LoanTap is an online platform committed to delivering customized loan products to millennials. They innovate in an otherwise dull loan segment, to deliver instant, flexible loans on consumer friendly terms to salaried professionals and businessmen.

The data science team at LoanTap is building an underwriting layer to determine the creditworthiness of MSMEs as well as individuals.

LoanTap deploys formal credit to salaried individuals and businesses 4 main financial instruments:

Personal Loan EMI Free Loan Personal Overdraft Advance Salary Loan This case study will focus on the underwriting process behind Personal Loan only

Understanding the data

loan_amnt : The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value.

term : The number of payments on the loan. Values are in months and can be either 36 or 60.

int_rate : Interest Rate on the loan

installment : The monthly payment owed by the borrower if the loan originates.

grade : LoanTap assigned loan grade

sub_grade : LoanTap assigned loan subgrade

emp_title :The job title supplied by the Borrower when applying for the loan.*

emp_length : Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.

home_ownership : The home ownership status provided by the borrower during registration or obtained from the credit report.

annual_inc : The self-reported annual income provided by the borrower during registration.

verification_status : Indicates if income was verified by LoanTap, not verified, or if the income source was verified

issue_d : The month which the loan was funded

loan_status : Current status of the loan - Target Variable

purpose : A category provided by the borrower for the loan request.

title : The loan title provided by the borrower

dti : A ratio calculated using the borrower’s total monthly debt payments on the total debt obligations, excluding mortgage and the requested LoanTap loan, divided by the borrower’s self-reported monthly income.

earliest_cr_line :The month the borrower's earliest reported credit line was opened

open_acc : The number of open credit lines in the borrower's credit file.

pub_rec : Number of derogatory public records

revol_bal : Total credit revolving balance

revol_util : Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.

total_acc : The total number of credit lines currently in the borrower's credit file

initial_list_status : The initial listing status of the loan. Possible values are – W, F

application_type : Indicates whether the loan is an individual application or a joint application with two co-borrowers

mort_acc : Number of mortgage accounts.

pub_rec_bankruptcies : Number of public record bankruptcies

Address: Address of the individual

```
import pandas as pd
import numpy as np
import seaborn as sns
from scipy import stats
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import classification_report,
precision_recall_curve, confusion_matrix
from sklearn.metrics import accuracy_score, roc_auc_score, roc_curve,
auc, ConfusionMatrixDisplay, RocCurveDisplay
from sklearn.model_selection import train_test_split, KFold,
cross_val_score
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from statsmodels.stats.outliers_influence import
variance_inflation_factor
import statsmodels.api as sm
from imblearn.over_sampling import SMOTE
import warnings

df = pd.read_csv("logistic_regression.csv")
df.head()
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	\
0	10000.0	36 months	11.44	329.48	B	B4	
1	8000.0	36 months	11.99	265.68	B	B5	
2	15600.0	36 months	10.49	506.97	B	B3	
3	7200.0	36 months	6.49	220.65	A	A2	
4	24375.0	60 months	17.27	609.33	C	C5	

```
                 emp_title emp_length home_ownership  annual_inc  ...
\
0                Marketing  10+ years           RENT    117000.0  ...

1          Credit analyst     4 years       MORTGAGE     65000.0  ...

2             Statistician   < 1 year           RENT     43057.0  ...

3          Client Advocate    6 years           RENT     54000.0  ...

4  Destiny Management Inc.    9 years       MORTGAGE     55000.0  ...


  open_acc pub_rec revol_bal revol_util total_acc  initial_list_status
\
0     16.0     0.0   36369.0       41.8      25.0                    w

1     17.0     0.0   20131.0       53.3      27.0                    f

2     13.0     0.0   11987.0       92.2      26.0                    f

3      6.0     0.0    5472.0       21.5      13.0                    f

4     13.0     0.0   24584.0       69.8      43.0                    f


  application_type  mort_acc  pub_rec_bankruptcies  \
0       INDIVIDUAL       0.0                   0.0
1       INDIVIDUAL       3.0                   0.0
2       INDIVIDUAL       0.0                   0.0
3       INDIVIDUAL       0.0                   0.0
4       INDIVIDUAL       1.0                   0.0

                                             address
0     0174 Michelle Gateway\r\nMendozaberg, OK 22690
1  1076 Carney Fort Apt. 347\r\nLoganmouth, SD 05113
2  87025 Mark Dale Apt. 269\r\nNew Sabrina, WV 05113
3            823 Reid Ford\r\nDelacruzside, MA 00813
4             679 Luna Roads\r\nGreggshire, VA 11650

[5 rows x 27 columns]

df.describe()

           loan_amnt       int_rate    installment    annual_inc  \
count  396030.000000  396030.000000  396030.000000  3.960300e+05
mean    14113.888089      13.639400     431.849698  7.420318e+04
std      8357.441341       4.472157     250.727790  6.163762e+04
min       500.000000       5.320000      16.080000  0.000000e+00
25%      8000.000000      10.490000     250.330000  4.500000e+04
50%     12000.000000      13.330000     375.430000  6.400000e+04
```

```
75%     20000.000000      16.490000     567.300000  9.000000e+04
max     40000.000000      30.990000    1533.810000  8.706582e+06

                dti       open_acc        pub_rec     revol_bal  \
count  396030.000000  396030.000000  396030.000000  3.960300e+05
mean       17.379514      11.311153       0.178191  1.584454e+04
std        18.019092       5.137649       0.530671  2.059184e+04
min         0.000000       0.000000       0.000000  0.000000e+00
25%        11.280000       8.000000       0.000000  6.025000e+03
50%        16.910000      10.000000       0.000000  1.118100e+04
75%        22.980000      14.000000       0.000000  1.962000e+04
max      9999.000000      90.000000      86.000000  1.743266e+06

          revol_util      total_acc       mort_acc
pub_rec_bankruptcies
count  395754.000000  396030.000000  358235.000000
395495.000000
mean       53.791749      25.414744       1.813991
0.121648
std        24.452193      11.886991       2.147930
0.356174
min         0.000000       2.000000       0.000000
0.000000
25%        35.800000      17.000000       0.000000
0.000000
50%        54.800000      24.000000       1.000000
0.000000
75%        72.900000      32.000000       3.000000
0.000000
max       892.300000     151.000000      34.000000
8.000000

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 396030 entries, 0 to 396029
Data columns (total 27 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   loan_amnt             396030 non-null  float64
 1   term                  396030 non-null  object
 2   int_rate              396030 non-null  float64
 3   installment           396030 non-null  float64
 4   grade                 396030 non-null  object
 5   sub_grade             396030 non-null  object
 6   emp_title             373103 non-null  object
 7   emp_length            377729 non-null  object
 8   home_ownership        396030 non-null  object
 9   annual_inc            396030 non-null  float64
 10  verification_status   396030 non-null  object
```

```
 11  issue_d               396030 non-null  object
 12  loan_status           396030 non-null  object
 13  purpose               396030 non-null  object
 14  title                 394274 non-null  object
 15  dti                   396030 non-null  float64
 16  earliest_cr_line      396030 non-null  object
 17  open_acc              396030 non-null  float64
 18  pub_rec               396030 non-null  float64
 19  revol_bal             396030 non-null  float64
 20  revol_util            395754 non-null  float64
 21  total_acc             396030 non-null  float64
 22  initial_list_status   396030 non-null  object
 23  application_type      396030 non-null  object
 24  mort_acc              358235 non-null  float64
 25  pub_rec_bankruptcies  395495 non-null  float64
 26  address               396030 non-null  object
dtypes: float64(12), object(15)
memory usage: 81.6+ MB
```

```
for i in df.columns:
    print(i, '-->> ', df[i].unique(), '\n')
```

```
loan_amnt -->>  [10000.  8000. 15600. ... 36275. 36475.   725.]

term -->>  [' 36 months' ' 60 months']

int_rate -->>  [11.44 11.99 10.49  6.49 17.27 13.33  5.32 11.14 10.99
16.29 13.11 14.64
  9.17 12.29  6.62  8.39 21.98  7.9   6.97  6.99 15.61 11.36 13.35
12.12
  9.99  8.19 18.75  6.03 14.99 16.78 13.67 13.98 16.99 19.91 17.86
21.49
 12.99 18.54  7.89 17.1  18.25 11.67  6.24  8.18 12.35 14.16 17.56
18.55
 22.15 10.39 15.99 16.07 24.99  9.67 19.19 21.   12.69 10.74  6.68
19.22
 11.49 16.55 19.97 24.7  13.49 18.24 16.49 25.78 25.83 18.64  7.51
13.99
 15.22 15.31  7.69 19.53 10.16  7.62  9.75 13.68 15.88 14.65  6.92
23.83
 10.75 18.49 20.31 17.57 27.31 19.99 22.99 12.59 10.37 14.33 13.53
22.45
 24.5  17.99  9.16 12.49 11.55 17.76 28.99 23.1  20.49 22.7  10.15
6.89
 19.52  8.9  14.3   9.49 25.99 24.08 13.05 14.98 16.59 11.26 25.89
14.48
 21.99 23.99  5.99 14.47 11.53  8.67  8.59 10.64 23.28 25.44  9.71
16.2
 19.24 24.11 15.8  15.96 14.49 18.99  5.79 19.29 14.54 14.09  9.25
19.05
```

```
 17.77 18.92 20.75 10.65 18.85 10.59 12.85 11.39 13.65 13.06  7.12
20.99
 13.61 12.73 14.46 16.24 25.49  7.39 10.78 20.8   7.88 15.95 12.39
21.18
 21.97 15.77  6.39 10.   12.53 13.43  7.49 25.57 21.48 18.39 11.47
7.26
 15.68 19.04 14.31 24.24  5.42 23.43 19.47  6.54 23.32 17.58 14.72
7.66
  9.76 13.23 13.48 12.42  9.8  11.71 14.27 21.15 22.95  8.49 17.74
15.59
 13.72  9.45  7.29 15.1  11.86 19.72 14.35 11.22 15.62 15.81 12.41
28.67
 11.48 13.66  9.91 23.76 17.14 18.84 12.23  6.17  8.94 14.22 19.03
25.29
  8.99  9.88 15.58 27.49  8.07 22.47 19.2  13.44 22.4  12.79 18.2
13.18
  7.24 14.84  5.93 15.28 13.85 25.28  8.    9.62 12.05 15.7  20.2
13.57
 21.67  7.4  25.8  12.68 11.83  7.37 11.11 14.85 16.   11.12 23.63  6.
  7.99  7.91 14.83 21.7  26.06 16.77 27.34 12.21  7.68 15.27 19.69
9.63
  7.14 20.5  16.02 12.84  7.74 15.33 19.79 22.2  18.62 17.49 16.89
15.21
 14.79 18.67  9.32 15.41 15.65 23.5  22.9  11.34 22.11 19.48 14.75
28.14
 13.22 23.4  23.13 28.18 12.88 22.06 24.49 16.45 21.6  28.49  8.38
6.76
 10.83 13.79  8.88 17.88 17.97 14.26  6.91 13.47  8.6  27.88  8.63
10.25
 14.91 12.74 10.96 25.88  7.43 16.4  20.25 24.89 12.87 20.16 14.17
12.18
 17.51 13.92 20.53 26.77 10.62 26.49 16.32 12.61 21.36 14.61 15.37
20.3
 14.59 16.7  19.89 10.95 18.17 18.21 17.93 22.39 24.83 13.8  19.42
23.7
  7.59 13.17 18.09 13.04 25.69  9.07 15.23 14.42 23.33 16.69 10.36
14.96
 10.38 26.24 24.2  12.98 20.85 13.36 26.57 23.52 22.78 13.16 15.13
25.11
 13.55 10.51 11.78  7.05 11.46 21.28 12.09 16.35  8.7  26.99 14.11
26.14
 16.82 23.26 18.79 10.28 19.36 18.3  17.06 17.19  7.75 17.34 20.89
22.35
 19.66 13.62 22.74 11.89 23.59  8.24 20.62 11.97 15.2  20.48 12.36
10.71
 25.09 20.11 27.79 29.49 11.58 19.13 11.66 13.75 30.74  9.38 27.99
11.59
  9.64 25.65  9.96 19.41 14.18 10.08 17.43 24.74 14.74 17.04 15.57
30.49
```

```
 17.8  10.91 14.82 29.96 12.92 12.22 15.45 11.72 10.2  14.7  20.69
15.05
 24.33 14.93 10.33 16.95 28.88 11.03 28.34 21.22 18.07  9.33 12.17
19.74
 20.9  20.03 17.39 29.67 12.04 23.22 10.01 22.48 24.76 13.3  20.77
10.14
 14.5  30.94  8.32 13.24 21.59 21.27 24.52 11.54 10.46 13.87 30.99
9.51
  9.83 19.39 12.86 30.79 21.74 11.09 16.11 17.26 22.85 18.91 18.43
9.2
 21.14 12.62 21.21 29.99 14.88 13.12 30.89 16.08 12.54 28.69 12.8
11.28
 23.91 22.94 19.16 20.86 11.63 19.82 11.41 21.82 12.72 20.4   9.7
18.72
 18.36 14.25 13.84 18.78 17.15 15.25 16.63 16.15 11.91 14.07  9.01
15.01
 21.64 15.83 18.53  7.42 12.67 15.76 16.33 30.84 13.93 14.12 14.28
20.17
 24.59 20.52 17.03 17.9  14.67 15.38 17.46 14.62 14.38 24.4  22.64
17.54
 17.44 15.07]

installment -->>  [329.48 265.68 506.97 ... 343.14 118.13 572.44]

grade -->>  ['B' 'A' 'C' 'E' 'D' 'F' 'G']

sub_grade -->>  ['B4' 'B5' 'B3' 'A2' 'C5' 'C3' 'A1' 'B2' 'C1' 'A5'
'E4' 'A4' 'A3' 'D1'
 'C2' 'B1' 'D3' 'D5' 'D2' 'E1' 'E2' 'E5' 'F4' 'E3' 'D4' 'G1' 'F5' 'G2'
 'C4' 'F1' 'F3' 'G5' 'G4' 'F2' 'G3']

emp_title -->>  ['Marketing' 'Credit analyst ' 'Statistician' ...
 "Michael's Arts & Crafts" 'licensed bankere' 'Gracon Services, Inc']

emp_length -->>  ['10+ years' '4 years' '< 1 year' '6 years' '9 years'
'2 years' '3 years'
 '8 years' '7 years' '5 years' '1 year' nan]

home_ownership -->>  ['RENT' 'MORTGAGE' 'OWN' 'OTHER' 'NONE' 'ANY']

annual_inc -->>  [117000.    65000.    43057.   ...  36111.    47212.
31789.88]

verification_status -->>  ['Not Verified' 'Source Verified'
'Verified']

issue_d -->>  ['Jan-2015' 'Nov-2014' 'Apr-2013' 'Sep-2015' 'Sep-2012'
'Oct-2014'
 'Apr-2012' 'Jun-2013' 'May-2014' 'Dec-2015' 'Apr-2015' 'Oct-2012'
 'Jul-2014' 'Feb-2013' 'Oct-2015' 'Jan-2014' 'Mar-2016' 'Apr-2014'
```

```
 'Jun-2011' 'Apr-2010' 'Jun-2014' 'Oct-2013' 'May-2013' 'Feb-2015'
 'Oct-2011' 'Jun-2015' 'Aug-2013' 'Feb-2014' 'Dec-2011' 'Mar-2013'
 'Jun-2016' 'Mar-2014' 'Nov-2013' 'Dec-2014' 'Apr-2016' 'Sep-2013'
 'May-2016' 'Jul-2015' 'Jul-2013' 'Aug-2014' 'May-2008' 'Mar-2010'
 'Dec-2013' 'Mar-2012' 'Mar-2015' 'Sep-2011' 'Jul-2012' 'Dec-2012'
 'Sep-2014' 'Nov-2012' 'Nov-2015' 'Jan-2011' 'May-2012' 'Feb-2016'
 'Jun-2012' 'Aug-2012' 'Jan-2016' 'May-2015' 'Oct-2016' 'Aug-2015'
 'Jul-2016' 'May-2009' 'Aug-2016' 'Jan-2012' 'Jan-2013' 'Nov-2010'
 'Jul-2011' 'Mar-2011' 'Feb-2012' 'May-2011' 'Aug-2010' 'Nov-2016'
 'Jul-2010' 'Sep-2010' 'Dec-2010' 'Feb-2011' 'Jun-2009' 'Aug-2011'
 'Dec-2016' 'Mar-2009' 'Jun-2010' 'May-2010' 'Nov-2011' 'Sep-2016'
 'Oct-2009' 'Mar-2008' 'Nov-2008' 'Dec-2009' 'Oct-2010' 'Sep-2009'
 'Oct-2007' 'Aug-2009' 'Jul-2009' 'Nov-2009' 'Jan-2010' 'Dec-2008'
 'Feb-2009' 'Oct-2008' 'Apr-2009' 'Feb-2010' 'Apr-2011' 'Apr-2008'
 'Aug-2008' 'Jan-2009' 'Feb-2008' 'Aug-2007' 'Sep-2008' 'Dec-2007'
 'Jan-2008' 'Sep-2007' 'Jun-2008' 'Jul-2008' 'Jun-2007' 'Nov-2007'
 'Jul-2007']

loan_status -->>  ['Fully Paid' 'Charged Off']

purpose -->>  ['vacation' 'debt_consolidation' 'credit_card'
'home_improvement'
 'small_business' 'major_purchase' 'other' 'medical' 'wedding' 'car'
 'moving' 'house' 'educational' 'renewable_energy']

title -->>  ['Vacation' 'Debt consolidation' 'Credit card refinancing'
...
 'Credit buster ' 'Loanforpayoff' 'Toxic Debt Payoff']

dti -->>  [26.24 22.05 12.79 ... 40.56 47.09 55.53]

earliest_cr_line -->>  ['Jun-1990' 'Jul-2004' 'Aug-2007' 'Sep-2006'
'Mar-1999' 'Jan-2005'
 'Aug-2005' 'Sep-1994' 'Jun-1994' 'Dec-1997' 'Dec-1990' 'May-1984'
 'Apr-1995' 'Jan-1997' 'May-2001' 'Mar-1982' 'Sep-1996' 'Jan-1990'
 'Mar-2000' 'Jan-2006' 'Oct-2006' 'Jan-2003' 'May-2008' 'Oct-2003'
 'Jun-2004' 'Jan-1999' 'Apr-1994' 'Apr-1998' 'Jul-2007' 'Apr-2002'
 'Oct-2007' 'Jun-2009' 'May-1997' 'Jul-2006' 'Sep-2003' 'Aug-1992'
 'Dec-1988' 'Feb-2002' 'Jan-1992' 'Aug-2001' 'Dec-2010' 'Oct-1999'
 'Sep-2004' 'Aug-1994' 'Jul-2003' 'Apr-2000' 'Dec-2004' 'Jun-1995'
 'Dec-2003' 'Jul-1994' 'Oct-1990' 'Dec-2001' 'Apr-1999' 'Feb-1995'
 'May-2003' 'Oct-2002' 'Mar-2004' 'Aug-2003' 'Oct-2000' 'Nov-2004'
 'Mar-2010' 'Mar-1996' 'May-1994' 'Jun-1996' 'Nov-1986' 'Jan-2001'
 'Jan-2002' 'Mar-2001' 'Sep-2012' 'Apr-2006' 'May-1998' 'Dec-2002'
 'Nov-2003' 'Oct-2005' 'May-1990' 'Jun-2003' 'Jun-2001' 'Jan-1998'
 'Oct-1978' 'Feb-2001' 'Jun-2006' 'Aug-1993' 'Apr-2001' 'Nov-2001'
 'Feb-2003' 'Jun-1993' 'Sep-1992' 'Nov-1992' 'Jun-1983' 'Oct-2001'
 'Jul-1999' 'Sep-1997' 'Nov-1993' 'Feb-1993' 'Apr-2007' 'Nov-1999'
 'Nov-2005' 'Dec-1992' 'Mar-1986' 'May-1989' 'Dec-2000' 'Mar-1991'
 'Mar-2005' 'Jun-2010' 'Dec-1998' 'Sep-2001' 'Nov-2000' 'Jan-1994'
```

```
'Aug-2002' 'Jan-2011' 'Aug-2008' 'Jun-2005' 'Nov-1997' 'May-1996'
'Apr-2010' 'May-1993' 'Sep-2005' 'Jun-1992' 'Apr-1986' 'Aug-1996'
'Aug-1997' 'Jul-2005' 'May-2011' 'Sep-2002' 'Jan-1989' 'Aug-1999'
'Feb-1992' 'Sep-1999' 'Jul-2001' 'May-1980' 'Oct-2008' 'Nov-2007'
'Apr-1997' 'Jun-1986' 'Sep-1998' 'Jun-1982' 'Oct-1981' 'Feb-1994'
'Dec-1984' 'Nov-1991' 'Nov-2006' 'Aug-2000' 'Oct-2004' 'Jun-2011'
'Apr-1988' 'May-2004' 'Aug-1988' 'Mar-1994' 'Aug-2004' 'Dec-2006'
'Nov-1998' 'Oct-1997' 'Mar-1989' 'Feb-1988' 'Jul-1982' 'Nov-1995'
'Mar-1997' 'Oct-1994' 'Jul-1998' 'Jun-2002' 'May-1991' 'Oct-2011'
'Sep-2007' 'Jan-2007' 'Jan-2010' 'Mar-1987' 'Feb-1997' 'Oct-1986'
'Mar-2002' 'Jul-1993' 'Mar-2007' 'Aug-1989' 'Oct-1995' 'May-2007'
'Dec-1993' 'Jun-1989' 'Apr-2004' 'Jun-1997' 'Apr-1996' 'Apr-1992'
'Oct-1998' 'Mar-1983' 'Mar-1985' 'Oct-1993' 'Feb-2000' 'Apr-2003'
'Oct-1985' 'Jul-1985' 'May-1978' 'Sep-2010' 'Oct-1996' 'Sep-2009'
'Jun-1999' 'Jan-2000' 'Sep-1987' 'Aug-1998' 'Jan-1995' 'Jul-1988'
'May-2000' 'Jun-1981' 'Feb-1998' 'Nov-1996' 'Aug-1967' 'Dec-1999'
'Aug-2006' 'Nov-2009' 'Jul-2000' 'Mar-1988' 'Jul-1992' 'Jul-1991'
'Mar-1990' 'May-1986' 'Jun-1991' 'Dec-1987' 'Jul-1996' 'Jul-1997'
'Aug-1990' 'Jan-1988' 'Dec-2005' 'Mar-2003' 'Feb-1999' 'Nov-1990'
'Jun-2000' 'Dec-1996' 'Jan-2004' 'May-1999' 'Sep-1972' 'Jul-1981'
'Sep-1993' 'Feb-2009' 'Nov-2002' 'Nov-1969' 'Jan-1993' 'May-2005'
'Sep-1982' 'Apr-1990' 'Feb-1996' 'Mar-1993' 'Apr-1978' 'Jul-1995'
'May-1995' 'Apr-1991' 'Mar-1998' 'Aug-1991' 'Jul-2002' 'Oct-1989'
'Apr-1984' 'Dec-2009' 'Sep-2000' 'Jan-1982' 'Jun-1998' 'Jan-1996'
'Nov-1987' 'May-2010' 'Jul-1989' 'Jun-1987' 'Oct-1987' 'Aug-1995'
'Feb-2004' 'Oct-1991' 'Dec-1989' 'Oct-1992' 'Feb-2005' 'Apr-1993'
'Dec-1985' 'Sep-1979' 'Feb-2007' 'Nov-1989' 'Apr-2005' 'Mar-1978'
'Sep-1985' 'Nov-1994' 'Jun-2008' 'Apr-1987' 'Dec-1983' 'Dec-2007'
'May-1979' 'May-1992' 'Jul-1990' 'Mar-1995' 'Feb-2006' 'Feb-1985'
'Sep-1989' 'Aug-2009' 'Nov-2008' 'Nov-1981' 'Jan-2008' 'Aug-1987'
'Nov-1985' 'Dec-1965' 'Sep-1995' 'Jan-1986' 'Oct-2009' 'May-2002'
'Aug-1980' 'Sep-1977' 'Sep-1988' 'Oct-1984' 'May-1988' 'Aug-1984'
'Nov-1988' 'May-1974' 'Nov-1982' 'Oct-1983' 'Sep-1991' 'Feb-1984'
'Feb-1991' 'Jan-1981' 'Jun-1985' 'Dec-1976' 'Dec-1994' 'Dec-1980'
'Sep-1984' 'Jun-2007' 'Aug-1979' 'Sep-2008' 'Apr-1983' 'Mar-2006'
'Jun-1984' 'Jul-1984' 'Jan-1985' 'Dec-1995' 'Apr-2008' 'Mar-2008'
'Jan-1983' 'Dec-1986' 'Jun-1979' 'Dec-1975' 'Nov-1983' 'Jul-1986'
'Nov-1977' 'Dec-1982' 'May-1985' 'Feb-1983' 'Aug-1982' 'Oct-1980'
'Mar-1979' 'Jan-1978' 'Mar-1984' 'May-1983' 'Jul-2008' 'Apr-1982'
'Jul-1983' 'Feb-1990' 'Dec-2008' 'Jul-1975' 'Dec-1971' 'Feb-2008'
'Mar-2011' 'Feb-1987' 'Feb-1989' 'Aug-1985' 'Jul-2010' 'Apr-1989'
'Feb-1980' 'May-2006' 'Nov-2010' 'Apr-2009' 'Feb-2010' 'May-1976'
'Feb-1981' 'Jan-2012' 'Oct-1988' 'Nov-1984' 'May-1982' 'Oct-1975'
'Jun-1988' 'May-1972' 'Apr-2013' 'Sep-1990' 'Oct-1982' 'Feb-2013'
'Mar-1992' 'Aug-1981' 'Feb-2011' 'Nov-1974' 'Feb-1978' 'Sep-1983'
'Jul-2011' 'Nov-1979' 'Aug-1983' 'Apr-1985' 'Jul-2009' 'Jan-1971'
'Jul-1987' 'Aug-1978' 'Aug-2010' 'Oct-1976' 'Aug-1986' 'Jan-1991'
'Dec-1991' 'May-2009' 'Aug-2011' 'Jun-1964' 'Jan-1974' 'May-1981'
'Jun-1972' 'Jun-1978' 'Sep-1986' 'Jan-1987' 'Jan-1975' 'Feb-1982'
```

```
 'Jan-1980' 'Feb-1977' 'Sep-1980' 'Nov-1978' 'Jul-1974' 'Jun-1970'
 'Jan-1984' 'Nov-1980' 'May-1987' 'Sep-1970' 'Jan-1976' 'Feb-1986'
 'Oct-2010' 'Apr-1979' 'Oct-1979' 'Jan-1979' 'Sep-2011' 'Jul-1979'
 'Sep-1975' 'Mar-1981' 'Aug-1971' 'Apr-1980' 'Apr-1977' 'Jan-1965'
 'Nov-1976' 'Nov-1970' 'Nov-2011' 'Nov-1973' 'Sep-1981' 'Jul-1980'
 'Mar-2012' 'Dec-1974' 'Mar-1977' 'Dec-1977' 'May-2012' 'Dec-1979'
 'Jan-2009' 'Jan-1970' 'Dec-2011' 'Feb-1979' 'Mar-1976' 'Jan-1973'
 'Oct-1973' 'Mar-1969' 'Oct-1977' 'Mar-1975' 'Aug-1977' 'Jun-1969'
 'Oct-1963' 'Nov-1960' 'Aug-1970' 'Feb-1975' 'Sep-1974' 'May-1966'
 'Apr-1972' 'Apr-1973' 'Apr-2012' 'May-1975' 'Sep-1966' 'Feb-1969'
 'Feb-2012' 'Jan-1961' 'Aug-1973' 'Feb-1972' 'Apr-1975' 'Jul-1978'
 'Oct-1970' 'Mar-1980' 'Sep-1976' 'Apr-2011' 'Nov-2012' 'Aug-1976'
 'Jun-1975' 'Apr-1981' 'Mar-2009' 'Jun-1977' 'Apr-1971' 'Sep-1969'
 'Jun-2012' 'Apr-1976' 'Feb-1965' 'Jul-1977' 'Jun-1976' 'Mar-1973'
 'Oct-1972' 'Dec-1978' 'Nov-1967' 'Sep-1967' 'Nov-1971' 'Jun-1980'
 'May-1964' 'Feb-1971' 'May-1970' 'Apr-1970' 'Mar-1971' 'Apr-1969'
 'Jan-1963' 'Jun-1974' 'Oct-1974' 'May-1977' 'Dec-1981' 'Jan-1969'
 'Feb-1976' 'Mar-1970' 'Aug-1968' 'Feb-1970' 'Jun-1971' 'Jun-1963'
 'Jun-2013' 'Mar-1972' 'Aug-2012' 'Jan-1967' 'Feb-1968' 'Dec-1969'
 'Jan-1977' 'Jul-1970' 'Feb-1973' 'Mar-1974' 'Feb-1974' 'Dec-1960'
 'Jul-1972' 'Jul-1973' 'Sep-1964' 'Jul-1965' 'Oct-1958' 'Jul-2012'
 'Jun-1973' 'Sep-1978' 'Nov-1975' 'Jul-1963' 'Jan-1964' 'Dec-1968'
 'May-1958' 'Sep-1973' 'May-1971' 'Dec-1972' 'Aug-1965' 'Jul-1976'
 'Oct-2012' 'May-1973' 'Apr-1955' 'Apr-1966' 'Jan-1968' 'Nov-1968'
 'Oct-1969' 'Mar-2013' 'Jan-2013' 'Jul-1967' 'Oct-1965' 'Jan-1966'
 'Aug-1972' 'Jul-1969' 'May-1965' 'Jan-1953' 'Aug-1974' 'May-1968'
 'Aug-1969' 'May-2013' 'Oct-1967' 'Aug-1975' 'Apr-1974' 'Sep-1971'
 'Apr-1968' 'Jul-1971' 'Jan-1972' 'Nov-1965' 'Dec-1970' 'Dec-1973'
 'Nov-1972' 'Oct-1959' 'Oct-1962' 'Apr-1967' 'Oct-1971' 'Nov-1963'
 'Oct-1968' 'Dec-1962' 'Jun-1960' 'Jan-1960' 'Sep-2013' 'May-1969'
 'Dec-1966' 'Feb-1967' 'Dec-1967' 'Aug-1961' 'Sep-1968' 'Oct-1964'
 'Aug-1966' 'Jul-1966' 'Apr-1964' 'Sep-1962' 'Jul-2013' 'Jun-1967'
 'Apr-1965' 'Jun-1966' 'Jan-1955' 'Jan-1962' 'Feb-1964' 'Aug-1958'
 'Jul-1968' 'May-1967' 'Dec-1959' 'Sep-1963' 'Dec-2012' 'Dec-1963'
 'Jan-1944' 'Jun-1965' 'May-1962' 'Mar-1967' 'Mar-1968' 'Jan-1956'
 'Sep-1965' 'Dec-1951' 'Aug-2013' 'Jun-1968' 'Mar-1965' 'Oct-1957'
 'Nov-1966' 'Dec-1958' 'Feb-1957' 'Feb-1963' 'Mar-1963' 'Jan-1959'
 'May-1955' 'Feb-1966' 'Nov-1950' 'Mar-1964' 'Jan-1958' 'Nov-1964'
 'Sep-1961' 'Apr-1963' 'Jul-1964' 'Nov-1955' 'Jun-1957' 'Dec-1964'
 'Nov-1953' 'Apr-1961' 'Mar-1966' 'Oct-1960' 'Jul-1959' 'Jul-1961'
 'Jan-1954' 'Dec-1956' 'Mar-1962' 'Jul-1960' 'Sep-1959' 'Dec-1950'
 'Oct-1966' 'Apr-1960' 'Jul-1958' 'Nov-1954' 'Nov-1957' 'Jun-1962'
 'May-1963' 'Jul-1955' 'Oct-1950' 'Dec-1961' 'Aug-1951' 'Oct-2013'
 'Aug-1964' 'Apr-1962' 'Jun-1955' 'Jul-1962' 'Jan-1957' 'Nov-1958'
 'Jul-1951' 'Nov-1959' 'Apr-1958' 'Mar-1960' 'Sep-1957' 'Nov-1961'
 'Sep-1960' 'May-1959' 'Jun-1959' 'Feb-1962' 'Sep-1956' 'Aug-1960'
 'Feb-1961' 'Jan-1948' 'Aug-1963' 'Oct-1961' 'Aug-1962' 'Aug-1959']

open_acc -->>  [16. 17. 13.  6.  8. 11.  5. 30.  9. 15. 12. 10. 18.
```

```
7.  4. 14. 20. 19.
 21. 23.  3. 26. 42. 22. 25. 28.  2. 34. 24. 27. 31. 32. 33.  1. 29.
36.
 40. 35. 37. 41. 44. 39. 49. 48. 38. 51. 50. 43. 46.  0. 47. 57. 53.
58.
 52. 54. 45. 90. 56. 55. 76.]

pub_rec -->>  [ 0.  1.  2.  3.  4.  6.  5.  8.  9. 10. 11.  7. 19. 13.
40. 17. 86. 12.
 24. 15.]

revol_bal -->>  [ 36369.  20131.  11987. ...  34531. 151912.  29244.]

revol_util -->>  [ 41.8   53.3   92.2  ...  56.26 111.4  128.1 ]

total_acc -->>  [ 25.  27.  26.  13.  43.  23.  15.  40.  37.  61.
35.  22.  20.  36.
  38.   7.  18.  10.  17.  29.  16.  21.  34.   9.  14.  59.  41.  19.
  12.  30.  56.  24.  28.   8.  52.  31.  44.  39.  50.  11.  62.  32.
   5.  33.  46.  42.   6.  49.  45.  57.  48.  67.  47.  51.  58.   3.
  55.  63.  53.   4.  71.  69.  54.  64.  81.  72.  60.  68.  65.  73.
  78.  84.   2.  76.  75.  79.  87.  77. 104.  89.  70. 105.  97.  66.
 108.  74.  80.  82.  91.  93. 106.  90.  85.  88.  83. 111.  86. 101.
 135.  92.  94.  95.  99. 102. 129. 110. 124. 151. 107. 118. 150. 115.
 117.  96.  98. 100. 116. 103.]

initial_list_status -->>  ['w' 'f']

application_type -->>  ['INDIVIDUAL' 'JOINT' 'DIRECT_PAY']

mort_acc -->>  [ 0.  3.  1.  4.  2.  6.  5. nan 10.  7. 12. 11.  8.
9. 13. 14. 22. 34.
 15. 25. 19. 16. 17. 32. 18. 24. 21. 20. 31. 28. 30. 23. 26. 27.]

pub_rec_bankruptcies -->>  [ 0.  1.  2.  3. nan  4.  5.  6.  7.  8.]

address -->>  ['0174 Michelle Gateway\r\nMendozaberg, OK 22690'
 '1076 Carney Fort Apt. 347\r\nLoganmouth, SD 05113'
 '87025 Mark Dale Apt. 269\r\nNew Sabrina, WV 05113' ...
 '953 Matthew Points Suite 414\r\nReedfort, NY 70466'
 '7843 Blake Freeway Apt. 229\r\nNew Michael, FL 29597'
 '787 Michelle Causeway\r\nBriannaton, AR 48052']


df.isna().sum()

loan_amnt                   0
term                        0
int_rate                    0
installment                 0
grade                       0
```

```
sub_grade                   0
emp_title               22927
emp_length              18301
home_ownership              0
annual_inc                  0
verification_status         0
issue_d                     0
loan_status                 0
purpose                     0
title                    1756
dti                         0
earliest_cr_line            0
open_acc                    0
pub_rec                     0
revol_bal                   0
revol_util                276
total_acc                   0
initial_list_status         0
application_type            0
mort_acc                37795
pub_rec_bankruptcies      535
address                     0
dtype: int64
```

```
df.dropna(inplace=True)
```

Dropping all null values

```
df.isna().sum()
```

```
loan_amnt               0
term                    0
int_rate                0
installment             0
grade                   0
sub_grade               0
emp_title               0
emp_length              0
home_ownership          0
annual_inc              0
verification_status     0
issue_d                 0
loan_status             0
purpose                 0
title                   0
dti                     0
earliest_cr_line        0
open_acc                0
pub_rec                 0
```

```
revol_bal               0
revol_util              0
total_acc               0
initial_list_status     0
application_type        0
mort_acc                0
pub_rec_bankruptcies    0
address                 0
dtype: int64
```

```
df.shape
```

```
(335867, 27)
```

```
df.duplicated().sum()
```

```
0
```

There are no duplicates in the dataset

Univariate Analysis

```
df.columns
```

```
Index(['loan_amnt', 'term', 'int_rate', 'installment', 'grade',
'sub_grade',
       'emp_title', 'emp_length', 'home_ownership', 'annual_inc',
       'verification_status', 'issue_d', 'loan_status', 'purpose',
'title',
       'dti', 'earliest_cr_line', 'open_acc', 'pub_rec', 'revol_bal',
       'revol_util', 'total_acc', 'initial_list_status',
'application_type',
       'mort_acc', 'pub_rec_bankruptcies', 'address'],
      dtype='object')
```

```
float_columns = df.select_dtypes(include=['float64']).columns
float_columns
```

```
Index(['loan_amnt', 'int_rate', 'installment', 'annual_inc', 'dti',
'open_acc',
       'pub_rec', 'revol_bal', 'revol_util', 'total_acc', 'mort_acc',
       'pub_rec_bankruptcies'],
      dtype='object')
```

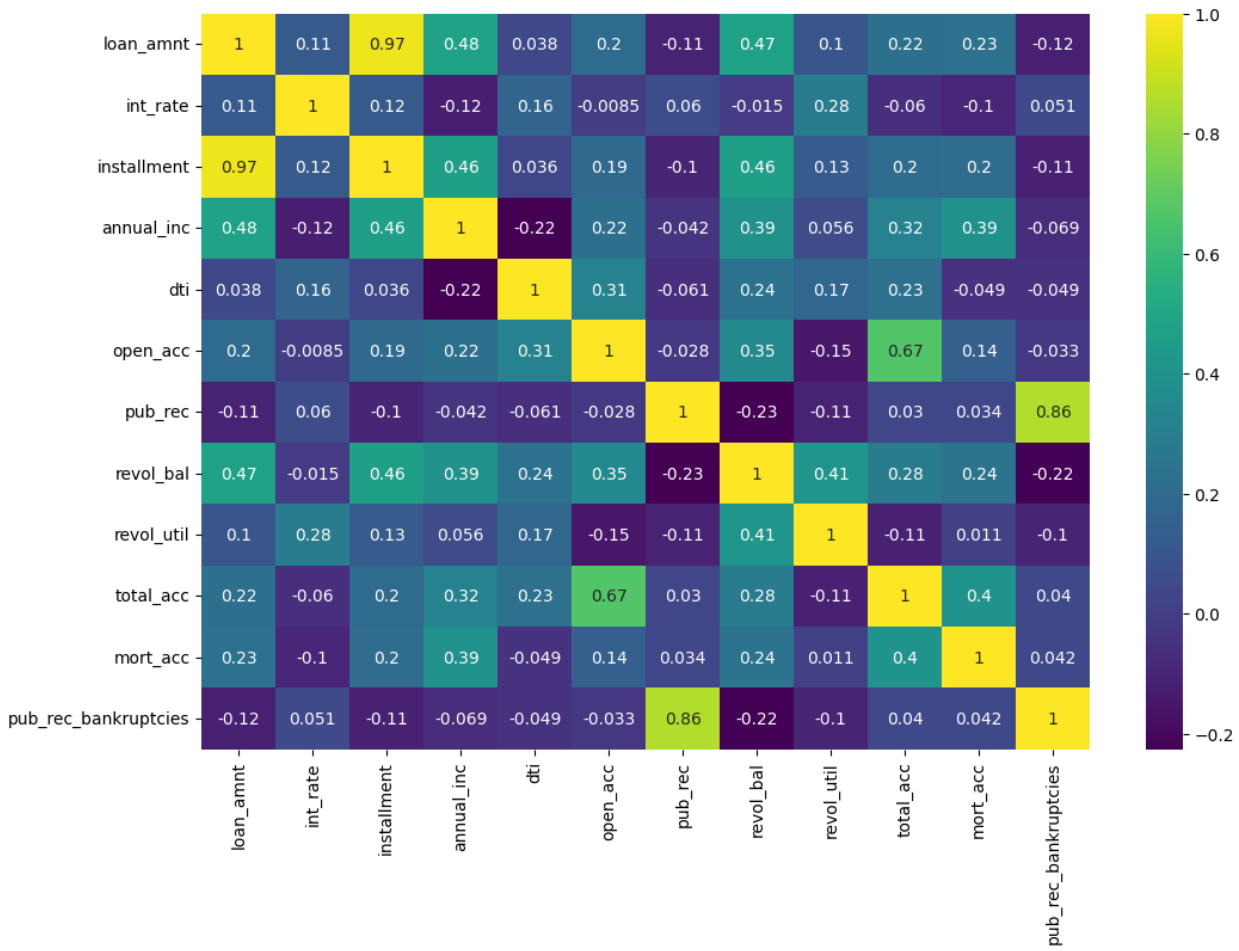
```
df_float = df[float_columns]
df_float
```

```
        loan_amnt  int_rate  installment  annual_inc    dti  open_acc
\
0         10000.0     11.44       329.48    117000.0  26.24      16.0
```

```
1          8000.0     11.99       265.68     65000.0  22.05       17.0

2         15600.0     10.49       506.97     43057.0  12.79       13.0

3          7200.0      6.49       220.65     54000.0   2.60        6.0

4         24375.0     17.27       609.33     55000.0  33.95       13.0

...           ...       ...          ...         ...    ...        ...

396024     6000.0     13.11       202.49     64000.0  10.81        7.0

396025    10000.0     10.99       217.38     40000.0  15.63        6.0

396026    21000.0     12.29       700.42    110000.0  21.45        6.0

396027     5000.0      9.99       161.32     56500.0  17.56       15.0

396028    21000.0     15.31       503.02     64000.0  15.88        9.0


        pub_rec  revol_bal  revol_util  total_acc  mort_acc  \
0           0.0    36369.0        41.8       25.0       0.0
1           0.0    20131.0        53.3       27.0       3.0
2           0.0    11987.0        92.2       26.0       0.0
3           0.0     5472.0        21.5       13.0       0.0
4           0.0    24584.0        69.8       43.0       1.0
...         ...        ...         ...        ...       ...
396024      0.0    11456.0        97.1        9.0       0.0
396025      0.0     1990.0        34.3       23.0       0.0
396026      0.0    43263.0        95.7        8.0       1.0
396027      0.0    32704.0        66.9       23.0       0.0
396028      0.0    15704.0        53.8       20.0       5.0

        pub_rec_bankruptcies
0                        0.0
1                        0.0
2                        0.0
3                        0.0
4                        0.0
...                      ...
396024                   0.0
396025                   0.0
396026                   0.0
396027                   0.0
396028                   0.0

[335867 rows x 12 columns]
```

```
plt.figure(figsize=(12,8))
sns.heatmap(df_float.corr(method='spearman'),annot=True,cmap='viridis'
```

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



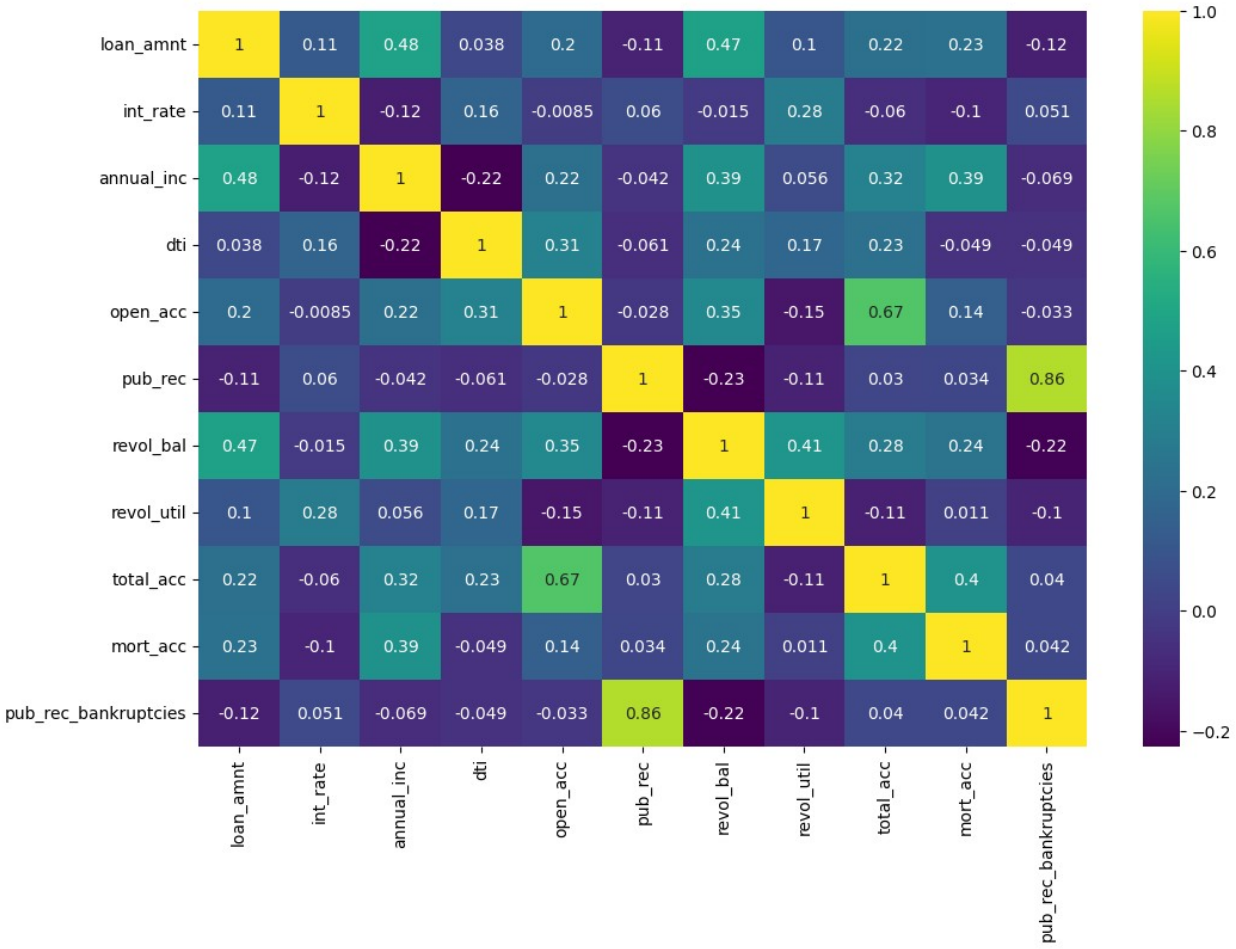
We noticed almost perfect correlation between "loan_amnt" the "installment" feature.

installment: The monthly payment owed by the borrower if the loan originates. loan_amnt: The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value. So, we can drop either one of those columns.

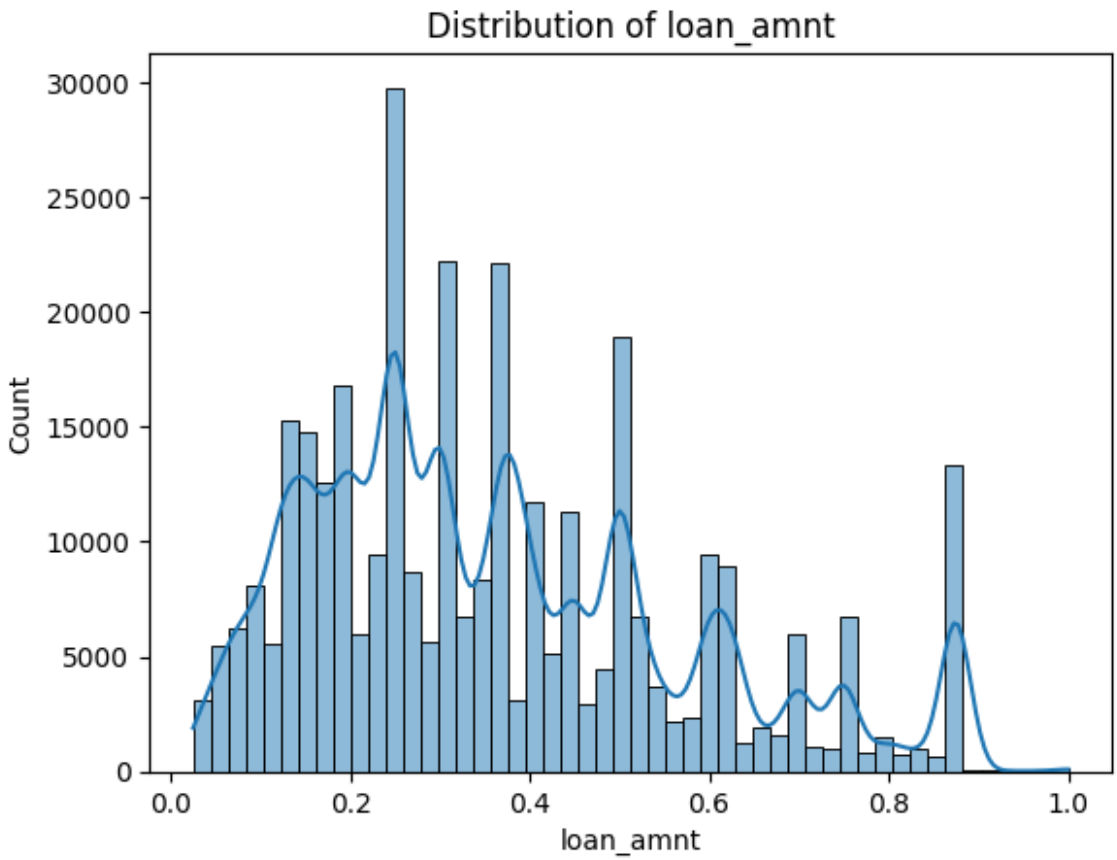
```
df.drop(columns=['installment'],axis=1,inplace=True)
df_float.drop(columns=['installment'],axis=1,inplace=True)
```

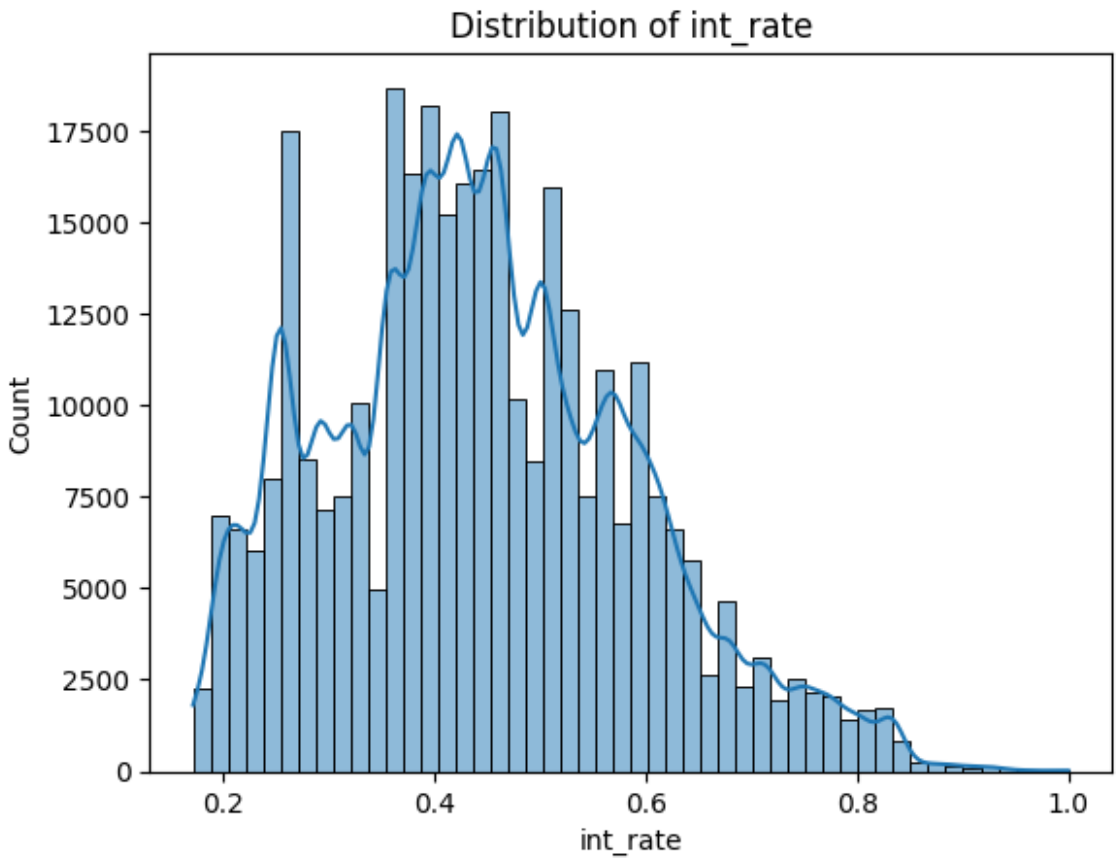
```
C:\Users\Rhythm Shah\AppData\Local\Temp\
ipykernel_22880\546979441.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df_float.drop(columns=['installment'],axis=1,inplace=True)
```

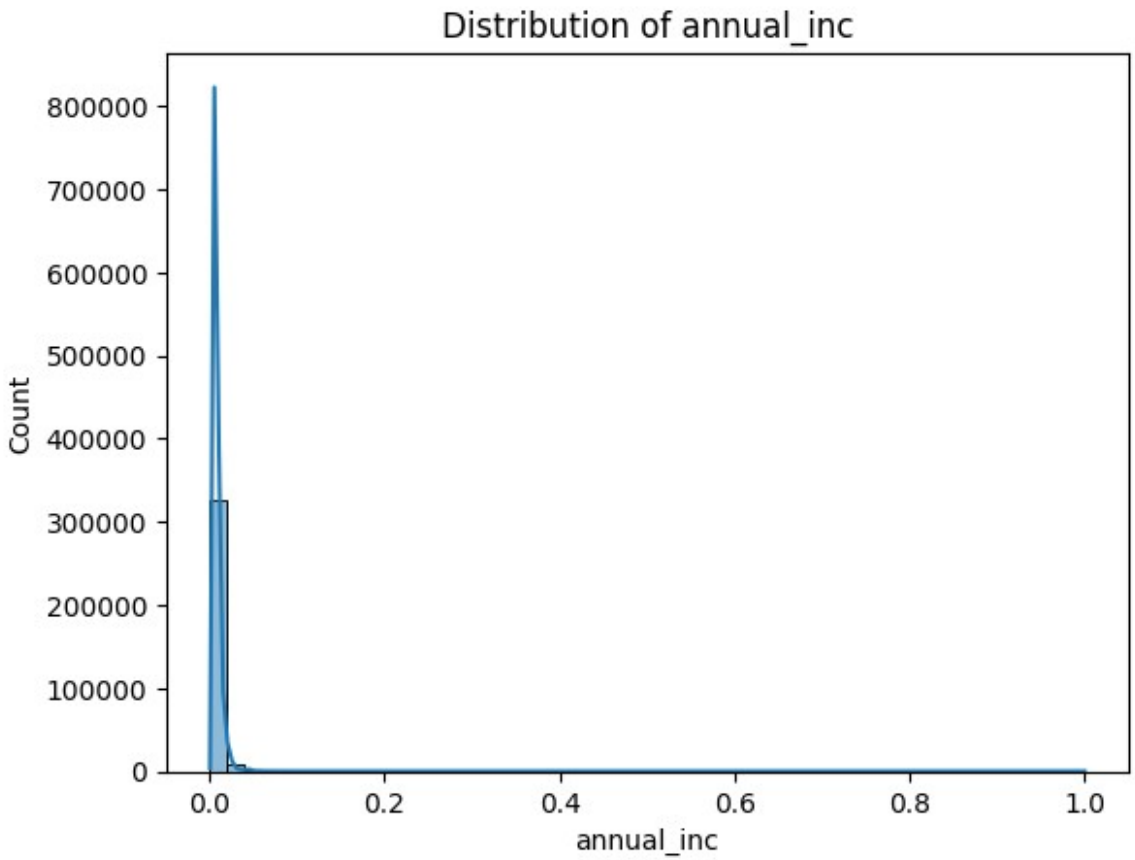
```
plt.figure(figsize=(12,8))
sns.heatmap(df_float.corr(method='spearman'),annot=True,cmap='viridis'
)
plt.show()
```

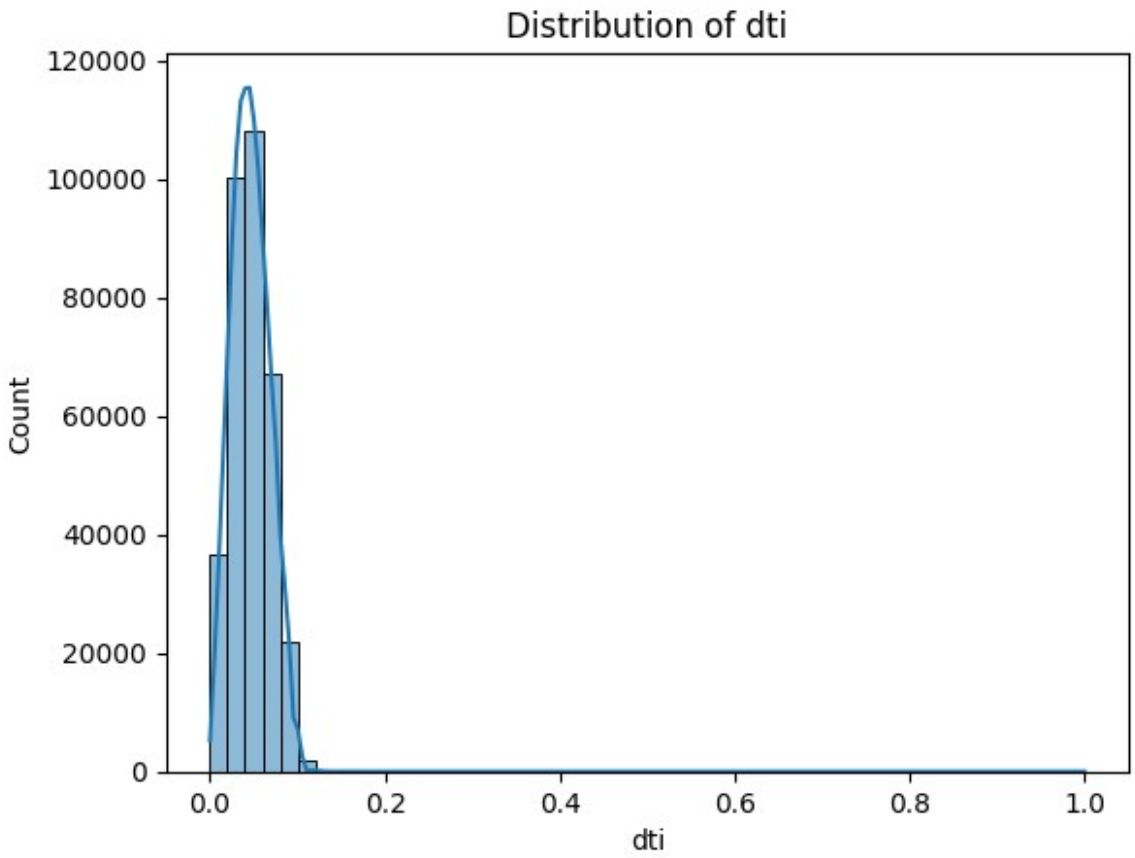


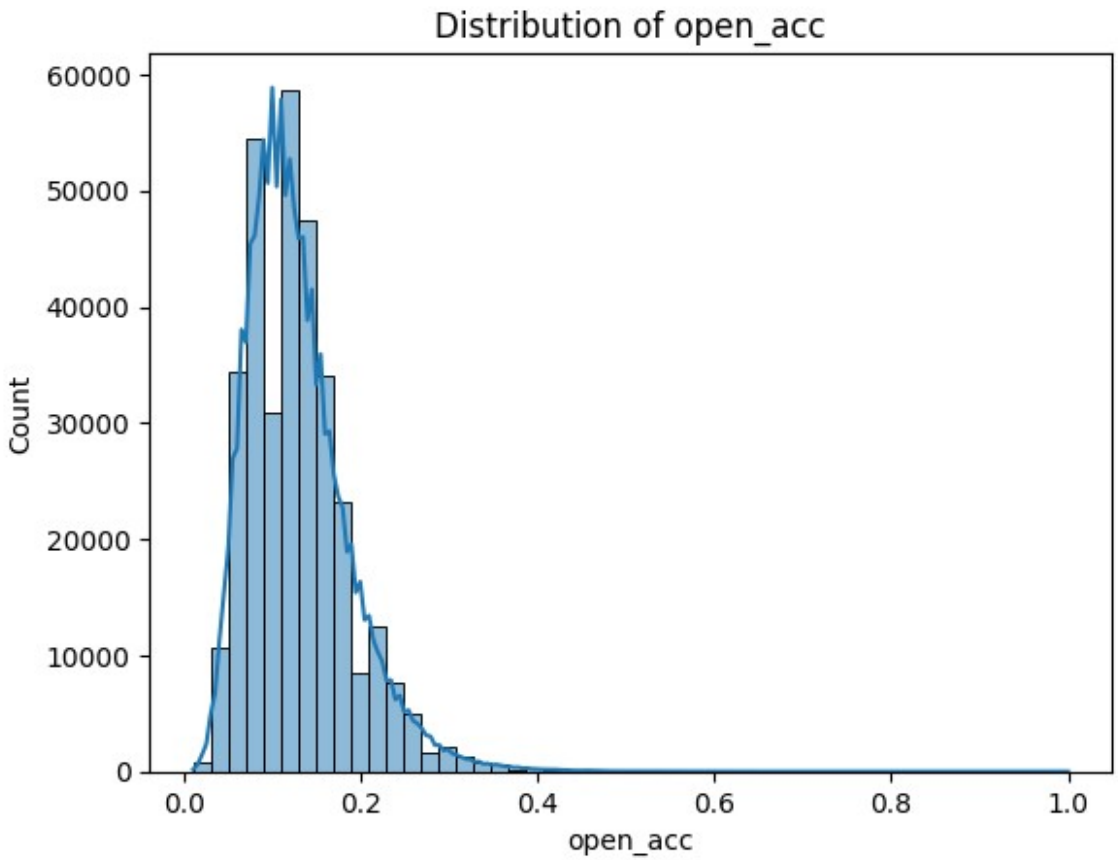

```
univariate_cols = df.select_dtypes('float64').columns.tolist()

for i in univariate_cols:
#     plt.figure(figsize=(12,5))
    plt.title("Distribution of {}".format(i))
    sns.histplot(df[i]/df[i].max(), kde=True, bins=50)
    plt.show()
```

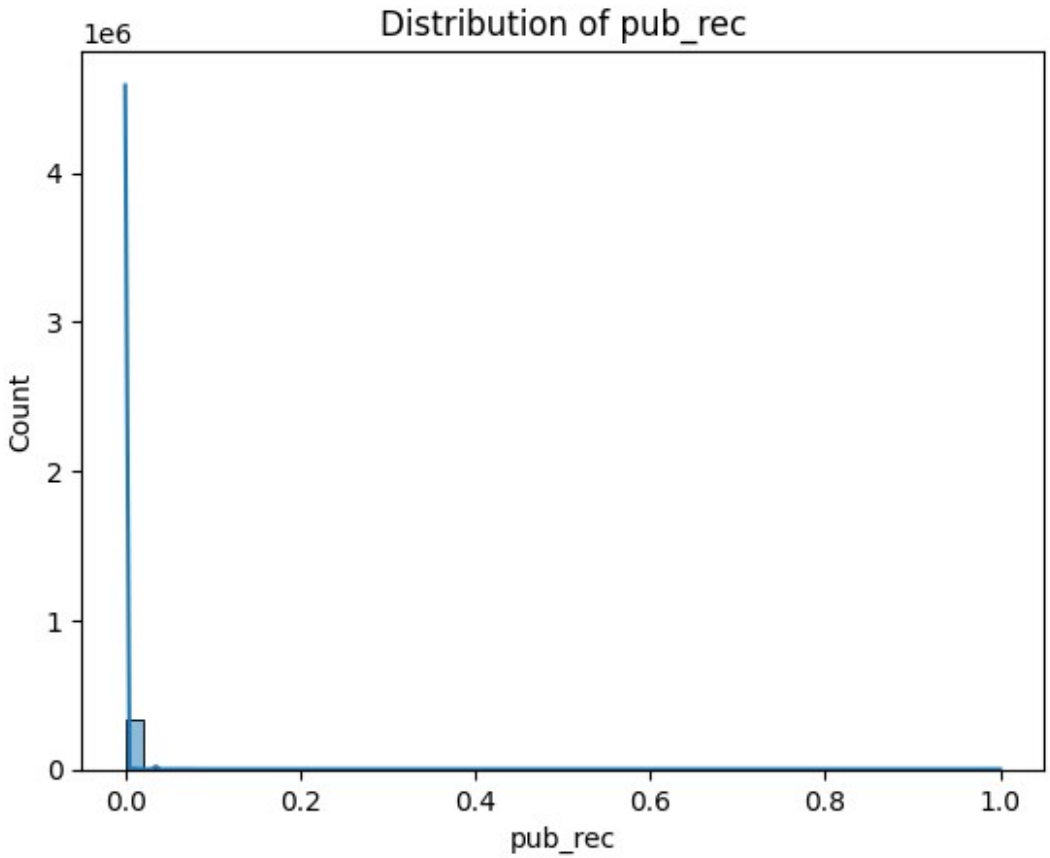

Distribution of loan_amnt
Count
loan_amnt
0
5000
10000
15000
20000
25000
30000
0.0
0.2
0.4
0.6
0.8
1.0

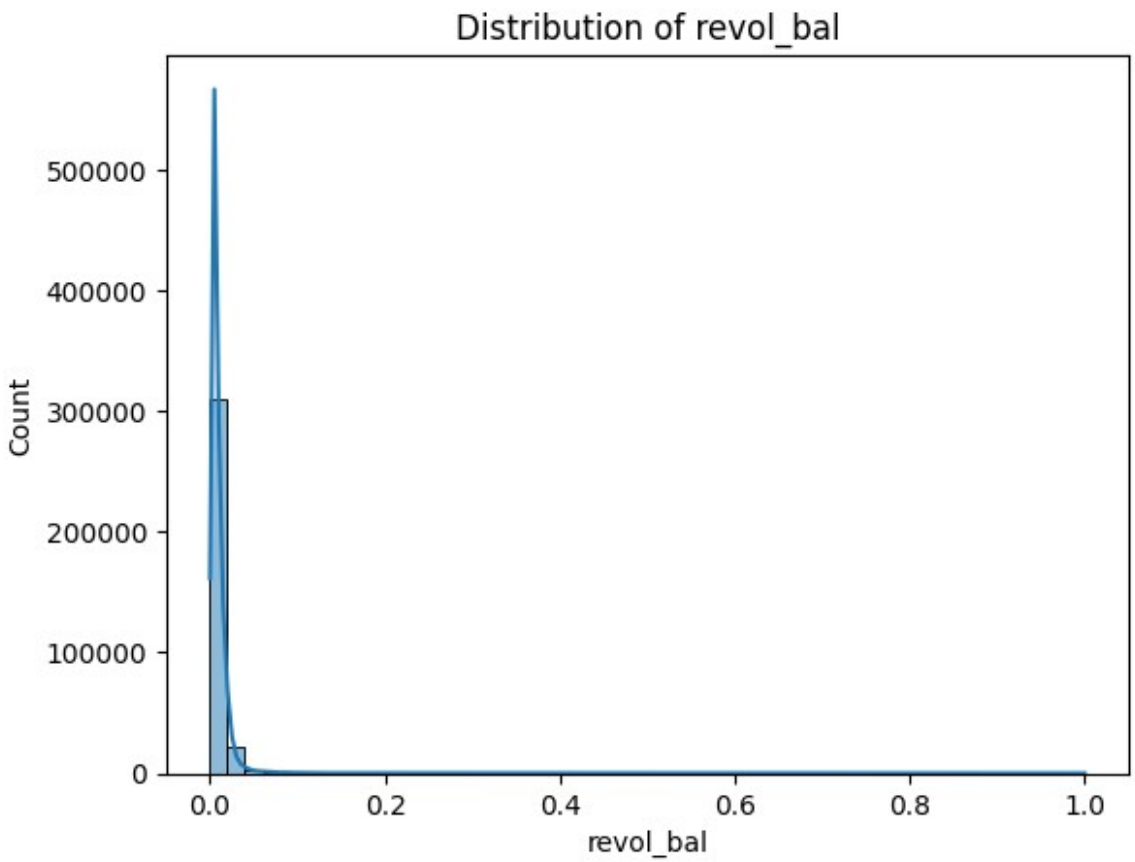

Distribution of int_rate
Count
17500
15000
12500
10000
7500
5000
2500
0
0.2
0.4
0.6
0.8
1.0
int_rate

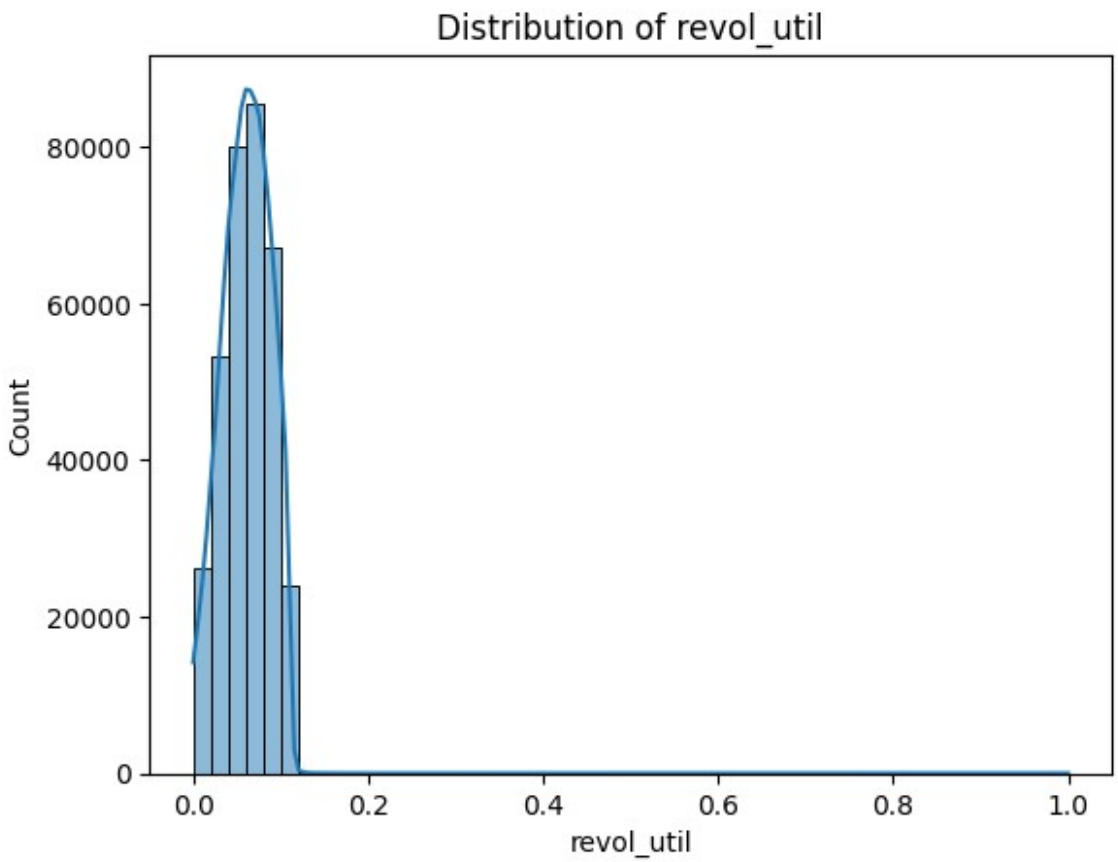

Distribution of annual_inc
800000
700000
600000
500000
400000
300000
200000
100000
0
Count
0.0
0.2
0.4
0.6
0.8
1.0
annual_inc

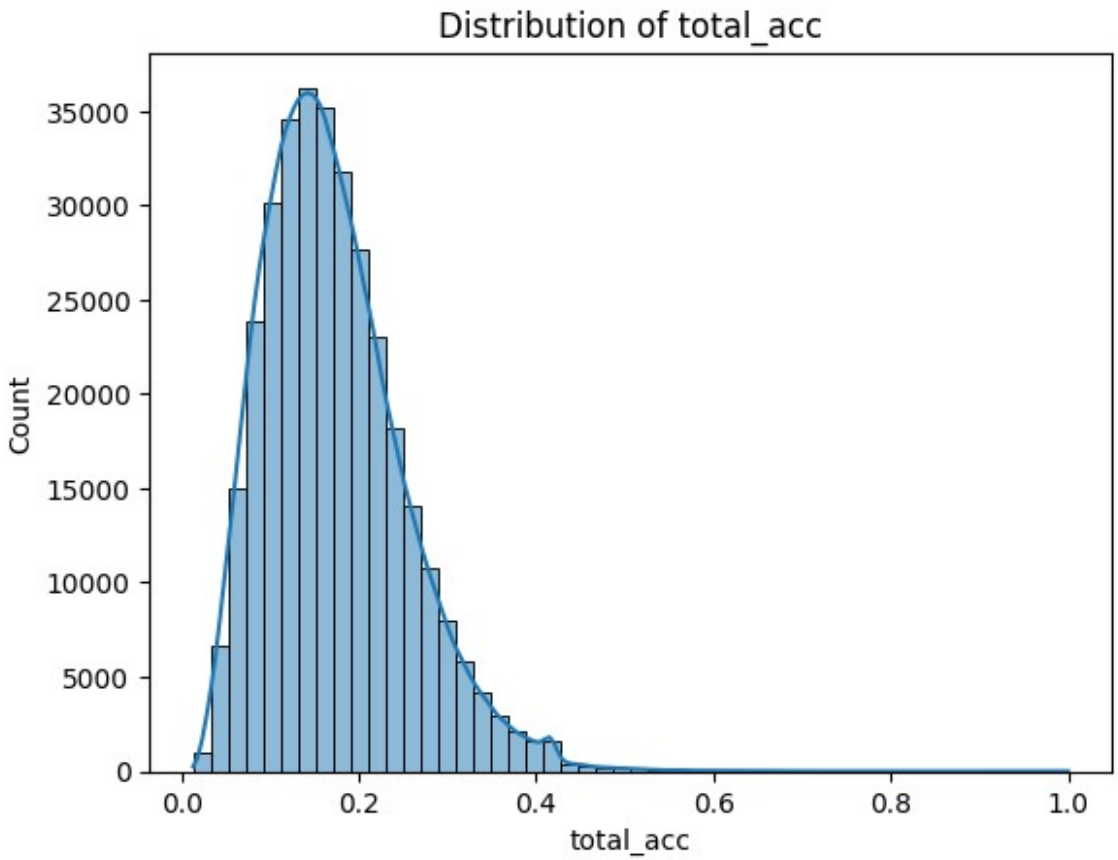

Distribution of dti
120000
100000
80000
60000
40000
20000
0
Count
0.0
0.2
0.4
0.6
0.8
1.0
dti

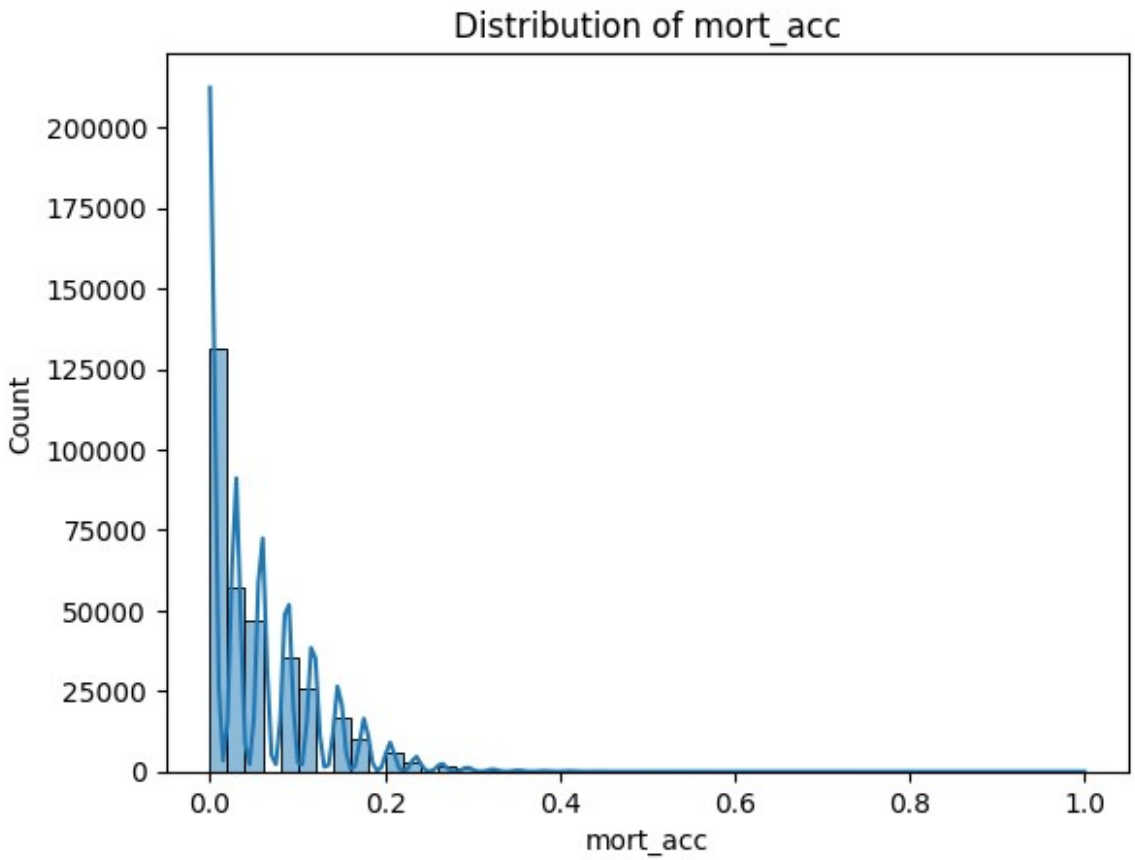

Distribution of open_acc
Count
60000
50000
40000
30000
20000
10000
0
0.0
0.2
0.4
0.6
0.8
1.0
open_acc

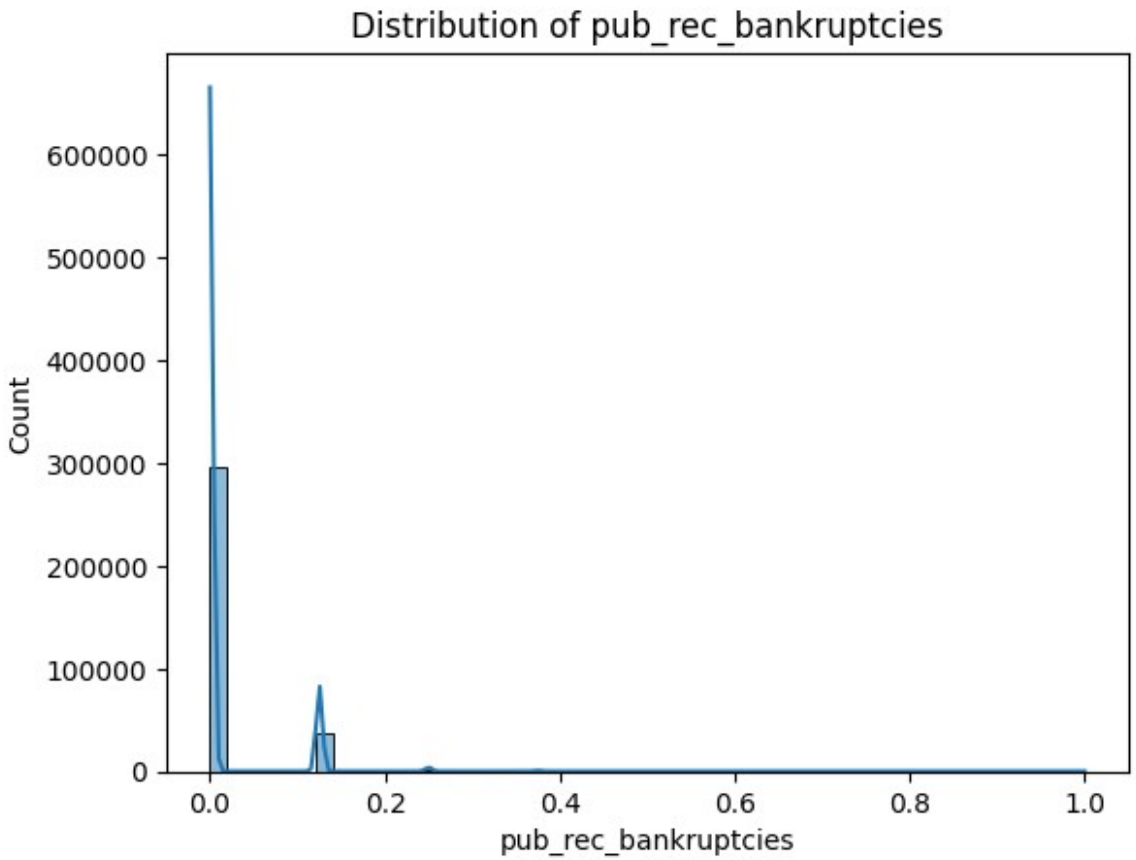

Distribution of pub_rec
1e6
4
3
2
1
0
Count
0.0
0.2
0.4
0.6
0.8
1.0
pub_rec


Distribution of revol_bal
Count
500000
400000
300000
200000
100000
0
0.0
0.2
0.4
0.6
0.8
1.0
revol_bal


Distribution of revol_util
Count
80000
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
1.0
revol_util

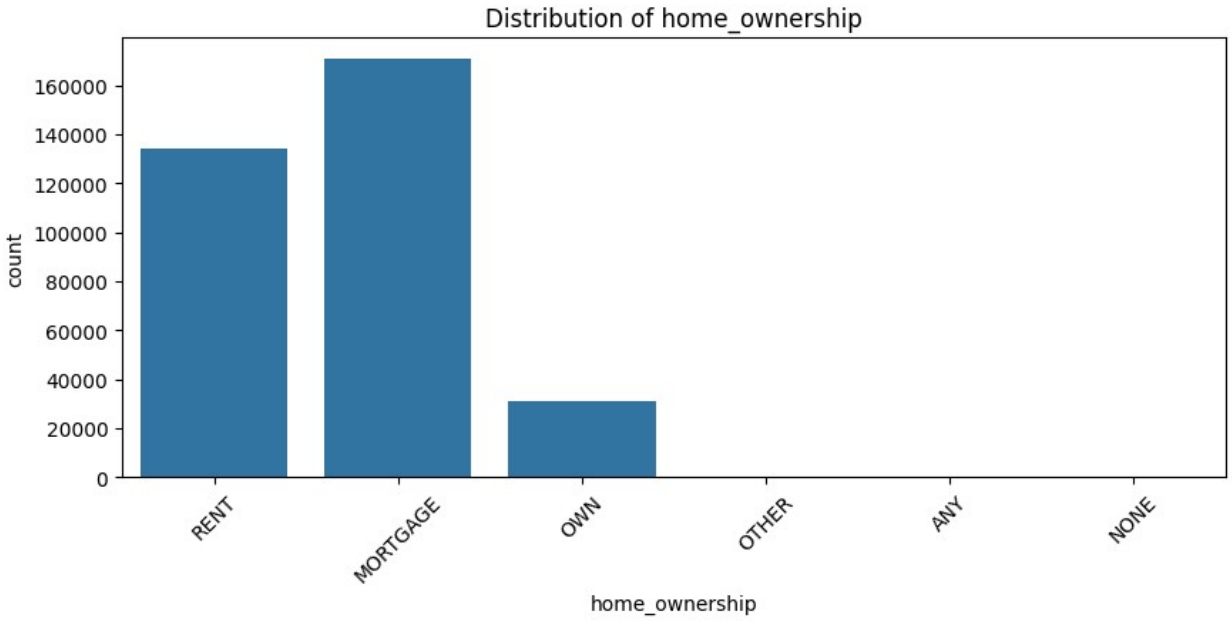

Distribution of total_acc
Count
35000
30000
25000
20000
15000
10000
5000
0
0.0
0.2
0.4
0.6
0.8
1.0
total_acc

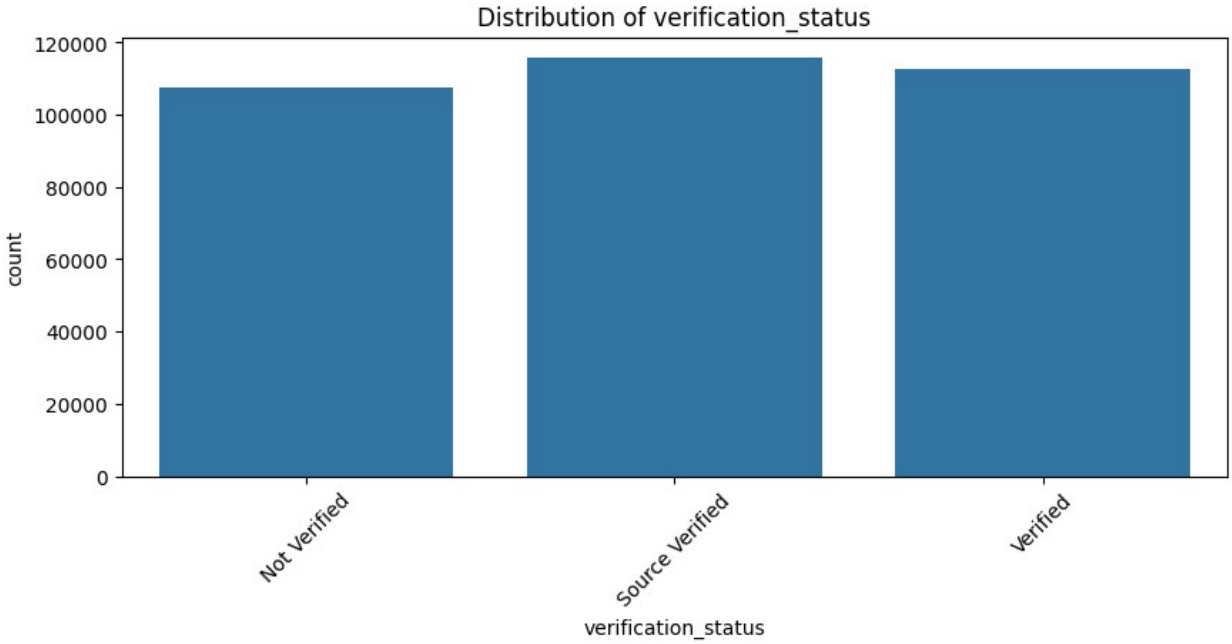

Distribution of mort_acc
Count
200000
175000
150000
125000
100000
75000
50000
25000
0
0.0
0.2
0.4
0.6
0.8
1.0
mort_acc

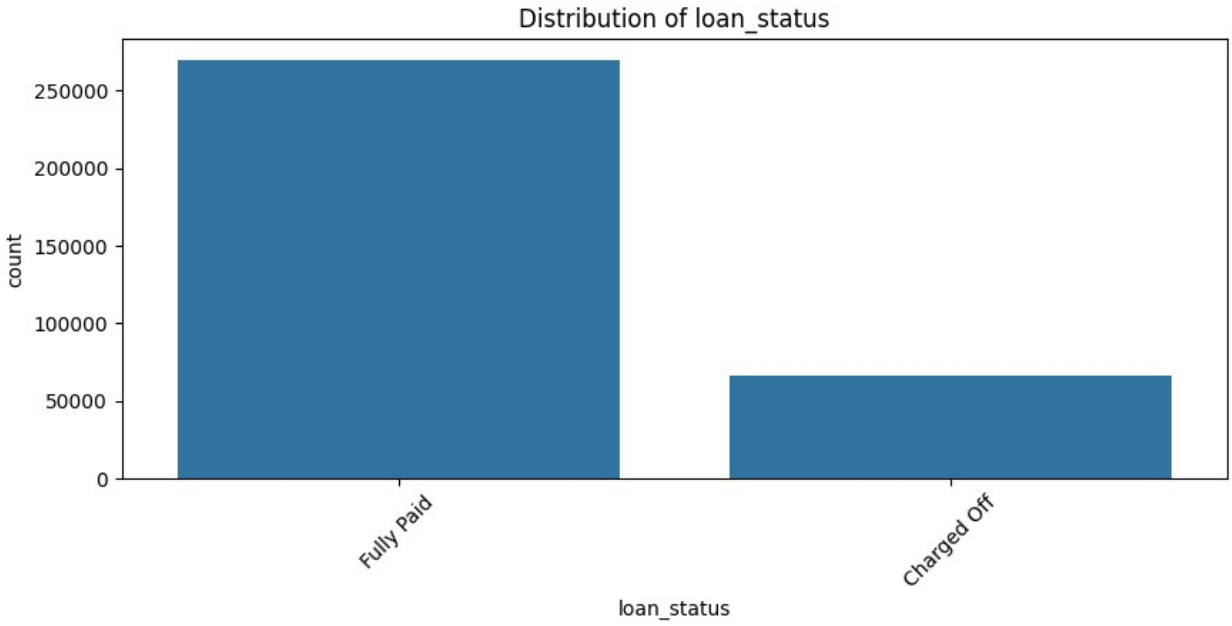


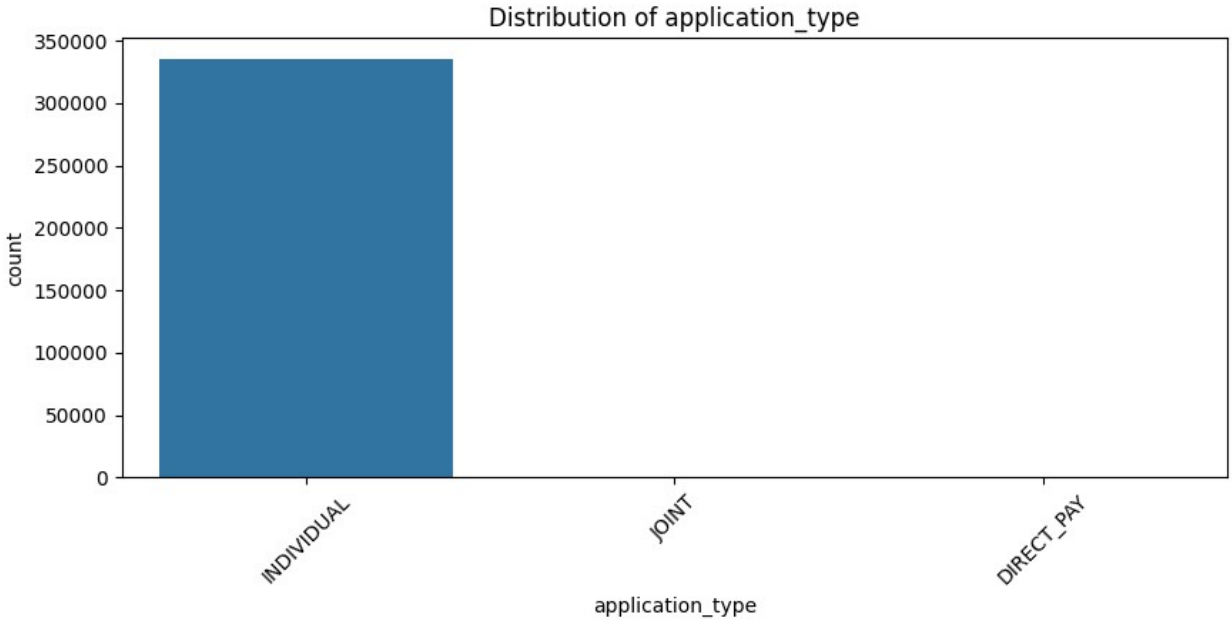
Most of the distribution is highly skewed which tells us that they might contain outliers Almost all the continuous features have outliers present in the dataset. They have to be standarsised

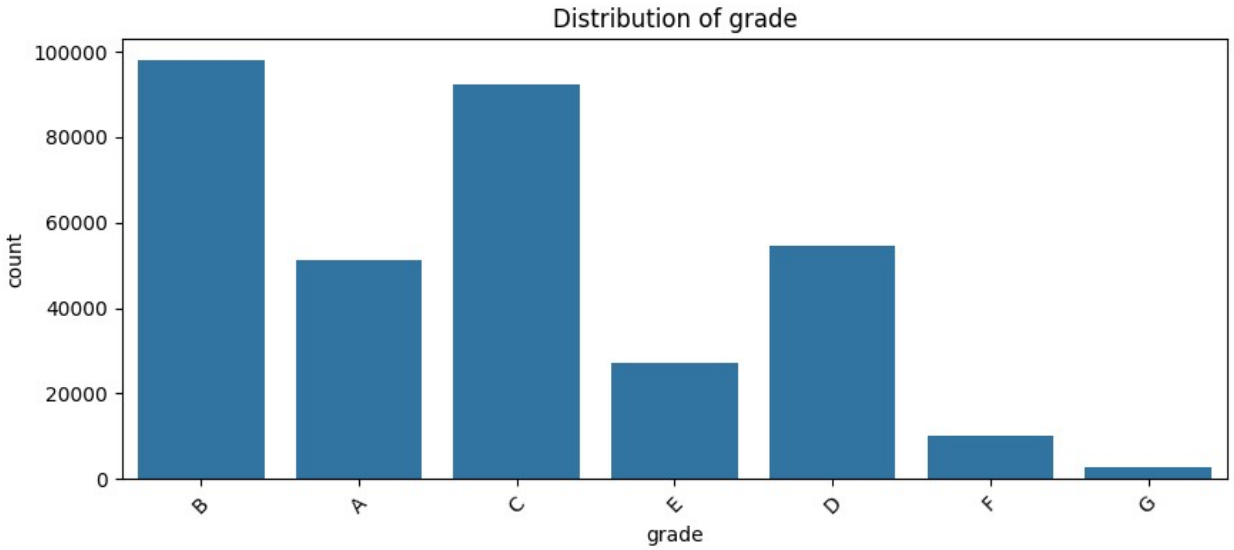
```
cat_vars = ['home_ownership', 'verification_status', 'loan_status',
'application_type', 'grade', 'sub_grade', 'term']
for i in cat_vars:
    plt.figure(figsize=(10, 4))
    plt.title(f'Distribution of {i}')
    sns.countplot(data=df, x=i)
    plt.xticks(rotation = 45)
    plt.show()
```

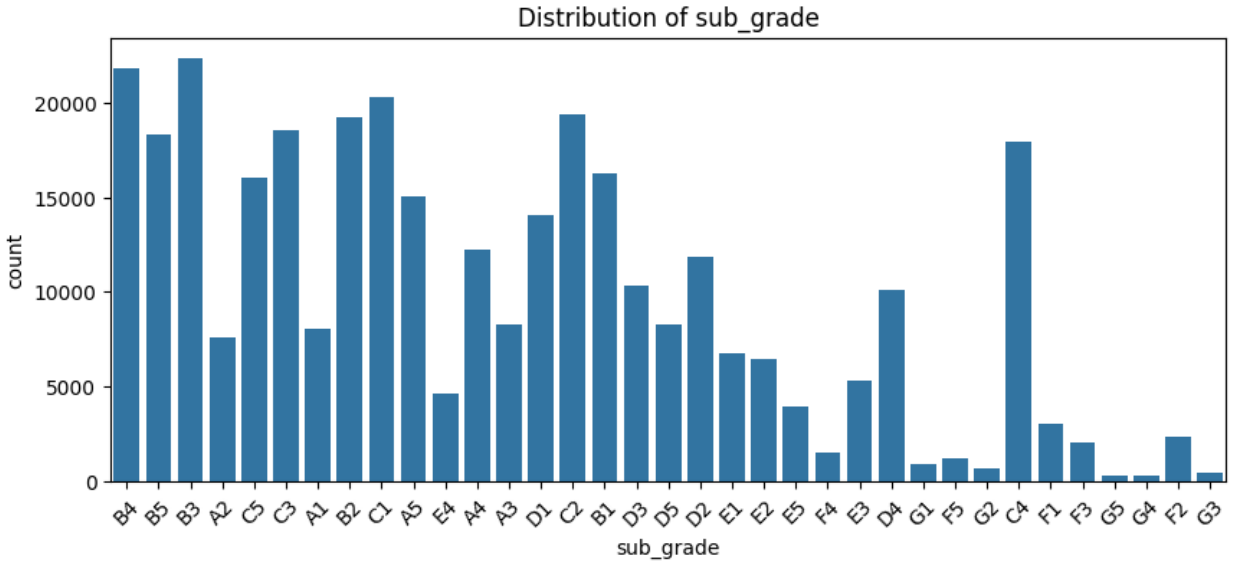

Distribution of home_ownership
count
160000
140000
120000
100000
80000
60000
40000
20000
0
RENT
MORTGAGE
OWN
OTHER
ANY
NONE
home_ownership

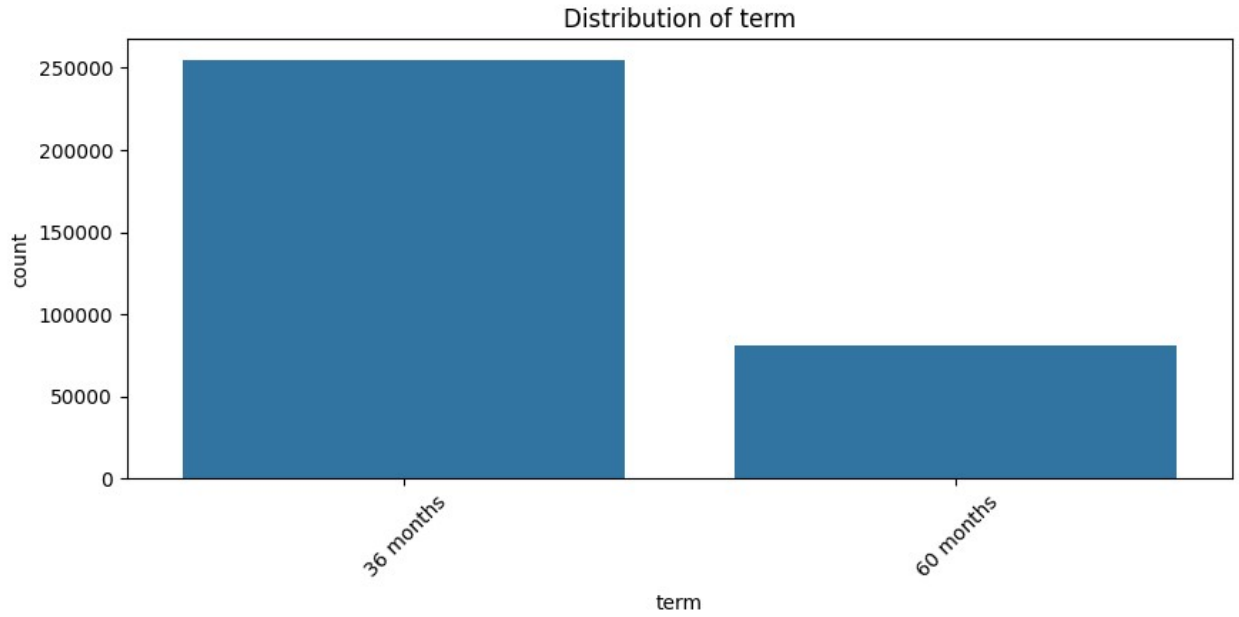

Distribution of verification_status
count
120000
100000
80000
60000
40000
20000
0
Not Verified
Source Verified
Verified
verification_status


Distribution of loan_status
count
250000
200000
150000
100000
50000
0
Fully Paid
Charged Off
loan_status


Distribution of application_type
count
350000
300000
250000
200000
150000
100000
50000
0
INDIVIDUAL
JOINT
DIRECT_PAY
application_type


Distribution of grade
count
100000
80000
60000
40000
20000
0
B
A
C
E
D
F
G
grade


Distribution of sub_grade
count
20000
15000
10000
5000
0
B4
B5
B3
A2
C5
C3
A1
B2
C1
A5
E4
A4
A3
D1
C2
B1
D3
D5
D2
E1
E2
E5
F4
E3
D4
G1
F5
G2
C4
F1
F3
G5
G4
F2
G3
sub_grade



All the application type is Individual

Most of the loan tenure is disbursed for 36 months

The grade of majority of people those who have took the loan is 'B' and have subgrade 'B3'.

So from that we can infer that people with grade 'B' and subgrade 'B3' are more likely to fully pay the loan.

Most of the people took loan for 36 months and full paid on time

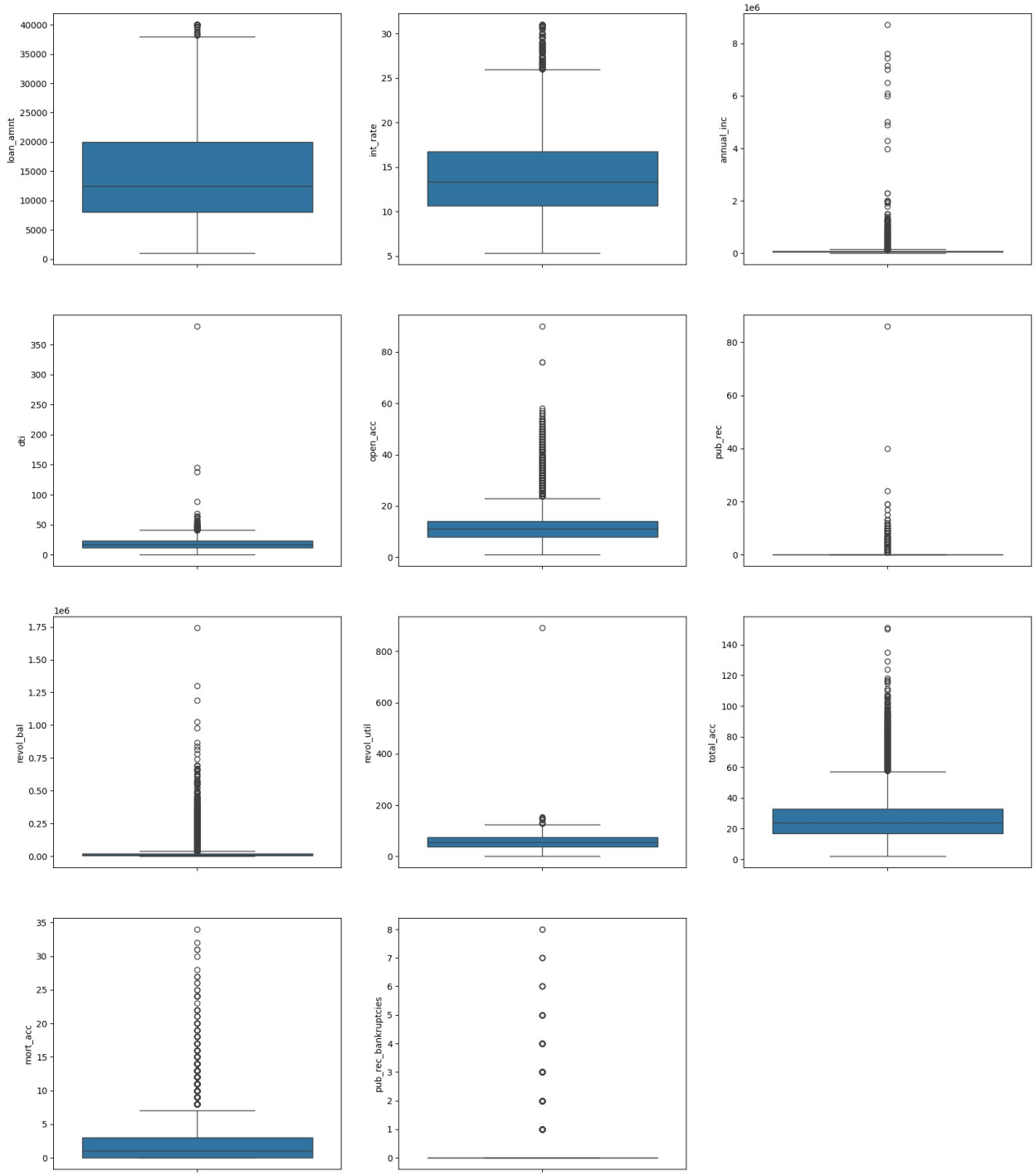
Most of people have home ownership as mortgage and rent

Most of the people took loan for debt consolidations

All the application type is Individual Most of the loan tenure is disbursed for 36 months The grade of majority of people those who have took the loan is 'B' and have subgrade 'B3'. So from that we can infer that people with grade 'B' and subgrade 'B3' are more likely to fully pay the loan.

Outlier Treatment

```
count = 0
plt.figure(figsize=(20,30))
for i in univariate_cols:
    count += 1
    plt.subplot(5,3,count)
    sns.boxplot(y= df[i])
```




```
df.shape
```

```
(335867, 26)
```

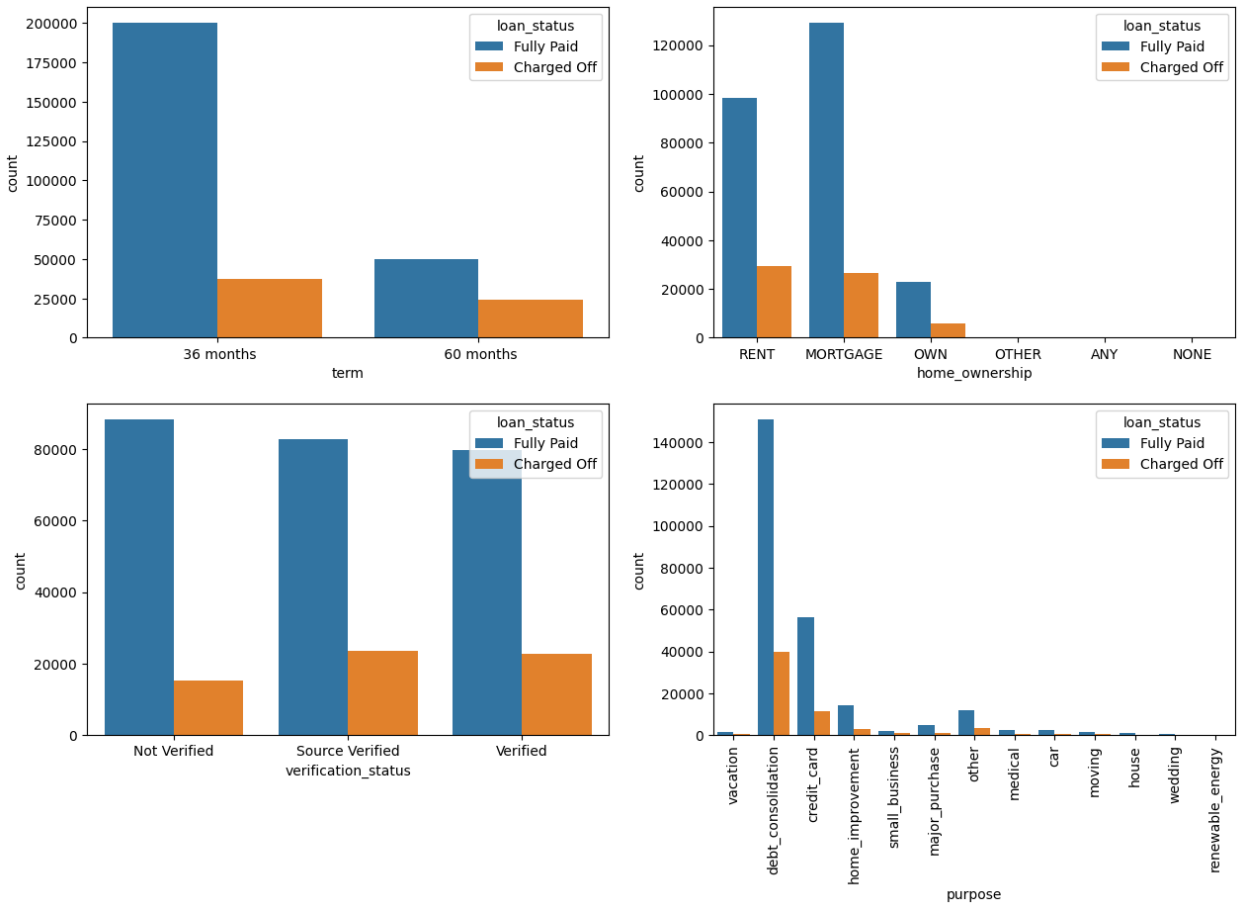
```
for col in univariate_cols:
    mean=df[col].mean()
    std=df[col].std()
```

```
    upper_limit=mean+3*std
    lower_limit=mean-3*std

    df=df[(df[col]<upper_limit) & (df[col]>lower_limit)]

df.shape
```

```
(312060, 26)
```

```
plt.figure(figsize=(15,20))

plt.subplot(4,2,1)
sns.countplot(x='term',data=df,hue='loan_status')

plt.subplot(4,2,2)
sns.countplot(x='home_ownership',data=df,hue='loan_status')

plt.subplot(4,2,3)
sns.countplot(x='verification_status',data=df,hue='loan_status')

plt.subplot(4,2,4)
g=sns.countplot(x='purpose',data=df,hue='loan_status')
g.set_xticklabels(g.get_xticklabels(),rotation=90)

plt.show()
```

```
C:\Users\Rhythm Shah\AppData\Local\Temp\
ipykernel_22880\2970726303.py:14: UserWarning: set_ticklabels() should
only be used with a fixed number of ticks, i.e. after set_ticks() or
using a FixedLocator.
  g.set_xticklabels(g.get_xticklabels(),rotation=90)
```



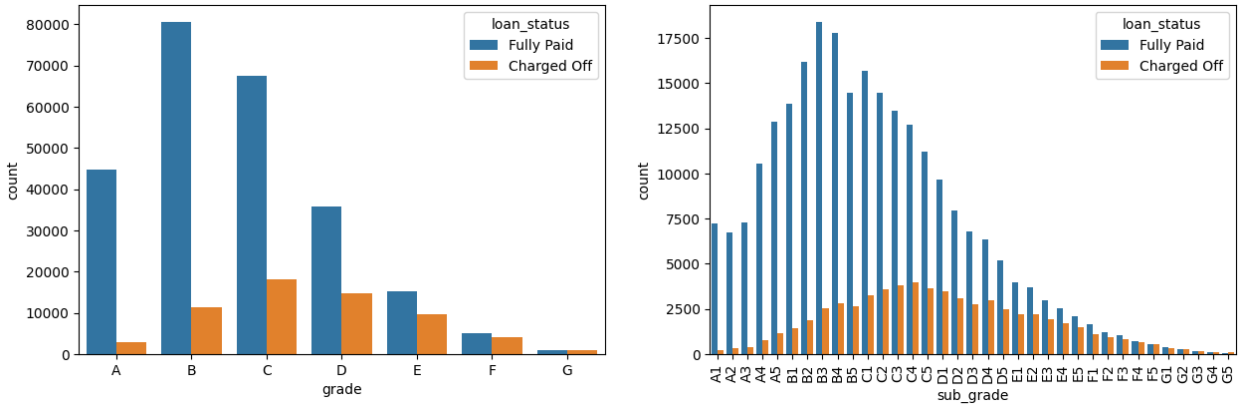
Most of the people took loan for 36 months and full paid on time

Most of people have home ownership as mortgage and rent

Most of the people took loan for debt consolidations

1. People with grades 'A' are more likely to fully pay their loan. (T/F)

```
plt.figure(figsize=(15, 10))
warnings.filterwarnings("ignore")

plt.subplot(2, 2, 1)
grade = sorted(df.grade.unique().tolist())
sns.countplot(x='grade', data=df, hue='loan_status', order=grade)

plt.subplot(2, 2, 2)
sub_grade = sorted(df.sub_grade.unique().tolist())
g = sns.countplot(x='sub_grade', data=df, hue='loan_status',
order=sub_grade)
g.set_xticklabels(g.get_xticklabels(), rotation=90)
plt.show()
```

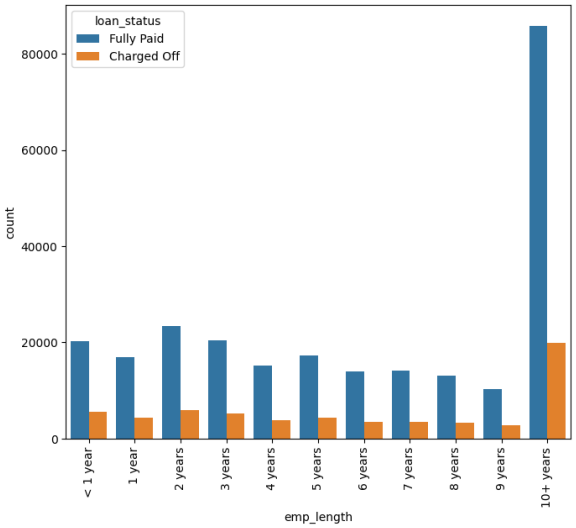


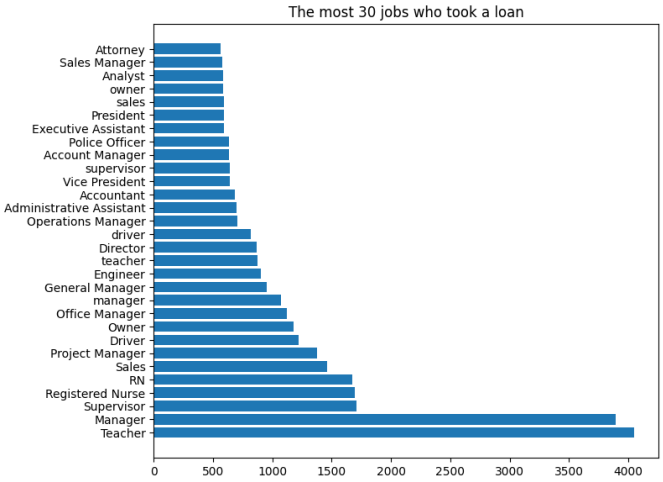
The grade of majority of people those who have fully paid the loan is 'B' and have subgrade 'B3'.

So from that we can infer that people with grade 'B' and subgrade 'B3' are more likely to fully pay the loan.

1. Name the top 2 afforded job titles.

```
plt.figure(figsize=(15,12))

plt.subplot(2,2,1)
order = ['< 1 year', '1 year', '2 years', '3 years', '4 years', '5 years',
         '6 years', '7 years', '8 years', '9 years', '10+ years',]
g=sns.countplot(x='emp_length',data=df,hue='loan_status',order=order)
g.set_xticklabels(g.get_xticklabels(),rotation=90)

plt.subplot(2,2,2)
plt.barh(df.emp_title.value_counts()
[:30].index,df.emp_title.value_counts()[:30])
plt.title("The most 30 jobs who took a loan")
plt.tight_layout()
plt.show()
```



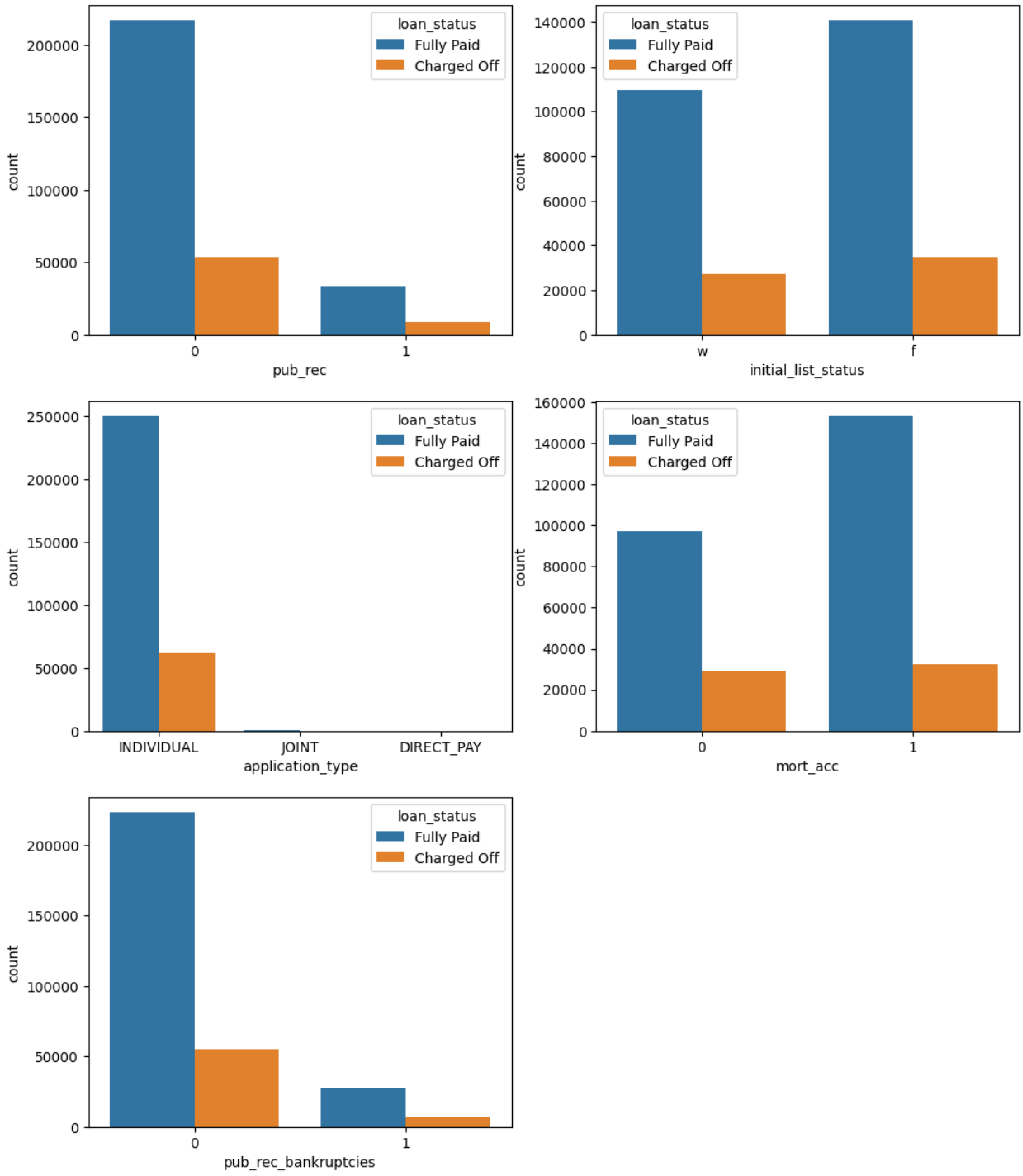


Manager and Teacher are the most afforded loan on titles

Person who employed for more than 10 years has successfully paid of the loan

```
def f1(number):
    if number == 0.0:
        return 0
    else:
        return 1

def f2(number):
    if number == 0.0:
        return 0
    elif number >= 1.0:
        return 1
    else:
        return number

df['pub_rec']=df.pub_rec.apply(f1)
df['mort_acc']=df.mort_acc.apply(f2)
df['pub_rec_bankruptcies']=df.pub_rec_bankruptcies.apply(f2)

plt.figure(figsize=(12,30))

plt.subplot(6,2,1)
sns.countplot(x='pub_rec',data=df,hue='loan_status')

plt.subplot(6,2,2)
sns.countplot(x='initial_list_status',data=df,hue='loan_status')

plt.subplot(6,2,3)
sns.countplot(x='application_type',data=df,hue='loan_status')

plt.subplot(6,2,4)
```

```
sns.countplot(x='mort_acc',data=df,hue='loan_status')

plt.subplot(6,2,5)
sns.countplot(x='pub_rec_bankruptcies',data=df,hue='loan_status')

plt.show()
```



Most the loan disbursed to the people whose do not hold bankrupties record have successfully paid loan

Feature Engineering

```
for i in df.columns:
    print(i, '-->> ', df[i].unique(), '\n')
```

```
loan_amnt -->>  [10000.  8000. 15600. ... 36275. 34175. 36475.]

term -->>  [' 36 months' ' 60 months']

int_rate -->>  [11.44 11.99 10.49  6.49 17.27 13.33  5.32 11.14 10.99
16.29 13.11  9.17
  6.62  8.39 21.98 12.29  7.9   6.97 15.61 13.35 12.12  9.99  8.19
18.75
 14.99 13.67 13.98 17.86 21.49 12.99 18.54 17.1  18.25 11.67  6.24
8.18
 12.35 14.16 18.55 22.15 15.99 24.99  9.67 19.19 21.   12.69 10.74
6.68
 19.22 16.99 16.55 19.97  7.89 24.7  16.49 25.78 25.83 13.99 15.22
15.31
  7.69 19.53 10.16  7.62  9.75 13.68 15.88 14.65 23.83 10.75 18.49
20.31
 17.57 27.31 22.99 14.33 13.53 22.45 14.64 24.5  17.99  9.16 12.49
11.55
 17.76 20.49 22.7  17.56  6.03  6.89 19.52  8.9  14.3   9.49 25.99
24.08
 13.05 14.98 16.59 10.15 25.89 21.99 23.99 11.49 14.47 11.53  8.67
13.49
  8.59 10.64 25.44  9.71 16.2  19.24 24.11 15.8  14.49 18.99 23.28
19.99
 18.24 14.09  9.25 19.05 17.77 18.92 13.65  7.12 16.78 14.46 12.59
16.24
 25.49  7.39 10.78 12.85 12.39 21.18 21.97  6.39  7.49 25.57 20.99
11.47
  7.26 14.31 24.24 23.43 19.47 23.32  9.76  6.99 21.15 21.48  6.92
22.95
 18.85 15.59 15.1  19.72 11.22 15.81 11.48 13.66 23.76  9.8  17.14
18.84
  8.49 25.29 22.47 19.2  11.39 22.4  12.79 18.2  13.18 25.28 12.05
20.2
 21.67 20.8  13.44 25.8  14.85  6.    7.99 23.1   7.91 21.7  26.06
5.93
 15.77  8.99 20.5  23.63 14.48 22.2  15.41 23.5  22.9  19.48 23.4
23.13
 12.88 24.49 21.6   8.38 17.97  8.6  12.74 25.88 24.89 26.77 19.89
20.75
 24.83  7.59 23.7  23.33 26.24 24.2  26.57 25.11  7.24 22.78 22.39
26.14
```

```
 23.26 22.35 26.49 25.69 25.65 24.74 22.74 26.99  8.24 25.09 24.76
24.52
 24.33]

grade -->>  ['B' 'A' 'C' 'E' 'D' 'F' 'G']

sub_grade -->>  ['B4' 'B5' 'B3' 'A2' 'C5' 'C3' 'A1' 'B2' 'A5' 'E4'
'C1' 'A4' 'A3' 'D1'
 'C2' 'B1' 'D3' 'D5' 'D2' 'E1' 'E2' 'E5' 'F4' 'E3' 'D4' 'G1' 'F5' 'G2'
 'C4' 'F1' 'F3' 'G4' 'F2' 'G3' 'G5']


emp_title -->>  ['Marketing' 'Credit analyst ' 'Statistician' ...
 "Michael's Arts & Crafts" 'licensed bankere' 'Gracon Services, Inc']

emp_length -->>  ['10+ years' '4 years' '< 1 year' '6 years' '9 years'
'2 years' '3 years'
 '7 years' '8 years' '5 years' '1 year']

home_ownership -->>  ['RENT' 'MORTGAGE' 'OWN' 'OTHER' 'ANY' 'NONE']

annual_inc -->>  [117000.  65000.  43057. ...  40311.  36111.  47212.]


verification_status -->>  ['Not Verified' 'Source Verified'
'Verified']

issue_d -->>  ['Jan-2015' 'Nov-2014' 'Apr-2013' 'Sep-2015' 'Sep-2012'
'Oct-2014'
 'Apr-2012' 'Jun-2013' 'Dec-2015' 'Oct-2012' 'Jul-2014' 'Feb-2013'
 'Oct-2015' 'Jan-2014' 'Mar-2016' 'Apr-2014' 'Jun-2014' 'Oct-2013'
 'May-2013' 'Feb-2015' 'Jun-2015' 'Mar-2013' 'Jun-2016' 'Mar-2014'
 'Nov-2013' 'Dec-2014' 'Sep-2013' 'May-2016' 'Jul-2015' 'Apr-2015'
 'Jul-2013' 'Aug-2013' 'Aug-2014' 'Dec-2013' 'Mar-2012' 'Mar-2015'
 'Jul-2012' 'Feb-2014' 'Dec-2012' 'Sep-2014' 'Nov-2012' 'Apr-2016'
 'May-2012' 'May-2014' 'Jun-2012' 'Aug-2012' 'May-2015' 'Oct-2016'
 'Aug-2015' 'Jul-2016' 'Aug-2016' 'Feb-2016' 'Jan-2013' 'Nov-2015'
 'Jan-2016' 'Nov-2016' 'Dec-2016' 'Sep-2016']

loan_status -->>  ['Fully Paid' 'Charged Off']

purpose -->>  ['vacation' 'debt_consolidation' 'credit_card'
'home_improvement'
 'small_business' 'major_purchase' 'other' 'medical' 'car' 'moving'
 'house' 'wedding' 'renewable_energy']

title -->>  ['Vacation' 'Debt consolidation' 'Credit card refinancing'
... 'Cc Debt'
 'Credit buster ' 'Loanforpayoff']

dti -->>  [26.24 22.05 12.79 ... 41.68 40.21 40.57]
```

```
earliest_cr_line -->>  ['Jun-1990' 'Jul-2004' 'Aug-2007' 'Sep-2006'
'Mar-1999' 'Jan-2005'
 'Aug-2005' 'Sep-1994' 'Jun-1994' 'Dec-1997' 'Dec-1990' 'Apr-1995'
 'May-2001' 'Mar-1982' 'Sep-1996' 'Jan-1990' 'Mar-2000' 'Jan-2006'
 'Jan-2003' 'May-2008' 'Jun-2004' 'Jan-1999' 'Apr-1994' 'Apr-1998'
 'Jul-2007' 'Oct-2007' 'May-1997' 'Jul-2006' 'Sep-2003' 'Feb-2002'
 'Jan-1992' 'Aug-2001' 'Dec-2010' 'Oct-1999' 'Sep-2004' 'Jul-2003'
 'Apr-2000' 'Dec-2004' 'Jun-1995' 'Dec-2003' 'Jul-1994' 'Oct-1990'
 'Apr-1999' 'Dec-2001' 'May-2003' 'Oct-2002' 'Mar-2004' 'Aug-2003'
 'Oct-2000' 'Nov-2004' 'Mar-2010' 'Mar-1996' 'May-1994' 'Jun-1996'
 'Nov-1986' 'Jan-2002' 'Mar-2001' 'Sep-2012' 'Apr-2006' 'May-1998'
 'Dec-2002' 'Oct-2005' 'May-1990' 'Jun-2003' 'Jun-2001' 'Feb-2001'
 'Jun-2006' 'Oct-2006' 'Aug-1993' 'Apr-2001' 'Nov-2001' 'Feb-2003'
 'Jun-1993' 'Sep-1992' 'Nov-1992' 'Jun-1983' 'Jul-1999' 'Sep-1997'
 'Nov-1993' 'Feb-1993' 'Apr-2007' 'Nov-1999' 'Nov-2005' 'Dec-1992'
 'Mar-1986' 'Dec-1988' 'May-1989' 'Dec-2000' 'Mar-2005' 'Jun-2010'
 'Jan-2001' 'Sep-2001' 'Jan-1994' 'Aug-2002' 'Nov-2003' 'Jan-2011'
 'Aug-2008' 'Jun-2005' 'Nov-1997' 'May-1993' 'Sep-2005' 'Apr-1986'
 'Aug-1996' 'Aug-1997' 'Jul-2005' 'May-2011' 'Sep-2002' 'Aug-1999'
 'May-1996' 'Feb-1992' 'Sep-1999' 'Jul-2001' 'Oct-2001' 'Oct-2008'
 'Nov-2007' 'Apr-1997' 'Jun-1986' 'Sep-1998' 'Jun-1982' 'Oct-1981'
 'Feb-1994' 'Dec-1984' 'Nov-1991' 'Nov-2006' 'Aug-2000' 'Oct-2004'
 'Apr-1988' 'May-2004' 'Aug-1988' 'Mar-1994' 'Aug-2004' 'Dec-2006'
 'Nov-1998' 'Oct-1997' 'Mar-1989' 'Jul-1982' 'Mar-1997' 'Oct-1994'
 'Jul-1998' 'Jun-2002' 'May-1991' 'Oct-2011' 'Sep-2007' 'Jan-2007'
 'Jan-2010' 'Mar-1987' 'Feb-1997' 'Oct-1986' 'Mar-2002' 'Jul-1993'
 'Mar-2007' 'Aug-1989' 'May-2007' 'Dec-1993' 'Jun-1989' 'Jun-1997'
 'Oct-2003' 'Apr-1996' 'Oct-1998' 'Mar-1983' 'Mar-1985' 'Oct-1993'
 'Apr-2002' 'Apr-2003' 'Jul-1985' 'May-1978' 'Sep-2010' 'Oct-1996'
 'Sep-2009' 'Jan-2000' 'Sep-1987' 'Jan-1995' 'May-2000' 'Jun-1981'
 'Nov-1996' 'Feb-1998' 'Aug-1967' 'Dec-1999' 'Aug-2006' 'Nov-2009'
 'Jul-1992' 'Jul-1991' 'Mar-1990' 'May-1986' 'Jun-1991' 'Dec-1987'
 'Jul-1996' 'Jul-1988' 'Jul-1997' 'Apr-2004' 'Dec-2005' 'Mar-2003'
 'Feb-1999' 'May-1984' 'Jun-2000' 'Dec-1996' 'Apr-2010' 'Jan-2004'
 'May-1999' 'Sep-1972' 'Jul-1981' 'Sep-1993' 'Feb-2009' 'Jul-2000'
 'Nov-2002' 'Apr-1992' 'Jan-1998' 'Jun-1999' 'Jan-1993' 'May-2005'
 'Sep-1982' 'Apr-1990' 'Dec-1998' 'Feb-1996' 'Mar-1993' 'Apr-1978'
 'Feb-2000' 'Jul-1995' 'May-1995' 'Apr-1991' 'Aug-1991' 'Jul-2002'
 'Oct-1989' 'Apr-1984' 'Dec-2009' 'Sep-2000' 'Jan-1982' 'Mar-1998'
 'Jun-1998' 'Jan-1996' 'Nov-1987' 'May-2010' 'Jun-1987' 'Jan-1989'
 'Feb-2004' 'Dec-1989' 'Oct-1992' 'Feb-2005' 'Apr-1993' 'Dec-1985'
 'Sep-1979' 'Feb-2007' 'Aug-1998' 'Nov-1989' 'Apr-2005' 'Mar-1978'
 'Sep-1985' 'Nov-1994' 'Nov-1990' 'Jun-2008' 'Apr-1987' 'Aug-1994'
 'Dec-2007' 'Mar-1991' 'Jul-1990' 'Jan-1988' 'Feb-2006' 'Feb-1985'
 'Sep-1989' 'Aug-2009' 'Nov-2008' 'Nov-1981' 'Jan-2008' 'Nov-1995'
 'Aug-1987' 'Nov-1985' 'Dec-1965' 'Sep-1995' 'Jan-1986' 'Oct-2009'
 'May-2002' 'Aug-1980' 'Mar-1995' 'Sep-1977' 'Sep-1988' 'Oct-1987'
 'Oct-1984' 'May-1988' 'Aug-1984' 'Nov-1988' 'May-1974' 'Jan-1997'
```

'Nov-1982' 'Oct-1991' 'Feb-1984' 'Jan-1981' 'Jul-1989' 'Dec-1976'
'Dec-1994' 'Dec-1980' 'Sep-1984' 'Jun-2007' 'Sep-2008' 'Apr-1983'
'Mar-2006' 'Jul-1984' 'Jan-1985' 'Dec-1995' 'Apr-2008' 'Aug-1995'
'Mar-2008' 'Jan-1983' 'Oct-1995' 'Sep-1991' 'Feb-1995' 'Jun-1979'
'Aug-1990' 'Jul-1986' 'Nov-1977' 'Dec-1982' 'May-1979' 'Aug-1992'
'Feb-1983' 'Aug-1982' 'Oct-1980' 'Mar-1979' 'Jan-1978' 'Jun-2011'
'Mar-1984' 'Nov-2000' 'May-1983' 'Jul-2008' 'Dec-1986' 'Apr-1982'
'Jul-1983' 'Feb-1990' 'Dec-2008' 'Jul-1975' 'Dec-1971' 'Feb-2008'
'Mar-2011' 'Feb-1987' 'Feb-1989' 'Aug-1985' 'Jul-2010' 'Apr-1989'
'Feb-1980' 'Feb-1988' 'May-2006' 'Nov-2010' 'Apr-2009' 'Feb-2010'
'May-1976' 'Feb-1981' 'Nov-1984' 'Jun-1988' 'May-1992' 'May-1972'
'Apr-2013' 'Oct-1988' 'Sep-1990' 'Jun-1992' 'Oct-1982' 'Feb-2013'
'Mar-1992' 'Jun-1985' 'Mar-1988' 'Aug-1981' 'Feb-2011' 'Nov-1974'
'Feb-1978' 'Jul-2011' 'Aug-1983' 'Apr-1985' 'Jul-2009' 'Aug-1979'
'Nov-1983' 'Sep-1983' 'Jul-1987' 'Aug-2010' 'Oct-1976' 'Oct-1985'
'Jan-1991' 'Dec-1991' 'May-2009' 'Aug-2011' 'Jan-1974' 'May-1981'
'Jun-1972' 'Feb-1991' 'Jun-1978' 'Sep-1986' 'Jan-2012' 'Jan-1980'
'Sep-1980' 'Dec-1975' 'Jan-1984' 'Nov-1980' 'May-1987' 'Sep-1970'
'Jan-1976' 'Feb-1986' 'Oct-2010' 'Apr-1979' 'Jan-1979' 'Sep-2011'
'Jul-1979' 'Sep-1975' 'Mar-1981' 'Apr-1980' 'Apr-1977' 'Nov-1970'
'Nov-2011' 'Mar-2012' 'Sep-1981' 'Mar-1977' 'Dec-1977' 'May-2012'
'Dec-1979' 'Oct-1983' 'Jan-2009' 'Jun-2009' 'Dec-2011' 'Oct-1975'
'Oct-1979' 'Mar-1969' 'Aug-1978' 'Jul-1980' 'Oct-1977' 'Jun-1969'
'Oct-1963' 'Nov-1960' 'Jan-1987' 'Feb-1979' 'Sep-1974' 'May-1966'
'Apr-1972' 'Apr-1973' 'May-1975' 'Sep-1966' 'Dec-1983' 'Aug-1986'
'Nov-1979' 'May-1980' 'Feb-1982' 'Feb-1969' 'Feb-2012' 'Aug-1973'
'Feb-1972' 'Apr-1975' 'Jul-1978' 'May-1985' 'Sep-1976' 'Apr-2011'
'Nov-2012' 'Jun-1984' 'Mar-1975' 'Apr-1981' 'Mar-2009' 'Jun-1977'
'Mar-1980' 'Oct-1973' 'Apr-2012' 'Apr-1971' 'Sep-1969' 'Oct-1978'
'Feb-1977' 'Jun-2012' 'Apr-1976' 'Feb-1965' 'Jul-1977' 'Jun-1976'
'Oct-1972' 'Aug-1977' 'Dec-1978' 'Jun-1975' 'Nov-1971' 'May-1964'
'Feb-1975' 'May-1982' 'Apr-1970' 'Apr-1969' 'May-1977' 'Dec-1981'
'Nov-1973' 'Feb-1976' 'Mar-1970' 'Feb-1971' 'Aug-1968' 'Aug-1976'
'Jun-1963' 'Nov-1976' 'Jul-1974' 'Jun-2013' 'Aug-2012' 'Dec-1969'
'Jul-1970' 'Feb-1973' 'Mar-1974' 'Feb-1974' 'Sep-1964' 'Jul-1965'
'Jun-1973' 'Nov-1975' 'Jul-1963' 'Jun-1974' 'Mar-1972' 'Nov-1978'
'Jan-1964' 'Mar-1971' 'Mar-1976' 'May-1958' 'Sep-1973' 'Jul-1973'
'Dec-1972' 'Jan-1977' 'Aug-1965' 'Jan-1975' 'Dec-1974' 'Jul-1976'
'Oct-2012' 'May-1973' 'Jul-1972' 'Sep-1978' 'Feb-1968' 'Nov-1968'
'Mar-2013' 'Jan-2013' 'Oct-1965' 'Jan-1966' 'Aug-1972' 'Jun-1980'
'Jul-1969' 'May-1965' 'Oct-1969' 'Aug-1974' 'May-1968' 'Aug-1969'
'May-2013' 'Jan-1973' 'Oct-1974' 'Jul-1967' 'Oct-1967' 'Apr-1974'
'Jan-1963' 'Apr-1968' 'Jul-1971' 'Aug-1970' 'May-1970' 'Dec-1973'
'Jan-1969' 'Nov-1972' 'Oct-1959' 'Apr-1967' 'Sep-1967' 'Jan-1968'
'Jul-2012' 'Nov-1963' 'Mar-1973' 'Oct-1971' 'Dec-1968' 'Jan-1960'
'Sep-2013' 'Sep-1971' 'May-1969' 'Dec-1966' 'Oct-1970' 'Nov-1967'
'Jan-1972' 'Dec-1967' 'Sep-1968' 'Oct-1964' 'Aug-1966' 'Jul-1966'
'Nov-1969' 'Apr-1964' 'May-1971' 'Jul-2013' 'Apr-1966' 'Jun-1967'
'Jan-1962' 'Aug-1975' 'Feb-1970' 'Dec-1959' 'Jan-1971' 'Dec-2012'

```
 'Jan-1970' 'Dec-1963' 'Jan-1944' 'Jun-1965' 'May-1962' 'Jun-1966'
 'Mar-1968' 'Jan-1967' 'Aug-2013' 'Oct-1968' 'Jun-1970' 'Jun-1968'
 'Oct-1957' 'Dec-1958' 'Mar-1967' 'Feb-1963' 'Jun-1971' 'Feb-1967'
 'Dec-1960' 'May-1955' 'Dec-1970' 'Nov-1950' 'Dec-1962' 'Aug-1971'
 'Jun-1957' 'Dec-1964' 'Nov-1966' 'Nov-1953' 'Jan-1965' 'Jan-1961'
 'Sep-1963' 'Oct-1960' 'Feb-1964' 'Mar-1966' 'Jul-1959' 'Jul-1968'
 'Mar-1963' 'Mar-1962' 'Nov-1965' 'Jul-1960' 'May-1967' 'Oct-1962'
 'Jul-1958' 'Nov-1954' 'Nov-1957' 'May-1963' 'Jul-1955' 'Oct-1950'
 'Dec-1961' 'Oct-2013' 'Feb-1966' 'Jun-1964' 'Apr-1962' 'Nov-1964'
 'Jun-1962' 'Sep-1959' 'Jul-1962' 'Nov-1958' 'Jul-1951' 'Jun-1960'
 'Jan-1959' 'Apr-1958' 'Mar-1960' 'Sep-1957' 'Sep-1960' 'May-1959'
 'Oct-1966' 'Jun-1959' 'Sep-1965' 'Jul-1964' 'Jan-1956' 'Feb-1961'
 'Apr-1965' 'Jan-1958' 'Aug-1963' 'Oct-1961' 'Aug-1962']

open_acc -->>  [16. 17. 13.  6.  8. 11.  5.  9. 15. 12. 10. 18.  7.
4. 14. 20. 19. 21.
 23.  3. 22. 25. 26.  2. 24. 27.  1.]

pub_rec -->>  [0 1]

revol_bal -->>  [36369. 20131. 11987. ... 29244. 31702. 28053.]

revol_util -->>  [ 41.8  53.3  92.2 ... 114.4 107.2 111.4]

total_acc -->>  [25. 27. 26. 13. 43. 23. 15. 40. 37. 35. 22. 20. 36.
38. 18. 17. 29. 16.
 21. 34.  9. 14. 59. 41. 19. 12. 30. 10. 56. 24. 28.  8. 52. 31. 44.
39.
 50. 11. 32.  5. 33. 46. 42.  6.  7. 49. 45. 57. 51. 55. 53.  4. 47.
48.
 58. 54.  3.  2.]

initial_list_status -->>  ['w' 'f']

application_type -->>  ['INDIVIDUAL' 'JOINT' 'DIRECT_PAY']

mort_acc -->>  [0 1]

pub_rec_bankruptcies -->>  [0 1]

address -->>  ['0174 Michelle Gateway\r\nMendozaberg, OK 22690'
 '1076 Carney Fort Apt. 347\r\nLoganmouth, SD 05113'
 '87025 Mark Dale Apt. 269\r\nNew Sabrina, WV 05113' ...
 '0114 Fowler Field Suite 028\r\nRachelborough, LA 05113'
 '953 Matthew Points Suite 414\r\nReedfort, NY 70466'
 '7843 Blake Freeway Apt. 229\r\nNew Michael, FL 29597']


# Converting term values to numerical val
term_values={' 36 months': 36, ' 60 months':60}
df['term'] = df.term.map(term_values)
```

```
# Mapping the target variable
df['loan_status'] = df.loan_status.map({'Fully Paid':0, 'Charged
Off':1})

# Initial List Status
df['initial_list_status'].unique()
np.array(['w', 'f'], dtype=object)
list_status = {'w': 0, 'f': 1}
df['initial_list_status'] = df.initial_list_status.map(list_status)

# Let's fetch ZIP from address and then drop the remaining details -
df['zip_code'] = df.address.apply(lambda x: x[-5:])
df['zip_code'].value_counts(normalize=True)*100

for i in df.columns:
    print(i, '-->> ', df[i].unique(), '\n')
```

```
loan_amnt -->>  [10000.  8000. 15600. ... 36275. 34175. 36475.]

term -->>  [36 60]

int_rate -->>  [11.44 11.99 10.49  6.49 17.27 13.33  5.32 11.14 10.99
16.29 13.11  9.17
  6.62  8.39 21.98 12.29  7.9   6.97 15.61 13.35 12.12  9.99  8.19
18.75
 14.99 13.67 13.98 17.86 21.49 12.99 18.54 17.1  18.25 11.67  6.24
8.18
 12.35 14.16 18.55 22.15 15.99 24.99  9.67 19.19 21.   12.69 10.74
6.68
 19.22 16.99 16.55 19.97  7.89 24.7  16.49 25.78 25.83 13.99 15.22
15.31
  7.69 19.53 10.16  7.62  9.75 13.68 15.88 14.65 23.83 10.75 18.49
20.31
 17.57 27.31 22.99 14.33 13.53 22.45 14.64 24.5  17.99  9.16 12.49
11.55
 17.76 20.49 22.7  17.56  6.03  6.89 19.52  8.9  14.3   9.49 25.99
24.08
 13.05 14.98 16.59 10.15 25.89 21.99 23.99 11.49 14.47 11.53  8.67
13.49
  8.59 10.64 25.44  9.71 16.2  19.24 24.11 15.8  14.49 18.99 23.28
19.99
 18.24 14.09  9.25 19.05 17.77 18.92 13.65  7.12 16.78 14.46 12.59
16.24
 25.49  7.39 10.78 12.85 12.39 21.18 21.97  6.39  7.49 25.57 20.99
11.47
  7.26 14.31 24.24 23.43 19.47 23.32  9.76  6.99 21.15 21.48  6.92
22.95
 18.85 15.59 15.1  19.72 11.22 15.81 11.48 13.66 23.76  9.8  17.14
18.84
```

```
  8.49 25.29 22.47 19.2  11.39 22.4  12.79 18.2  13.18 25.28 12.05
20.2
 21.67 20.8  13.44 25.8  14.85  6.    7.99 23.1   7.91 21.7  26.06
5.93
 15.77  8.99 20.5  23.63 14.48 22.2  15.41 23.5  22.9  19.48 23.4
23.13
 12.88 24.49 21.6   8.38 17.97  8.6  12.74 25.88 24.89 26.77 19.89
20.75
 24.83  7.59 23.7  23.33 26.24 24.2  26.57 25.11  7.24 22.78 22.39
26.14
 23.26 22.35 26.49 25.69 25.65 24.74 22.74 26.99  8.24 25.09 24.76
24.52
 24.33]

grade -->>  ['B' 'A' 'C' 'E' 'D' 'F' 'G']

sub_grade -->>  ['B4' 'B5' 'B3' 'A2' 'C5' 'C3' 'A1' 'B2' 'A5' 'E4'
'C1' 'A4' 'A3' 'D1'
 'C2' 'B1' 'D3' 'D5' 'D2' 'E1' 'E2' 'E5' 'F4' 'E3' 'D4' 'G1' 'F5' 'G2'
 'C4' 'F1' 'F3' 'G4' 'F2' 'G3' 'G5']

emp_title -->>  ['Marketing' 'Credit analyst ' 'Statistician' ...
 "Michael's Arts & Crafts" 'licensed bankere' 'Gracon Services, Inc']

emp_length -->>  ['10+ years' '4 years' '< 1 year' '6 years' '9 years'
'2 years' '3 years'
 '7 years' '8 years' '5 years' '1 year']

home_ownership -->>  ['RENT' 'MORTGAGE' 'OWN' 'OTHER' 'ANY' 'NONE']

annual_inc -->>  [117000.  65000.  43057. ...  40311.  36111.  47212.]


verification_status -->>  ['Not Verified' 'Source Verified'
'Verified']

issue_d -->>  ['Jan-2015' 'Nov-2014' 'Apr-2013' 'Sep-2015' 'Sep-2012'
'Oct-2014'
 'Apr-2012' 'Jun-2013' 'Dec-2015' 'Oct-2012' 'Jul-2014' 'Feb-2013'
 'Oct-2015' 'Jan-2014' 'Mar-2016' 'Apr-2014' 'Jun-2014' 'Oct-2013'
 'May-2013' 'Feb-2015' 'Jun-2015' 'Mar-2013' 'Jun-2016' 'Mar-2014'
 'Nov-2013' 'Dec-2014' 'Sep-2013' 'May-2016' 'Jul-2015' 'Apr-2015'
 'Jul-2013' 'Aug-2013' 'Aug-2014' 'Dec-2013' 'Mar-2012' 'Mar-2015'
 'Jul-2012' 'Feb-2014' 'Dec-2012' 'Sep-2014' 'Nov-2012' 'Apr-2016'
 'May-2012' 'May-2014' 'Jun-2012' 'Aug-2012' 'May-2015' 'Oct-2016'
 'Aug-2015' 'Jul-2016' 'Aug-2016' 'Feb-2016' 'Jan-2013' 'Nov-2015'
 'Jan-2016' 'Nov-2016' 'Dec-2016' 'Sep-2016']

loan_status -->>  [0 1]
```

```
purpose -->>  ['vacation' 'debt_consolidation' 'credit_card'
'home_improvement'
 'small_business' 'major_purchase' 'other' 'medical' 'car' 'moving'
 'house' 'wedding' 'renewable_energy']

title -->>  ['Vacation' 'Debt consolidation' 'Credit card refinancing'
... 'Cc Debt'
 'Credit buster ' 'Loanforpayoff']

dti -->>  [26.24 22.05 12.79 ... 41.68 40.21 40.57]

earliest_cr_line -->>  ['Jun-1990' 'Jul-2004' 'Aug-2007' 'Sep-2006'
'Mar-1999' 'Jan-2005'
 'Aug-2005' 'Sep-1994' 'Jun-1994' 'Dec-1997' 'Dec-1990' 'Apr-1995'
 'May-2001' 'Mar-1982' 'Sep-1996' 'Jan-1990' 'Mar-2000' 'Jan-2006'
 'Jan-2003' 'May-2008' 'Jun-2004' 'Jan-1999' 'Apr-1994' 'Apr-1998'
 'Jul-2007' 'Oct-2007' 'May-1997' 'Jul-2006' 'Sep-2003' 'Feb-2002'
 'Jan-1992' 'Aug-2001' 'Dec-2010' 'Oct-1999' 'Sep-2004' 'Jul-2003'
 'Apr-2000' 'Dec-2004' 'Jun-1995' 'Dec-2003' 'Jul-1994' 'Oct-1990'
 'Apr-1999' 'Dec-2001' 'May-2003' 'Oct-2002' 'Mar-2004' 'Aug-2003'
 'Oct-2000' 'Nov-2004' 'Mar-2010' 'Mar-1996' 'May-1994' 'Jun-1996'
 'Nov-1986' 'Jan-2002' 'Mar-2001' 'Sep-2012' 'Apr-2006' 'May-1998'
 'Dec-2002' 'Oct-2005' 'May-1990' 'Jun-2003' 'Jun-2001' 'Feb-2001'
 'Jun-2006' 'Oct-2006' 'Aug-1993' 'Apr-2001' 'Nov-2001' 'Feb-2003'
 'Jun-1993' 'Sep-1992' 'Nov-1992' 'Jun-1983' 'Jul-1999' 'Sep-1997'
 'Nov-1993' 'Feb-1993' 'Apr-2007' 'Nov-1999' 'Nov-2005' 'Dec-1992'
 'Mar-1986' 'Dec-1988' 'May-1989' 'Dec-2000' 'Mar-2005' 'Jun-2010'
 'Jan-2001' 'Sep-2001' 'Jan-1994' 'Aug-2002' 'Nov-2003' 'Jan-2011'
 'Aug-2008' 'Jun-2005' 'Nov-1997' 'May-1993' 'Sep-2005' 'Apr-1986'
 'Aug-1996' 'Aug-1997' 'Jul-2005' 'May-2011' 'Sep-2002' 'Aug-1999'
 'May-1996' 'Feb-1992' 'Sep-1999' 'Jul-2001' 'Oct-2001' 'Oct-2008'
 'Nov-2007' 'Apr-1997' 'Jun-1986' 'Sep-1998' 'Jun-1982' 'Oct-1981'
 'Feb-1994' 'Dec-1984' 'Nov-1991' 'Nov-2006' 'Aug-2000' 'Oct-2004'
 'Apr-1988' 'May-2004' 'Aug-1988' 'Mar-1994' 'Aug-2004' 'Dec-2006'
 'Nov-1998' 'Oct-1997' 'Mar-1989' 'Jul-1982' 'Mar-1997' 'Oct-1994'
 'Jul-1998' 'Jun-2002' 'May-1991' 'Oct-2011' 'Sep-2007' 'Jan-2007'
 'Jan-2010' 'Mar-1987' 'Feb-1997' 'Oct-1986' 'Mar-2002' 'Jul-1993'
 'Mar-2007' 'Aug-1989' 'May-2007' 'Dec-1993' 'Jun-1989' 'Jun-1997'
 'Oct-2003' 'Apr-1996' 'Oct-1998' 'Mar-1983' 'Mar-1985' 'Oct-1993'
 'Apr-2002' 'Apr-2003' 'Jul-1985' 'May-1978' 'Sep-2010' 'Oct-1996'
 'Sep-2009' 'Jan-2000' 'Sep-1987' 'Jan-1995' 'May-2000' 'Jun-1981'
 'Nov-1996' 'Feb-1998' 'Aug-1967' 'Dec-1999' 'Aug-2006' 'Nov-2009'
 'Jul-1992' 'Jul-1991' 'Mar-1990' 'May-1986' 'Jun-1991' 'Dec-1987'
 'Jul-1996' 'Jul-1988' 'Jul-1997' 'Apr-2004' 'Dec-2005' 'Mar-2003'
 'Feb-1999' 'May-1984' 'Jun-2000' 'Dec-1996' 'Apr-2010' 'Jan-2004'
 'May-1999' 'Sep-1972' 'Jul-1981' 'Sep-1993' 'Feb-2009' 'Jul-2000'
 'Nov-2002' 'Apr-1992' 'Jan-1998' 'Jun-1999' 'Jan-1993' 'May-2005'
 'Sep-1982' 'Apr-1990' 'Dec-1998' 'Feb-1996' 'Mar-1993' 'Apr-1978'
 'Feb-2000' 'Jul-1995' 'May-1995' 'Apr-1991' 'Aug-1991' 'Jul-2002'
 'Oct-1989' 'Apr-1984' 'Dec-2009' 'Sep-2000' 'Jan-1982' 'Mar-1998'
```

'Jun-1998' 'Jan-1996' 'Nov-1987' 'May-2010' 'Jun-1987' 'Jan-1989'
'Feb-2004' 'Dec-1989' 'Oct-1992' 'Feb-2005' 'Apr-1993' 'Dec-1985'
'Sep-1979' 'Feb-2007' 'Aug-1998' 'Nov-1989' 'Apr-2005' 'Mar-1978'
'Sep-1985' 'Nov-1994' 'Nov-1990' 'Jun-2008' 'Apr-1987' 'Aug-1994'
'Dec-2007' 'Mar-1991' 'Jul-1990' 'Jan-1988' 'Feb-2006' 'Feb-1985'
'Sep-1989' 'Aug-2009' 'Nov-2008' 'Nov-1981' 'Jan-2008' 'Nov-1995'
'Aug-1987' 'Nov-1985' 'Dec-1965' 'Sep-1995' 'Jan-1986' 'Oct-2009'
'May-2002' 'Aug-1980' 'Mar-1995' 'Sep-1977' 'Sep-1988' 'Oct-1987'
'Oct-1984' 'May-1988' 'Aug-1984' 'Nov-1988' 'May-1974' 'Jan-1997'
'Nov-1982' 'Oct-1991' 'Feb-1984' 'Jan-1981' 'Jul-1989' 'Dec-1976'
'Dec-1994' 'Dec-1980' 'Sep-1984' 'Jun-2007' 'Sep-2008' 'Apr-1983'
'Mar-2006' 'Jul-1984' 'Jan-1985' 'Dec-1995' 'Apr-2008' 'Aug-1995'
'Mar-2008' 'Jan-1983' 'Oct-1995' 'Sep-1991' 'Feb-1995' 'Jun-1979'
'Aug-1990' 'Jul-1986' 'Nov-1977' 'Dec-1982' 'May-1979' 'Aug-1992'
'Feb-1983' 'Aug-1982' 'Oct-1980' 'Mar-1979' 'Jan-1978' 'Jun-2011'
'Mar-1984' 'Nov-2000' 'May-1983' 'Jul-2008' 'Dec-1986' 'Apr-1982'
'Jul-1983' 'Feb-1990' 'Dec-2008' 'Jul-1975' 'Dec-1971' 'Feb-2008'
'Mar-2011' 'Feb-1987' 'Feb-1989' 'Aug-1985' 'Jul-2010' 'Apr-1989'
'Feb-1980' 'Feb-1988' 'May-2006' 'Nov-2010' 'Apr-2009' 'Feb-2010'
'May-1976' 'Feb-1981' 'Nov-1984' 'Jun-1988' 'May-1992' 'May-1972'
'Apr-2013' 'Oct-1988' 'Sep-1990' 'Jun-1992' 'Oct-1982' 'Feb-2013'
'Mar-1992' 'Jun-1985' 'Mar-1988' 'Aug-1981' 'Feb-2011' 'Nov-1974'
'Feb-1978' 'Jul-2011' 'Aug-1983' 'Apr-1985' 'Jul-2009' 'Aug-1979'
'Nov-1983' 'Sep-1983' 'Jul-1987' 'Aug-2010' 'Oct-1976' 'Oct-1985'
'Jan-1991' 'Dec-1991' 'May-2009' 'Aug-2011' 'Jan-1974' 'May-1981'
'Jun-1972' 'Feb-1991' 'Jun-1978' 'Sep-1986' 'Jan-2012' 'Jan-1980'
'Sep-1980' 'Dec-1975' 'Jan-1984' 'Nov-1980' 'May-1987' 'Sep-1970'
'Jan-1976' 'Feb-1986' 'Oct-2010' 'Apr-1979' 'Jan-1979' 'Sep-2011'
'Jul-1979' 'Sep-1975' 'Mar-1981' 'Apr-1980' 'Apr-1977' 'Nov-1970'
'Nov-2011' 'Mar-2012' 'Sep-1981' 'Mar-1977' 'Dec-1977' 'May-2012'
'Dec-1979' 'Oct-1983' 'Jan-2009' 'Jun-2009' 'Dec-2011' 'Oct-1975'
'Oct-1979' 'Mar-1969' 'Aug-1978' 'Jul-1980' 'Oct-1977' 'Jun-1969'
'Oct-1963' 'Nov-1960' 'Jan-1987' 'Feb-1979' 'Sep-1974' 'May-1966'
'Apr-1972' 'Apr-1973' 'May-1975' 'Sep-1966' 'Dec-1983' 'Aug-1986'
'Nov-1979' 'May-1980' 'Feb-1982' 'Feb-1969' 'Feb-2012' 'Aug-1973'
'Feb-1972' 'Apr-1975' 'Jul-1978' 'May-1985' 'Sep-1976' 'Apr-2011'
'Nov-2012' 'Jun-1984' 'Mar-1975' 'Apr-1981' 'Mar-2009' 'Jun-1977'
'Mar-1980' 'Oct-1973' 'Apr-2012' 'Apr-1971' 'Sep-1969' 'Oct-1978'
'Feb-1977' 'Jun-2012' 'Apr-1976' 'Feb-1965' 'Jul-1977' 'Jun-1976'
'Oct-1972' 'Aug-1977' 'Dec-1978' 'Jun-1975' 'Nov-1971' 'May-1964'
'Feb-1975' 'May-1982' 'Apr-1970' 'Apr-1969' 'May-1977' 'Dec-1981'
'Nov-1973' 'Feb-1976' 'Mar-1970' 'Feb-1971' 'Aug-1968' 'Aug-1976'
'Jun-1963' 'Nov-1976' 'Jul-1974' 'Jun-2013' 'Aug-2012' 'Dec-1969'
'Jul-1970' 'Feb-1973' 'Mar-1974' 'Feb-1974' 'Sep-1964' 'Jul-1965'
'Jun-1973' 'Nov-1975' 'Jul-1963' 'Jun-1974' 'Mar-1972' 'Nov-1978'
'Jan-1964' 'Mar-1971' 'Mar-1976' 'May-1958' 'Sep-1973' 'Jul-1973'
'Dec-1972' 'Jan-1977' 'Aug-1965' 'Jan-1975' 'Dec-1974' 'Jul-1976'
'Oct-2012' 'May-1973' 'Jul-1972' 'Sep-1978' 'Feb-1968' 'Nov-1968'
'Mar-2013' 'Jan-2013' 'Oct-1965' 'Jan-1966' 'Aug-1972' 'Jun-1980'

```
 'Jul-1969' 'May-1965' 'Oct-1969' 'Aug-1974' 'May-1968' 'Aug-1969'
 'May-2013' 'Jan-1973' 'Oct-1974' 'Jul-1967' 'Oct-1967' 'Apr-1974'
 'Jan-1963' 'Apr-1968' 'Jul-1971' 'Aug-1970' 'May-1970' 'Dec-1973'
 'Jan-1969' 'Nov-1972' 'Oct-1959' 'Apr-1967' 'Sep-1967' 'Jan-1968'
 'Jul-2012' 'Nov-1963' 'Mar-1973' 'Oct-1971' 'Dec-1968' 'Jan-1960'
 'Sep-2013' 'Sep-1971' 'May-1969' 'Dec-1966' 'Oct-1970' 'Nov-1967'
 'Jan-1972' 'Dec-1967' 'Sep-1968' 'Oct-1964' 'Aug-1966' 'Jul-1966'
 'Nov-1969' 'Apr-1964' 'May-1971' 'Jul-2013' 'Apr-1966' 'Jun-1967'
 'Jan-1962' 'Aug-1975' 'Feb-1970' 'Dec-1959' 'Jan-1971' 'Dec-2012'
 'Jan-1970' 'Dec-1963' 'Jan-1944' 'Jun-1965' 'May-1962' 'Jun-1966'
 'Mar-1968' 'Jan-1967' 'Aug-2013' 'Oct-1968' 'Jun-1970' 'Jun-1968'
 'Oct-1957' 'Dec-1958' 'Mar-1967' 'Feb-1963' 'Jun-1971' 'Feb-1967'
 'Dec-1960' 'May-1955' 'Dec-1970' 'Nov-1950' 'Dec-1962' 'Aug-1971'
 'Jun-1957' 'Dec-1964' 'Nov-1966' 'Nov-1953' 'Jan-1965' 'Jan-1961'
 'Sep-1963' 'Oct-1960' 'Feb-1964' 'Mar-1966' 'Jul-1959' 'Jul-1968'
 'Mar-1963' 'Mar-1962' 'Nov-1965' 'Jul-1960' 'May-1967' 'Oct-1962'
 'Jul-1958' 'Nov-1954' 'Nov-1957' 'May-1963' 'Jul-1955' 'Oct-1950'
 'Dec-1961' 'Oct-2013' 'Feb-1966' 'Jun-1964' 'Apr-1962' 'Nov-1964'
 'Jun-1962' 'Sep-1959' 'Jul-1962' 'Nov-1958' 'Jul-1951' 'Jun-1960'
 'Jan-1959' 'Apr-1958' 'Mar-1960' 'Sep-1957' 'Sep-1960' 'May-1959'
 'Oct-1966' 'Jun-1959' 'Sep-1965' 'Jul-1964' 'Jan-1956' 'Feb-1961'
 'Apr-1965' 'Jan-1958' 'Aug-1963' 'Oct-1961' 'Aug-1962']

open_acc -->>  [16. 17. 13.  6.  8. 11.  5.  9. 15. 12. 10. 18.  7.
4. 14. 20. 19. 21.
 23.  3. 22. 25. 26.  2. 24. 27.  1.]

pub_rec -->>  [0 1]

revol_bal -->>  [36369. 20131. 11987. ... 29244. 31702. 28053.]

revol_util -->>  [ 41.8  53.3  92.2 ... 114.4 107.2 111.4]

total_acc -->>  [25. 27. 26. 13. 43. 23. 15. 40. 37. 35. 22. 20. 36.
38. 18. 17. 29. 16.
 21. 34.  9. 14. 59. 41. 19. 12. 30. 10. 56. 24. 28.  8. 52. 31. 44.
39.
 50. 11. 32.  5. 33. 46. 42.  6.  7. 49. 45. 57. 51. 55. 53.  4. 47.
48.
 58. 54.  3.  2.]

initial_list_status -->>  [0 1]

application_type -->>  ['INDIVIDUAL' 'JOINT' 'DIRECT_PAY']

mort_acc -->>  [0 1]

pub_rec_bankruptcies -->>  [0 1]

address -->>  ['0174 Michelle Gateway\r\nMendozaberg, OK 22690'
```

```
 '1076 Carney Fort Apt. 347\r\nLoganmouth, SD 05113'
 '87025 Mark Dale Apt. 269\r\nNew Sabrina, WV 05113' ...
 '0114 Fowler Field Suite 028\r\nRachelborough, LA 05113'
 '953 Matthew Points Suite 414\r\nReedfort, NY 70466'
 '7843 Blake Freeway Apt. 229\r\nNew Michael, FL 29597']

zip_code -->>  ['22690' '05113' '00813' '11650' '30723' '70466'
'29597' '48052' '86630'
 '93700']


df.head()

   loan_amnt  term  int_rate grade sub_grade                emp_title
\
0    10000.0    36     11.44     B        B4                Marketing

1     8000.0    36     11.99     B        B5          Credit analyst

2    15600.0    36     10.49     B        B3             Statistician

3     7200.0    36      6.49     A        A2          Client Advocate

4    24375.0    60     17.27     C        C5  Destiny Management Inc.


  emp_length home_ownership  annual_inc verification_status  ...
pub_rec  \
0  10+ years           RENT    117000.0        Not Verified  ...
0
1    4 years       MORTGAGE     65000.0        Not Verified  ...
0
2   < 1 year           RENT     43057.0     Source Verified  ...
0
3    6 years           RENT     54000.0        Not Verified  ...
0
4    9 years       MORTGAGE     55000.0            Verified  ...
0

   revol_bal revol_util total_acc  initial_list_status
application_type  \
0    36369.0       41.8      25.0                    0
INDIVIDUAL
1    20131.0       53.3      27.0                    1
INDIVIDUAL
2    11987.0       92.2      26.0                    1
INDIVIDUAL
3     5472.0       21.5      13.0                    1
INDIVIDUAL
4    24584.0       69.8      43.0                    1
INDIVIDUAL
```

```
   mort_acc  pub_rec_bankruptcies  \
0         0                     0
1         1                     0
2         0                     0
3         0                     0
4         1                     0

                                                address  zip_code
0     0174 Michelle Gateway\r\nMendozaberg, OK 22690     22690
1  1076 Carney Fort Apt. 347\r\nLoganmouth, SD 05113     05113
2  87025 Mark Dale Apt. 269\r\nNew Sabrina, WV 05113     05113
3            823 Reid Ford\r\nDelacruzside, MA 00813     00813
4             679 Luna Roads\r\nGreggshire, VA 11650     11650

[5 rows x 27 columns]
```

```
df['issue_d'].head()
```

```
0    Jan-2015
1    Jan-2015
2    Jan-2015
3    Nov-2014
4    Apr-2013
Name: issue_d, dtype: object
```

```
df['issue_month'] = df['issue_d'].apply(lambda x : str(x).split('-')
[0])
df['issue_year'] = df['issue_d'].apply(lambda x : str(x).split('-')
[1])

df = df.drop(columns=['issue_d'], axis=1)

df[['earliest_cr_line']].head()
```

```
  earliest_cr_line
0         Jun-1990
1         Jul-2004
2         Aug-2007
3         Sep-2006
4         Mar-1999
```

```
df['earliest_cr_line_month'] = df['earliest_cr_line'].apply(lambda x :
str(x).split('-')[0])
df['earliest_cr_line_year'] = df['earliest_cr_line'].apply(lambda x :
str(x).split('-')[1])

df = df.drop(columns=['earliest_cr_line'], axis=1)

df.columns
```

```
Index(['loan_amnt', 'term', 'int_rate', 'grade', 'sub_grade',
'emp_title',
       'emp_length', 'home_ownership', 'annual_inc',
'verification_status',
       'loan_status', 'purpose', 'title', 'dti', 'open_acc',
'pub_rec',
       'revol_bal', 'revol_util', 'total_acc', 'initial_list_status',
       'application_type', 'mort_acc', 'pub_rec_bankruptcies',
'address',
       'zip_code', 'issue_month', 'issue_year',
'earliest_cr_line_month',
       'earliest_cr_line_year'],
      dtype='object')
```

```
df = df.drop(columns=['address','zip_code','title',], axis=1)
```

```
df.columns
```

```
Index(['loan_amnt', 'term', 'int_rate', 'grade', 'sub_grade',
'emp_title',
       'emp_length', 'home_ownership', 'annual_inc',
'verification_status',
       'loan_status', 'purpose', 'dti', 'open_acc', 'pub_rec',
'revol_bal',
       'revol_util', 'total_acc', 'initial_list_status',
'application_type',
       'mort_acc', 'pub_rec_bankruptcies', 'issue_month',
'issue_year',
       'earliest_cr_line_month', 'earliest_cr_line_year'],
      dtype='object')
```

```
for col in df.columns:
  print(col, '->', df[col].nunique())
```

```
loan_amnt -> 1383
term -> 2
int_rate -> 241
grade -> 7
sub_grade -> 35
emp_title -> 143140
emp_length -> 11
home_ownership -> 6
annual_inc -> 19887
verification_status -> 3
loan_status -> 2
purpose -> 13
dti -> 4045
open_acc -> 27
pub_rec -> 2
revol_bal -> 47298
```

```
revol_util -> 1135
total_acc -> 58
initial_list_status -> 2
application_type -> 3
mort_acc -> 2
pub_rec_bankruptcies -> 2
issue_month -> 12
issue_year -> 5
earliest_cr_line_month -> 12
earliest_cr_line_year -> 64
```

```
label_encoder = LabelEncoder()
```

```
df['term'] = label_encoder.fit_transform(df['term'])
df['grade'] = label_encoder.fit_transform(df['grade'])
df['sub_grade'] = label_encoder.fit_transform(df['sub_grade'])
df['emp_length'] = label_encoder.fit_transform(df['emp_length'])
df['home_ownership'] =
label_encoder.fit_transform(df['home_ownership'])
df['verification_status'] =
label_encoder.fit_transform(df['verification_status'])
df['loan_status'] = label_encoder.fit_transform(df['loan_status'])
df['purpose'] = label_encoder.fit_transform(df['purpose'])
df['pub_rec'] = label_encoder.fit_transform(df['pub_rec'])
df['initial_list_status'] =
label_encoder.fit_transform(df['initial_list_status'])
df['application_type'] =
label_encoder.fit_transform(df['application_type'])
df['mort_acc'] = label_encoder.fit_transform(df['mort_acc'])
df['pub_rec_bankruptcies'] =
label_encoder.fit_transform(df['pub_rec_bankruptcies'])
df['open_acc'] = label_encoder.fit_transform(df['open_acc'])
df['issue_month'] = label_encoder.fit_transform(df['issue_month'])
df['issue_year'] = label_encoder.fit_transform(df['issue_year'])
df['earliest_cr_line_month'] =
label_encoder.fit_transform(df['earliest_cr_line_month'])
df['earliest_cr_line_year'] =
label_encoder.fit_transform(df['earliest_cr_line_year'])
```

```
df.head()
```

```
   loan_amnt  term  int_rate  grade  sub_grade
emp_title  \
0    10000.0     0     11.44      1          8
Marketing
1     8000.0     0     11.99      1          9          Credit analyst

2    15600.0     0     10.49      1          7
Statistician
3     7200.0     0      6.49      0          1          Client
```

```
Advocate
4    24375.0     1     17.27      2          14  Destiny Management
Inc.

   emp_length  home_ownership  annual_inc  verification_status  ...  \
0           1               5    117000.0                    0  ...
1           4               1     65000.0                    0  ...
2          10               5     43057.0                    1  ...
3           6               5     54000.0                    0  ...
4           9               1     55000.0                    2  ...

   revol_util  total_acc  initial_list_status  application_type
mort_acc  \
0        41.8       25.0                    0                 1
0
1        53.3       27.0                    1                 1
1
2        92.2       26.0                    1                 1
0
3        21.5       13.0                    1                 1
0
4        69.8       43.0                    1                 1
1

   pub_rec_bankruptcies  issue_month  issue_year
earliest_cr_line_month  \
0                     0            4           3
6
1                     0            4           3
5
2                     0            4           3
1
3                     0            9           2
11
4                     0            0           1
7

   earliest_cr_line_year
0                     40
1                     54
2                     57
3                     56
4                     49

[5 rows x 26 columns]
```

```
df1 = pd.DataFrame({'emp_title' : df['emp_title'], 'target' :
df['loan_status']})
target_mean = df1.groupby(by=['emp_title'])['target'].mean()
```

```
df['emp_title'] = df1['emp_title'].map(target_mean)
df.head()
```

```
   loan_amnt  term  int_rate  grade  sub_grade  emp_title  emp_length
\
0    10000.0     0     11.44      1          8   0.240964           1

1     8000.0     0     11.99      1          9   0.333333           4

2    15600.0     0     10.49      1          7   0.200000          10

3     7200.0     0      6.49      0          1   0.000000           6

4    24375.0     1     17.27      2         14   1.000000           9


   home_ownership  annual_inc  verification_status  ...  revol_util  \
0               5    117000.0                    0  ...        41.8
1               1     65000.0                    0  ...        53.3
2               5     43057.0                    1  ...        92.2
3               5     54000.0                    0  ...        21.5
4               1     55000.0                    2  ...        69.8

   total_acc  initial_list_status  application_type  mort_acc  \
0       25.0                    0                 1         0
1       27.0                    1                 1         1
2       26.0                    1                 1         0
3       13.0                    1                 1         0
4       43.0                    1                 1         1

   pub_rec_bankruptcies  issue_month  issue_year
earliest_cr_line_month  \
0                     0            4           3
6
1                     0            4           3
5
2                     0            4           3
1
3                     0            9           2
11
4                     0            0           1
7

   earliest_cr_line_year
0                     40
1                     54
2                     57
3                     56
4                     49
```

```
[5 rows x 26 columns]
```

```
X=df.drop('loan_status',axis=1)
y=df['loan_status']
X_train, X_test, y_train, y_test =
train_test_split(X,y,test_size=0.30,stratify=y,random_state=42)
print(X_train.shape)
print(X_test.shape)
```

```
(218442, 25)
(93618, 25)
```

```
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

MinMaxScaler - For each value in a feature, MinMaxScaler subtracts the minimum value in the feature and then divides by the range. The range is the difference between the original maximum and original minimum.

MinMaxScaler preserves the shape of the original distribution. It doesn't meaningfully change the information embedded in the original data.

```
model = LogisticRegression(max_iter=1000)
model.fit(X_train,y_train)
```

```
LogisticRegression(max_iter=1000)
```

```
y_pred = model.predict(X_test)
```

```
model.score(X_test,y_test)
```

```
0.8883013950308701
```

Accuracy of Logistic Regression Classifier on test set: 0.888

```
x_sm = sm.add_constant(X_train)
sm_model = sm.OLS(y_train, x_sm)
result = sm_model.fit()
print(result.summary())
```

```
                            OLS Regression Results

==============================================================================
========
Dep. Variable:            loan_status   R-squared:
0.493
Model:                            OLS   Adj. R-squared:
0.493
Method:                 Least Squares   F-statistic:
```

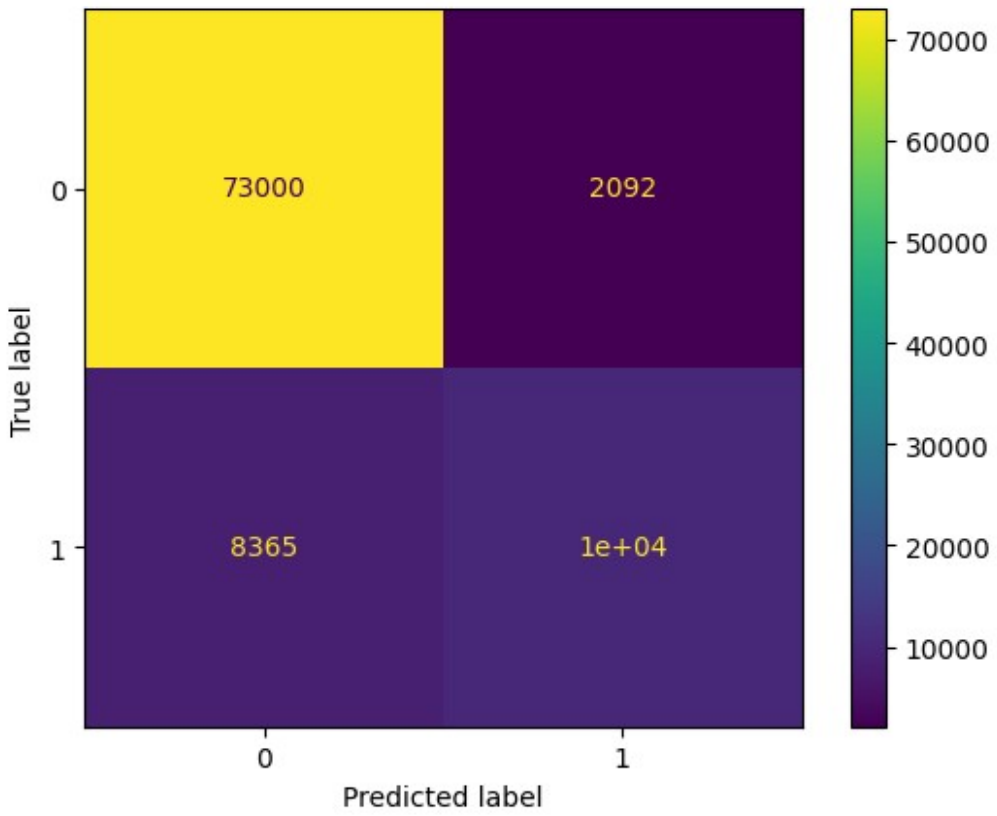
```
8503.
Date:                Mon, 18 Dec 2023   Prob (F-statistic):
0.00
Time:                        22:29:23   Log-Likelihood:
-34693.
No. Observations:              218442   AIC:
6.944e+04
Df Residuals:                  218416   BIC:
6.971e+04
Df Model:                          25

Covariance Type:            nonrobust

==============================================================================
                 coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
const         -0.0225      0.020     -1.122      0.262      -0.062
0.017
x1             0.0461      0.004     11.861      0.000       0.038
0.054
x2             0.0409      0.002     22.629      0.000       0.037
0.044
x3            -0.4537      0.019    -23.315      0.000      -0.492
-0.416
x4             0.0180      0.013      1.387      0.165      -0.007
0.043
x5             0.6724      0.026     26.239      0.000       0.622
0.723
x6             0.9506      0.002    414.839      0.000       0.946
0.955
x7             0.0129      0.002      6.688      0.000       0.009
0.017
x8             0.0200      0.002     11.678      0.000       0.017
0.023
x9             0.0017      0.006      0.307      0.759      -0.009
0.013
x10            0.0011      0.002      0.667      0.505      -0.002
0.004
x11           -0.0017      0.004     -0.436      0.663      -0.009
0.006
x12            0.0914      0.004     24.012      0.000       0.084
0.099
x13            0.0615      0.005     12.637      0.000       0.052
0.071
x14            0.0095      0.004      2.455      0.014       0.002
0.017
```

```
x15            -0.0297      0.005     -5.688      0.000      -0.040
-0.019
x16             0.0444      0.004     11.402      0.000       0.037
0.052
x17            -0.0448      0.005     -9.671      0.000      -0.054
-0.036
x18             0.0082      0.001      6.129      0.000       0.006
0.011
x19            -0.1559      0.038     -4.099      0.000      -0.230
-0.081
x20            -0.0091      0.002     -5.123      0.000      -0.013
-0.006
x21            -0.0166      0.004     -3.926      0.000      -0.025
-0.008
x22            -0.0074      0.002     -3.784      0.000      -0.011
-0.004
x23            -0.0624      0.003    -19.983      0.000      -0.069
-0.056
x24        -2.272e-05      0.002     -0.012      0.990      -0.004
0.004
x25            -0.0076      0.006     -1.248      0.212      -0.020
0.004
==============================================================================
========
Omnibus:                    55389.581   Durbin-Watson:
1.994
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
121380.913
Skew:                           1.474   Prob(JB):
0.00
Kurtosis:                       5.156   Cond. No.
163.
==============================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

```
dic = {}
for coef, col in zip(model.coef_[0], df.columns):
  dic[col] = abs(coef)
a = sorted(dic.items(), key = lambda x: (x[1], x[0]))
for i in a:
  print(i)
```

```
('issue_year', 0.00456714534578184)
('loan_status', 0.011176908794713572)
('pub_rec_bankruptcies', 0.023389120417974012)
('issue_month', 0.047701926039003424)
```

```
('grade', 0.05170318341278431)
('verification_status', 0.06319578744910836)
('total_acc', 0.08641689774049566)
('annual_inc', 0.0977822765509364)
('application_type', 0.10787115506537491)
('earliest_cr_line_month', 0.1283548118959878)
('open_acc', 0.13003336325452106)
('emp_length', 0.1501358949257413)
('mort_acc', 0.19879094359684973)
('home_ownership', 0.2639314249331047)
('revol_util', 0.4243464562687922)
('term', 0.4262550662994685)
('pub_rec', 0.4588173410020658)
('loan_amnt', 0.5952418785532139)
('revol_bal', 0.7822503571456346)
('dti', 0.8448560919194711)
('purpose', 1.0330763651936297)
('initial_list_status', 1.2157771845681498)
('int_rate', 3.279122936360776)
('sub_grade', 5.723287773324654)
('emp_title', 8.673865421293499)

cm = confusion_matrix(y_test,y_pred)
print(cm)
ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=model.classes_).plot()

[[73000  2092]
 [ 8365 10161]]

<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x199f5316050>
```



There is significant value for false negative and false positive. Which will hamper our prediction due to type-1 or type-2 error.

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.90      0.97      0.93     75092
           1       0.83      0.55      0.66     18526

    accuracy                           0.89     93618
   macro avg       0.86      0.76      0.80     93618
weighted avg       0.88      0.89      0.88     93618
```

Precision score and recall score for full paid status is almost same indicates that model is doing decent job which correctly classified the both of the scenarios

Precision score for charged off status is more than recall score which is perfect

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((218442, 25), (93618, 25), (218442,), (93618,))
```

ROC Curve - An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

True Positive Rate False Positive Rate True Positive Rate (TPR) is a synonym for recall and is therefore defined as follows:

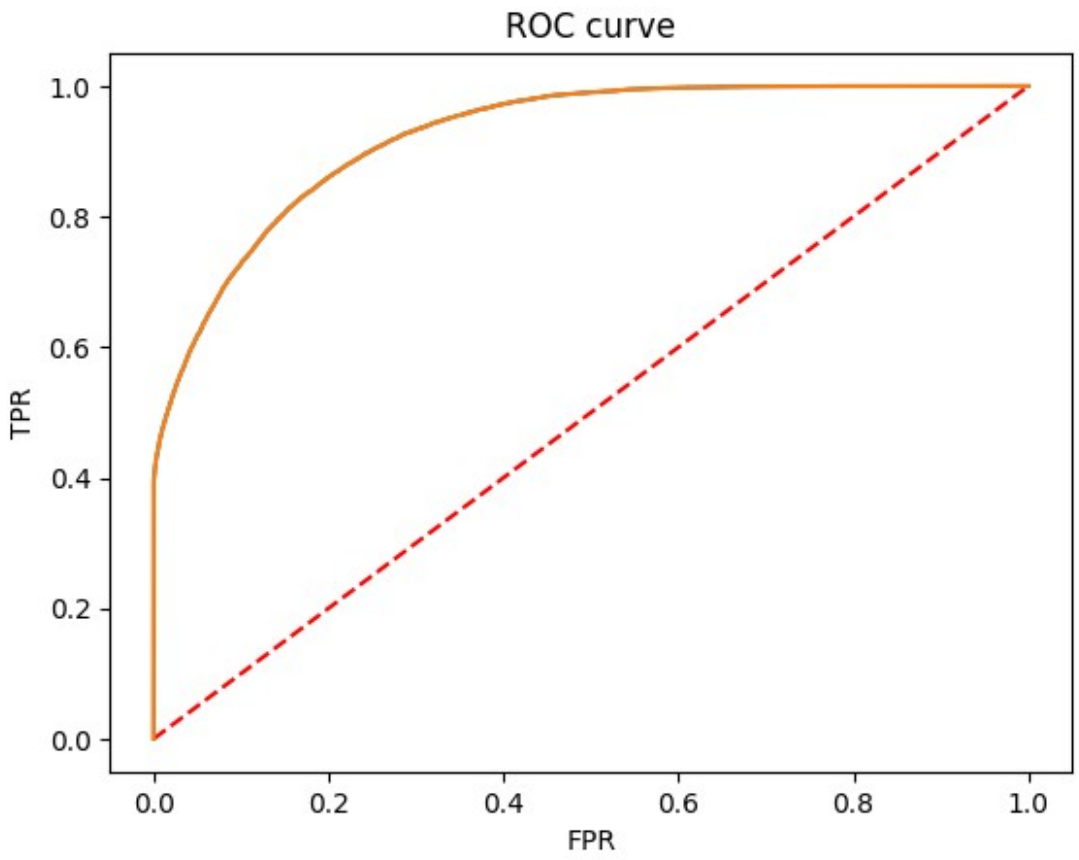
TPR=(TP)/(TP+FN) False Positive Rate (FPR) is defined as follows:

FPR=(FP)/(FP+TN) An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives. The following figure shows a typical ROC curve.

AUC (Area under the ROC Curve) - AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1).

AUC provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example. For example, given the following examples, which are arranged from left to right in ascending order of logistic regression predictions.

```
prob = (model.predict_proba(X_test))[:,1]
fpr, tpr, thr = roc_curve(y_test, prob)
logit_roc_auc = roc_auc_score(y_test,model.predict(X_test))

plt.plot(fpr,tpr)
plt.plot(fpr,fpr,'--',color='red' )
plt.plot(fpr,tpr,label='Logistic Regression (area = %0.2f)' %
logit_roc_auc)
plt.title('ROC curve')
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.show()
```



ROC-AUC curve is grossing the area which indicates that model is performing well. There is still room for some model improvement

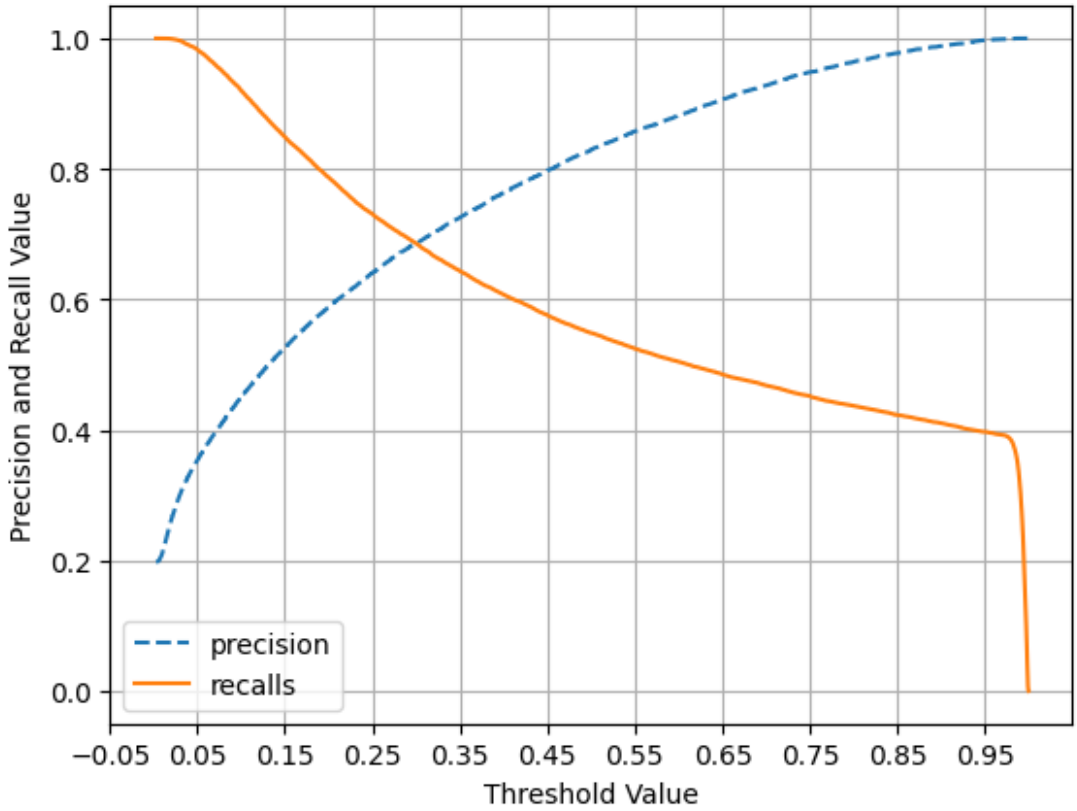
By collecting more data, using a more complex model, or tuning the hyperparameters, it is possible to improve the model's performance.

```
roc_auc_score(y_test,prob)
```

```
0.9235783931593985
```

```
def precission_recall_curve_plot(y_test,pred_proba_c1):
    precisions, recalls, thresholds =
precision_recall_curve(y_test,pred_proba_c1)

    threshold_boundary = thresholds.shape[0]
    #plot precision

plt.plot(thresholds,precisions[0:threshold_boundary],linestyle='--',la
bel='precision')
    #plot recall
    plt.plot(thresholds,recalls[0:threshold_boundary],label='recalls')

    start,end=plt.xlim()
    plt.xticks(np.round(np.arange(start,end,0.1),2))
```

```
    plt.xlabel('Threshold Value')
    plt.ylabel('Precision and Recall Value')
    plt.legend()
    plt.grid()
    plt.show()

precission_recall_curve_plot(y_test,model.predict_proba(X_test)[:,1])
```



Multicollinearity check using Variance Inflation Factor (VIF) - Multicollinearity occurs when two or more independent variables are highly correlated with one another in a regression model. Multicollinearity can be a problem in a regression model because we would not be able to distinguish between the individual effects of the independent variables on the dependent variable.

Multicollinearity can be detected via various methods. One such method is Variance Inflation Factor aka VIF. In VIF method, we pick each independent feature and regress it against all of the other independent features. VIF score of an independent variable represents how well the variable is explained by other independent variables.

```
def calc_vif(X):
    # Calculating the VIF
    vif=pd.DataFrame()
    vif['Feature']=X.columns
```

```
    vif['VIF']=[variance_inflation_factor(X.values,i) for i in
range(X.shape[1])]
    vif['VIF']=round(vif['VIF'],2)
    vif=vif.sort_values(by='VIF',ascending=False)
    return vif

calc_vif(X)[:5]
```

	Feature	VIF
2	int_rate	407.60
4	sub_grade	245.48
18	application_type	187.12
3	grade	64.70
24	earliest_cr_line_year	58.98

```
X.drop(columns=['int_rate'],axis=1,inplace=True)
calc_vif(X)[:5]
```

	Feature	VIF
17	application_type	105.68
3	sub_grade	89.62
2	grade	64.66
23	earliest_cr_line_year	58.63
15	total_acc	13.26

```
X.drop(columns=['application_type'],axis=1,inplace=True)
calc_vif(X)[:5]
```

	Feature	VIF
3	sub_grade	88.98
2	grade	63.94
22	earliest_cr_line_year	26.43
15	total_acc	12.82
11	open_acc	12.09

```
X.drop(columns=['sub_grade'],axis=1,inplace=True)
calc_vif(X)[:5]
```

	Feature	VIF
21	earliest_cr_line_year	25.85
14	total_acc	12.82
10	open_acc	12.08
13	revol_util	8.91
6	annual_inc	8.25

```
X.drop(columns=['earliest_cr_line_year'],axis=1,inplace=True)
calc_vif(X)[:5]
```

	Feature	VIF
14	total_acc	12.74
10	open_acc	11.56

```
6   annual_inc   8.11
9          dti   8.02
13  revol_util   7.98
```

```
X.drop(columns=['total_acc'],axis=1,inplace=True)
calc_vif(X)[:5]
```

```
       Feature   VIF
13  revol_util  7.98
6   annual_inc  7.89
9          dti  7.84
10    open_acc  7.39
0    loan_amnt  7.33
```

```
X=scaler.fit_transform(X)

kfold=KFold(n_splits=5)
accuracy=np.mean(cross_val_score(model,X,y,cv=kfold,scoring='accuracy'
,n_jobs=-1))
print("Cross Validation accuracy : {:.3f}".format(accuracy))
```

```
Cross Validation accuracy : 0.888
```

```
sm=SMOTE(random_state=42)
X_train_res,y_train_res=sm.fit_resample(X_train,y_train.ravel())

print('After OverSampling, the shape of train_X:
{}'.format(X_train_res.shape))
print('After OverSampling, the shape of train_y: {} \
n'.format(y_train_res.shape))

print("After OverSampling, counts of label '1':
{}".format(sum(y_train_res == 1)))
print("After OverSampling, counts of label '0':
{}".format(sum(y_train_res == 0)))
```

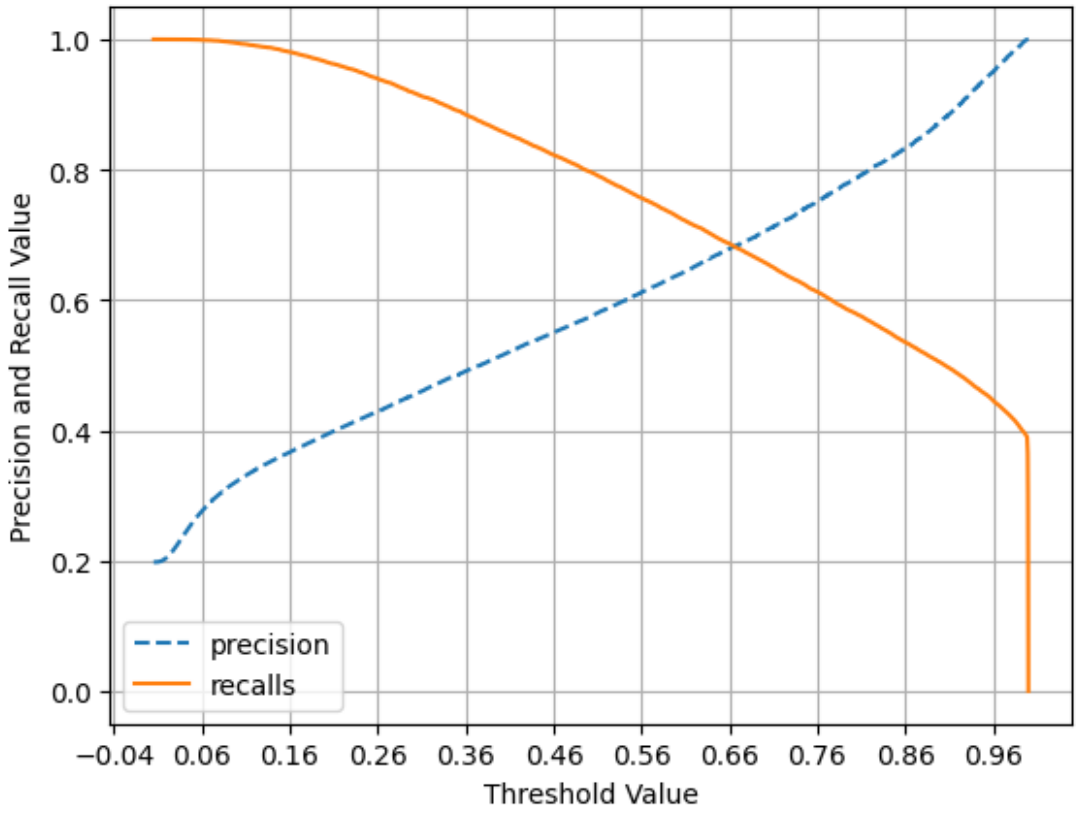
```
After OverSampling, the shape of train_X: (350428, 25)
After OverSampling, the shape of train_y: (350428,)

After OverSampling, counts of label '1': 175214
After OverSampling, counts of label '0': 175214
```

```
lr1 = LogisticRegression(max_iter=1000)
lr1.fit(X_train_res, y_train_res)
predictions = lr1.predict(X_test)

# Classification Report
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

           0       0.94      0.85      0.90     75092
```

```
           1       0.57      0.80      0.67     18526

    accuracy                           0.84     93618
   macro avg       0.76      0.83      0.78     93618
weighted avg       0.87      0.84      0.85     93618
```

```
precission_recall_curve_plot(y_test, lr1.predict_proba(X_test)[:,1])
```




```
df = pd.read_csv('logistic_regression.csv')
```

1. What percentage of customers have fully paid their Loan Amount?

```
df ['loan_status'].value_counts(normalize=True)*100
```

```
loan_status
Fully Paid     80.387092
Charged Off    19.612908
Name: proportion, dtype: float64
```

80%

1. Comment about the correlation between Loan Amount and Installment features.

The spearman correlation coefficient between Loan Amount and Installmen is very high (i.e. 0.97)

1. The majority of people have home ownership as _____.

```
df ['home_ownership'].value_counts(normalize=True)*100

home_ownership
MORTGAGE    50.084085
RENT        40.347953
OWN          9.531096
OTHER        0.028281
NONE         0.007828
ANY          0.000758
Name: proportion, dtype: float64

df['home_ownership'].value_counts()

home_ownership
MORTGAGE    198348
RENT        159790
OWN          37746
OTHER          112
NONE            31
ANY              3
Name: count, dtype: int64
```

Mortgage

1. People with grades 'A' are more likely to fully pay their loan. (T/F)

True.

Out of all people with grade 'A', 93% got their loan approved.

1. Name the top 2 afforded job titles.

Teacher & Manager

1. Thinking from a bank's perspective, which metric should our primary focus be on.. ROC AUC Precision Recall F1 Score

It should be on f1 score. as we need to give importance to both precision and recall. We don't want to miss potential customers and at the same time we also don't want to give loan to defaulters

1. How does the gap in precision and recall affect the bank?

Recall score: 0.94 and Precision score: 0.85. Which tells us that there are more false positives than the false negatives.

From Confusion Matrix it can be seen that FP = 10% of total cases & FN = 0.9% of Total Cases

If Recall value is low (i.e. FN are high), it means Bank is loosing in opportunity cost.

If Precision value is low (i.e. FP are high), it means Bank's NPA (defaulters) may increase.

1. Which were the features that heavily affected the outcome?

Using RFE we were able to identify top_20 features which has high impact on Outcome. This include:

int_rate: Interest Rate

sub_grade: loan subgrade

term : number of payments on the loan

home_ownership

purpose

application_type

pincode (from address)

emp_title: job title supplied by the Borrower

1. Will the results be affected by geographical location? (Yes/No)

pincode (derived from address) has significant impact on the outcome.

How can we make sure that our model can detect real defaulters and there are less false positives? This is important as we can lose out on an opportunity to finance more individuals and earn interest on it.

Answer - Since data is imbalances by making the data balance we can try to avoid false positives. For evaluation metrics, we should be focusing on the macro average f1-score because we don't want to make false positive prediction and at the same we want to detect the defualers. Since NPA (non-performing asset) is a real problem in this industry, it’s important we play safe and shouldn’t disburse loans to anyone

Answer - Below are the most features and their importance while making the prediction. So these variables can help the managers to identify which are customers who are more likely to pay the loan amount fully,

Actional Insights and Recommendations

80% of the customers have paid the loan fully.

20% of the customers are the defaulters.

The organization can the trained model to make prediction for whether a person will likely to pay the loan amount or he will be a defaulter.

Cross Validation accuracy and testing accuracy is almost same which infers model is performing the decent job. We can trust this model for unseen data

By collecting more data, using a more complex model, or tuning the hyperparameters, it is possible to improve the model's performance.

ROC AUC curve area of 0.73, the model is correctly classifying about 73% of the instances. This is a good performance, but there is still room for improvement.

The precision-recall curve allows us to see how the precision and recall trade-off as we vary the threshold. A higher threshold will result in higher precision, but lower recall, and vice versa. The ideal point on the curve is the one that best meets the needs of the specific application.

After balancing the dataset, there is significant change observed in the precion and recall score for both of the classes.

Accuracy of Logistic Regression Classifier on test set: 0.888 which is decent and not by chance.