```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df=pd.read_csv("/content/drive/MyDrive/logistic_regression.csv")
```

```
# Making an copy of dataset
loantap=df.copy()
loantap.head()
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_length	home_ownership	annual_inc	...	open_acc	pu
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+ years	RENT	117000.0	...	16.0	
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4 years	MORTGAGE	65000.0	...	17.0	
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	< 1 year	RENT	43057.0	...	13.0	
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6 years	RENT	54000.0	...	6.0	
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9 years	MORTGAGE	55000.0	...	13.0	

5 rows × 27 columns

Exploratory of Data Analysis

```
loantap.shape
```

```
(396030, 27)
```

```
#data info
loantap.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
```

```
#Checking Null values
loantap.isnull().sum()
```

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

```
# percentage of null values in each colum
round(loantap.isnull().sum()/len(loantap)* 100,3)
```

```
loan_amnt               0.000
term                    0.000
int_rate                0.000
installment             0.000
grade                   0.000
sub_grade               0.000
emp_title               5.789
emp_length              4.621
home_ownership          0.000
annual_inc              0.000
verification_status     0.000
issue_d                 0.000
loan_status             0.000
purpose                 0.000
title                   0.443
dti                     0.000
earliest_cr_line        0.000
open_acc                0.000
pub_rec                 0.000
revol_bal               0.000
revol_util              0.070
total_acc               0.000
initial_list_status     0.000
application_type        0.000
mort_acc                9.543
pub_rec_bankruptcies    0.135
address                 0.000
dtype: float64
```

Analysizing Basic Metrics

```
loantap.describe().T
```

	count	mean	std	min	25%	50%	75%	max
loan_amnt	396030.0	14113.888089	8357.441341	500.00	8000.00	12000.00	20000.00	40000.00
int_rate	396030.0	13.639400	4.472157	5.32	10.49	13.33	16.49	30.99
installment	396030.0	431.849698	250.727790	16.08	250.33	375.43	567.30	1533.81
annual_inc	396030.0	74203.175798	61637.621158	0.00	45000.00	64000.00	90000.00	8706582.00
dti	396030.0	17.379514	18.019092	0.00	11.28	16.91	22.98	9999.00
open_acc	396030.0	11.311153	5.137649	0.00	8.00	10.00	14.00	90.00
pub_rec	396030.0	0.178191	0.530671	0.00	0.00	0.00	0.00	86.00
revol_bal	396030.0	15844.539853	20591.836109	0.00	6025.00	11181.00	19620.00	1743266.00
revol_util	395754.0	53.791749	24.452193	0.00	35.80	54.80	72.90	892.30
total_acc	396030.0	25.414744	11.886991	2.00	17.00	24.00	32.00	151.00
mort_acc	358235.0	1.813991	2.147930	0.00	0.00	1.00	3.00	34.00
pub_rec_bankruptcies	395495.0	0.121648	0.356174	0.00	0.00	0.00	0.00	8.00

Insights

Outlier: The significant difference between mean and median and Standard deviation indicate key attribute like loan amount,annual inc,revol_bal has outlier. **Loan Duration Preference**: A preference for 36-month loan terms among borrowers suggests a balance between manageable installments.

Home Ownership Trends: The prevalence of applicants with mortgaged homes suggests financial stability or a need for substantial, property-secured loans.

Successful Loan Repayment: Most loans being fully paid off reflects positively on borrowers' financial commitment, indicating effective lending criteria.

Debt Consolidation Dominance: The primary use of loans for debt consolidation highlights a common strategy to manage or reduce high-interest debt.

Individual Borrowers: The predominance of individual applicants suggests that personal loans are a major market segment.

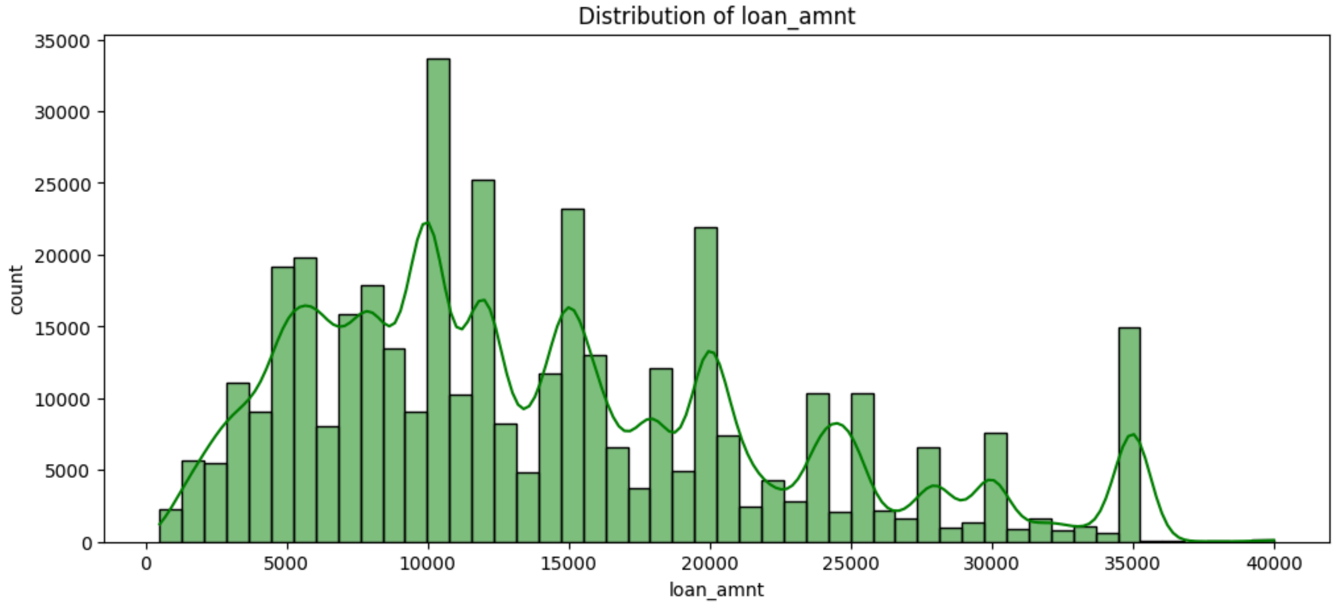
Application Type:Almost all applications are from individuals, with very few joint applications.

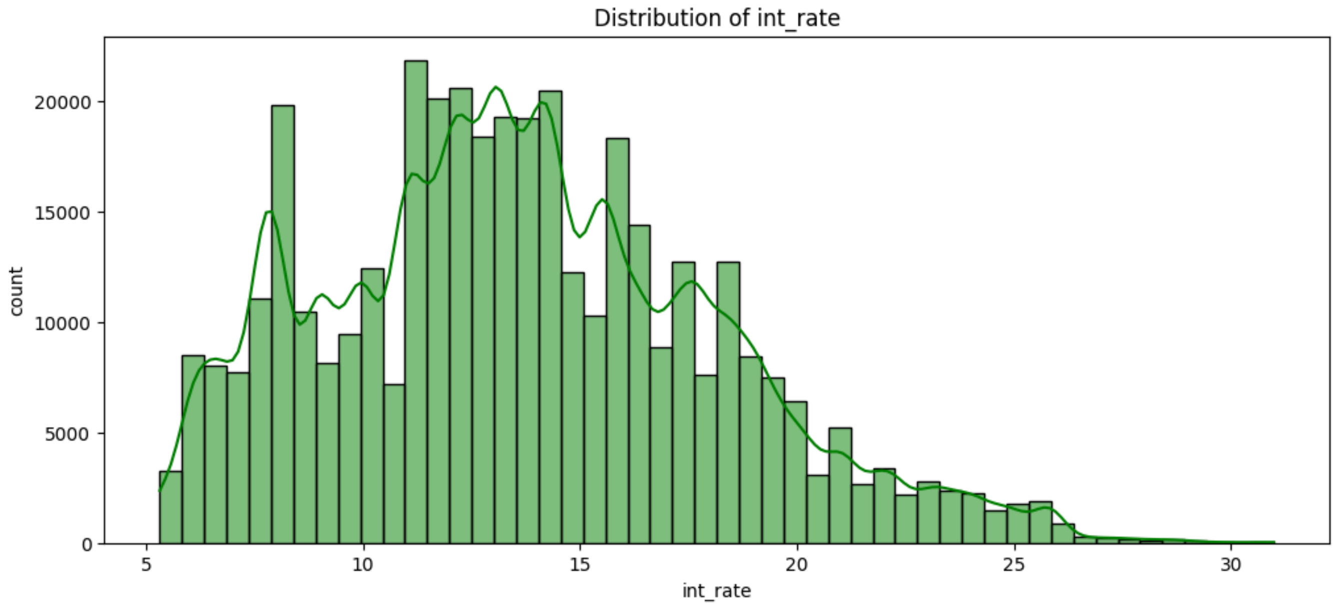
```
#separating float and object columns
n_column=loantap.select_dtypes('float64').columns.tolist()
n_column
```

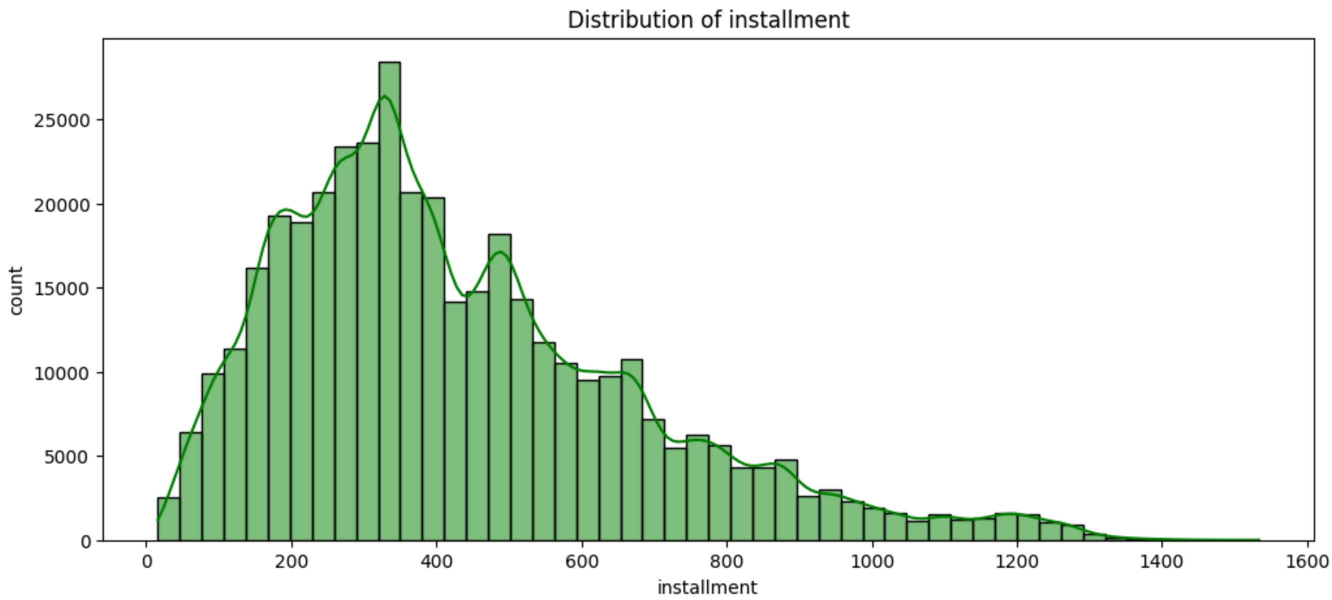
```
['loan_amnt',
 'int_rate',
 'installment',
 'annual_inc',
 'dti',
 'open_acc',
 'pub_rec',
 'revol_bal',
 'revol_util',
 'total_acc',
 'mort_acc',
 'pub_rec_bankruptcies']
```

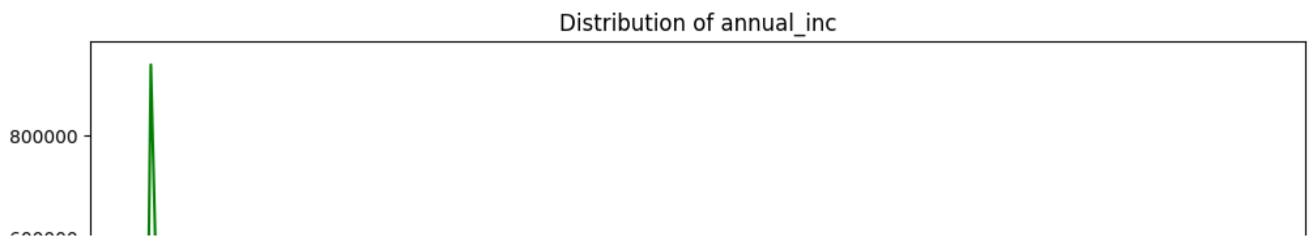
```
n_categorical=['home_ownership', 'verification_status', 'loan_status', 'application_type', 'grade', 'sub_grade', 'term']
```

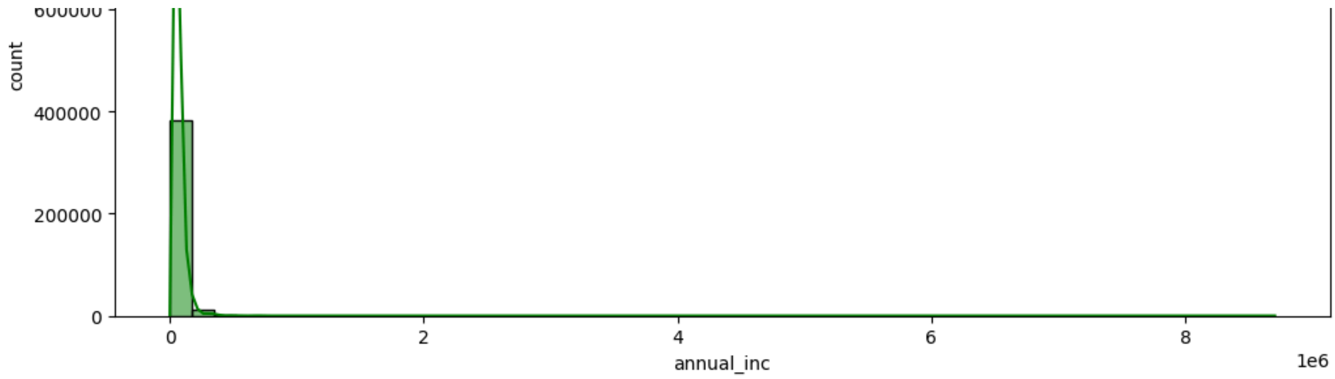
```
for i in n_column:
  plt.figure(figsize=(12,5))
  sns.histplot(data=loantap,x=i,kde=True,bins=50,color="green")
  plt.xlabel(i)
  plt.ylabel("count")
  plt.title("Distribution of "+i)
  plt.show()
```

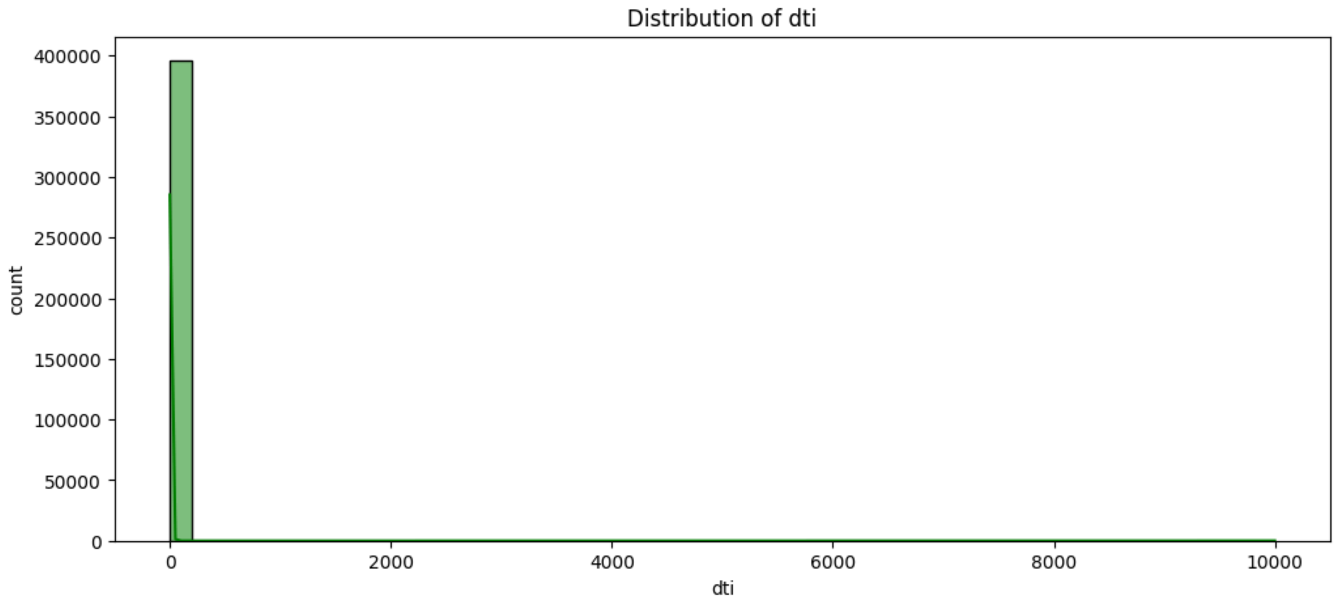

Distribution of loan_amnt
count
loan_amnt
0
5000
10000
15000
20000
25000
30000
35000
40000

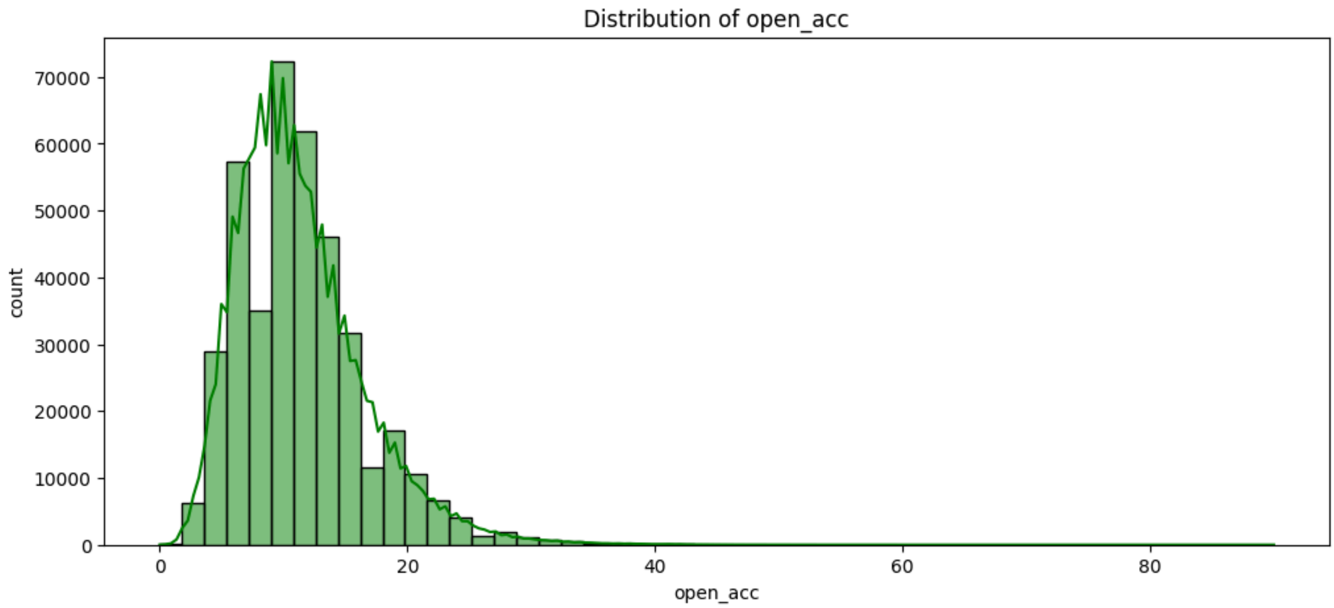

Distribution of int_rate
count
int_rate
0
5000
10000
15000
20000
5
10
15
20
25
30


Distribution of installment
count
installment
0
5000
10000
15000
20000
25000
200
400
600
800
1000
1200
1400
1600

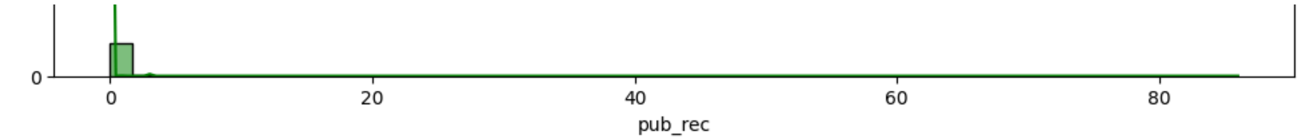

Distribution of annual_inc
800000


count
annual_inc
1e6

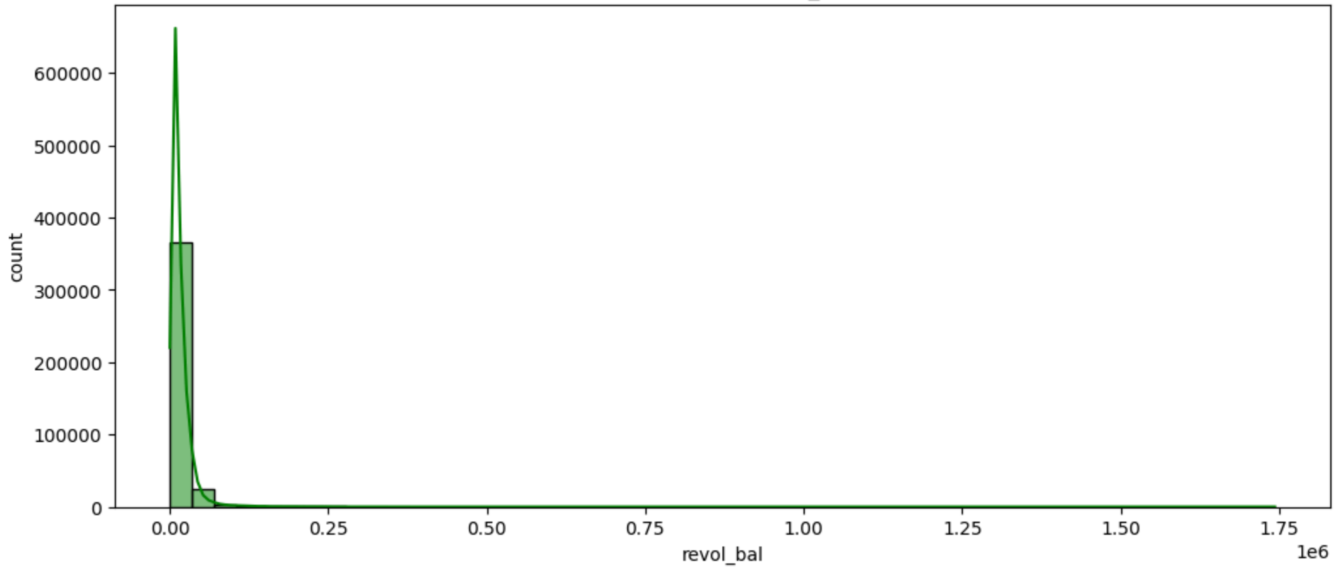

Distribution of dti
count
dti


Distribution of open_acc
count
open_acc

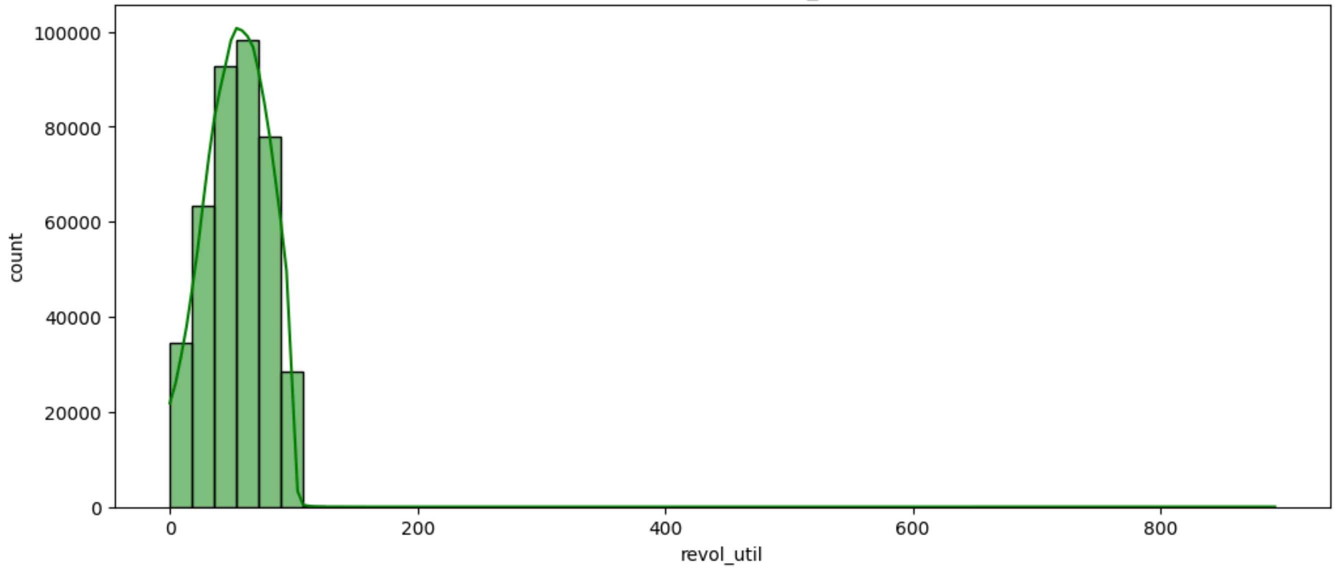

Distribution of pub_rec
1e6
count



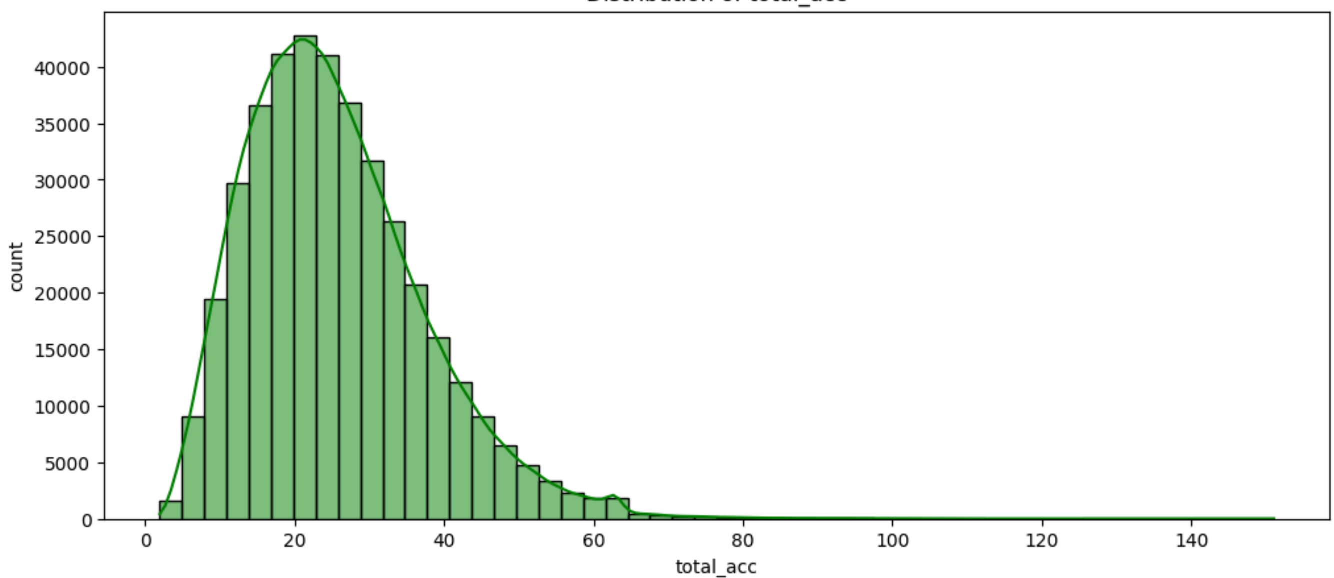
Distribution of revol_bal



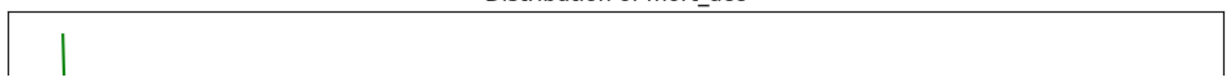
Distribution of revol_util

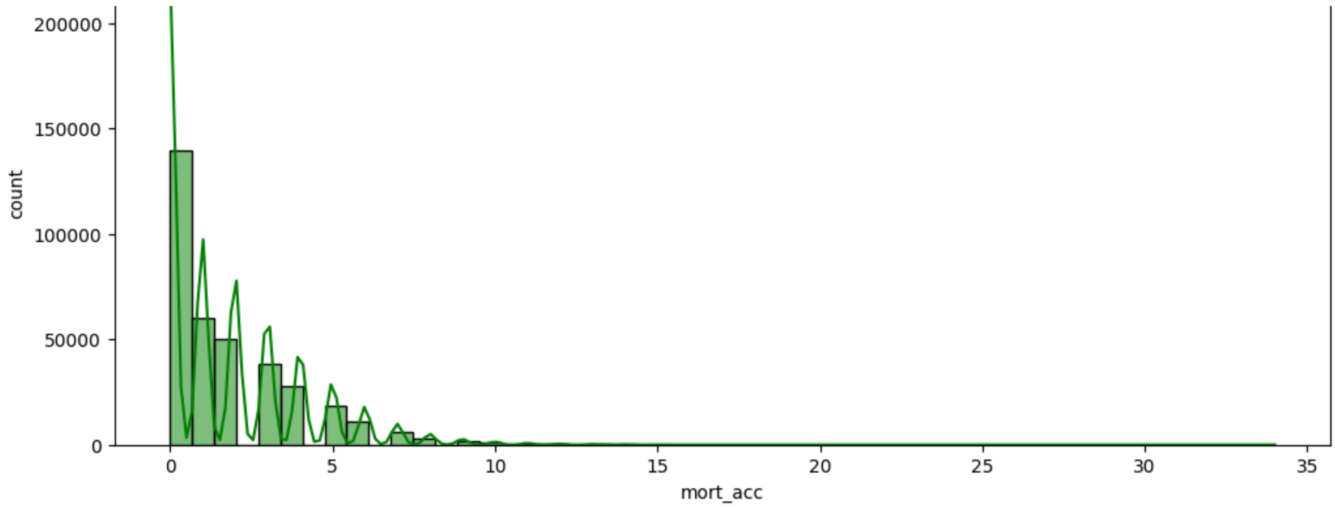


Distribution of total_acc

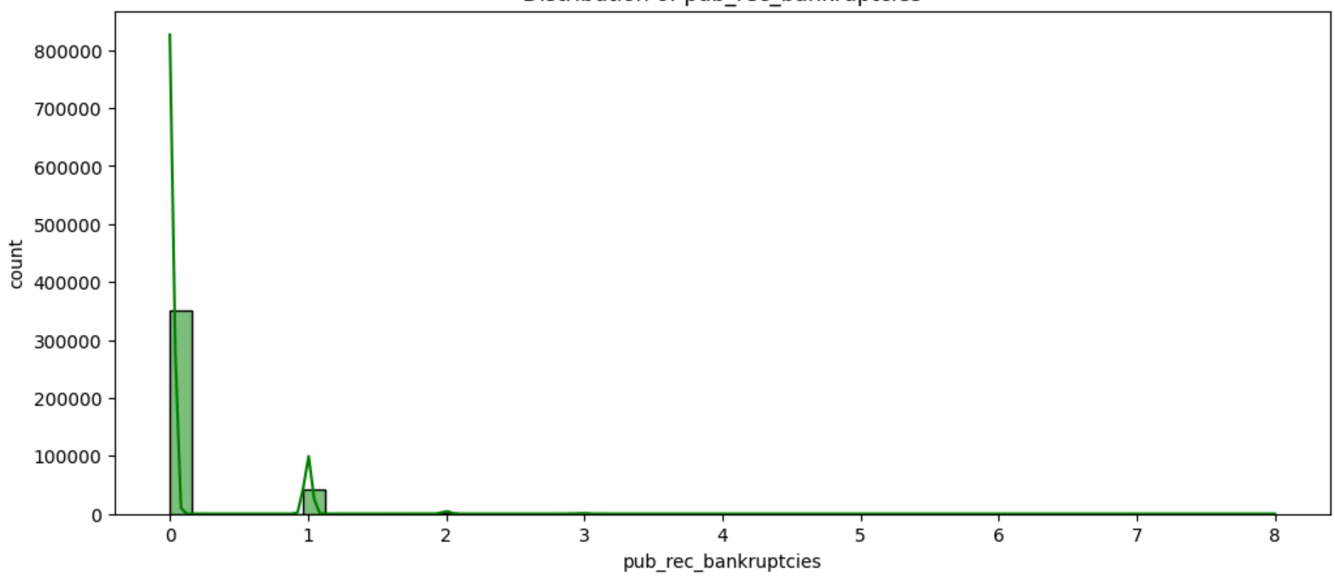


Distribution of mort_acc





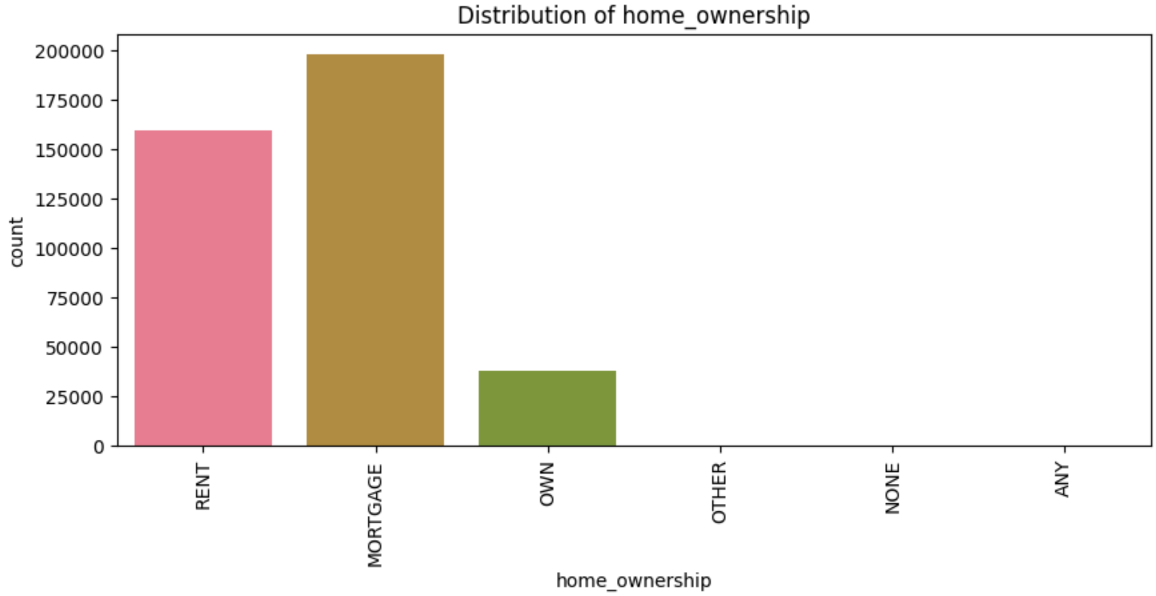
Distribution of pub_rec_bankruptcies



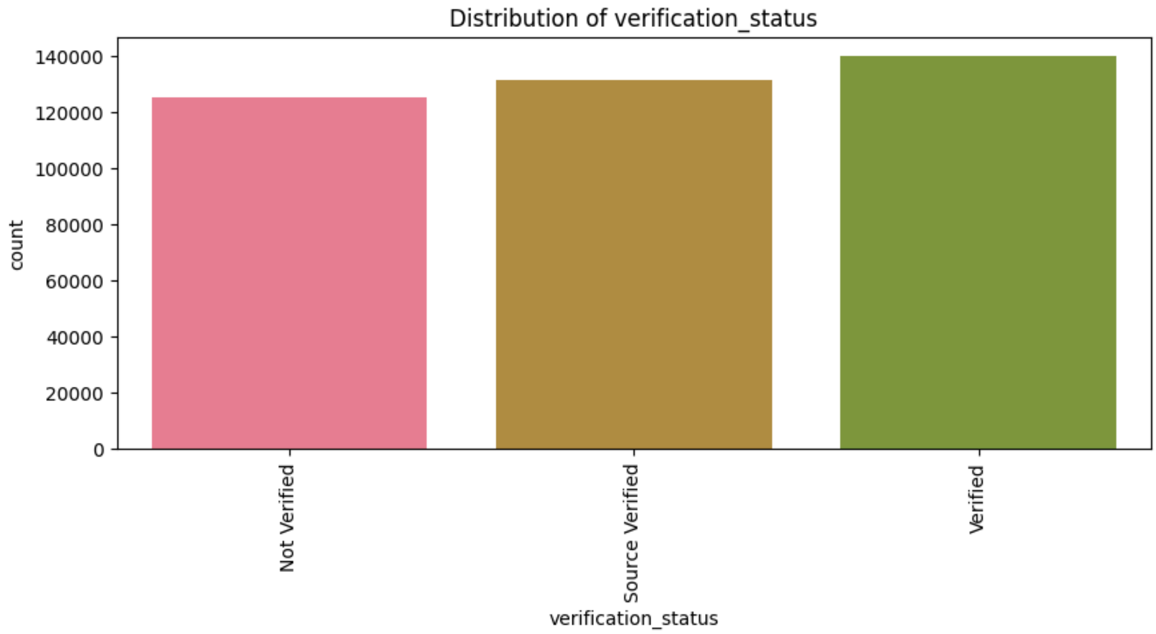

```
custom_palette=sns.color_palette("husl", 7)


for i in n_categorical:
  plt.figure(figsize=(10,4))
  sns.countplot(data=loantap,x=i,palette=custom_palette,hue=i,legend=False)
  plt.xlabel(i)
  plt.ylabel("count")
  plt.title("Distribution of "+i)
  plt.xticks(rotation=90)
  plt.show()
```

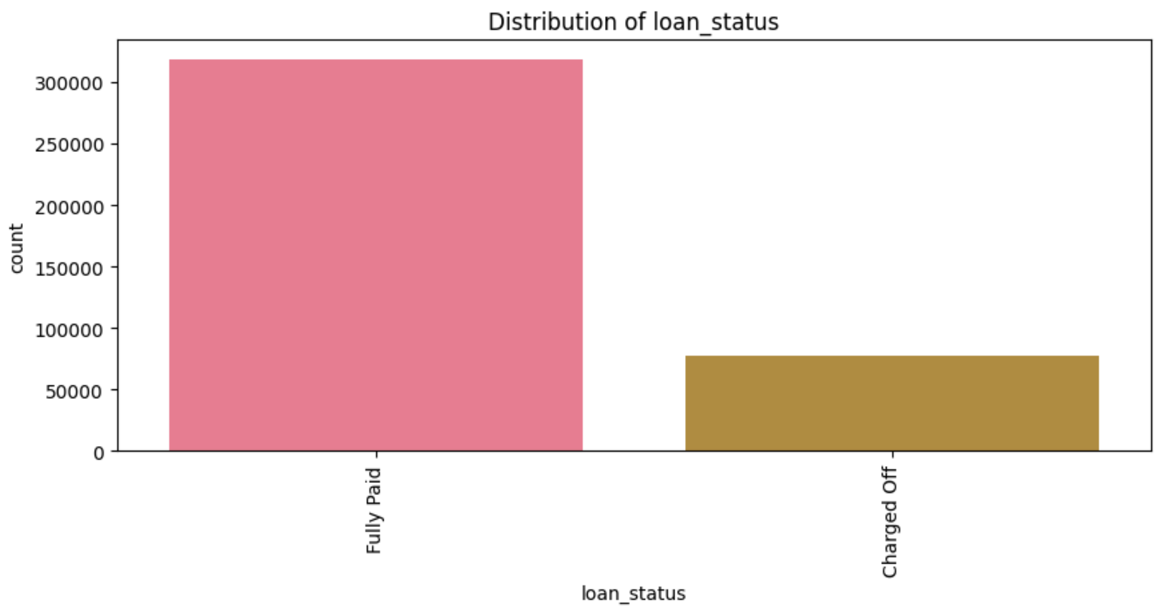
```
<ipython-input-17-9913e48f2fa0>:3: UserWarning: The palette list has more values (7) than needed (6), which may not be intended.
  sns.countplot(data=loantap,x=i,palette=custom_palette,hue=i,legend=False)
```




```
<ipython-input-17-9913e48f2fa0>:3: UserWarning: The palette list has more values (7) than needed (3), which may not be intended.
  sns.countplot(data=loantap,x=i,palette=custom_palette,hue=i,legend=False)
```

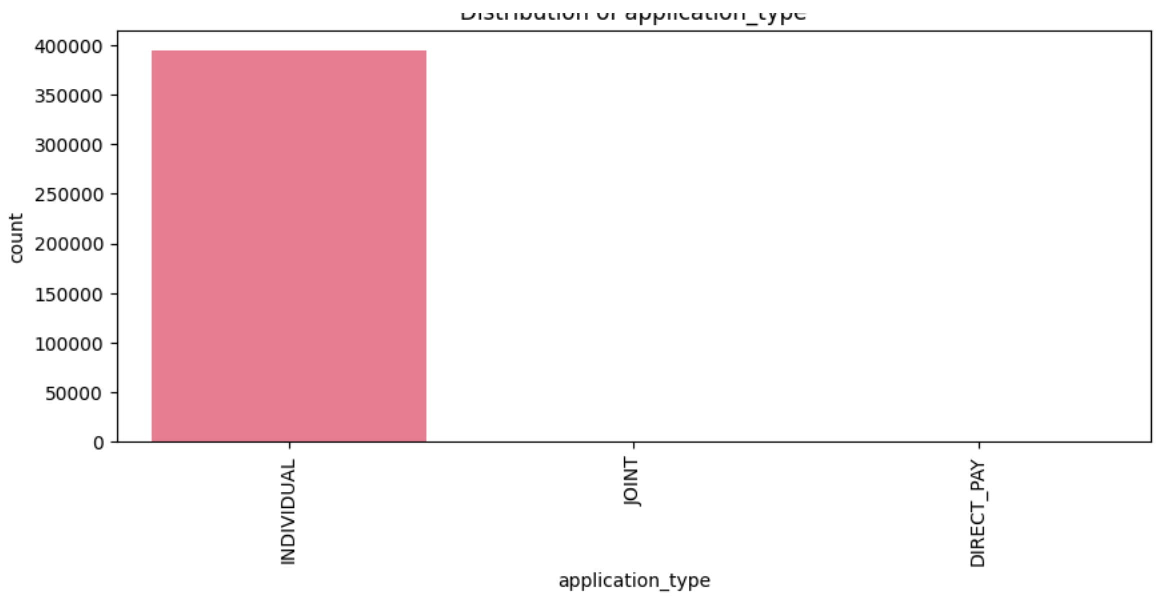


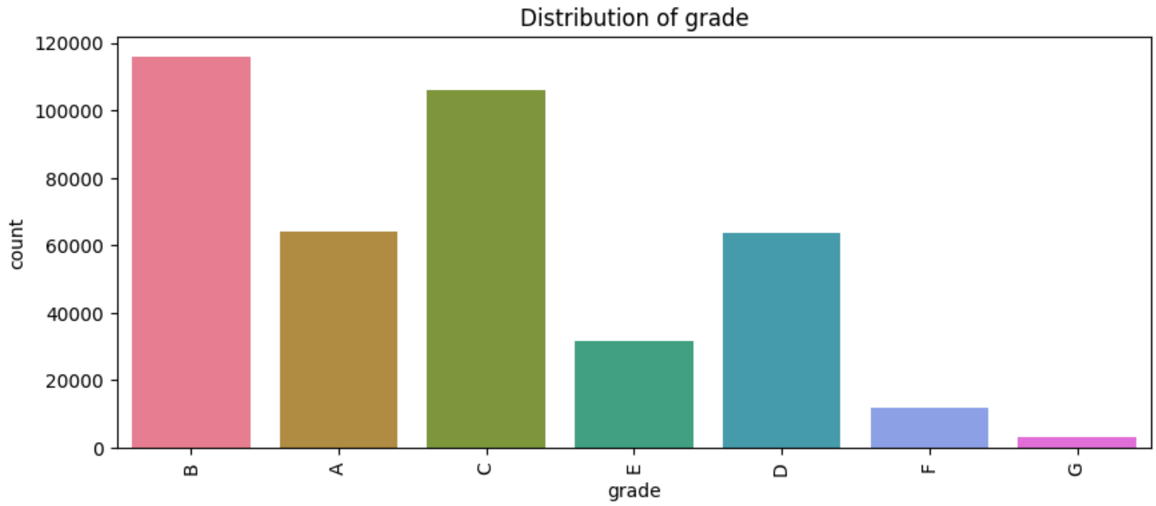

```
<ipython-input-17-9913e48f2fa0>:3: UserWarning: The palette list has more values (7) than needed (2), which may not be intended.
  sns.countplot(data=loantap,x=i,palette=custom_palette,hue=i,legend=False)
```



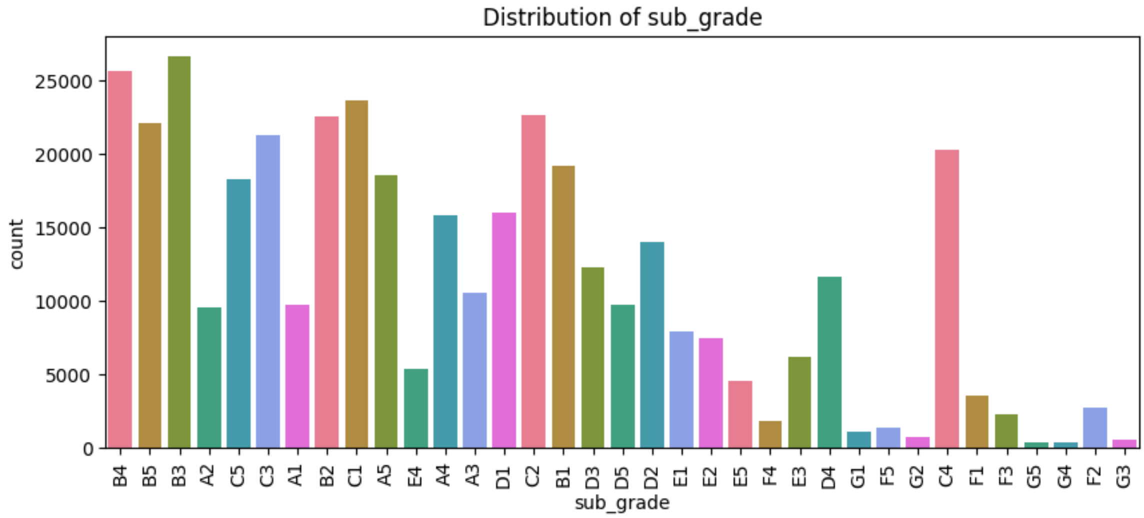

```
<ipython-input-17-9913e48f2fa0>:3: UserWarning: The palette list has more values (7) than needed (3), which may not be intended.
  sns.countplot(data=loantap,x=i,palette=custom_palette,hue=i,legend=False)
```

Distribution of application_type

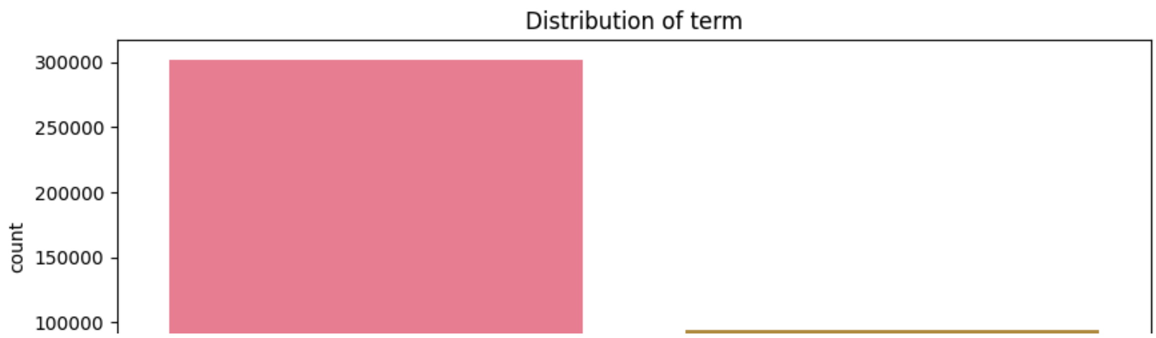


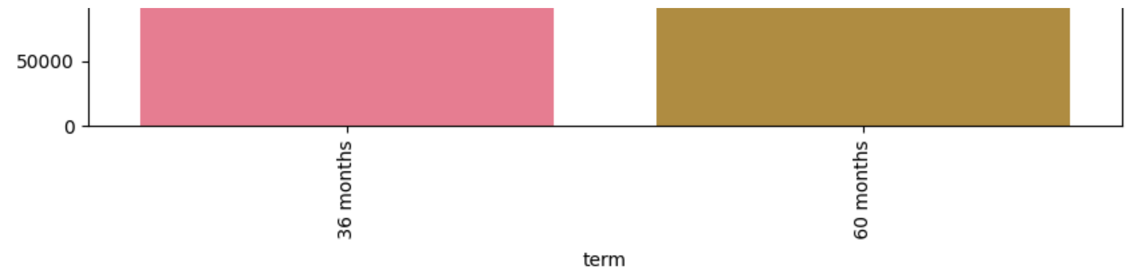



```
<ipython-input-17-9913e48f2fa0>:3: UserWarning: 
The palette list has fewer values (7) than needed (35) and will cycle, which may produce an uninterpretable plot.
  sns.countplot(data=loantap,x=i,palette=custom_palette,hue=i,legend=False)
```




```
<ipython-input-17-9913e48f2fa0>:3: UserWarning: The palette list has more values (7) than needed (2), which may not be intended.
  sns.countplot(data=loantap,x=i,palette=custom_palette,hue=i,legend=False)
```





Bivarient Analysis

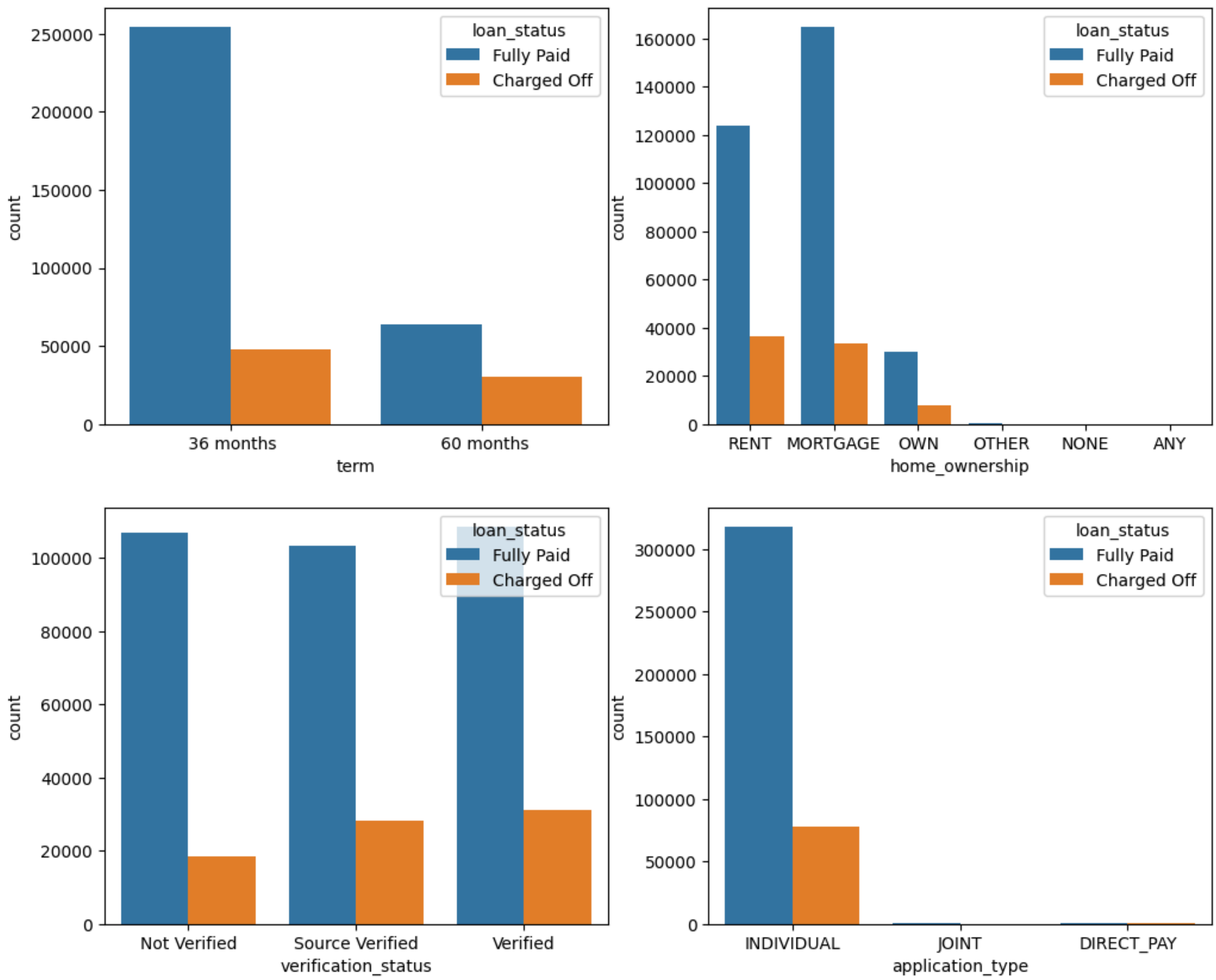
```
plt.figure(figsize=(12,10))

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

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

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

plt.subplot(2,2,4)
sns.countplot(data=loantap,x='application_type',hue='loan_status')
plt.show()
```




```
grade=sorted(loantap['grade'].unique())
grade
```

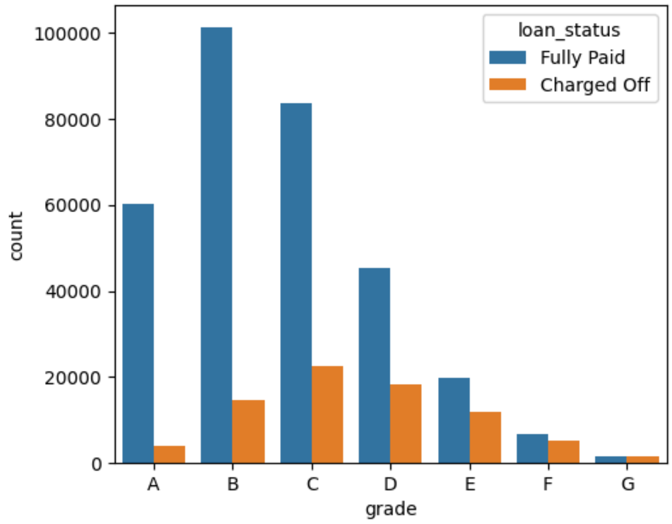
```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

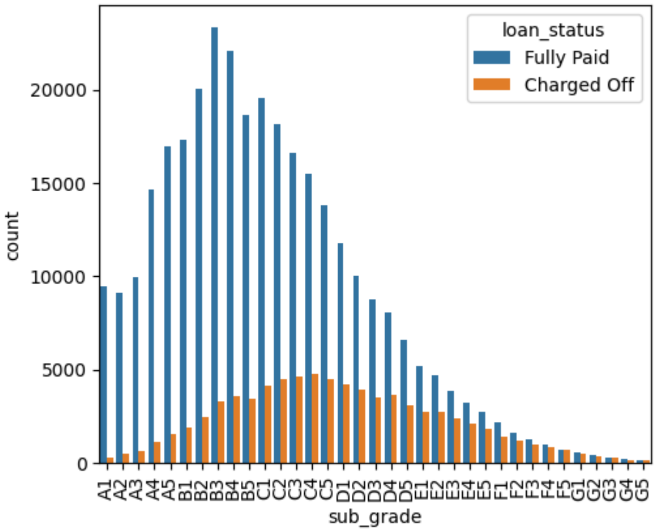
```
plt.figure(figsize=(12,10))

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


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

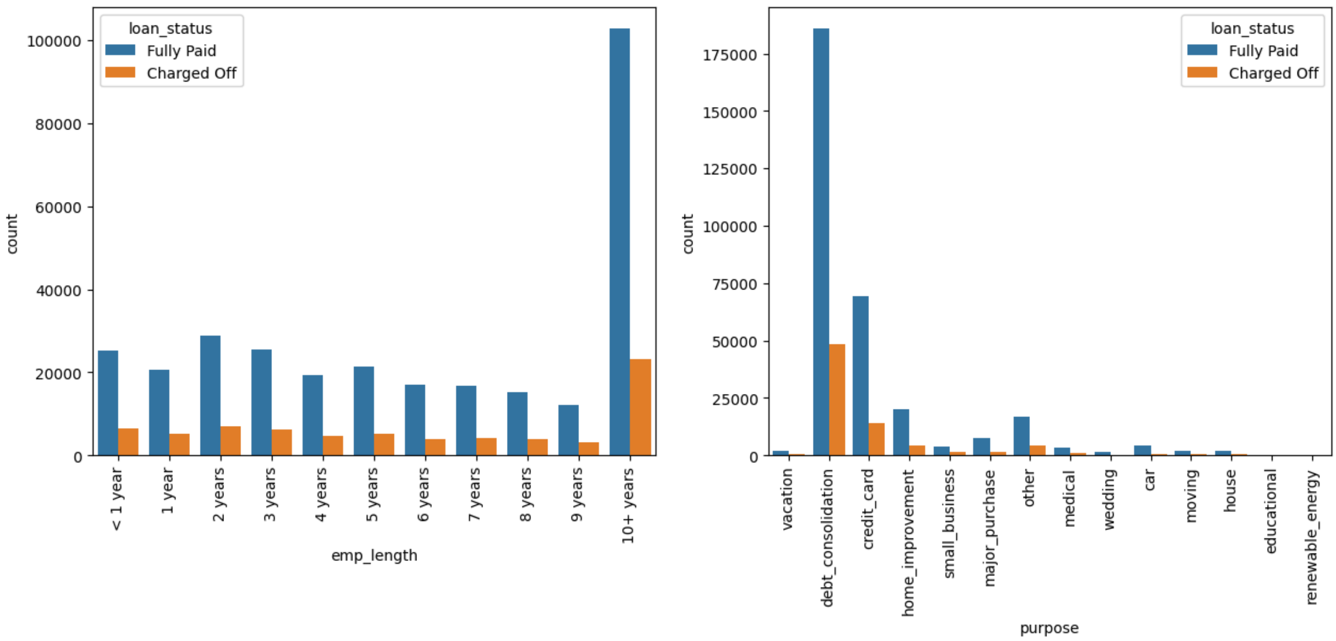
```
<ipython-input-20-d58c52b94201>:11: UserWarning: FixedFormatter should only be used together with FixedLocator
  g.set_xticklabels(g.get_xticklabels(),rotation=90)
```






```
plt.figure(figsize=(15,12))
plt.subplot(2,2,1)
order = ['< 1 year', '1 year', '2 years', '3 years', '4 years', '5 years',
          '6 years', '7 years', '8 years', '9 years', '10+ years',]
sns.countplot(data=loantap,x='emp_length',hue='loan_status',order=order)
x=plt.xticks(rotation=90)

plt.subplot(2,2,2)
sns.countplot(data=loantap,x='purpose',hue='loan_status')
x=plt.xticks(rotation=90)
```



Insights

Loan term : 36 month loan term has high completion rate.

Borrower Situation: Mortgages and rental are the most common borrwer of loan with high completion rate.

Creditworthiness: Borrowers with a credit grade of "B" and a subgrade of "B3" tend to have the highest repayment rates.

Occupations: Managers and teachers are the professions with the highest loan approval rates.

Repayment: Individuals employed for over 10 years demonstrate a strong track record of loan repayment.

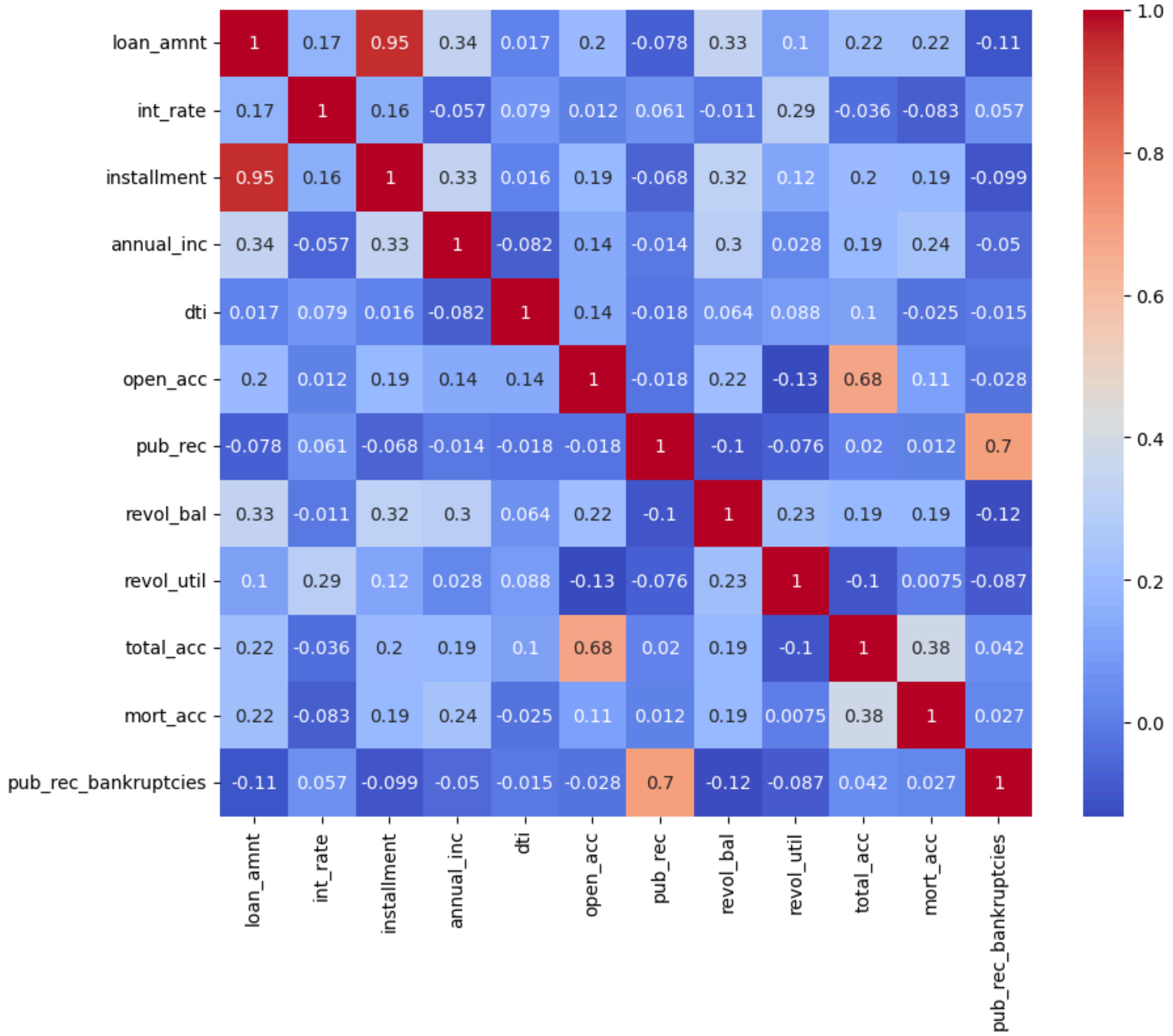
Correlation Analysis

```
loantap.corr(numeric_only=True)
```

	loan_amnt	int_rate	installment	annual_inc	dti	open_acc	pub_rec	revol_bal	revol_util	total_acc	mo
loan_amnt	1.000000	0.168921	0.953929	0.336887	0.016636	0.198556	-0.077779	0.328320	0.099911	0.223886	0.
int_rate	0.168921	1.000000	0.162758	-0.056771	0.079038	0.011649	0.060986	-0.011280	0.293659	-0.036404	-0.
installment	0.953929	0.162758	1.000000	0.330381	0.015786	0.188973	-0.067892	0.316455	0.123915	0.202430	0.
annual_inc	0.336887	-0.056771	0.330381	1.000000	-0.081685	0.136150	-0.013720	0.299773	0.027871	0.193023	0.
dti	0.016636	0.079038	0.015786	-0.081685	1.000000	0.136181	-0.017639	0.063571	0.088375	0.102128	-0.
open_acc	0.198556	0.011649	0.188973	0.136150	0.136181	1.000000	-0.018392	0.221192	-0.131420	0.680728	0.
pub_rec	-0.077779	0.060986	-0.067892	-0.013720	-0.017639	-0.018392	1.000000	-0.101664	-0.075910	0.019723	0.
revol_bal	0.328320	-0.011280	0.316455	0.299773	0.063571	0.221192	-0.101664	1.000000	0.226346	0.191616	0.
revol_util	0.099911	0.293659	0.123915	0.027871	0.088375	-0.131420	-0.075910	0.226346	1.000000	-0.104273	0.
total_acc	0.223886	-0.036404	0.202430	0.193023	0.102128	0.680728	0.019723	0.191616	-0.104273	1.000000	0.
mort_acc	0.222315	-0.082583	0.193694	0.236320	-0.025439	0.109205	0.011552	0.194925	0.007514	0.381072	1.
pub_rec_bankruptcies	-0.106539	0.057450	-0.098628	-0.050162	-0.014558	-0.027732	0.699408	-0.124532	-0.086751	0.042035	0.

```
plt.figure(figsize=(10,8))
sns.heatmap(loantap.corr(numeric_only=True),annot=True,cmap='coolwarm')
```

<Axes: >



Insigths

Positive Correlation :

- Loan amount and installment has obvious correlation around 0.95
- Negative record of borrower credit profile(pub_rec) correlation with Backruptcy record of borrower(pub_rec_bankruptcies) correlation around 0.7
- Open accunt and Pub_rec have strong correlation around 0.68

Data Processing using Feature Engineering

```
def pub_rec(number):
  if number==0.0:
    return 0
  else:
    return 1
def mort_acc(number):
  if number==0.0:
    return 0
  elif number > 1.0:
    return 1
  else:
    return number
def pub_rec_bankruptcies(number):
  if number==0.0:
    return 0
  elif number > 1.0:
    return 1
  else:
    return number
```

```
loantap['pub_rec']=loantap['pub_rec'].apply(pub_rec)
loantap['mort_acc']=loantap['mort_acc'].apply(mort_acc)
loantap['pub_rec_bankruptcies']=loantap['pub_rec_bankruptcies'].apply(pub_rec_bankruptcies)
```

```
loantap[['pub_rec','mort_acc','pub_rec_bankruptcies']].nunique()
```

```
pub_rec                 2
mort_acc                2
pub_rec_bankruptcies    2
dtype: int64
```

Duplicate checks

```
#duplicate check
loantap.duplicated().sum()
```

```
0
```

```
#Missing value
loantap.isnull().sum()
```

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

```
numeric_columns=loantap.select_dtypes('float64','integer')

total_acc_avg=numeric_columns.groupby('total_acc')['mort_acc'].mean()
# filling mort_acc null value with mean
def fill_mort_acc(total_acc,mort_acc):
  if np.isnan(mort_acc):
    return total_acc_avg[total_acc]
  else:
    return mort_acc

loantap['mort_acc']=loantap.apply(lambda x: fill_mort_acc(x['total_acc'],x['mort_acc']) ,axis=1)
```

```
# mort_acc null value get filled
loantap.isnull().sum()
```

```
loan_amnt                   0
term                        0
int_rate                    0
installment                 0
grade                       0
sub_grade                   0
emp_title               22927
emp_length              18301
home_ownership              0
annual_inc                  0
verification_status         0
issue_d                     0
loan_status                 0
purpose                     0
title                    1756
```

```
dti                          0
earliest_cr_line             0
open_acc                     0
pub_rec                      0
revol_bal                    0
revol_util                 276
total_acc                    0
initial_list_status          0
application_type             0
mort_acc                     0
pub_rec_bankruptcies       535
address                      0
dtype: int64
```

```
loantap.shape
```

```
(396030, 27)
```

```
#Dropping remaining null values
loantap.dropna(inplace=True)
loantap.shape
```

```
(370621, 27)
```

Outlier Detection

```
loantap.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')
```

```
def box_plot(column):
  if column in n_column:
    plt.figure(figsize=(10,5))
    sns.boxplot(data=loantap,x=column)
    plt.title("Box Plot of "+column)
    plt.show()
  else:
    print("Not a numeric column")
```

```
for col in n_column:
  box_plot(col)
```