LoanTapML_HarishSV

November 2, 2025

[ ]:
```
#Business Case: LoanTap Logistic Regression
#Loantap is a leading financial technology company based in India, specializing␣
↪in providing flexible and innovative loan products to individuals and␣
↪businesses
```

[ ]:
```
#Our Task:
#As a data scientist at LoanTap, you are tasked with analyzing the dataset to␣
↪determine the creditworthiness of potential borrowers.
#Ultimate objective is to build a logistic regression model, evaluate its␣
↪performance, and provide actionable insights for the underwriting process.
```

[ ]:
```
#Exploratory Data Analysis
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from scipy.stats import ttest_ind,chi2_contingency

from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, KFold, cross_val_score
from sklearn.preprocessing import MinMaxScaler, LabelEncoder, StandardScaler
from sklearn.metrics import (
    accuracy_score, confusion_matrix, classification_report,
    roc_auc_score, roc_curve, auc, precision_recall_curve,␣
↪average_precision_score,
    ConfusionMatrixDisplay,␣
↪RocCurveDisplay,f1_score,recall_score,precision_score
)

from statsmodels.stats.outliers_influence import variance_inflation_factor
from imblearn.over_sampling import SMOTE

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

[ ]:

```python
from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving logistic_regression.csv to logistic_regression.csv
```

[ ]:

```python
lt_data =pd.read_csv('logistic_regression.csv')
```

[ ]:

```python
df = lt_data.copy()
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]
```

[ ]:
```
pd.set_option('display.max_columns', None)
```

[ ]:
```
df.shape
```

[ ]:
```
(396030, 27)
```

[ ]:
```
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
```

[ ]:
```
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')
```

[ ]:
```
#Statistical Summary
df.describe().T
```

[ ]:
```
                         count          mean           std     min       25%  \
loan_amnt             396030.0  14113.888089   8357.441341  500.00   8000.00
int_rate              396030.0     13.639400      4.472157    5.32     10.49
installment           396030.0    431.849698    250.727790   16.08    250.33
annual_inc            396030.0  74203.175798  61637.621158    0.00  45000.00
dti                   396030.0     17.379514     18.019092    0.00     11.28
open_acc              396030.0     11.311153      5.137649    0.00      8.00
pub_rec               396030.0      0.178191      0.530671    0.00      0.00
revol_bal             396030.0  15844.539853  20591.836109    0.00   6025.00
revol_util            395754.0     53.791749     24.452193    0.00     35.80
total_acc             396030.0     25.414744     11.886991    2.00     17.00
mort_acc              358235.0      1.813991      2.147930    0.00      0.00
pub_rec_bankruptcies  395495.0      0.121648      0.356174    0.00      0.00

                           50%       75%         max
loan_amnt             12000.00  20000.00    40000.00
int_rate                 13.33     16.49       30.99
installment             375.43    567.30     1533.81
annual_inc            64000.00  90000.00  8706582.00
dti                      16.91     22.98     9999.00
open_acc                 10.00     14.00       90.00
pub_rec                   0.00      0.00       86.00
revol_bal             11181.00  19620.00  1743266.00
revol_util               54.80     72.90      892.30
total_acc                24.00     32.00      151.00
mort_acc                  1.00      3.00       34.00
pub_rec_bankruptcies      0.00      0.00        8.00
```

[ ]:
```
df.describe(include='object').T
```

[ ]:
```
                      count  unique                        top    freq
term                 396030       2                  36 months  302005
grade                396030       7                          B  116018
sub_grade            396030      35                         B3   26655
emp_title            373103  173105                    Teacher    4389
emp_length           377729      11                  10+ years  126041
home_ownership       396030       6                   MORTGAGE  198348
verification_status  396030       3                   Verified  139563
```

```
issue_d              396030     115                   Oct-2014   14846
loan_status          396030       2                 Fully Paid  318357
purpose              396030      14         debt_consolidation  234507
title                394274   48816         Debt consolidation  152472
earliest_cr_line     396030     684                   Oct-2000    3017
initial_list_status  396030       2                          f  238066
application_type     396030       3                 INDIVIDUAL  395319
address              396030  393700  USS Johnson\r\nFPO AE 48052       8
```

[ ]:
```
#Duplicate Detection
df[df.duplicated()]
```

[ ]:
```
Empty DataFrame
Columns: [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]
Index: []
```

Insights The dataset does not contain any duplicates.

[ ]:
```
#Null Detection
df.isna().any()[df.isna().any()]
```

[ ]:
```
emp_title               True
emp_length              True
title                   True
revol_util              True
mort_acc                True
pub_rec_bankruptcies    True
dtype: bool
```

[ ]:
```
df.isna().sum().sort_values(ascending=False)
```

[ ]:
```
mort_acc                37795
emp_title               22927
emp_length              18301
title                    1756
pub_rec_bankruptcies      535
revol_util                276
installment                 0
int_rate                    0
term                        0
grade                       0
loan_amnt                   0
verification_status         0
```

```
annual_inc              0
home_ownership          0
sub_grade               0
dti                     0
issue_d                 0
loan_status             0
purpose                 0
pub_rec                 0
open_acc                0
earliest_cr_line        0
revol_bal               0
initial_list_status     0
total_acc               0
application_type        0
address                 0
dtype: int64
```

[ ]:
```
def missing_data(df):
    total_missing_df = df.isnull().sum().sort_values(ascending =False)
    percent_missing_df = (df.isnull().sum()/df.isna().count()*100).
 ↪sort_values(ascending=False)
    missing_data_df = pd.concat([total_missing_df, percent_missing_df], axis=1,␣
 ↪keys=['Total', 'Percent'])
    return missing_data_df

missing_pct = missing_data(df)
missing_pct[missing_pct['Total']>0]
```

[ ]:

	Total	Percent
mort_acc	37795	9.543469
emp_title	22927	5.789208
emp_length	18301	4.621115
title	1756	0.443401
pub_rec_bankruptcies	535	0.135091
revol_util	276	0.069692

Insight

emp_title has 5.78% missing values

emp_length has 4.62% missing values

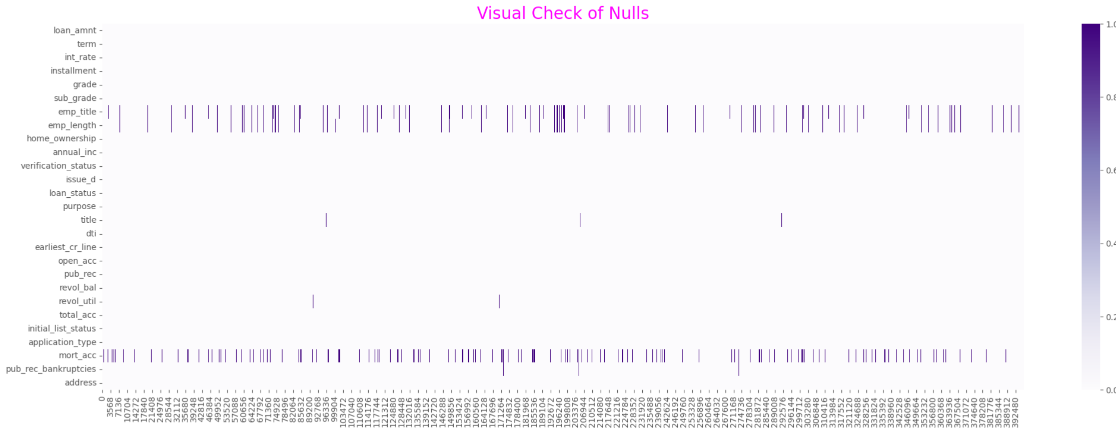
title has 0.44% missing values

revol_until has 0.06% missing values

mort_acc has 9.54% missing values

pub_rec_bankruptcies has 0.13% missing values

```
[ ]: #Since ML algorithm do not work on columns which has missing values so we need
     ↪to impute these missing values.
     plt.figure(figsize=(25,8))
     plt.style.use('ggplot')
     sns.heatmap(df.isnull().T,cmap='Purples')
     plt.title('Visual Check of Nulls',fontsize=20,color='magenta')
     plt.show()
```




```
[ ]: df.isna().sum().sum()
     # since there are 81590 rows are null , we cant drop na ...
```

```
[ ]: np.int64(81590)
```

```
[ ]: #checking the unique values for columns
     for _ in df.columns:
         print()
         print(f'Total Unique Values in {_} column are :- {df[_].nunique()}')
         print(f'Unique Values in {_} column are :-\n {df[_].unique()}')
         print(f'Value_counts of {_} column :-\n {df[_].value_counts()}')
         print()
         print('-'*120)
```

```
Total Unique Values in loan_amnt column are :- 1397
Unique Values in loan_amnt column are :-
 [10000.  8000. 15600. … 36275. 36475.   725.]
Value_counts of loan_amnt column :-
 loan_amnt
10000.0    27668
12000.0    21366
15000.0    19903
20000.0    18969
```

```
35000.0    14576
           …
39200.0        1
38750.0        1
36275.0        1
36475.0        1
725.0          1
Name: count, Length: 1397, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in term column are :- 2
Unique Values in term column are :-
 [' 36 months' ' 60 months']
Value_counts of term column :-
 term
36 months    302005
60 months     94025
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in int_rate column are :- 566
Unique Values in int_rate column are :-
 [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]
Value_counts of int_rate column :-
 int_rate
10.99    12411
12.99     9632
15.61     9350
11.99     8582
8.90      8019
         …
14.38        1
24.40        1
22.64        1
17.54        1
17.44        1
Name: count, Length: 566, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in installment column are :- 55706
Unique Values in installment column are :-
 [329.48 265.68 506.97 … 343.14 118.13 572.44]
Value_counts of installment column :-
```

```
 installment
327.34     968
332.10     791
491.01     736
336.90     686
392.81     683
          …
1146.14      1
218.49       1
961.66       1
569.10       1
555.96       1
Name: count, Length: 55706, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in grade column are :- 7
Unique Values in grade column are :-
 ['B' 'A' 'C' 'E' 'D' 'F' 'G']
Value_counts of grade column :-
 grade
B    116018
C    105987
A     64187
D     63524
E     31488
F     11772
G      3054
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in sub_grade column are :- 35
Unique Values in sub_grade column are :-
 ['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']
Value_counts of sub_grade column :-
 sub_grade
B3    26655
B4    25601
C1    23662
C2    22580
B2    22495
B5    22085
C3    21221
```

```
C4    20280
B1    19182
A5    18526
C5    18244
D1    15993
A4    15789
D2    13951
D3    12223
D4    11657
A3    10576
A1     9729
D5     9700
A2     9567
E1     7917
E2     7431
E3     6207
E4     5361
E5     4572
F1     3536
F2     2766
F3     2286
F4     1787
F5     1397
G1     1058
G2      754
G3      552
G4      374
G5      316
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in emp_title column are :- 173105
Unique Values in emp_title column are :-
 ['Marketing' 'Credit analyst ' 'Statistician' …
 "Michael's Arts & Crafts" 'licensed bankere' 'Gracon Services, Inc']
Value_counts of emp_title column :-
 emp_title
Teacher                    4389
Manager                    4250
Registered Nurse           1856
RN                         1846
Supervisor                 1830
                           …
OMIV Supervisor               1
SVP, Technology               1
sikorsky                      1
```

```
Postman                          1
Sr. Facilities Caretaker         1
Name: count, Length: 173105, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in emp_length column are :- 11
Unique Values in emp_length column are :-
 ['10+ years' '4 years' '< 1 year' '6 years' '9 years' '2 years' '3 years'
 '8 years' '7 years' '5 years' '1 year' nan]
Value_counts of emp_length column :-
 emp_length
10+ years    126041
2 years       35827
< 1 year      31725
3 years       31665
5 years       26495
1 year        25882
4 years       23952
6 years       20841
7 years       20819
8 years       19168
9 years       15314
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in home_ownership column are :- 6
Unique Values in home_ownership column are :-
 ['RENT' 'MORTGAGE' 'OWN' 'OTHER' 'NONE' 'ANY']
Value_counts of home_ownership column :-
 home_ownership
MORTGAGE    198348
RENT        159790
OWN          37746
OTHER          112
NONE            31
ANY              3
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in annual_inc column are :- 27197
Unique Values in annual_inc column are :-
 [117000.    65000.    43057.   …  36111.    47212.    31789.88]
```

```
Value_counts of annual_inc column :-
 annual_inc
60000.0    15313
50000.0    13303
65000.0    11333
70000.0    10674
40000.0    10629
           …
67842.0        1
72179.0        1
50416.0        1
46820.8        1
87622.0        1
Name: count, Length: 27197, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in verification_status column are :- 3
Unique Values in verification_status column are :-
 ['Not Verified' 'Source Verified' 'Verified']
Value_counts of verification_status column :-
 verification_status
Verified           139563
Source Verified    131385
Not Verified       125082
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in issue_d column are :- 115
Unique Values in issue_d column are :-
 ['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']
Value_counts of issue_d column :-
 issue_d
Oct-2014    14846
Jul-2014    12609
Jan-2015    11705
Dec-2013    10618
Nov-2013    10496
            …
Aug-2007       26
Sep-2008       25
Nov-2007       22
Sep-2007       15
Jun-2007        1
Name: count, Length: 115, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in loan_status column are :- 2
Unique Values in loan_status column are :-
 ['Fully Paid' 'Charged Off']
Value_counts of loan_status column :-
 loan_status
Fully Paid     318357
Charged Off     77673
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in purpose column are :- 14
Unique Values in purpose column are :-
 ['vacation' 'debt_consolidation' 'credit_card' 'home_improvement'
 'small_business' 'major_purchase' 'other' 'medical' 'wedding' 'car'
 'moving' 'house' 'educational' 'renewable_energy']
Value_counts of purpose column :-
 purpose
debt_consolidation    234507
credit_card            83019
home_improvement       24030
other                  21185
major_purchase          8790
small_business          5701
```

```
car                   4697
medical               4196
moving                2854
vacation              2452
house                 2201
wedding               1812
renewable_energy       329
educational            257
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in title column are :- 48816
Unique Values in title column are :-
 ['Vacation' 'Debt consolidation' 'Credit card refinancing' …
 'Credit buster ' 'Loanforpayoff' 'Toxic Debt Payoff']
Value_counts of title column :-
 title
Debt consolidation        152472
Credit card refinancing    51487
Home improvement           15264
Other                      12930
Debt Consolidation         11608
                            …
creditcardrefi                 1
Debt/Home                      1
Peace Of Mind Loan             1
Blazer repair                  1
Out of my rut                  1
Name: count, Length: 48816, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in dti column are :- 4262
Unique Values in dti column are :-
 [26.24 22.05 12.79 … 40.56 47.09 55.53]
Value_counts of dti column :-
 dti
0.00      313
14.40     310
19.20     302
16.80     301
18.00     300
         …
47.05       1
46.52       1
```

```
1622.00      1
40.21        1
189.90       1
Name: count, Length: 4262, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in earliest_cr_line column are :- 684
Unique Values in earliest_cr_line column are :-
 ['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']
Value_counts of earliest_cr_line column :-
 earliest_cr_line
Oct-2000    3017
Aug-2000    2935
Oct-2001    2896
Aug-2001    2884
Nov-2000    2736
            …
Feb-1957       1
Nov-1950       1
May-1955       1
Sep-1961       1
Nov-1955       1
Name: count, Length: 684, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in open_acc column are :- 61
Unique Values in open_acc column are :-
```

```
 [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.]
Value_counts of open_acc column :-
 open_acc
9.0     36779
10.0    35441
8.0     35137
11.0    32695
7.0     31328
        …
56.0        2
55.0        2
57.0        1
58.0        1
90.0        1
Name: count, Length: 61, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in pub_rec column are :- 20
Unique Values in pub_rec column are :-
 [ 0.  1.  2.  3.  4.  6.  5.  8.  9. 10. 11.  7. 19. 13. 40. 17. 86. 12.
 24. 15.]
Value_counts of pub_rec column :-
 pub_rec
0.0     338272
1.0      49739
2.0       5476
3.0       1521
4.0        527
5.0        237
6.0        122
7.0         56
8.0         34
9.0         12
10.0        11
11.0         8
13.0         4
12.0         4
19.0         2
40.0         1
17.0         1
86.0         1
24.0         1
15.0         1
```

```
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in revol_bal column are :- 55622
Unique Values in revol_bal column are :-
 [ 36369.  20131.  11987. …  34531. 151912.  29244.]
Value_counts of revol_bal column :-
 revol_bal
0.0         2128
5655.0        41
7792.0        38
6095.0        38
3953.0        37
            …
43895.0        1
46733.0        1
36519.0        1
212269.0       1
71547.0        1
Name: count, Length: 55622, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in revol_util column are :- 1226
Unique Values in revol_util column are :-
 [ 41.8   53.3   92.2  …  56.26 111.4  128.1 ]
Value_counts of revol_util column :-
 revol_util
0.00      2213
53.00      752
60.00      739
61.00      734
55.00      730
          …
146.10       1
109.30       1
108.10       1
115.30       1
37.63        1
Name: count, Length: 1226, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in total_acc column are :- 118
```

```
Unique Values in total_acc column are :-
 [ 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.]
Value_counts of total_acc column :-
 total_acc
21.0     14280
22.0     14260
20.0     14228
23.0     13923
24.0     13878
         …
150.0        1
117.0        1
115.0        1
100.0        1
103.0        1
Name: count, Length: 118, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in initial_list_status column are :- 2
Unique Values in initial_list_status column are :-
 ['w' 'f']
Value_counts of initial_list_status column :-
 initial_list_status
f    238066
w    157964
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in application_type column are :- 3
Unique Values in application_type column are :-
 ['INDIVIDUAL' 'JOINT' 'DIRECT_PAY']
Value_counts of application_type column :-
 application_type
INDIVIDUAL    395319
JOINT            425
DIRECT_PAY       286
```

```
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in mort_acc column are :- 33
Unique Values in mort_acc column are :-
 [ 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.]
Value_counts of mort_acc column :-
 mort_acc
0.0     139777
1.0      60416
2.0      49948
3.0      38049
4.0      27887
5.0      18194
6.0      11069
7.0       6052
8.0       3121
9.0       1656
10.0       865
11.0       479
12.0       264
13.0       146
14.0       107
15.0        61
16.0        37
17.0        22
18.0        18
19.0        15
20.0        13
24.0        10
22.0         7
21.0         4
25.0         4
27.0         3
26.0         2
32.0         2
31.0         2
23.0         2
34.0         1
28.0         1
30.0         1
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------
```

```
Total Unique Values in pub_rec_bankruptcies column are :- 9
Unique Values in pub_rec_bankruptcies column are :-
 [ 0.  1.  2.  3. nan  4.  5.  6.  7.  8.]
Value_counts of pub_rec_bankruptcies column :-
 pub_rec_bankruptcies
0.0    350380
1.0     42790
2.0      1847
3.0       351
4.0        82
5.0        32
6.0         7
7.0         4
8.0         2
Name: count, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------

Total Unique Values in address column are :- 393700
Unique Values in address column are :-
 ['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']
Value_counts of address column :-
 address
USS Johnson\r\nFPO AE 48052                                  8
USNS Johnson\r\nFPO AE 05113                                 8
USS Smith\r\nFPO AP 70466                                    8
USCGC Smith\r\nFPO AE 70466                                  8
USNS Johnson\r\nFPO AP 48052                                 7
                                                            ..
8141 Cox Greens Suite 186\r\nMadisonstad, VT 05113           1
8803 Sean Highway Suite 029\r\nNorth Nicoleshire, AK 11650    1
594 Nicole Mission Apt. 620\r\nNew Patrick, NJ 00813         1
7336 Sean Groves Apt. 893\r\nDariusborough, NJ 05113         1
9160 Tucker Squares\r\nSouth Paul, MO 30723                  1
Name: count, Length: 393700, dtype: int64

--------------------------------------------------------------------------------
----------------------------------------
```

[ ]:
```
#Null Treatment:
df.loc[df['revol_util'].isna(),'revol_util'] = 0.0
df.loc[df['mort_acc'].isna(),'mort_acc'] = 0.0
df.loc[df['pub_rec_bankruptcies'].isna(),'pub_rec_bankruptcies'] = 0.0
df.loc[df['emp_title'].isna(),'emp_title'] = 'No Employee Title'
df.loc[df['title'].isna(),'title'] = 'Unavailable'
df['emp_length'] = df['emp_length'].fillna('< 1 year')
```

[ ]:
```
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.describe().T
```

[ ]:
```
                      count          mean           std     min       25%  \
loan_amnt          396030.0  14113.888089   8357.441341  500.00   8000.00
int_rate           396030.0     13.639400      4.472157    5.32     10.49
installment        396030.0    431.849698    250.727790   16.08    250.33
annual_inc         396030.0  74203.175798  61637.621158    0.00  45000.00
dti                396030.0     17.379514     18.019092    0.00     11.28
```

open_acc	396030.0	11.311153	5.137649	0.00	8.00
pub_rec	396030.0	0.178191	0.530671	0.00	0.00
revol_bal	396030.0	15844.539853	20591.836109	0.00	6025.00
revol_util	396030.0	53.754260	24.484857	0.00	35.80
total_acc	396030.0	25.414744	11.886991	2.00	17.00
mort_acc	396030.0	1.640873	2.111249	0.00	0.00
pub_rec_bankruptcies	396030.0	0.121483	0.355962	0.00	0.00

	50%	75%	max
loan_amnt	12000.00	20000.00	40000.00
int_rate	13.33	16.49	30.99
installment	375.43	567.30	1533.81
annual_inc	64000.00	90000.00	8706582.00
dti	16.91	22.98	9999.00
open_acc	10.00	14.00	90.00
pub_rec	0.00	0.00	86.00
revol_bal	11181.00	19620.00	1743266.00
revol_util	54.80	72.90	892.30
total_acc	24.00	32.00	151.00
mort_acc	1.00	3.00	34.00
pub_rec_bankruptcies	0.00	0.00	8.00

[ ]:
```
df.describe(include='object').T
```

[ ]:

	count	unique	top	freq
term	396030	2	36 months	302005
grade	396030	7	B	116018
sub_grade	396030	35	B3	26655
emp_title	396030	173106	No Employee Title	22927
emp_length	396030	11	10+ years	126041
home_ownership	396030	6	MORTGAGE	198348
verification_status	396030	3	Verified	139563
issue_d	396030	115	Oct-2014	14846
loan_status	396030	2	Fully Paid	318357
purpose	396030	14	debt_consolidation	234507
title	396030	48817	Debt consolidation	152472
earliest_cr_line	396030	684	Oct-2000	3017
initial_list_status	396030	2	f	238066
application_type	396030	3	INDIVIDUAL	395319
address	396030	393700	USS Johnson\r\nFPO AE 48052	8

[ ]:
```
#Feature Engineering
```

[ ]:
```
df['pub_rec'] = [1 if i > 1 else 0 for i in df['pub_rec']]
df['mort_acc'] = [1 if i > 1 else 0 for i in df['mort_acc']]
df['pub_rec_bankruptcies'] = [1 if i > 1 else 0 for i in␣
 ↪df['pub_rec_bankruptcies']]
```

[ ]:
```
df.sample()
```

[ ]:
```
        loan_amnt        term  int_rate  installment grade sub_grade  \
153741    15000.0   36 months     11.14       492.08     B        B2

             emp_title emp_length home_ownership  annual_inc  \
153741  Shawnee County    7 years       MORTGAGE     43000.0

       verification_status   issue_d loan_status     purpose    title    dti  \
153741            Verified  Dec-2012  Fully Paid  credit_card  CC REFI  26.07

       earliest_cr_line  open_acc  pub_rec  revol_bal  revol_util  total_acc  \
153741         Jan-2001       9.0        0    12241.0        80.5       28.0

       initial_list_status application_type  mort_acc  pub_rec_bankruptcies  \
153741                   f       INDIVIDUAL         1                     0

                                                   address
153741  2746 Wood Plaza Suite 589\r\nWhiteside, OH 05113
```

[ ]:
```
#Split issue_date into month and year
df[['issue_month', 'issue_year']] = df['issue_d'].str.split('-', expand=True)
df.drop(['issue_d'], axis=1, inplace=True)
```

[ ]:
```
#Split er_cr_line date into month and year
df[['er_cr_line_m', 'er_cr_line_y']] = df['earliest_cr_line'].str.split('-',␣
 ↪expand=True)
df.drop(['earliest_cr_line'], axis=1, inplace=True)
```

[ ]:
```
df['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
                                ...
396025     12951 Williams Crossing\r\nJohnnyville, DC 30723
396026    0114 Fowler Field Suite 028\r\nRachelborough, …
396027    953 Matthew Points Suite 414\r\nReedfort, NY 7…
396028    7843 Blake Freeway Apt. 229\r\nNew Michael, FL…
396029        787 Michelle Causeway\r\nBriannaton, AR 48052
Name: address, Length: 396030, dtype: object
```

[ ]:
```
#Split address into State and Zip code
import re
df[['state','zipcode']] = df['address'].str.extract(r'([A-Z]{2}) (\d{5})')
```

```python
df.drop(['address'], axis=1, inplace=True)
```

[ ]:
```python
df['state'].nunique() , df['zipcode'].nunique()
```

[ ]:
```
(54, 10)
```

[ ]:
```python
df['state'].isna().sum() , df['zipcode'].isna().sum()
```

[ ]:
```
(np.int64(0), np.int64(0))
```

[ ]:
```python
df['emp_length_yrs'] = df['emp_length'].str.extract('(\d+)')
df.drop(['emp_length'], axis=1, inplace=True)
```

[ ]:
```python
df['term'] = df['term'].str.split().str[0].astype('object')
```

[ ]:
```python
df.sample()
```

[ ]:
```
        loan_amnt term  int_rate  installment grade sub_grade  \
371826    19200.0   60      9.71       405.21     B        B1

                 emp_title home_ownership  annual_inc verification_status  \
371826  FRIENDLY CHEVROLET       MORTGAGE     55000.0     Source Verified

        loan_status             purpose        title    dti  open_acc  \
371826  Charged Off  debt_consolidation  CREDIT CARD  28.89      12.0

        pub_rec  revol_bal  revol_util  total_acc initial_list_status  \
371826        0    18998.0        24.2       31.0                   f

       application_type  mort_acc  pub_rec_bankruptcies issue_month  \
371826       INDIVIDUAL         1                     0         Jul

       issue_year er_cr_line_m er_cr_line_y state zipcode emp_length_yrs
371826       2013          Jun         1998    DE   22690              1
```

[ ]:
```python
df.shape
```

[ ]:
```
(396030, 30)
```

[ ]:
```python
# List of categorical columns
cat_cols = df.select_dtypes(include='object')

# List of numerical columns
num_cols = df.select_dtypes(exclude='object')
```

[ ]:
```python
cat_cols.sample(3)
```

```
[ ]:        term grade sub_grade                             emp_title  \
321098   36     D        D2               elite h.o.a. mgt. inc.
84453    36     B        B3  Senior Windows Systems Administator
256846   36     C        C1                          Team Leader

       home_ownership verification_status loan_status            purpose  \
321098           RENT        Not Verified  Fully Paid  debt_consolidation
84453        MORTGAGE            Verified  Fully Paid    home_improvement
256846       MORTGAGE        Not Verified  Fully Paid               other

                    title initial_list_status application_type issue_month  \
321098  finally paid off                    w       INDIVIDUAL         Jul
84453   Home improvement                    w       INDIVIDUAL         Nov
256846             Other                    f       INDIVIDUAL         Apr

       issue_year er_cr_line_m er_cr_line_y state zipcode emp_length_yrs
321098       2013          Oct         2002    MN   05113             10
84453        2015          Jan         2007    VA   70466              4
256846       2015          Apr         1998    SD   00813             10
```

```
[ ]: num_cols.sample(3)
```

```
[ ]:         loan_amnt  int_rate  installment  annual_inc    dti  open_acc  \
294137    15000.0     10.16       485.14     75000.0  16.66       9.0
380736    26000.0      5.32       782.99    106000.0  17.41      14.0
220412    15000.0     14.09       513.33    100000.0  12.97       5.0

        pub_rec  revol_bal  revol_util  total_acc  mort_acc  \
294137        0    11242.0        82.1       30.0         1
380736        0    19167.0        50.8       29.0         1
220412        0    11383.0        82.8        5.0         0

        pub_rec_bankruptcies
294137                     0
380736                     0
220412                     0
```

```
[ ]: num_cols.skew()
```

```
[ ]: loan_amnt                0.777285
int_rate                 0.420669
installment              0.983598
annual_inc              41.042725
dti                    431.051225
open_acc                 1.213019
pub_rec                  6.812303
revol_bal               11.727515
```

```
revol_util              -0.074238
total_acc                0.864328
mort_acc                 0.412225
pub_rec_bankruptcies    12.936099
dtype: float64
```

Insights

Features are Right skewed

Action

Need to apply log transformations in order to normalise them

[ ]:
```
df1 = df.copy()
```

[ ]:
```
df1.sample()
```

[ ]:
```
       loan_amnt term  int_rate  installment grade sub_grade  \
49162     8400.0   36     16.78       298.57     C        C5

                  emp_title home_ownership  annual_inc verification_status  \
49162  RBC Wealth Management           RENT     36000.0        Not Verified

      loan_status purpose  title   dti  open_acc  pub_rec  revol_bal  \
49162  Fully Paid   other  Other  8.77      11.0        0     7716.0

       revol_util  total_acc initial_list_status application_type  mort_acc  \
49162        29.5       24.0                   f       INDIVIDUAL         0

       pub_rec_bankruptcies issue_month issue_year er_cr_line_m er_cr_line_y  \
49162                     0         Aug       2013          Jun         2004

      state zipcode emp_length_yrs
49162    UT   22690              5
```

[ ]:
```
#Q1. 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
```

Insights:

Target variable distribution is 80%-20%.

Data is significantly imbalanced

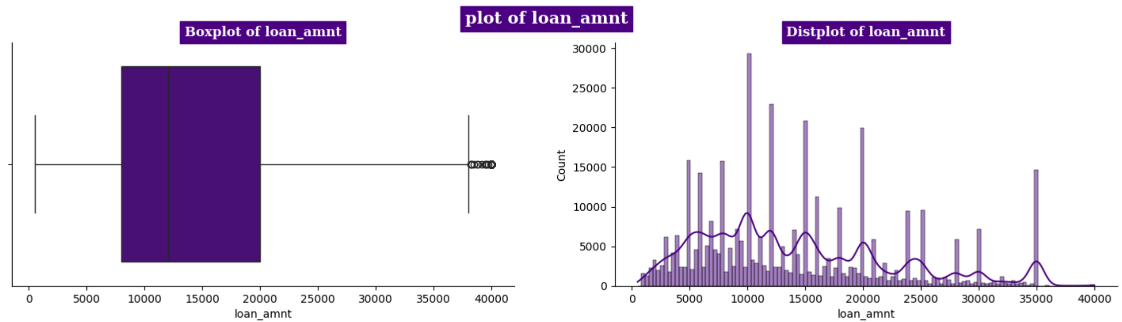
[ ]:
```
#Graphical Analysis:
```

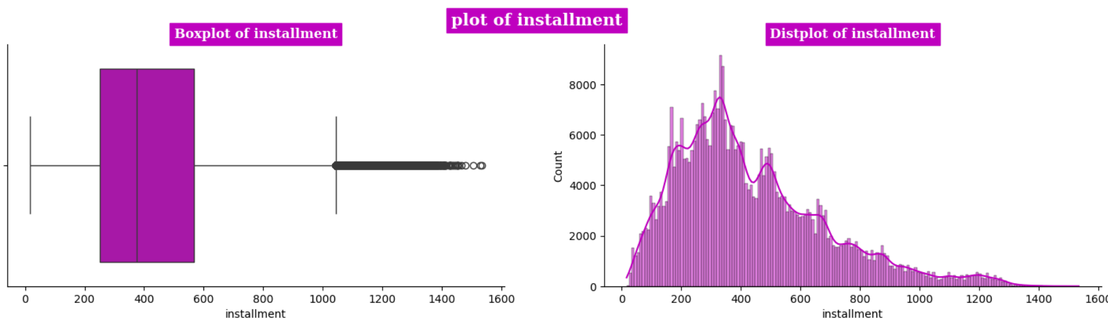
[ ]:
```
cp =␣
↪['indigo','m','darkviolet','magenta','mediumorchid','violet','purple','orchid','mediumpurpl
```

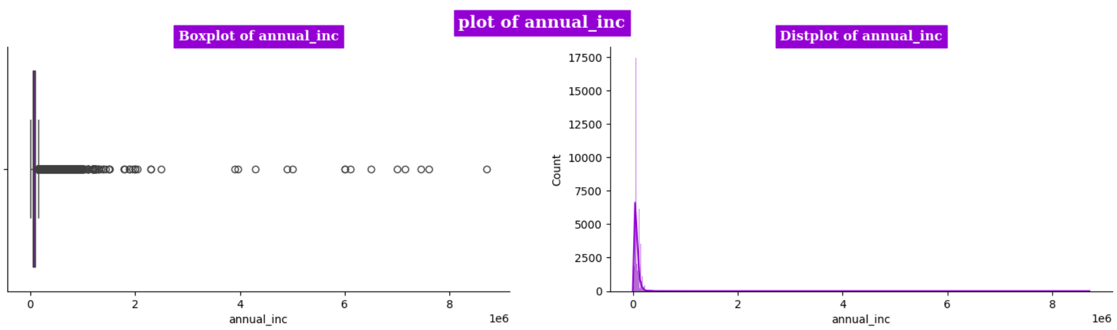
[ ]:
```
num_cols.iloc[:,[0,2,3,4,5,6,8,9,10]].sample()
```

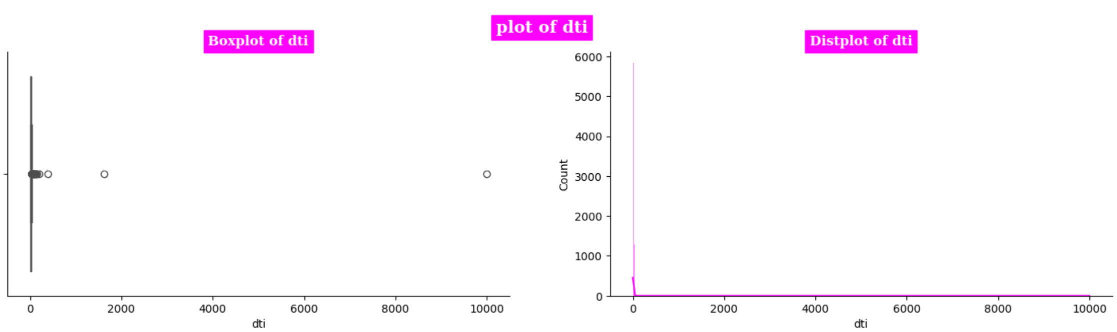
[ ]:
```
        loan_amnt  installment  annual_inc   dti  open_acc  pub_rec  \
377512    12000.0       375.49    115000.0  4.89      10.0        0

        revol_util  total_acc  mort_acc
377512        30.1       21.0         0
```

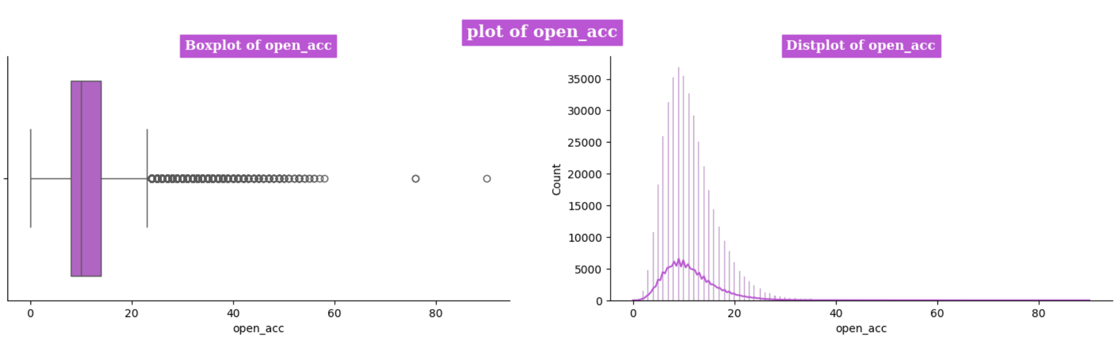
[ ]:
```
plt.style.use('default')
#plt.style.use('seaborn-bright')
outlier_graphical_cols = num_cols.iloc[:,[0,2,3,4,5,6,8,9,10]]
for _,col in enumerate(outlier_graphical_cols.columns):
    plt.figure(figsize=(18,4))
    plt.suptitle(f'plot of␣
↪{col}',fontsize=15,fontfamily='serif',fontweight='bold',backgroundcolor=cp[_],color='w')
    plt.subplot(121)
    sns.boxplot(x=df[col],color=cp[_])
    plt.title(f'Boxplot of␣
↪{col}',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[_],color='w')
    plt.subplot(122)
    sns.histplot(x=df[col], kde=True,color=cp[_])
    plt.title(f'Distplot of␣
↪{col}',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[_],color='w')
    sns.despine()
    plt.show()
```

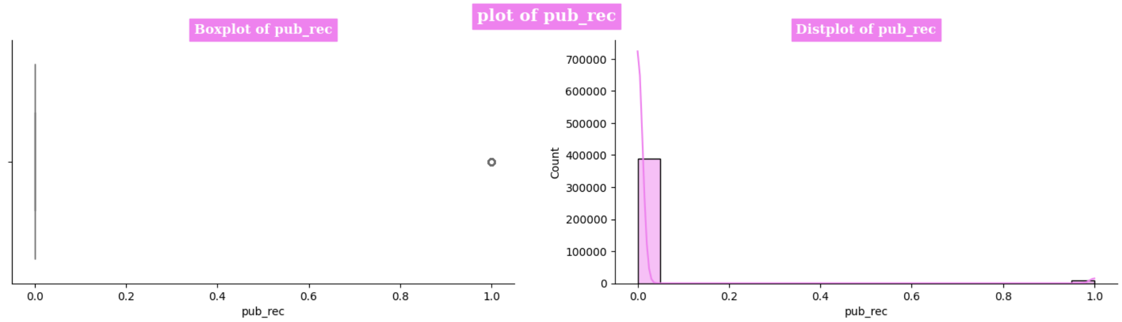


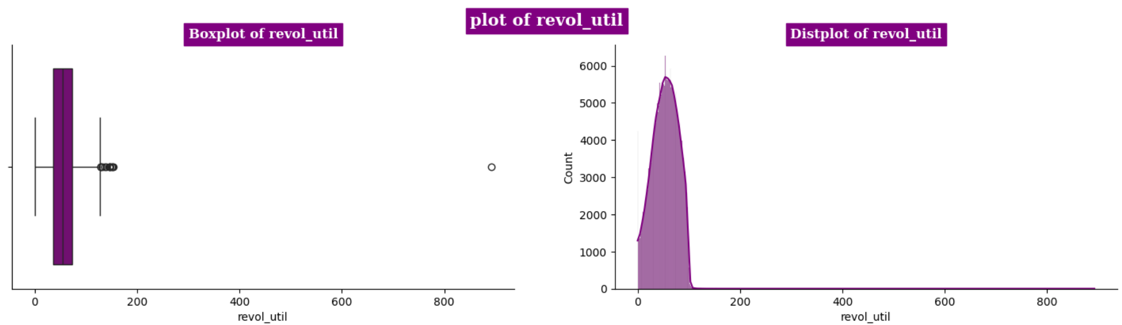

plot of installment
Boxplot of installment
0 200 400 600 800 1000 1200 1400 1600
installment
Distplot of installment
Count
0 2000 4000 6000 8000
0 200 400 600 800 1000 1200 1400 1600
installment

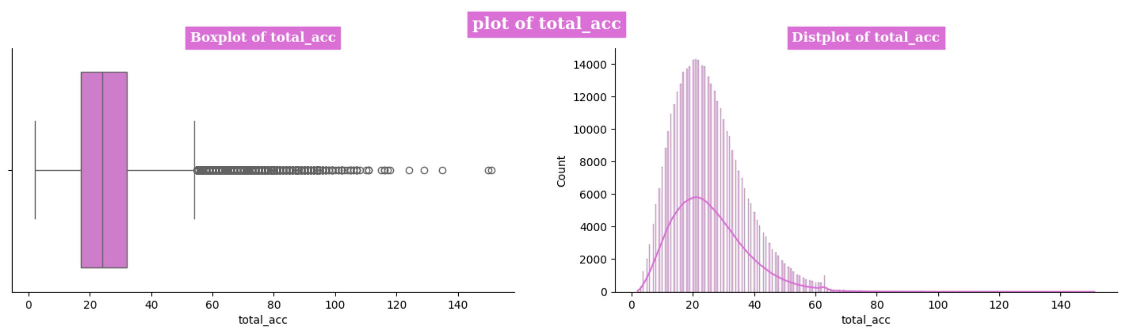

plot of annual_inc
Boxplot of annual_inc
0 2 4 6 8
annual_inc
1e6
Distplot of annual_inc
Count
0 2500 5000 7500 10000 12500 15000 17500
0 2 4 6 8
annual_inc
1e6

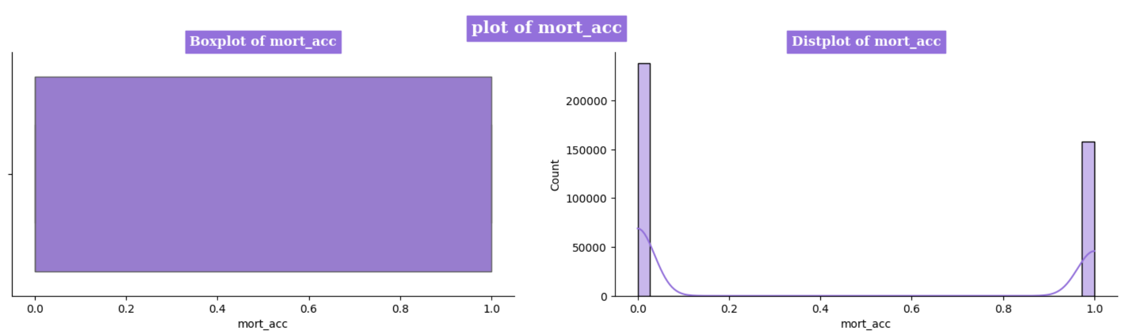

plot of dti
Boxplot of dti
0 2000 4000 6000 8000 10000
dti
Distplot of dti
Count
0 1000 2000 3000 4000 5000 6000
0 2000 4000 6000 8000 10000
dti


plot of open_acc
Boxplot of open_acc
0 20 40 60 80
open_acc
Distplot of open_acc
Count
0 5000 10000 15000 20000 25000 30000 35000
0 20 40 60 80
open_acc


plot of pub_rec
Boxplot of pub_rec
Distplot of pub_rec
pub_rec
Count


plot of revol_util
Boxplot of revol_util
Distplot of revol_util
revol_util
Count


plot of total_acc
Boxplot of total_acc
Distplot of total_acc
total_acc
Count


plot of mort_acc
Boxplot of mort_acc
Distplot of mort_acc
mort_acc
Count

Insights:

The analysis suggests a prevalence of outliers, prompting further investigation into outlier detection techniques.

Among the numerical features, Potential outliers may still be present.

Notably, features such as Pub_rec, Mort_acc, and Pub_rec_bankruptcies display a sparse distribution of unique values, indicating the potential benefit of generating binary features from these variables.

[ ]:
```
#Countplots of various categorical features w.r.t. to target variable␣
↪loan_status
plt.figure(figsize=(16,17))
plt.suptitle('Countplots of various categorical features w.r.t. to target␣
↪variable loan_status',
            ␣
↪fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor=cp[1],color='w')
plt.subplot(321)
sns.countplot(data=df, x='loan_status',palette=cp)
plt.title('Loan Status␣
↪Counts',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[2],color='w')
plt.subplot(322)
sns.countplot(data=df, x='loan_status', hue='term',palette=cp)
plt.title('Term wise loan status␣
↪count',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[3],color='w')
plt.subplot(323)
sns.countplot(data=df, x='home_ownership', hue='loan_status',palette=cp)
plt.title('Loan Status Vs Home␣
↪Ownership',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[4],color='w'
plt.subplot(324)
sns.countplot(data=df, x='verification_status', hue='loan_status',palette=cp)
plt.title('Loan Status Vs Verification␣
↪Status',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[5],color='w')
plt.subplot(325)
sns.countplot(data=df, x='issue_month', hue='loan_status',palette=cp)
plt.title('Loan Status Vs␣
↪issue_month',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[6],color='
plt.subplot(326)
sns.countplot(data=df, x='zipcode', hue='loan_status',palette=cp)
plt.title('Loan Status Vs␣
↪zipcode',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[7],color='w')
sns.despine()
plt.show()
```

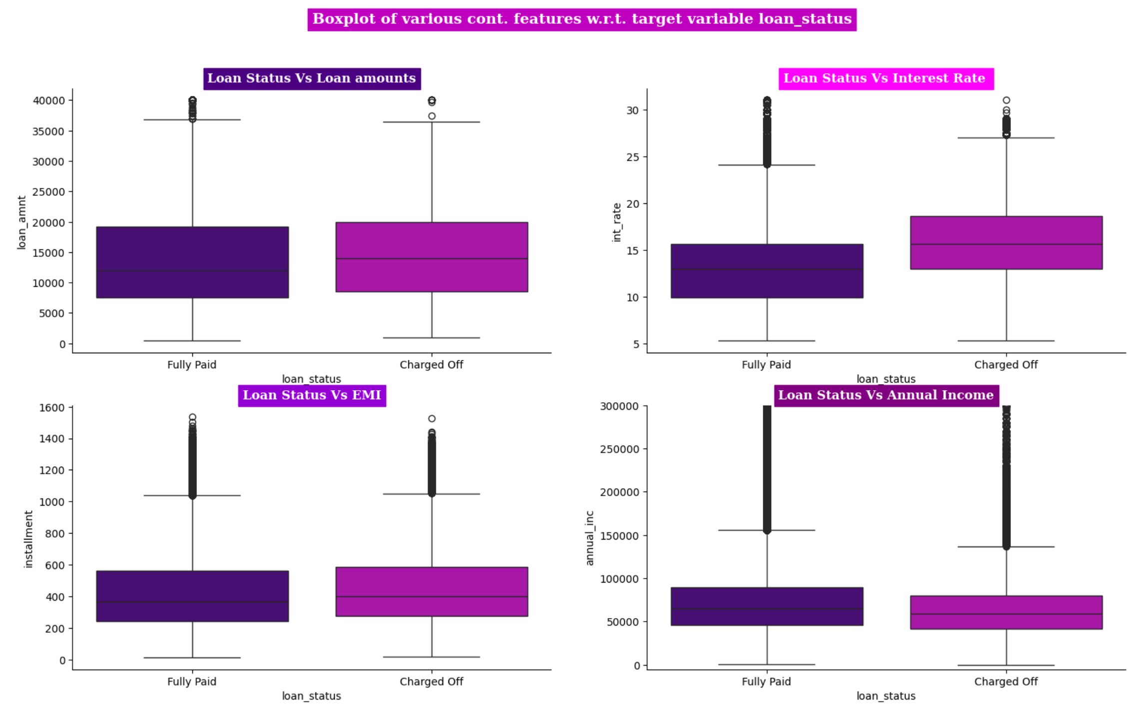

```
[ ]: zip_codes = ["11650", "86630", "93700"]
states = df[df['zipcode'].isin(zip_codes)]['state']

for zip_code, state in zip(zip_codes, states):
    print(f"Zip code: {zip_code}, State: {state}")
```

```
Zip code: 11650, State: VA
Zip code: 86630, State: MI
Zip code: 93700, State: MD
```

Observations:

It's been observed that loans haven't been completely repaid in zip codes 11650, 86630, and 93700. Loans haven't been repaid by borrowers residing in 'VA', 'MI', and 'MD'.

[ ]:
```
#Boxplot of various cont. features w.r.t. target variable loan_status
plt.figure(figsize=(18,10))
plt.suptitle('Boxplot of various cont. features w.r.t. target variable␣
 ↪loan_status',
                    ␣
 ↪fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor=cp[1],color='w')
plt.subplot(221)
sns.boxplot(data=df, x='loan_status', y='loan_amnt',palette=cp)
plt.title('Loan Status Vs Loan␣
 ↪amounts',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[0],color='w')
plt.subplot(222)
sns.boxplot(data=df, x='loan_status', y='int_rate',palette=cp)
plt.title('Loan Status Vs Interest Rate␣
 ↪',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[3],color='w')
plt.subplot(223)
sns.boxplot(data=df, x='loan_status', y='installment',palette=cp)
plt.title('Loan Status Vs␣
 ↪EMI',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[2],color='w')
plt.subplot(224)
sns.boxplot(data=df, x='loan_status', y='annual_inc',palette=cp)
plt.ylim(bottom=-5000, top=300000)
plt.title('Loan Status Vs Annual␣
 ↪Income',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[6],color='w')
sns.despine()
plt.show()
```



Observations:

Charged Off customers exhibit a notably higher median interest rate compared to Fully Paid customers.

The median annual income of Charged Off customers is lower than that of Fully Paid customers.

Charged Off customers tend to have a higher median EMI compared to Fully Paid customers.

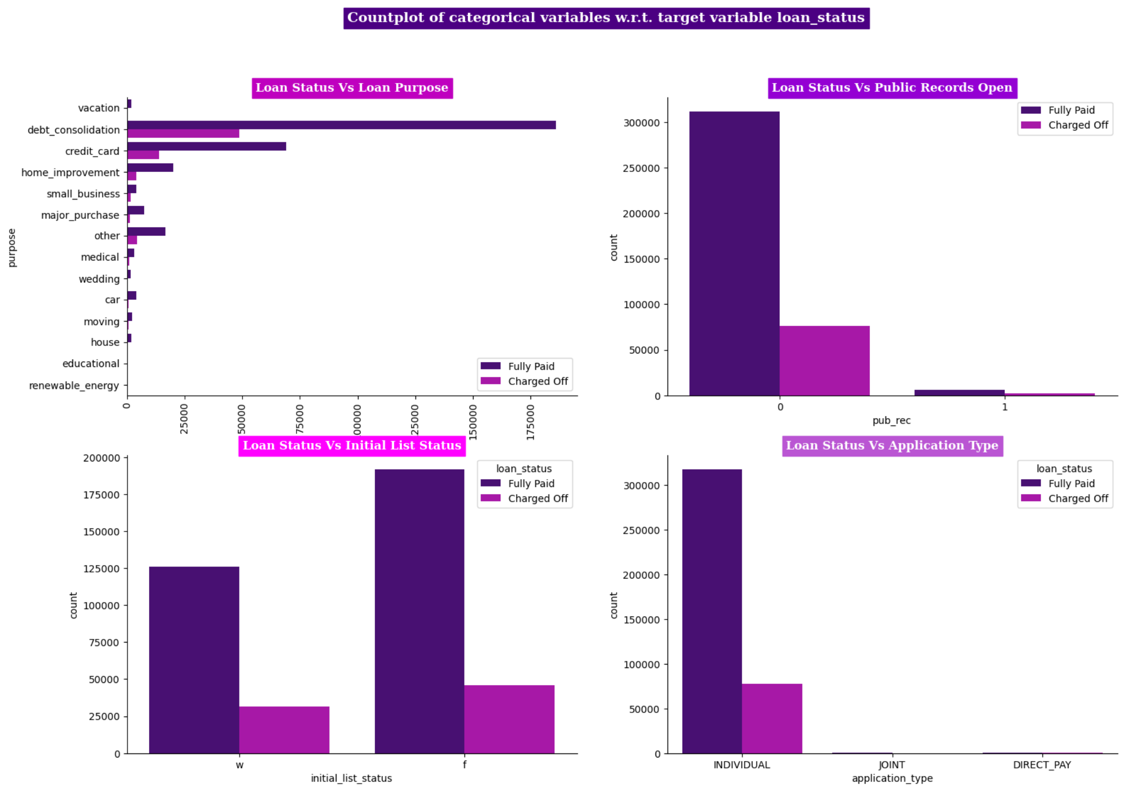
The median loan amount for Charged Off customers surpasses that of Fully Paid customers.

[ ]:
```
df.sample()
```

[ ]:
```
        loan_amnt term  int_rate  installment grade sub_grade  \
388782     7200.0   36      6.62       221.07     A        A2

                        emp_title home_ownership  annual_inc  \
388782  Michigan State University       MORTGAGE    182004.0

       verification_status loan_status           purpose             title  \
388782        Not Verified  Fully Paid  home_improvement  Home Improvement

          dti  open_acc  pub_rec  revol_bal  revol_util  total_acc  \
388782  10.28       7.0        0    71642.0        69.7       35.0

       initial_list_status application_type  mort_acc  pub_rec_bankruptcies  \
388782                   f       INDIVIDUAL         0                     0
```

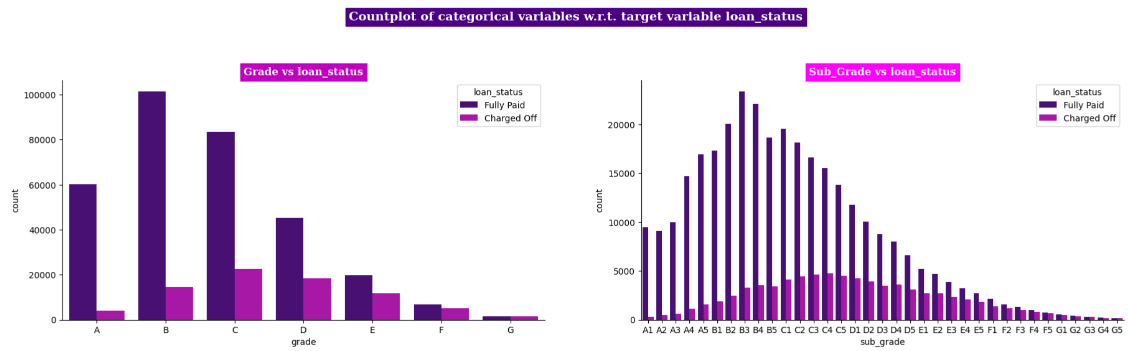
```
       issue_month issue_year er_cr_line_m er_cr_line_y state zipcode  \
388782         Sep       2011          Sep         1968    NM   05113

       emp_length_yrs
388782              2
```

```
[ ]: #Countplot of categorical variables w.r.t. target variable loan_status
plt.figure(figsize=(18,12))
plt.suptitle('Countplot of categorical variables w.r.t. target variable␣
 ↪loan_status',
           ␣
 ↪fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor=cp[0],color='w')
plt.subplot(221)
sns.countplot(data=df, y='purpose', hue='loan_status',palette=cp)
plt.xticks(rotation=90)
plt.title('Loan Status Vs Loan␣
 ↪Purpose',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[1],color='w')
plt.legend(loc=4)
plt.subplot(222)
sns.countplot(data=df, x='pub_rec',hue='loan_status',palette=cp)
plt.title('Loan Status Vs Public Records␣
 ↪Open',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[2],color='w')
plt.legend(loc=1)
plt.subplot(223)
sns.countplot(data=df, x='initial_list_status', hue='loan_status',palette=cp)
plt.title('Loan Status Vs Initial List␣
 ↪Status',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[3],color='w')
plt.subplot(224)
sns.countplot(data=df, x='application_type',hue='loan_status',palette=cp)
plt.title('Loan Status Vs Application␣
 ↪Type',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[4],color='w')
sns.despine()
plt.show()
```



[ ]:

```
plt.figure(figsize=(22,11))
plt.suptitle('Countplot of categorical variables w.r.t. target variable␣
 ↪loan_status',
                                                    ␣
 ↪fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor=cp[0],color='w')
plt.subplot(221)
grade = sorted(df.grade.unique().tolist())
sns.countplot(x='grade', data=df, hue='loan_status', order=grade,palette=cp)
plt.title('Grade vs␣
 ↪loan_status',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[1],color='
plt.subplot(222)
sub_grade = sorted(df.sub_grade.unique().tolist())
sns.countplot(x='sub_grade', data=df, hue='loan_status',␣
 ↪order=sub_grade,palette=cp)
plt.title('Sub_Grade vs␣
 ↪loan_status',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[3],color='
sns.despine()
plt.show()
```



Observations:

Top 2 loan purpose categories are Debit Consolidation and Credit Card.

Topmost loan type application is INDIVIDUAL.

The distribution of open_acc appears to be relatively normal when visualized graphically.

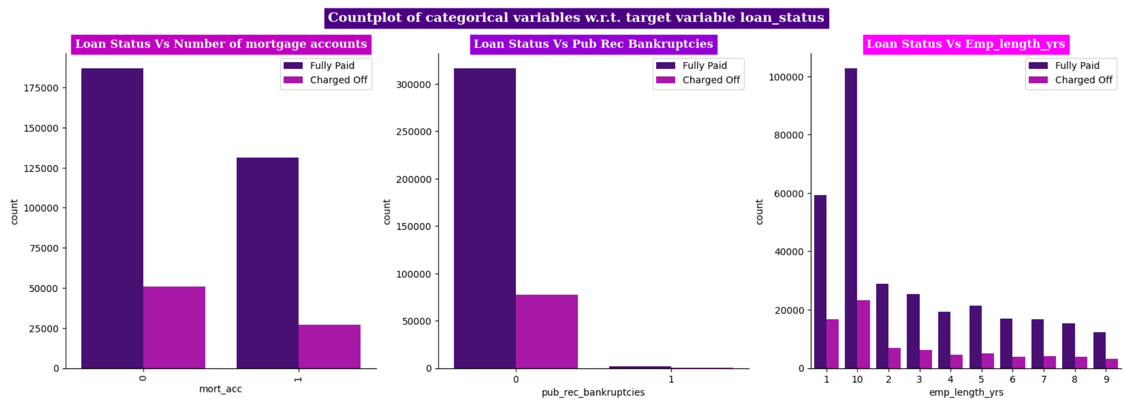
Charged Off and Fully Paid categories exhibit similar distributions.

[ ]:
```
df.sample()
```

[ ]:
```
        loan_amnt term  int_rate  installment grade sub_grade  \
180406    15000.0   36      16.2       528.84     C        C4

               emp_title home_ownership  annual_inc verification_status  \
180406  HESS Corporation            OWN     85000.0        Not Verified

        loan_status           purpose        title    dti  open_acc  pub_rec  \
180406  Charged Off  home_improvement  HomeImprove  26.29      12.0        0

        revol_bal  revol_util  total_acc initial_list_status application_type  \
180406    14223.0        24.2       17.0                   f       INDIVIDUAL

        mort_acc  pub_rec_bankruptcies issue_month issue_year er_cr_line_m  \
180406         0                     0         Sep       2013          Sep

       er_cr_line_y state zipcode emp_length_yrs
180406         2002    ME   93700              3
```

[ ]:
```
#Countplot for various categorical features w.r.t. target variable loan_status

plt.figure(figsize=(20,6))
plt.suptitle('Countplot of categorical variables w.r.t. target variable␣
 ↪loan_status',
           ␣
 ↪fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor=cp[0],color='w')
```

```
plt.subplot(131)
sns.countplot(data=df, x='mort_acc',hue='loan_status',palette=cp)
plt.xticks(rotation=90)
plt.title('Loan Status Vs Number of mortgage␣
↪accounts',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[1],color='w')
plt.legend(loc=1)
plt.subplot(132)
sns.countplot(data=df, x='pub_rec_bankruptcies',hue='loan_status',palette=cp)
plt.title('Loan Status Vs Pub Rec␣
↪Bankruptcies',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[2],color=
plt.legend(loc=1)
plt.subplot(133)
order = sorted(df.emp_length_yrs.unique().tolist())
sns.countplot(data=df,␣
↪x='emp_length_yrs',hue='loan_status',order=order,palette=cp)
plt.title('Loan Status Vs␣
↪Emp_length_yrs',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[3],colo
plt.legend(loc=1)
sns.despine()
plt.show()
```



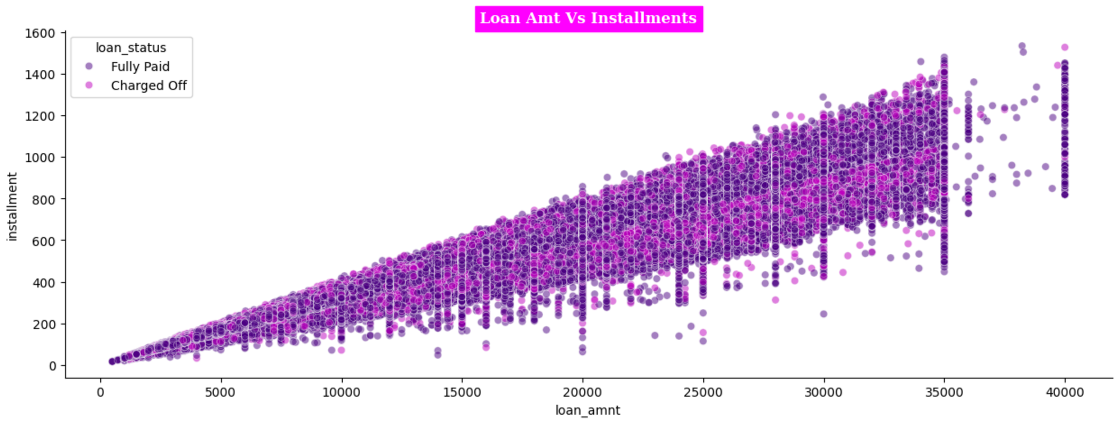
[ ]:
```
#Q2. Comment about the correlation between Loan Amount and Installment features.
df[['loan_amnt', 'installment']].corr()
```

[ ]:

	loan_amnt	installment
loan_amnt	1.000000	0.953929
installment	0.953929	1.000000

[ ]:
```
plt.figure(figsize = (15,5))
sns.scatterplot(data = df, x = 'loan_amnt', y = 'installment', alpha = 0.5, hue␣
↪= 'loan_status', palette = cp)
```

```
plt.title('Loan Amt Vs␣
 ↪Installments',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[3],color=
sns.despine()
plt.show()
```



Insights:

The correlation coefficient measures the strength and direction of the linear relationship between two variables. In this case, the correlation coefficient between 'loan_amnt' and 'installment' is quite high, approximately 0.95, indicating a strong positive linear relationship between these two variables.

Loan Terms:

Understanding the relationship between loan amount and installment payments is crucial for setting appropriate loan terms. Lenders can adjust loan terms such as interest rates and repayment periods based on the borrower's ability to handle installment payments associated with different loan amounts.

Potential Multicollinearity:

When building predictive models, it's essential to be cautious of multicollinearity between highly correlated predictor variables. Multicollinearity can lead to unstable estimates and difficulties in interpreting the model coefficients. Therefore, it might be necessary to address multicollinearity through techniques such as variable selection or regularization.

[ ]:
```
#Q3. The majority of people have home ownership as _______.
(df['home_ownership'].value_counts(normalize=True)*100).to_frame()
```

[ ]:

	proportion
home_ownership	
MORTGAGE	50.084085
RENT	40.347953
OWN	9.531096
OTHER	0.028281

```
NONE            0.007828
ANY             0.000758
```

Insights:

Mortgage holders comprise the majority with approximately 50.08%, indicating that a significant portion of individuals own homes through Mortgage agreements.

Renters constitute a substantial portion, accounting for around 40.35% of home ownership types.This suggests a sizable demographic of individuals who opt for renting rather than owning a home.

[ ]:
```
#Q4. People with grades 'A' are more likely to fully pay their loan. (T/F)
pd.crosstab(df['grade'],df['loan_status'], normalize = 'index')
```

[ ]:
```
loan_status  Charged Off  Fully Paid
grade
A               0.062879    0.937121
B               0.125730    0.874270
C               0.211809    0.788191
D               0.288678    0.711322
E               0.373634    0.626366
F               0.427880    0.572120
G               0.478389    0.521611
```

Insights:

True. Grade 'A' borrowers demonstrate a significantly high likelihood of fully repaying their loans, with approximately 93.71% of loans being fully paid. This suggests that borrowers with the highest credit rating are more inclined to fulfill their loan obligations successfully.

The proportion of charged-off loans for grade 'A' borrowers is relatively low, standing at approximately 6.29%. This indicates a low default rate among borrowers with the highest credit rating, emphasizing their creditworthiness and reliability in loan repayment.

[ ]:
```
#Q5. Name the top 2 afforded job titles.
df[df['emp_title'] != 'No Employee Title']['emp_title'].value_counts().
 ↪to_frame().head()
```

[ ]:
```
                  count
emp_title
Teacher            4389
Manager            4250
Registered Nurse   1856
RN                 1846
Supervisor         1830
```

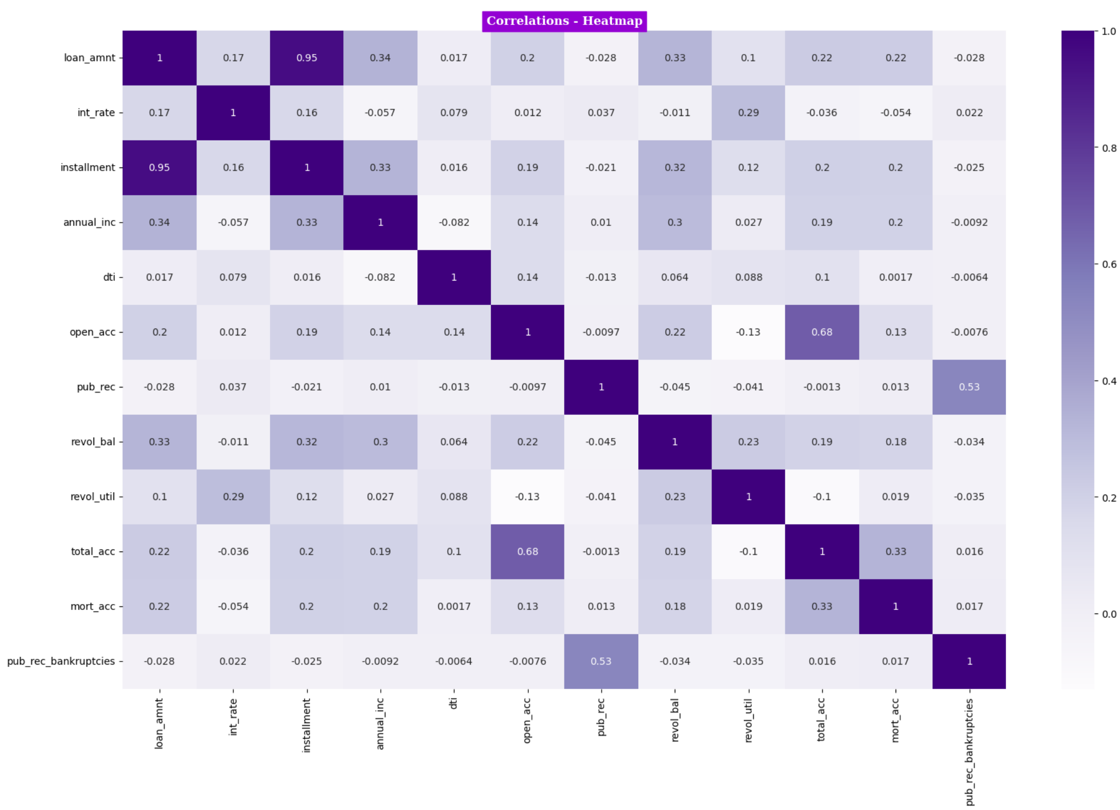
[ ]:
```
df.groupby('emp_title')['loan_status'].count().sort_values(ascending=False).
 ↪to_frame()[1:6]
```

```
                     loan_status
emp_title
Teacher                     4389
Manager                     4250
Registered Nurse            1856
RN                          1846
Supervisor                  1830
```

Insights:

The Most afforded job titles are Teachers & Managers.

```
plt.figure(figsize=(20,12))
sns.heatmap(num_cols.corr(), annot=True, cmap='Purples')
plt.title('Correlations - Heatmap',fontsize=12,fontfamily='serif',fontweight='bold',backgroundcolor=cp[2],color='w')
plt.show()
```



Observations:

There exists a strong correlation between loan_amnt and installment, indicating that higher loan amounts correspond to larger installment payments.

The variables total_acc and open_acc exhibit a significant correlation.

There is a notable correlation between pub_rec_bankruptcies and pub_rec.

[ ]:
```
#Outlier Treatment:
numerical_cols = df.select_dtypes(include=np.number).columns
numerical_cols
```

[ ]:
```
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')
```

[ ]:
```
# outlier treatment
def remove_outliers_zscore(df, threshold=2): #(considering 2 std.dev away from␣
 ↪mean approx 95% of data)
    """
    Remove outliers from a DataFrame using the Z-score method.

    Parameters:
        df (DataFrame): The input DataFrame.
        threshold (float): The Z-score threshold for identifying outliers.
                           Observations with a Z-score greater than this␣
 ↪threshold
                           will be considered as outliers.

    Returns:
        DataFrame: The DataFrame with outliers removed.
    """
    # Calculate Z-scores for numerical columns
    z_scores = (df[numerical_cols] - df[numerical_cols].mean()) /␣
 ↪df[numerical_cols].std()

    # Identify outliers
    outliers = np.abs(z_scores) > threshold

    # Keep non-outliers for numerical columns
    df_cleaned = df[~outliers.any(axis=1)]

    return df_cleaned
cleaned_df = remove_outliers_zscore(df1)
print(cleaned_df.shape)
```

```
(311392, 30)
```

[ ]:
```
def clip_outliers_zscore(df, threshold=2):
    """
    Clip outliers in a DataFrame using the Z-score method.
```

```
    Parameters:
        df (DataFrame): The input DataFrame.
        threshold (float): The Z-score threshold for identifying outliers.
                           Observations with a Z-score greater than this␣
↪threshold
                           will be considered as outliers.

    Returns:
        DataFrame: The DataFrame with outliers clipped.
    """
    # Calculate Z-scores for numerical columns
    z_scores = (df[numerical_cols] - df[numerical_cols].mean()) /␣
↪df[numerical_cols].std()

    # Clip outliers
    clipped_values = df[numerical_cols].clip(df[numerical_cols].mean() -␣
↪threshold * df[numerical_cols].std(),
                                             df[numerical_cols].mean() +␣
↪threshold * df[numerical_cols].std(),
                                             axis=1)

    # Assign clipped values to original DataFrame
    df_clipped = df.copy()
    df_clipped[numerical_cols] = clipped_values

    return df_clipped

clipped_df = clip_outliers_zscore(df1)
print(clipped_df.shape)
```

```
(396030, 30)
```

[ ]:
```
data = cleaned_df.copy()
cp_data = clipped_df.copy()
data.sample()
```

[ ]:
```
        loan_amnt term  int_rate  installment grade sub_grade  \
395673     7000.0   36     13.67       238.13     C        C4

                 emp_title home_ownership  annual_inc verification_status  \
395673  No Employee Title           RENT     53000.0            Verified

       loan_status             purpose              title  dti  open_acc  \
395673  Fully Paid  debt_consolidation  Debt consolidation  5.5       6.0

        pub_rec  revol_bal  revol_util  total_acc initial_list_status  \
395673        0     6776.0        81.6       13.0                   w
```

```
       application_type  mort_acc  pub_rec_bankruptcies issue_month  \
395673       INDIVIDUAL         0                     0         Nov

       issue_year er_cr_line_m er_cr_line_y state zipcode emp_length_yrs
395673       2015          Nov         2002    OK   05113              1
```

[ ]:
```
data['pub_rec_bankruptcies'].value_counts() , data['pub_rec'].value_counts()
```

[ ]:
```
(pub_rec_bankruptcies
 0    311392
 Name: count, dtype: int64,
 pub_rec
 0    311392
 Name: count, dtype: int64)
```

[ ]:
```
cp_data['pub_rec_bankruptcies'].value_counts() , cp_data['pub_rec'].
 ↪value_counts()
```

[ ]:
```
(pub_rec_bankruptcies
 0.000000    393705
 0.158662      2325
 Name: count, dtype: int64,
 pub_rec
 0.000000    388011
 0.301947      8019
 Name: count, dtype: int64)
```

[ ]:
```
data.shape
```

[ ]:
```
(311392, 30)
```

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

```
<class 'pandas.core.frame.DataFrame'>
Index: 311392 entries, 0 to 396029
Data columns (total 30 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   loan_amnt             311392 non-null  float64
 1   term                  311392 non-null  object
 2   int_rate              311392 non-null  float64
 3   installment           311392 non-null  float64
 4   grade                 311392 non-null  object
 5   sub_grade             311392 non-null  object
 6   emp_title             311392 non-null  object
 7   home_ownership        311392 non-null  object
```

```
 8   annual_inc            311392 non-null  float64
 9   verification_status   311392 non-null  object
 10  loan_status           311392 non-null  object
 11  purpose               311392 non-null  object
 12  title                 311392 non-null  object
 13  dti                   311392 non-null  float64
 14  open_acc              311392 non-null  float64
 15  pub_rec               311392 non-null  int64
 16  revol_bal             311392 non-null  float64
 17  revol_util            311392 non-null  float64
 18  total_acc             311392 non-null  float64
 19  initial_list_status   311392 non-null  object
 20  application_type      311392 non-null  object
 21  mort_acc              311392 non-null  int64
 22  pub_rec_bankruptcies  311392 non-null  int64
 23  issue_month           311392 non-null  object
 24  issue_year            311392 non-null  object
 25  er_cr_line_m          311392 non-null  object
 26  er_cr_line_y          311392 non-null  object
 27  state                 311392 non-null  object
 28  zipcode               311392 non-null  object
 29  emp_length_yrs        311392 non-null  object
dtypes: float64(9), int64(3), object(18)
memory usage: 73.6+ MB
```

[ ]:
```
#Manual encoding
data['loan_status']=data.loan_status.map({'Fully Paid':1, 'Charged Off':0})

data['initial_list_status']=data.initial_list_status.map({'w':0, 'f':1})
```

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

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

                 emp_title home_ownership  annual_inc verification_status  \
0                Marketing           RENT    117000.0        Not Verified
1          Credit analyst        MORTGAGE     65000.0        Not Verified
2             Statistician           RENT     43057.0     Source Verified
3           Client Advocate          RENT     54000.0        Not Verified
4  Destiny Management Inc.       MORTGAGE     55000.0            Verified

   loan_status               purpose                     title    dti  open_acc  \
```

```
0             1           vacation                 Vacation  26.24      16.0
1             1 debt_consolidation       Debt consolidation  22.05      17.0
2             1        credit_card  Credit card refinancing  12.79      13.0
3             1        credit_card  Credit card refinancing   2.60       6.0
4             0        credit_card    Credit Card Refinance  33.95      13.0

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

  application_type  mort_acc  pub_rec_bankruptcies issue_month issue_year  \
0       INDIVIDUAL         0                     0         Jan       2015
1       INDIVIDUAL         1                     0         Jan       2015
2       INDIVIDUAL         0                     0         Jan       2015
3       INDIVIDUAL         0                     0         Nov       2014
4       INDIVIDUAL         0                     0         Apr       2013

  er_cr_line_m er_cr_line_y state zipcode emp_length_yrs
0          Jun         1990    OK   22690             10
1          Jul         2004    SD   05113              4
2          Aug         2007    WV   05113              1
3          Sep         2006    MA   00813              6
4          Mar         1999    VA   11650              9
```

[ ]:
```
#Feature selection - done by hypothesis testing & VIF(multicolinearity)

#Find VIF after modelling and remove features with high VIF (>5):
```

[ ]:
```
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
```

[ ]:
```
cat_cols = data.select_dtypes(include=['object']).columns.tolist()
for col in cat_cols:
    chi2, p, dof, expected = chi2_contingency(pd.crosstab(data[col],␣
 ↪data['loan_status']))
    if p > 0.05:
```

```
        print('>>>>>>> Independent feature - Not Significant:',col,' >> p value:
 ↪',p)
```

```
>>>>>>> Independent feature - Not Significant: emp_title  >> p value:
0.5367121560200798
>>>>>>> Independent feature - Not Significant: title  >> p value: 1.0
>>>>>>> Independent feature - Not Significant: er_cr_line_m  >> p value:
0.2722117086158036
>>>>>>> Independent feature - Not Significant: state  >> p value:
0.76047808977373
```

[ ]:
```
## dropping cols based on correlation(heatmap,hypothesis testing)
lt = data.
 ↪drop(columns=['emp_title','title','sub_grade','er_cr_line_m','er_cr_line_y','initial_list_s
                          ␣
 ↪'state','issue_month','issue_year','pub_rec','pub_rec_bankruptcies'],axis=1)
lt.shape
```

[ ]: (311392, 19)

[ ]:
```
lt.sample()
```

[ ]:
```
        loan_amnt term  int_rate  installment grade home_ownership  \
350767     5000.0   36     15.31       174.09     C           RENT

        annual_inc verification_status  loan_status            purpose   dti  \
350767     37000.0        Not Verified            0  debt_consolidation  7.14

        open_acc  revol_bal  revol_util  total_acc application_type  mort_acc  \
350767      10.0     6215.0        53.9       10.0       INDIVIDUAL         0

       zipcode emp_length_yrs
350767   48052             10
```

[ ]:
```
#### Performing OneHotEncoding on feature having multiple variable
dummies=['zipcode',␣
 ↪'grade','purpose','home_ownership','verification_status','application_type']
ltd = pd.get_dummies(lt, columns=dummies, drop_first=True)*1
```

[ ]:
```
ltd.shape
```

[ ]: (311392, 50)

[ ]:
```
ltd.dtypes
```

[ ]:
```
loan_amnt                             float64
term                                   object
```

```
int_rate                                float64
installment                             float64
annual_inc                              float64
loan_status                               int64
dti                                     float64
open_acc                                float64
revol_bal                               float64
revol_util                              float64
total_acc                               float64
mort_acc                                  int64
emp_length_yrs                           object
zipcode_05113                             int64
zipcode_11650                             int64
zipcode_22690                             int64
zipcode_29597                             int64
zipcode_30723                             int64
zipcode_48052                             int64
zipcode_70466                             int64
zipcode_86630                             int64
zipcode_93700                             int64
grade_B                                   int64
grade_C                                   int64
grade_D                                   int64
grade_E                                   int64
grade_F                                   int64
grade_G                                   int64
purpose_credit_card                       int64
purpose_debt_consolidation                int64
purpose_educational                       int64
purpose_home_improvement                  int64
purpose_house                             int64
purpose_major_purchase                    int64
purpose_medical                           int64
purpose_moving                            int64
purpose_other                             int64
purpose_renewable_energy                  int64
purpose_small_business                    int64
purpose_vacation                          int64
purpose_wedding                           int64
home_ownership_MORTGAGE                   int64
home_ownership_NONE                       int64
home_ownership_OTHER                      int64
home_ownership_OWN                        int64
home_ownership_RENT                       int64
verification_status_Source Verified       int64
verification_status_Verified              int64
application_type_INDIVIDUAL               int64
```

```
application_type_JOINT                    int64
dtype: object
```

[ ]:
```
ltd.sample(8)
```

[ ]:

	loan_amnt	term	int_rate	installment	annual_inc	loan_status	dti
109263	5500.0	36	14.74	189.96	30000.00	1	16.04
76290	10000.0	60	11.11	217.98	43000.00	1	15.68
338394	10000.0	36	11.99	332.10	51933.84	1	6.26
167192	19700.0	60	17.27	492.47	55000.00	1	11.96
394687	7200.0	36	19.52	265.83	41000.00	1	3.37
251990	2500.0	36	18.25	90.70	29000.00	1	20.36
273994	20000.0	36	13.68	680.45	82000.00	1	6.63
108548	11000.0	36	10.99	360.08	137000.00	1	26.46

	open_acc	revol_bal	revol_util	total_acc	mort_acc	emp_length_yrs
109263	6.0	2640.0	80.0	9.0	0	4
76290	8.0	8472.0	71.2	26.0	0	9
338394	8.0	7373.0	45.2	17.0	1	1
167192	6.0	12579.0	66.9	17.0	1	10
394687	10.0	4287.0	22.0	16.0	0	2
251990	7.0	3107.0	32.7	13.0	0	10
273994	10.0	15810.0	48.9	18.0	1	10
108548	7.0	24584.0	98.3	22.0	1	10

	zipcode_05113	zipcode_11650	zipcode_22690	zipcode_29597
109263	0	0	0	1
76290	0	0	0	0
338394	1	0	0	0
167192	0	0	0	0
394687	0	0	0	0
251990	0	0	0	0
273994	0	0	0	0
108548	0	0	0	1

	zipcode_30723	zipcode_48052	zipcode_70466	zipcode_86630
109263	0	0	0	0
76290	0	0	1	0
338394	0	0	0	0
167192	0	0	0	0
394687	0	0	1	0
251990	1	0	0	0
273994	0	1	0	0
108548	0	0	0	0

	zipcode_93700	grade_B	grade_C	grade_D	grade_E	grade_F	grade_G
109263	0	0	0	1	0	0	0

76290	0	1	0	0	0	0	0
338394	0	1	0	0	0	0	0
167192	0	0	1	0	0	0	0
394687	0	0	0	1	0	0	0
251990	0	0	0	0	1	0	0
273994	0	0	1	0	0	0	0
108548	0	1	0	0	0	0	0

	purpose_credit_card	purpose_debt_consolidation	purpose_educational \
109263	0	1	0
76290	0	0	0
338394	0	1	0
167192	0	1	0
394687	0	1	0
251990	0	1	0
273994	0	1	0
108548	0	0	0

	purpose_home_improvement	purpose_house	purpose_major_purchase \
109263	0	0	0
76290	0	0	0
338394	0	0	0
167192	0	0	0
394687	0	0	0
251990	0	0	0
273994	0	0	0
108548	0	0	0

	purpose_medical	purpose_moving	purpose_other \
109263	0	0	0
76290	0	0	0
338394	0	0	0
167192	0	0	0
394687	0	0	0
251990	0	0	0
273994	0	0	0
108548	0	0	0

	purpose_renewable_energy	purpose_small_business	purpose_vacation \
109263	0	0	0
76290	1	0	0
338394	0	0	0
167192	0	0	0
394687	0	0	0
251990	0	0	0
273994	0	0	0
108548	0	0	0

```
        purpose_wedding  home_ownership_MORTGAGE  home_ownership_NONE  \
109263                0                        0                    0
76290                 0                        1                    0
338394                0                        1                    0
167192                0                        1                    0
394687                0                        1                    0
251990                0                        1                    0
273994                0                        1                    0
108548                0                        1                    0

        home_ownership_OTHER  home_ownership_OWN  home_ownership_RENT  \
109263                     0                   0                    1
76290                      0                   0                    0
338394                     0                   0                    0
167192                     0                   0                    0
394687                     0                   0                    0
251990                     0                   0                    0
273994                     0                   0                    0
108548                     0                   0                    0

        verification_status_Source Verified  verification_status_Verified  \
109263                                    0                             0
76290                                     1                             0
338394                                    0                             1
167192                                    0                             1
394687                                    0                             0
251990                                    1                             0
273994                                    1                             0
108548                                    0                             0

        application_type_INDIVIDUAL  application_type_JOINT
109263                            1                       0
76290                             1                       0
338394                            1                       0
167192                            1                       0
394687                            1                       0
251990                            1                       0
273994                            1                       0
108548                            1                       0
```

[ ]:

```
#Model:

#Prepare X and y dataset i.e. independent and dependent datasets

X = ltd.drop(['loan_status'], axis=1)
y = ltd['loan_status']
```

[ ]:
```
#Split the data into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.
 ↪2,stratify=y,random_state=42)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(249113, 49)
(62279, 49)
(249113,)
(62279,)
```

[ ]:
```
#Minmax scaling the data
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
X_train = pd.DataFrame(X_train, columns=X.columns)
X_test = pd.DataFrame(X_test, columns=X.columns)
```

[ ]:
```
X_train.head()
```

[ ]:

	loan_amnt	term	int_rate	installment	annual_inc	dti	open_acc	\
0	0.379538	0.0	0.339161	0.411590	0.207250	0.465341	0.368421	
1	0.643564	1.0	0.680070	0.524221	0.367868	0.252652	0.473684	
2	0.168317	0.0	0.208625	0.176198	0.134712	0.357576	0.368421	
3	0.379538	1.0	0.680070	0.307444	0.367868	0.449242	0.315789	
4	0.368812	0.0	0.543706	0.421460	0.246109	0.315530	0.263158	

	revol_bal	revol_util	total_acc	mort_acc	emp_length_yrs	zipcode_05113	\
0	0.171897	0.419816	0.276596	0.0	0.111111	0.0	
1	0.221905	0.590398	0.340426	0.0	1.000000	0.0	
2	0.052236	0.304392	0.212766	0.0	0.000000	0.0	
3	0.255109	0.767109	0.297872	1.0	1.000000	0.0	
4	0.090649	0.614913	0.361702	0.0	0.000000	1.0	

	zipcode_11650	zipcode_22690	zipcode_29597	zipcode_30723	zipcode_48052	\
0	0.0	0.0	0.0	0.0	0.0	
1	0.0	0.0	1.0	0.0	0.0	
2	1.0	0.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	1.0	0.0	
4	0.0	0.0	0.0	0.0	0.0	

	zipcode_70466	zipcode_86630	zipcode_93700	grade_B	grade_C	grade_D	\
0	0.0	0.0	0.0	1.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	0.0	1.0	
2	0.0	0.0	0.0	0.0	0.0	0.0	

```
3           0.0            0.0            0.0      0.0      0.0      1.0
4           0.0            0.0            0.0      0.0      1.0      0.0

   grade_E  grade_F  grade_G  purpose_credit_card  purpose_debt_consolidation  \
0      0.0      0.0      0.0                  0.0                         1.0
1      0.0      0.0      0.0                  0.0                         1.0
2      0.0      0.0      0.0                  1.0                         0.0
3      0.0      0.0      0.0                  0.0                         0.0
4      0.0      0.0      0.0                  0.0                         1.0

   purpose_educational  purpose_home_improvement  purpose_house  \
0                  0.0                       0.0            0.0
1                  0.0                       0.0            0.0
2                  0.0                       0.0            0.0
3                  0.0                       0.0            0.0
4                  0.0                       0.0            0.0

   purpose_major_purchase  purpose_medical  purpose_moving  purpose_other  \
0                     0.0              0.0             0.0            0.0
1                     0.0              0.0             0.0            0.0
2                     0.0              0.0             0.0            0.0
3                     0.0              0.0             0.0            0.0
4                     0.0              0.0             0.0            0.0

   purpose_renewable_energy  purpose_small_business  purpose_vacation  \
0                       0.0                     0.0               0.0
1                       0.0                     0.0               0.0
2                       0.0                     0.0               0.0
3                       0.0                     1.0               0.0
4                       0.0                     0.0               0.0

   purpose_wedding  home_ownership_MORTGAGE  home_ownership_NONE  \
0              0.0                      0.0                  0.0
1              0.0                      1.0                  0.0
2              0.0                      0.0                  0.0
3              0.0                      0.0                  0.0
4              0.0                      0.0                  0.0

   home_ownership_OTHER  home_ownership_OWN  home_ownership_RENT  \
0                   0.0                 0.0                 1.0
1                   0.0                 0.0                 0.0
2                   0.0                 0.0                 1.0
3                   0.0                 0.0                 1.0
4                   0.0                 0.0                 1.0

   verification_status_Source Verified  verification_status_Verified  \
0                                  0.0                           0.0
```

```
1                                  1.0                            0.0
2                                  0.0                            0.0
3                                  0.0                            0.0
4                                  0.0                            0.0

   application_type_INDIVIDUAL  application_type_JOINT
0                          1.0                     0.0
1                          1.0                     0.0
2                          1.0                     0.0
3                          1.0                     0.0
4                          1.0                     0.0
```

[ ]:
```
#Model-1
#Fit the Model on training data
logreg_model = LogisticRegression()
logreg_model.fit(X_train, y_train)
```

[ ]:
```
LogisticRegression()
```

[ ]:
```
#Predit the data on test dataset
y_train_pred = logreg_model.predict(X_train)
y_test_pred = logreg_model.predict(X_test)
```

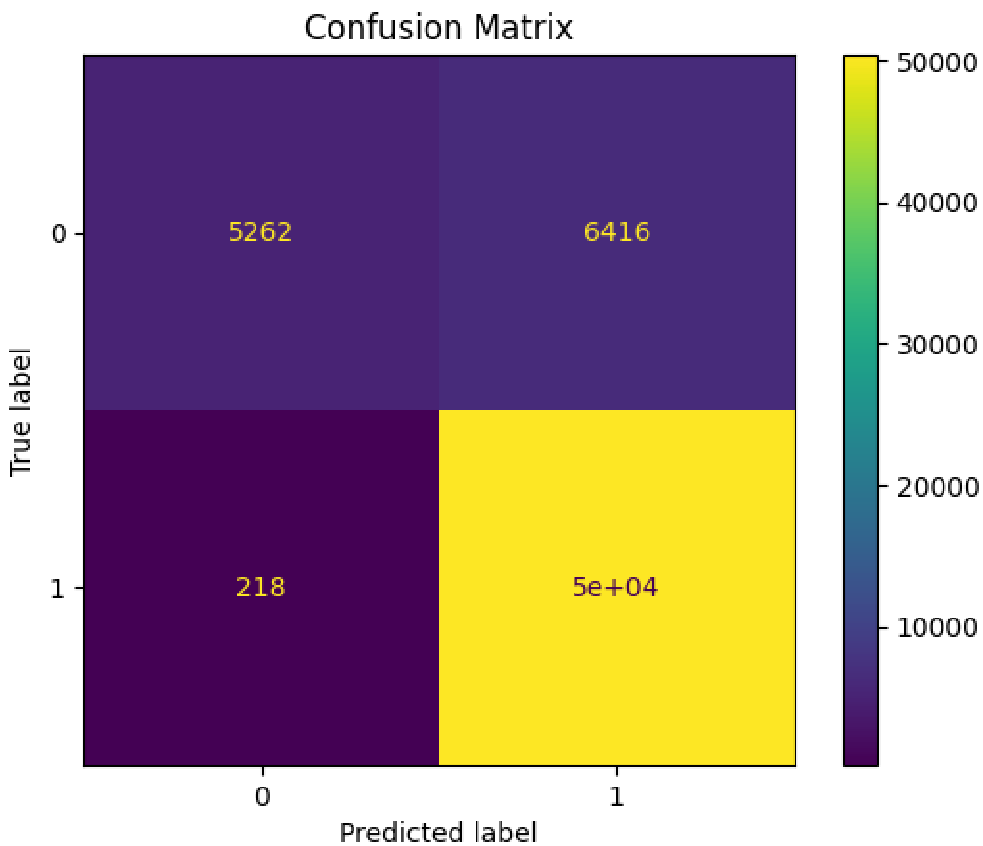
[ ]:
```
logreg_model.score(X_test, y_test) , logreg_model.score(X_test, y_test_pred)
#If logreg_model.score(X_test, y_test) consistently returns 1, it would imply␣
↪that your model is predicting the test set perfectly,
#which could be a sign of overfitting, data leakage, or an issue with the␣
↪evaluation process.
```

[ ]:
```
(0.8934793429566948, 1.0)
```

[ ]:
```
#Model Evaluation
print('Train Accuracy :', round(logreg_model.score(X_train, y_train),2))
print('Train F1 Score:',round(f1_score(y_train,y_train_pred),2))
print('Train Recall Score:',round(recall_score(y_train,y_train_pred),2))
print('Train Precision Score:',round(precision_score(y_train,y_train_pred),2))

print('\nTest Accuracy :',round(logreg_model.score(X_test,y_test),2))
print('Test F1 Score:',round(f1_score(y_test,y_test_pred),2))
print('Test Recall Score:',round(recall_score(y_test,y_test_pred),2))
print('Test Precision Score:',round(precision_score(y_test,y_test_pred),2))

# Confusion Matrix
cm = confusion_matrix(y_test, y_test_pred)
disp = ConfusionMatrixDisplay(cm)
disp.plot()
plt.title('Confusion Matrix')
```

```
plt.show()
```

```
Train Accuracy : 0.89
Train F1 Score: 0.94
Train Recall Score: 1.0
Train Precision Score: 0.89

Test Accuracy : 0.89
Test F1 Score: 0.94
Test Recall Score: 1.0
Test Precision Score: 0.89
```



[ ]:
```
print(classification_report(y_test,y_test_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.45      0.61     11678
           1       0.89      1.00      0.94     50601

    accuracy                           0.89     62279
```

```
   macro avg       0.92      0.72      0.78     62279
weighted avg       0.90      0.89      0.88     62279
```

[ ]:
```
#Here the recall value for the 'charged off' is very low, Hence will build a␣
↪better modeL
```

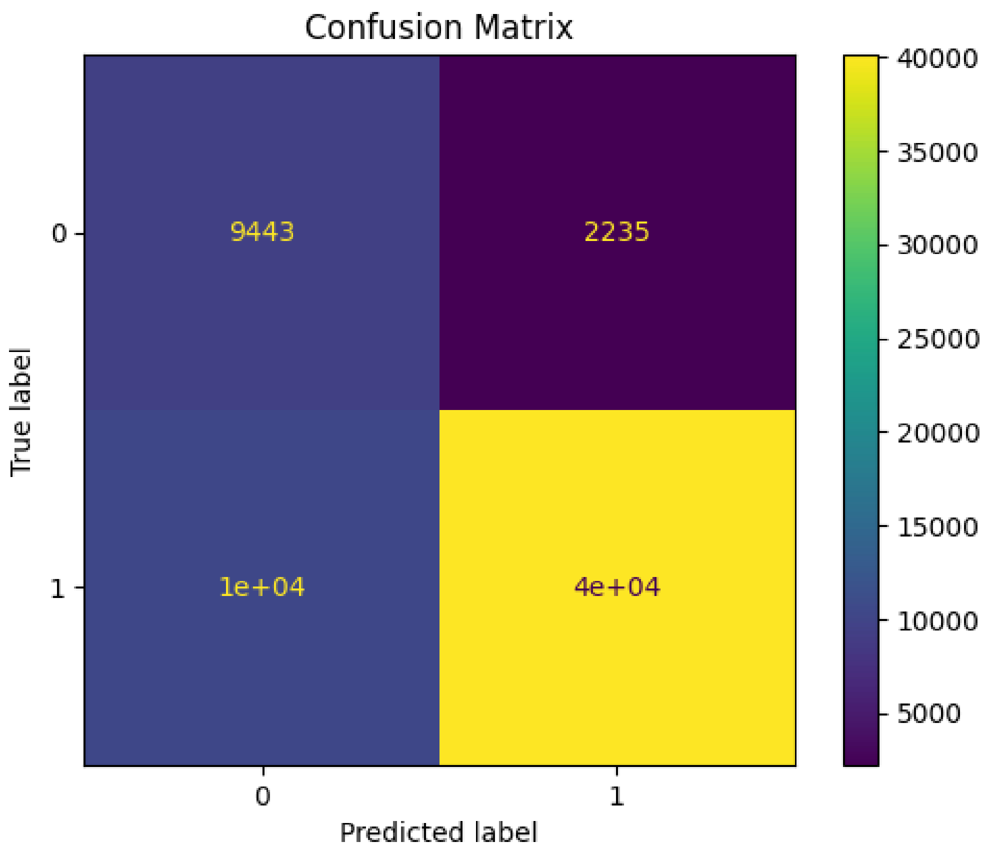
[ ]:
```
#Model-2

# Oversampling to balance the target variable

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

print(f"Before OverSampling, count of label 1: {sum(y_train == 1)}")
print(f"Before OverSampling, count of label 0: {sum(y_train == 0)}")
print(f"After OverSampling, count of label 1: {sum(y_train_res == 1)}")
print(f"After OverSampling, count of label 0: {sum(y_train_res == 0)}")
```

```
Before OverSampling, count of label 1: 202401
Before OverSampling, count of label 0: 46712
After OverSampling, count of label 1: 202401
After OverSampling, count of label 0: 202401
```

[ ]:
```
model = LogisticRegression()
model.fit(X_train_res, y_train_res)
train_preds = model.predict(X_train)
test_preds = model.predict(X_test)

#Model Evaluation
print('Train Accuracy :', round(model.score(X_train, y_train),2))
print('Train F1 Score:',round(f1_score(y_train,train_preds),2))
print('Train Recall Score:',round(recall_score(y_train,train_preds),2))
print('Train Precision Score:',round(precision_score(y_train,train_preds),2))

print('\nTest Accuracy :',round(model.score(X_test,y_test),2))
print('Test F1 Score:',round(f1_score(y_test,test_preds),2))
print('Test Recall Score:',round(recall_score(y_test,test_preds),2))
print('Test Precision Score:',round(precision_score(y_test,test_preds),2))

# Confusion Matrix
cm = confusion_matrix(y_test, test_preds)
disp = ConfusionMatrixDisplay(cm)
disp.plot()
plt.title('Confusion Matrix')
plt.show()
```

```
Train Accuracy : 0.79
```

```
Train F1 Score: 0.86
Train Recall Score: 0.79
Train Precision Score: 0.95

Test Accuracy : 0.8
Test F1 Score: 0.86
Test Recall Score: 0.79
Test Precision Score: 0.95
```



[ ]:
```python
y_pred = test_preds
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.47      0.81      0.60     11678
           1       0.95      0.79      0.86     50601

    accuracy                           0.80     62279
   macro avg       0.71      0.80      0.73     62279
weighted avg       0.86      0.80      0.81     62279
```

Observations:

The model demonstrates a high recall score, successfully identifying 80% of actual defaulters.

However, the precision for the positive class (defaulters) is low; only 47% of predicted defaulters are actually defaulters.

This high recall and low precision indicate that while the model is effective at flagging most defaulters, it also results in many false positives.Consequently, many deserving customers may be denied loans.

The low precision adversely affects the F1 score, reducing it to 60%, despite an overall accuracy of 80%. This highlights the trade-off between precision and recall in the model's performance.

Explanation:

The model is good at catching most people who don't pay back their loans it catches 80% of them.

But, when it says someone won't pay back, it's right only half of the time.47% So, there's a chance it's making mistakes and wrongly flagging people.

Because of these mistakes, some people who deserve loans might not get them.

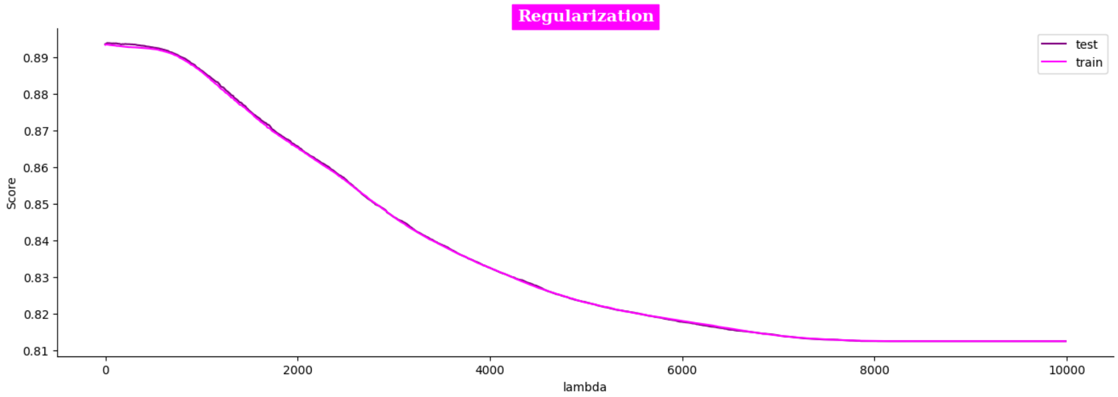
Even though the model seems okay overall, its balance between being right and not making mistakes isn't great. It's like a seesaw; when one side goes up, the other goes down.

[ ]:
```
#Regularization Model
```

[ ]:
```
#Try with different regularization factor lamda and choose the best to build␣
↪the model

lamb = np.arange(0.01, 10000, 10)
train_scores = []
test_scores = []

for lam in lamb:
    model = LogisticRegression(C = 1/lam)
    model.fit(X_train, y_train)

    tr_score = model.score(X_train, y_train)
    te_score = model.score(X_test, y_test)

    train_scores.append(tr_score)
    test_scores.append(te_score)
```

[ ]:
```
#Plot the train and test scores with respect lambda values i.e. regularization␣
↪factors
ran = np.arange(0.01, 10000, 10)
plt.figure(figsize=(16,5))
sns.lineplot(x=ran,y=test_scores,color='purple',label='test')
```

```
sns.lineplot(x=ran,y=train_scores,color='magenta',label='train')
plt.
 ↪title('Regularization',fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor='ma
plt.xlabel("lambda")
plt.ylabel("Score")
sns.despine()
plt.show()
```



[ ]:
```
#Check the index of best test score and the check the best test score

print(np.argmax(test_scores))
print(test_scores[np.argmax(test_scores)])
```

```
2
0.8939128759292859
```

[ ]:
```
#Calculate the best lambda value based on the index of best test score

best_lamb = 0.01 + (10*2)
best_lamb
```

[ ]: 20.01

[ ]:
```
#Fit the model using best lambda

reg_model = LogisticRegression(C=1/best_lamb)
reg_model.fit(X_train, y_train)
```

[ ]: LogisticRegression(C=0.04997501249375312)

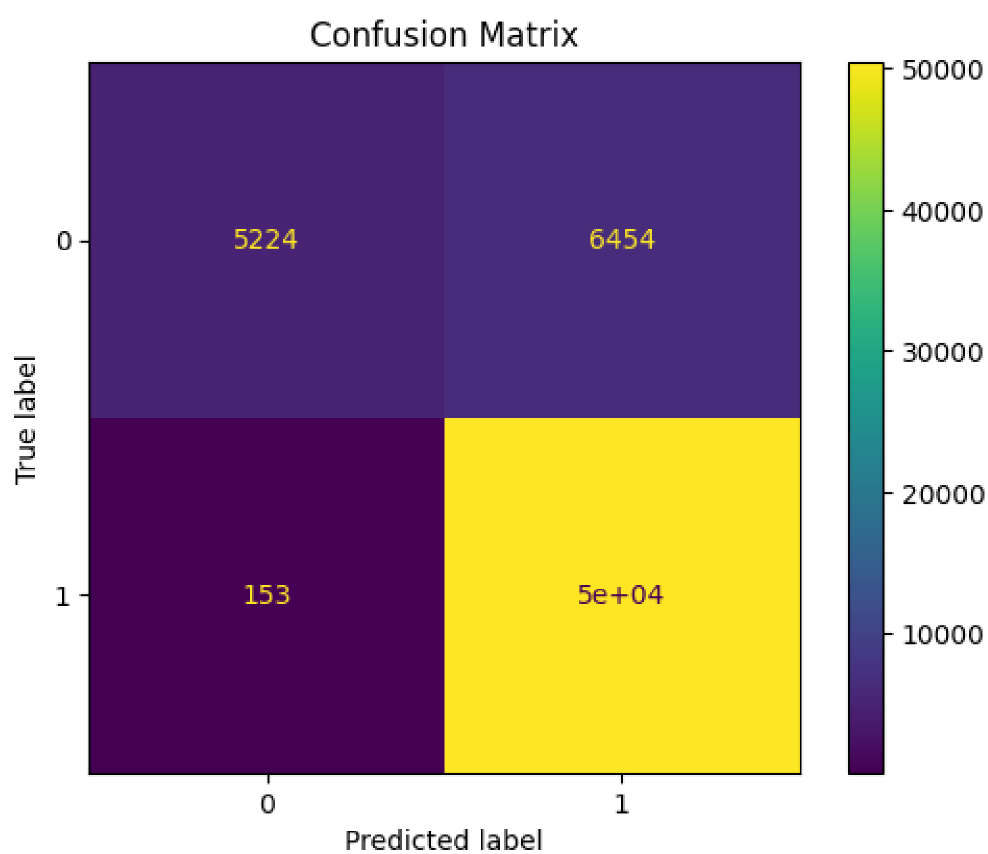
[ ]:
```
#Predict the y_values and y_probability values

y_reg_pred = reg_model.predict(X_test)
```

```
y_reg_pred_proba = reg_model.predict_proba(X_test)
```

[ ]:
```
#Print model score

print(f'Logistic Regression Model Score with best lambda: ',end='')
print(round(model.score(X_test, y_test)*100,2),'%')
```

```
Logistic Regression Model Score with best lambda: 81.25 %
```

[ ]:
```
# Confusion Matrix
cm = confusion_matrix(y_test, y_reg_pred)
disp = ConfusionMatrixDisplay(cm)
disp.plot()
plt.title('Confusion Matrix')
plt.show()
```



[ ]:
```
print(classification_report(y_test, y_reg_pred))
```

```
              precision    recall  f1-score   support
```

```
              0       0.97      0.45      0.61     11678
              1       0.89      1.00      0.94     50601

       accuracy                           0.89     62279
      macro avg       0.93      0.72      0.78     62279
   weighted avg       0.90      0.89      0.88     62279
```

Observations from classification report:

Regularized model

Precision : 89%

Recall : 100%

F1-score : 94%

Accuracy : 89%

[ ]:
```
#K-fold - Cross_validation
#cross validation accuracy has to be approx 89%
```

[ ]:
```
x=scaler.fit_transform(X)

kfold = KFold(n_splits=10)
accuracy = np.mean(cross_val_score(reg_model,x,y,cv=kfold,scoring='accuracy'))
print("Cross Validation accuracy : {:.3f}".format(accuracy))
```

```
Cross Validation accuracy : 0.894
```

[ ]:
```
cm = confusion_matrix(y_test, y_reg_pred)
cm_df = pd.DataFrame(cm, index=['Defaulter','Fully paid'],␣
 ↪columns=['Defaulter','Fully paid'])
cm_df
```

[ ]:

	Defaulter	Fully paid
Defaulter	5224	6454
Fully paid	153	50448

Insights:

TN = 5223 (True Negative: Correctly predicted Charged Off)

TP = 50450 (True Positive: Correctly predicted Fully Paid)

FP = 6455 (False Positive: Predicted Fully Paid but actually Charged Off)

FN = 151 (False Negative: Predicted Charged Off but actually Fully Paid)

Actual Negative (Charged Off) = 5223 + 6455 = 11678

Actual Positive (Fully Paid) = 151 + 50450 = 50601

Predicted Negative (Charged Off) = 5223 + 151 = 5374

Predicted Positive (Fully Paid) = 6455 + 50450 = 56905

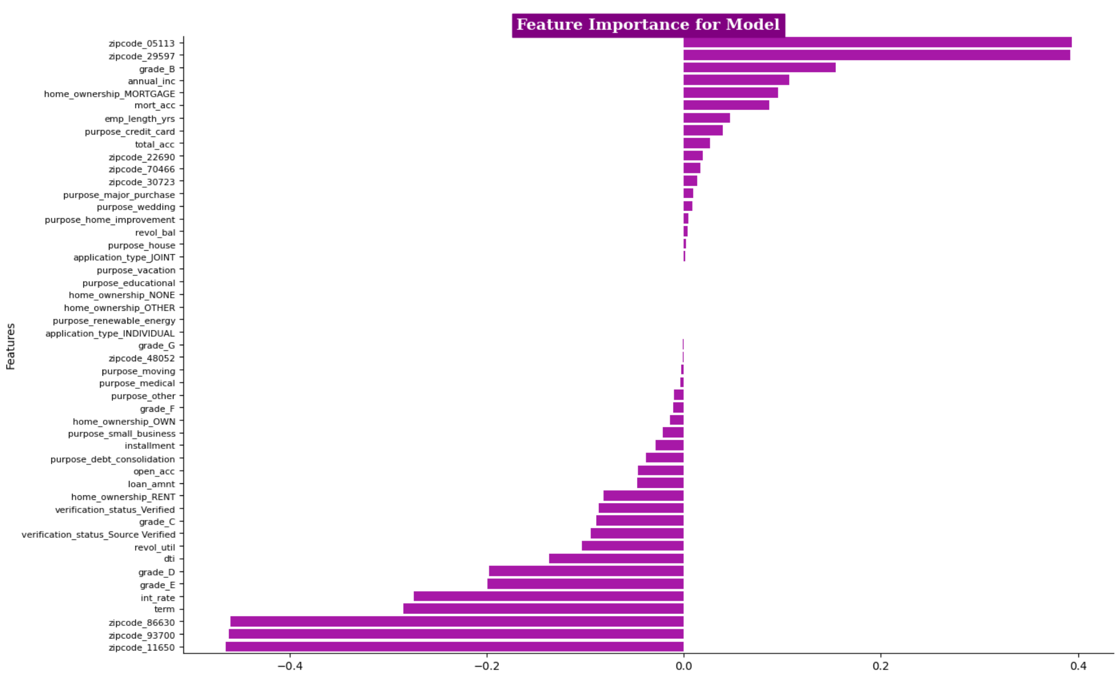
[ ]:
```python
#Collect the model coefficients and print those in dataframe format
coeff_df = pd.DataFrame()
coeff_df['Features'] = X_train_res.columns
coeff_df['Weights'] = model.coef_[0]
coeff_df['ABS_Weights'] = abs(coeff_df['Weights'])
coeff_df = coeff_df.sort_values(['ABS_Weights'], ascending=False)
coeff_df
```

[ ]:

	Features	Weights	ABS_Weights
13	zipcode_11650	-0.465074	0.465074
20	zipcode_93700	-0.461597	0.461597
19	zipcode_86630	-0.460198	0.460198
12	zipcode_05113	0.393420	0.393420
15	zipcode_29597	0.392065	0.392065
1	term	-0.284876	0.284876
2	int_rate	-0.274231	0.274231
24	grade_E	-0.199183	0.199183
23	grade_D	-0.197492	0.197492
21	grade_B	0.153863	0.153863
5	dti	-0.136682	0.136682
4	annual_inc	0.107301	0.107301
8	revol_util	-0.103188	0.103188
40	home_ownership_MORTGAGE	0.095547	0.095547
45	verification_status_Source Verified	-0.094784	0.094784
22	grade_C	-0.088887	0.088887
10	mort_acc	0.086515	0.086515
46	verification_status_Verified	-0.086047	0.086047
44	home_ownership_RENT	-0.081123	0.081123
0	loan_amnt	-0.047541	0.047541
11	emp_length_yrs	0.047248	0.047248
6	open_acc	-0.046230	0.046230
27	purpose_credit_card	0.039204	0.039204
28	purpose_debt_consolidation	-0.038272	0.038272
3	installment	-0.028448	0.028448
9	total_acc	0.026258	0.026258
37	purpose_small_business	-0.020956	0.020956
14	zipcode_22690	0.019340	0.019340
18	zipcode_70466	0.016947	0.016947
43	home_ownership_OWN	-0.014123	0.014123
16	zipcode_30723	0.013786	0.013786
25	grade_F	-0.010375	0.010375
35	purpose_other	-0.009954	0.009954
32	purpose_major_purchase	0.009729	0.009729
39	purpose_wedding	0.008596	0.008596

```
30             purpose_home_improvement  0.004803     0.004803
7                             revol_bal  0.003796     0.003796
33                      purpose_medical -0.003333     0.003333
34                       purpose_moving -0.002790     0.002790
31                        purpose_house  0.002104     0.002104
17                        zipcode_48052 -0.001237     0.001237
48                application_type_JOINT  0.001189     0.001189
26                              grade_G -0.000720     0.000720
47           application_type_INDIVIDUAL -0.000514     0.000514
36             purpose_renewable_energy -0.000335     0.000335
42                  home_ownership_OTHER -0.000185     0.000185
41                   home_ownership_NONE -0.000161     0.000161
29                  purpose_educational -0.000156     0.000156
38                     purpose_vacation -0.000053     0.000053
```

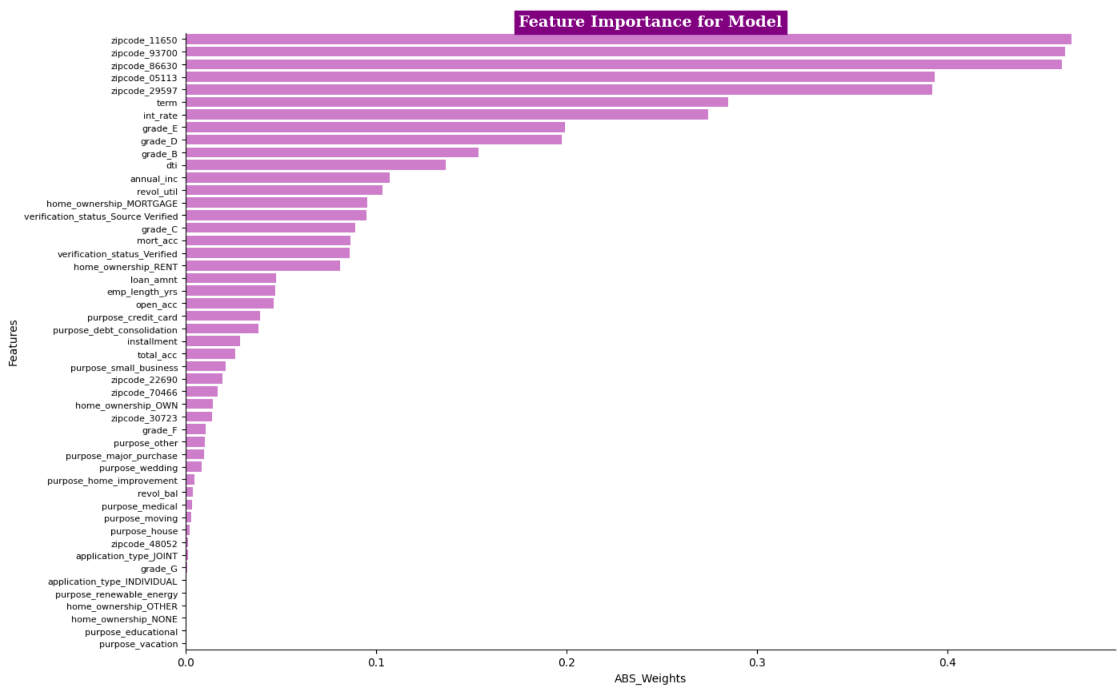
[ ]:
```python
imp_feature = coeff_df.sort_values(by='Weights',ascending=False)
plt.figure(figsize=(15,10))
sns.barplot(y = imp_feature['Features'],
            x = imp_feature['Weights'],color='m')
plt.title("Feature Importance for␣
 ↪Model",fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor='purple',color='w')
plt.xlabel("Weights")
plt.yticks(fontsize=8)
plt.ylabel("Features")
sns.despine()
plt.show()
```



```
[ ]: #Logistic Regression model intercept

model.intercept_
```

```
[ ]: array([1.76790228])
```

```
[ ]: plt.figure(figsize=(15,10))
sns.barplot(y = coeff_df['Features'],x = coeff_df['ABS_Weights'],color='orchid')
plt.title("Feature Importance for Model",fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor='purple',color='w')
plt.xlabel("ABS_Weights")
plt.yticks(fontsize=8)
plt.ylabel("Features")
sns.despine()
plt.show()
```



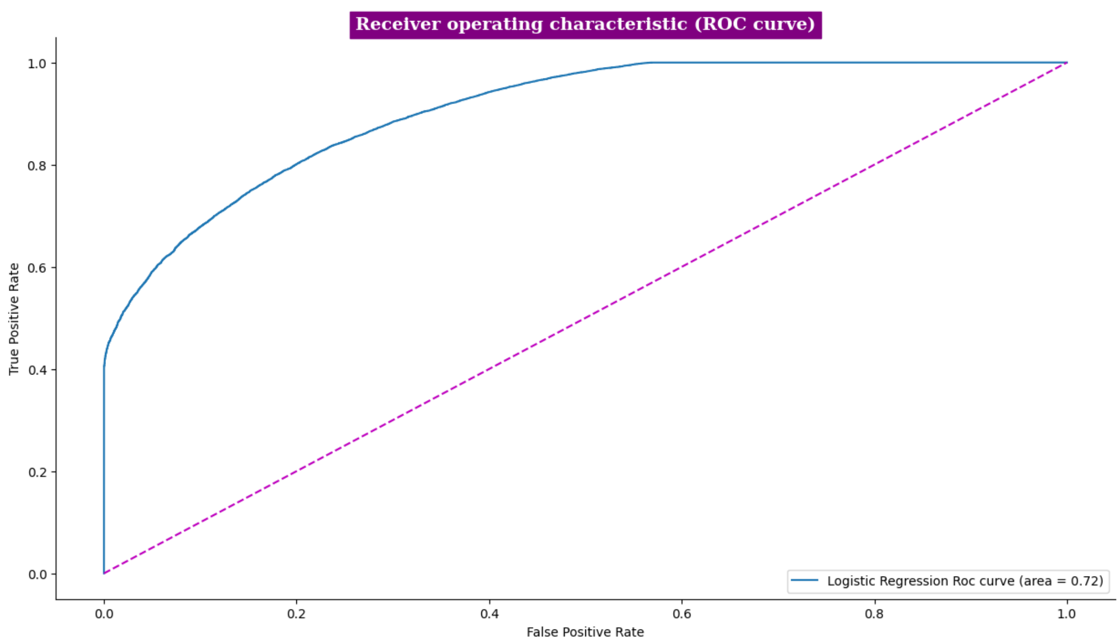
Observations:

The model has assigned significant weight to the zip_code, Annual Income, grade features, indicating that certain zip codes strongly influence the prediction of defaulters.

Features such as dti (debt-to-income ratio), open_acc (number of open accounts), and loan_amnt (loan amount) also have high positive coefficients, highlighting their importance in predicting default risk.

On the other hand, several zip codes have large negative coefficients, suggesting that they are associated with a lower likelihood of default.

[ ]:
```
#ROC AUC curve
# area under ROC curve
logit_roc_auc = roc_auc_score(y_test,y_reg_pred)

# Compute the false positive rate, true positive rate, and thresholds
fpr,tpr,thresholds = roc_curve(y_test,y_reg_pred_proba[:,1])

# Compute the area under the ROC curve
roc_auc = auc(fpr, tpr)

# plot ROC curve
plt.figure(figsize=(15,8))
plt.plot(fpr,tpr,label='Logistic Regression Roc curve (area = %0.2f)'%␣
 ↪logit_roc_auc)
plt.plot([0,1],[0,1],'m--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic (ROC␣
 ↪curve)',fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor='purple',color='w'
plt.legend(loc="lower right")
sns.despine()
plt.show()
```



[ ]:
```
logit_roc_auc
```

[ ]: np.float64(0.7221566085466022)

[ ]:
```
roc_auc = auc(fpr, tpr)
roc_auc
```

[ ]: np.float64(0.9036968327803755)

Insights:

Trade-off in Performance:

The ROC curve area, representing model performance, is 72%. This indicates that the model effectively distinguishes between classes 72% of the time.

Ideally, we aim for a higher True Positive Rate (TPR) and a lower False Positive Rate (FPR) to ensure accurate predictions.

The ROC curve illustrates that as True Positives increase, there's a simultaneous increase in False Positives.

Misclassification:

This trade-off implies that while identifying more Fully Paid customers, there's a heightened risk of misclassifying Charged Off customers as Fully Paid, potentially leading to Non-Performing Assets (NPAs).

These points emphasize the need to mitigate this risk:

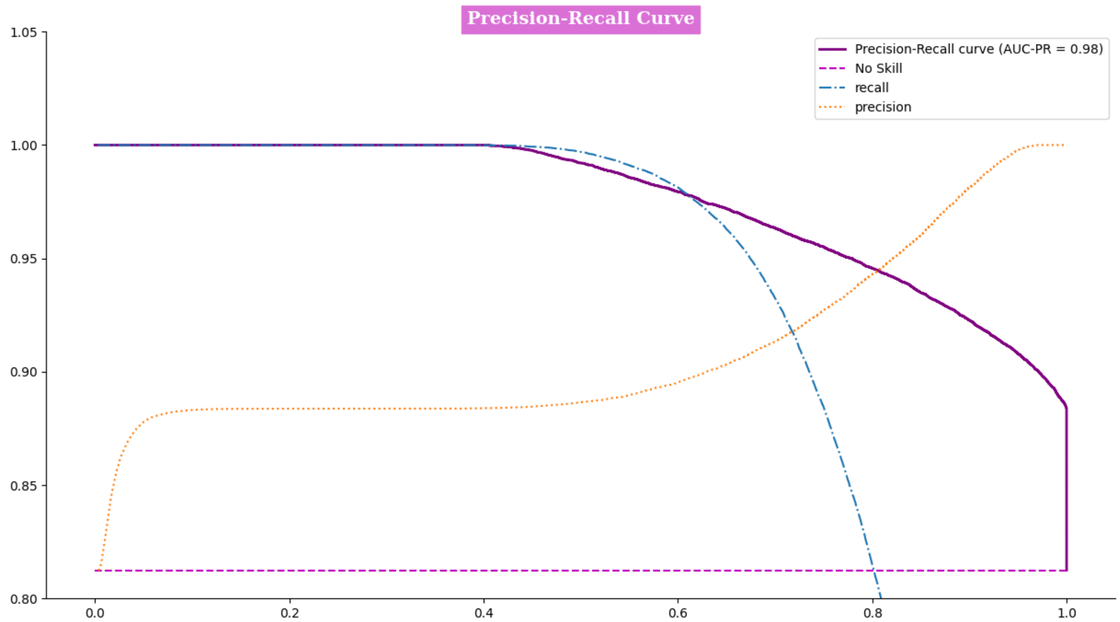
Reducing FPR while maintaining TPR is crucial to minimize misclassifications and associated risks.

By shifting False Positives towards the left on the ROC curve, the model's overall performance, as measured by AUC, can improve.

This improvement in AUC relies on maintaining a high True Positive Rate while reducing False Positives.

[ ]:
```
precision, recall, thresholds = precision_recall_curve(y_test,␣
↪y_reg_pred_proba[:,1])

average_precision = average_precision_score(y_test, y_reg_pred_proba[:,1])

no_skill = len(y_test[y_test==1]) / len(y_test)

plt.figure(figsize=(15,8))
plt.plot(recall, precision, color='purple', lw=2, label=f'Precision-Recall␣
↪curve (AUC-PR = {average_precision:.2f})')
plt.plot([0, 1], [no_skill, no_skill], linestyle='--', label='No Skill',␣
↪color='m')
plt.plot(thresholds, recall[0:thresholds.shape[0]], label='recall',linestyle='-.
↪')
plt.plot(thresholds, precision[0:thresholds.shape[0]],␣
↪label='precision',linestyle='dotted')
```

```
# plt.xlim([0.0, 1.0])
plt.ylim([0.8, 1.05])
plt.title('Precision-Recall␣
↪Curve',fontsize=14,fontfamily='serif',fontweight='bold',backgroundcolor='orchid',color='w')
plt.legend(loc='upper right')
sns.despine()
plt.show()
```



[ ]:
```
auc(recall, precision).round(3)
```

[ ]: np.float64(0.975)

Observations:

Insight:

The Area Under the Curve (AUC) for the precision-recall curve is 0.975. This high AUC value suggests that the model achieves excellent performance in distinguishing between positive and negative classes, showcasing strong precision-recall characteristics.

Precision-Recall Curve Superiority: Precision-recall curves are pivotal, especially in imbalanced datasets, focusing on accurate predictions of the relevant class (Class 1 - Fully paid in this case).

Irrelevance of True Negatives: Precision and recall computations disregard true negatives, simplifying focus to the correct prediction of Fully Paid customers.

AUC Strengthens Model Evaluation: A high AUC (97.5%) underscores the model's robustness in distinguishing between classes, indicating its efficacy.

Precision Enhancement Priority: Optimal model refinement centers on elevating precision by minimizing False Positives, vital for improving overall performance and mitigating risks.

[ ]:
```
# balenced Model
lr = LogisticRegression(max_iter=1000, class_weight='balanced')

lr_model = lr.fit(X_train, y_train)

print(classification_report(y_test, lr_model.predict(X_test)))

cm_bal = confusion_matrix(y_test, lr_model.predict(X_test))
cm_bal_df = pd.DataFrame(cm_bal, index=['Defaulter','Fully paid'],
    columns=['Defaulter','Fully paid'])
cm_bal_df
```

```
              precision    recall  f1-score   support

           0       0.47      0.81      0.60     11678
           1       0.95      0.79      0.86     50601

    accuracy                           0.79     62279
   macro avg       0.71      0.80      0.73     62279
weighted avg       0.86      0.79      0.81     62279
```

[ ]:

	Defaulter	Fully paid
Defaulter	9468	2210
Fully paid	10586	40015

Observations from classification report:

Balenced model

Precision : 95%

Recall : 79%

F1-score : 86%

Accuracy : 79%

Insights:

TN = 9466 (True Negative: Correctly predicted Charged Off)

TP = 40028 (True Positive: Correctly predicted Fully Paid)

FP = 2212 (False Positive: Predicted Fully Paid but actually Charged Off)

FN = 10573 (False Negative: Predicted Charged Off but actually Fully Paid)

Actual Negative (Charged Off) = 9466 + 2212 = 11678

Actual Positive (Fully Paid) = 10573 + 40028 = 50601

Predicted Negative (Charged Off) = 9466 + 10573 = 20039

Predicted Positive (Fully Paid) = 2212 + 40028 = 42240

[ ]:
```
lr_model.intercept_
```

[ ]:
```
array([6.35576692])
```

[ ]:
```
#Q6: Thinking from a bank's perspective, which metric should our primary focus␣
↪be on..
#a. ROC AUC
#b. Precision
#c. Recall
#d. F1 Score
```

Ans:

From a bank's perspective, minimizing risks and maximizing profitability are paramount. ROC AUC (Receiver Operating Characteristic Area Under Curve) is indeed a crucial metric because it encompasses both True Positive Rate (TPR) and False Positive Rate (FPR)

Bank's primary **focus should be on ROC AUC** , because bank needs to reduce FPR (False Positive Rate) and needs to increase the TPR (True Positive Rate).

Maximizing TPR ensures that the bank correctly identifies customers who fully pay their loans (reducing False Negatives), while minimizing FPR ensures that the bank doesn't wrongly classify customers as fully paid when they're actually charged off (reducing False Positives).

By optimizing ROC AUC, the bank can strike a balance between correctly identifying creditworthy customers and minimizing the risk of defaulters, thereby enhancing the overall performance and reliability of its credit scoring model.

[ ]:
```
#Another approach:
```

Since I'm having High Recall value of 100% in Regularized model(most efficient model:

1. From a bank's perspective, the primary focus should be on minimizing risks while maximizing profitability. Therefore, the most relevant metric would be Precision.
2. Precision represents the proportion of correctly predicted positive instances (e.g., customers who fully pay their loans) out of all instances predicted as positive. In the context of a bank, precision reflects the accuracy of identifying creditworthy customers who are likely to repay their loans. Maximizing precision ensures that the bank minimizes the number of false positives, which are instances where the bank incorrectly identifies customers as creditworthy when they are not. By prioritizing precision, the bank can reduce the risk of loan defaults and associated financial losses.
3. While ROC AUC, Recall, and F1 Score are also important metrics, precision aligns closely with the bank's objective of minimizing risks and ensuring the quality of its loan portfolio.

[ ]:
```
#Q7. How does the gap in precision and recall affect the bank?
```

Ans:

To comprehend the errors made by a model, it's crucial to evaluate both false positives and false negatives, which are gauged through metrics like recall and precision. When recall is low, it poses a significant risk for the bank.

So, the gap between precision and recall will affect the bank. As the gap widens, there will be increase in incorrect predictions.

Good precision means less False Positives. i.e. Less NPA loan accounts.

Good recall means less False Negatives. i.e. not loosing on good customer.

[ ]:
```
#Q8. Which were the features that heavily affected the outcome?
```

Ans:

Address(Zipcode), Annual_Income, Grade seems to be most important feature in our case.

Loan duration term, Total Credit balance revol_bal, : Monthly debt vs. monthly income ratio dti, Interest int_rate also has high weights(coeffients) in the model .

[ ]:
```
#Q9. Will the results be affected by geographical location? (Yes/No)
"Yes, we can see that zip_code (Address) is a very important feature so␣
 ↪geographical location has impact on our result."
```

[ ]:
```
#Business Recommendations for LoanTap
```

Focus on maximizing the F1 score and area under the Precision-Recall Curve to effectively manage the precision-recall trade-off. This ensures identifying most defaulters while reducing false positives, enhancing risk management.

Consider using more complex classifiers like Random Forests or XGBoost and perform hyperparameter tuning to enhance model performance and capture intricate relationships in the data.

Employed stratified k-fold cross-validation to ensure representative distribution of minority class in each fold, providing reliable estimates of model performance.

Policy Adjustments Based on Insights

Cross-Validation:

Model Improvement:

Optimize Loan Approval Strategy:

Scrutinize loans with lower grades more rigorously and consider adjusting interest rates to compensate for higher risk.

Implement targeted strategies for high-risk zip codes, such as additional verification steps or higher interest rates.

Evaluate small business loans with additional financial health checks and collateral requirements to mitigate default risk.

By implementing these recommendations, LoanTap can enhance their loan approval process, minimize the risk of NPAs, and ensure sustainable growth and financial stability.

[ ]:
```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!pip install pypandoc
```

```
Reading package lists… Done
Building dependency tree… Done
Reading state information… Done
pandoc is already the newest version (2.9.2.1-3ubuntu2).
pandoc set to manually installed.
The following additional packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono
  fonts-texgyre fonts-urw-base35 libapache-pom-java libcommons-logging-java
  libcommons-parent-java libfontbox-java libgs9 libgs9-common libidn12
  libijs-0.35 libjbig2dec0 libkpathsea6 libpdfbox-java libptexenc1 libruby3.0
  libsynctex2 libteckit0 libtexlua53 libtexluajit2 libwoff1 libzzip-0-13
  lmodern poppler-data preview-latex-style rake ruby ruby-net-telnet
  ruby-rubygems ruby-webrick ruby-xmlrpc ruby3.0 rubygems-integration t1utils
  teckit tex-common tex-gyre texlive-base texlive-binaries
  texlive-fonts-recommended texlive-latex-base texlive-latex-recommended
  texlive-pictures texlive-plain-generic tipa xfonts-encodings xfonts-utils
Suggested packages:
  fonts-noto fonts-freefont-otf | fonts-freefont-ttf libavalon-framework-java
  libcommons-logging-java-doc libexcalibur-logkit-java liblog4j1.2-java
  poppler-utils ghostscript fonts-japanese-mincho | fonts-ipafont-mincho
  fonts-japanese-gothic | fonts-ipafont-gothic fonts-arphic-ukai
  fonts-arphic-uming fonts-nanum ri ruby-dev bundler debhelper gv
  | postscript-viewer perl-tk xpdf | pdf-viewer xzdec
  texlive-fonts-recommended-doc texlive-latex-base-doc python3-pygments
  icc-profiles libfile-which-perl libspreadsheet-parseexcel-perl
  texlive-latex-extra-doc texlive-latex-recommended-doc texlive-luatex
  texlive-pstricks dot2tex prerex texlive-pictures-doc vprerex
  default-jre-headless tipa-doc
The following NEW packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono
  fonts-texgyre fonts-urw-base35 libapache-pom-java libcommons-logging-java
  libcommons-parent-java libfontbox-java libgs9 libgs9-common libidn12
  libijs-0.35 libjbig2dec0 libkpathsea6 libpdfbox-java libptexenc1 libruby3.0
  libsynctex2 libteckit0 libtexlua53 libtexluajit2 libwoff1 libzzip-0-13
  lmodern poppler-data preview-latex-style rake ruby ruby-net-telnet
  ruby-rubygems ruby-webrick ruby-xmlrpc ruby3.0 rubygems-integration t1utils
  teckit tex-common tex-gyre texlive texlive-base texlive-binaries
  texlive-fonts-recommended texlive-latex-base texlive-latex-extra
  texlive-latex-recommended texlive-pictures texlive-plain-generic
  texlive-xetex tipa xfonts-encodings xfonts-utils
0 upgraded, 54 newly installed, 0 to remove and 41 not upgraded.
```

```
Need to get 182 MB of archives.
After this operation, 571 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-droid-fallback all
1:6.0.1r16-1.1build1 [1,805 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1
[2,696 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 poppler-data all
0.4.11-1 [2,171 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-common all 6.17
[33.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-urw-base35 all
20200910-1 [6,367 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9-common
all 9.55.0~dfsg1-0ubuntu5.13 [753 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libidn12 amd64
1.38-4ubuntu1 [60.0 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/main amd64 libijs-0.35 amd64
0.35-15build2 [16.5 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/main amd64 libjbig2dec0 amd64
0.19-3build2 [64.7 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9 amd64
9.55.0~dfsg1-0ubuntu5.13 [5,032 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libkpathsea6
amd64 2021.20210626.59705-1ubuntu0.2 [60.4 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy/main amd64 libwoff1 amd64
1.0.2-1build4 [45.2 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy/universe amd64 dvisvgm amd64
2.13.1-1 [1,221 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-lmodern all
2.004.5-6.1 [4,532 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-noto-mono all
20201225-1build1 [397 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-texgyre all
20180621-3.1 [10.2 MB]
Get:17 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libapache-pom-java
all 18-1 [4,720 B]
Get:18 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-parent-
java all 43-1 [10.8 kB]
Get:19 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-logging-
java all 1.2-2 [60.3 kB]
Get:20 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libptexenc1
amd64 2021.20210626.59705-1ubuntu0.2 [39.1 kB]
Get:21 http://archive.ubuntu.com/ubuntu jammy/main amd64 rubygems-integration
all 1.18 [5,336 B]
Get:22 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby3.0 amd64
3.0.2-7ubuntu2.11 [50.1 kB]
Get:23 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby-rubygems
all 3.3.5-2ubuntu1.2 [228 kB]
```

```
Get:24 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby amd64 1:3.0~exp1
[5,100 B]
Get:25 http://archive.ubuntu.com/ubuntu jammy/main amd64 rake all 13.0.6-2 [61.7
kB]
Get:26 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby-net-telnet all
0.1.1-2 [12.6 kB]
Get:27 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby-webrick
all 1.7.0-3ubuntu0.2 [52.5 kB]
Get:28 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby-xmlrpc all
0.3.2-1ubuntu0.1 [24.9 kB]
Get:29 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libruby3.0
amd64 3.0.2-7ubuntu2.11 [5,114 kB]
Get:30 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libsynctex2
amd64 2021.20210626.59705-1ubuntu0.2 [55.6 kB]
Get:31 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libteckit0 amd64
2.5.11+ds1-1 [421 kB]
Get:32 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexlua53
amd64 2021.20210626.59705-1ubuntu0.2 [120 kB]
Get:33 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexluajit2
amd64 2021.20210626.59705-1ubuntu0.2 [267 kB]
Get:34 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libzzip-0-13 amd64
0.13.72+dfsg.1-1.1 [27.0 kB]
Get:35 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-encodings all
1:1.0.5-0ubuntu2 [578 kB]
Get:36 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-utils amd64
1:7.7+6build2 [94.6 kB]
Get:37 http://archive.ubuntu.com/ubuntu jammy/universe amd64 lmodern all
2.004.5-6.1 [9,471 kB]
Get:38 http://archive.ubuntu.com/ubuntu jammy/universe amd64 preview-latex-style
all 12.2-1ubuntu1 [185 kB]
Get:39 http://archive.ubuntu.com/ubuntu jammy/main amd64 t1utils amd64
1.41-4build2 [61.3 kB]
Get:40 http://archive.ubuntu.com/ubuntu jammy/universe amd64 teckit amd64
2.5.11+ds1-1 [699 kB]
Get:41 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-gyre all
20180621-3.1 [6,209 kB]
Get:42 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 texlive-
binaries amd64 2021.20210626.59705-1ubuntu0.2 [9,860 kB]
Get:43 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-base all
2021.20220204-1 [21.0 MB]
Get:44 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-fonts-
recommended all 2021.20220204-1 [4,972 kB]
Get:45 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-base
all 2021.20220204-1 [1,128 kB]
Get:46 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-
recommended all 2021.20220204-1 [14.4 MB]
Get:47 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive all
2021.20220204-1 [14.3 kB]
```

```
Get:48 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libfontbox-java all
1:1.8.16-2 [207 kB]
Get:49 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libpdfbox-java all
1:1.8.16-2 [5,199 kB]
Get:50 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-pictures
all 2021.20220204-1 [8,720 kB]
Get:51 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-extra
all 2021.20220204-1 [13.9 MB]
Get:52 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-plain-
generic all 2021.20220204-1 [27.5 MB]
Get:53 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tipa all 2:1.3-21
[2,967 kB]
Get:54 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-xetex all
2021.20220204-1 [12.4 MB]
Fetched 182 MB in 8s (23.5 MB/s)
Extracting templates from packages: 100%
Preconfiguring packages …
Selecting previously unselected package fonts-droid-fallback.
(Reading database … 125080 files and directories currently installed.)
Preparing to unpack …/00-fonts-droid-fallback_1%3a6.0.1r16-1.1build1_all.deb
…
Unpacking fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Selecting previously unselected package fonts-lato.
Preparing to unpack …/01-fonts-lato_2.0-2.1_all.deb …
Unpacking fonts-lato (2.0-2.1) …
Selecting previously unselected package poppler-data.
Preparing to unpack …/02-poppler-data_0.4.11-1_all.deb …
Unpacking poppler-data (0.4.11-1) …
Selecting previously unselected package tex-common.
Preparing to unpack …/03-tex-common_6.17_all.deb …
Unpacking tex-common (6.17) …
Selecting previously unselected package fonts-urw-base35.
Preparing to unpack …/04-fonts-urw-base35_20200910-1_all.deb …
Unpacking fonts-urw-base35 (20200910-1) …
Selecting previously unselected package libgs9-common.
Preparing to unpack …/05-libgs9-common_9.55.0~dfsg1-0ubuntu5.13_all.deb …
Unpacking libgs9-common (9.55.0~dfsg1-0ubuntu5.13) …
Selecting previously unselected package libidn12:amd64.
Preparing to unpack …/06-libidn12_1.38-4ubuntu1_amd64.deb …
Unpacking libidn12:amd64 (1.38-4ubuntu1) …
Selecting previously unselected package libijs-0.35:amd64.
Preparing to unpack …/07-libijs-0.35_0.35-15build2_amd64.deb …
Unpacking libijs-0.35:amd64 (0.35-15build2) …
Selecting previously unselected package libjbig2dec0:amd64.
Preparing to unpack …/08-libjbig2dec0_0.19-3build2_amd64.deb …
Unpacking libjbig2dec0:amd64 (0.19-3build2) …
Selecting previously unselected package libgs9:amd64.
Preparing to unpack …/09-libgs9_9.55.0~dfsg1-0ubuntu5.13_amd64.deb …
```

```
Unpacking libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.13) …
Selecting previously unselected package libkpathsea6:amd64.
Preparing to unpack …/10-libkpathsea6_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libwoff1:amd64.
Preparing to unpack …/11-libwoff1_1.0.2-1build4_amd64.deb …
Unpacking libwoff1:amd64 (1.0.2-1build4) …
Selecting previously unselected package dvisvgm.
Preparing to unpack …/12-dvisvgm_2.13.1-1_amd64.deb …
Unpacking dvisvgm (2.13.1-1) …
Selecting previously unselected package fonts-lmodern.
Preparing to unpack …/13-fonts-lmodern_2.004.5-6.1_all.deb …
Unpacking fonts-lmodern (2.004.5-6.1) …
Selecting previously unselected package fonts-noto-mono.
Preparing to unpack …/14-fonts-noto-mono_20201225-1build1_all.deb …
Unpacking fonts-noto-mono (20201225-1build1) …
Selecting previously unselected package fonts-texgyre.
Preparing to unpack …/15-fonts-texgyre_20180621-3.1_all.deb …
Unpacking fonts-texgyre (20180621-3.1) …
Selecting previously unselected package libapache-pom-java.
Preparing to unpack …/16-libapache-pom-java_18-1_all.deb …
Unpacking libapache-pom-java (18-1) …
Selecting previously unselected package libcommons-parent-java.
Preparing to unpack …/17-libcommons-parent-java_43-1_all.deb …
Unpacking libcommons-parent-java (43-1) …
Selecting previously unselected package libcommons-logging-java.
Preparing to unpack …/18-libcommons-logging-java_1.2-2_all.deb …
Unpacking libcommons-logging-java (1.2-2) …
Selecting previously unselected package libptexenc1:amd64.
Preparing to unpack …/19-libptexenc1_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package rubygems-integration.
Preparing to unpack …/20-rubygems-integration_1.18_all.deb …
Unpacking rubygems-integration (1.18) …
Selecting previously unselected package ruby3.0.
Preparing to unpack …/21-ruby3.0_3.0.2-7ubuntu2.11_amd64.deb …
Unpacking ruby3.0 (3.0.2-7ubuntu2.11) …
Selecting previously unselected package ruby-rubygems.
Preparing to unpack …/22-ruby-rubygems_3.3.5-2ubuntu1.2_all.deb …
Unpacking ruby-rubygems (3.3.5-2ubuntu1.2) …
Selecting previously unselected package ruby.
Preparing to unpack …/23-ruby_1%3a3.0~exp1_amd64.deb …
Unpacking ruby (1:3.0~exp1) …
Selecting previously unselected package rake.
Preparing to unpack …/24-rake_13.0.6-2_all.deb …
Unpacking rake (13.0.6-2) …
```

```
Selecting previously unselected package ruby-net-telnet.
Preparing to unpack …/25-ruby-net-telnet_0.1.1-2_all.deb …
Unpacking ruby-net-telnet (0.1.1-2) …
Selecting previously unselected package ruby-webrick.
Preparing to unpack …/26-ruby-webrick_1.7.0-3ubuntu0.2_all.deb …
Unpacking ruby-webrick (1.7.0-3ubuntu0.2) …
Selecting previously unselected package ruby-xmlrpc.
Preparing to unpack …/27-ruby-xmlrpc_0.3.2-1ubuntu0.1_all.deb …
Unpacking ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Selecting previously unselected package libruby3.0:amd64.
Preparing to unpack …/28-libruby3.0_3.0.2-7ubuntu2.11_amd64.deb …
Unpacking libruby3.0:amd64 (3.0.2-7ubuntu2.11) …
Selecting previously unselected package libsynctex2:amd64.
Preparing to unpack …/29-libsynctex2_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libteckit0:amd64.
Preparing to unpack …/30-libteckit0_2.5.11+ds1-1_amd64.deb …
Unpacking libteckit0:amd64 (2.5.11+ds1-1) …
Selecting previously unselected package libtexlua53:amd64.
Preparing to unpack …/31-libtexlua53_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libtexluajit2:amd64.
Preparing to unpack
…/32-libtexluajit2_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libzzip-0-13:amd64.
Preparing to unpack …/33-libzzip-0-13_0.13.72+dfsg.1-1.1_amd64.deb …
Unpacking libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Selecting previously unselected package xfonts-encodings.
Preparing to unpack …/34-xfonts-encodings_1%3a1.0.5-0ubuntu2_all.deb …
Unpacking xfonts-encodings (1:1.0.5-0ubuntu2) …
Selecting previously unselected package xfonts-utils.
Preparing to unpack …/35-xfonts-utils_1%3a7.7+6build2_amd64.deb …
Unpacking xfonts-utils (1:7.7+6build2) …
Selecting previously unselected package lmodern.
Preparing to unpack …/36-lmodern_2.004.5-6.1_all.deb …
Unpacking lmodern (2.004.5-6.1) …
Selecting previously unselected package preview-latex-style.
Preparing to unpack …/37-preview-latex-style_12.2-1ubuntu1_all.deb …
Unpacking preview-latex-style (12.2-1ubuntu1) …
Selecting previously unselected package t1utils.
Preparing to unpack …/38-t1utils_1.41-4build2_amd64.deb …
Unpacking t1utils (1.41-4build2) …
Selecting previously unselected package teckit.
Preparing to unpack …/39-teckit_2.5.11+ds1-1_amd64.deb …
Unpacking teckit (2.5.11+ds1-1) …
```

```
Selecting previously unselected package tex-gyre.
Preparing to unpack …/40-tex-gyre_20180621-3.1_all.deb …
Unpacking tex-gyre (20180621-3.1) …
Selecting previously unselected package texlive-binaries.
Preparing to unpack …/41-texlive-
binaries_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package texlive-base.
Preparing to unpack …/42-texlive-base_2021.20220204-1_all.deb …
Unpacking texlive-base (2021.20220204-1) …
Selecting previously unselected package texlive-fonts-recommended.
Preparing to unpack …/43-texlive-fonts-recommended_2021.20220204-1_all.deb …
Unpacking texlive-fonts-recommended (2021.20220204-1) …
Selecting previously unselected package texlive-latex-base.
Preparing to unpack …/44-texlive-latex-base_2021.20220204-1_all.deb …
Unpacking texlive-latex-base (2021.20220204-1) …
Selecting previously unselected package texlive-latex-recommended.
Preparing to unpack …/45-texlive-latex-recommended_2021.20220204-1_all.deb …
Unpacking texlive-latex-recommended (2021.20220204-1) …
Selecting previously unselected package texlive.
Preparing to unpack …/46-texlive_2021.20220204-1_all.deb …
Unpacking texlive (2021.20220204-1) …
Selecting previously unselected package libfontbox-java.
Preparing to unpack …/47-libfontbox-java_1%3a1.8.16-2_all.deb …
Unpacking libfontbox-java (1:1.8.16-2) …
Selecting previously unselected package libpdfbox-java.
Preparing to unpack …/48-libpdfbox-java_1%3a1.8.16-2_all.deb …
Unpacking libpdfbox-java (1:1.8.16-2) …
Selecting previously unselected package texlive-pictures.
Preparing to unpack …/49-texlive-pictures_2021.20220204-1_all.deb …
Unpacking texlive-pictures (2021.20220204-1) …
Selecting previously unselected package texlive-latex-extra.
Preparing to unpack …/50-texlive-latex-extra_2021.20220204-1_all.deb …
Unpacking texlive-latex-extra (2021.20220204-1) …
Selecting previously unselected package texlive-plain-generic.
Preparing to unpack …/51-texlive-plain-generic_2021.20220204-1_all.deb …
Unpacking texlive-plain-generic (2021.20220204-1) …
Selecting previously unselected package tipa.
Preparing to unpack …/52-tipa_2%3a1.3-21_all.deb …
Unpacking tipa (2:1.3-21) …
Selecting previously unselected package texlive-xetex.
Preparing to unpack …/53-texlive-xetex_2021.20220204-1_all.deb …
Unpacking texlive-xetex (2021.20220204-1) …
Setting up fonts-lato (2.0-2.1) …
Setting up fonts-noto-mono (20201225-1build1) …
Setting up libwoff1:amd64 (1.0.2-1build4) …
Setting up libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libijs-0.35:amd64 (0.35-15build2) …
```

```
Setting up libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libfontbox-java (1:1.8.16-2) …
Setting up rubygems-integration (1.18) …
Setting up libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Setting up fonts-urw-base35 (20200910-1) …
Setting up poppler-data (0.4.11-1) …
Setting up tex-common (6.17) …
update-language: texlive-base not installed and configured, doing nothing!
Setting up libjbig2dec0:amd64 (0.19-3build2) …
Setting up libteckit0:amd64 (2.5.11+ds1-1) …
Setting up libapache-pom-java (18-1) …
Setting up ruby-net-telnet (0.1.1-2) …
Setting up xfonts-encodings (1:1.0.5-0ubuntu2) …
Setting up t1utils (1.41-4build2) …
Setting up libidn12:amd64 (1.38-4ubuntu1) …
Setting up fonts-texgyre (20180621-3.1) …
Setting up libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up ruby-webrick (1.7.0-3ubuntu0.2) …
Setting up fonts-lmodern (2.004.5-6.1) …
Setting up fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Setting up ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Setting up libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libgs9-common (9.55.0~dfsg1-0ubuntu5.13) …
Setting up teckit (2.5.11+ds1-1) …
Setting up libpdfbox-java (1:1.8.16-2) …
Setting up libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.13) …
Setting up preview-latex-style (12.2-1ubuntu1) …
Setting up libcommons-parent-java (43-1) …
Setting up dvisvgm (2.13.1-1) …
Setting up libcommons-logging-java (1.2-2) …
Setting up xfonts-utils (1:7.7+6build2) …
Setting up libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
update-alternatives: using /usr/bin/xdvi-xaw to provide /usr/bin/xdvi.bin
(xdvi.bin) in auto mode
update-alternatives: using /usr/bin/bibtex.original to provide /usr/bin/bibtex
(bibtex) in auto mode
Setting up lmodern (2.004.5-6.1) …
Setting up texlive-base (2021.20220204-1) …
/usr/bin/ucfr
/usr/bin/ucfr
/usr/bin/ucfr
/usr/bin/ucfr
mktexlsr: Updating /var/lib/texmf/ls-R-TEXLIVEDIST…
mktexlsr: Updating /var/lib/texmf/ls-R-TEXMFMAIN…
mktexlsr: Updating /var/lib/texmf/ls-R…
mktexlsr: Done.
tl-paper: setting paper size for dvips to a4:
```

```
/var/lib/texmf/dvips/config/config-paper.ps
tl-paper: setting paper size for dvipdfmx to a4:
/var/lib/texmf/dvipdfmx/dvipdfmx-paper.cfg
tl-paper: setting paper size for xdvi to a4: /var/lib/texmf/xdvi/XDvi-paper
tl-paper: setting paper size for pdftex to a4: /var/lib/texmf/tex/generic/tex-
ini-files/pdftexconfig.tex
Setting up tex-gyre (20180621-3.1) …
Setting up texlive-plain-generic (2021.20220204-1) …
Setting up texlive-latex-base (2021.20220204-1) …
Setting up texlive-latex-recommended (2021.20220204-1) …
Setting up texlive-pictures (2021.20220204-1) …
Setting up texlive-fonts-recommended (2021.20220204-1) …
Setting up tipa (2:1.3-21) …
Setting up texlive (2021.20220204-1) …
Setting up texlive-latex-extra (2021.20220204-1) …
Setting up texlive-xetex (2021.20220204-1) …
Setting up rake (13.0.6-2) …
Setting up libruby3.0:amd64 (3.0.2-7ubuntu2.11) …
Setting up ruby3.0 (3.0.2-7ubuntu2.11) …
Setting up ruby (1:3.0~exp1) …
Setting up ruby-rubygems (3.3.5-2ubuntu1.2) …
Processing triggers for man-db (2.10.2-1) …
Processing triggers for mailcap (3.70+nmu1ubuntu1) …
Processing triggers for fontconfig (2.13.1-4.2ubuntu5) …
Processing triggers for libc-bin (2.35-0ubuntu3.8) …
/sbin/ldconfig.real: /usr/local/lib/libur_loader.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libhwloc.so.15 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic
link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_opencl.so.0 is not a symbolic
link

/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtcm.so.1 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_level_zero.so.0 is not a
symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtcm_debug.so.1 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link
```

```
/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_level_zero_v2.so.0 is not a
symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libumf.so.1 is not a symbolic link

Processing triggers for tex-common (6.17) …
Running updmap-sys. This may take some time… done.
Running mktexlsr /var/lib/texmf … done.
Building format(s) --all.
        This may take some time… done.
Collecting pypandoc
  Downloading pypandoc-1.15-py3-none-any.whl.metadata (16 kB)
Downloading pypandoc-1.15-py3-none-any.whl (21 kB)
Installing collected packages: pypandoc
Successfully installed pypandoc-1.15
```

[136]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[ ]:
```
!ls "/content/drive/My Drive/Colab Notebooks"
```

```
AerofitCase_HarishSV.ipynb    NetflixCase_HarishSV.ipynb
DelhiveryCase_HarishSV.ipynb  NetflixEDA_HarishSV.ipynb
Jamboree_HarishSV.ipynb       NetflixEDA_HarishSV.pdf
Jamboree_HarishSV.pdf         ScalerClustering_HarishSV.ipynb
LoanTap_HarishSV.ipynb        ScalerClustering_HarishSV.pdf
LoanTap_HarishSV.pdf          WalmartCase_HarishSV.ipynb
LoanTapML_HarishSV.ipynb      YuluCase_HarishSV.ipynb
```

[ ]:
```
!jupyter nbconvert --to pdf "/content/drive/My Drive/Colab Notebooks/Untitled0.
↪ipynb"
```

```
[NbConvertApp] Converting notebook /content/drive/My Drive/Colab
Notebooks/Untitled0.ipynb to pdf
[NbConvertApp] Writing 38414 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
```

```
[NbConvertApp] Writing 34848 bytes to /content/drive/My Drive/Colab
Notebooks/Untitled0.pdf
```