Sonal-jatav-lending-club-case-study

0.1 Lending Club Case Study

0.1.1 Objective:

EDA on loan dataset to analyse factors predominant in defaulting loan repayment

Problem Statement: A consumer finance company that lends different loans to urban customers wants to be able to assess the risks related to granting loan to a customer.

Based on `risk assessment` company should be able to identify and `accept` or `reject` loan application. 1. How likely a person is to default a loan and is a bad candidate to grant loan 2. If a person is high risk then should the interest rate be increased 3. Is the customer a good candidate to lend

GOAL Reduce `credit loss` for the company by evaluating the `driving factors/ variables` for defaulting loan repayment

---

Steps

1. Understand Data
2. Data cleaning - remove outliers, null values, fix data types
3. Data Analysis - Univariate, Bivariate, Multivariate
4. Visualise results
5. Conclusion

Import Libraries

[1]:
```
#Load the necessary Libraries
import pandas as pd
import numpy as np
# Charts and plots
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
# Warnings library
import warnings #warning
warnings.filterwarnings('ignore')
```

```
#Removing display limit of dataframe (optional cell to run)
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
```

0.1.2 Load Data

[2]:
```
# Load Data
loan = pd.read_csv('loan.csv')
data_dict = pd.read_excel('Data_Dictionary.xlsx')

# loan.head()
```

[3]:
```
# Shape of dataset
loan.shape
```

[3]:
```
(39717, 111)
```

[4]:
```
# Data types of all columns
loan.dtypes
```

[4]:
```
id                          int64
member_id                   int64
loan_amnt                   int64
funded_amnt                 int64
funded_amnt_inv           float64
term                       object
int_rate                   object
installment               float64
grade                      object
sub_grade                  object
emp_title                  object
emp_length                 object
home_ownership             object
annual_inc                float64
verification_status        object
issue_d                    object
loan_status                object
pymnt_plan                 object
url                        object
desc                       object
purpose                    object
title                      object
zip_code                   object
addr_state                 object
```

```
dti                                float64
delinq_2yrs                          int64
earliest_cr_line                    object
inq_last_6mths                       int64
mths_since_last_delinq             float64
mths_since_last_record             float64
open_acc                             int64
pub_rec                              int64
revol_bal                            int64
revol_util                          object
total_acc                            int64
initial_list_status                 object
out_prncp                          float64
out_prncp_inv                      float64
total_pymnt                        float64
total_pymnt_inv                    float64
total_rec_prncp                    float64
total_rec_int                      float64
total_rec_late_fee                 float64
recoveries                         float64
collection_recovery_fee            float64
last_pymnt_d                        object
last_pymnt_amnt                    float64
next_pymnt_d                        object
last_credit_pull_d                  object
collections_12_mths_ex_med         float64
mths_since_last_major_derog        float64
policy_code                          int64
application_type                    object
annual_inc_joint                   float64
dti_joint                          float64
verification_status_joint          float64
acc_now_delinq                       int64
tot_coll_amt                       float64
tot_cur_bal                        float64
open_acc_6m                        float64
open_il_6m                         float64
open_il_12m                        float64
open_il_24m                        float64
mths_since_rcnt_il                 float64
total_bal_il                       float64
il_util                            float64
open_rv_12m                        float64
open_rv_24m                        float64
max_bal_bc                         float64
all_util                           float64
total_rev_hi_lim                   float64
```

```
inq_fi                              float64
total_cu_tl                         float64
inq_last_12m                        float64
acc_open_past_24mths                float64
avg_cur_bal                         float64
bc_open_to_buy                      float64
bc_util                             float64
chargeoff_within_12_mths            float64
delinq_amnt                           int64
mo_sin_old_il_acct                  float64
mo_sin_old_rev_tl_op                float64
mo_sin_rcnt_rev_tl_op               float64
mo_sin_rcnt_tl                      float64
mort_acc                            float64
mths_since_recent_bc                float64
mths_since_recent_bc_dlq            float64
mths_since_recent_inq               float64
mths_since_recent_revol_delinq      float64
num_accts_ever_120_pd               float64
num_actv_bc_tl                      float64
num_actv_rev_tl                     float64
num_bc_sats                         float64
num_bc_tl                           float64
num_il_tl                           float64
num_op_rev_tl                       float64
num_rev_accts                       float64
num_rev_tl_bal_gt_0                 float64
num_sats                            float64
num_tl_120dpd_2m                    float64
num_tl_30dpd                        float64
num_tl_90g_dpd_24m                  float64
num_tl_op_past_12m                  float64
pct_tl_nvr_dlq                      float64
percent_bc_gt_75                    float64
pub_rec_bankruptcies                float64
tax_liens                           float64
tot_hi_cred_lim                     float64
total_bal_ex_mort                   float64
total_bc_limit                      float64
total_il_high_credit_limit          float64
dtype: object
```

0.1.3 Cleanup Data

Clean null values, unused columns

[5]:
```
# Check null values
loan.isnull().sum().sort_values(ascending=False)
```

[5]:
```
verification_status_joint         39717
annual_inc_joint                  39717
mo_sin_old_rev_tl_op              39717
mo_sin_old_il_acct                39717
bc_util                           39717
bc_open_to_buy                    39717
avg_cur_bal                       39717
acc_open_past_24mths              39717
inq_last_12m                      39717
total_cu_tl                       39717
inq_fi                            39717
total_rev_hi_lim                  39717
all_util                          39717
max_bal_bc                        39717
open_rv_24m                       39717
open_rv_12m                       39717
il_util                           39717
total_bal_il                      39717
mths_since_rcnt_il                39717
open_il_24m                       39717
open_il_12m                       39717
open_il_6m                        39717
open_acc_6m                       39717
tot_cur_bal                       39717
tot_coll_amt                      39717
mo_sin_rcnt_rev_tl_op             39717
mo_sin_rcnt_tl                    39717
mort_acc                          39717
num_rev_tl_bal_gt_0               39717
total_bc_limit                    39717
total_bal_ex_mort                 39717
tot_hi_cred_lim                   39717
percent_bc_gt_75                  39717
pct_tl_nvr_dlq                    39717
num_tl_op_past_12m                39717
num_tl_90g_dpd_24m                39717
num_tl_30dpd                      39717
num_tl_120dpd_2m                  39717
num_sats                          39717
num_rev_accts                     39717
mths_since_recent_bc              39717
num_op_rev_tl                     39717
num_il_tl                         39717
num_bc_tl                         39717
```

num_bc_sats	39717
num_actv_rev_tl	39717
num_actv_bc_tl	39717
num_accts_ever_120_pd	39717
mths_since_recent_revol_delinq	39717
mths_since_recent_inq	39717
mths_since_recent_bc_dlq	39717
dti_joint	39717
total_il_high_credit_limit	39717
mths_since_last_major_derog	39717
next_pymnt_d	38577
mths_since_last_record	36931
mths_since_last_delinq	25682
desc	12942
emp_title	2459
emp_length	1075
pub_rec_bankruptcies	697
last_pymnt_d	71
collections_12_mths_ex_med	56
chargeoff_within_12_mths	56
revol_util	50
tax_liens	39
title	11
last_credit_pull_d	2
pymnt_plan	0
url	0
loan_status	0
purpose	0
issue_d	0
verification_status	0
application_type	0
annual_inc	0
home_ownership	0
zip_code	0
grade	0
installment	0
int_rate	0
term	0
funded_amnt_inv	0
funded_amnt	0
loan_amnt	0
sub_grade	0
inq_last_6mths	0
addr_state	0
dti	0
member_id	0
acc_now_delinq	0

```
last_pymnt_amnt                 0
collection_recovery_fee         0
recoveries                      0
total_rec_late_fee              0
total_rec_int                   0
total_rec_prncp                 0
total_pymnt_inv                 0
total_pymnt                     0
out_prncp_inv                   0
out_prncp                       0
initial_list_status             0
total_acc                       0
revol_bal                       0
pub_rec                         0
open_acc                        0
delinq_amnt                     0
policy_code                     0
earliest_cr_line                0
delinq_2yrs                     0
id                              0
dtype: int64
```

[6]:
```
# Remove columns with all null values

loan = loan.loc[:, ~loan.isnull().all()]
print(loan.shape)

print(loan.isnull().sum().sort_values(ascending=False))
```

```
(39717, 57)
next_pymnt_d                 38577
mths_since_last_record       36931
mths_since_last_delinq       25682
desc                         12942
emp_title                     2459
emp_length                    1075
pub_rec_bankruptcies           697
last_pymnt_d                    71
chargeoff_within_12_mths        56
collections_12_mths_ex_med      56
revol_util                      50
tax_liens                       39
title                           11
last_credit_pull_d               2
home_ownership                   0
int_rate                         0
out_prncp_inv                    0
```

```
total_pymnt                   0
total_pymnt_inv               0
total_rec_prncp               0
total_rec_int                 0
total_rec_late_fee            0
recoveries                    0
collection_recovery_fee       0
term                          0
last_pymnt_amnt               0
initial_list_status           0
funded_amnt_inv               0
policy_code                   0
application_type              0
acc_now_delinq                0
funded_amnt                   0
delinq_amnt                   0
loan_amnt                     0
out_prncp                     0
total_acc                     0
annual_inc                    0
addr_state                    0
verification_status           0
issue_d                       0
loan_status                   0
pymnt_plan                    0
url                           0
sub_grade                     0
purpose                       0
zip_code                      0
dti                           0
installment                   0
delinq_2yrs                   0
earliest_cr_line              0
inq_last_6mths                0
member_id                     0
grade                         0
open_acc                      0
pub_rec                       0
revol_bal                     0
id                            0
dtype: int64
```

[7]:
```
# Remove columns with more than 60% null values

loan = loan.loc[:, ~((loan.isnull().sum()/loan.shape[0])*100 > 60.0)]
print(loan.shape)
```

```
print(loan.isnull().sum().sort_values(ascending=False))
```

```
(39717, 54)
desc                          12942
emp_title                      2459
emp_length                     1075
pub_rec_bankruptcies            697
last_pymnt_d                     71
collections_12_mths_ex_med       56
chargeoff_within_12_mths         56
revol_util                       50
tax_liens                        39
title                            11
last_credit_pull_d                2
total_rec_prncp                   0
out_prncp                         0
initial_list_status               0
out_prncp_inv                     0
total_acc                         0
total_pymnt                       0
total_pymnt_inv                   0
collection_recovery_fee           0
total_rec_int                     0
total_rec_late_fee                0
recoveries                        0
pub_rec                           0
last_pymnt_amnt                   0
policy_code                       0
application_type                  0
acc_now_delinq                    0
delinq_amnt                       0
revol_bal                         0
id                                0
open_acc                          0
member_id                         0
loan_amnt                         0
funded_amnt                       0
funded_amnt_inv                   0
term                              0
int_rate                          0
installment                       0
grade                             0
sub_grade                         0
home_ownership                    0
annual_inc                        0
verification_status               0
issue_d                           0
```

```
loan_status                    0
pymnt_plan                     0
url                            0
purpose                        0
zip_code                       0
addr_state                     0
dti                            0
delinq_2yrs                    0
earliest_cr_line               0
inq_last_6mths                 0
dtype: int64
```

[8]:
```
# Shape after cleaning null values
loan.shape
```

[8]:
```
(39717, 54)
```

[9]:
```
## Remove non unique columns that only have 1 unique value
loan = loan.loc[:, ~(loan.nunique() == 1)]

loan.shape
```

[9]:
```
(39717, 45)
```

[10]:
```
loan.nunique().sort_values(ascending=True)
```

[10]:
```
term                        2
pub_rec_bankruptcies        3
loan_status                 3
verification_status         3
pub_rec                     5
home_ownership              5
grade                       7
inq_last_6mths              9
delinq_2yrs                11
emp_length                 11
purpose                    14
sub_grade                  35
open_acc                   40
addr_state                 50
issue_d                    55
total_acc                  82
last_pymnt_d              101
last_credit_pull_d        106
int_rate                  371
earliest_cr_line          526
zip_code                  823
```

```
loan_amnt                     885
funded_amnt                  1041
revol_util                   1089
out_prncp                    1137
out_prncp_inv                1138
total_rec_late_fee           1356
collection_recovery_fee      2616
dti                          2868
recoveries                   4040
annual_inc                   5318
total_rec_prncp              7976
funded_amnt_inv              8205
installment                 15383
title                       19615
revol_bal                   21711
desc                        26526
emp_title                   28820
last_pymnt_amnt             34930
total_rec_int               35148
total_pymnt_inv             37518
total_pymnt                 37850
member_id                   39717
url                         39717
id                          39717
dtype: int64
```

[11]:
```
## Remove columns
# id: redundant column to member_id
# url, desc, earliest_cr_line, revol_bal, title, emp_title,␣
 ↪collection_recovery_fee: not required for credit loss analysis
# zip_code: masked therefore cannot be of help in analysis

loan = loan.drop(['id', 'url', 'desc', 'earliest_cr_line', 'revol_bal',␣
 ↪'title', 'emp_title', 'zip_code', 'collection_recovery_fee', 'last_pymnt_d',␣
 ↪'last_credit_pull_d'], axis=1)
loan.shape
```

[11]:
```
(39717, 34)
```

[12]:
```
## Removing extra strings and convert to appropriate datatypes

loan['term'] = loan['term'].apply(lambda x: int(x.replace(' months', '')))
loan['int_rate'] = loan['int_rate'].apply(lambda x: float(x.replace('%', '')))
loan['revol_util'] = loan['revol_util'].apply(lambda x: float(str(x).
 ↪replace('%', '')))

# Assuming '< 1 year' as 0.7 and converting to float values
```

```
loan['emp_length'] = loan['emp_length'].str.replace('< 1', '0.7').
 ↪replace('[$a-zA-Z+]', '', regex=True).astype('float')

# Change column type to category
for col in ['grade', 'sub_grade', 'home_ownership', 'verification_status',␣
 ↪'loan_status', 'purpose', 'addr_state']:
    loan[col] = loan[col].astype('category')

# Change column type to datetime
loan['issue_d'] = pd.to_datetime(loan['issue_d'],format='%b-%y')
loan['issue_d_month'] = pd.to_datetime(loan['issue_d'],format='%b-%y').dt.month
loan['issue_d_year'] = pd.to_datetime(loan['issue_d'],format='%b-%y').dt.year

loan.dtypes
```

[12]:
```
member_id                      int64
loan_amnt                      int64
funded_amnt                    int64
funded_amnt_inv              float64
term                           int64
int_rate                     float64
installment                  float64
grade                       category
sub_grade                   category
emp_length                   float64
home_ownership              category
annual_inc                   float64
verification_status         category
issue_d                datetime64[ns]
loan_status                 category
purpose                     category
addr_state                  category
dti                          float64
delinq_2yrs                    int64
inq_last_6mths                 int64
open_acc                       int64
pub_rec                        int64
revol_util                   float64
total_acc                      int64
out_prncp                    float64
out_prncp_inv                float64
total_pymnt                  float64
total_pymnt_inv              float64
total_rec_prncp              float64
total_rec_int                float64
total_rec_late_fee           float64
recoveries                   float64
```

```
last_pymnt_amnt               float64
pub_rec_bankruptcies          float64
issue_d_month                   int32
issue_d_year                    int32
dtype: object
```

[13]:
```
# Round all float types to 2 decimals

for cols in loan.columns:
    if(loan[cols].dtype == 'float64'):
        loan[cols] = loan[cols].round(2)
```

[14]:
```
loan.head(10)
```

[14]:

	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	\
0	1296599	5000	5000	4975.0	36	10.65	
1	1314167	2500	2500	2500.0	60	15.27	
2	1313524	2400	2400	2400.0	36	15.96	
3	1277178	10000	10000	10000.0	36	13.49	
4	1311748	3000	3000	3000.0	60	12.69	
5	1311441	5000	5000	5000.0	36	7.90	
6	1304742	7000	7000	7000.0	60	15.96	
7	1288686	3000	3000	3000.0	36	18.64	
8	1306957	5600	5600	5600.0	60	21.28	
9	1306721	5375	5375	5350.0	60	12.69	

	installment	grade	sub_grade	emp_length	home_ownership	annual_inc	\
0	162.87	B	B2	10.0	RENT	24000.0	
1	59.83	C	C4	0.7	RENT	30000.0	
2	84.33	C	C5	10.0	RENT	12252.0	
3	339.31	C	C1	10.0	RENT	49200.0	
4	67.79	B	B5	1.0	RENT	80000.0	
5	156.46	A	A4	3.0	RENT	36000.0	
6	170.08	C	C5	8.0	RENT	47004.0	
7	109.43	E	E1	9.0	RENT	48000.0	
8	152.39	F	F2	4.0	OWN	40000.0	
9	121.45	B	B5	0.7	RENT	15000.0	

	verification_status	issue_d	loan_status	purpose	addr_state	\
0	Verified	2011-12-01	Fully Paid	credit_card	AZ	
1	Source Verified	2011-12-01	Charged Off	car	GA	
2	Not Verified	2011-12-01	Fully Paid	small_business	IL	
3	Source Verified	2011-12-01	Fully Paid	other	CA	
4	Source Verified	2011-12-01	Current	other	OR	
5	Source Verified	2011-12-01	Fully Paid	wedding	AZ	
6	Not Verified	2011-12-01	Fully Paid	debt_consolidation	NC	
7	Source Verified	2011-12-01	Fully Paid	car	CA	

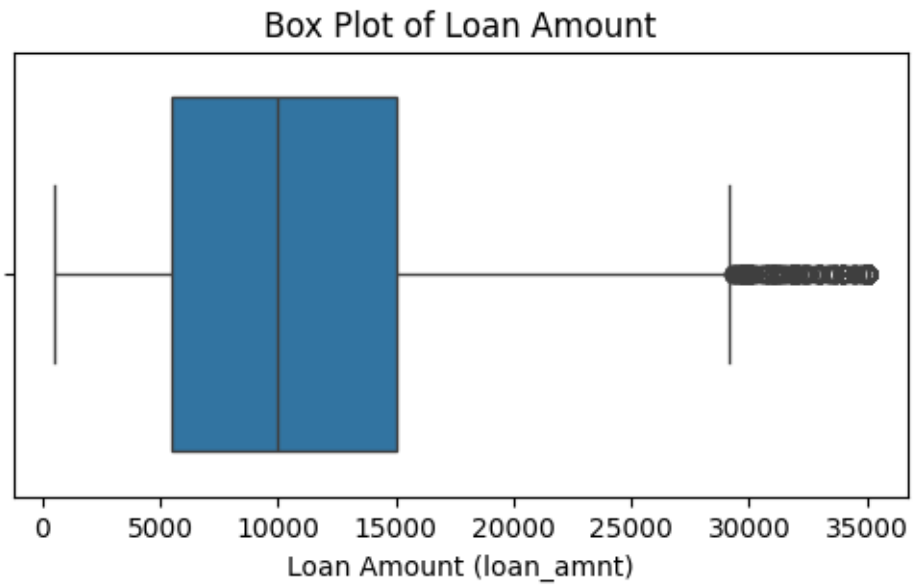
```
8     Source Verified 2011-12-01  Charged Off      small_business         CA
9            Verified 2011-12-01  Charged Off               other         TX

     dti  delinq_2yrs  inq_last_6mths  open_acc  pub_rec  revol_util  \
0  27.65            0               1         3        0        83.7
1   1.00            0               5         3        0         9.4
2   8.72            0               2         2        0        98.5
3  20.00            0               1        10        0        21.0
4  17.94            0               0        15        0        53.9
5  11.20            0               3         9        0        28.3
6  23.51            0               1         7        0        85.6
7   5.35            0               2         4        0        87.5
8   5.55            0               2        11        0        32.6
9  18.08            0               0         2        0        36.5

   total_acc  out_prncp  out_prncp_inv  total_pymnt  total_pymnt_inv  \
0          9       0.00           0.00      5863.16          5833.84
1          4       0.00           0.00      1008.71          1008.71
2         10       0.00           0.00      3005.67          3005.67
3         37       0.00           0.00     12231.89         12231.89
4         38     524.06         524.06      3513.33          3513.33
5         12       0.00           0.00      5632.21          5632.21
6         11       0.00           0.00     10110.84         10110.84
7          4       0.00           0.00      3939.14          3939.14
8         13       0.00           0.00       646.02           646.02
9          3       0.00           0.00      1476.19          1469.34

   total_rec_prncp  total_rec_int  total_rec_late_fee  recoveries  \
0          5000.00         863.16                0.00        0.00
1           456.46         435.17                0.00      117.08
2          2400.00         605.67                0.00        0.00
3         10000.00        2214.92               16.97        0.00
4          2475.94        1037.39                0.00        0.00
5          5000.00         632.21                0.00        0.00
6          6985.61        3125.23                0.00        0.00
7          3000.00         939.14                0.00        0.00
8           162.02         294.94                0.00      189.06
9           673.48         533.42                0.00      269.29

   last_pymnt_amnt  pub_rec_bankruptcies  issue_d_month  issue_d_year
0           171.62                   0.0             12          2011
1           119.66                   0.0             12          2011
2           649.91                   0.0             12          2011
3           357.48                   0.0             12          2011
4            67.79                   0.0             12          2011
5           161.03                   0.0             12          2011
6          1313.76                   0.0             12          2011
```

7	111.34	0.0	12	2011
8	152.39	0.0	12	2011
9	121.45	0.0	12	2011

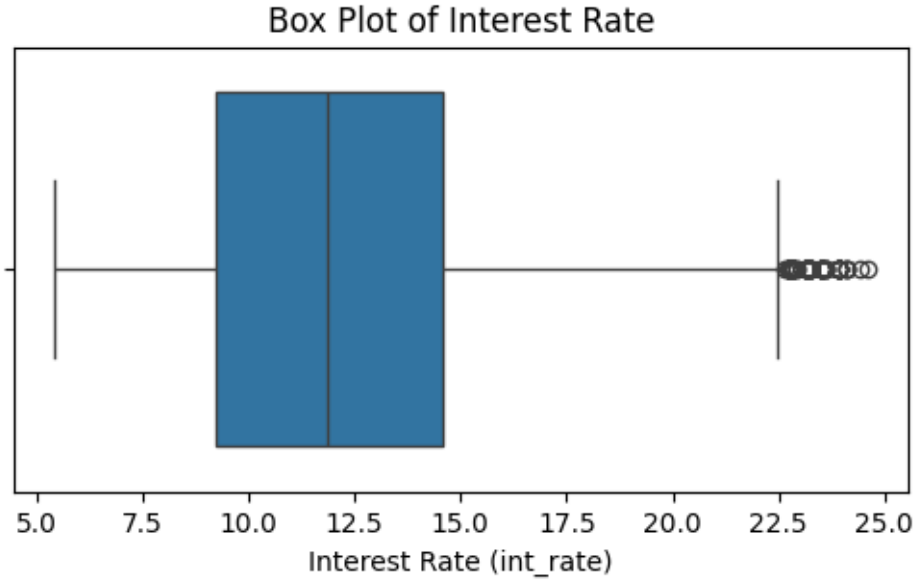
0.1.4 Finding and cleaning Outliers

[15]:
```
numerical_cols = ['loan_amnt', 'int_rate', 'annual_inc', 'dti']
numerical_col_desc = {
    'loan_amnt': 'Loan Amount',
    'int_rate': 'Interest Rate',
    'annual_inc': 'Annual Income',
    'dti': 'Debt to Income Ratio'
}
for col in numerical_cols:
    quantile = loan[col].quantile([0.5, 0.75,0.90, 0.95, 0.97,0.98, 0.99])
    print('\n' + numerical_col_desc[col] + ' Quantiles')
    print(quantile)
    plt.figure(figsize=(6, 3))
    sns.boxplot(x=loan[col]).set(
        xlabel=numerical_col_desc[col] + ' (' +col+ ')'
    )
    plt.title(f'Box Plot of {numerical_col_desc[col]}')
    plt.show()
```

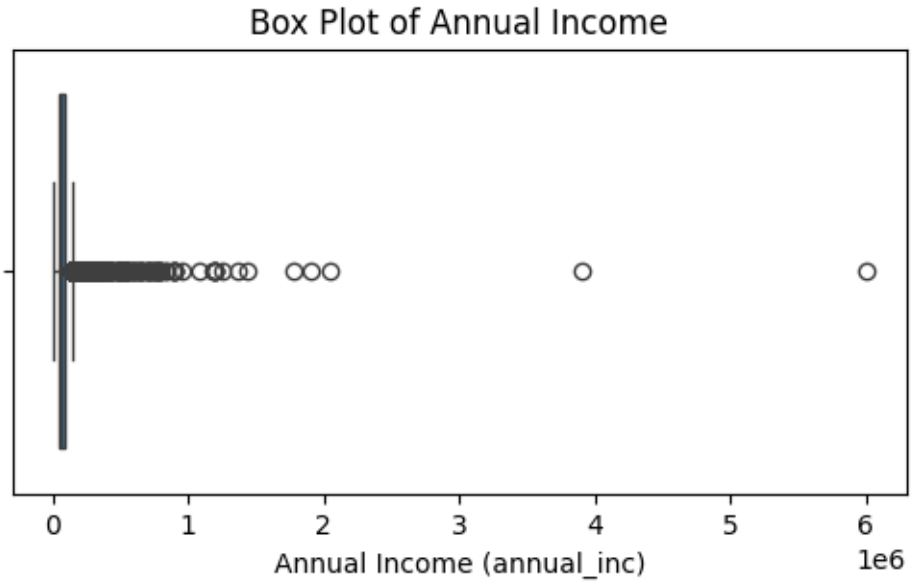
```
Loan Amount Quantiles
0.50    10000.0
0.75    15000.0
0.90    22000.0
0.95    25000.0
0.97    30000.0
0.98    31468.0
0.99    35000.0
Name: loan_amnt, dtype: float64
```



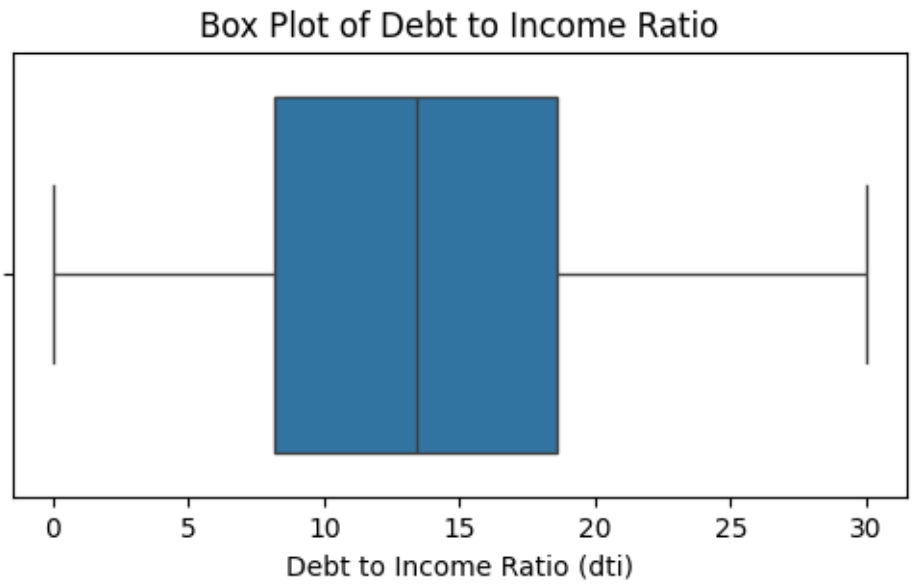

```
Interest Rate Quantiles
0.50    11.86
0.75    14.59
0.90    16.89
0.95    18.54
0.97    19.42
0.98    20.25
0.99    20.99
Name: int_rate, dtype: float64
```



```
Annual Income Quantiles
0.50     59000.00
0.75     82300.00
0.90    116000.00
0.95    142000.00
0.97    165757.92
0.98    187000.00
0.99    234999.36
Name: annual_inc, dtype: float64
```




```
Debt to Income Ratio Quantiles
0.50    13.40
0.75    18.60
0.90    22.33
0.95    23.84
0.97    24.54
0.98    24.88
0.99    26.68
Name: dti, dtype: float64
```



[16]:
```
## Removing Outliers
# Annual Income starts increasing exponentially from 95 Percentile onwards, so␣
 ↪they need to be removed
per_95_annual_inc = loan['annual_inc'].quantile(0.95)
loan = loan[loan['annual_inc']<=per_95_annual_inc]

loan.shape
```

[16]: (37743, 36)

0.1.5 Inference:

Based on boxplots Annual Income had the most outliers above 95 percentile and have been removed

[17]:
```
## Removing records with Current loan status, since they can/ cannot change to␣
 ↪Fully Paid or Charged off
loan = loan[~(loan['loan_status']=='Current')]

loan.shape
```

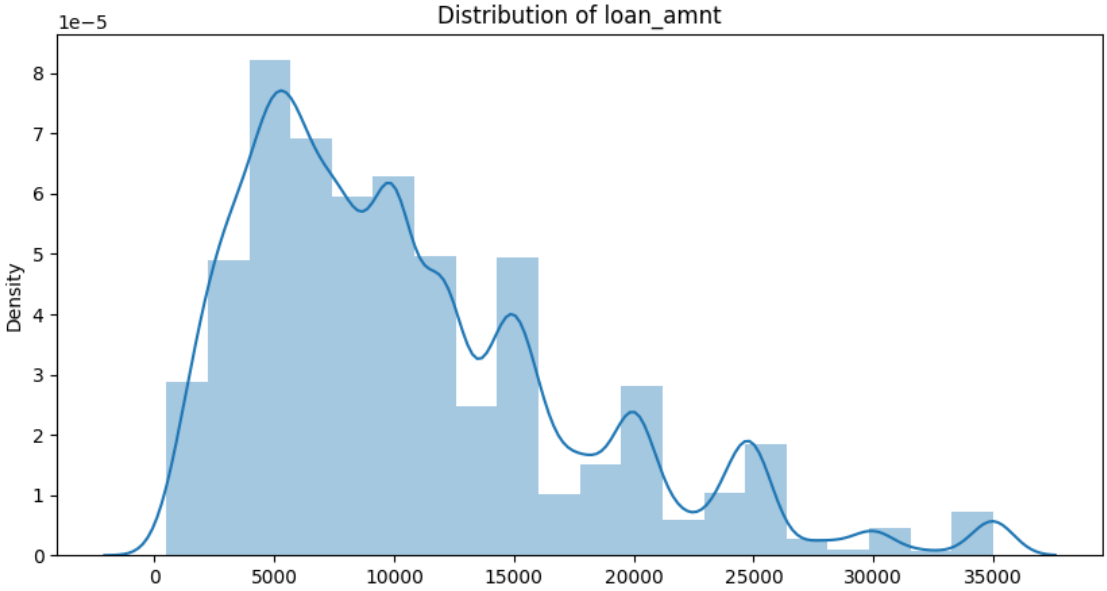
[17]: (36689, 36)

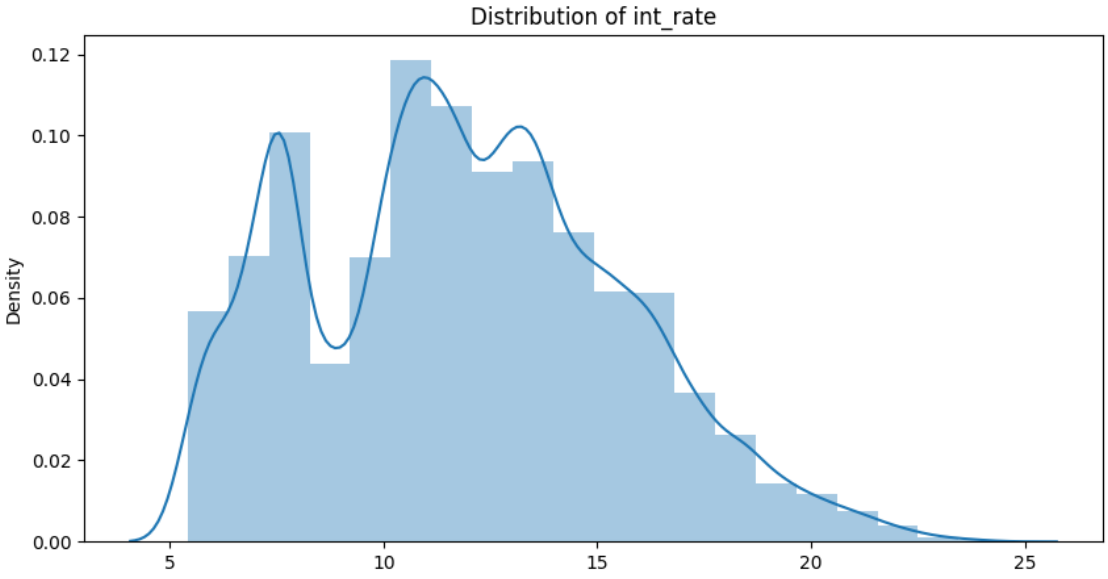
0.1.6 Inference:

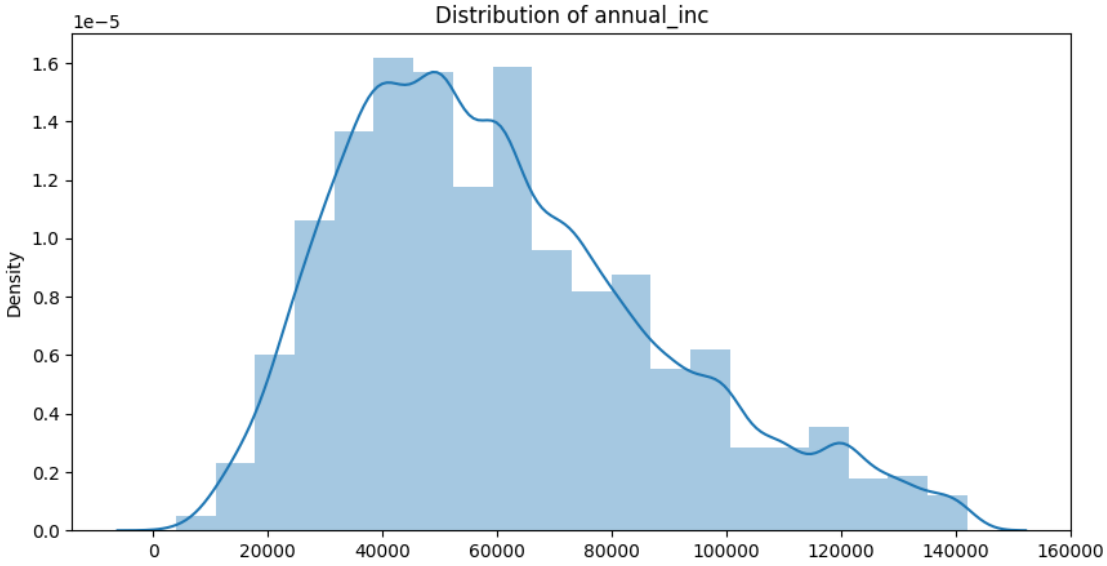
Records with 'Current' loan status cannot be considered in evaluation since they can either pay out fully or default loan

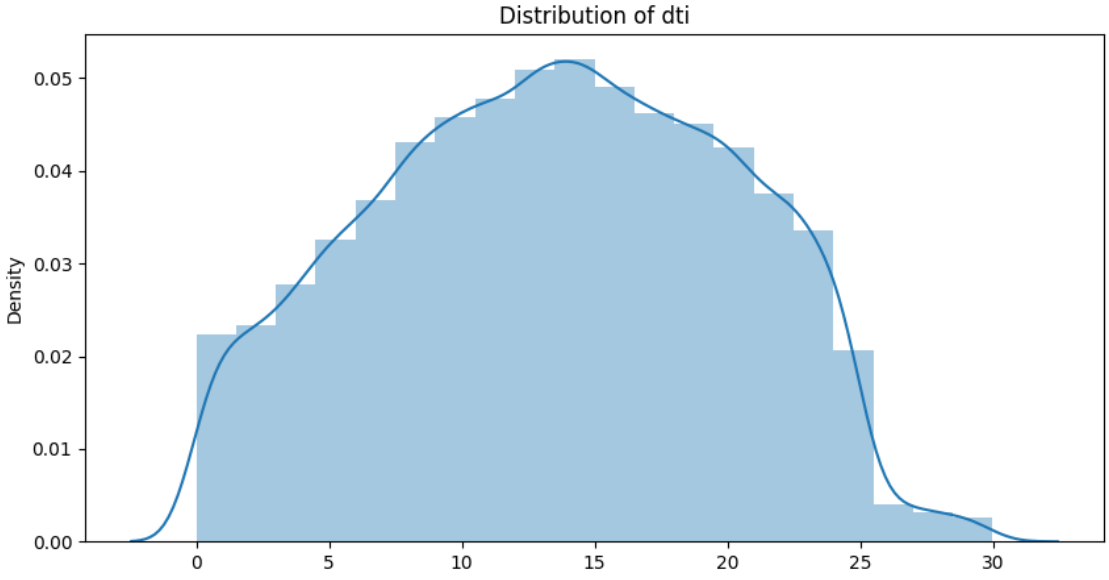
0.1.7 Univariate Analysis

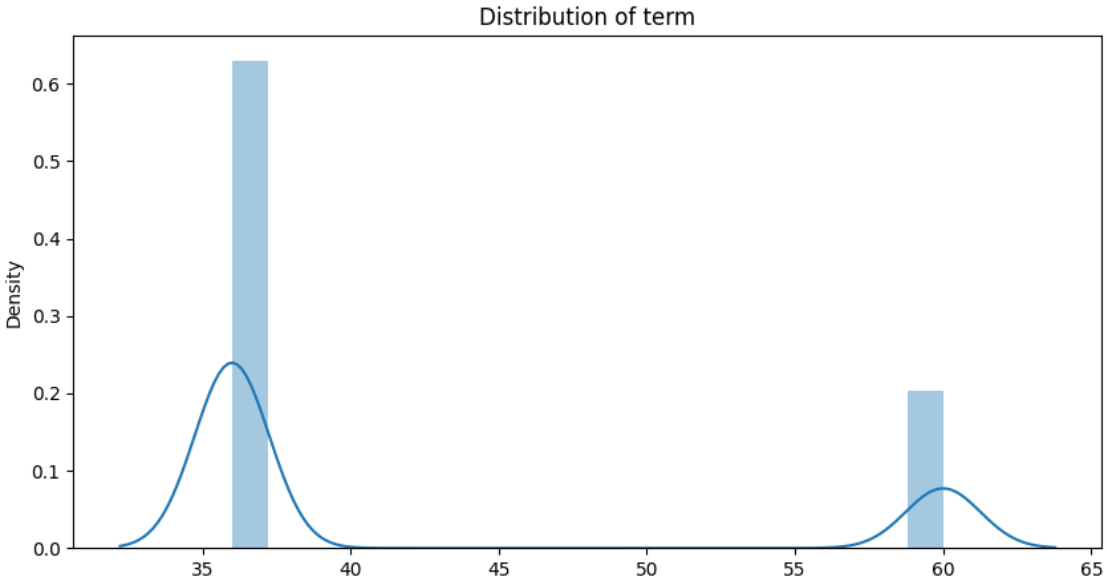
[18]:
```
numerical_cols = ['loan_amnt', 'int_rate', 'annual_inc', 'dti', 'term']

for col in numerical_cols:
    plt.figure(figsize=(10, 5))
    sns.distplot(sorted(loan[col]),kde=True,bins=20)
    plt.title(f'Distribution of {col}')
    plt.show()
```






Distribution of annual_inc
1e−5
Density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
0
20000
40000
60000
80000
100000
120000
140000
160000


Distribution of dti
Density
0.00
0.01
0.02
0.03
0.04
0.05
0
5
10
15
20
25
30



0.1.8 Inference:

Loan Amount: Most loans are small to moderate amounts with a few large loans.

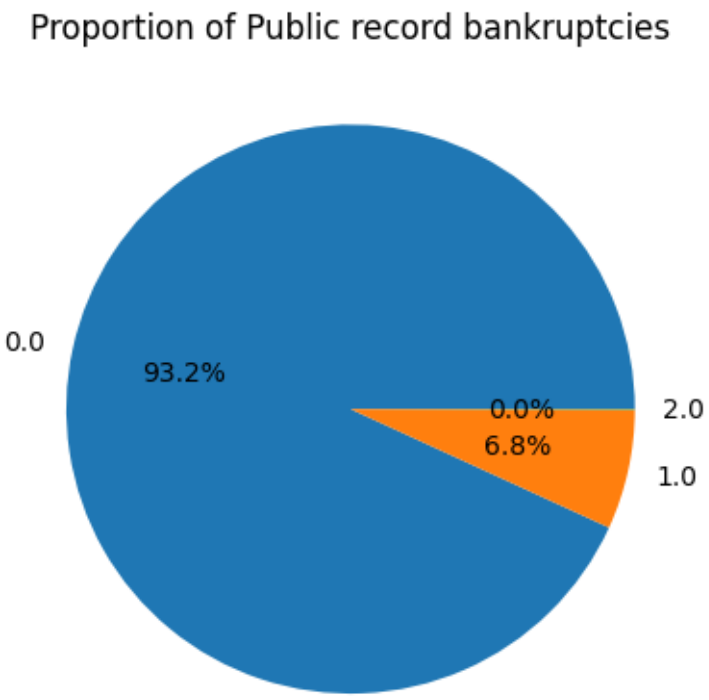
Interest Rate: Interest rates vary, but there might be common rates around certain values.

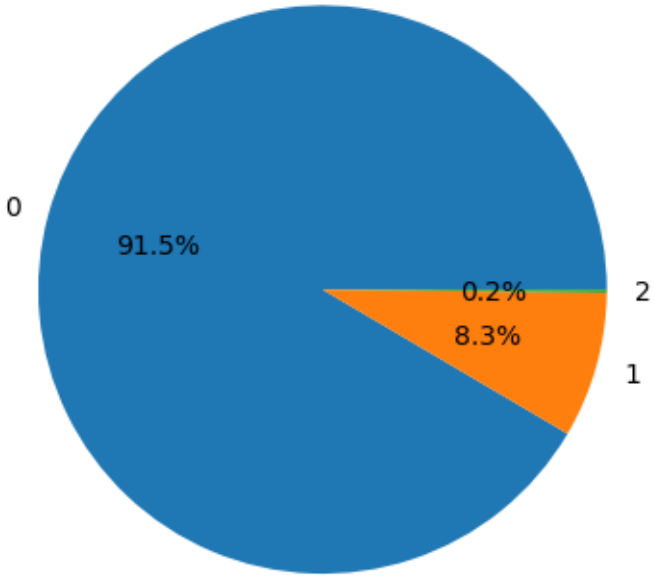
Annual Income: Most borrowers have moderate incomes; a few have very high incomes.

Debt-to-Income Ratio: Most borrowers have a manageable Debt To Income ratio, but some might have higher values indicating more debt.

Term: A significant proportion of loans are of term 36months

[19]:
```
# Proportion of charged off loans with Public bankruptcy records or derogatory␣
↪public records
loan_charged_off = loan[loan['loan_status'] == 'Charged Off']
loan_charged_off['pub_rec_bankruptcies'].value_counts().plot.pie(autopct='%1.
↪1f%%')
plt.title('Proportion of Public record bankruptcies')
plt.ylabel('')
plt.show()

loan_charged_off['pub_rec'].value_counts().plot.pie(autopct='%1.1f%%')
plt.title('Proportion of Derogatory public records')
plt.ylabel('')
plt.show()
```


Proportion of Public record bankruptcies
0.0
93.2%
0.0%
2.0
6.8%
1.0


Proportion of Derogatory public records
0
91.5%
0.2%
2
8.3%
1

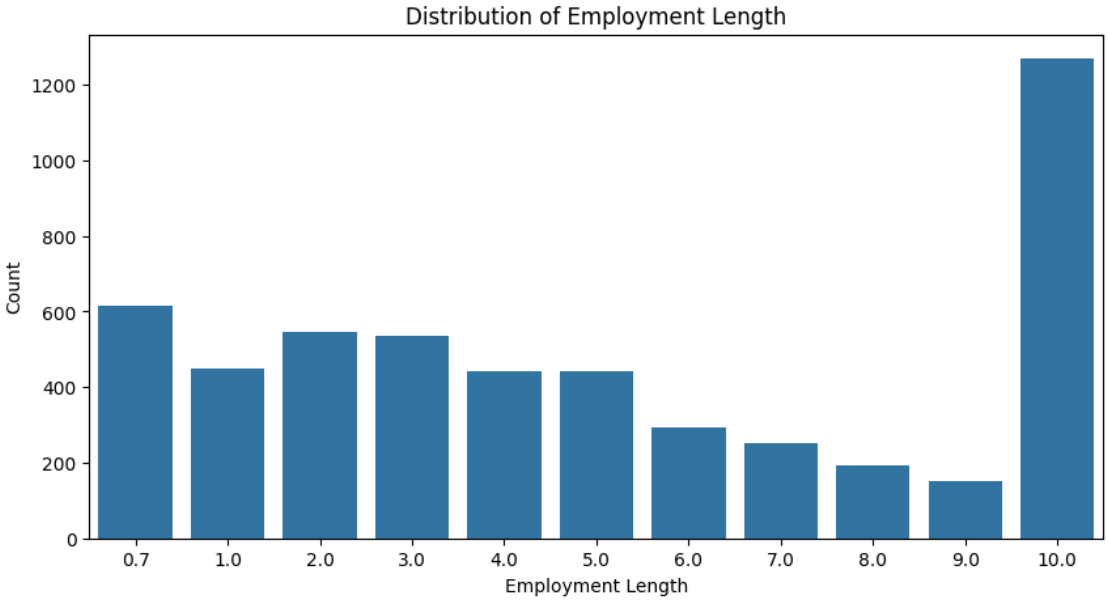
0.1.9 Inference:

Public record bankruptcies: A very small percentage of loan defaulters have filed bankruptcies.

Derogatory public records: A very small percentage of loan defaulters have derogatory public records.

Although these can be high risk customers to lend

[20]:
```
# Distribution of employment Length on Charged Off loans
plt.figure(figsize=(10,5))
sns.countplot(x='emp_length', data=loan[loan['loan_status'] == 'Charged Off'])
plt.xlabel('Employment Length')
plt.ylabel('Count')
plt.title('Distribution of Employment Length')
plt.show()
```

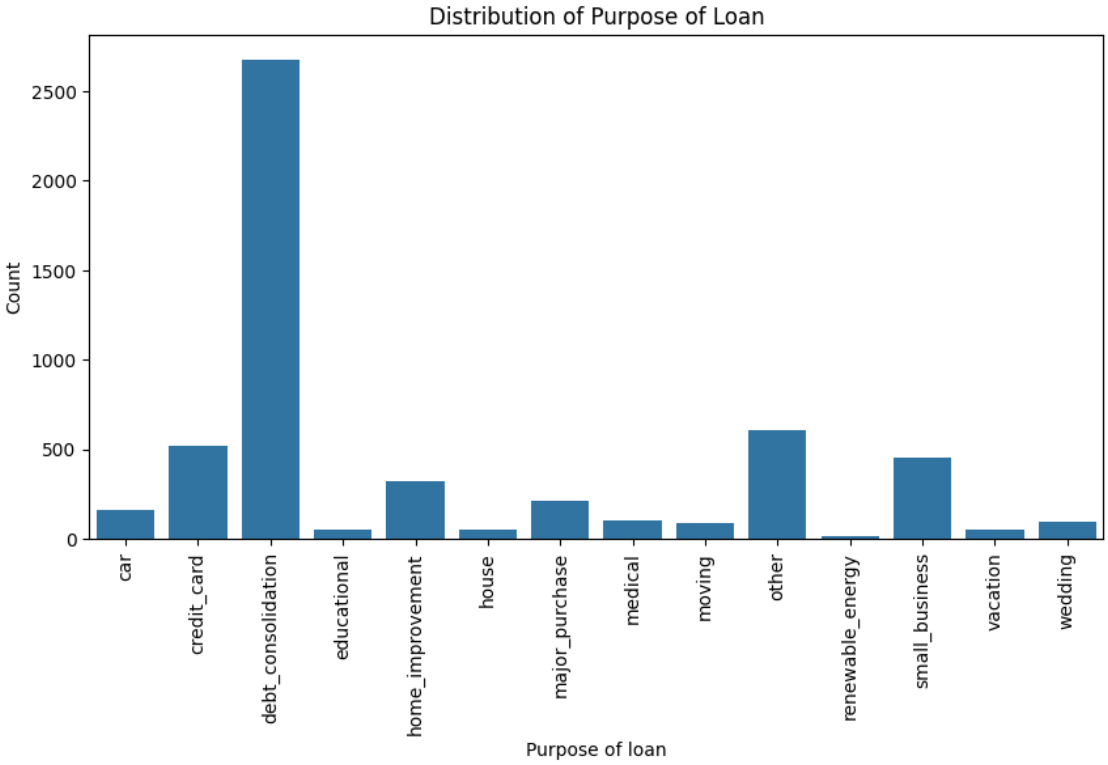


0.1.10 Inference:

Majority of the Charged off loans belong to customers with employment length around 10 and above years

[21]:
```
# Distribution of employment Length on Charged Off loans
plt.figure(figsize=(10,5))
sns.countplot(x='purpose', data=loan[loan['loan_status'] == 'Charged Off'])
plt.xlabel('Purpose of loan')
```

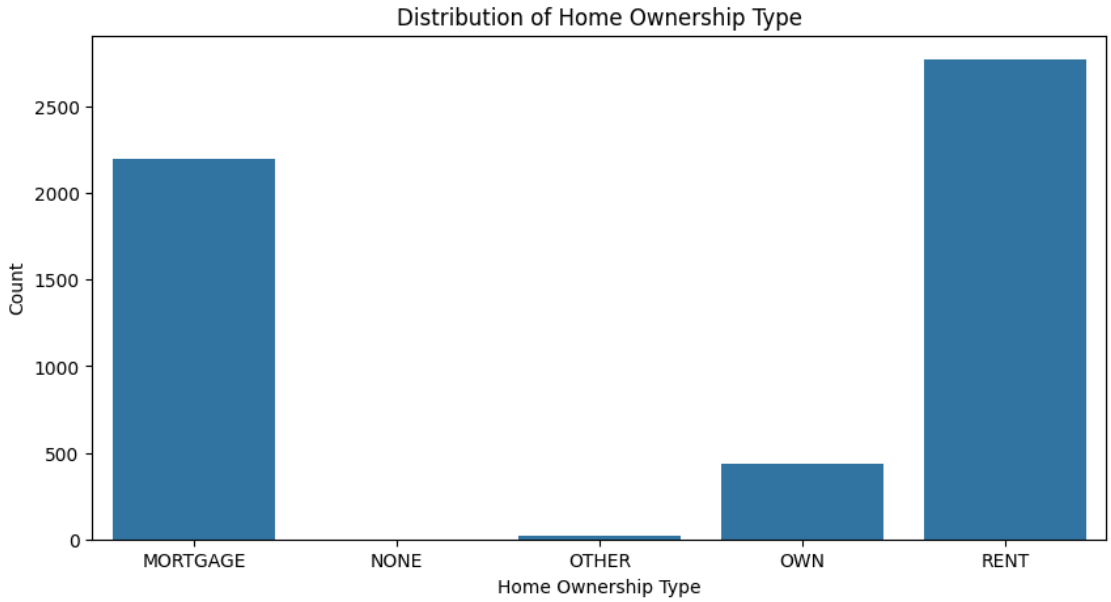
```
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.title('Distribution of Purpose of Loan')
plt.show()
```



0.1.11 Inference:

Majority of the Charged off loans belong to customers who took loan for purpose of another debt consolidation

[22]:
```
# Distribution of employment Length on Charged Off loans
plt.figure(figsize=(10,5))
sns.countplot(x='home_ownership', data=loan[loan['loan_status'] == 'Charged␣
↪Off'])
plt.xlabel('Home Ownership Type')
plt.ylabel('Count')
plt.title('Distribution of Home Ownership Type')
plt.show()
```



0.1.12 Inference:

Majority of the Charged off loans belong to customers who live in a rented place and second to it who already have mortgage on their homes

---

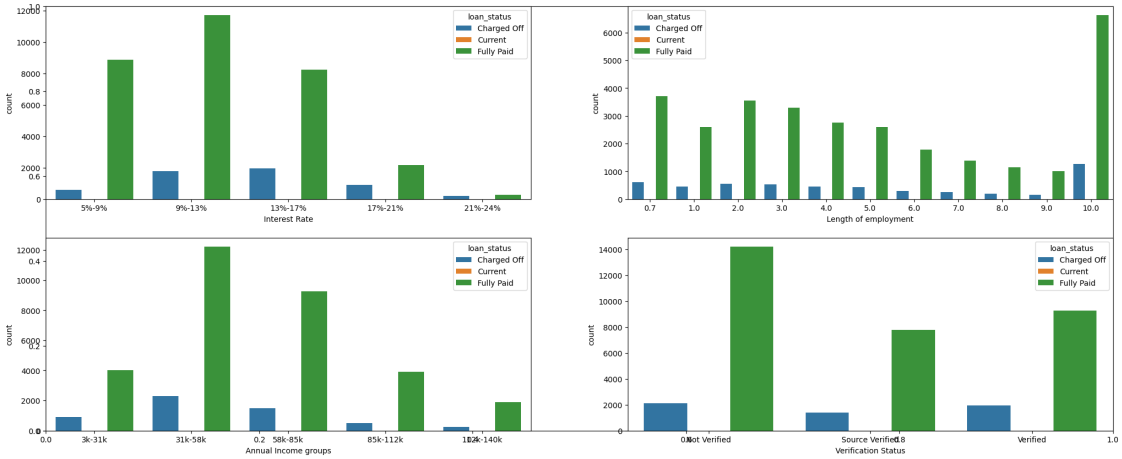
0.1.13 Bivariate Analysis

[23]:
```
# Create Bins for int_rate,open_acc,revol_util,total_acc
loan['open_acc_groups'] = pd.cut(loan['open_acc'],bins = 5,precision␣
 ↪=0,labels=['2-10','10-19','19-27','27-36','36-44'])
loan['total_acc_groups'] = pd.cut(loan['total_acc'], bins=5,precision␣
 ↪=0,labels=['2-20','20-37','37-55','55-74','74-90'])
loan['annual_inc_groups'] = pd.cut(loan['annual_inc'], bins=5,precision␣
 ↪=0,labels =['3k-31k','31k-58k','58k-85k','85k-112k','112k-140k'])
loan['loan_amnt_groups'] = pd.cut(loan['open_acc'],bins = 7,precision␣
 ↪=0,labels=['0-5000', '5000-10000', '10000-15000', '15000-20000',␣
 ↪'20000-25000', '25000-30000', '30000-36000'])
loan['int_rate_groups'] = pd.cut(loan['int_rate'], bins=5,precision␣
 ↪=0,labels=['5%-9%','9%-13%','13%-17%','17%-21%','21%-24%'])
```

[24]:
```
## Show the count plot
fig, ax = plt.subplots(figsize = (25,10))
plt.subplot(221)
sns.countplot(x='int_rate_groups', hue='loan_status', data=loan)
plt.xlabel('Interest Rate')
```

```
plt.subplot(222)
sns.countplot(x='emp_length', hue='loan_status', data=loan)
plt.xlabel('Length of employment')

plt.subplot(223)
sns.countplot(x='annual_inc_groups', hue='loan_status', data=loan)
plt.xlabel('Annual Income groups')

plt.subplot(224)
sns.countplot(x = 'verification_status', data = loan, hue='loan_status')
plt.xlabel('Verification Status')
```

[24]:
```
Text(0.5, 0, 'Verification Status')
```



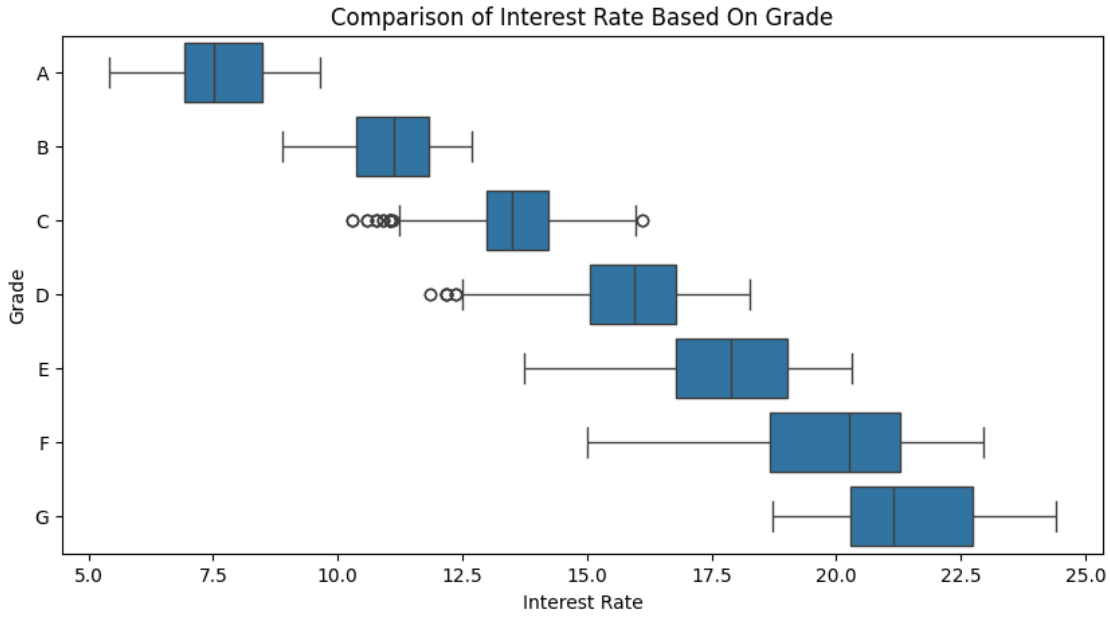
0.1.14 Inference:

Factors showing higher charged off loans

1. Interest Rate: Higher interest rates correlate with a higher risk of loan default.
2. Length of Employment: Stable, long-term employment is associated with better loan repayment, though it's not a strong standalone predictor.
3. Annual Income Groups: Higher annual incomes (above 58k) are associated with a higher likelihood of fully repaying loans. Lower income groups (3k-31k) show a higher risk of default, suggesting that income level is an important factor in assessing loan repayment capability.
4. Verification Status: Verified income and employment significantly reduce the risk of loan default.

[25]:
```
# Comparison of interest rate based on grade
plt.figure(figsize=(10,5))
sns.boxplot(data=loan[loan['loan_status'] == 'Charged␣
↪Off'],x='int_rate',y='grade')
```

```
plt.xlabel('Interest Rate')
plt.ylabel('Grade')
plt.title('Comparison of Interest Rate Based On Grade',fontsize=12)
plt.show()
```

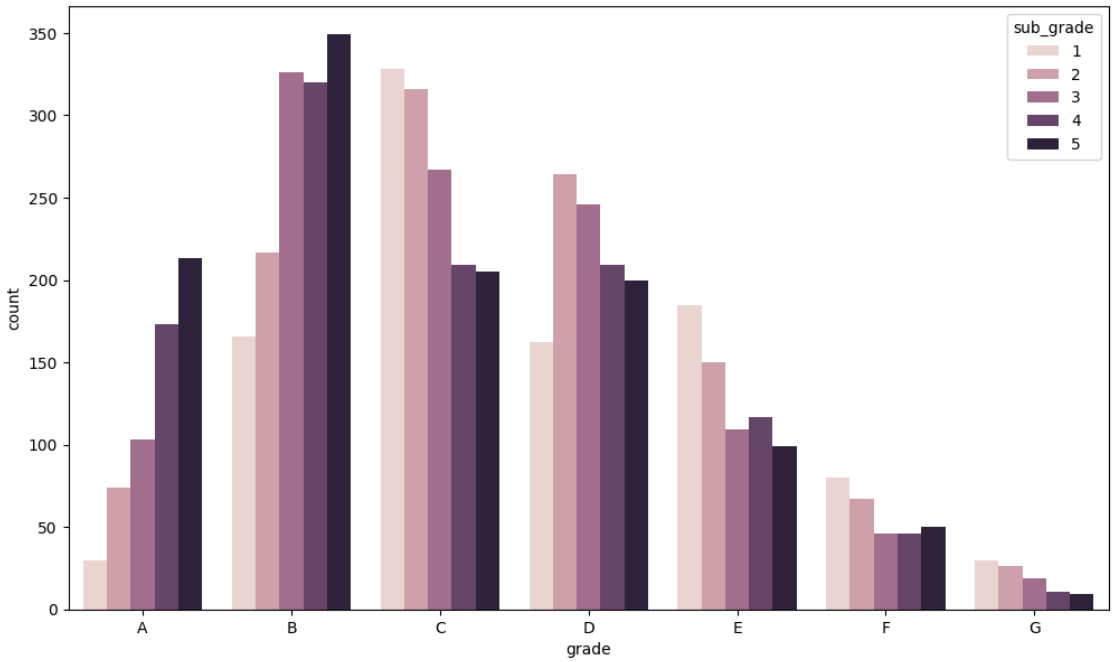


[26]:
```
# Convert subgrade to numeric data for plotting countplot
loan.sub_grade = pd.to_numeric(loan.sub_grade.apply(lambda x : x[-1]))
loan.sub_grade.head(5)
```

[26]:
```
0    2
1    4
2    5
3    1
5    4
Name: sub_grade, dtype: int64
```

[27]:
```
# Countplot of subgrades for Charged off

# fig, ax = plt.subplots(figsize=(12,7))
fig, ax = plt.subplots(figsize=(12,7))
sns.set_palette('colorblind')
sns.countplot(x = 'grade', order = ['A', 'B', 'C', 'D', 'E', 'F', 'G'] , hue =␣
 ↪'sub_grade',data = loan[loan.loan_status == 'Charged Off'])
```

[27]:
```
<Axes: xlabel='grade', ylabel='count'>
```



0.1.15 Inference:

Grades

1. Higher Grades (A, B): Likely to have fewer charged-off loans due to lower risk.
2. Moderate Grades (C, D): Higher likelihood of charged-off loans compared to higher grades.
3. Lower Grades (E, F, G): Higher risk categories with more charged-off loans.

[28]:
```
# Scatter plot of Loan amount vs Interest rates
fig = px.scatter(loan, x='loan_amnt', y='int_rate', title='Loan Amount vs␣
 ↪Interest Rate', color="loan_status", marginal_x="histogram",␣
 ↪marginal_y="histogram", width=1100, height=600, trendline="ols",␣
 ↪trendline_scope="overall")
fig.update_layout(
    xaxis_title="Loan Amount",
    yaxis_title="Interest Rate",
    legend_title="Loan Status",
    paper_bgcolor="LightBlue"
)

fig.show()
```

0.1.16 Inference:

1. Higher Interest Rates: Charged Off loans are more concentrated at higher interest rates, suggesting that borrowers with higher interest rates are at a greater risk of default.

2. Loan Amounts: There is no clear differentiation in loan amounts between Fully Paid and Charged Off loans, indicating that loan amount alone may not be a strong predictor of default.
3. Combined Effect: While higher loan amounts are generally associated with higher interest rates, the risk of default appears to be more closely related to the interest rate rather than the loan amount.

[29]:
```
# Scatter plot to derive metrics for Loan amount vs Public Record Bankruptcies
fig = px.scatter(loan[loan['loan_status'] == 'Charged Off'],␣
 ↪x='pub_rec_bankruptcies', y='annual_inc', title='Loan Amount vs Public␣
 ↪Record of Bankruptcies', color="loan_status", marginal_x="histogram",␣
 ↪marginal_y="histogram", width=1100, height=600, trendline="ols",␣
 ↪trendline_scope="overall")
fig.update_layout(
    xaxis_title="Public Record of Bankruptcies",
    yaxis_title="Loan Amount",
    legend_title="Loan Status",
    paper_bgcolor="LightBlue"
)
```

0.1.17 Inference:

Loan amount vs Public Record of Bankruptcies is inconclusive to derive a loan amount range that is relative to bankruptcies

[30]:
```
# Plotly Box Plot interest rate vs loan status
fig = px.box(loan, x='loan_status', y='int_rate', title='Interest Rate vs Loan␣
 ↪Status')
fig.show()
# Plotly Box Plot Annual Income vs Loan Status
fig = px.box(loan, x='loan_status', y='annual_inc', title='Annual Income vs␣
 ↪Loan Status')
fig.show()
# Plotly Box Plot Debt to Income Ratio vs Loan Status
fig = px.box(loan, x='loan_status', y='dti', title='Debt to Income Ratio vs␣
 ↪Loan Status')
fig.show()
```

0.1.18 Inference:

Maximum number of loan defaulters(Charged Off loan_status) were in the following ranges: Interest Rate - 11.26 - 16.32

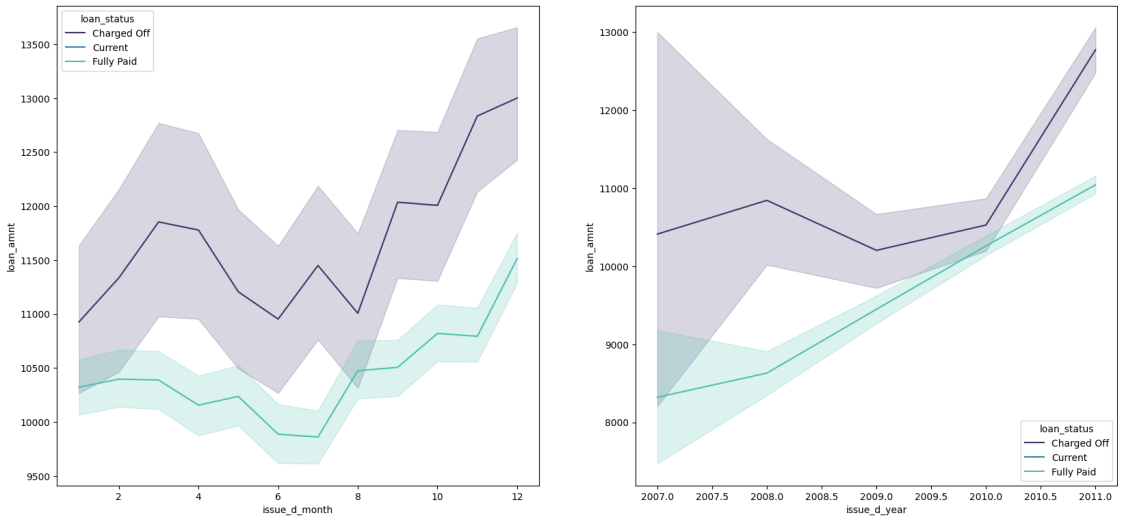
Annual Income - 36.3k - 71.67k

Debt to Income Ratio - 9.18 - 19.40

[31]:
```
# Lineplot for timeline of defaulters
plt.figure(figsize=(20,20))
```

```
plt.subplot(221)
sns.lineplot(data =loan,y='loan_amnt', x='issue_d_month', hue␣
 ↪='loan_status',palette="mako")
plt.subplot(222)
sns.lineplot(data =loan,y='loan_amnt', x='issue_d_year', hue␣
 ↪='loan_status',palette="mako")
```

[31]: <Axes: xlabel='issue_d_year', ylabel='loan_amnt'>

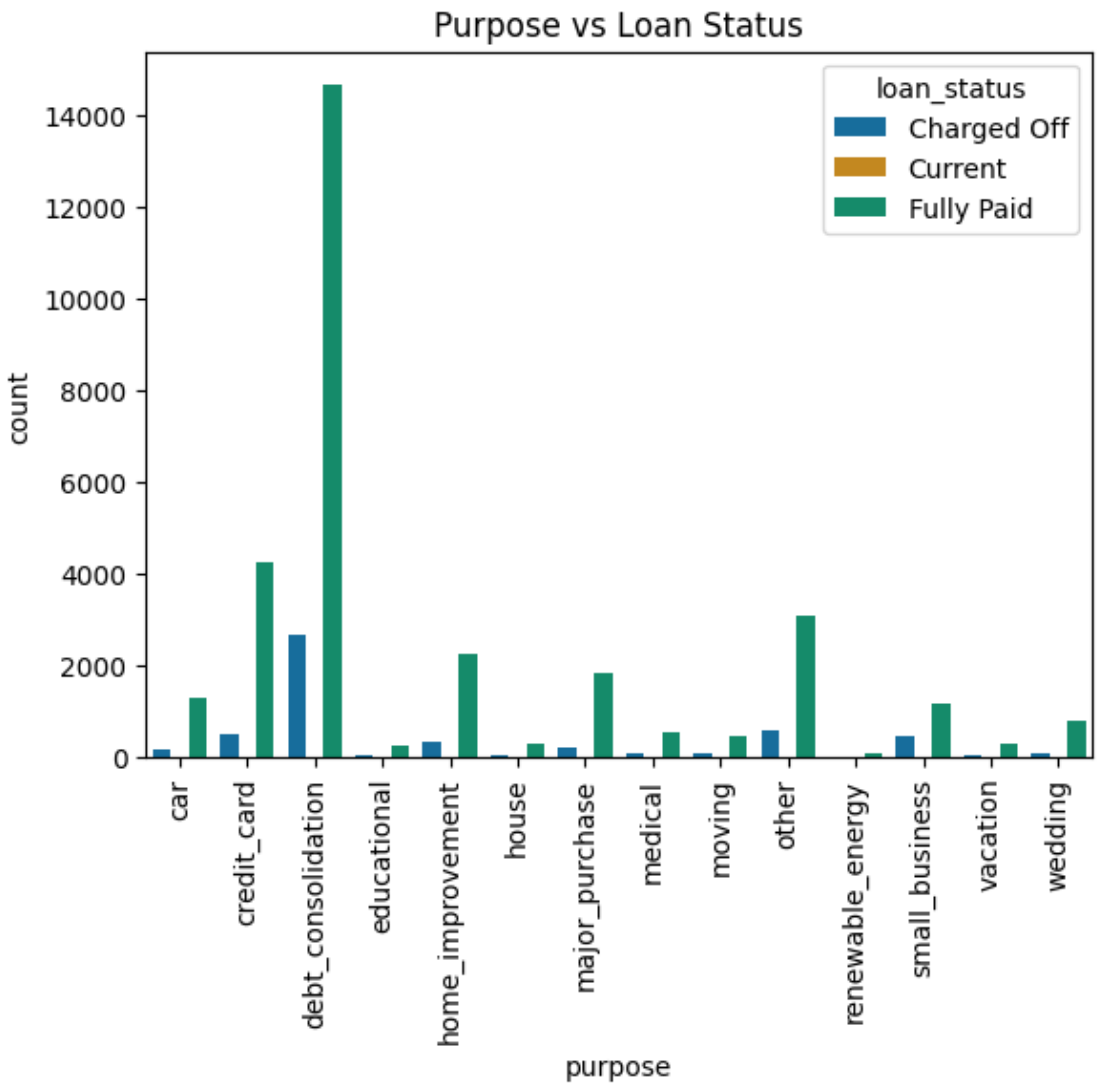


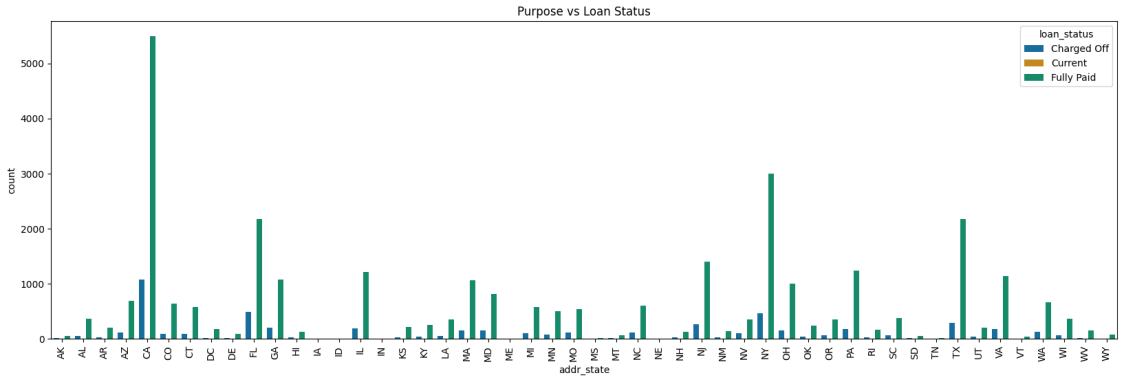
0.1.19 Inference:

Highest number loan defaulters(Charged Off loan_status) are as follows: Requested loan during the month - December

Requested loan during the year - 2011

[32]:
```
# Count plots for purpose and customer state vs loan_status
sns.countplot(x='purpose', hue='loan_status', data=loan)
plt.title('Purpose vs Loan Status')
plt.xticks(rotation=90)
plt.show()

plt.figure(figsize=(20,6))
sns.countplot(x='addr_state', hue='loan_status', data=loan)
plt.title('Purpose vs Loan Status')
plt.xticks(rotation=90)
plt.show()
```


Purpose vs Loan Status
loan_status
Charged Off
Current
Fully Paid
count
14000
12000
10000
8000
6000
4000
2000
0
car
credit_card
debt_consolidation
educational
home_improvement
house
major_purchase
medical
moving
other
renewable_energy
small_business
vacation
wedding
purpose


Purpose vs Loan Status
loan_status
Charged Off
Current
Fully Paid
count
addr_state

0.1.20 Inference:

Highest number loan defaulters(Charged Off loan_status) are as follows: Requested loan for Purpose - Debt Consolidation

Requested loan from State - CA - California

[33]:
```
# Plot histograms for different bins of relevant numeric data
column_bins = ['int_rate_groups', 'open_acc_groups', 'total_acc_groups',␣
 ↪'annual_inc_groups', 'loan_amnt_groups']
column_bins_desc = ['Intereset Rate Groups', 'Open Account Groups', 'Total␣
 ↪Account Groups', 'Annual Income Groups', 'Loan Amount Groups']
for (index, cols) in enumerate(column_bins):
    fig = px.histogram(loan, x=cols, color='loan_status', barmode='group',
                       title=column_bins_desc[index] + ' vs Loan Status')
    fig.update_layout(
        xaxis_title=column_bins_desc[index],
        yaxis_title="",
        legend_title="Loan Status",
        title=dict(x=0.5, y=0.95),
        font=dict(color="#939393", size=14),
        title_font_size=16,
        title_font_color="#939393",
        plot_bgcolor='#E3E2E2'
    )
    fig.show()
```

0.1.21 Inference:

Highest number loan defaulters(Charged Off loan_status) are as follows: 9%-17% Interest rates

2-10 open credit lines

2-37 total credit lines currently in the borrower's credit file

31k - 85k Annual Income

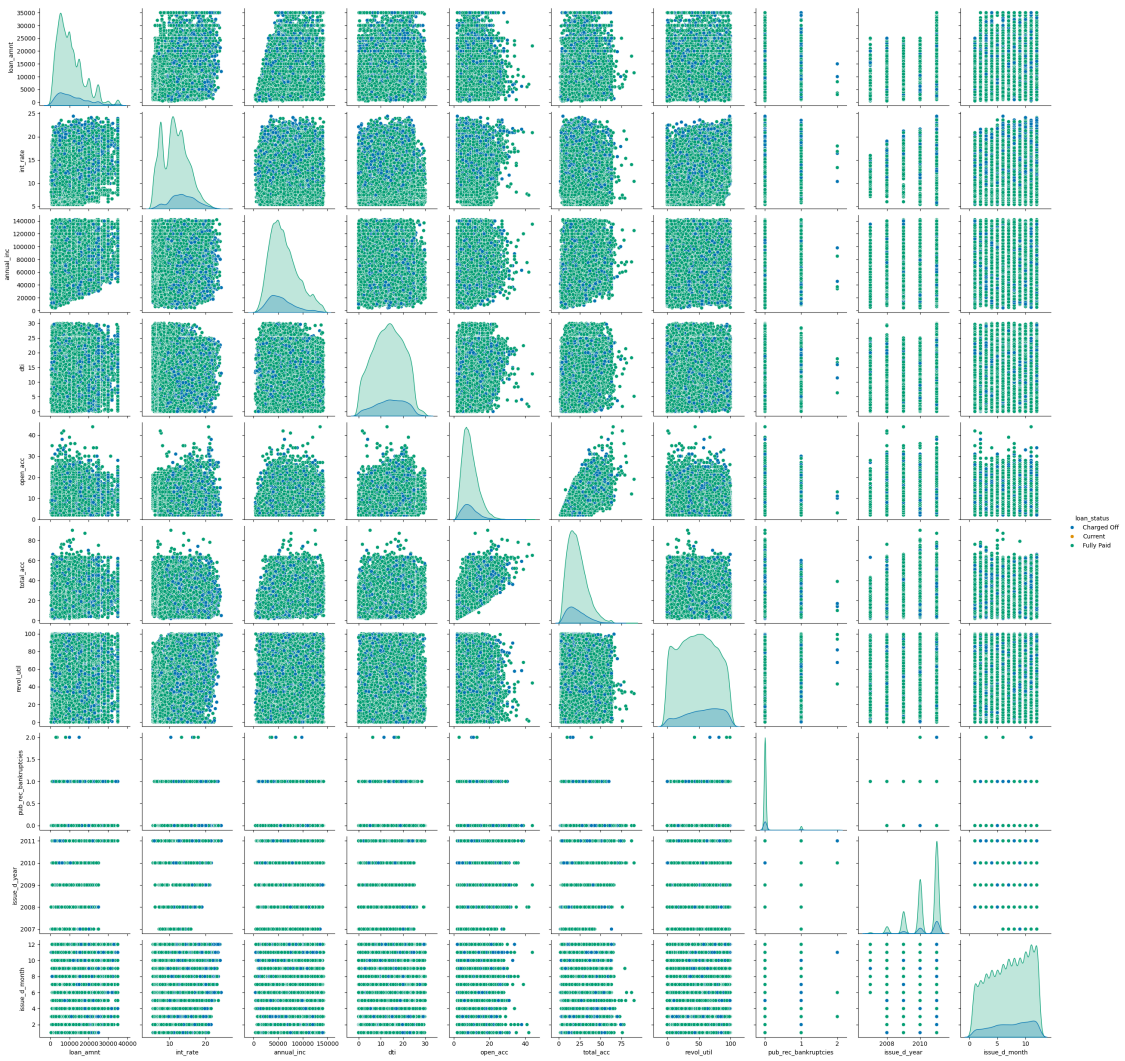
5k - 10k Loan Amount

0.1.22 Multivariate Analysis

Based on relevant numeric columns against loan_status(Charged Off, Fully Paid)

[34]:
```
numeric_cols = ['loan_amnt', 'int_rate', 'annual_inc', 'dti', 'open_acc',␣
 ↪'total_acc', 'revol_util', 'pub_rec_bankruptcies', 'issue_d_year',␣
 ↪'issue_d_month']
# Pair plot for deriving dependant factors for defaulting loan
sns.pairplot(loan[numeric_cols + ['loan_status']], hue='loan_status',␣
 ↪diag_kind='auto')
```

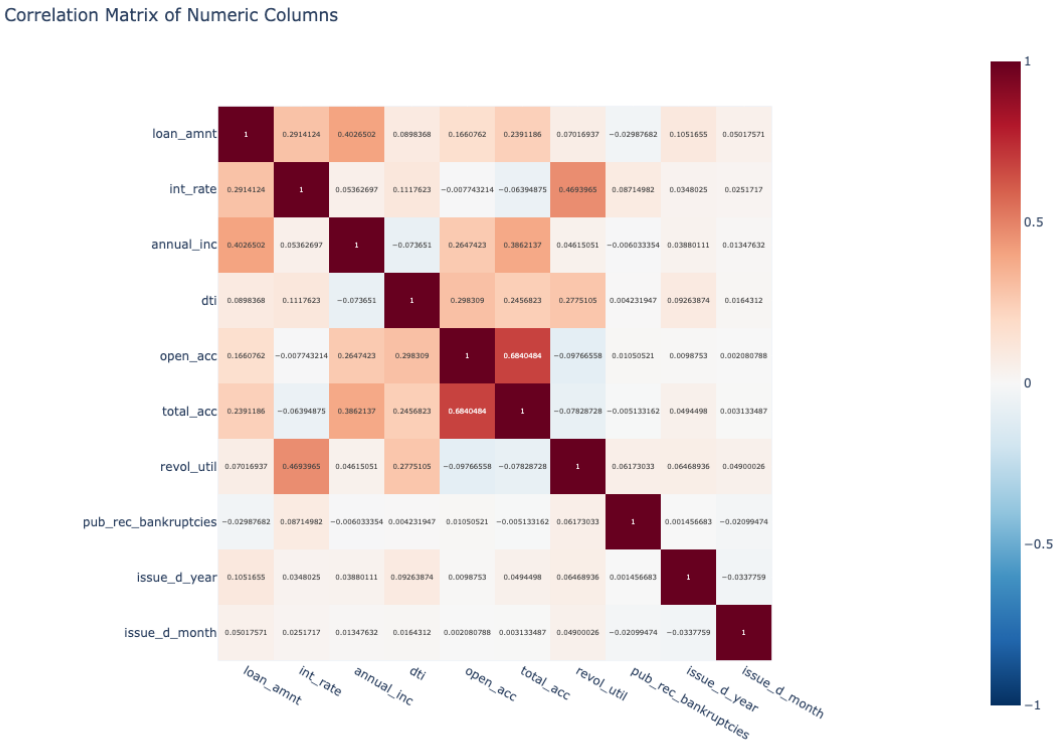
```
# Show plot
plt.show()
```



[35]:
```
# Compute correlation matrix and draw a heatmap
correlation_matrix = loan[numeric_cols].corr()
print(correlation_matrix)
fig = px.imshow(correlation_matrix, text_auto=True,␣
  ↪color_continuous_scale='RdBu_r', zmin=-1, zmax=1, width=800, height=800)
fig.update_layout(title='Correlation Matrix of Numeric Columns')
fig.show()
```

```
                loan_amnt  int_rate  annual_inc       dti  open_acc  \
loan_amnt        1.000000  0.291412    0.402650  0.089837  0.166076
int_rate         0.291412  1.000000    0.053627  0.111762 -0.007743
annual_inc       0.402650  0.053627    1.000000 -0.073651  0.264742
```

```
dti                   0.089837  0.111762   -0.073651  1.000000  0.298309
open_acc              0.166076 -0.007743    0.264742  0.298309  1.000000
total_acc             0.239119 -0.063949    0.386214  0.245682  0.684048
revol_util            0.070169  0.469396    0.046151  0.277511 -0.097666
pub_rec_bankruptcies -0.029877  0.087150   -0.006033  0.004232  0.010505
issue_d_year          0.105165  0.034803    0.038801  0.092639  0.009875
issue_d_month         0.050176  0.025172    0.013476  0.016431  0.002081

                     total_acc  revol_util  pub_rec_bankruptcies  \
loan_amnt             0.239119    0.070169             -0.029877
int_rate             -0.063949    0.469396              0.087150
annual_inc            0.386214    0.046151             -0.006033
dti                   0.245682    0.277511              0.004232
open_acc              0.684048   -0.097666              0.010505
total_acc             1.000000   -0.078287             -0.005133
revol_util           -0.078287    1.000000              0.061730
pub_rec_bankruptcies -0.005133    0.061730              1.000000
issue_d_year          0.049450    0.064689              0.001457
issue_d_month         0.003133    0.049000             -0.020995

                     issue_d_year  issue_d_month
loan_amnt                0.105165       0.050176
int_rate                 0.034803       0.025172
annual_inc               0.038801       0.013476
dti                      0.092639       0.016431
open_acc                 0.009875       0.002081
total_acc                0.049450       0.003133
revol_util               0.064689       0.049000
pub_rec_bankruptcies     0.001457      -0.020995
issue_d_year             1.000000      -0.033776
issue_d_month           -0.033776       1.000000
```



0.1.23 Inference:

Higher interest rates, high DTI ratios, and high revolving utilization are key indicators of higher default risk.

Lower annual income may also contribute to higher default risk.

[36]:
```
# Create a scatter matrix of relevant numeric columns
fig = px.scatter_matrix(loan,
    dimensions=numeric_cols,
    width=1200, height=1000,
    color="loan_status")
fig.show()
```

0.1.24 Inference:

Interest Rate (int_rate): Higher rates are associated with higher default risk.

Annual Income (annual_inc): Lower incomes are linked to higher default rates.

Debt-to-Income Ratio (dti): Higher DTI ratios are indicative of higher default risk.

Revolving Utilization (revol_util): Higher utilization rates correlate with higher default risk.

Public Record Bankruptcies (pub_rec_bankruptcies): History of bankruptcies increases default likelihood.

[ ]: