --------------   -----
 0   Unique_ID                            233154 non-null  int64
 1   Disbursed_Amount                     233154 non-null  int64
 2   Asset_Cost                           233154 non-null  int64
 3   ltv                                  233154 non-null  float64
 4   Branch_ID                            233154 non-null  int64
 5   Supplier_ID                          233154 non-null  int64
 6   Manufacturer_ID                      233154 non-null  int64
 7   Current_Pincode_ID                   233154 non-null  int64
 8   Date_Of_Birth                        233154 non-null  datetime64[ns]
 9   Employment_Type                      225493 non-null  object
 10  Disbursal_Date                       233154 non-null  datetime64[ns]
 11  State_ID                             233154 non-null  int64
 12  Employee_Code_ID                     233154 non-null  int64
 13  MobileNo_Avl_Flag                    233154 non-null  int64
 14  Aadhar_Flag                          233154 non-null  int64
 15  PAN_Flag                             233154 non-null  int64
 16  VoterID_Flag                         233154 non-null  int64
 17  Driving_Flag                         233154 non-null  int64
 18  Passport_Flag                        233154 non-null  int64
 19  Perform_CNS_Score                    233154 non-null  int64
 20  Perform_CNS_Score_Description        233154 non-null  object
 21  PRI_NO_OF_ACCTS                      233154 non-null  int64
 22  PRI_ACTIVE_ACCTS                     233154 non-null  int64
 23  PRI_OVERDUE_ACCTS                    233154 non-null  int64
 24  PRI_CURRENT_BALANCE                  233154 non-null  int64
 25  PRI_SANCTIONED_AMOUNT                233154 non-null  int64
 26  PRI_DISBURSED_AMOUNT                 233154 non-null  int64
 27  SEC_NO_OF_ACCTS                      233154 non-null  int64
 28  SEC_ACTIVE_ACCTS                     233154 non-null  int64
 29  SEC_OVERDUE_ACCTS                    233154 non-null  int64
 30  SEC_CURRENT_BALANCE                  233154 non-null  int64
 31  SEC_SANCTIONED_AMOUNT                233154 non-null  int64
 32  SEC_DISBURSED_AMOUNT                 233154 non-null  int64
 33  PRIMARY_INSTAL_AMT                   233154 non-null  int64
 34  SEC_INSTAL_AMT                       233154 non-null  int64
 35  NEW_ACCTS_IN_LAST_SIX_MONTHS         233154 non-null  int64
 36  DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS  233154 non-null  int64
 37  AVERAGE_ACCT_AGE                     233154 non-null  object
 38  CREDIT_HISTORY_LENGTH                233154 non-null  object
 39  NO_OF_INQUIRIES                      233154 non-null  int64
 40  Loan_Default                         233154 non-null  int64
dtypes: datetime64[ns](2), float64(1), int64(34), object(4)
memory usage: 72.9+ MB
```

The presented data might also contain some missing values therefore, exploration will also lead to devising strategies to fill in the missing values while exploring the data

In [11]:

```
data['Employment_Type'].value_counts()
```

Out[11]:
```
Self employed    127635
Salaried          97858
Name: Employment_Type, dtype: int64
```

In [12]:
```
data['Employment_Type'].fillna("Self employed", inplace = True)
```

In [164…
```
data['Employment_Type'].to_excel('Employment_Typesr.xlsx')
```

In [13]:
```
data.isnull()
```

Out[13]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufa
0	False	False	False	False	False	False	
1	False	False	False	False	False	False	
2	False	False	False	False	False	False	
3	False	False	False	False	False	False	
4	False	False	False	False	False	False	
...	...	...	...	...	...	...	
233149	False	False	False	False	False	False	
233150	False	False	False	False	False	False	
233151	False	False	False	False	False	False	
233152	False	False	False	False	False	False	
233153	False	False	False	False	False	False	

233154 rows × 41 columns

In [14]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 233154 entries, 0 to 233153
Data columns (total 41 columns):
 #   Column                               Non-Null Count   Dtype
---  ------                               --------------   -----
 0   Unique_ID                            233154 non-null  int64
 1   Disbursed_Amount                     233154 non-null  int64
 2   Asset_Cost                           233154 non-null  int64
 3   ltv                                  233154 non-null  float64
 4   Branch_ID                            233154 non-null  int64
 5   Supplier_ID                          233154 non-null  int64
 6   Manufacturer_ID                      233154 non-null  int64
 7   Current_Pincode_ID                   233154 non-null  int64
 8   Date_Of_Birth                        233154 non-null  datetime64[ns]
 9   Employment_Type                      233154 non-null  object
 10  Disbursal_Date                       233154 non-null  datetime64[ns]
 11  State_ID                             233154 non-null  int64
 12  Employee_Code_ID                     233154 non-null  int64
 13  MobileNo_Avl_Flag                    233154 non-null  int64
 14  Aadhar_Flag                          233154 non-null  int64
 15  PAN_Flag                             233154 non-null  int64
 16  VoterID_Flag                         233154 non-null  int64
 17  Driving_Flag                         233154 non-null  int64
 18  Passport_Flag                        233154 non-null  int64
 19  Perform_CNS_Score                    233154 non-null  int64
 20  Perform_CNS_Score_Description        233154 non-null  object
 21  PRI_NO_OF_ACCTS                      233154 non-null  int64
 22  PRI_ACTIVE_ACCTS                     233154 non-null  int64
 23  PRI_OVERDUE_ACCTS                    233154 non-null  int64
 24  PRI_CURRENT_BALANCE                  233154 non-null  int64
 25  PRI_SANCTIONED_AMOUNT                233154 non-null  int64
 26  PRI_DISBURSED_AMOUNT                 233154 non-null  int64
 27  SEC_NO_OF_ACCTS                      233154 non-null  int64
 28  SEC_ACTIVE_ACCTS                     233154 non-null  int64
 29  SEC_OVERDUE_ACCTS                    233154 non-null  int64
 30  SEC_CURRENT_BALANCE                  233154 non-null  int64
 31  SEC_SANCTIONED_AMOUNT                233154 non-null  int64
 32  SEC_DISBURSED_AMOUNT                 233154 non-null  int64
 33  PRIMARY_INSTAL_AMT                   233154 non-null  int64
 34  SEC_INSTAL_AMT                       233154 non-null  int64
 35  NEW_ACCTS_IN_LAST_SIX_MONTHS         233154 non-null  int64
 36  DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS  233154 non-null  int64
 37  AVERAGE_ACCT_AGE                     233154 non-null  object
 38  CREDIT_HISTORY_LENGTH                233154 non-null  object
 39  NO_OF_INQUIRIES                      233154 non-null  int64
 40  Loan_Default                         233154 non-null  int64
dtypes: datetime64[ns](2), float64(1), int64(34), object(4)
memory usage: 72.9+ MB
```

Performing EDA:

Provide the statistical description of the quantitative data variables

In [15]:

```
data.describe()
```

Out[15]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID
count	233154.000000	233154.000000	2.331540e+05	233154.000000	233154.000000
mean	535917.573376	54356.993528	7.586507e+04	74.746530	72.936094
std	68315.693711	12971.314171	1.894478e+04	11.456636	69.834995
min	417428.000000	13320.000000	3.700000e+04	10.030000	1.000000
25%	476786.250000	47145.000000	6.571700e+04	68.880000	14.000000
50%	535978.500000	53803.000000	7.094600e+04	76.800000	61.000000
75%	595039.750000	60413.000000	7.920175e+04	83.670000	130.000000
max	671084.000000	990572.000000	1.628992e+06	95.000000	261.000000

8 rows × 35 columns

Explain how is the target variable distributed overall

Study the distribution of the target variable across various categories like branch, city, state, branch, supplier, manufacturer, etc.

In [16]:

```
data.columns
```

Out[16]:

```
Index(['Unique_ID', 'Disbursed_Amount', 'Asset_Cost', 'ltv', 'Branch_ID',
       'Supplier_ID', 'Manufacturer_ID', 'Current_Pincode_ID', 'Date_Of_Bir
th',
       'Employment_Type', 'Disbursal_Date', 'State_ID', 'Employee_Code_ID',
       'MobileNo_Avl_Flag', 'Aadhar_Flag', 'PAN_Flag', 'VoterID_Flag',
       'Driving_Flag', 'Passport_Flag', 'Perform_CNS_Score',
       'Perform_CNS_Score_Description', 'PRI_NO_OF_ACCTS', 'PRI_ACTIVE_ACCT
S',
       'PRI_OVERDUE_ACCTS', 'PRI_CURRENT_BALANCE', 'PRI_SANCTIONED_AMOUNT',
       'PRI_DISBURSED_AMOUNT', 'SEC_NO_OF_ACCTS', 'SEC_ACTIVE_ACCTS',
       'SEC_OVERDUE_ACCTS', 'SEC_CURRENT_BALANCE', 'SEC_SANCTIONED_AMOUNT',
       'SEC_DISBURSED_AMOUNT', 'PRIMARY_INSTAL_AMT', 'SEC_INSTAL_AMT',
       'NEW_ACCTS_IN_LAST_SIX_MONTHS', 'DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS
',
       'AVERAGE_ACCT_AGE', 'CREDIT_HISTORY_LENGTH', 'NO_OF_INQUIRIES',
       'Loan_Default'],
      dtype='object')
```

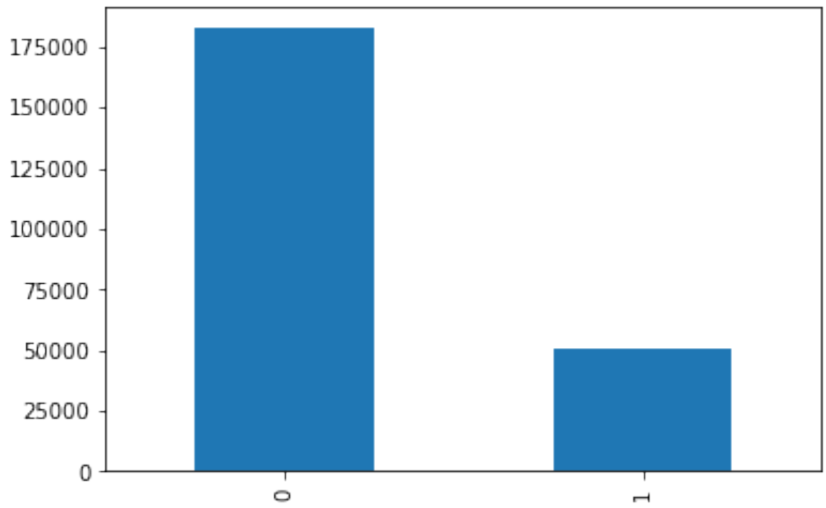
In [17]:

```
import seaborn as sns
import matplotlib as plt
%matplotlib inline
```

In [18]:

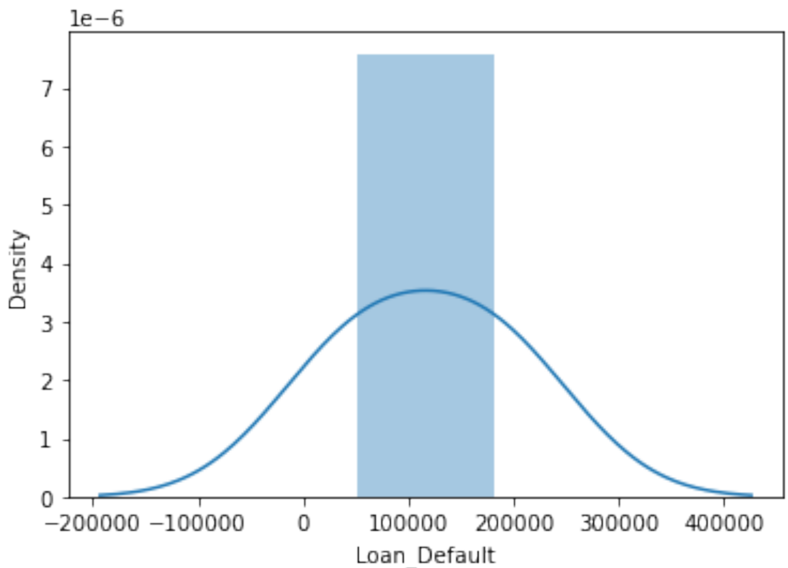
```
data['Loan_Default'].value_counts().plot(kind='bar')
```

Out[18]: <AxesSubplot:>



In [19]:
```
sns.distplot(data.Loan_Default.value_counts())
```

Out[19]: <AxesSubplot:xlabel='Loan_Default', ylabel='Density'>



In [20]:
```
data.columns
```

Out[20]:
```
Index(['Unique_ID', 'Disbursed_Amount', 'Asset_Cost', 'ltv', 'Branch_ID',
       'Supplier_ID', 'Manufacturer_ID', 'Current_Pincode_ID', 'Date_Of_Bir
th',
       'Employment_Type', 'Disbursal_Date', 'State_ID', 'Employee_Code_ID',
       'MobileNo_Avl_Flag', 'Aadhar_Flag', 'PAN_Flag', 'VoterID_Flag',
       'Driving_Flag', 'Passport_Flag', 'Perform_CNS_Score',
       'Perform_CNS_Score_Description', 'PRI_NO_OF_ACCTS', 'PRI_ACTIVE_ACCT
S',
       'PRI_OVERDUE_ACCTS', 'PRI_CURRENT_BALANCE', 'PRI_SANCTIONED_AMOUNT',
       'PRI_DISBURSED_AMOUNT', 'SEC_NO_OF_ACCTS', 'SEC_ACTIVE_ACCTS',
       'SEC_OVERDUE_ACCTS', 'SEC_CURRENT_BALANCE', 'SEC_SANCTIONED_AMOUNT',
       'SEC_DISBURSED_AMOUNT', 'PRIMARY_INSTAL_AMT', 'SEC_INSTAL_AMT',
       'NEW_ACCTS_IN_LAST_SIX_MONTHS', 'DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS
',
       'AVERAGE_ACCT_AGE', 'CREDIT_HISTORY_LENGTH', 'NO_OF_INQUIRIES',
       'Loan_Default'],
      dtype='object')
```

In [21]:
```
from sklearn.preprocessing import LabelEncoder
```

In [22]:
```
Le = LabelEncoder()

data['Branch_ID'] = Le.fit_transform(data['Branch_ID'])

data['Supplier_ID'] = Le.fit_transform(data['Supplier_ID'])

data['Manufacturer_ID'] = Le.fit_transform(data['Manufacturer_ID'])

data['Current_Pincode_ID'] = Le.fit_transform(data['Current_Pincode_ID'])

data['State_ID'] = Le.fit_transform(data['State_ID'])
```

In [23]:
```
data
```

Out[23]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufa
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	
2	539055	52378	60300	88.39	30	1415	
3	529269	46349	61500	76.42	30	1415	
4	563215	43594	78256	57.50	30	1378	
...	...	...	...	...	...	...	
233149	561031	57759	76350	77.28	3	1237	
233150	649600	55009	71200	78.72	57	606	
233151	603445	58513	68000	88.24	55	1775	
233152	442948	22824	40458	61.79	65	387	
233153	545300	35299	72698	52.27	2	133	

233154 rows × 41 columns

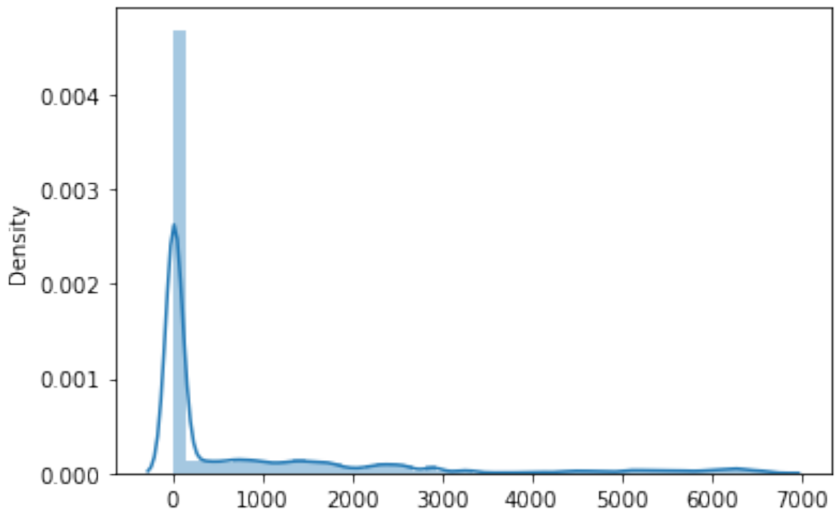
In [24]:

```
x = (data.Branch_ID, data.Supplier_ID, data.Manufacturer_ID, data.Current_I

y = (data.Loan_Default)
```

In [25]:

```
sns.distplot(x)
```

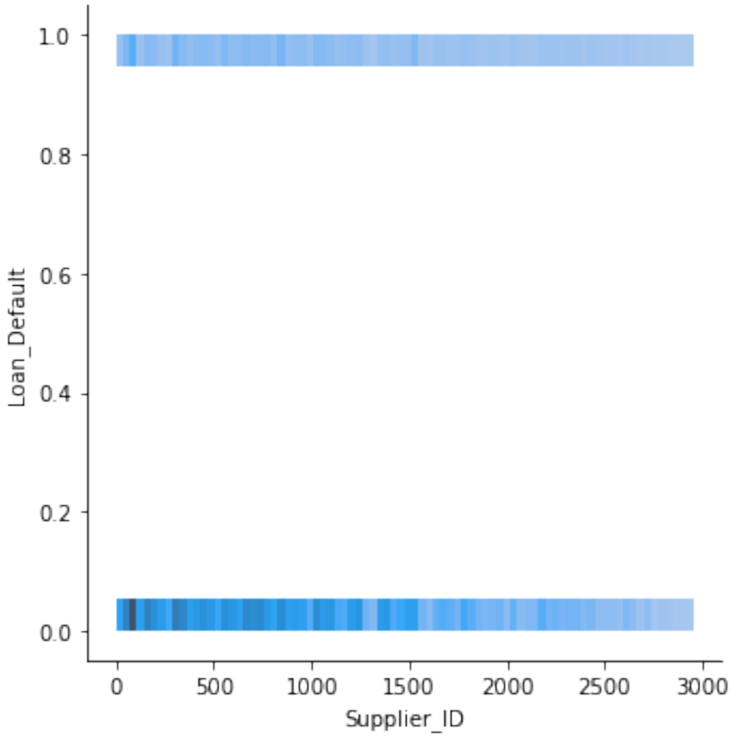
Out[25]: <AxesSubplot:ylabel='Density'>



In [26]:

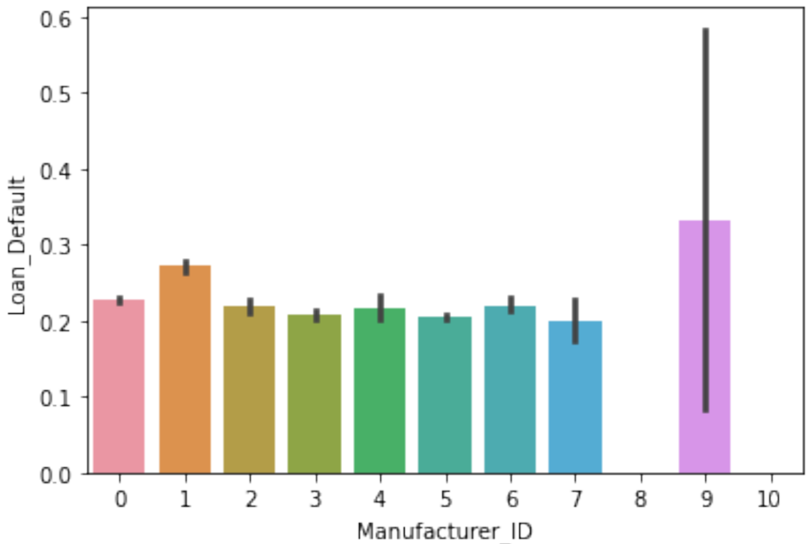
```
sns.displot(x=data.Supplier_ID, y=data.Loan_Default)
```

Out[26]: <seaborn.axisgrid.FacetGrid at 0x7f82c71b9fd0>



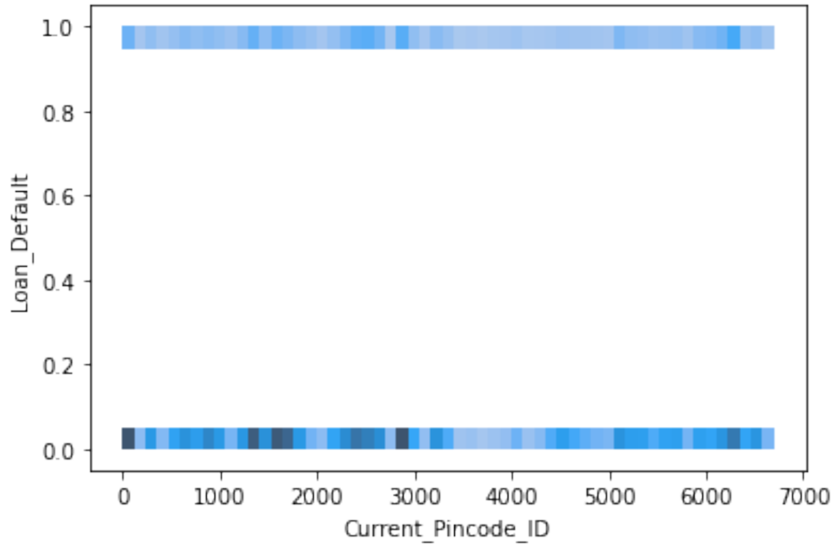
In [27]:
```
sns.barplot(x=data.Manufacturer_ID, y=data.Loan_Default)
```

Out[27]: <AxesSubplot:xlabel='Manufacturer_ID', ylabel='Loan_Default'>

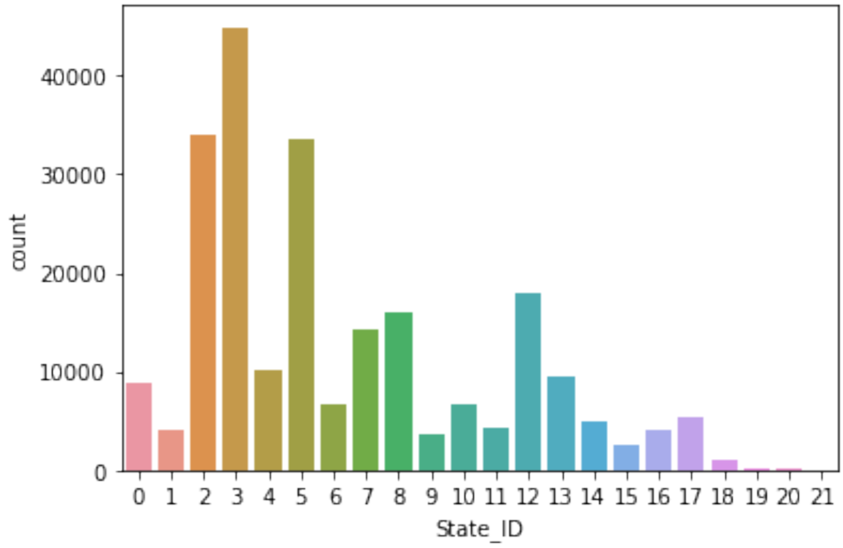


In [28]:
```
sns.histplot(x=data.Current_Pincode_ID, y=data.Loan_Default)
```

Out[28]:
```
<AxesSubplot:xlabel='Current_Pincode_ID', ylabel='Loan_Default'>
```



In [29]:
```
sns.countplot(data['State_ID']);
```



What are the different employment types given in the data? Can a strategy be developed to fill in the missing values (if any)?

In [30]:
```
data['Employment_Type'].unique()
```

Out[30]:
```
array(['Salaried', 'Self employed'], dtype=object)
```

In [31]:
```
data['Employment_Type'].isna().any()
```

Out[31]:
```
False
```

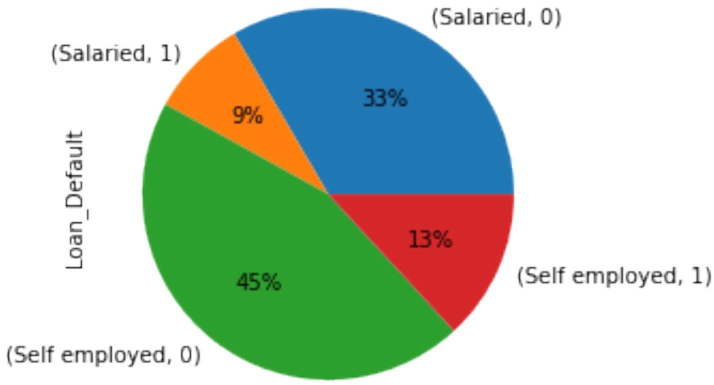
Missing values already treated above with mode

Use pie charts to express how different types of employment defines defaulter and non-defaulters.

In [32]:
```
data['Loan_Default'].groupby(data['Employment_Type']).value_counts()
```

Out[32]:
```
Employment_Type  Loan_Default
Salaried         0                77948
                 1                19910
Self employed    0               104595
                 1                30701
Name: Loan_Default, dtype: int64
```

In [33]:
```
data['Loan_Default'].groupby(data['Employment_Type']).value_counts().plot(l
```



In [34]:
```
data.head(1)
```

Out[34]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	

1 rows × 41 columns

In [35]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 233154 entries, 0 to 233153
Data columns (total 41 columns):
 #   Column                               Non-Null Count   Dtype
---  ------                               --------------   -----
 0   Unique_ID                            233154 non-null  int64
 1   Disbursed_Amount                     233154 non-null  int64
 2   Asset_Cost                           233154 non-null  int64
 3   ltv                                  233154 non-null  float64
 4   Branch_ID                            233154 non-null  int64
 5   Supplier_ID                          233154 non-null  int64
 6   Manufacturer_ID                      233154 non-null  int64
 7   Current_Pincode_ID                   233154 non-null  int64
 8   Date_Of_Birth                        233154 non-null  datetime64[ns]
 9   Employment_Type                      233154 non-null  object
 10  Disbursal_Date                       233154 non-null  datetime64[ns]
 11  State_ID                             233154 non-null  int64
 12  Employee_Code_ID                     233154 non-null  int64
 13  MobileNo_Avl_Flag                    233154 non-null  int64
 14  Aadhar_Flag                          233154 non-null  int64
 15  PAN_Flag                             233154 non-null  int64
 16  VoterID_Flag                         233154 non-null  int64
 17  Driving_Flag                         233154 non-null  int64
 18  Passport_Flag                        233154 non-null  int64
 19  Perform_CNS_Score                    233154 non-null  int64
 20  Perform_CNS_Score_Description        233154 non-null  object
 21  PRI_NO_OF_ACCTS                      233154 non-null  int64
 22  PRI_ACTIVE_ACCTS                     233154 non-null  int64
 23  PRI_OVERDUE_ACCTS                    233154 non-null  int64
 24  PRI_CURRENT_BALANCE                  233154 non-null  int64
 25  PRI_SANCTIONED_AMOUNT                233154 non-null  int64
 26  PRI_DISBURSED_AMOUNT                 233154 non-null  int64
 27  SEC_NO_OF_ACCTS                      233154 non-null  int64
 28  SEC_ACTIVE_ACCTS                     233154 non-null  int64
 29  SEC_OVERDUE_ACCTS                    233154 non-null  int64
 30  SEC_CURRENT_BALANCE                  233154 non-null  int64
 31  SEC_SANCTIONED_AMOUNT                233154 non-null  int64
 32  SEC_DISBURSED_AMOUNT                 233154 non-null  int64
 33  PRIMARY_INSTAL_AMT                   233154 non-null  int64
 34  SEC_INSTAL_AMT                       233154 non-null  int64
 35  NEW_ACCTS_IN_LAST_SIX_MONTHS         233154 non-null  int64
 36  DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS  233154 non-null  int64
 37  AVERAGE_ACCT_AGE                     233154 non-null  object
 38  CREDIT_HISTORY_LENGTH                233154 non-null  object
 39  NO_OF_INQUIRIES                      233154 non-null  int64
 40  Loan_Default                         233154 non-null  int64
dtypes: datetime64[ns](2), float64(1), int64(34), object(4)
memory usage: 72.9+ MB
```

In [36]:

```
data['Date_Of_Birth'].dtype
```

Out[36]:

```
dtype('<M8[ns]')
```

Converting Date_of_birth to Age

In [37]:
```
now = pd.Timestamp('now')

data['Date_Of_Birth'] = pd.to_datetime(data['Date_Of_Birth'], format='%d-%m

data['Date_Of_Birth'] = data['Date_Of_Birth'].where(data['Date_Of_Birth'] <

data['Age'] = (now - data['Date_Of_Birth']).astype('<m8[Y]')
```

In [38]:
```
data.head(2)
```

Out[38]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	

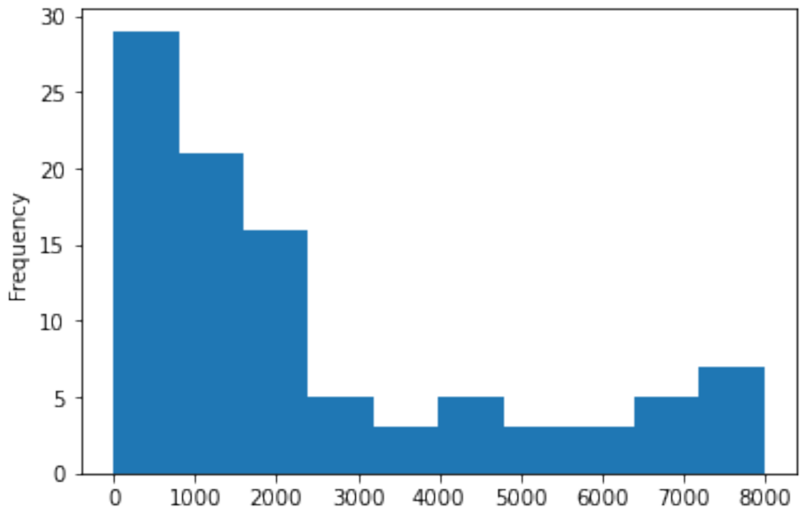
2 rows × 42 columns

Has age got anything to do with defaulting? What is the distribution of age w.r.t. to the defaulters and non-defaulters?

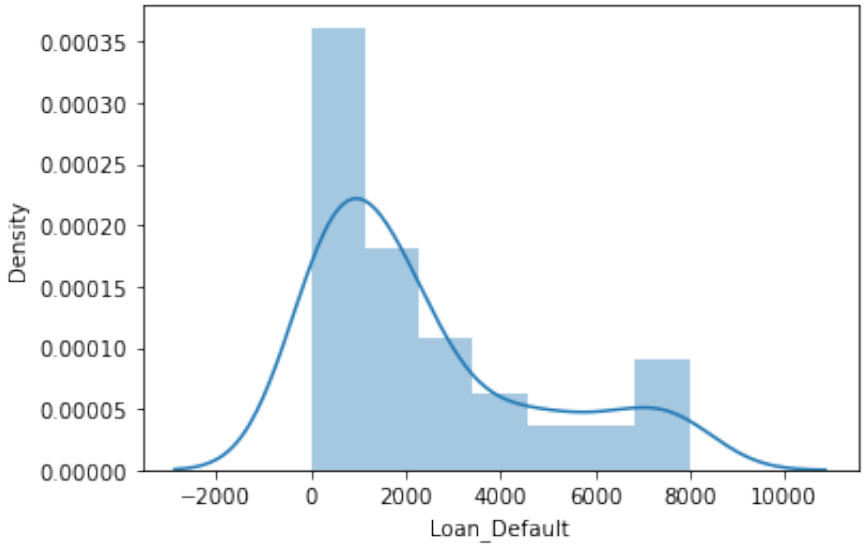
In [39]:
```
data['Loan_Default'].groupby(data['Age']).value_counts()
```

Out[39]:
```
Age   Loan_Default
21.0  0                 66
      1                 20
22.0  0                838
      1                285
23.0  0               1359
                      ...
67.0  0                 95
      1                 17
68.0  0                  5
      1                  1
72.0  0                  1
Name: Loan_Default, Length: 97, dtype: int64
```

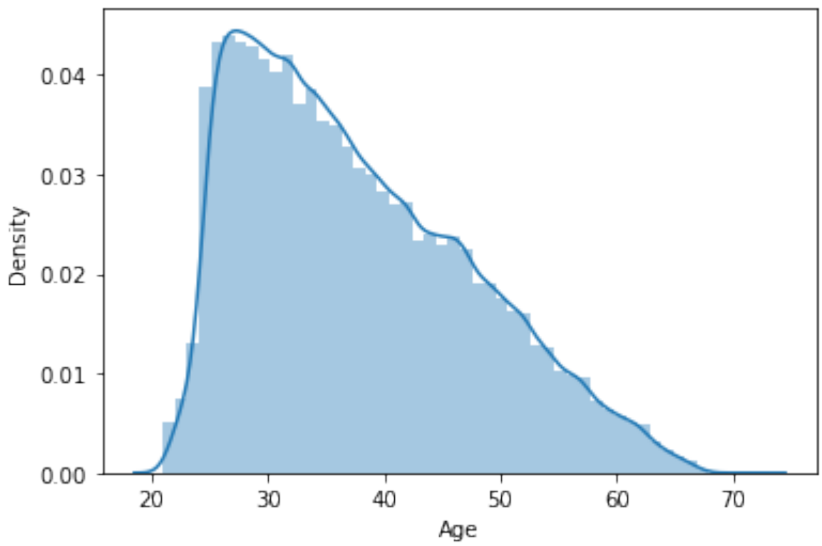
In [40]:
```
data['Loan_Default'].groupby(data['Age']).value_counts().plot(kind='hist').
```



In [41]:
```
sns.distplot(data['Loan_Default'].groupby(data['Age']).value_counts());
```



In [42]:
```
sns.distplot(data['Age']);
```



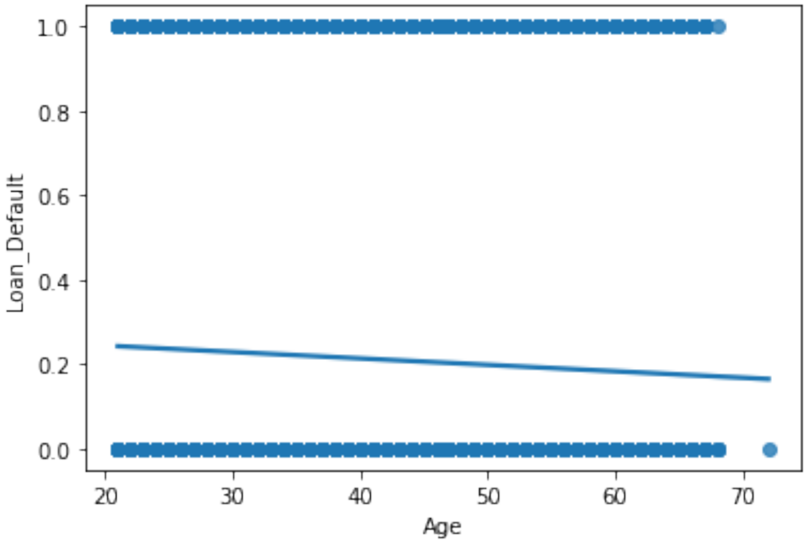
In [43]:
```
data[["Loan_Default", "Age"]].corr()
```

Out[43]:

	Loan_Default	Age
Loan_Default	1.000000	-0.036306
Age	-0.036306	1.000000

In [44]:
```
sns.regplot(x=data['Age'], y=data['Loan_Default'])
```

Out[44]: <AxesSubplot:xlabel='Age', ylabel='Loan_Default'>



What type of ID was presented by most of the customers for proof?

In [45]:
```
data.columns
```

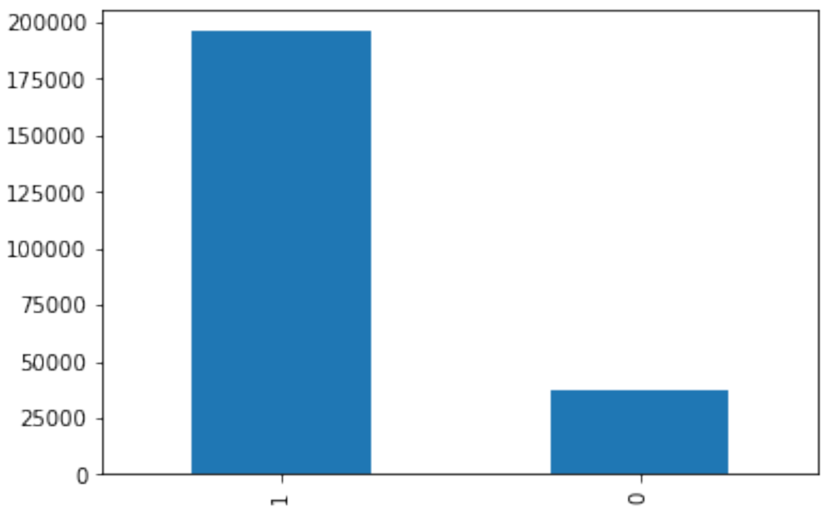
Out[45]:
```
Index(['Unique_ID', 'Disbursed_Amount', 'Asset_Cost', 'ltv', 'Branch_ID',
       'Supplier_ID', 'Manufacturer_ID', 'Current_Pincode_ID', 'Date_Of_Bir
th',
       'Employment_Type', 'Disbursal_Date', 'State_ID', 'Employee_Code_ID',
       'MobileNo_Avl_Flag', 'Aadhar_Flag', 'PAN_Flag', 'VoterID_Flag',
       'Driving_Flag', 'Passport_Flag', 'Perform_CNS_Score',
       'Perform_CNS_Score_Description', 'PRI_NO_OF_ACCTS', 'PRI_ACTIVE_ACCT
S',
       'PRI_OVERDUE_ACCTS', 'PRI_CURRENT_BALANCE', 'PRI_SANCTIONED_AMOUNT',
       'PRI_DISBURSED_AMOUNT', 'SEC_NO_OF_ACCTS', 'SEC_ACTIVE_ACCTS',
       'SEC_OVERDUE_ACCTS', 'SEC_CURRENT_BALANCE', 'SEC_SANCTIONED_AMOUNT',
       'SEC_DISBURSED_AMOUNT', 'PRIMARY_INSTAL_AMT', 'SEC_INSTAL_AMT',
       'NEW_ACCTS_IN_LAST_SIX_MONTHS', 'DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS
',
       'AVERAGE_ACCT_AGE', 'CREDIT_HISTORY_LENGTH', 'NO_OF_INQUIRIES',
       'Loan_Default', 'Age'],
      dtype='object')
```

In [46]:
```
print(data['Aadhar_Flag'].value_counts())

print(data['PAN_Flag'].value_counts())

print(data['VoterID_Flag'].value_counts())

print(data['Driving_Flag'].value_counts())

print(data['Passport_Flag'].value_counts())
```

```
1    195924
0     37230
Name: Aadhar_Flag, dtype: int64
0    215533
1     17621
Name: PAN_Flag, dtype: int64
0    199360
1     33794
Name: VoterID_Flag, dtype: int64
0    227735
1      5419
Name: Driving_Flag, dtype: int64
0    232658
1       496
Name: Passport_Flag, dtype: int64
```

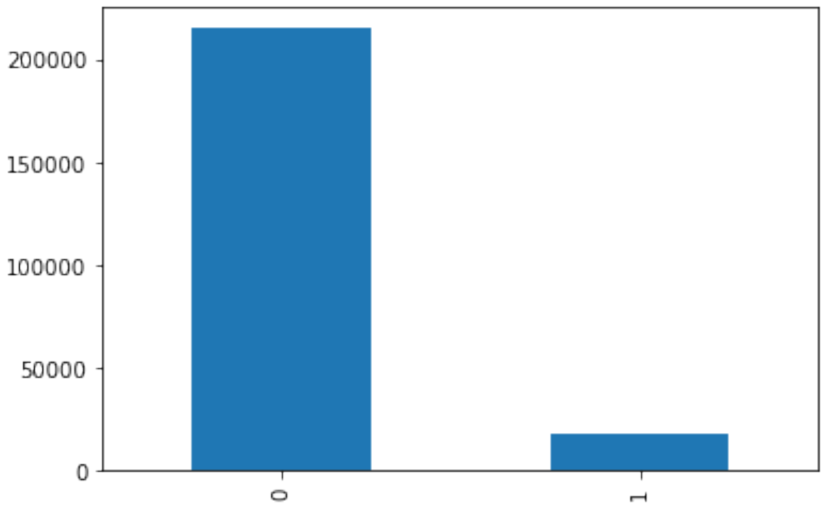
In [47]:
```
data['Aadhar_Flag'].value_counts().plot(kind='bar')
```

Out[47]: <AxesSubplot:>



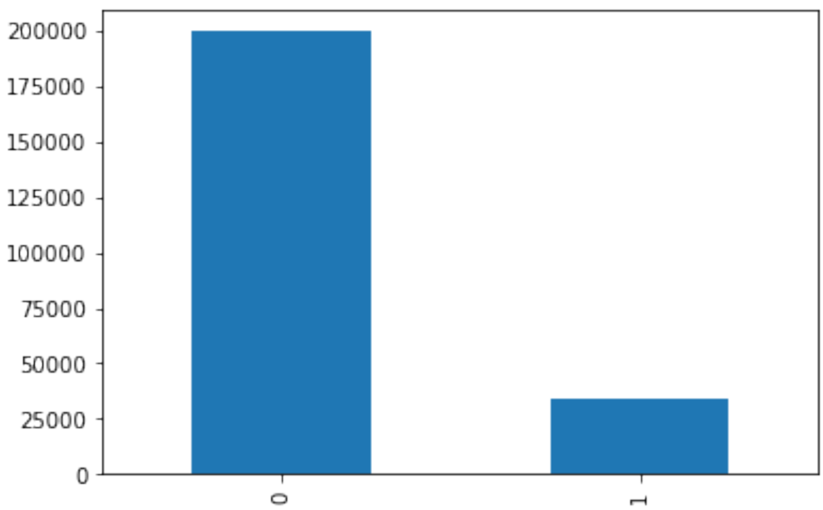
In [48]:
```
data['PAN_Flag'].value_counts().plot(kind='bar')
```

Out[48]: <AxesSubplot:>



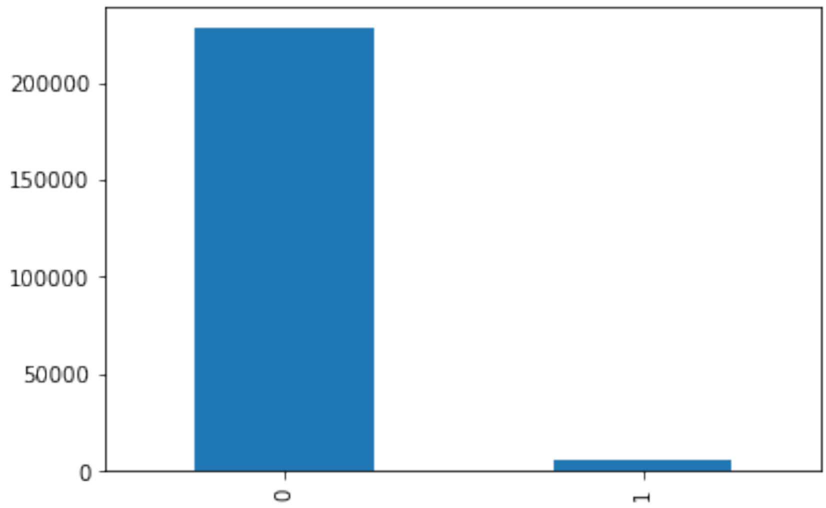
In [49]:
```
data['VoterID_Flag'].value_counts().plot(kind='bar')
```

Out[49]: <AxesSubplot:>



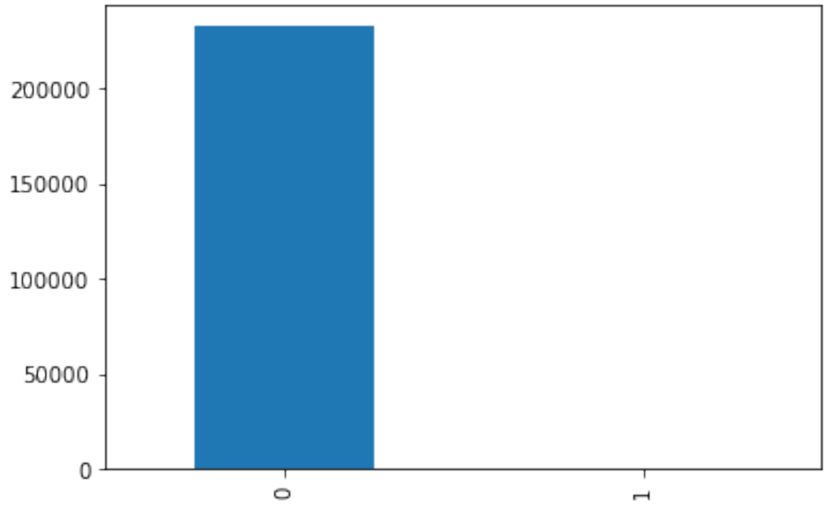
In [50]:
```
data['Driving_Flag'].value_counts().plot(kind='bar')
```

Out[50]: <AxesSubplot:>



In [51]:
```
data['Passport_Flag'].value_counts().plot(kind='bar')
```

Out[51]: <AxesSubplot:>



Adhar was presented by most of the customers for proof

Project Task: Week 2

Performing EDA and Modeling:

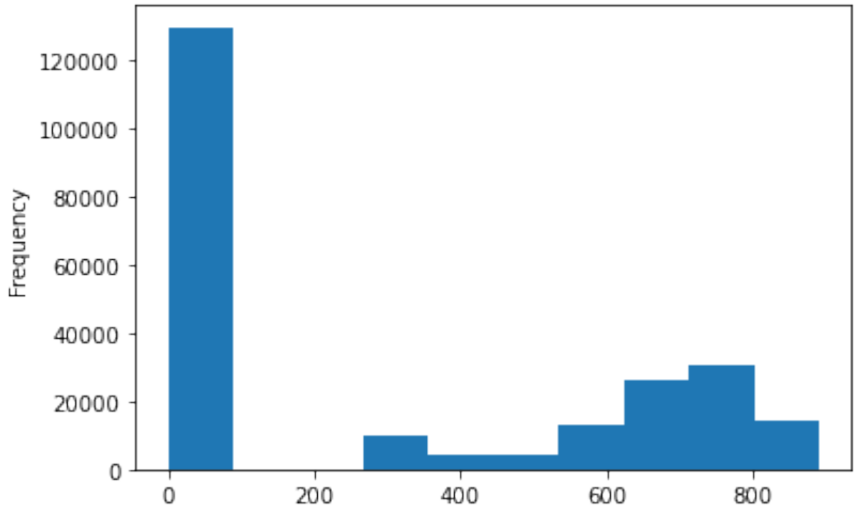
Study the credit bureau score distribution. Compare the distribution for defaulters vs. non-defaulters. Explore in detail.

In [52]:
```
data['Perform_CNS_Score'].value_counts()
```

Out[52]:
```
0      116950
300      8776
738      8662
825      7393
15       3765
        ...
863         1
847         1
867         1
834         1
884         1
Name: Perform_CNS_Score, Length: 573, dtype: int64
```

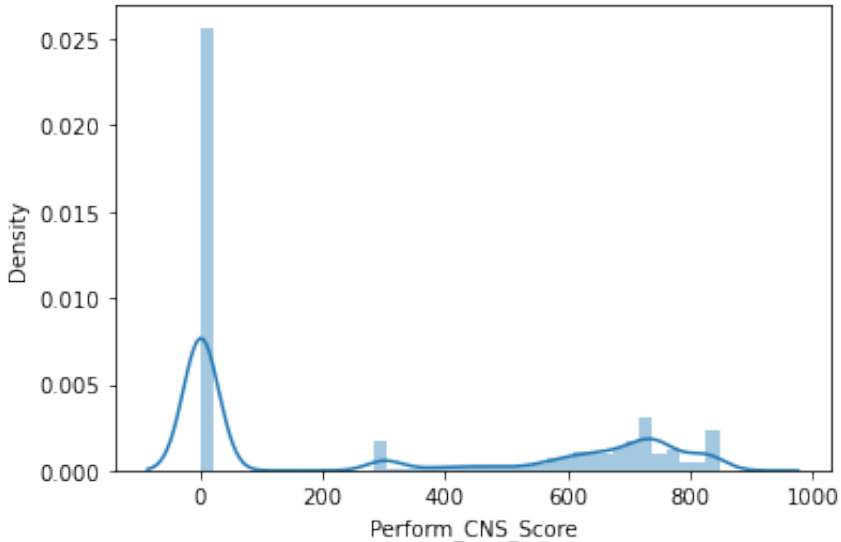
In [53]:
```
data['Perform_CNS_Score'].plot(kind='hist')
```

Out[53]: <AxesSubplot:ylabel='Frequency'>



In [54]:
```
sns.distplot(data['Perform_CNS_Score'])
```

Out[54]: <AxesSubplot:xlabel='Perform_CNS_Score', ylabel='Density'>



In [55]:
```
data['Perform_CNS_Score_Description'].values
```

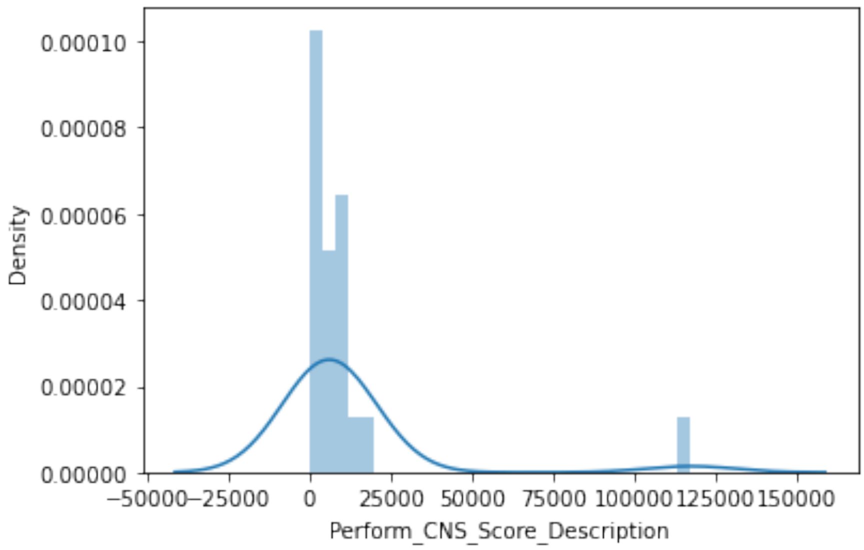
Out[55]:
```
array(['No Bureau History Available', 'No Bureau History Available',
       'No Bureau History Available', ...,
       'Not Scored: More than 50 active Accounts found',
       'Not Scored: More than 50 active Accounts found',
       'Not Scored: More than 50 active Accounts found'], dtype=object)
```

In [56]:
```
data['Perform_CNS_Score_Description'].value_counts()
```

Out[56]:
```
No Bureau History Available                               116950
C-Very Low Risk                                            16045
A-Very Low Risk                                            14124
D-Very Low Risk                                            11358
B-Very Low Risk                                             9201
M-Very High Risk                                            8776
F-Low Risk                                                  8485
K-High Risk                                                 8277
H-Medium Risk                                               6855
E-Low Risk                                                  5821
I-Medium Risk                                               5557
G-Low Risk                                                  3988
Not Scored: Sufficient History Not Available                3765
J-High Risk                                                 3748
Not Scored: Not Enough Info available on the customer       3672
Not Scored: No Activity seen on the customer (Inactive)     2885
Not Scored: No Updates available in last 36 months          1534
L-Very High Risk                                            1134
Not Scored: Only a Guarantor                                 976
Not Scored: More than 50 active Accounts found                 3
Name: Perform_CNS_Score_Description, dtype: int64
```

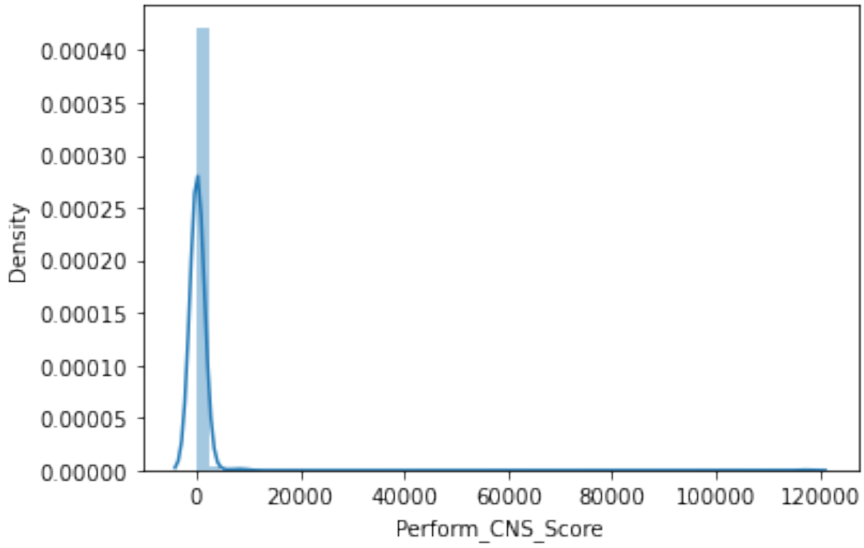
In [57]:
```
sns.distplot(data['Perform_CNS_Score_Description'].value_counts())
```

Out[57]: <AxesSubplot:xlabel='Perform_CNS_Score_Description', ylabel='Density'>



In [58]:
```
sns.distplot(data['Perform_CNS_Score'].value_counts())
```

Out[58]: <AxesSubplot:xlabel='Perform_CNS_Score', ylabel='Density'>

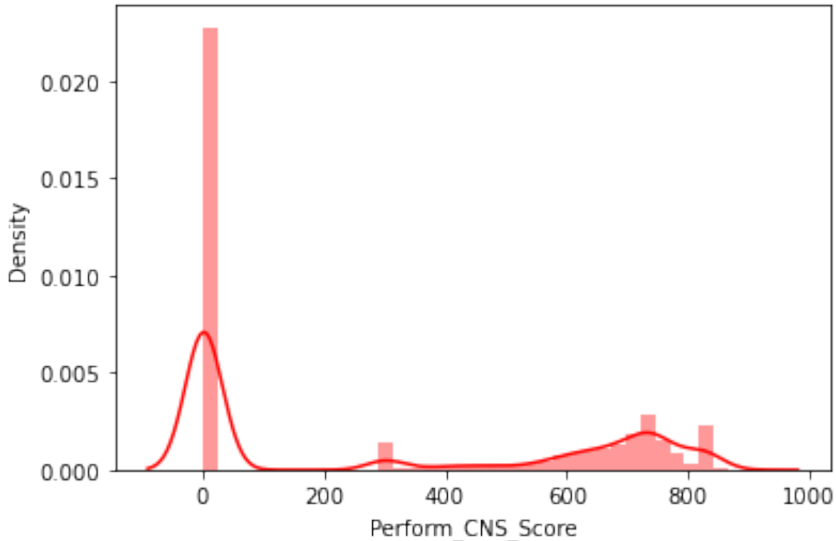


In [59]:
```
data['Loan_Default'].value_counts()
```

Out[59]:
```
0    182543
1     50611
Name: Loan_Default, dtype: int64
```

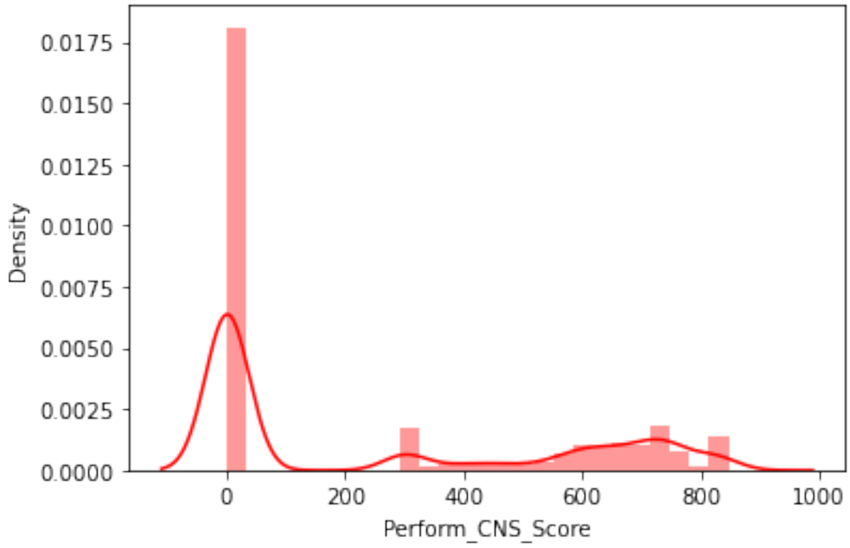
In [60]:
```
sns.distplot(data[data['Loan_Default']==0]['Perform_CNS_Score'],color='r',]
```

Out[60]: <AxesSubplot:xlabel='Perform_CNS_Score', ylabel='Density'>



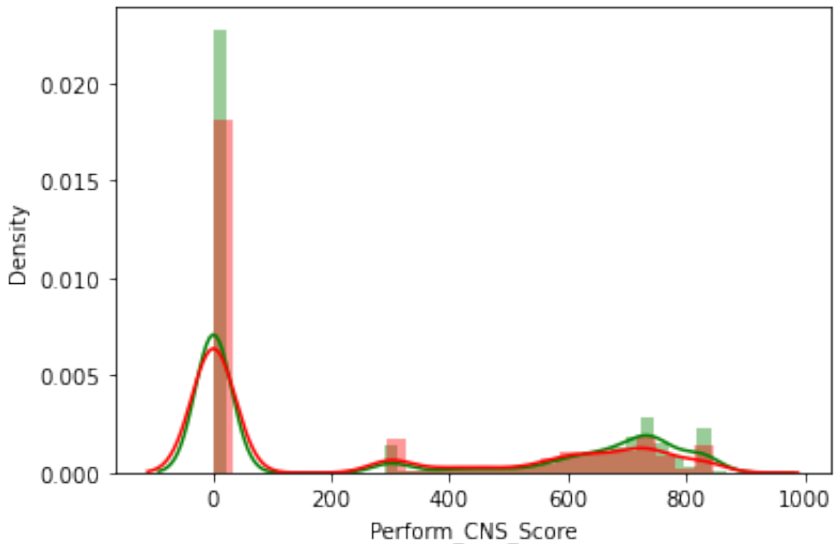
In [61]:
```
sns.distplot(data[data['Loan_Default']==1]['Perform_CNS_Score'],color='r',]
```

Out[61]: <AxesSubplot:xlabel='Perform_CNS_Score', ylabel='Density'>



In [62]:
```
sns.distplot(data[data['Loan_Default']==0]['Perform_CNS_Score'],color='g',
sns.distplot(data[data['Loan_Default']==1]['Perform_CNS_Score'],color='r',
```

Out[62]: <AxesSubplot:xlabel='Perform_CNS_Score', ylabel='Density'>



CNS score is lower for defaulter as compared to non - defaulters

Explore the primary and secondary account details. Is the information in some way related to the loan default probability?

In [63]:
```
data['PRI_NO_OF_ACCTS'].values
```

Out[63]:
```
array([  0,   0,   0, ...,  68,  72, 194])
```

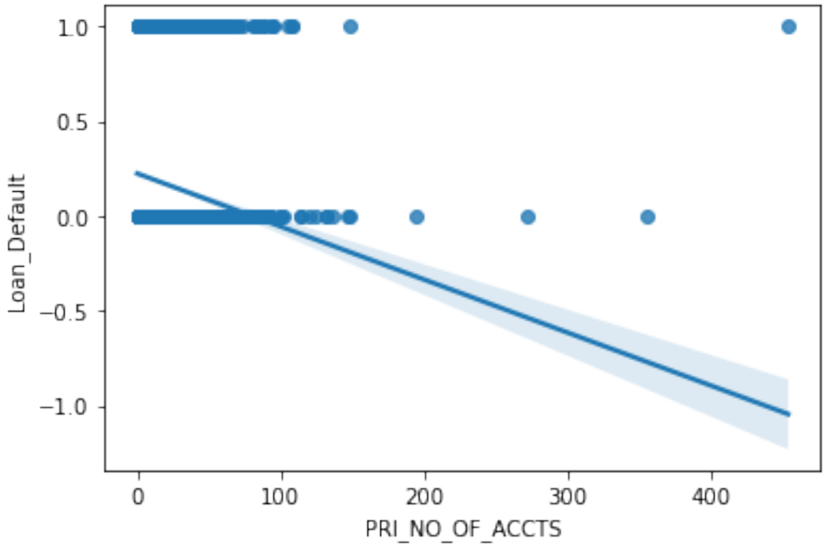
In [64]:
```
data[['Loan_Default','PRI_NO_OF_ACCTS']].corr()
```

Out[64]:

	Loan_Default	PRI_NO_OF_ACCTS
Loan_Default	1.000000	-0.035456
PRI_NO_OF_ACCTS	-0.035456	1.000000

In [65]:
```
sns.regplot(y=data['Loan_Default'], x=data['PRI_NO_OF_ACCTS'])
```

Out[65]: <AxesSubplot:xlabel='PRI_NO_OF_ACCTS', ylabel='Loan_Default'>



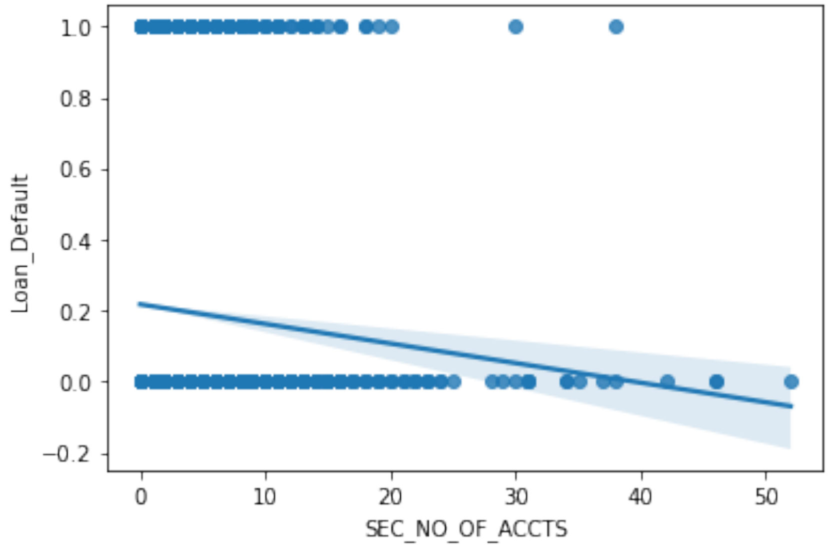
In [66]:
```
data[['Loan_Default','SEC_NO_OF_ACCTS']].corr()
```

Out[66]:

	Loan_Default	SEC_NO_OF_ACCTS
Loan_Default	1.000000	-0.008385
SEC_NO_OF_ACCTS	-0.008385	1.000000

In [67]:
```
sns.regplot(x=data['SEC_NO_OF_ACCTS'], y=data['Loan_Default'])
```

Out[67]: <AxesSubplot:xlabel='SEC_NO_OF_ACCTS', ylabel='Loan_Default'>



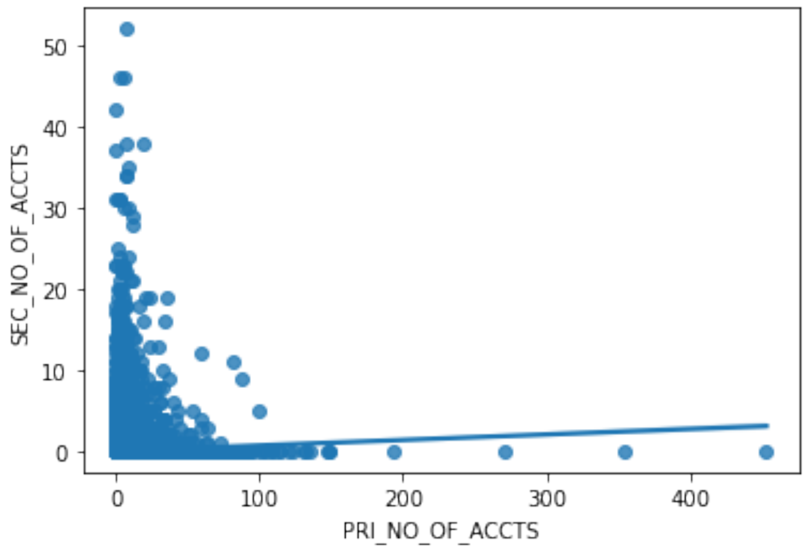
In [68]:
```
data[['PRI_NO_OF_ACCTS','SEC_NO_OF_ACCTS']].corr()
```

Out[68]:

	PRI_NO_OF_ACCTS	SEC_NO_OF_ACCTS
PRI_NO_OF_ACCTS	1.000000	0.056434
SEC_NO_OF_ACCTS	0.056434	1.000000

In [69]:
```
sns.regplot(x=data['PRI_NO_OF_ACCTS'], y=data['SEC_NO_OF_ACCTS'])
```

Out[69]: <AxesSubplot:xlabel='PRI_NO_OF_ACCTS', ylabel='SEC_NO_OF_ACCTS'>



Is there a difference between the sanctioned and disbursed amount of primary and secondary loans? Study the difference by providing appropriate statistics and graphs.

In [70]:
```
data['PRI_SANCTIONED_AMOUNT'].describe()
```

Out[70]:
```
count    2.331540e+05
mean     2.185039e+05
std      2.374794e+06
min      0.000000e+00
25%      0.000000e+00
50%      0.000000e+00
75%      6.250000e+04
max      1.000000e+09
Name: PRI_SANCTIONED_AMOUNT, dtype: float64
```

In [71]:
```
data['PRI_DISBURSED_AMOUNT'].describe()
```

Out[71]:
```
count    2.331540e+05
mean     2.180659e+05
std      2.377744e+06
min      0.000000e+00
25%      0.000000e+00
50%      0.000000e+00
75%      6.080000e+04
max      1.000000e+09
Name: PRI_DISBURSED_AMOUNT, dtype: float64
```

In [72]:
```
data['PRI_SANCTIONED_AMOUNT'].skew()
```

Out[72]:
```
323.69721207047974
```

In [73]:
```
data['PRI_DISBURSED_AMOUNT'].skew()
```

Out[73]:
```
322.5414945101688
```

In [74]:
```
data['SEC_SANCTIONED_AMOUNT'].describe()
```

Out[74]:
```
count    2.331540e+05
mean     7.295923e+03
std      1.831560e+05
min      0.000000e+00
25%      0.000000e+00
50%      0.000000e+00
75%      0.000000e+00
max      3.000000e+07
Name: SEC_SANCTIONED_AMOUNT, dtype: float64
```

In [75]:
```
data['SEC_DISBURSED_AMOUNT'].describe()
```

Out[75]:
```
count    2.331540e+05
mean     7.179998e+03
std      1.825925e+05
min      0.000000e+00
25%      0.000000e+00
50%      0.000000e+00
75%      0.000000e+00
max      3.000000e+07
Name: SEC_DISBURSED_AMOUNT, dtype: float64
```

In [76]:
```
data['SEC_SANCTIONED_AMOUNT'].skew()
```

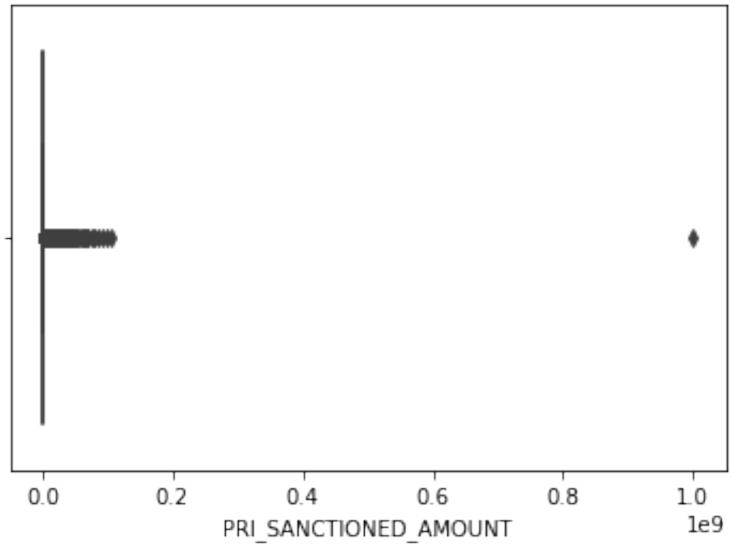
Out[76]:
```
75.25493196054583
```

In [77]:
```
data['SEC_DISBURSED_AMOUNT'].skew()
```

Out[77]:
```
75.76425191107285
```

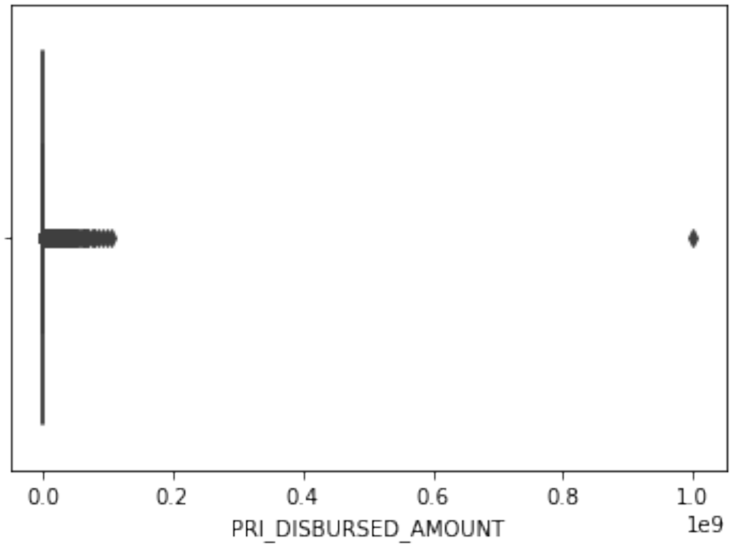
In [78]:
```
sns.boxplot(data['PRI_SANCTIONED_AMOUNT'])
```

Out[78]:
```
<AxesSubplot:xlabel='PRI_SANCTIONED_AMOUNT'>
```



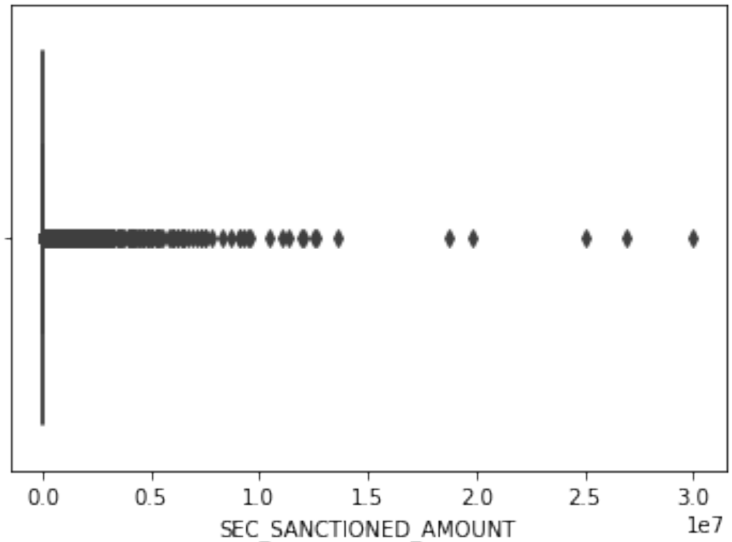
In [79]:
```
sns.boxplot(data['PRI_DISBURSED_AMOUNT'])
```

Out[79]: <AxesSubplot:xlabel='PRI_DISBURSED_AMOUNT'>



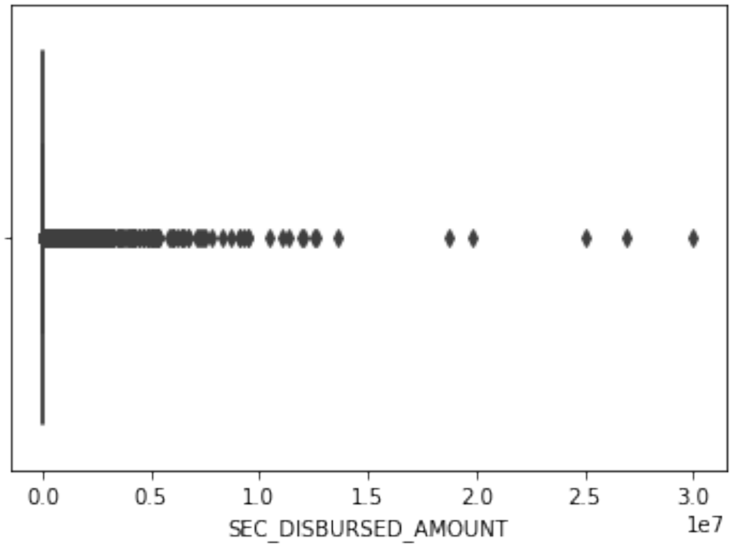
In [80]:
```
sns.boxplot(data['SEC_SANCTIONED_AMOUNT'])
```

Out[80]: <AxesSubplot:xlabel='SEC_SANCTIONED_AMOUNT'>



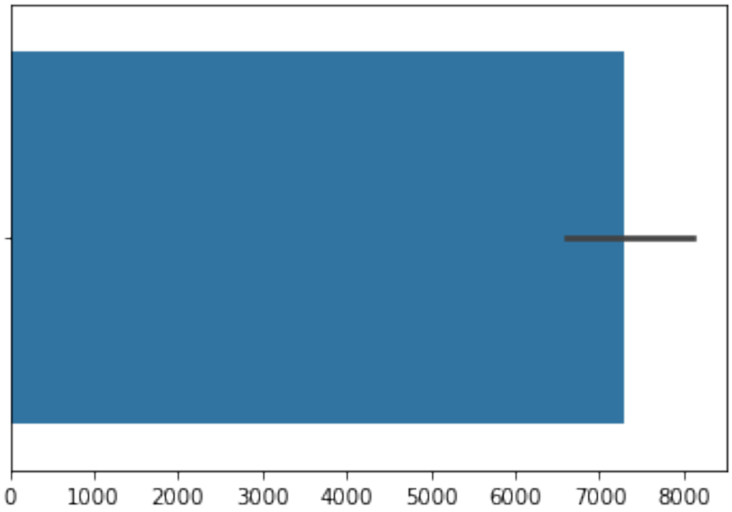
In [81]:
```
sns.boxplot(data['SEC_DISBURSED_AMOUNT'])
```

Out[81]: <AxesSubplot:xlabel='SEC_DISBURSED_AMOUNT'>



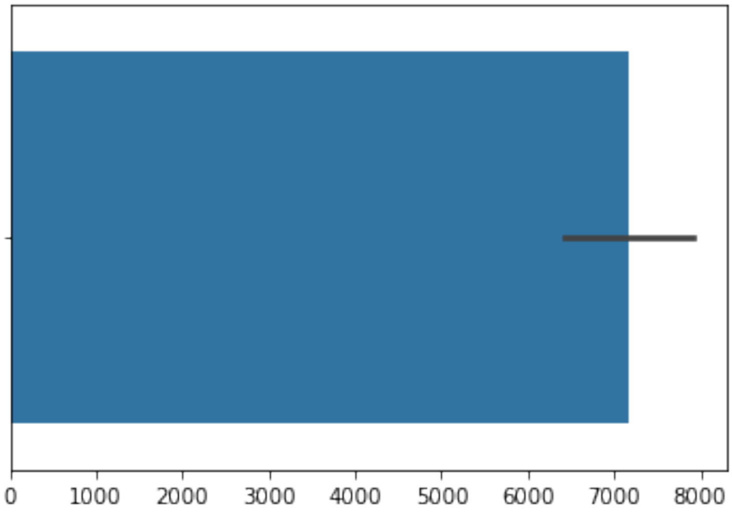
In [82]:
```
sns.barplot(data['SEC_SANCTIONED_AMOUNT'].values)
```

Out[82]: <AxesSubplot:>



In [83]:
```
sns.barplot(data['SEC_DISBURSED_AMOUNT'].values)
```

Out[83]: <AxesSubplot:>



Do customer who make higher number of enquiries end up being higher risk candidates?

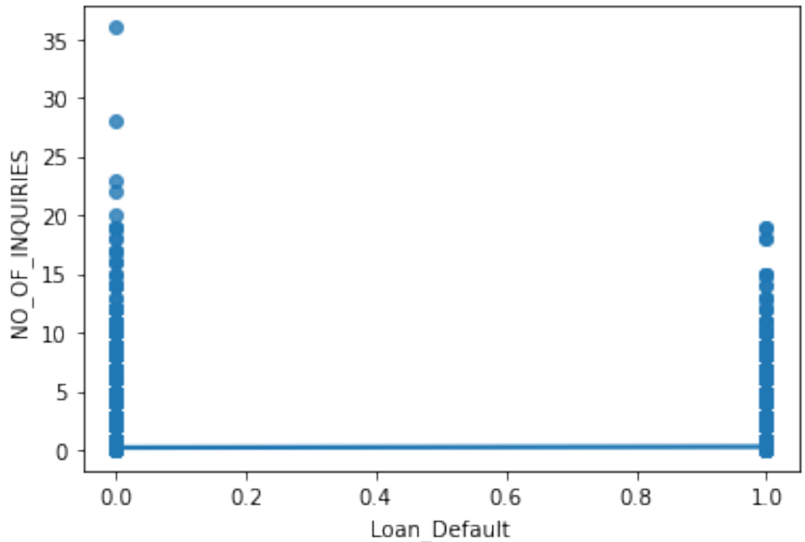
In [84]:
```
data[['Loan_Default','NO_OF_INQUIRIES']].corr()
```

Out[84]:

	Loan_Default	NO_OF_INQUIRIES
Loan_Default	1.000000	0.043678
NO_OF_INQUIRIES	0.043678	1.000000

In [85]:
```
sns.regplot(y=data['NO_OF_INQUIRIES'], x=data['Loan_Default'])
```

Out[85]: <AxesSubplot:xlabel='Loan_Default', ylabel='NO_OF_INQUIRIES'>

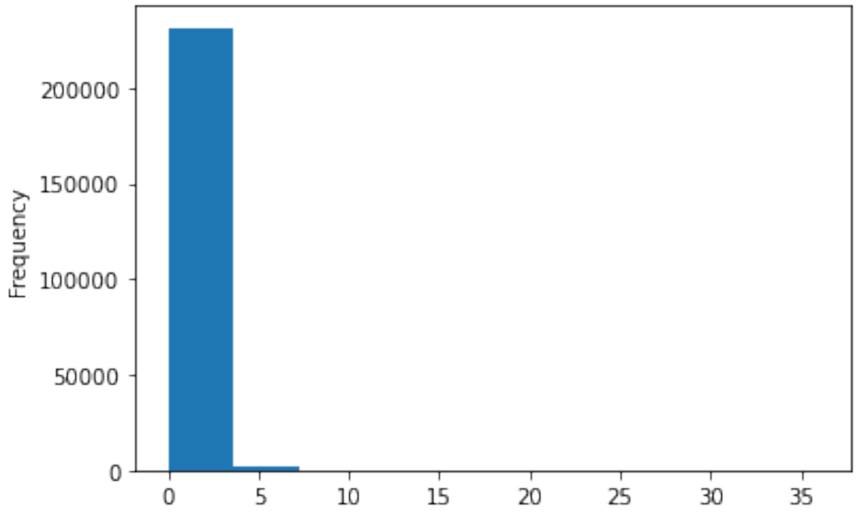


In [86]:
```
data['NO_OF_INQUIRIES'].value_counts()
```

Out[86]:
```
0     201961
1      22285
2       5409
3       1767
4        760
5        343
6        239
7        135
8        105
9         44
10        34
11        15
12        14
14         8
15         7
19         6
13         6
17         4
18         4
16         3
28         1
20         1
23         1
36         1
22         1
Name: NO_OF_INQUIRIES, dtype: int64
```

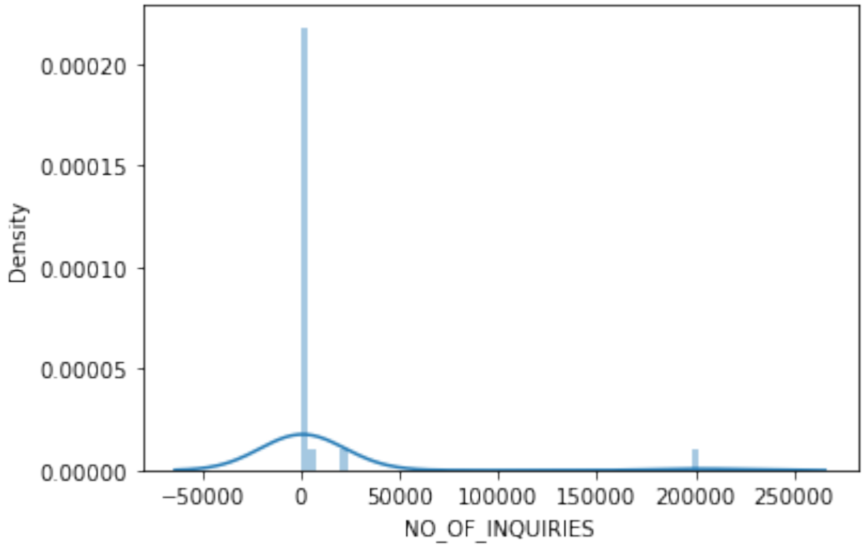
In [87]:
```
data['NO_OF_INQUIRIES'].plot(kind='hist')
```

Out[87]: <AxesSubplot:ylabel='Frequency'>



In [88]:
```
sns.distplot(data['NO_OF_INQUIRIES'].value_counts())
```

Out[88]: <AxesSubplot:xlabel='NO_OF_INQUIRIES', ylabel='Density'>



Is credit history, that is new loans in last six months, loans defaulted in last six months, time since first loan, etc., a significant factor in estimating probability of loan defaulters?

In [89]:
```
data.head(2)
```

Out[89]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	

2 rows × 42 columns

In [90]:
```
data.columns
```

Out[90]:
```
Index(['Unique_ID', 'Disbursed_Amount', 'Asset_Cost', 'ltv', 'Branch_ID',
       'Supplier_ID', 'Manufacturer_ID', 'Current_Pincode_ID', 'Date_Of_Bir
th',
       'Employment_Type', 'Disbursal_Date', 'State_ID', 'Employee_Code_ID',
       'MobileNo_Avl_Flag', 'Aadhar_Flag', 'PAN_Flag', 'VoterID_Flag',
       'Driving_Flag', 'Passport_Flag', 'Perform_CNS_Score',
       'Perform_CNS_Score_Description', 'PRI_NO_OF_ACCTS', 'PRI_ACTIVE_ACCT
S',
       'PRI_OVERDUE_ACCTS', 'PRI_CURRENT_BALANCE', 'PRI_SANCTIONED_AMOUNT',
       'PRI_DISBURSED_AMOUNT', 'SEC_NO_OF_ACCTS', 'SEC_ACTIVE_ACCTS',
       'SEC_OVERDUE_ACCTS', 'SEC_CURRENT_BALANCE', 'SEC_SANCTIONED_AMOUNT',
       'SEC_DISBURSED_AMOUNT', 'PRIMARY_INSTAL_AMT', 'SEC_INSTAL_AMT',
       'NEW_ACCTS_IN_LAST_SIX_MONTHS', 'DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS
',
       'AVERAGE_ACCT_AGE', 'CREDIT_HISTORY_LENGTH', 'NO_OF_INQUIRIES',
       'Loan_Default', 'Age'],
      dtype='object')
```

In [91]:
```
data.nunique()
```

Out[91]:
```
Unique_ID                              233154
Disbursed_Amount                        24565
Asset_Cost                              46252
ltv                                      6579
Branch_ID                                  82
Supplier_ID                              2953
Manufacturer_ID                            11
Current_Pincode_ID                       6698
Date_Of_Birth                           15433
Employment_Type                             2
Disbursal_Date                             84
State_ID                                   22
Employee_Code_ID                         3270
MobileNo_Avl_Flag                           1
Aadhar_Flag                                 2
PAN_Flag                                    2
VoterID_Flag                                2
Driving_Flag                                2
Passport_Flag                               2
Perform_CNS_Score                         573
Perform_CNS_Score_Description              20
PRI_NO_OF_ACCTS                           108
PRI_ACTIVE_ACCTS                           40
PRI_OVERDUE_ACCTS                          22
PRI_CURRENT_BALANCE                     71341
PRI_SANCTIONED_AMOUNT                   44390
PRI_DISBURSED_AMOUNT                    47909
SEC_NO_OF_ACCTS                            37
SEC_ACTIVE_ACCTS                           23
SEC_OVERDUE_ACCTS                           9
SEC_CURRENT_BALANCE                      3246
SEC_SANCTIONED_AMOUNT                    2223
SEC_DISBURSED_AMOUNT                     2553
PRIMARY_INSTAL_AMT                      28067
SEC_INSTAL_AMT                           1918
NEW_ACCTS_IN_LAST_SIX_MONTHS               26
DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS        14
AVERAGE_ACCT_AGE                          192
CREDIT_HISTORY_LENGTH                     294
NO_OF_INQUIRIES                            25
Loan_Default                                2
Age                                        49
dtype: int64
```

In [92]:
```
data['Disbursal_Date']
```

Out[92]:
```
0         2018-08-03
1         2018-08-01
2         2018-09-26
3         2018-09-23
4         2018-10-08
             ...
233149    2018-10-06
233150    2018-10-31
233151    2018-10-23
233152    2018-08-17
233153    2018-09-28
Name: Disbursal_Date, Length: 233154, dtype: datetime64[ns]
```

Yes, credit history, that is new loans in last six months, loans defaulted in last six months, time since first loan, etc., a significant factor in estimating probability of loan defaulters.

In [93]:
```
now = pd.Timestamp('now')
data['Disbursal_Date'] = pd.to_datetime(data['Disbursal_Date'], format='%d-
data['Disbursal_Date'] = data['Disbursal_Date'].where(data['Disbursal_Date
data['Time_Since_Loan_Dispursed_In_Yrs'] = (now - data['Disbursal_Date']).a
data=data.drop('Disbursal_Date',axis=1)
```

In [94]:
```
data.head()
```

Out[94]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	
2	539055	52378	60300	88.39	30	1415	
3	529269	46349	61500	76.42	30	1415	
4	563215	43594	78256	57.50	30	1378	

5 rows × 42 columns

Converting CHL & AAA

In [95]:
```
data['CREDIT_HISTORY_LENGTH']
```

Out[95]:

```
0          0yrs 0mon
1          0yrs 0mon
2          0yrs 0mon
3          0yrs 0mon
4          0yrs 0mon
              ...
233149     2yrs 4mon
233150     1yrs 5mon
233151    3yrs 10mon
233152     3yrs 2mon
233153     5yrs 4mon
Name: CREDIT_HISTORY_LENGTH, Length: 233154, dtype: object
```

In [96]:

```
data[['CREDIT_HISTORY_LENGTH_1','CREDIT_HISTORY_LENGTH_2']]=data['CREDIT_HI
data=data.drop(columns ='CREDIT_HISTORY_LENGTH')
```

In [97]:

```
data.head()
```

Out[97]:

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	
2	539055	52378	60300	88.39	30	1415	
3	529269	46349	61500	76.42	30	1415	
4	563215	43594	78256	57.50	30	1378	

5 rows × 43 columns

In [98]:

```
#stripping months and years
data['CREDIT_HISTORY_LENGTH_1']=data['CREDIT_HISTORY_LENGTH_1'].str.strip(
```

In [99]:

```
#stripping months and years
data['CREDIT_HISTORY_LENGTH_2']= data['CREDIT_HISTORY_LENGTH_2'].str.strip


#converting datatype
data['CREDIT_HISTORY_LENGTH_1'] = data['CREDIT_HISTORY_LENGTH_1'].astype(in
data['CREDIT_HISTORY_LENGTH_2'] = data['CREDIT_HISTORY_LENGTH_2'].astype(in

# since we need to conctanate month value lets divide by 12 and round them
data['CREDIT_HISTORY_LENGTH_2']=round((data['CREDIT_HISTORY_LENGTH_2']/12)
```

In [100…

```
data.head()
```

Out[100…

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	
2	539055	52378	60300	88.39	30	1415	
3	529269	46349	61500	76.42	30	1415	
4	563215	43594	78256	57.50	30	1378	

5 rows × 43 columns

In [101…

```
data['CREDIT_HISTORY_LENGTH']= data['CREDIT_HISTORY_LENGTH_1'].astype(float
```

In [102…

```
data['CREDIT_HISTORY_LENGTH']
```

Out[102…

```
0         0.00
1         0.00
2         0.00
3         0.00
4         0.00
          ...
233149    2.33
233150    1.42
233151    3.83
233152    3.17
233153    5.33
Name: CREDIT_HISTORY_LENGTH, Length: 233154, dtype: float64
```

In [103…

```
data['CREDIT_HISTORY_LENGTH'].value_counts()
```

Out[103…

```
0.00     119127
0.50       4761
2.08       4745
0.58       4017
2.00       3833
          ...
21.17         1
26.92         1
28.75         1
28.58         1
27.33         1
Name: CREDIT_HISTORY_LENGTH, Length: 294, dtype: int64
```

In [104…

```
data['AVERAGE_ACCT_AGE']
```

Out[104…
```
0          0yrs 0mon
1          0yrs 0mon
2          0yrs 0mon
3          0yrs 0mon
4          0yrs 0mon
              ...
233149     2yrs 4mon
233150     1yrs 5mon
233151     0yrs 9mon
233152     1yrs 2mon
233153    2yrs 11mon
Name: AVERAGE_ACCT_AGE, Length: 233154, dtype: object
```

In [105…
```
data[['AVERAGE_ACCT_AGE_1','AVERAGE_ACCT_AGE_2']]=data['AVERAGE_ACCT_AGE']
data=data.drop(columns ='AVERAGE_ACCT_AGE')
```

In [106…
```
#stripping months and years
data['AVERAGE_ACCT_AGE_1']=data['AVERAGE_ACCT_AGE_1'].str.strip('yrs')
```

In [107…
```
#stripping months and years
data['AVERAGE_ACCT_AGE_2']= data['AVERAGE_ACCT_AGE_2'].str.strip('mon')
```

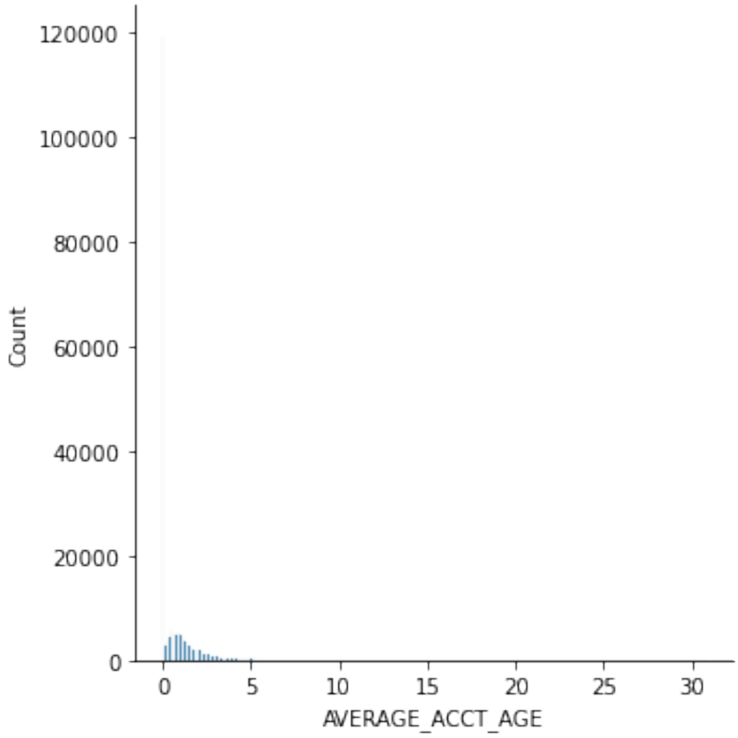
In [108…
```
#converting datatype
data['AVERAGE_ACCT_AGE_1'] = data['AVERAGE_ACCT_AGE_1'].astype(int)
data['AVERAGE_ACCT_AGE_2'] = data['AVERAGE_ACCT_AGE_2'].astype(int)
```

In [109…
```
# since we need to conctanate month value lets divide by 12 and round them
data['AVERAGE_ACCT_AGE_2']=round((data['AVERAGE_ACCT_AGE_2']/12),2)
```

In [110…
```
data['AVERAGE_ACCT_AGE']= data['AVERAGE_ACCT_AGE_1'].astype(float) + data[
```

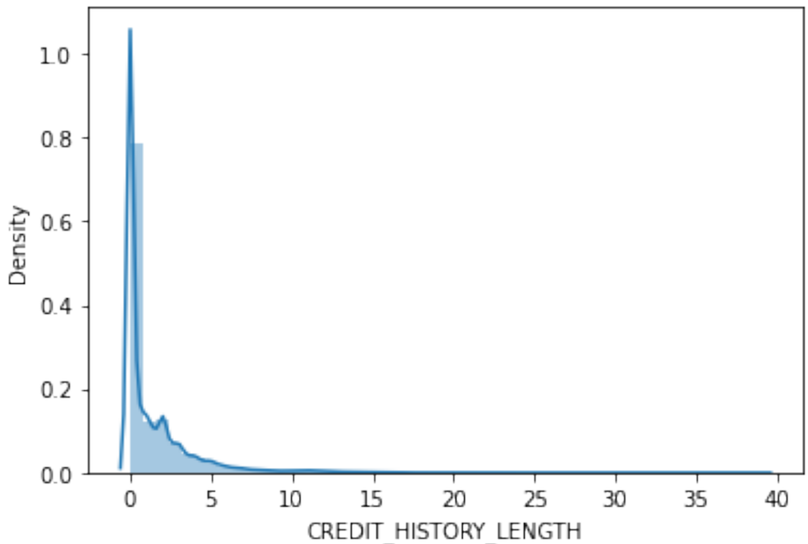
In [111…
```
sns.displot(data['AVERAGE_ACCT_AGE'])
```

Out[111… <seaborn.axisgrid.FacetGrid at 0x7f82b8b35d30>



In [112…
```
sns.distplot(data['CREDIT_HISTORY_LENGTH'])
```

Out[112… <AxesSubplot:xlabel='CREDIT_HISTORY_LENGTH', ylabel='Density'>



In [113…
```
data.head(2)
```

Out[113...

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	

2 rows × 46 columns

In [114...

```
data=data.drop(columns ='CREDIT_HISTORY_LENGTH_1')

data=data.drop(columns ='CREDIT_HISTORY_LENGTH_2')

data=data.drop(columns ='AVERAGE_ACCT_AGE_1')

data=data.drop(columns ='AVERAGE_ACCT_AGE_2')

data=data.drop(columns ='Date_Of_Birth')
```

In [115...

```
data.head()
```

Out[115...

	Unique_ID	Disbursed_Amount	Asset_Cost	ltv	Branch_ID	Supplier_ID	Manufacturer
0	420825	50578	58400	89.55	30	1415	
1	417566	53278	61360	89.63	30	1415	
2	539055	52378	60300	88.39	30	1415	
3	529269	46349	61500	76.42	30	1415	
4	563215	43594	78256	57.50	30	1378	

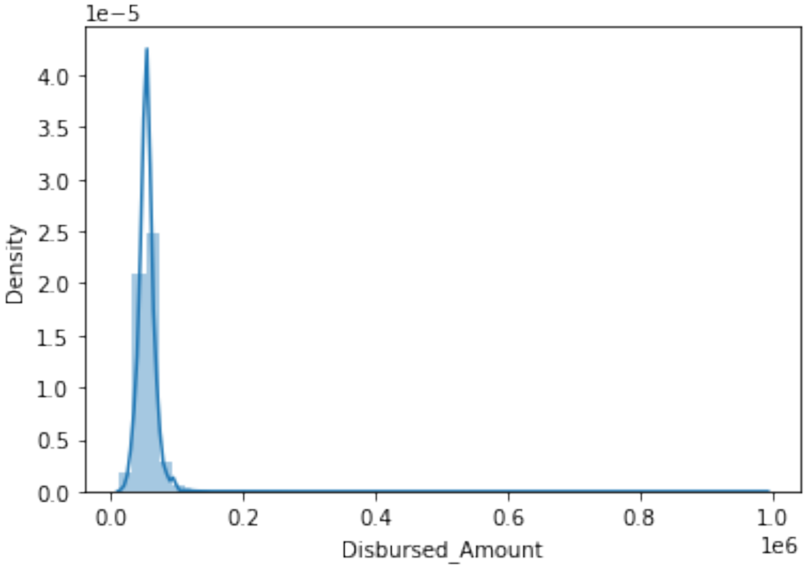
5 rows × 41 columns

In [116...

```
sns.distplot(data['Disbursed_Amount'])
```

Out [116…

```
<AxesSubplot:xlabel='Disbursed_Amount', ylabel='Density'>
```

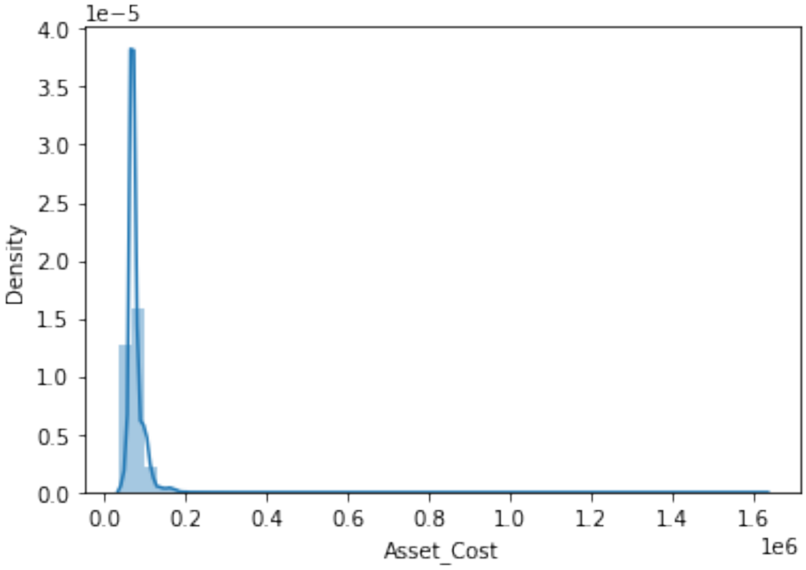


In [117…

```
sns.distplot(data['Asset_Cost'])
```

Out [117…

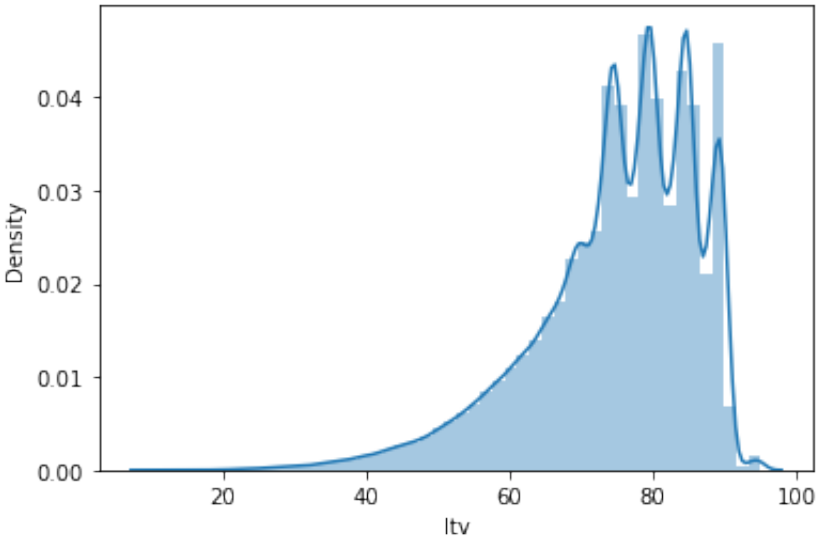
```
<AxesSubplot:xlabel='Asset_Cost', ylabel='Density'>
```



In [118…

```
sns.distplot(data['ltv'])
```

Out[118... <AxesSubplot:xlabel='ltv', ylabel='Density'>



In [119...
```
data['Perform_CNS_Score_Description'].nunique()
```

Out[119... 20

In [120...
```
# for Perform_CNS_Score_Description reducing the 20 classes by replacing w
data['Perform_CNS_Score_Description'].replace(to_replace=['Not Scored: More
                                                          'No Bureau Hi
                                                          'Not Scored:
                                                          'Not Enough I
                                                          'Not Scored:
                                                          'Not Scored:
                                                          'Not Scored:
                                              value= 'No_score', inplac
```

In [121...
```
data['Perform_CNS_Score_Description'].value_counts()
```

Out[121...
```
No_score              129785
C-Very Low Risk        16045
A-Very Low Risk        14124
D-Very Low Risk        11358
B-Very Low Risk         9201
M-Very High Risk        8776
F-Low Risk              8485
K-High Risk             8277
H-Medium Risk           6855
E-Low Risk              5821
I-Medium Risk           5557
G-Low Risk              3988
J-High Risk             3748
L-Very High Risk        1134
Name: Perform_CNS_Score_Description, dtype: int64
```

In [122…

```
data['Perform_CNS_Score_Description'].nunique()
```

Out[122…

```
14
```

In [123…

```
Very_Low_Risk=['A-Very Low Risk','B-Very Low Risk','C-Very Low Risk','D-Ve
Low_Risk= ['E-Low Risk','F-Low Risk','G-Low Risk']
Midium_Risk= ['H-Medium Risk','I-Medium Risk']
High_Risk= ['J-High Risk','K-High Risk']
Very_High_Risk=['L-Very High Risk','M-Very High Risk']
```

In [124…

```
data['Perform_CNS_Score_Description'].replace(to_replace='No_score',value =

data['Perform_CNS_Score_Description'].replace(to_replace=Very_Low_Risk, val

data['Perform_CNS_Score_Description'].replace(to_replace=Low_Risk, value= 

data['Perform_CNS_Score_Description'].replace(to_replace=Midium_Risk, value

data['Perform_CNS_Score_Description'].replace(to_replace=High_Risk, value= 

data['Perform_CNS_Score_Description'].replace(to_replace=Very_High_Risk, va
```

In [125…

```
data['Perform_CNS_Score_Description'].value_counts()
```

Out[125…

```
0    129785
1     50728
2     18294
3     12412
4     12025
5      9910
Name: Perform_CNS_Score_Description, dtype: int64
```

In [126…

```
data.nunique()
```

Out[126…

```
Unique_ID                              233154
Disbursed_Amount                        24565
Asset_Cost                              46252
ltv                                      6579
Branch_ID                                  82
Supplier_ID                              2953
Manufacturer_ID                            11
Current_Pincode_ID                       6698
Employment_Type                             2
State_ID                                   22
Employee_Code_ID                         3270
MobileNo_Avl_Flag                           1
Aadhar_Flag                                 2
PAN_Flag                                    2
VoterID_Flag                                2
Driving_Flag                                2
Passport_Flag                               2
Perform_CNS_Score                         573
Perform_CNS_Score_Description               6
PRI_NO_OF_ACCTS                           108
PRI_ACTIVE_ACCTS                           40
PRI_OVERDUE_ACCTS                          22
PRI_CURRENT_BALANCE                     71341
PRI_SANCTIONED_AMOUNT                   44390
PRI_DISBURSED_AMOUNT                    47909
SEC_NO_OF_ACCTS                            37
SEC_ACTIVE_ACCTS                           23
SEC_OVERDUE_ACCTS                           9
SEC_CURRENT_BALANCE                      3246
SEC_SANCTIONED_AMOUNT                    2223
SEC_DISBURSED_AMOUNT                     2553
PRIMARY_INSTAL_AMT                      28067
SEC_INSTAL_AMT                           1918
NEW_ACCTS_IN_LAST_SIX_MONTHS               26
DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS        14
NO_OF_INQUIRIES                            25
Loan_Default                                2
Age                                        49
Time_Since_Loan_Dispursed_In_Yrs            1
CREDIT_HISTORY_LENGTH                     294
AVERAGE_ACCT_AGE                          192
dtype: int64
```

Droping

Here we need to look out for ID related variables such as manufacturer id employee id etc. how do we need whether we should select these variables? lets look from a financial stand point

1. UniqueID - Unique id of loan candidate. this is purely nominal and needs to be dropped.
2. Supplier_id - Denotes distinct supplier. needs to be dropped beacuse it is nominal.
3. Current_Pincode_ID - Denotes location and has nothing to do with probablity of default or default prediction will be dropped.
4. Branch_ID - There are 82 branch.ids denoting 82 seperate branches this variable denotes the branch from which the vehicle was taken .This is purely nominal and has no order at all so we can drop this variable.
5. State_ID - Denotes the states registration of the vehicle (like MH for maharashtra , KA for karnataka) this may matter because prices of vehicals vary state to state.
6. Employee_Code_ID - Nominal.
7. VoterID_Flag - Certainly will make an impact,so it should be there in our analysis.
8. Manufacturer_ID - There are 11 such variables and it denotes unique manufacturer for vehicles, So this should be considered since prices vary manufacturer to manufacturer.

In [127…

```
data=data.drop(columns ='Unique_ID')

data=data.drop(columns ='Supplier_ID')

data=data.drop(columns ='Current_Pincode_ID')

data=data.drop(columns ='Branch_ID')

data=data.drop(columns ='Employee_Code_ID')
```

In [128…

```
data=data.drop(columns ='MobileNo_Avl_Flag')
```

In [129…

```
data.nunique()
```

Out[129…

```
Disbursed_Amount                      24565
Asset_Cost                            46252
ltv                                    6579
Manufacturer_ID                          11
Employment_Type                           2
State_ID                                 22
Aadhar_Flag                               2
PAN_Flag                                  2
VoterID_Flag                              2
Driving_Flag                              2
Passport_Flag                             2
Perform_CNS_Score                       573
Perform_CNS_Score_Description             6
PRI_NO_OF_ACCTS                         108
PRI_ACTIVE_ACCTS                         40
PRI_OVERDUE_ACCTS                        22
PRI_CURRENT_BALANCE                   71341
PRI_SANCTIONED_AMOUNT                 44390
PRI_DISBURSED_AMOUNT                  47909
SEC_NO_OF_ACCTS                          37
SEC_ACTIVE_ACCTS                         23
SEC_OVERDUE_ACCTS                         9
SEC_CURRENT_BALANCE                    3246
SEC_SANCTIONED_AMOUNT                  2223
SEC_DISBURSED_AMOUNT                   2553
PRIMARY_INSTAL_AMT                    28067
SEC_INSTAL_AMT                         1918
NEW_ACCTS_IN_LAST_SIX_MONTHS             26
DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS      14
NO_OF_INQUIRIES                          25
Loan_Default                              2
Age                                      49
Time_Since_Loan_Dispursed_In_Yrs          1
CREDIT_HISTORY_LENGTH                   294
AVERAGE_ACCT_AGE                        192
dtype: int64
```

In [130…

```
data=data.drop(columns ='Perform_CNS_Score')

data=data.drop(columns ='PRI_NO_OF_ACCTS')

data=data.drop(columns ='SEC_NO_OF_ACCTS')
```

In [131…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 233154 entries, 0 to 233153
Data columns (total 32 columns):
 #   Column                               Non-Null Count   Dtype  
---  ------                               --------------   -----  
 0   Disbursed_Amount                     233154 non-null  int64  
 1   Asset_Cost                           233154 non-null  int64  
 2   ltv                                  233154 non-null  float64
 3   Manufacturer_ID                      233154 non-null  int64  
 4   Employment_Type                      233154 non-null  object 
 5   State_ID                             233154 non-null  int64  
 6   Aadhar_Flag                          233154 non-null  int64  
 7   PAN_Flag                             233154 non-null  int64  
 8   VoterID_Flag                         233154 non-null  int64  
 9   Driving_Flag                         233154 non-null  int64  
 10  Passport_Flag                        233154 non-null  int64  
 11  Perform_CNS_Score_Description        233154 non-null  int64  
 12  PRI_ACTIVE_ACCTS                     233154 non-null  int64  
 13  PRI_OVERDUE_ACCTS                    233154 non-null  int64  
 14  PRI_CURRENT_BALANCE                  233154 non-null  int64  
 15  PRI_SANCTIONED_AMOUNT                233154 non-null  int64  
 16  PRI_DISBURSED_AMOUNT                 233154 non-null  int64  
 17  SEC_ACTIVE_ACCTS                     233154 non-null  int64  
 18  SEC_OVERDUE_ACCTS                    233154 non-null  int64  
 19  SEC_CURRENT_BALANCE                  233154 non-null  int64  
 20  SEC_SANCTIONED_AMOUNT                233154 non-null  int64  
 21  SEC_DISBURSED_AMOUNT                 233154 non-null  int64  
 22  PRIMARY_INSTAL_AMT                   233154 non-null  int64  
 23  SEC_INSTAL_AMT                       233154 non-null  int64  
 24  NEW_ACCTS_IN_LAST_SIX_MONTHS         233154 non-null  int64  
 25  DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS  233154 non-null  int64  
 26  NO_OF_INQUIRIES                      233154 non-null  int64  
 27  Loan_Default                         233154 non-null  int64  
 28  Age                                  233154 non-null  float64
 29  Time_Since_Loan_Dispursed_In_Yrs     233154 non-null  float64
 30  CREDIT_HISTORY_LENGTH                233154 non-null  float64
 31  AVERAGE_ACCT_AGE                     233154 non-null  float64
dtypes: float64(5), int64(26), object(1)
memory usage: 56.9+ MB
```

In [132…

```
data['Employment_Type'] = Le.fit_transform(data['Employment_Type'])
data['Employment_Type']
```

Out[132…

```
0         0
1         1
2         1
3         0
4         1
         ..
233149    1
233150    1
233151    1
233152    1
233153    1
Name: Employment_Type, Length: 233154, dtype: int64
```

Feature Selection

In [133…

```
data.head()
```

Out[133…

	Disbursed_Amount	Asset_Cost	ltv	Manufacturer_ID	Employment_Type	State_ID	A
0	50578	58400	89.55	0	0	5	
1	53278	61360	89.63	0	1	5	
2	52378	60300	88.39	0	1	5	
3	46349	61500	76.42	0	0	5	
4	43594	78256	57.50	5	1	5	

5 rows × 32 columns

Split the data into training and test set in the ratio of 70:30 respectively

In [134…

```
X = data.loc[:, data.columns != 'Loan_Default']  # independent variables
y = data.loc[:, data.columns == 'Loan_Default']  # Target variable
```

In [135…

```
X = pd.get_dummies(X,drop_first=True)
```

In [136…

```
y.head()
```

Out[136…

	Loan_Default
0	0
1	0
2	1
3	0
4	0

In [137…

```
X.head()
```

Out[137...

	Disbursed_Amount	Asset_Cost	ltv	Manufacturer_ID	Employment_Type	State_ID	A
0	50578	58400	89.55	0	0	5	
1	53278	61360	89.63	0	1	5	
2	52378	60300	88.39	0	1	5	
3	46349	61500	76.42	0	0	5	
4	43594	78256	57.50	5	1	5	

5 rows × 31 columns

Create the training and test data set in the ratio of 70:30 respectively

In [138...

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,rand
```

In [139...

```
X_train.shape,X_test.shape
```

Out[139...

```
((163207, 31), (69947, 31))
```

In [140...

```
X_train.head()
```

Out[140...

	Disbursed_Amount	Asset_Cost	ltv	Manufacturer_ID	Employment_Type	State_
196231	59947	68000	89.71	5	1	
87386	56259	66216	87.59	5	1	
80118	47549	62240	78.73	3	1	
94938	66169	105104	63.75	3	1	
20441	57853	76195	81.37	5	1	

5 rows × 31 columns

In [141…
```
## importing necessary metrics to evaluate model performance

from sklearn.metrics import confusion_matrix, recall_score, precision_score

# Blanks list to store model name, training score, testing score, recall, p

algo= []
tr = []
te = []
recall = []
precision = []
roc = []
```

Logistic Regression

In [142…
```
# Logistic Regression
from sklearn.linear_model import LogisticRegression
model = LogisticRegression(random_state=7)

model.fit(X_train, y_train)
```

Out[142…
```
LogisticRegression(random_state=7)
```

In [143…
```
model.coef_.round(2)
```

Out[143…
```
array([[ 0., -0., -0., -0., -0., -0., -0., -0.,  0., -0., -0.,  0., -0.,
         0., -0., -0.,  0., -0., -0.,  0., -0.,  0., -0.,  0., -0.,  0.,
         0., -0., -0., -0., -0.]])
```

In [144…
```
model.intercept_.round(2)
```

Out[144…
```
array([-0.])
```

In [145…
```
y_pred_class=model.predict(X_test)
y_pred_prob=model.predict_proba(X_test)
```

In [146…
```
y_pred_class[:20]
```

Out[146…
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

In [147…
```
y_pred_class[:5][:]
```

Out[147…
```
array([0, 0, 0, 0, 0])
```

In [148…
```
y_pred_prob[:5,:]
```

Out[148…
```
array([[0.72924768, 0.27075232],
       [0.77390203, 0.22609797],
       [0.75944397, 0.24055603],
       [0.79513968, 0.20486032],
       [0.93225424, 0.06774576]])
```

In [149…
```
y_pred_prob[:5,0]
```

Out[149…
```
array([0.72924768, 0.77390203, 0.75944397, 0.79513968, 0.93225424])
```

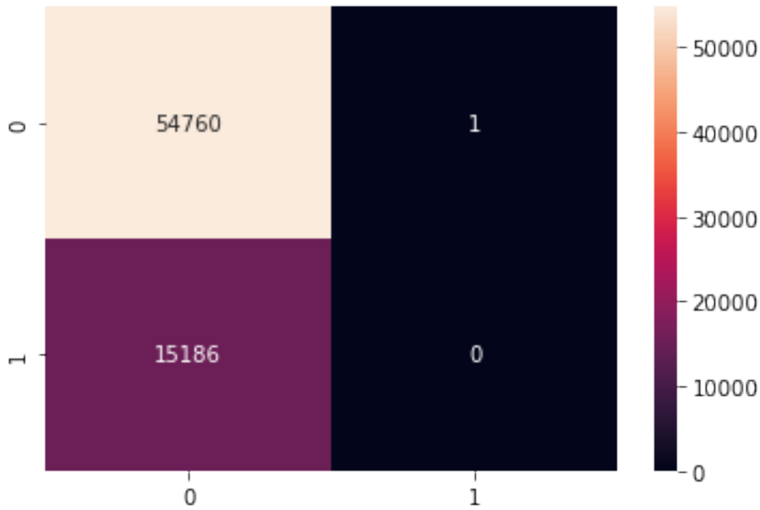
In [150…
```
#y_pred_prob[:20,:]
(y_pred_prob[:5,0]>0.5)*1
```

Out[150…
```
array([1, 1, 1, 1, 1])
```

Confusion Matrix

In [155…
```
## function to get confusion matrix in a proper format
def draw_cm( actual, predicted ):
    cm = confusion_matrix( actual, predicted)
    sns.heatmap(cm, annot=True,  fmt='.0f', xticklabels = [0,1] , yticklabe
```

In [156…
```
draw_cm(y_test,y_pred_class);
```



In [157…
```
(54760 + 0) / (15186+1+0+54760) #Accuracy
```

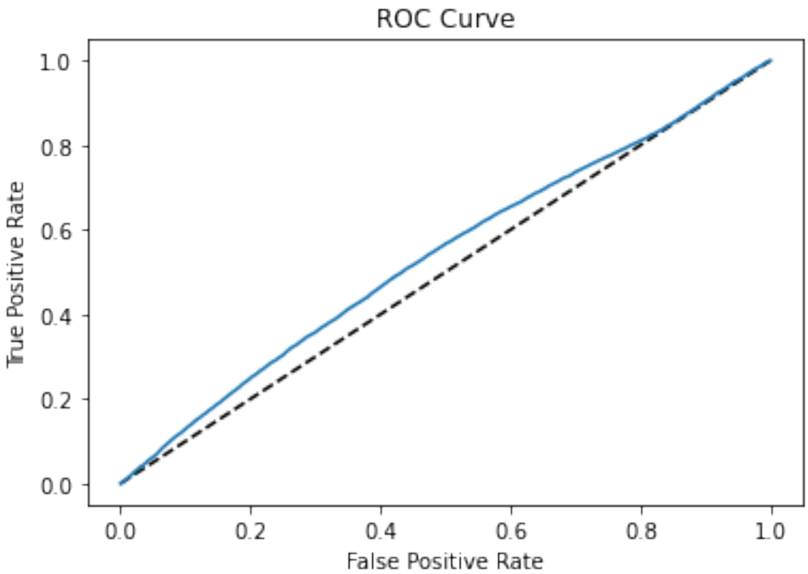
Out[157…
```
0.7828784651235936
```

ROC Curve

In [158…
```
fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob[:,1])
```

In [160…

```
import matplotlib.pyplot as plt
%matplotlib inline
# Plot ROC curve
plt.plot([0, 1], [0, 1], 'k--')
plt.plot(fpr, tpr)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.show()
```



In [161…

```
roc_df=pd.DataFrame([fpr,tpr,thresholds]).T
roc_df.columns=['fpr','tpr','thresholds']
roc_df
```

Out[161…

	fpr	tpr	thresholds
0	0.000000	0.000000	1.990877e+00
1	0.000018	0.000000	9.908767e-01
2	0.000091	0.000000	3.570814e-01
3	0.000091	0.000132	3.563843e-01
4	0.000347	0.000132	3.492578e-01
...	...	...	...
23742	0.999489	0.999802	4.467177e-03
23743	0.999489	0.999934	4.068667e-03
23744	0.999854	0.999934	6.053714e-04
23745	0.999854	1.000000	5.682926e-05
23746	1.000000	1.000000	3.097067e-13

23747 rows × 3 columns

In [163…

```
data.to_excel('data22.xlsx')
```

Dashboarding:

Visualize the data using Tableau to help user explore data to have a better understanding

Demonstrate the variables associated with each other and factors to build a dashboard

https://public.tableau.com/app/profile/rushikesh.khankar/viz/Predi publish=yes