Importing, Understanding, and Inspecting Data

In [55]:
```
# Importing necessary libraries

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE
```

In [10]:
```
data = pd.read_excel('C:/Users/vinay/Desktop/Siplilearn/Data Analyst Masters Cap
print(data.head())
```

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

   manufacturer_id  Current_pincode_ID Date.of.Birth Employment.Type  ...  \
0               45                1441    1984-01-01        Salaried  ...
1               45                1497    1985-08-24   Self employed  ...
2               45                1495    1977-12-09   Self employed  ...
3               45                1502    1988-06-01        Salaried  ...
4               86                1499    1994-07-14   Self employed  ...

  SEC.SANCTIONED.AMOUNT  SEC.DISBURSED.AMOUNT  PRIMARY.INSTAL.AMT  \
0                     0                     0                   0
1                     0                     0                   0
2                     0                     0                   0
3                     0                     0                   0
4                     0                     0                   0

   SEC.INSTAL.AMT  NEW.ACCTS.IN.LAST.SIX.MONTHS  \
0               0                             0
1               0                             0
2               0                             0
3               0                             0
4               0                             0

   DELINQUENT.ACCTS.IN.LAST.SIX.MONTHS  AVERAGE.ACCT.AGE  \
0                                    0         0yrs 0mon
1                                    0         0yrs 0mon
2                                    0         0yrs 0mon
3                                    0         0yrs 0mon
4                                    0         0yrs 0mon

   CREDIT.HISTORY.LENGTH  NO.OF_INQUIRIES  loan_default
0             0yrs 0mon                0             0
1             0yrs 0mon                0             0
2             0yrs 0mon                1             1
3             0yrs 0mon                0             0
4             0yrs 0mon                0             0

[5 rows x 41 columns]
```

In [27]:
```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 233154 entries, 0 to 233153
Data columns (total 41 columns):
 #   Column                               Non-Null Count   Dtype
---  ------                               --------------   -----
 0   UniqueID                             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  DisbursalDate                        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
None
```

In [8]:
```
missing_values = data.isnull().sum()
print(missing_values)
```

```
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 [29]:
```
duplicates = data.duplicated().sum()
print(duplicates)
```

```
0
```

In [16]:
```
print(data.nunique())
```

```
UniqueID                              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
DisbursalDate                             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
dtype: int64
```

In [31]:
```
missing_col = data.isnull().sum()[data.isnull().sum()>0]
print (missing_col)
```

```
Employment.Type    7661
dtype: int64
```

In [35]:
```
for col in missing_col.index:
    if data[col].dtype == 'object':
        data[col].fillna(data[col].mode()[0], inplace=True)
    elif np.issubdtype(data[col].dtype, np.number):
        data[col].fillna(data[col].median(), inplace=True)
    elif np.issubdtype(data[col].dtype, np.datetime64):
        data[col].fillna(data[col].mode()[0], inplace=True)
```

In [37]:
```
    print(data.isnull().sum().sum())
```

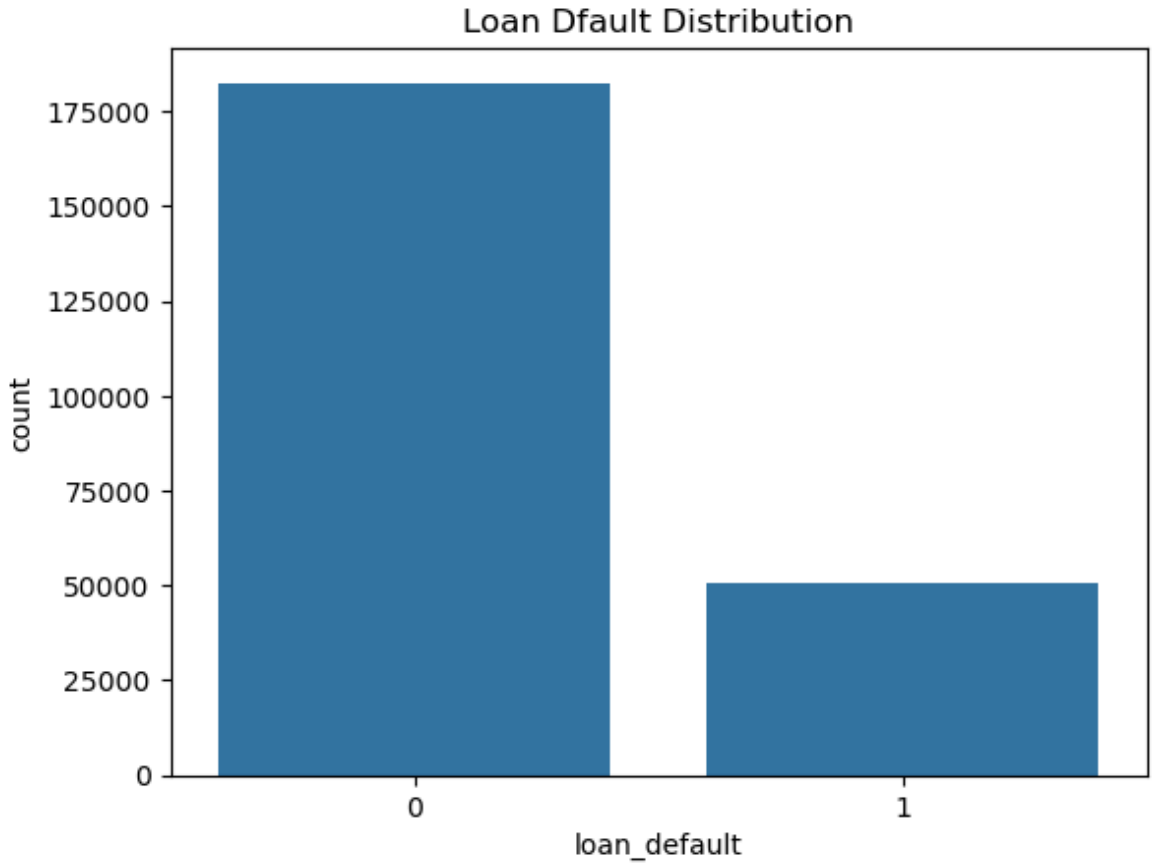
```
0
```

In [39]:
```
key_columns = ['UniqueID', 'loan_default', 'disbursed_amount']
duplicate_rows = data.duplicated(subset=key_columns).sum()
print(duplicate_rows)
```

```
0
```

Performing EDA

In [38]:
```
sns.countplot(x=data['loan_default'])
plt.title('Loan Dfault Distribution')
plt.show()
```



In [41]:
```
print(data['loan_default'].value_counts(normalize=True) * 100)
```

```
loan_default
0    78.292888
1    21.707112
Name: proportion, dtype: float64
```

In [43]:
```
data.describe()
```

Out[43]:

	UniqueID	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
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
std	68315.693711	12971.314171	1.894478e+04	11.456636	69.834995

8 rows × 37 columns

In [45]:
```
print(data['asset_cost'].unique())
```

```
[ 58400  61360  60300 ... 100244 115285  82734]
```

In [47]:
```
print(data['asset_cost'].describe())
```

```
count    2.331540e+05
mean     7.586507e+04
std      1.894478e+04
min      3.700000e+04
25%      6.571700e+04
50%      7.094600e+04
75%      7.920175e+04
max      1.628992e+06
Name: asset_cost, dtype: float64
```

In [49]:
```
print(data['disbursed_amount'].describe())
```

```
count    233154.000000
mean      54356.993528
std       12971.314171
min       13320.000000
25%       47145.000000
50%       53803.000000
75%       60413.000000
max      990572.000000
Name: disbursed_amount, dtype: float64
```

In [51]:
```
print(data.loc[data['asset_cost'] > 1000000, 'asset_cost'])
```

```
198852    1328954
228130    1628992
Name: asset_cost, dtype: int64
```

In [53]:
```
data.columns = (
    data.columns
    .str.lower()
    .str.replace('','_')
    .str.replace('.', '_')
    .str.replace('-','_')
)

print(data.columns)
```

```
Index(['_u_n_i_q_u_e_i_d_', '_d_i_s_b_u_r_s_e_d___a_m_o_u_n_t_',
       '_a_s_s_e_t___c_o_s_t_', '_l_t_v_', '_b_r_a_n_c_h___i_d_',
       '_s_u_p_p_l_i_e_r___i_d_', '_m_a_n_u_f_a_c_t_u_r_e_r___i_d_',
       '_c_u_r_r_e_n_t___p_i_n_c_o_d_e___i_d_', '_d_a_t_e___o_f___b_i_r_t_h_',
       '_e_m_p_l_o_y_m_e_n_t___t_y_p_e_', '_d_i_s_b_u_r_s_a_l_d_a_t_e_',
       '_s_t_a_t_e___i_d_', '_e_m_p_l_o_y_e_e___c_o_d_e___i_d_',
       '_m_o_b_i_l_e_n_o___a_v_l___f_l_a_g_', '_a_a_d_h_a_r___f_l_a_g_',
       '_p_a_n___f_l_a_g_', '_v_o_t_e_r_i_d___f_l_a_g_',
       '_d_r_i_v_i_n_g___f_l_a_g_', '_p_a_s_s_p_o_r_t___f_l_a_g_',
       '_p_e_r_f_o_r_m___c_n_s___s_c_o_r_e_',
       '_p_e_r_f_o_r_m___c_n_s___s_c_o_r_e___d_e_s_c_r_i_p_t_i_o_n_',
       '_p_r_i___n_o___o_f___a_c_c_t_s_', '_p_r_i___a_c_t_i_v_e___a_c_c_t_s_',
       '_p_r_i___o_v_e_r_d_u_e___a_c_c_t_s_',
       '_p_r_i___c_u_r_r_e_n_t___b_a_l_a_n_c_e_',
       '_p_r_i___s_a_n_c_t_i_o_n_e_d___a_m_o_u_n_t_',
       '_p_r_i___d_i_s_b_u_r_s_e_d___a_m_o_u_n_t_',
       '_s_e_c___n_o___o_f___a_c_c_t_s_', '_s_e_c___a_c_t_i_v_e___a_c_c_t_s_',
       '_s_e_c___o_v_e_r_d_u_e___a_c_c_t_s_',
       '_s_e_c___c_u_r_r_e_n_t___b_a_l_a_n_c_e_',
       '_s_e_c___s_a_n_c_t_i_o_n_e_d___a_m_o_u_n_t_',
       '_s_e_c___d_i_s_b_u_r_s_e_d___a_m_o_u_n_t_',
       '_p_r_i_m_a_r_y___i_n_s_t_a_l___a_m_t_',
       '_s_e_c___i_n_s_t_a_l___a_m_t_',
       '_n_e_w___a_c_c_t_s___i_n___l_a_s_t___s_i_x___m_o_n_t_h_s_',
       '_d_e_l_i_n_q_u_e_n_t___a_c_c_t_s___i_n___l_a_s_t___s_i_x___m_o_n_t_h_s_',
       '_a_v_e_r_a_g_e___a_c_c_t___a_g_e_',
       '_c_r_e_d_i_t___h_i_s_t_o_r_y___l_e_n_g_t_h_',
       '_n_o___o_f___i_n_q_u_i_r_i_e_s_', '_l_o_a_n___d_e_f_a_u_l_t_'],
      dtype='object')
```

In [55]:
```
data.columns = data.columns.str.replace('_+','_', regex = True)
```

In [57]:
```
data.columns = data.columns.str.strip('_')
```

In [59]:
```
data
```

Out[59]:

	u_n_i_q_u_e_i_d	d_i_s_b_u_r_s_e_d_a_m_o_u_n_t	a_s_s_e_t_c_o_s_t	l_t_v	b_r_a_n_c
0	420825	50578	58400	89.55	
1	417566	53278	61360	89.63	
2	539055	52378	60300	88.39	
3	529269	46349	61500	76.42	
4	563215	43594	78256	57.50	
...	...	...	...	...	
233149	561031	57759	76350	77.28	
233150	649600	55009	71200	78.72	
233151	603445	58513	68000	88.24	
233152	442948	22824	40458	61.79	
233153	545300	35299	72698	52.27	

233154 rows × 41 columns

In [61]:
```
data.columns = data.columns.str.replace('_', '')
```

In [46]:
```
data
```

Out[46]:

	uniqueid	disbursedamount	assetcost	ltv	branchid	supplierid	manufactur
0	420825	50578	58400	89.55	67	22807	
1	417566	53278	61360	89.63	67	22807	
2	539055	52378	60300	88.39	67	22807	
3	529269	46349	61500	76.42	67	22807	
4	563215	43594	78256	57.50	67	22744	
...	...	...	...	...	...	...	
233149	561031	57759	76350	77.28	5	22289	
233150	649600	55009	71200	78.72	138	17408	
233151	603445	58513	68000	88.24	135	23313	
233152	442948	22824	40458	61.79	160	16212	
233153	545300	35299	72698	52.27	3	14573	

233154 rows × 41 columns

In [63]:
```
data.columns = (
    data.columns
```

```
    .str.lower()
    .str.replace(' ','_')
    .str.replace('.', '_')
    .str.replace('-','_')
)

print(data.columns)
```

```
Index(['uniqueid', 'disbursedamount', 'assetcost', 'ltv', 'branchid',
       'supplierid', 'manufacturerid', 'currentpincodeid', 'dateofbirth',
       'employmenttype', 'disbursaldate', 'stateid', 'employeecodeid',
       'mobilenoavlflag', 'aadharflag', 'panflag', 'voteridflag',
       'drivingflag', 'passportflag', 'performcnsscore',
       'performcnsscoredescription', 'prinoofaccts', 'priactiveaccts',
       'prioverdueaccts', 'pricurrentbalance', 'prisanctionedamount',
       'pridisbursedamount', 'secnoofaccts', 'secactiveaccts',
       'secoverdueaccts', 'seccurrentbalance', 'secsanctionedamount',
       'secdisbursedamount', 'primaryinstalamt', 'secinstalamt',
       'newacctsinlastsixmonths', 'delinquentacctsinlastsixmonths',
       'averageacctage', 'credithistorylength', 'noofinquiries',
       'loandefault'],
      dtype='object')
```

In [65]:
```
import re

def format_column_name(col_name):
    col_name = re.sub(r'(?<!^)(?=[A-Z])', '_', col_name)
    return col_name.lower()

data.columns = [format_column_name(col) for col in data.columns]

print(data.columns)
```

```
Index(['uniqueid', 'disbursedamount', 'assetcost', 'ltv', 'branchid',
       'supplierid', 'manufacturerid', 'currentpincodeid', 'dateofbirth',
       'employmenttype', 'disbursaldate', 'stateid', 'employeecodeid',
       'mobilenoavlflag', 'aadharflag', 'panflag', 'voteridflag',
       'drivingflag', 'passportflag', 'performcnsscore',
       'performcnsscoredescription', 'prinoofaccts', 'priactiveaccts',
       'prioverdueaccts', 'pricurrentbalance', 'prisanctionedamount',
       'pridisbursedamount', 'secnoofaccts', 'secactiveaccts',
       'secoverdueaccts', 'seccurrentbalance', 'secsanctionedamount',
       'secdisbursedamount', 'primaryinstalamt', 'secinstalamt',
       'newacctsinlastsixmonths', 'delinquentacctsinlastsixmonths',
       'averageacctage', 'credithistorylength', 'noofinquiries',
       'loandefault'],
      dtype='object')
```

In [67]:
```
data.rename(columns={
    'uniqueid': 'unique_id',
    'disbursedamount': 'disbursed_amount',
    'assetcost': 'asset_cost',
    'ltv': 'ltv',
    'branchid': 'branch_id',
    'supplierid': 'supplier_id',
    'manufacturerid': 'manufacturer_id',
    'currentpincodeid': 'current_pincode_id',
    'dateofbirth': 'date_of_birth',
    'employmenttype': 'employment_type',
    'disbursaldate': 'disbursal_date',
    'stateid': 'state_id',
```

```
    'employeecodeid': 'employee_code_id',
    'mobilenoavlflag': 'mobile_no_avl_flag',
    'aadharflag': 'aadhar_flag',
    'panflag': 'pan_flag',
    'voteridflag': 'voter_id_flag',
    'drivingflag': 'driving_flag',
    'passportflag': 'passport_flag',
    'performcnsscore': 'perform_cns_score',
    'performcnsscoredescription': 'perform_cns_score_description',
    'prinoofaccts': 'pri_no_of_accts',
    'priactiveaccts': 'pri_active_accts',
    'prioverdueaccts': 'pri_overdue_accts',
    'pricurrentbalance': 'pri_current_balance',
    'prisanctionedamount': 'pri_sanctioned_amount',
    'pridisbursedamount': 'pri_disbursed_amount',
    'secnoofaccts': 'sec_no_of_accts',
    'secactiveaccts': 'sec_active_accts',
    'secoverdueaccts': 'sec_overdue_accts',
    'seccurrentbalance': 'sec_current_balance',
    'secsanctionedamount': 'sec_sanctioned_amount',
    'secdisbursedamount': 'sec_disbursed_amount',
    'primaryinstalamt': 'primary_instal_amt',
    'secinstalamt': 'sec_instal_amt',
    'newacctsinlastsixmonths': 'new_accts_in_last_six_months',
    'delinquentacctsinlastsixmonths': 'delinquent_accts_in_last_six_months',
    'averageacctage': 'average_acct_age',
    'credithistorylength': 'credit_history_length',
    'noofinquiries': 'no_of_inquiries',
    'loandefault': 'loan_default'
}, inplace=True)

print(data.columns)
```

```
Index(['unique_id', 'disbursed_amount', 'asset_cost', 'ltv', 'branch_id',
       'supplier_id', 'manufacturer_id', 'current_pincode_id', 'date_of_birth',
       'employment_type', 'disbursal_date', 'state_id', 'employee_code_id',
       'mobile_no_avl_flag', 'aadhar_flag', 'pan_flag', 'voter_id_flag',
       'driving_flag', 'passport_flag', 'perform_cns_score',
       'perform_cns_score_description', 'pri_no_of_accts', 'pri_active_accts',
       '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 [69]:

```
selected_columns = [
    'disbursed_amount', 'asset_cost', 'ltv', 'perform_cns_score',
    'pri_no_of_accts', 'pri_active_accts', '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', 'no_of_inquiries'
    'loan_default'
]
```

```
data[selected_columns].describe()
```

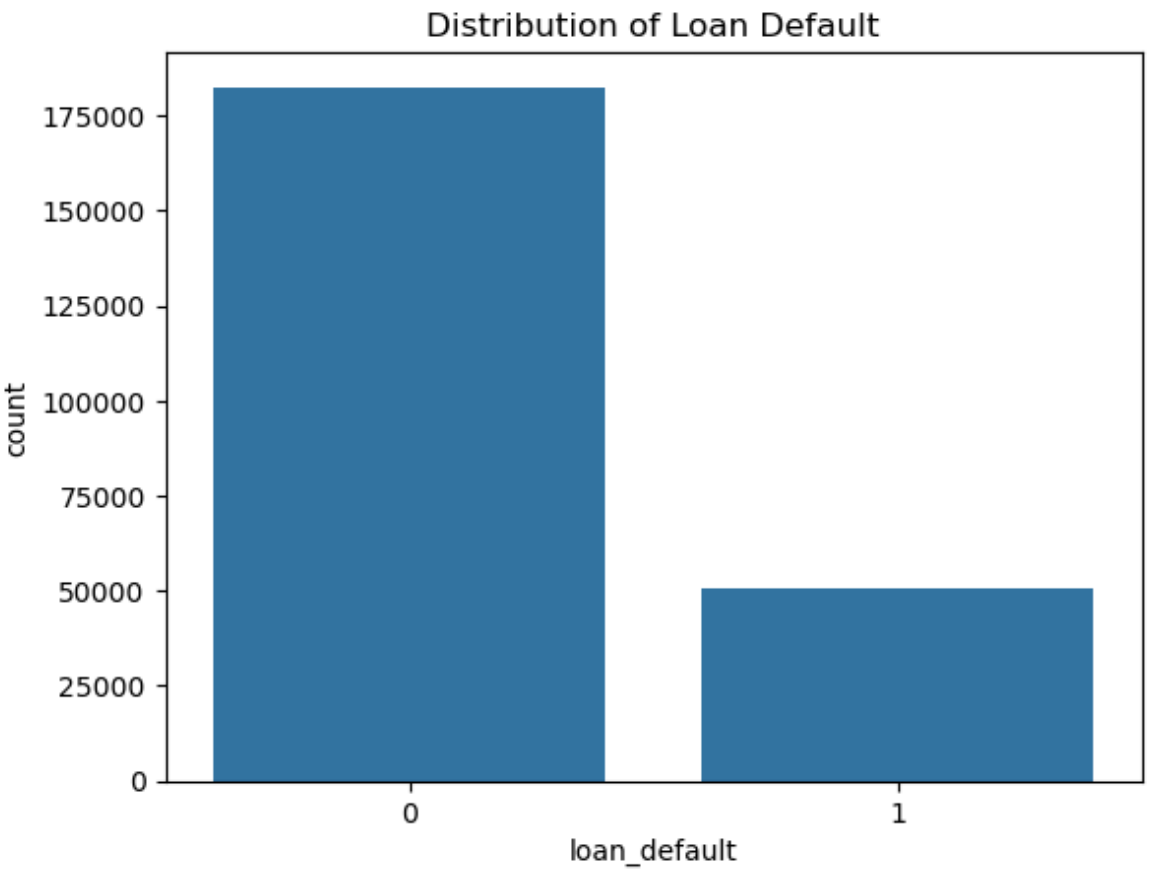
Out[69]:

	disbursed_amount	asset_cost	ltv	perform_cns_score	pri_no_of_ac
count	233154.000000	2.331540e+05	233154.000000	233154.000000	233154.0000
mean	54356.993528	7.586507e+04	74.746530	289.462994	2.4406
std	12971.314171	1.894478e+04	11.456636	338.374779	5.2172
min	13320.000000	3.700000e+04	10.030000	0.000000	0.0000
25%	47145.000000	6.571700e+04	68.880000	0.000000	0.0000
50%	53803.000000	7.094600e+04	76.800000	0.000000	0.0000
75%	60413.000000	7.920175e+04	83.670000	678.000000	3.0000
max	990572.000000	1.628992e+06	95.000000	890.000000	453.0000

8 rows × 22 columns

In [71]:

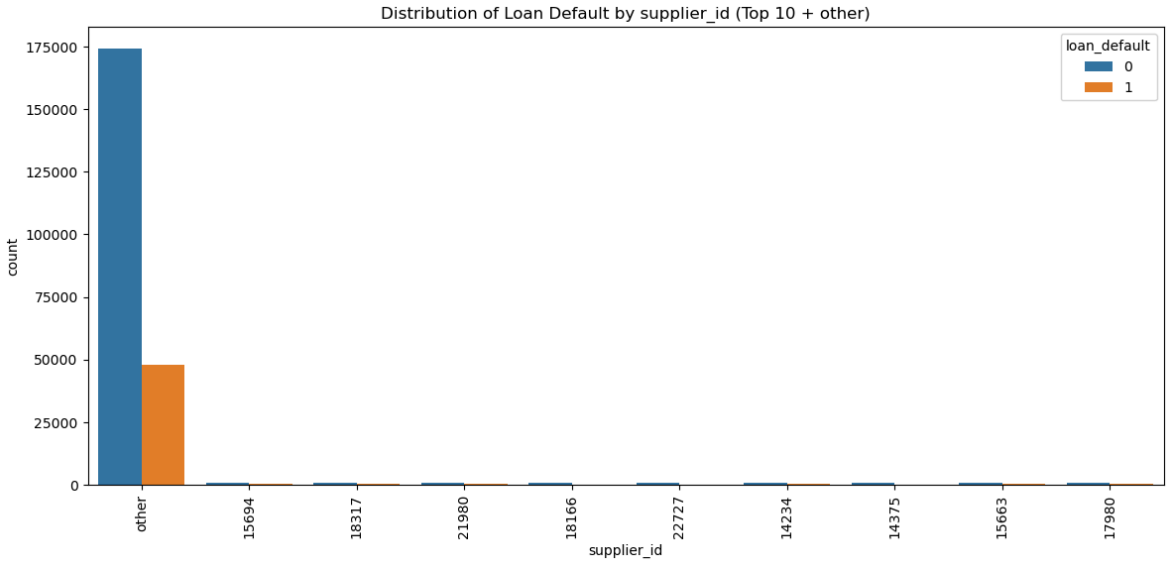
```
sns.countplot(x='loan_default', data = data)
plt.title('Distribution of Loan Default')
plt.show()
```



In [106…

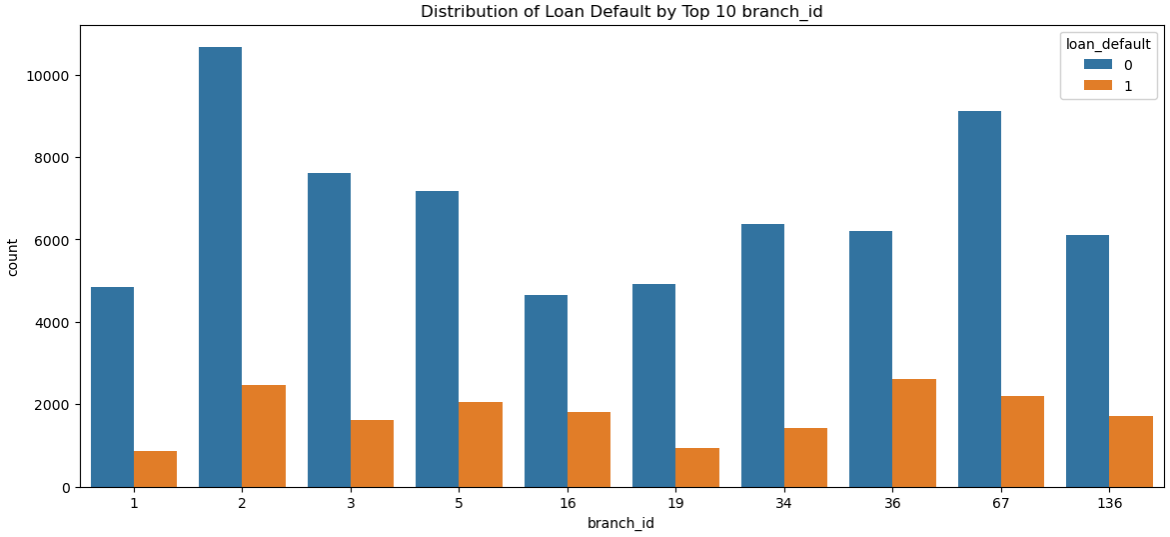
```
top_n = 10
top_categories = data['supplier_id'].value_counts().nlargest(top_n).index
data['supplier_id'] = np.where(data['supplier_id'].isin(top_categories), data['s
```

```
plt.figure(figsize=(14,6))
sns.countplot(x='supplier_id', hue='loan_default', data=data)
plt.title(f'Distribution of Loan Default by supplier_id (Top {top_n} + other)')
plt.xticks(rotation=90)
plt.show()
```



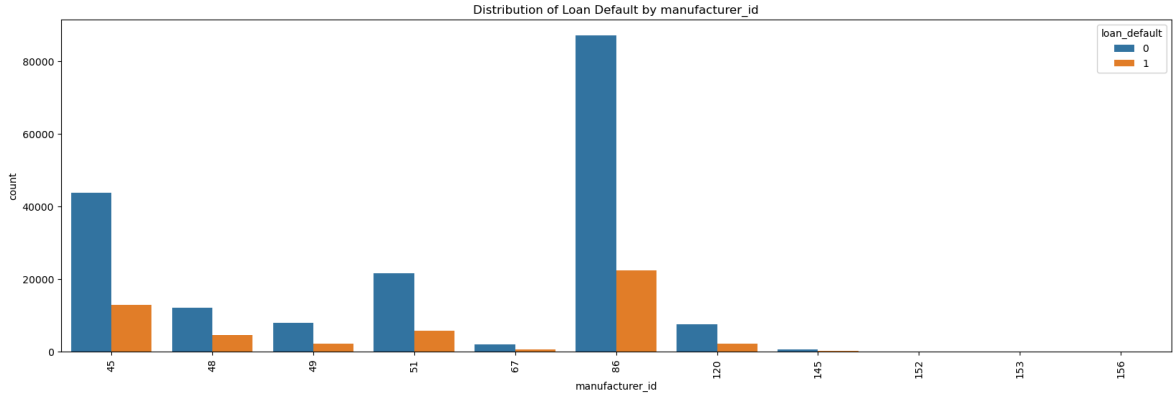
In [108…

```
top_n = 10
top_categories = data['branch_id'].value_counts().nlargest(top_n).index
filtered_data = data[data['branch_id'].isin(top_categories)]

plt.figure(figsize=(14, 6))
sns.countplot(x='branch_id', hue='loan_default', data=filtered_data)
plt.title(f'Distribution of Loan Default by Top {top_n} branch_id')
plt.show()
```

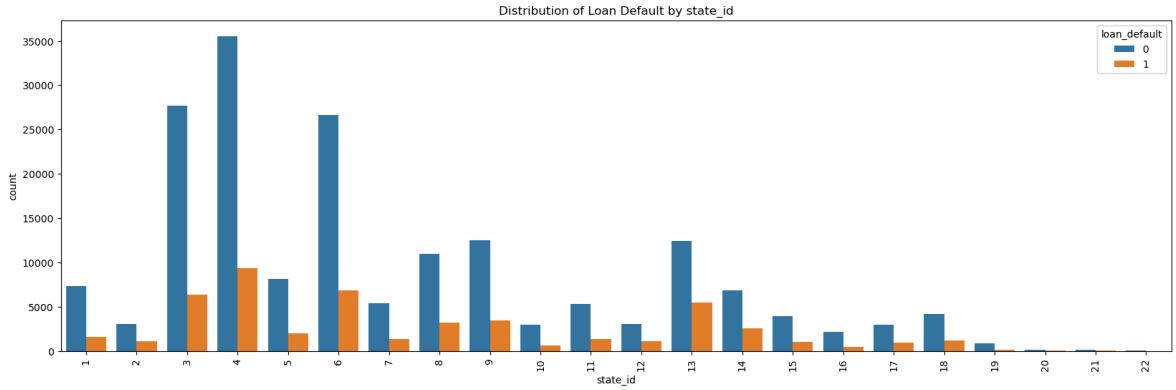


In [114…

```
plt.figure(figsize=(20, 6))
sns.countplot(x='manufacturer_id', hue='loan_default', data=data)
plt.title(f'Distribution of Loan Default by manufacturer_id')
plt.xticks(rotation=90)
plt.show()
```



In [116…
```
plt.figure(figsize=(20, 6))
sns.countplot(x='state_id', hue='loan_default', data=data)
plt.title('Distribution of Loan Default by state_id')
plt.xticks(rotation=90)
plt.show()
```



In [120…
```
data.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone\Ba
```

In [28]:
```
data = pd.read_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Caps
```

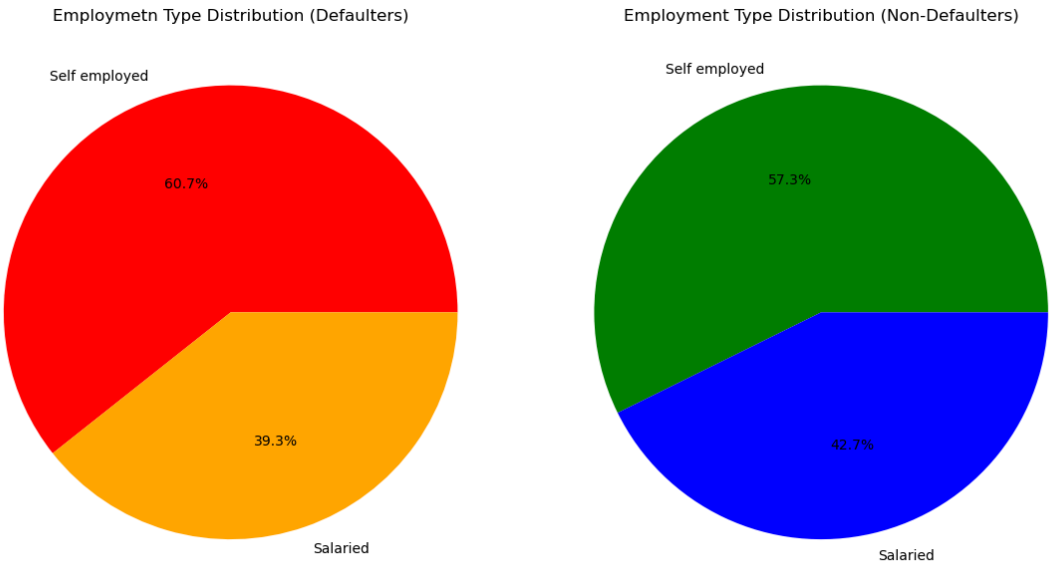
In [30]:
```
employment_types = data['employment_type'].unique()

print(employment_types)
```

```
['Salaried' 'Self employed']
```

In [32]:
```
missing_values = data['employment_type'].isnull().sum()

print(missing_values)
```

```
0
```

In [34]:
```
defaulters = data[data['loan_default'] == 1]['employment_type'].value_counts()
non_defaulters = data[data['loan_default'] == 0]['employment_type'].value_counts

fig, axes = plt.subplots(1, 2, figsize=(12,6))

axes[0].pie(defaulters, labels=defaulters.index, autopct='%1.1f%%', colors=['red
axes[0].set_title('Employmetn Type Distribution (Defaulters)')

axes[1].pie(non_defaulters, labels=non_defaulters.index, autopct='%1.1f%%', colo
axes[1].set_title('Employment Type Distribution (Non-Defaulters)')

plt.tight_layout()
plt.show()
```



In [44]:
```
data['date_of_birth'] = pd.to_datetime(data['date_of_birth'], errors='coerce')

data['age'] = (pd.to_datetime('today')- data['date_of_birth']).dt.days // 365

print(data[['date_of_birth', 'age']].head())
```

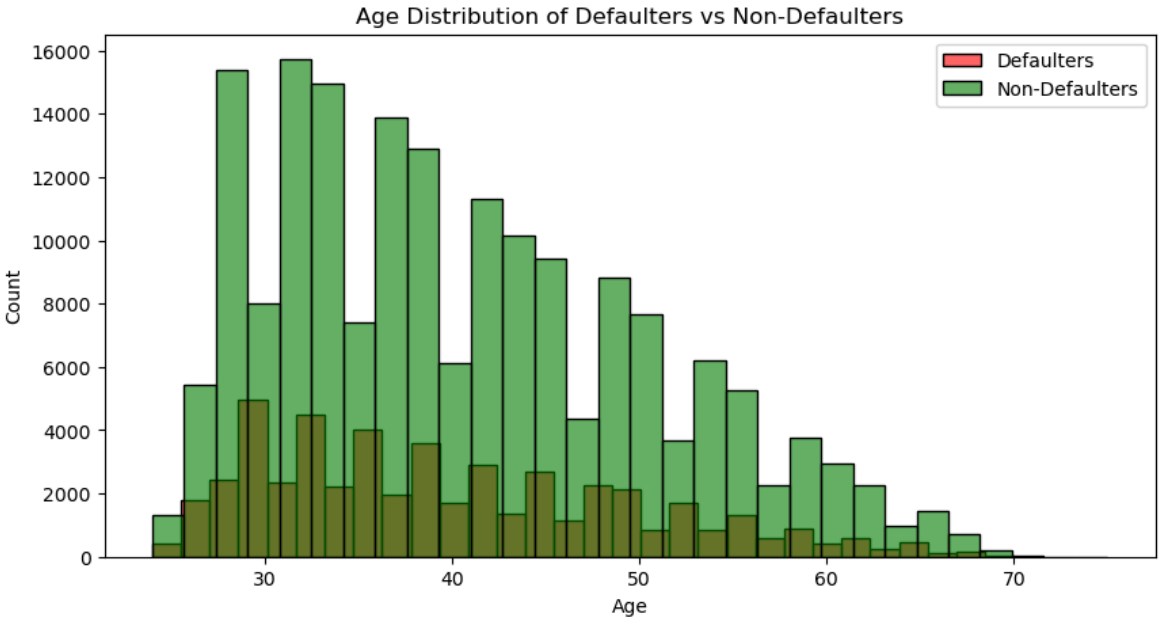
```
  date_of_birth  age
0    1984-01-01   41
1    1985-08-24   39
2    1977-12-09   47
3    1988-06-01   36
4    1994-07-14   30
```

In [52]:
```
missing_ages = data['age'].isnull().sum()
print(missing_ages)

invalid_ages = data[(data['age'] <18) | (data['age'] >100)]
print(invalid_ages['date_of_birth'])
```

```
0
Series([], Name: date_of_birth, dtype: datetime64[ns])
```

In [58]:
```
plt.figure(figsize=(10,5))
sns.histplot(data[data['loan_default'] == 1]['age'], bins=30, color='red', alpha
sns.histplot(data[data['loan_default'] == 0]['age'], bins=30, color='green', alp

plt.title('Age Distribution of Defaulters vs Non-Defaulters')
```

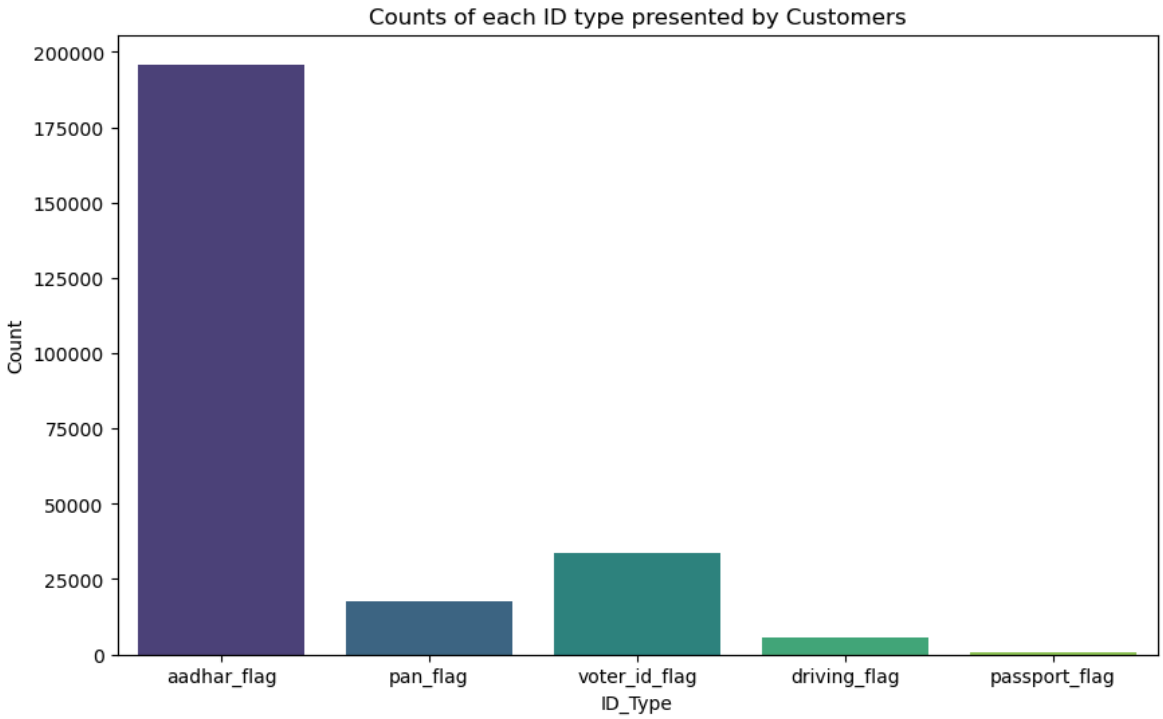
```
plt.xlabel('Age')
plt.ylabel('Count')
plt.legend()
plt.show()
```



In [74]:
```
id_columns = ['aadhar_flag', 'pan_flag', 'voter_id_flag', 'driving_flag', 'passp

id_counts = pd.DataFrame(index=id_columns, columns=['count'])

for col in id_columns:
    id_counts.loc[col, 'count'] = data[col].sum()

print(id_counts)
```

```
                count
aadhar_flag    195924
pan_flag        17621
voter_id_flag   33794
driving_flag     5419
passport_flag     496
```

In [98]:
```
plt.figure(figsize=(10,6))
sns.barplot(x=id_counts.index, y=id_counts['count'], hue=id_counts.index, palett

plt.title('Counts of each ID type presented by Customers')
plt.xlabel('ID_Type')
plt.ylabel('Count')

plt.show()
```



In [104…
```
credit_bureau_cols = data.filter(like='Credit Bureau', axis=1)

print(credit_bureau_cols)
```

```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 4
1, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 6
1, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 8
1, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, ...]

[233154 rows x 0 columns]
```

In [106…
```
missing_scores = data['perform_cns_score'].isnull().sum()
print(missing_scores)
```

```
0
```

In [120…
```
invalid_scores = data[(data['perform_cns_score'] < 300) | (data['perform_cns_sco
print(invalid_scores['perform_cns_score'])
```

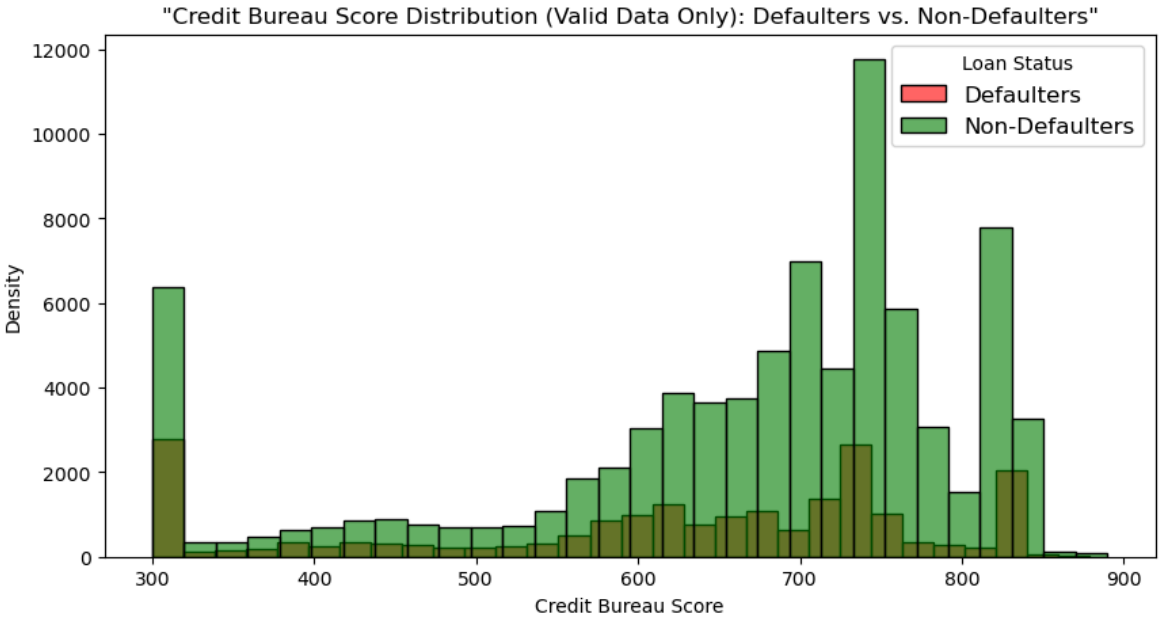
```
0          0
1          0
2          0
3          0
4          0
          ..
233149    14
233150    14
233151    11
233152    11
233153    11
Name: perform_cns_score, Length: 129785, dtype: int64
```

In [122…
```
data.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone\Ba
```

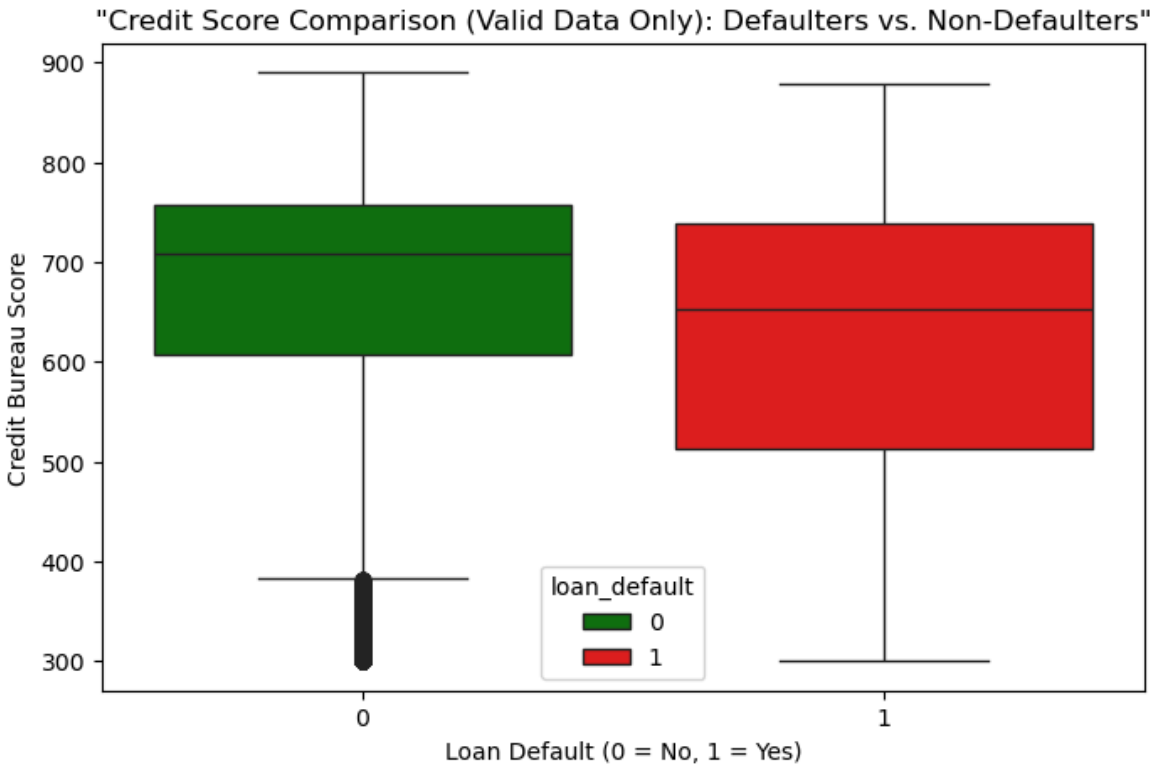
In [126…
```
data_v2 = data[(data['perform_cns_score'] >= 300) & (data['perform_cns_score'] <
print(data.shape)
print(data_v2.shape)
```

```
(233154, 42)
(103369, 42)
```

In [143…
```
plt.figure(figsize=(10,5))
sns.histplot(data_v2[data_v2['loan_default'] ==1]['perform_cns_score'], bins=30,
sns.histplot(data_v2[data_v2['loan_default'] ==0]['perform_cns_score'], bins=30,

plt.title('"Credit Bureau Score Distribution (Valid Data Only): Defaulters vs. N
plt.xlabel('Credit Bureau Score')
plt.ylabel('Density')
plt.legend(title="Loan Status", fontsize=12)
plt.show()
```



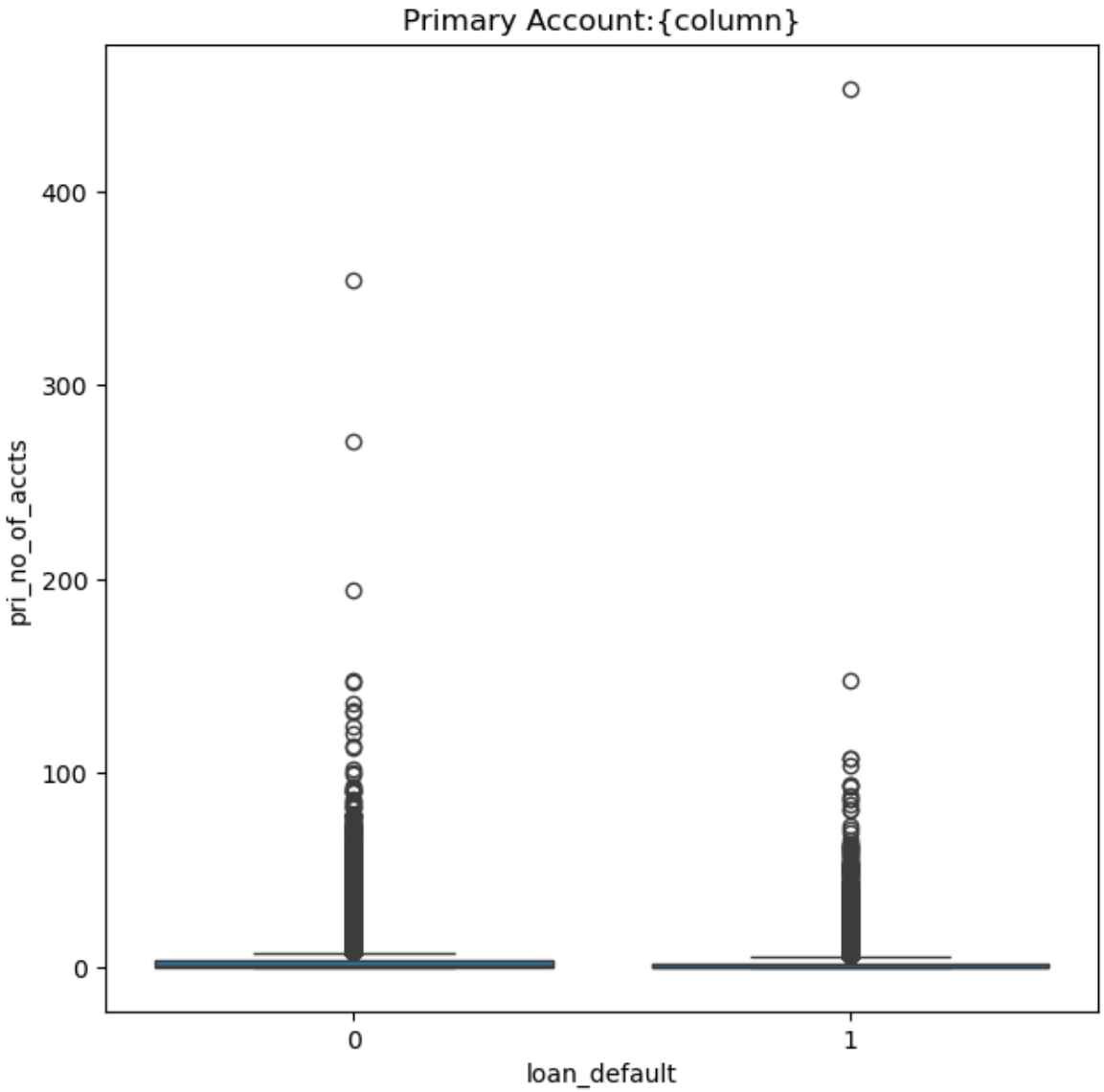
In [147…
```
plt.figure(figsize=(8,5))
sns.boxplot(x='loan_default', y='perform_cns_score', data= data_v2,hue ='loan_de

plt.title('"Credit Score Comparison (Valid Data Only): Defaulters vs. Non-Defaul
plt.xlabel('Loan Default (0 = No, 1 = Yes)')
plt.ylabel('Credit Bureau Score')
plt.show()
```

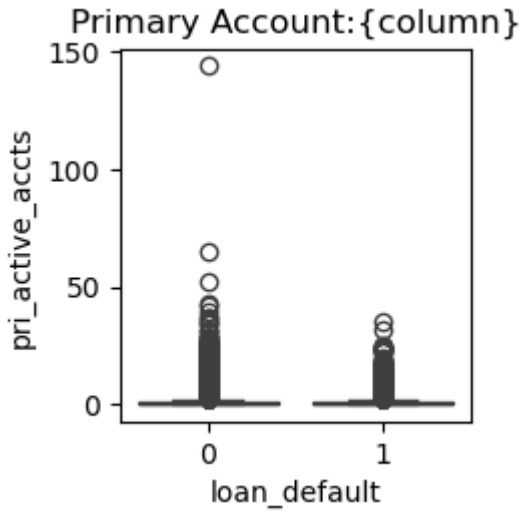


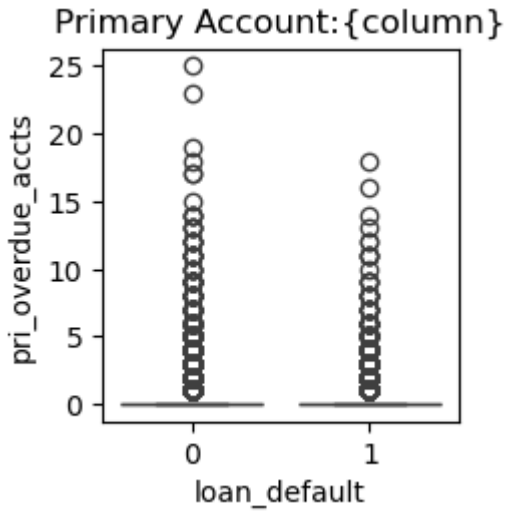
In [159…
```
data.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone\Ba
data_v2.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone
```

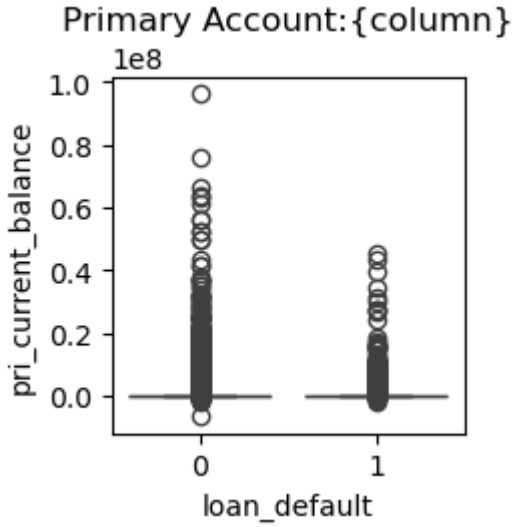
In [6]:
```
data = pd.read_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Caps
data_v2 = pd.read_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters C
```

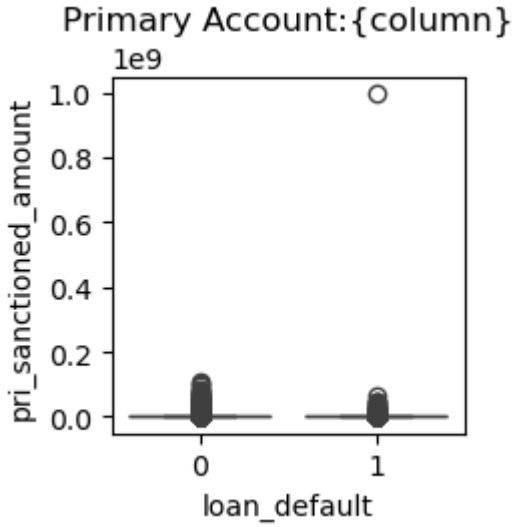
In [14]:
```
primary_account_columns = ['pri_no_of_accts', 'pri_active_accts', 'pri_overdue_a
secondary_account_columns = ['sec_no_of_accts', 'sec_active_accts', 'sec_overdue

plt.figure(figsize=(18,12))
for i, column in enumerate(primary_account_columns, 1):
    plt.subplot(2,3,1)
    sns.boxplot(x='loan_default', y=column, data=data)
    plt.title('Primary Account:{column}')
    plt.tight_layout()
    plt.show()

plt.figure(figsize=(18,12))
for i, column in enumerate(secondary_account_columns, 1):
    plt.subplot(2,3,1)
    sns.boxplot(x='loan_default', y=column, data=data)
    plt.title('secondary Account:{column}')
    plt.tight_layout()
    plt.show()
```

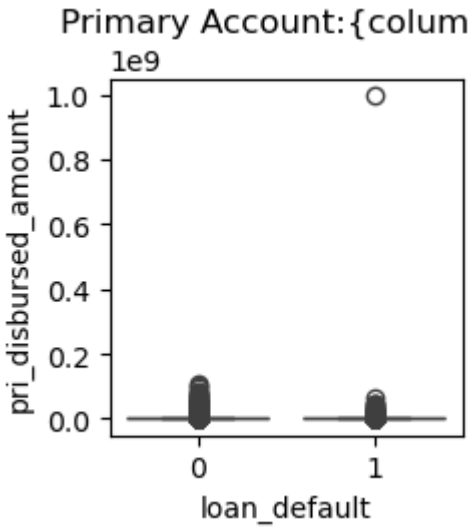

Primary Account:{column}
pri_no_of_accts
400
300
200
100
0
0
1
loan_default

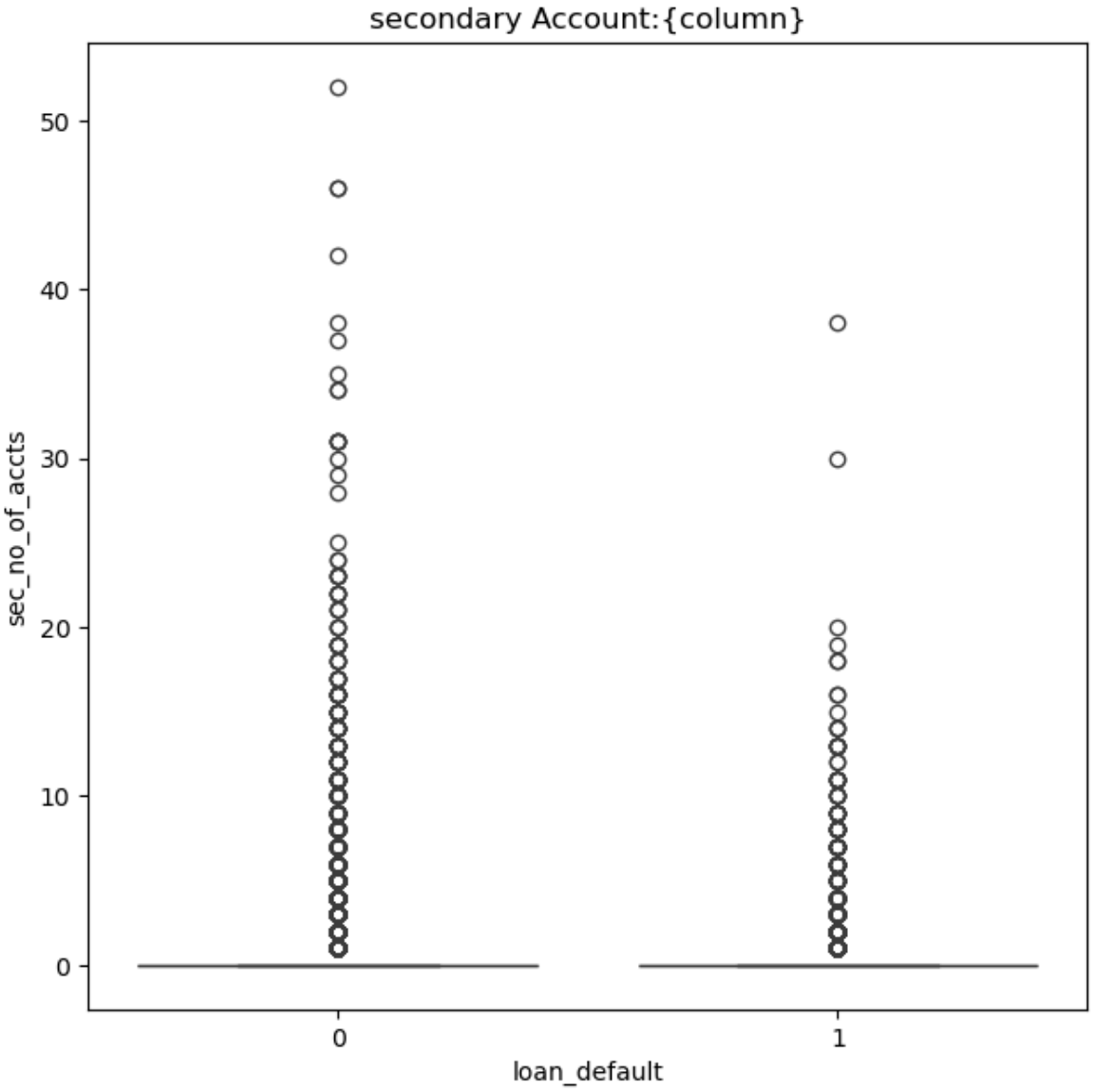

Primary Account:{column}
pri_active_accts
150
100
50
0
0
1
loan_default

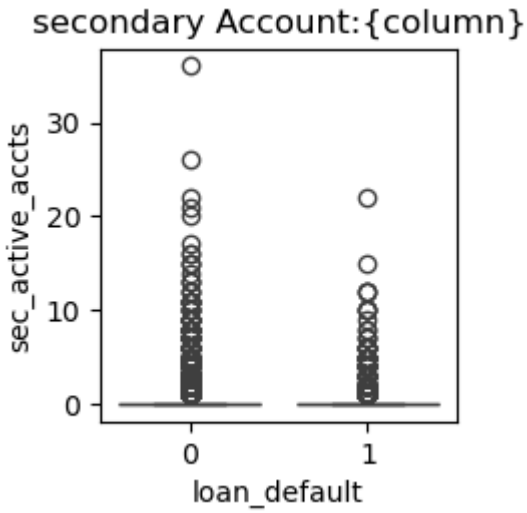

Primary Account:{column}
pri_overdue_accts
25
20
15
10
5
0
0
1
loan_default

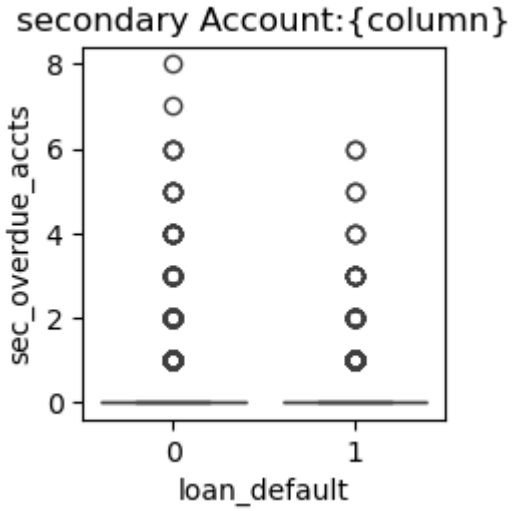

Primary Account:{column}
1e8
pri_current_balance
1.0
0.8
0.6
0.4
0.2
0.0
0
1
loan_default

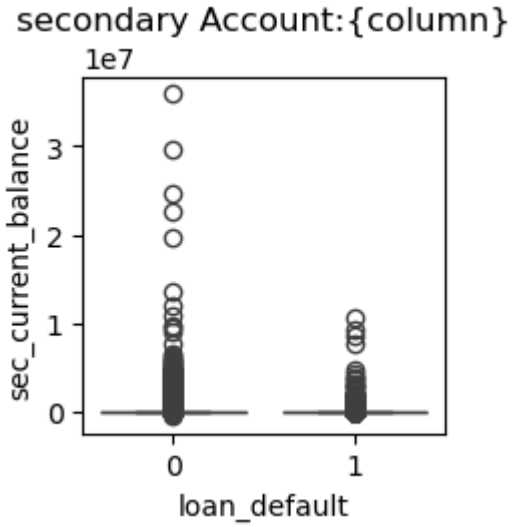

Primary Account:{column}
1e9
pri_sanctioned_amount
1.0
0.8
0.6
0.4
0.2
0.0
0
1
loan_default

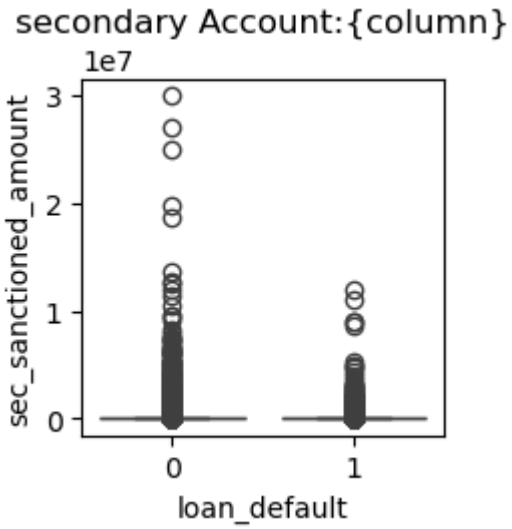

Primary Account:{column}
1e9
1.0
0.8
0.6
0.4
0.2
0.0
pri_disbursed_amount
0
1
loan_default

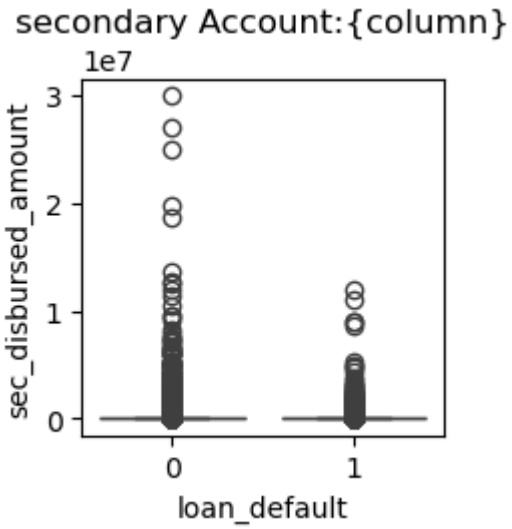

secondary Account:{column}
50
40
30
20
10
0
sec_no_of_accts
0
1
loan_default


secondary Account:{column}
sec_active_accts
30
20
10
0
0
1
loan_default

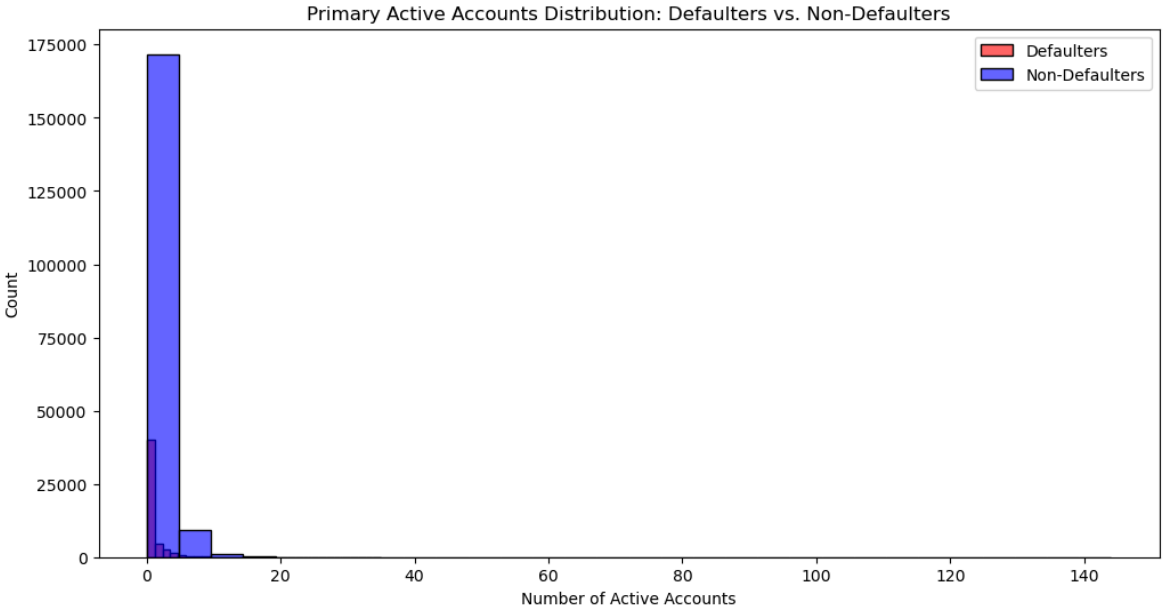

secondary Account:{column}
sec_overdue_accts
8
6
4
2
0
0
1
loan_default


secondary Account:{column}
1e7
sec_current_balance
3
2
1
0
0
1
loan_default





In [28]:
```
plt.figure(figsize=(12,6))

sns.histplot(data[data['loan_default'] == 1]['pri_active_accts'], bins=30, color
sns.histplot(data[data['loan_default'] == 0]['pri_active_accts'], bins=30, color

plt.title("Primary Active Accounts Distribution: Defaulters vs. Non-Defaulters")
plt.xlabel("Number of Active Accounts")
plt.ylabel("Count")
plt.legend()

plt.show()
```



In [40]:
```
pd.set_option('display.float_format', '{:.0f}'.format)
loan_amount_summary = data[['pri_sanctioned_amount', 'pri_disbursed_amount', 'se
print(loan_amount_summary)
```

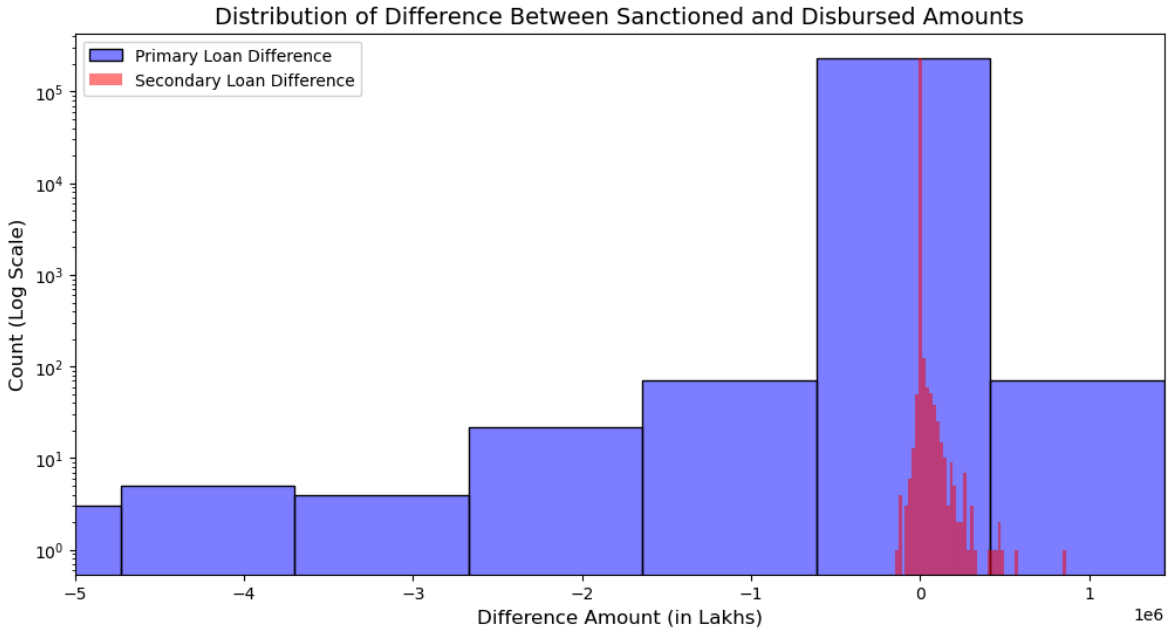
	pri_sanctioned_amount	pri_disbursed_amount	sec_sanctioned_amount	\
count	233154	233154	233154	
mean	218504	218066	7296	
std	2374794	2377744	183156	
min	0	0	0	
25%	0	0	0	
50%	0	0	0	
75%	62500	60800	0	
max	1000000000	1000000000	30000000	

	sec_disbursed_amount
count	233154
mean	7180
std	182593
min	0
25%	0
50%	0
75%	0
max	30000000

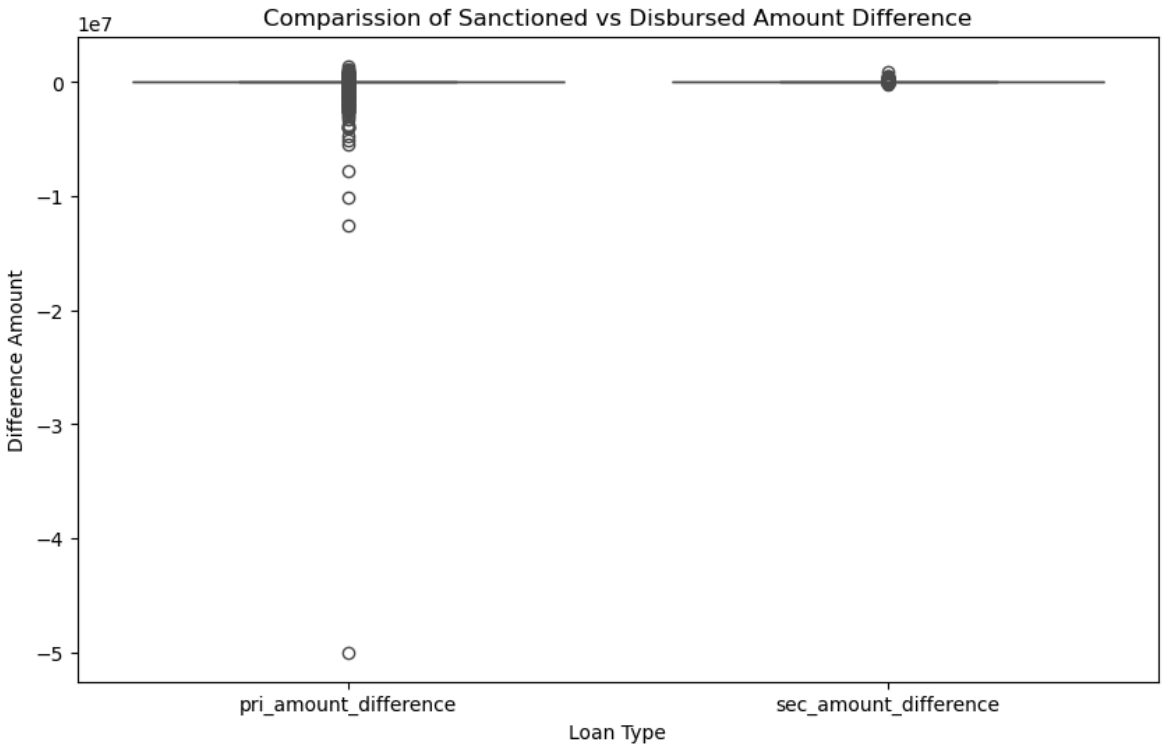
In [54]:
```
data['pri_amount_difference'] = data['pri_sanctioned_amount'] - data['pri_disbur
data['sec_amount_difference'] = data['sec_sanctioned_amount'] - data['sec_disbur

amount_diff_summary = data[['pri_amount_difference', 'sec_amount_difference']].d
print(amount_diff_summary)
```

	pri_amount_difference	sec_amount_difference
count	233154	233154
mean	438	116
std	118979	4896
min	-50000000	-149432
25%	0	0
50%	0	0
75%	0	0
max	1444196	865353

In [70]:
```
plt.figure(figsize=(12,6))

x_min = max(data[['pri_amount_difference', 'sec_amount_difference']].min().min()
x_max = min(data[['pri_amount_difference', 'sec_amount_difference']].max().max()

sns.histplot(data['pri_amount_difference'], bins=50, color='blue', alpha=0.5, la

sns.histplot(data['sec_amount_difference'], bins=50, color='red', alpha=0.5, lab

plt.title("Distribution of Difference Between Sanctioned and Disbursed Amounts",
plt.xlabel("Difference Amount (in Lakhs)", fontsize=12)
plt.ylabel("Count (Log Scale)", fontsize=12)

plt.xlim(x_min, x_max)
plt.yscale("log")
plt.legend()
plt.show()
```



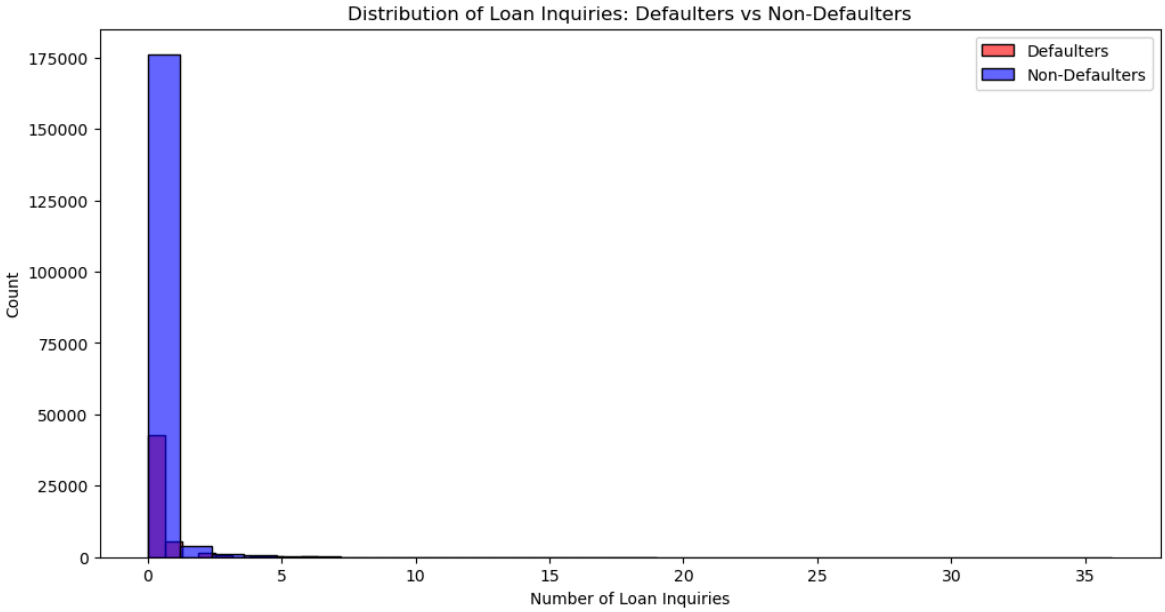
In [72]:
```
plt.figure(figsize=(10,6))

sns.boxplot(data=data[['pri_amount_difference', 'sec_amount_difference']], palet

plt.title('Comparission of Sanctioned vs Disbursed Amount Difference')
plt.xlabel('Loan Type')
plt.ylabel('Difference Amount')

plt.show()
```



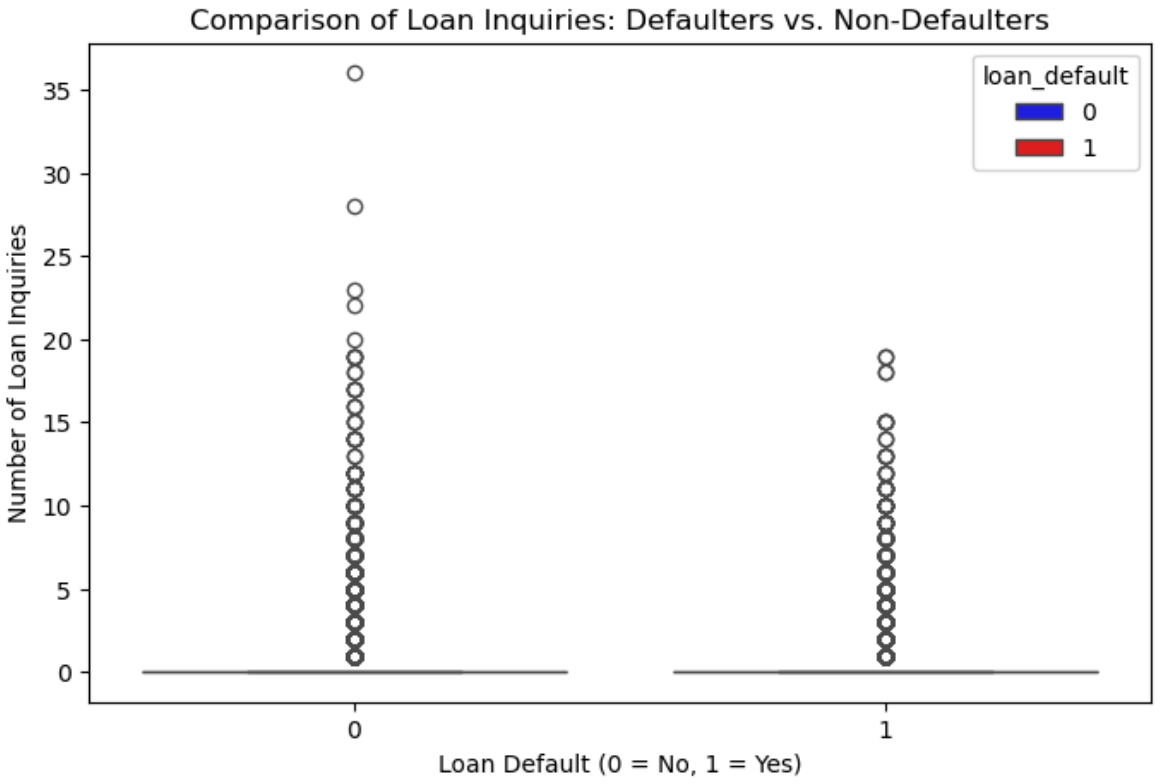
In [76]:
```
inquiry_summary = data.groupby('loan_default')['no_of_inquiries'].describe()
print(inquiry_summary)
```

```
              count  mean  std  min  25%  50%  75%  max
loan_default
0            182543     0    1    0    0    0    0   36
1             50611     0    1    0    0    0    0   19
```

In [82]:
```
plt.figure(figsize=(12,6))

sns.histplot(data[data['loan_default'] == 1]['no_of_inquiries'], bins=30, color=
sns.histplot(data[data['loan_default'] == 0]['no_of_inquiries'], bins=30, color=

plt.title('Distribution of Loan Inquiries: Defaulters vs Non-Defaulters')
plt.xlabel('Number of Loan Inquiries')
plt.ylabel('Count')
plt.legend()
plt.show()
```



In [94]:
```
plt.figure(figsize=(8,5))

sns.boxplot(x='loan_default', y='no_of_inquiries', data=data, hue='loan_default'

plt.title("Comparison of Loan Inquiries: Defaulters vs. Non-Defaulters")
plt.xlabel("Loan Default (0 = No, 1 = Yes)")
plt.ylabel("Number of Loan Inquiries")
plt.show()
```



In [98]:
```
print(data.dtypes)
```

```
unique_id                              int64
disbursed_amount                       int64
asset_cost                             int64
ltv                                  float64
branch_id                              int64
supplier_id                           object
manufacturer_id                        int64
current_pincode_id                     int64
date_of_birth                         object
employment_type                       object
disbursal_date                        object
state_id                               int64
employee_code_id                       int64
mobile_no_avl_flag                     int64
aadhar_flag                            int64
pan_flag                               int64
voter_id_flag                          int64
driving_flag                           int64
passport_flag                          int64
perform_cns_score                      int64
perform_cns_score_description         object
pri_no_of_accts                        int64
pri_active_accts                       int64
pri_overdue_accts                      int64
pri_current_balance                    int64
pri_sanctioned_amount                  int64
pri_disbursed_amount                   int64
sec_no_of_accts                        int64
sec_active_accts                       int64
sec_overdue_accts                      int64
sec_current_balance                    int64
sec_sanctioned_amount                  int64
sec_disbursed_amount                   int64
primary_instal_amt                     int64
sec_instal_amt                         int64
new_accts_in_last_six_months           int64
delinquent_accts_in_last_six_months    int64
average_acct_age                      object
credit_history_length                 object
no_of_inquiries                        int64
loan_default                           int64
age                                    int64
pri_amount_difference                  int64
sec_amount_difference                  int64
dtype: object
```

In [106…
```
def convert_credit_history_to_months(value):
    if pd.isnull(value) or value == '':
        return None
    parts = value.split()
    years = 0
    months = 0

    for part in parts:
        if 'yrs' in part:
            years = int(part.replace('yrs', ""))
        elif 'mon' in part:
            months = int(part.replace('mon', ""))

    return (years * 12) + months
```

In [110…
```
data['credit_history_length_months'] = data['credit_history_length'].apply(conve

data['credit_history_length_months'] = pd.to_numeric(data['credit_history_length
```

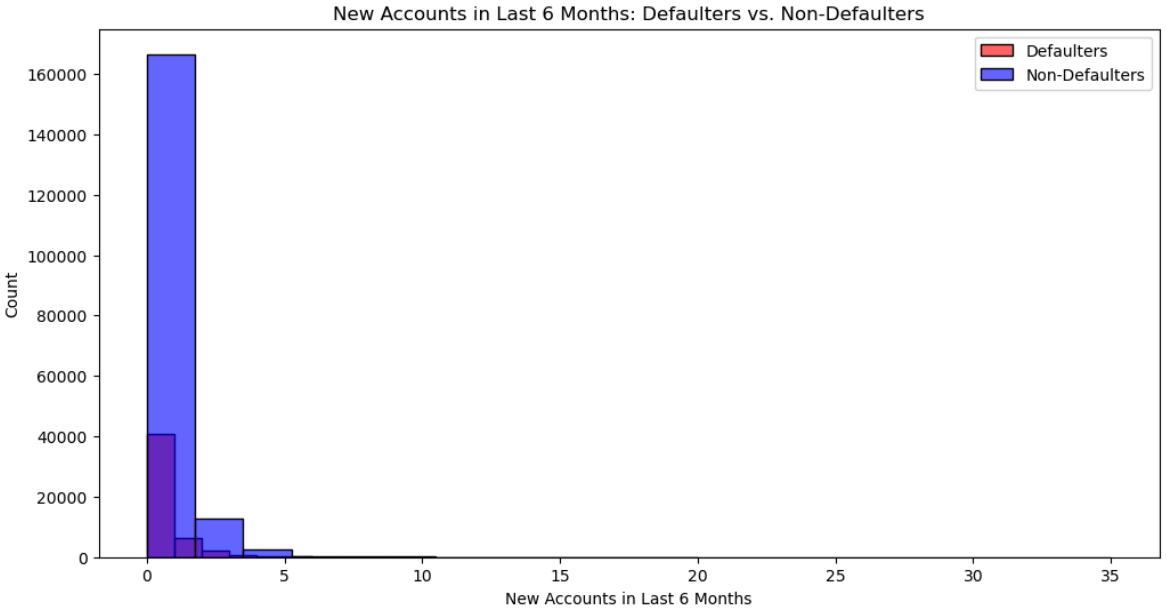
In [112…
```
print(data['credit_history_length_months'])
```

```
0          0
1          0
2          0
3          0
4          0
          ..
233149    28
233150    17
233151    46
233152    38
233153    64
Name: credit_history_length_months, Length: 233154, dtype: int64
```

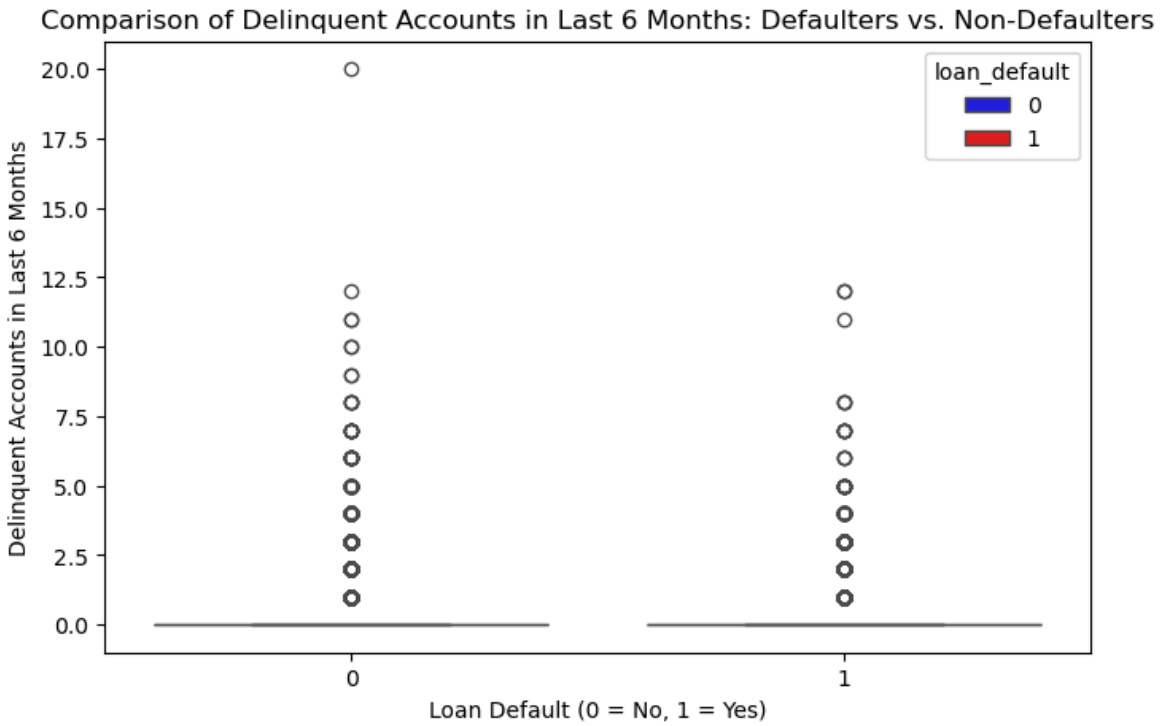
In [120…
```
credit_history_summary = data.groupby('loan_default')[['new_accts_in_last_six_mo
                                                       'delinquent_accts_in_last
                                                       'credit_history_length_mo


print(credit_history_summary)
```

```
             new_accts_in_last_six_months                                \
                                    count mean std min 25% 50% 75% max
loan_default
0                                  182543    0   1   0   0   0   0  35
1                                   50611    0   1   0   0   0   0  20

             delinquent_accts_in_last_six_months       ...          \
                                           count mean  ... 75% max
loan_default                                           ...
0                                         182543    0  ...   0  20
1                                          50611    0  ...   0  12

             credit_history_length_months
                                    count mean std min 25% 50% 75% max
loan_default
0                                  182543   17  29   0   0   0  24 449
1                                   50611   14  26   0   0   0  21 468

[2 rows x 24 columns]
```

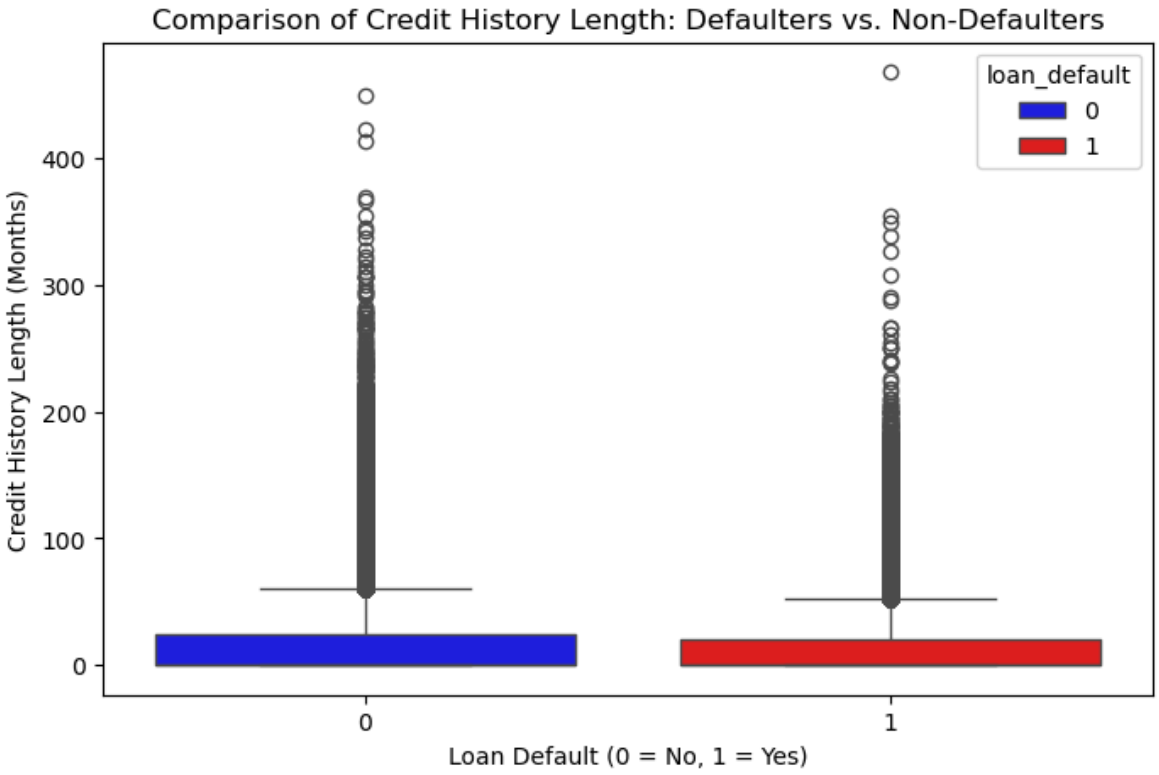
In [139…
```
plt.figure(figsize=(12,6))

sns.histplot(data[data['loan_default'] == 1]['new_accts_in_last_six_months'], bi
sns.histplot(data[data['loan_default'] == 0]['new_accts_in_last_six_months'], bi

plt.title("New Accounts in Last 6 Months: Defaulters vs. Non-Defaulters")
plt.xlabel("New Accounts in Last 6 Months")
plt.ylabel("Count")
plt.legend()
plt.show()
```



In [141…
```
plt.figure(figsize=(8,5))
sns.boxplot(x='loan_default', y='delinquent_accts_in_last_six_months', data=data
plt.title("Comparison of Delinquent Accounts in Last 6 Months: Defaulters vs. No
plt.xlabel("Loan Default (0 = No, 1 = Yes)")
plt.ylabel("Delinquent Accounts in Last 6 Months")
plt.show()
```



In [143…
```
plt.figure(figsize=(8,5))
sns.boxplot(x='loan_default', y='credit_history_length_months', data=data, hue='
plt.title("Comparison of Credit History Length: Defaulters vs. Non-Defaulters")
plt.xlabel("Loan Default (0 = No, 1 = Yes)")
plt.ylabel("Credit History Length (Months)")
plt.show()
```



In [145…
```
print(data.dtypes)
```

```
unique_id                            int64
disbursed_amount                     int64
asset_cost                           int64
ltv                                float64
branch_id                            int64
supplier_id                         object
manufacturer_id                      int64
current_pincode_id                   int64
date_of_birth                       object
employment_type                     object
disbursal_date                      object
state_id                             int64
employee_code_id                     int64
mobile_no_avl_flag                   int64
aadhar_flag                          int64
pan_flag                             int64
voter_id_flag                        int64
driving_flag                         int64
passport_flag                        int64
perform_cns_score                    int64
perform_cns_score_description       object
pri_no_of_accts                      int64
pri_active_accts                     int64
pri_overdue_accts                    int64
pri_current_balance                  int64
pri_sanctioned_amount                int64
pri_disbursed_amount                 int64
sec_no_of_accts                      int64
sec_active_accts                     int64
sec_overdue_accts                    int64
sec_current_balance                  int64
sec_sanctioned_amount                int64
sec_disbursed_amount                 int64
primary_instal_amt                   int64
sec_instal_amt                       int64
new_accts_in_last_six_months         int64
delinquent_accts_in_last_six_months  int64
average_acct_age                    object
credit_history_length               object
no_of_inquiries                      int64
loan_default                         int64
age                                  int64
pri_amount_difference                int64
sec_amount_difference                int64
credit_history_length_months         int64
dtype: object
```

In [150…
```
data.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone\Ba
data_v2.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone
```

In [6]:
```
data = pd.read_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Caps
data_v2 = pd.read_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters C
```

In [63]:
```
features = ['disbursed_amount', 'ltv', 'new_accts_in_last_six_months',
            'delinquent_accts_in_last_six_months', 'credit_history_length_months
            'no_of_inquiries', 'pri_active_accts', 'pri_overdue_accts']

x = data[features]
y = data['loan_default']
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_

scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.fit_transform(x_test)
```

In [65]:
```
model = LogisticRegression(max_iter=1000)
model.fit(x_train, y_train)
```

Out[65]:
```
LogisticRegression
LogisticRegression(max_iter=1000)
```

In [67]:
```
y_pred_prob = model.predict_proba(x_test)[:, 1]
y_pred = (y_pred_prob > 0.2).astype(int)
```

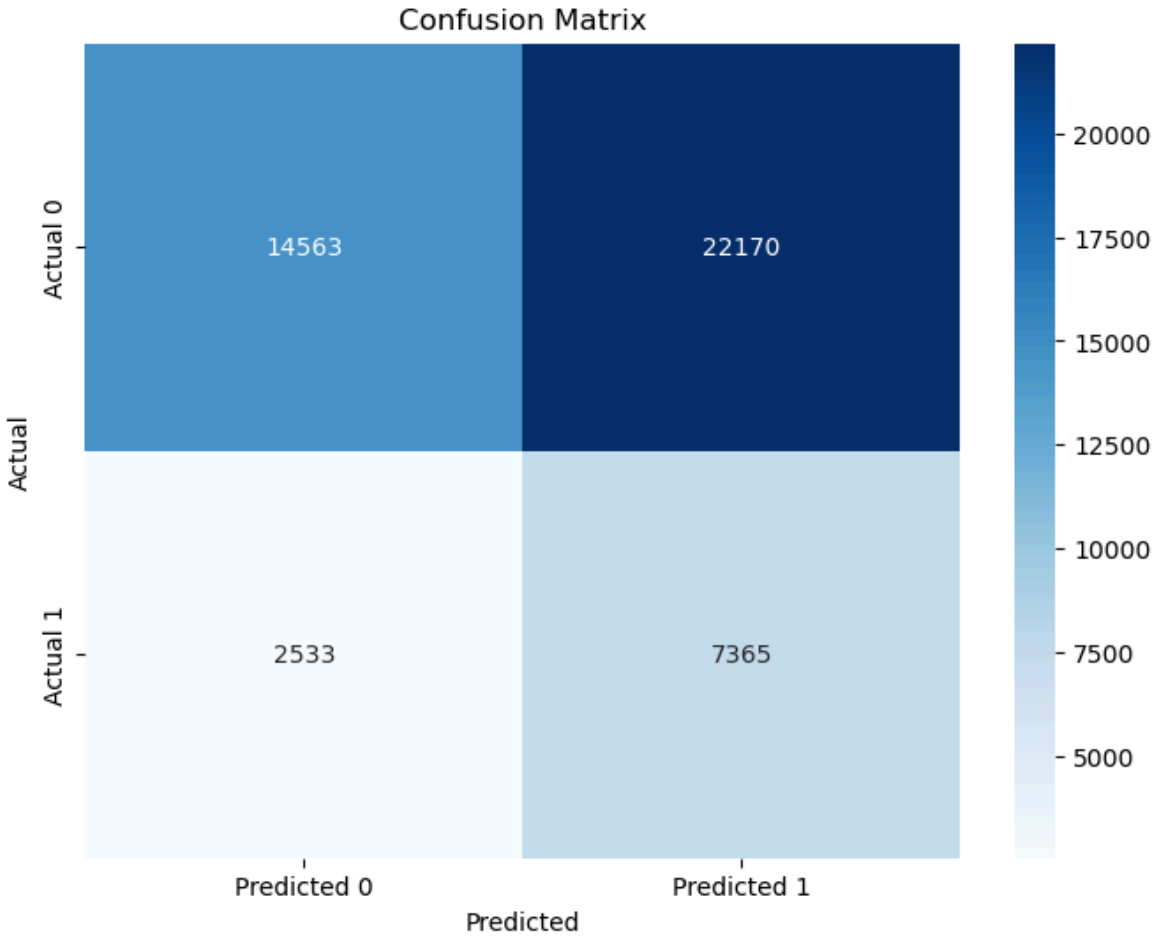
In [69]:
```
conf_matrix = confusion_matrix(y_test, y_pred)
print(conf_matrix)
```

```
[[14563 22170]
 [ 2533  7365]]
```

In [71]:
```
class_report = classification_report(y_test, y_pred)
print(class_report)
```

```
              precision    recall  f1-score   support

           0       0.85      0.40      0.54     36733
           1       0.25      0.74      0.37      9898

    accuracy                           0.47     46631
   macro avg       0.55      0.57      0.46     46631
weighted avg       0.72      0.47      0.51     46631
```

In [73]:
```
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['Predic
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



In [75]:
```
data.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone\Ba
data_v2.to_csv(r'C:\Users\vinay\Desktop\Siplilearn\Data Analyst Masters Capstone
```

In [ ]: