

In [143]: *# Importing all the libraries and the csv for analysis*

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from scipy.stats import chi2_contingency
d1 = pd.read_csv('loan.csv', low_memory=False)
```

In [5]: *# head for verification*  
d1.head()

Out[5]:

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	insta
0	1077501	1296599	5000	5000	4975.0	36 months	10.65%	
1	1077430	1314167	2500	2500	2500.0	60 months	15.27%	
2	1077175	1313524	2400	2400	2400.0	36 months	15.96%	
3	1076863	1277178	10000	10000	10000.0	36 months	13.49%	
4	1075358	1311748	3000	3000	3000.0	60 months	12.69%	

5 rows × 111 columns

In [31]: print(d1.columns.tolist())

```
['id', 'member_id', 'loan_amnt', 'funded_amnt', 'funded_amnt_inv', 'term',
'int_rate', 'installment', 'grade', 'sub_grade', 'emp_title', 'emp_length',
'home_ownership', 'annual_inc', 'verification_status', 'issue_d', 'loan_stat
us', 'pymnt_plan', 'url', 'desc', 'purpose', 'title', 'zip_code', 'addr_stat
e', 'dti', 'delinq_2yrs', 'earliest_cr_line', 'inq_last_6mths', 'mths_since_
last_delinq', 'mths_since_last_record', 'open_acc', 'pub_rec', 'revol_bal',
'revol_util', 'total_acc', 'initial_list_status', 'out_prncp', 'out_prncp_in
v', 'total_pymnt', 'total_pymnt_inv', 'total_rec_prncp', 'total_rec_int', 't
otal_rec_late_fee', 'recoveries', 'collection_recovery_fee', 'last_pymnt_d',
'last_pymnt_amnt', 'next_pymnt_d', 'last_credit_pull_d', 'collections_12_mth
s_ex_med', 'mths_since_last_major_derog', 'policy_code', 'application_type',
'annual_inc_joint', 'dti_joint', 'verification_status_joint', 'acc_now_delin
q', 'tot_coll_amt', 'tot_cur_bal', 'open_acc_6m', 'open_il_6m', 'open_il_12
m', 'open_il_24m', 'mths_since_rcnt_il', 'total_bal_il', 'il_util', 'open_rv
_12m', 'open_rv_24m', 'max_bal_bc', 'all_util', 'total_rev_hi_lim', 'inq_f
i', 'total_cu_tl', 'inq_last_12m', 'acc_open_past_24mths', 'avg_cur_bal', 'b
c_open_to_buy', 'bc_util', 'chargeoff_within_12_mths', 'delinq_amnt', 'mo_si
n_old_il_acct', 'mo_sin_old_rev_tl_op', 'mo_sin_rcnt_rev_tl_op', 'mo_sin_rcn
t_tl', 'mort_acc', 'mths_since_recent_bc', 'mths_since_recent_bc_dlq', 'mths
_since_recent_inq', 'mths_since_recent_revol_delinq', 'num_accts_ever_120_p
d', 'num_actv_bc_tl', 'num_actv_rev_tl', 'num_bc_sats', 'num_bc_tl', 'num_il
_tl', 'num_op_rev_tl', 'num_rev_accts', 'num_rev_tl_bal_gt_0', 'num_sats',
'num_tl_120dpd_2m', 'num_tl_30dpd', 'num_tl_90g_dpd_24m', 'num_tl_op_past_12
m', 'pct_tl_nvr_dlq', 'percent_bc_gt_75', 'pub_rec_bankruptcies', 'tax_lien
s', 'tot_hi_cred_lim', 'total_bal_ex_mort', 'total_bc_limit', 'total_il_high
_credit_limit']
```

The list of selected columns from the given ones which seems important for this analysis:

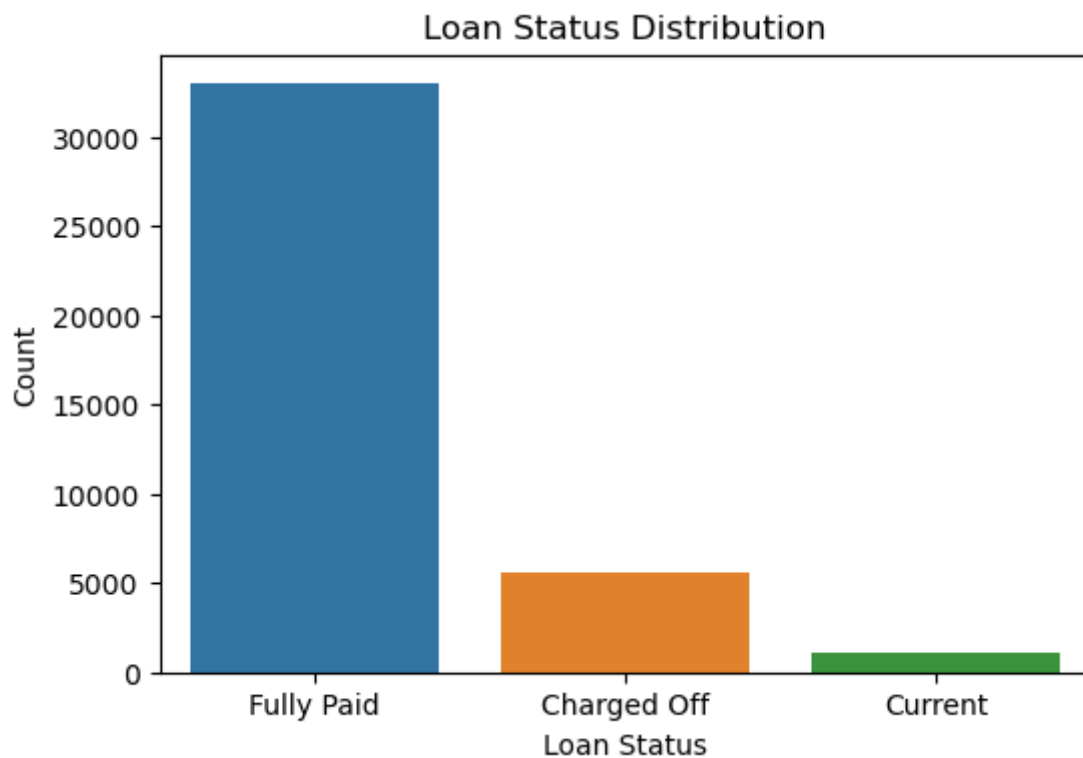
1. **loan\_status**: This is the target variable, indicating whether the loan was fully paid or charged off (defaulted). It's essential for understanding loan outcomes.
2. **annual\_inc**: The borrower's self-reported annual income provides insights into their ability to make loan payments and manage their finances.
3. **dti**: The debt-to-income ratio helps assess the borrower's financial health by comparing their monthly debt payments to their income.
4. **emp\_length**: Employment length is crucial for evaluating the borrower's stability and income source, which impacts their repayment capability.
5. **int\_rate**: The interest rate on the loan affects the borrower's monthly payment and overall affordability.
6. **purpose**: The purpose of the loan can provide insights into the borrower's financial goals and intentions.
7. **term**: The loan term affects the monthly payment and can influence the borrower's ability to repay.
8. **open\_acc** and **total\_acc**: These two variables give an overview of the borrower's credit history and current credit lines.
9. **revol\_util**: The revolving line utilization rate reflects the borrower's credit utilization, impacting their credit score and risk.
10. **pub\_rec** and **pub\_rec\_bankruptcies**: These variables indicate derogatory public records and bankruptcies, highlighting potential credit risk.
11. **addr\_state** and **zip\_code**: Location might influence economic conditions and repayment capability.
12. **avg\_cur\_bal**: The average current balance of all accounts can provide insights into the borrower's financial stability.
13. **delinq\_2yrs**: The number of past-due incidences of delinquency over the past 2 years reflects credit behavior history.

```
In [53]: selected_columns = [  
    'id', 'loan_status', 'annual_inc', 'dti', 'emp_length',  
    'int_rate', 'purpose', 'term', 'open_acc', 'total_acc',  
    'revol_util', 'pub_rec', 'pub_rec_bankruptcies', 'addr_state',  
    'zip_code', 'avg_cur_bal', 'delinq_2yrs', 'loan_amnt'  
]  
  
d2 = d1[selected_columns].copy()
```

### Univariate Exploration

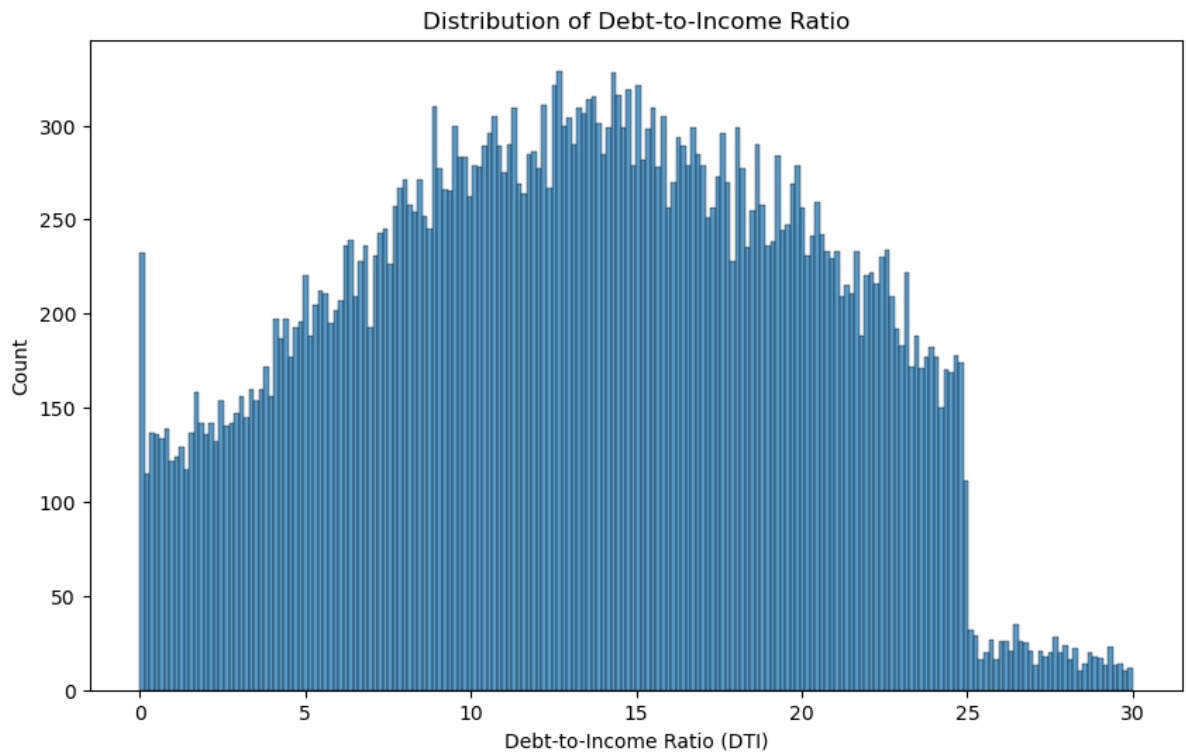
```
In [27]: # Explore loan status distribution  
plt.figure(figsize=(6, 4))  
sns.countplot(x='loan_status', data=d2)  
plt.title('Loan Status Distribution')
```

```
plt.xlabel('Loan Status')  
plt.ylabel('Count')  
plt.show()
```



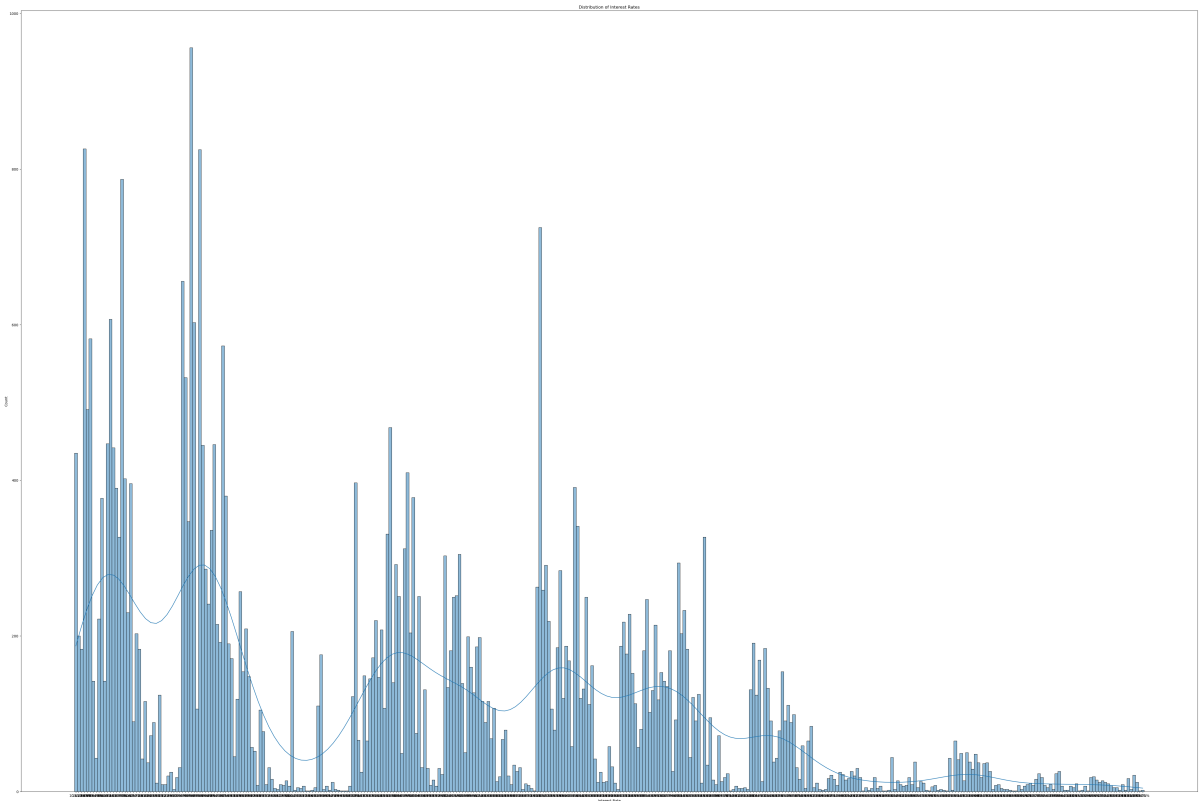
This initial visualization provides you with a quick overview of the proportion of loan statuses in the dataset. We can further investigate how other variables influence these statuses through bivariate analysis, correlation analysis, and more visualizations.

```
In [29]: plt.figure(figsize=(10, 6))  
sns.histplot(data=d2, x='dti', bins=200)  
plt.xlabel('Debt-to-Income Ratio (DTI)')  
plt.ylabel('Count')  
plt.title('Distribution of Debt-to-Income Ratio')  
plt.show()
```



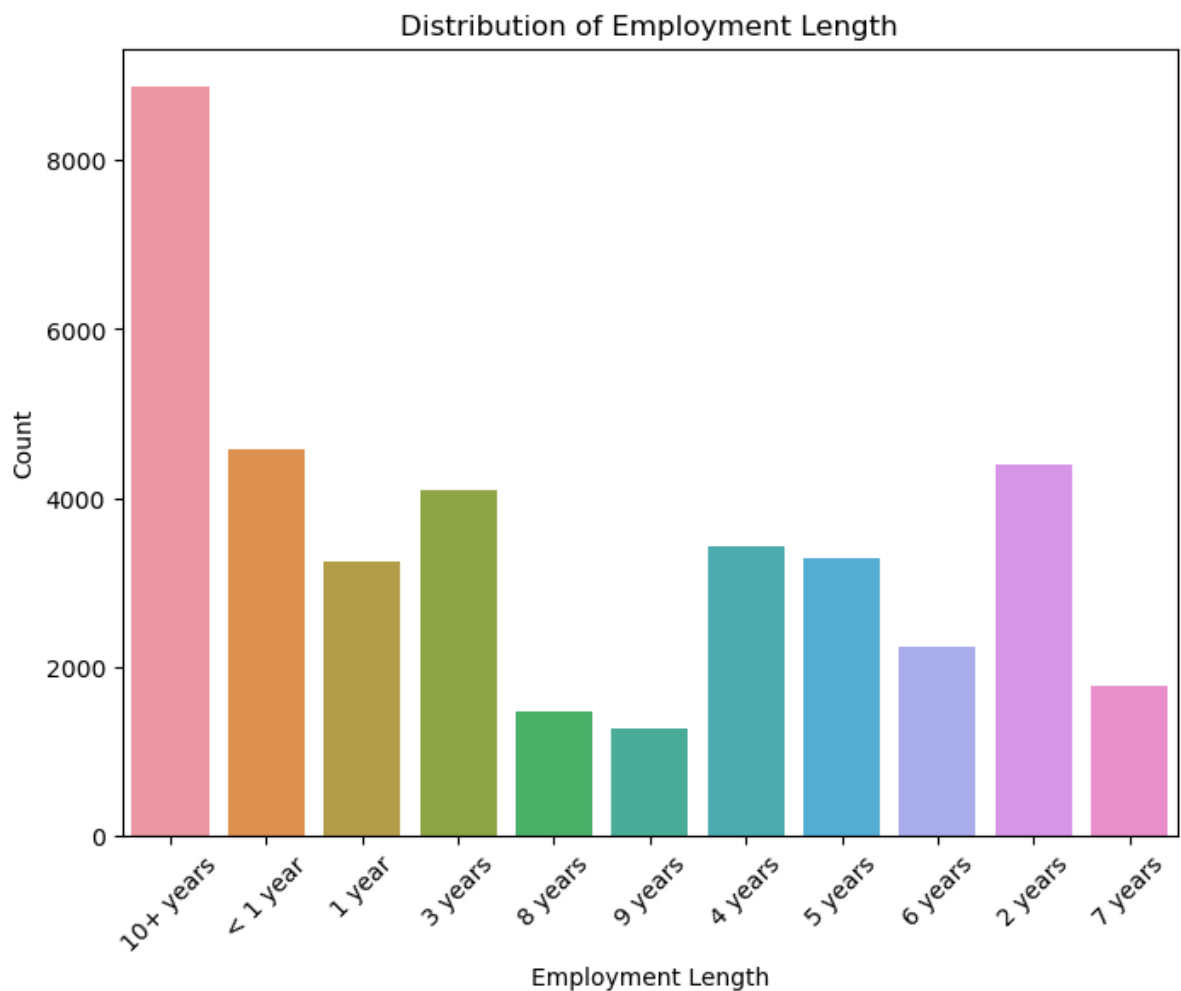
The Debt-to-Income Ratio (DTI) represents the ratio of a borrower's monthly debt payments to their monthly income. The histogram shows how many loans fall into different ranges of DTI values. You can observe whether there are any specific ranges of DTI that have higher or lower loan counts, which could indicate how this variable might influence loan default.

```
In [41]: plt.figure(figsize=(60, 40))
sns.histplot(data=d2, x='int_rate', bins=10, kde=True)
plt.xlabel('Interest Rate')
plt.ylabel('Count')
plt.title('Distribution of Interest Rates')
plt.show()
```



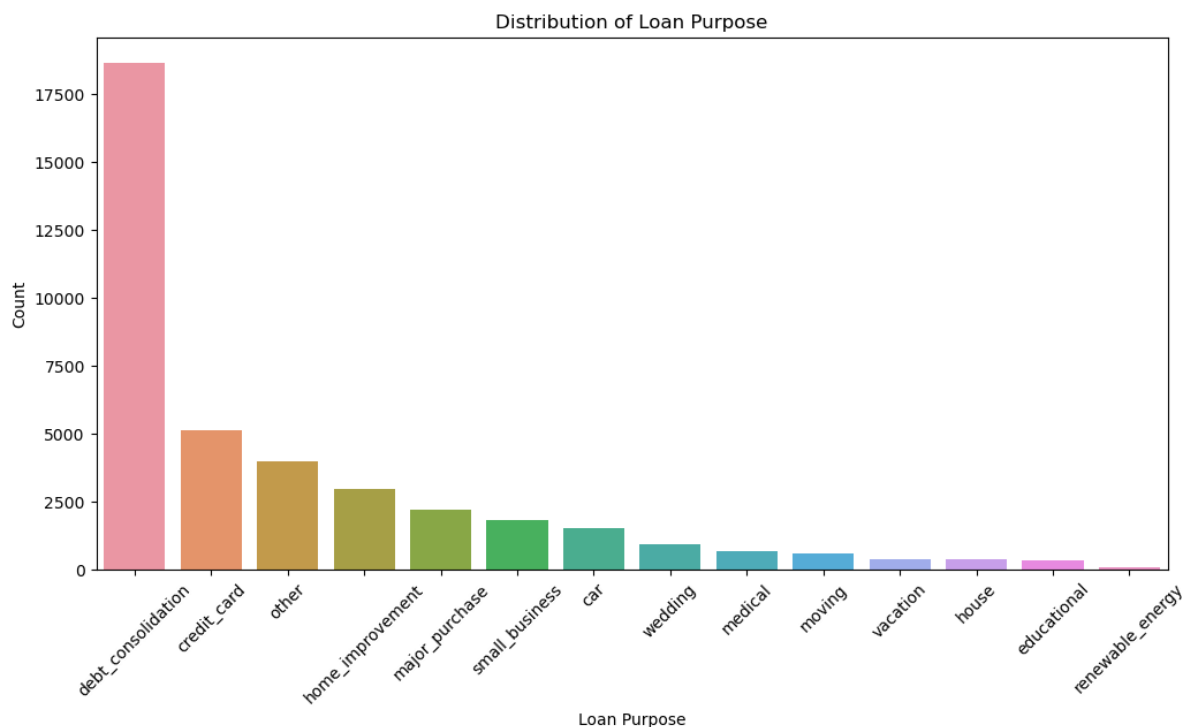
Interest rate is the percentage charged on the loan amount as interest. The histogram displays the frequency of loans with different interest rate ranges. This analysis helps you understand the spread of interest rates and whether higher interest rates are associated with different loan statuses.

```
In [43]: plt.figure(figsize=(8, 6))
sns.countplot(data=d2, x='emp_length')
plt.xlabel('Employment Length')
plt.ylabel('Count')
plt.title('Distribution of Employment Length')
plt.xticks(rotation=45)
plt.show()
```



This analysis provides the count of loans based on the length of employment of the borrowers. You can identify the distribution of borrowers across different employment lengths, such as '1 year', '2 years', '3 years', and so on. This variable could potentially provide insights into whether borrowers with longer employment lengths are less likely to default.

```
In [46]: plt.figure(figsize=(12, 6))
sns.countplot(data=d2, x='purpose', order=d2['purpose'].value_counts().index)
plt.xlabel('Loan Purpose')
plt.ylabel('Count')
plt.title('Distribution of Loan Purpose')
plt.xticks(rotation=45)
plt.show()
```



This analysis shows the count of loans based on their stated purpose. The x-axis lists various loan purposes, such as 'debt\_consolidation', 'credit\_card', 'home\_improvement', etc. By examining the distribution, you can understand the most common reasons borrowers are seeking loans, and whether certain purposes have different default rates.

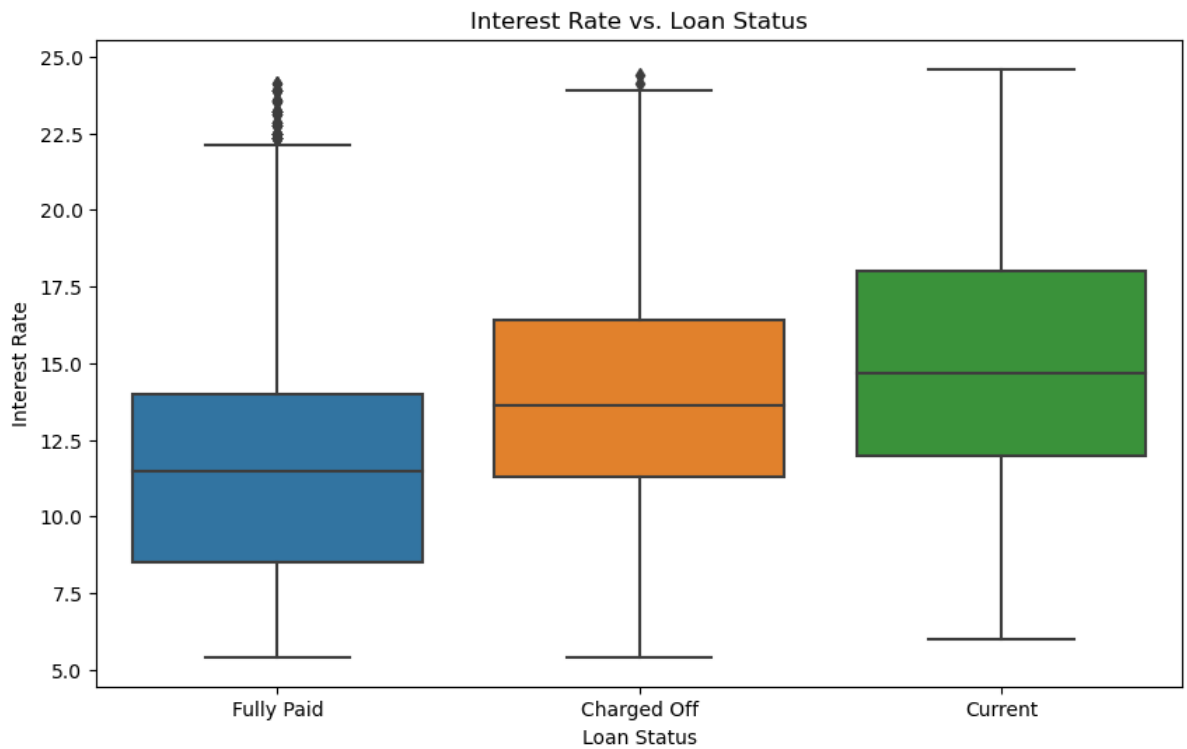
### ***Bivariate exploration***

In [49]: `intrt = d2.copy()`

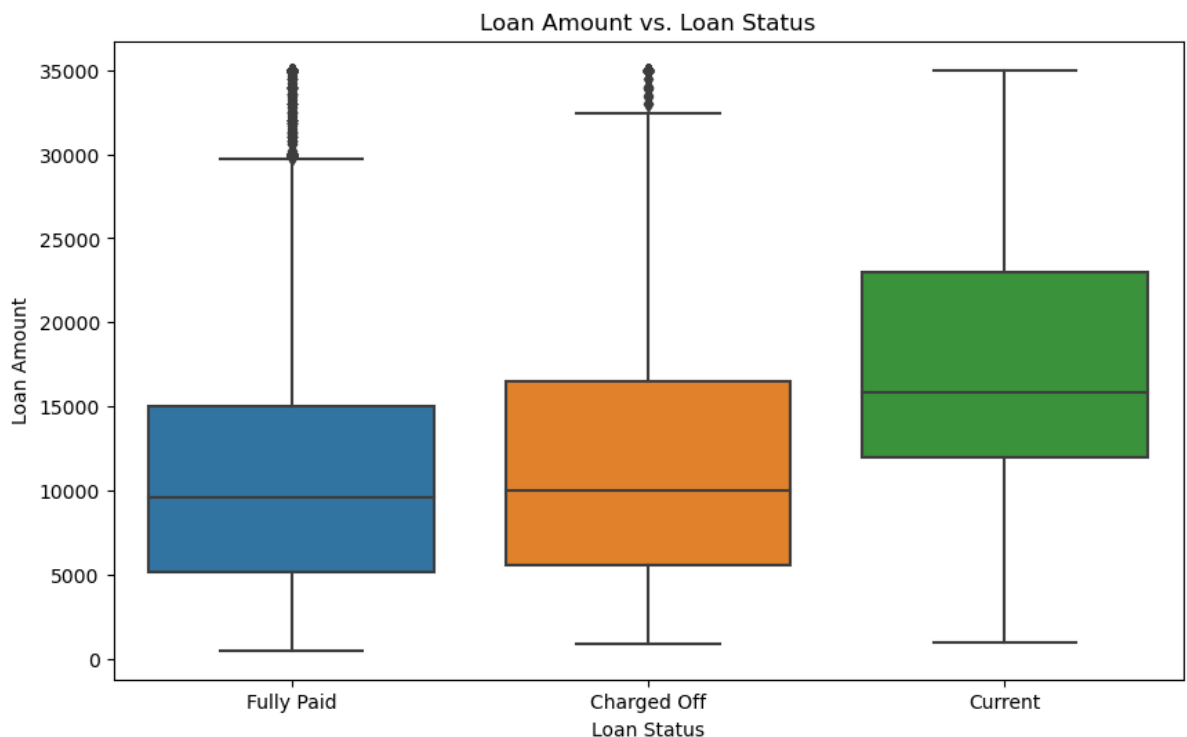
Creating a new copy of the dataset for plotting interest rate and loan status relation

```
In [50]: # Convert 'int_rate' column to numeric
intrt['int_rate'] = pd.to_numeric(intrt['int_rate'].str.replace('%', ''), errors='coerce')

# Create the box plot
plt.figure(figsize=(10, 6))
sns.boxplot(data=intrt, x='loan_status', y='int_rate')
plt.xlabel('Loan Status')
plt.ylabel('Interest Rate')
plt.title('Interest Rate vs. Loan Status')
plt.show()
```



```
In [54]: plt.figure(figsize=(10, 6))
sns.boxplot(data=d2, x='loan_status', y='loan_amnt')
plt.xlabel('Loan Status')
plt.ylabel('Loan Amount')
plt.title('Loan Amount vs. Loan Status')
plt.show()
```

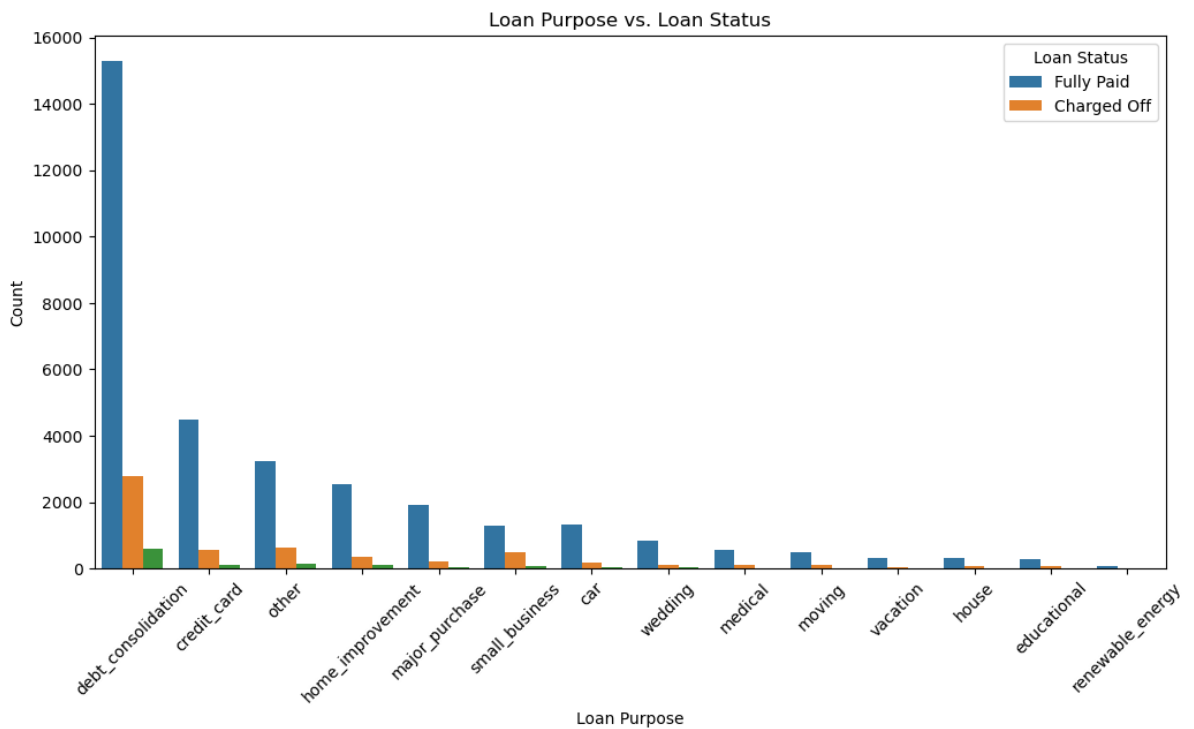


This box plot compares the distribution of loan amounts for fully paid loans and defaulted loans. It helps us identify any differences in loan amounts between the two groups.

```
In [61]: plt.figure(figsize=(12, 6))
sns.countplot(data=d2, x='purpose', hue='loan_status', order=d2['purpose'].v
plt.xlabel('Loan Purpose')
```

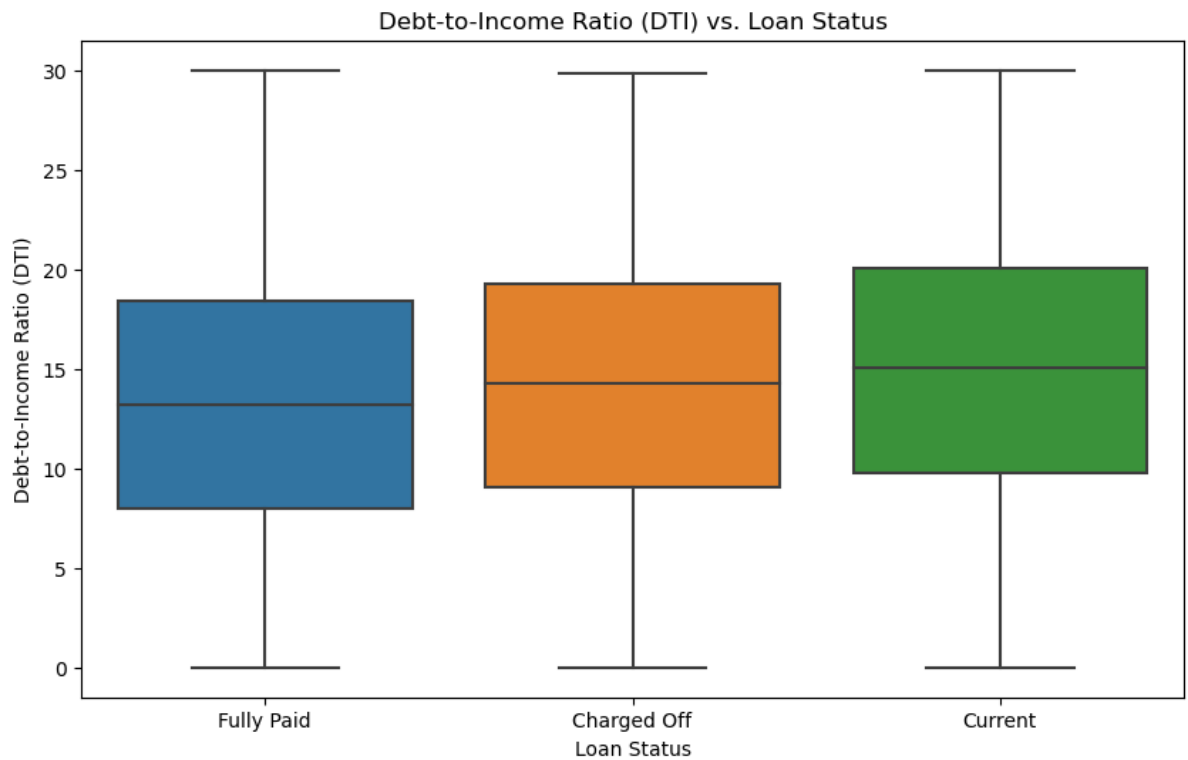


```
plt.ylabel('Count')
plt.title('Loan Purpose vs. Loan Status')
plt.xticks(rotation=45)
plt.legend(title='Loan Status', labels=['Fully Paid', 'Charged Off'])
plt.show()
```



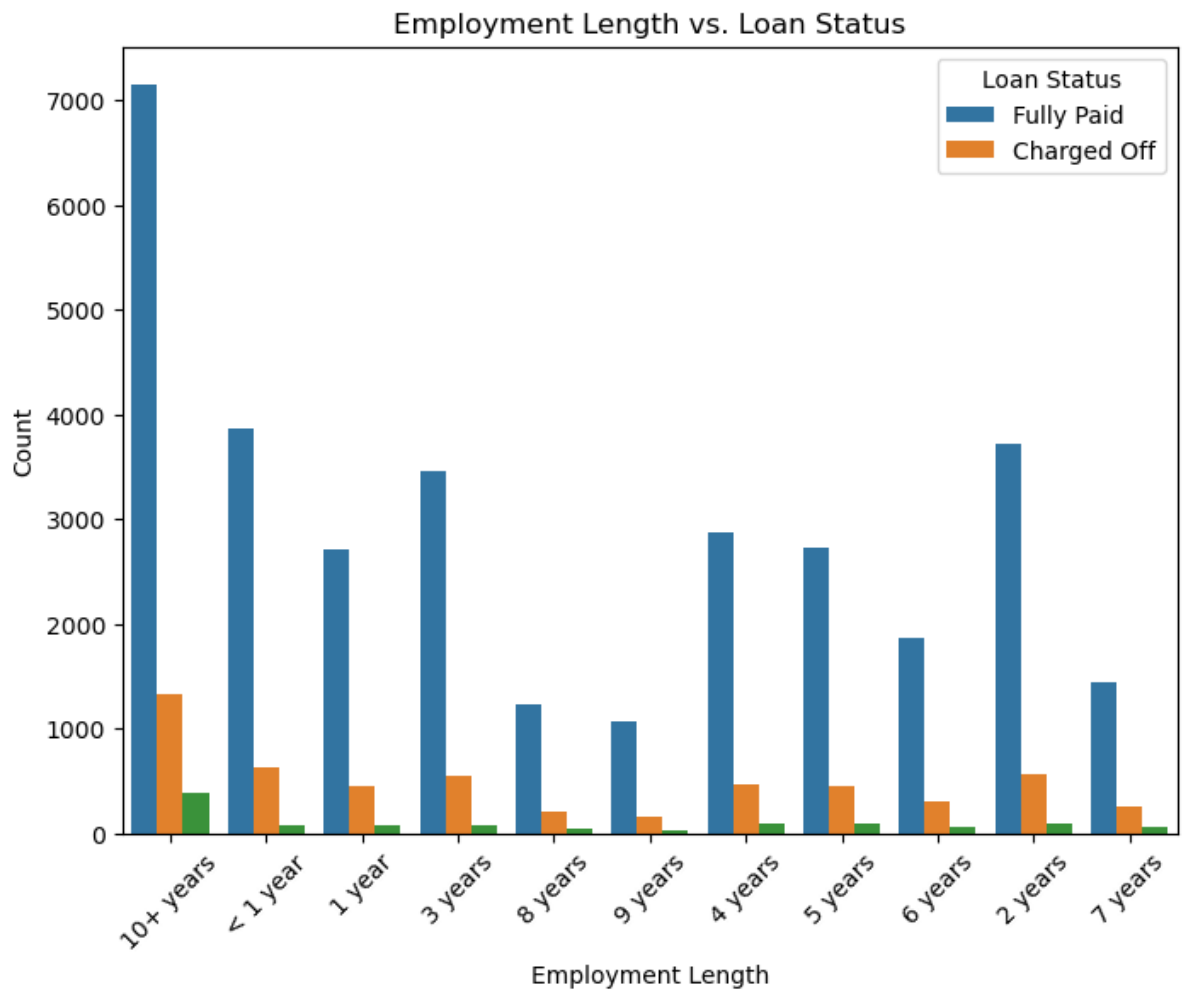
This count plot shows the distribution of loan purposes based on loan status. It helps us understand how different loan purposes relate to loan default, whether certain purposes have higher default rates, and how they compare to fully paid loans.

```
In [64]: plt.figure(figsize=(10, 6))
sns.boxplot(data=d2, x='loan_status', y='dti')
plt.xlabel('Loan Status')
plt.ylabel('Debt-to-Income Ratio (DTI)')
plt.title('Debt-to-Income Ratio (DTI) vs. Loan Status')
plt.show()
```



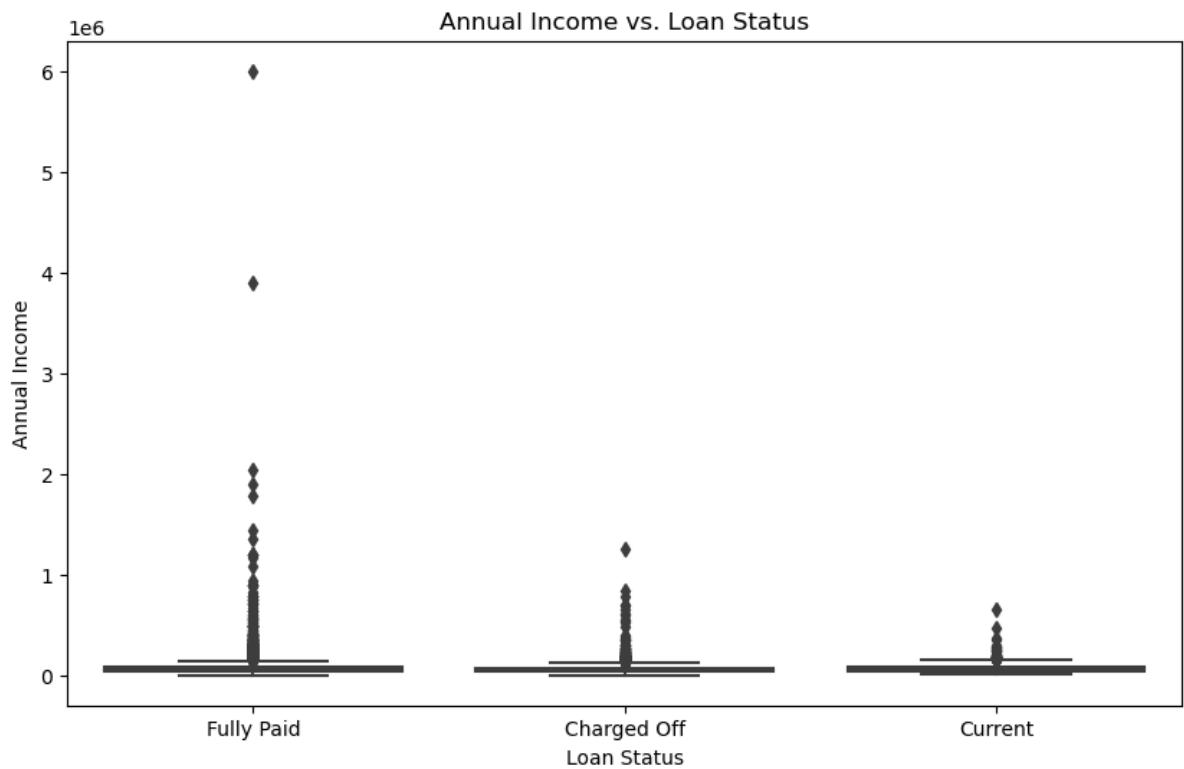
This box plot compares the distribution of debt-to-income ratios (DTI) for fully paid loans and defaulted loans. It helps us understand if borrowers with higher DTI ratios are more likely to default.

```
In [65]: plt.figure(figsize=(8, 6))
sns.countplot(data=d1, x='emp_length', hue='loan_status')
plt.xlabel('Employment Length')
plt.ylabel('Count')
plt.title('Employment Length vs. Loan Status')
plt.xticks(rotation=45)
plt.legend(title='Loan Status', labels=['Fully Paid', 'Charged Off'])
plt.show()
```



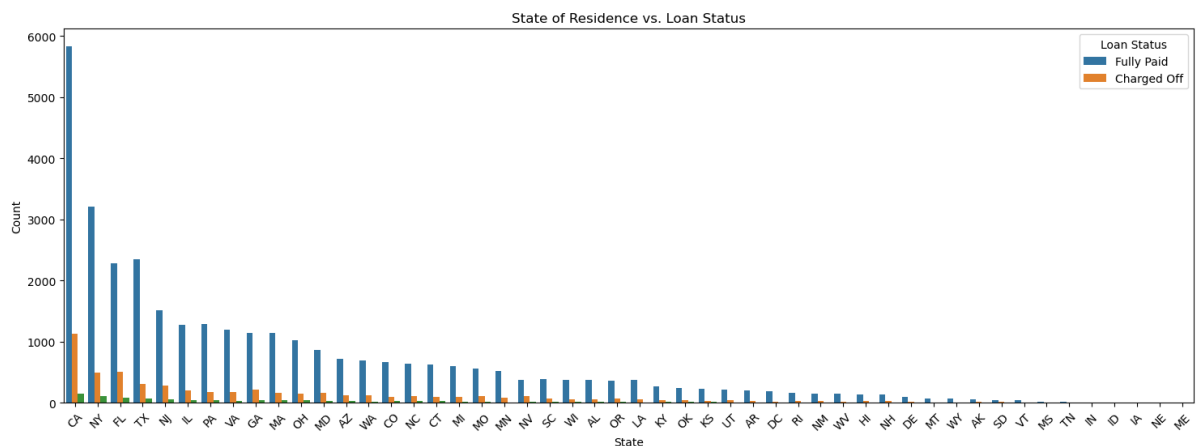
This count plot shows the distribution of loan statuses based on different employment lengths. It helps us understand how the length of employment relates to loan default.

```
In [67]: plt.figure(figsize=(10, 6))
sns.boxplot(data=d1, x='loan_status', y='annual_inc')
plt.xlabel('Loan Status')
plt.ylabel('Annual Income')
plt.title('Annual Income vs. Loan Status')
plt.show()
```



This box plot compares the distribution of annual incomes for fully paid loans and defaulted loans. It helps us understand if borrowers' annual income influences the likelihood of loan default.

```
In [68]: plt.figure(figsize=(18, 6))
sns.countplot(data=d2, x='addr_state', hue='loan_status', order=d2['addr_state'].order())
plt.xlabel('State')
plt.ylabel('Count')
plt.title('State of Residence vs. Loan Status')
plt.xticks(rotation=45)
plt.legend(title='Loan Status', labels=['Fully Paid', 'Charged Off'])
plt.show()
```



This count plot shows the distribution of loan statuses based on the state of residence. It helps us understand if loan default rates vary by state.

```
In [ ]: # d3 = d2.copy()
# d3 for derived variables creation
```

```
In [70]: d3['int_rate'] = pd.to_numeric(d3['int_rate'].str.replace('%', ''), errors='coerce')
# Define interest rate ranges and labels
```

```

interest_rate_ranges = [(0, 10), (10, 15), (15, 20), (20, 100)]
interest_rate_labels = ['Low', 'Medium', 'High', 'Very High']

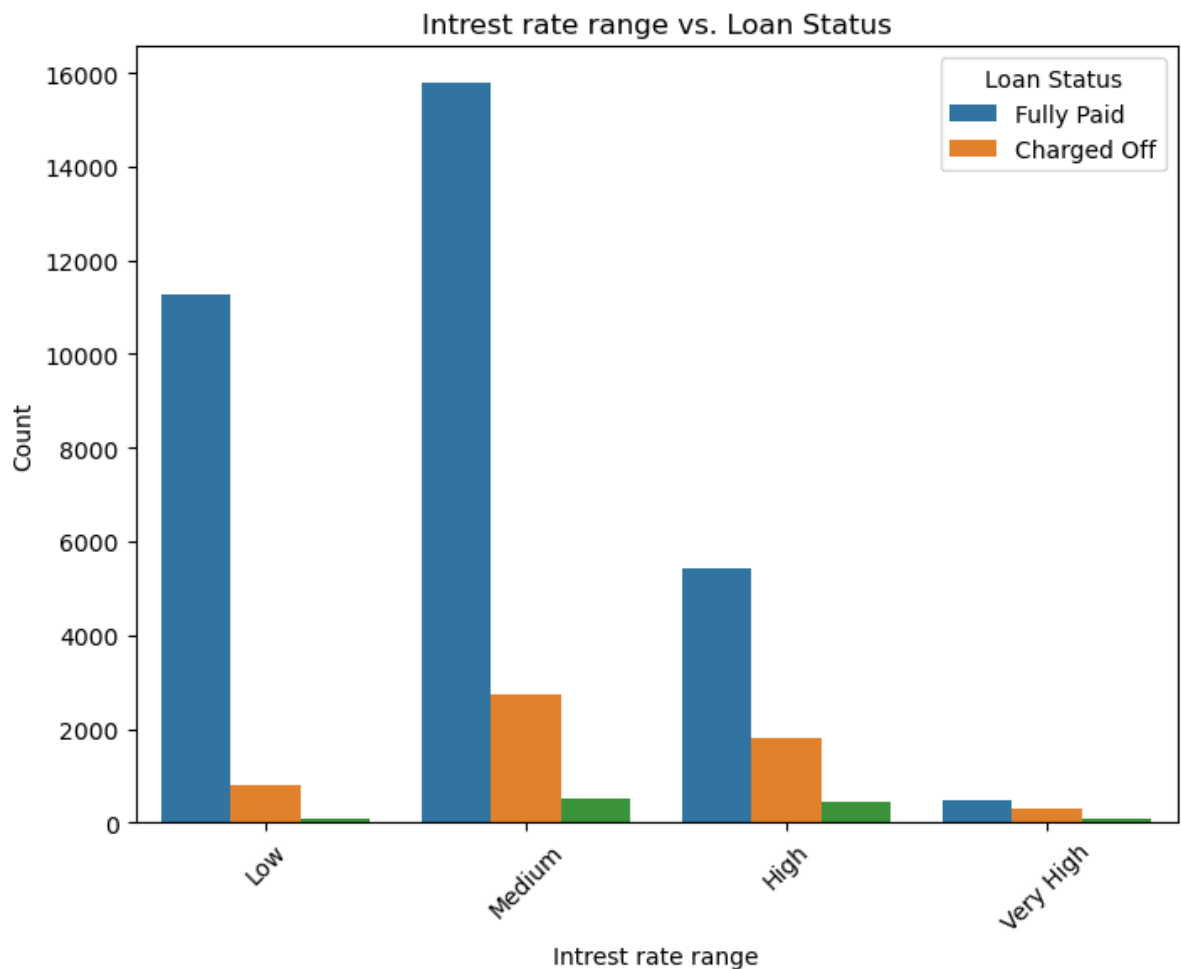
# Create a new column 'int_rate_range' with the interest rate ranges
d3['int_rate_range'] = pd.cut(d3['int_rate'], bins=[r[0] for r in interest_r
                                labels=interest_rate_labels, right=

```

```

In [74]: plt.figure(figsize=(8, 6))
sns.countplot(data=d3, x='int_rate_range', hue='loan_status')
plt.xlabel('Intrest rate range')
plt.ylabel('Count')
plt.title('Intrest rate range vs. Loan Status')
plt.xticks(rotation=45)
plt.legend(title='Loan Status', labels=['Fully Paid', 'Charged Off'])
plt.show()

```



Creating categorical bins for interest rates, such as 'low', 'medium', 'high', and 'very high'. This can provide a more understanding of how different interest rate ranges impact loan default.

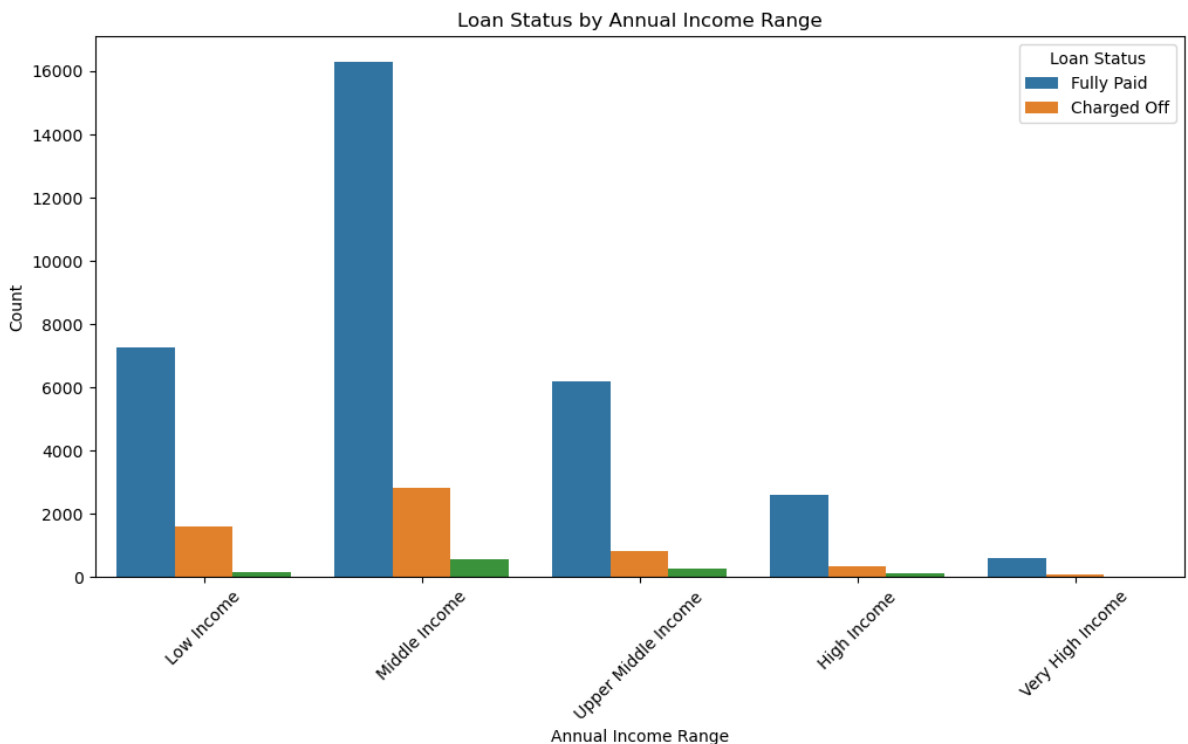
```

In [75]: income_ranges = [(0, 40000), (40000, 80000), (80000, 120000), (120000, 200000)]
income_labels = ['Low Income', 'Middle Income', 'Upper Middle Income', 'High Income']
d3['income_range'] = pd.cut(d3['annual_inc'], bins=[r[0] for r in income_ranges],
                             labels=income_labels, right=False)

plt.figure(figsize=(12, 6))
sns.countplot(data=d3, x='income_range', hue='loan_status', order=income_labels)
plt.xlabel('Annual Income Range')
plt.ylabel('Count')
plt.title('Loan Status by Annual Income Range')
plt.xticks(rotation=45)

```

```
plt.legend(title='Loan Status', labels=['Fully Paid', 'Charged Off'])
plt.show()
```



Creating income brackets, such as 'low income', 'middle income', and 'high income'. This can show whether borrowers in certain income ranges are more likely to default.

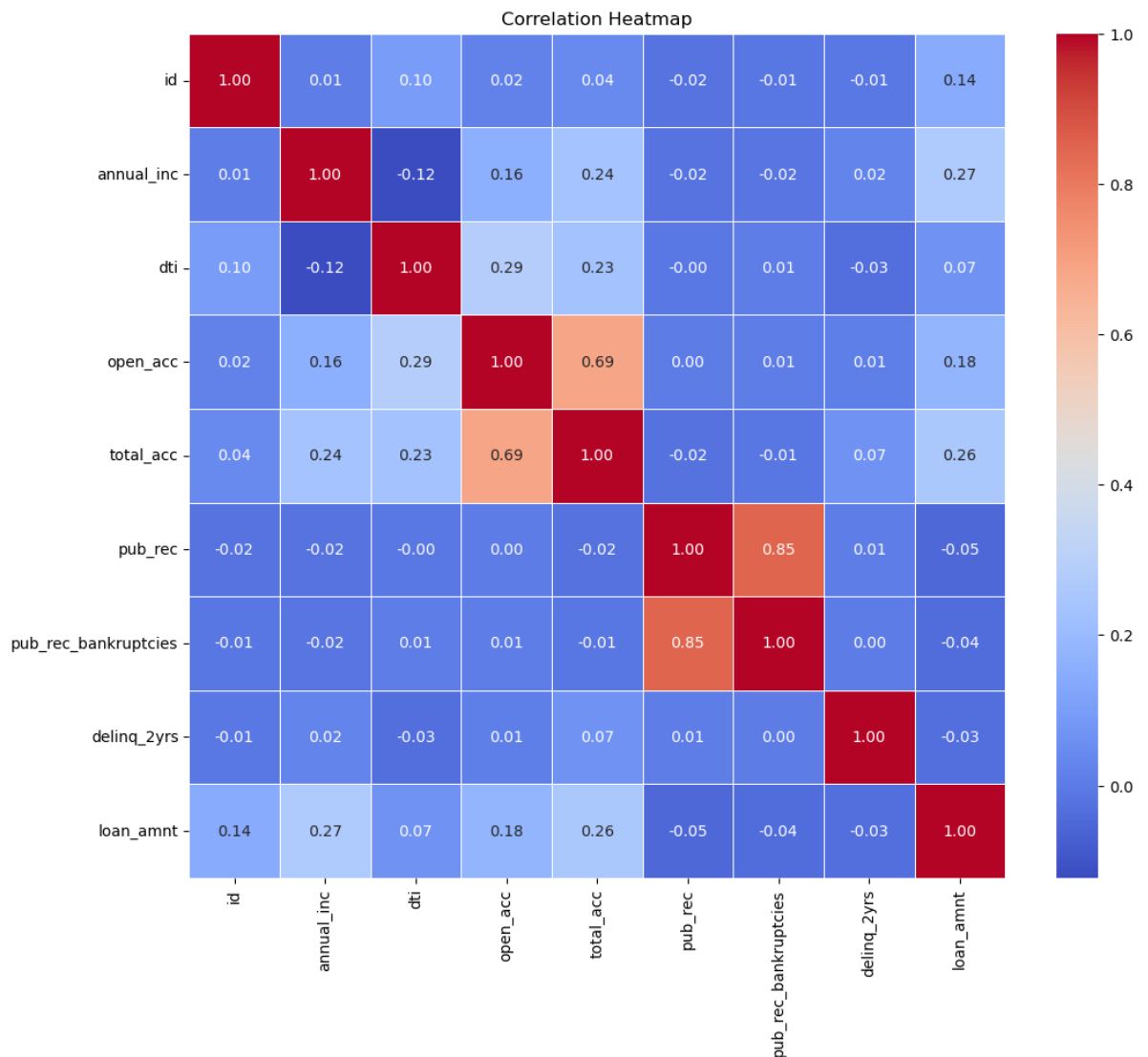
### Corelation matrix

```
In [79]: # d4 = d2.drop(columns=['avg_cur_bal'])
#removing the avg_cur_bal column cause its creating issues in the correlation
```

```
Out[79]: Index(['id', 'loan_status', 'annual_inc', 'dti', 'emp_length', 'int_rate',
        'purpose', 'term', 'open_acc', 'total_acc', 'revol_util', 'pub_rec',
        'pub_rec_bankruptcies', 'addr_state', 'zip_code', 'delinq_2yrs',
        'loan_amnt'],
        dtype='object')
```

```
In [78]: correlation_matrix = d4.corr()
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, cmap='coolwarm', annot=True, fmt=".2f", line
plt.title('Correlation Heatmap')
plt.show()
```

```
/var/folders/8j/l5qhm695n3cctcl4h0dzkqr0000gp/T/ipykernel_81762/2316707516.
py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only valid
columns or specify the value of numeric_only to silence this warning.
correlation_matrix = d4.corr()
```

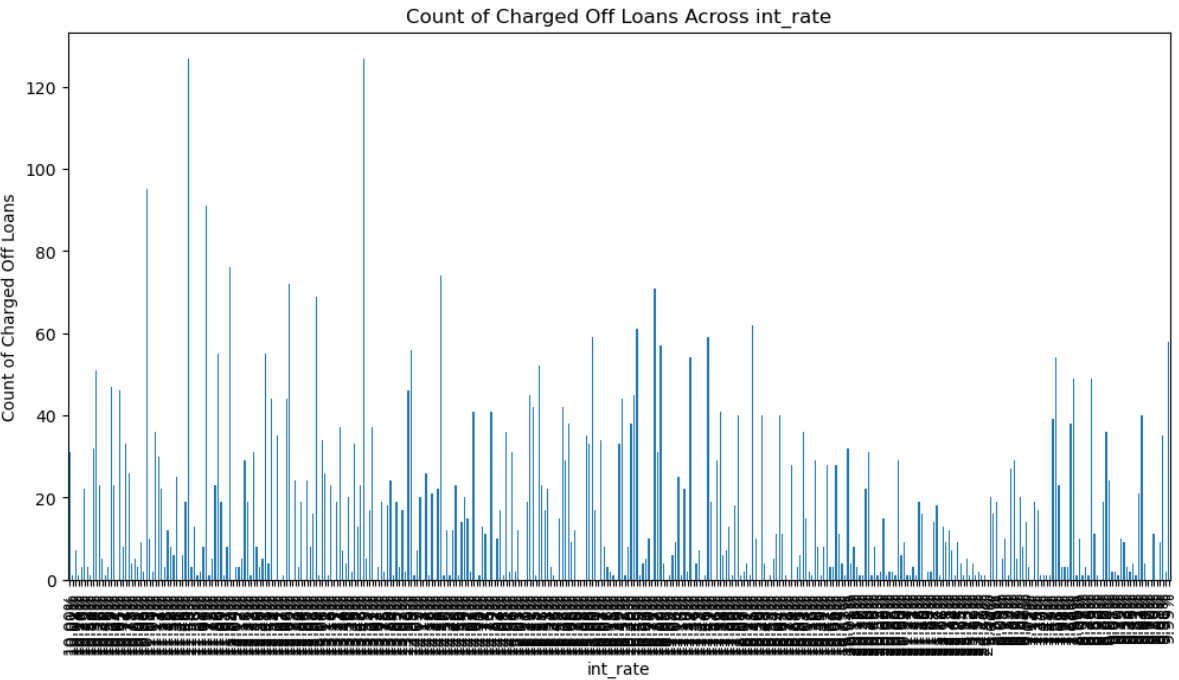
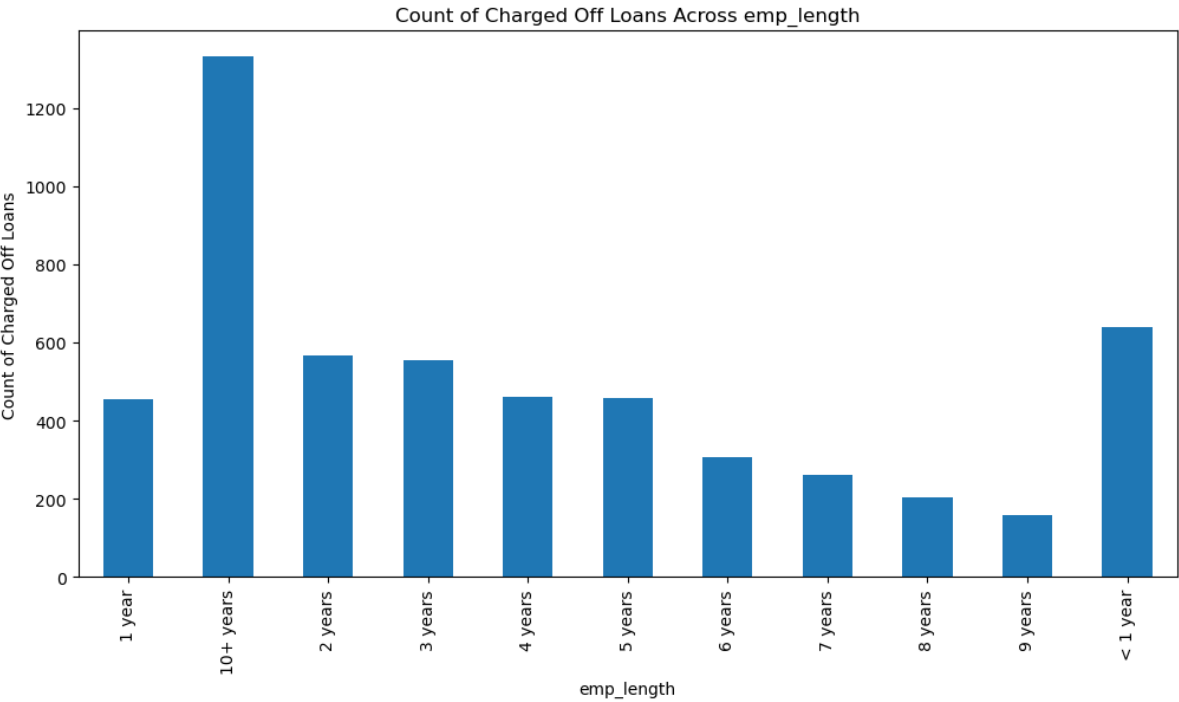


unable to find much relations here which are useful

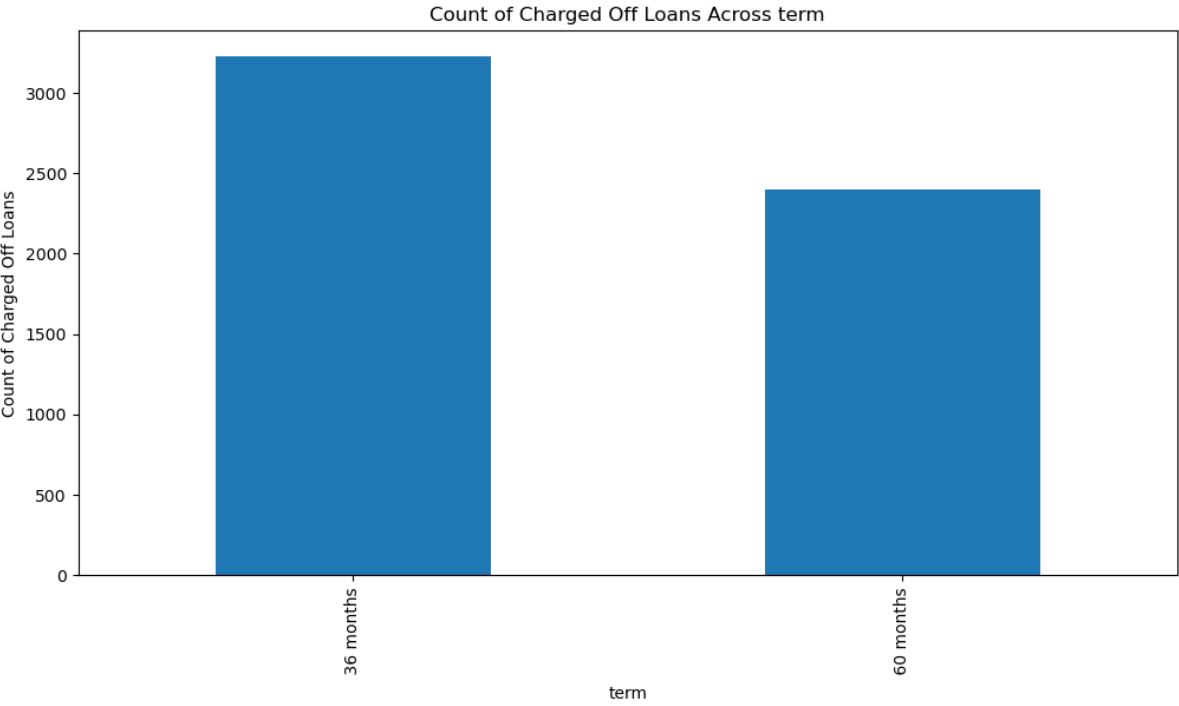
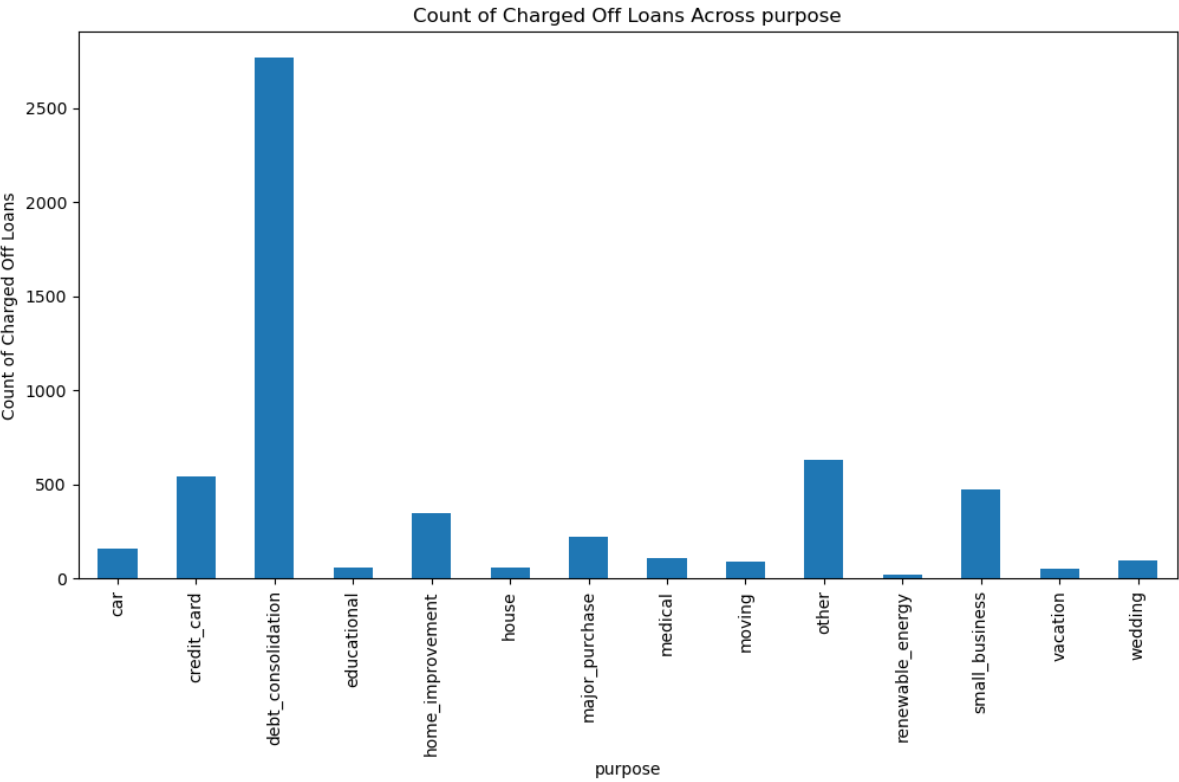
In [126... `d2.columns`

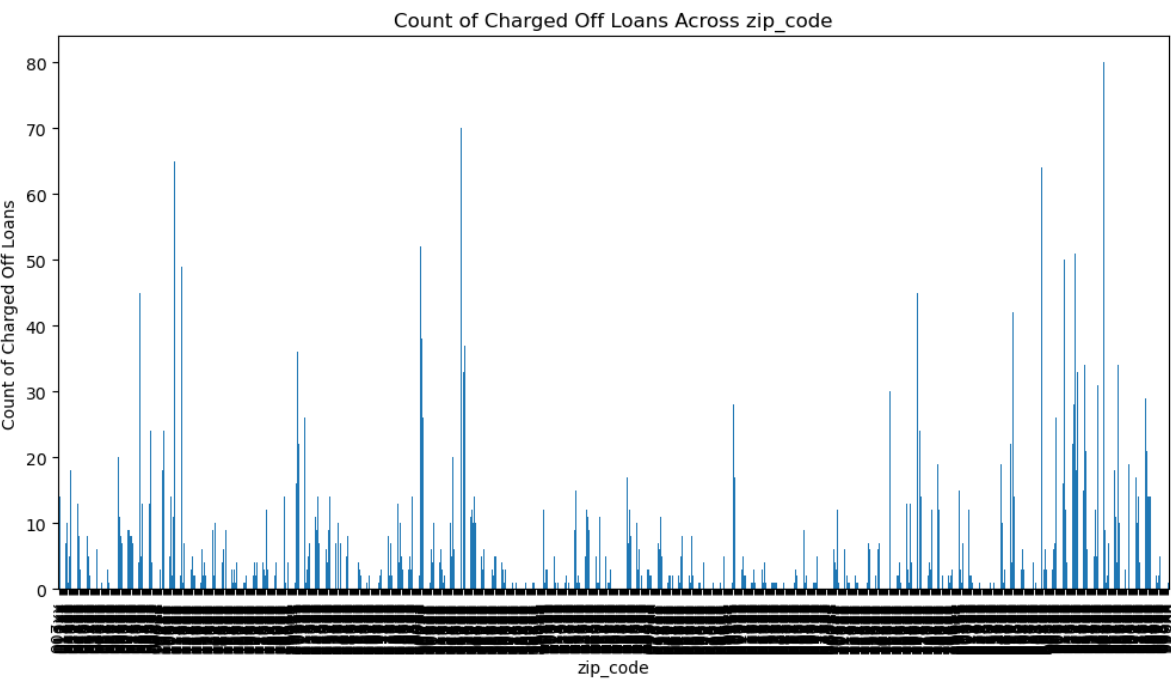
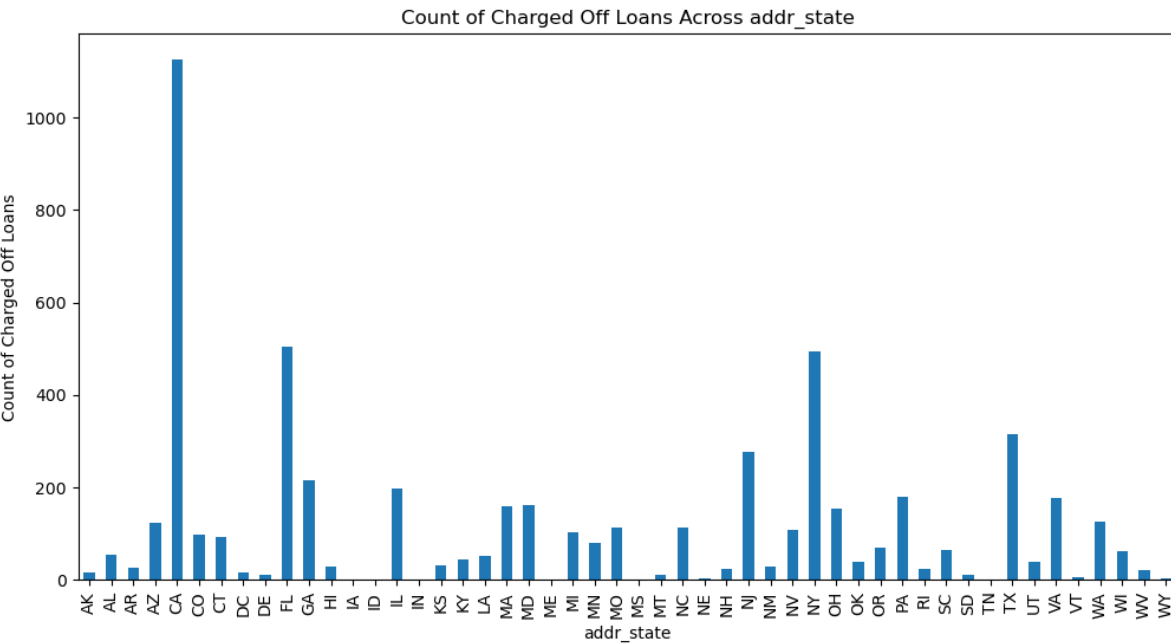
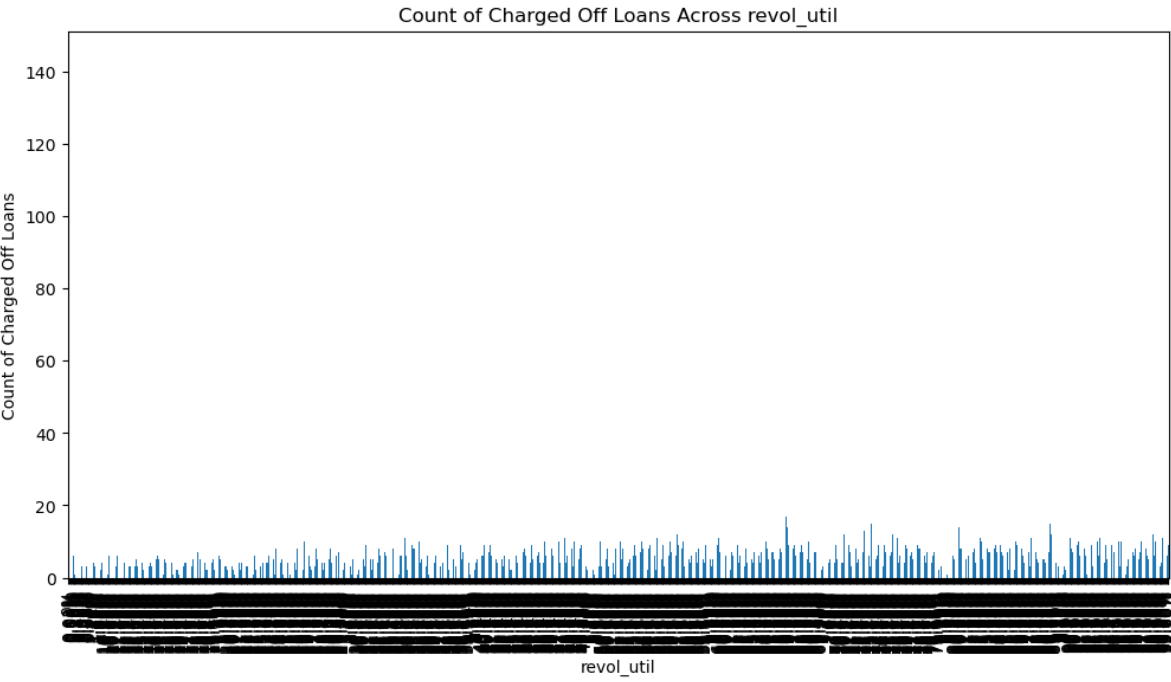
Out[126]: Index(['id', 'loan\_status', 'annual\_inc', 'dti', 'emp\_length', 'int\_rate', 'purpose', 'term', 'open\_acc', 'total\_acc', 'revol\_util', 'pub\_rec', 'pub\_rec\_bankruptcies', 'addr\_state', 'zip\_code', 'avg\_cur\_bal', 'delinq\_2yrs', 'loan\_amnt'], dtype='object')

```
In [129... categorical_columns = d2.select_dtypes(include='object').columns
strong_categorical = []
for column in categorical_columns:
    if column != 'loan_status':
        cross_tab = pd.crosstab(d2[column], d2['loan_status'])
        charged_off_counts = cross_tab['Charged Off']
        charged_off_counts.plot(kind='bar', figsize=(12, 6))
        plt.xticks(rotation=90)
        plt.title(f'Count of Charged Off Loans Across {column}')
        plt.xlabel(column)
        plt.ylabel('Count of Charged Off Loans')
        plt.show()
```









```
In [140... categorical_columns = d2.select_dtypes(include='number').columns
categorical_columns
```

```
Out[140]: Index(['id', 'annual_inc', 'dti', 'open_acc', 'total_acc', 'pub_rec',
        'pub_rec_bankruptcies', 'avg_cur_bal', 'delinq_2yrs', 'loan_amnt'],
        dtype='object')
```

**validating the defaulters across different numerical variables.**

### Annual Income

```
In [146... column='annual_inc'

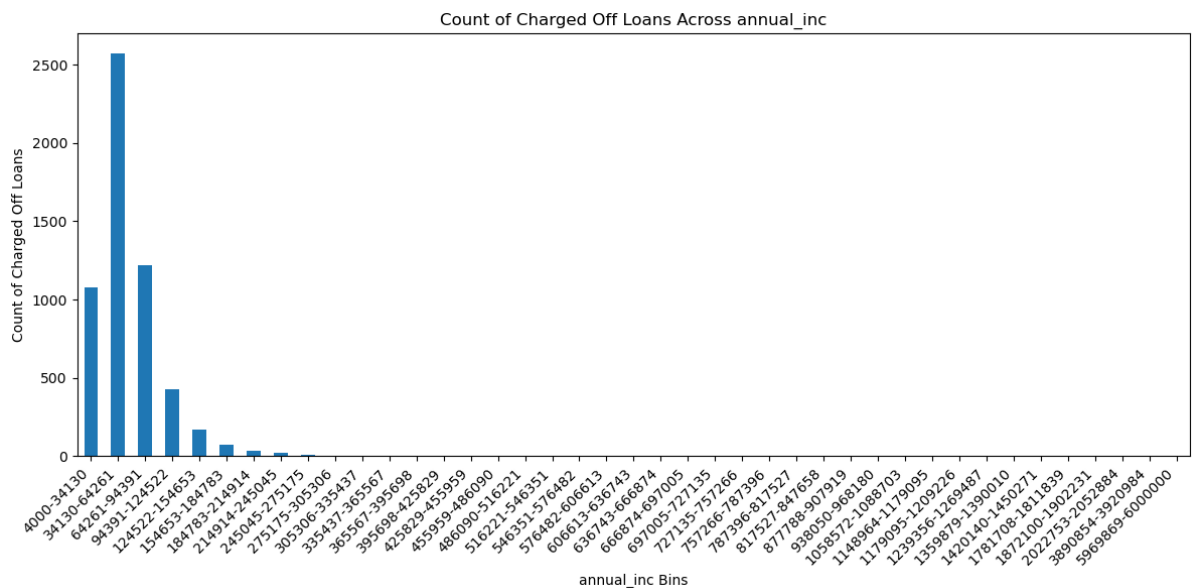
# Define bin edges
bin_edges = np.linspace(d2[column].min(), d2[column].max(), num=200)

# Create bins and labels
bin_labels = [f'{int(bin_edges[i])}-{int(bin_edges[i+1])}' for i in range(len(bin_edges)-1)]

# Assign each value to a bin
d2['income_bin'] = pd.cut(d2[column], bins=bin_edges, labels=bin_labels, include_lowest=True)

# Create cross-tabulation
cross_tab = pd.crosstab(d2['income_bin'], d2['loan_status'])
charged_off_counts = cross_tab['Charged Off']

# Plot the bar chart
charged_off_counts.plot(kind='bar', figsize=(12, 6))
plt.xticks(rotation=45, ha='right')
plt.title(f'Count of Charged Off Loans Across {column}')
plt.xlabel(f'{column} Bins')
plt.ylabel('Count of Charged Off Loans')
plt.tight_layout()
plt.show()
```



### DTI

```
In [173... column='dti'

# Define bin edges
bin_edges = np.linspace(d2[column].min(), d2[column].max(), num=15)
```

```

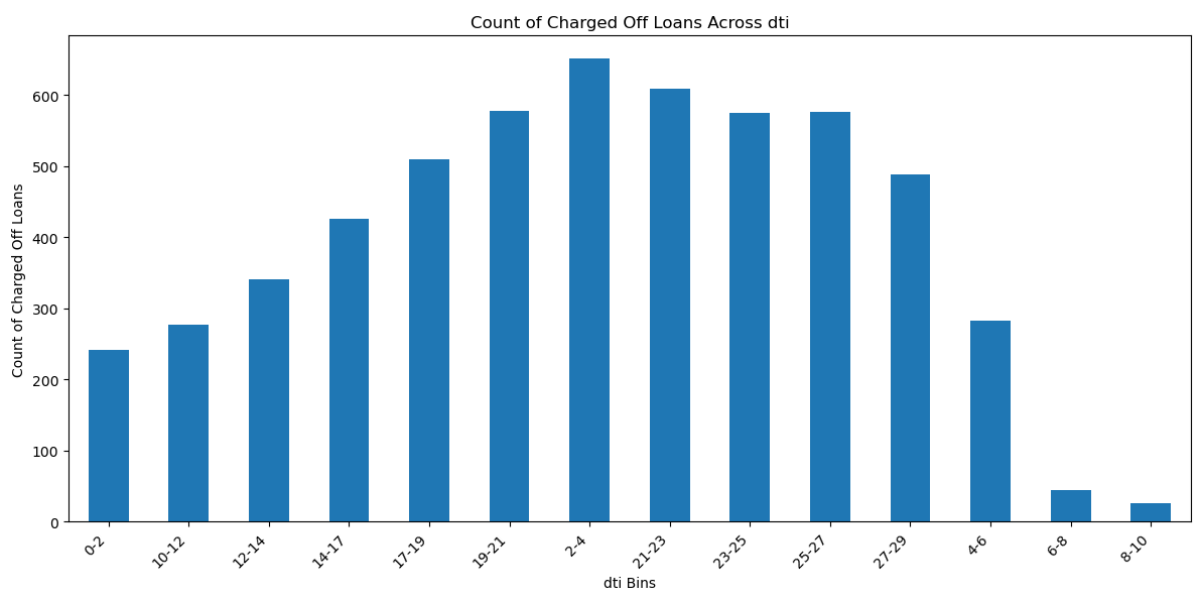
# Create bins and labels
bin_labels = [f'{int(bin_edges[i])}-{int(bin_edges[i+1])}' for i in range(len(bin_edges)-1)]
unique_bin_labels = np.unique(bin_labels) # Get unique labels

# Assign each value to a bin
d2[f'{column}_bins'] = pd.cut(d2[column], bins=bin_edges, labels=unique_bin_labels)

# Create cross-tabulation
cross_tab = pd.crosstab(d2[f'{column}_bins'], d2['loan_status'])
charged_off_counts = cross_tab['Charged Off']

# Plot the bar chart
charged_off_counts.plot(kind='bar', figsize=(12, 6))
plt.xticks(rotation=45, ha='right')
plt.title(f'Count of Charged Off Loans Across {column}')
plt.xlabel(f'{column} Bins')
plt.ylabel('Count of Charged Off Loans')
plt.tight_layout()
plt.show()

```



## Bankruptcy

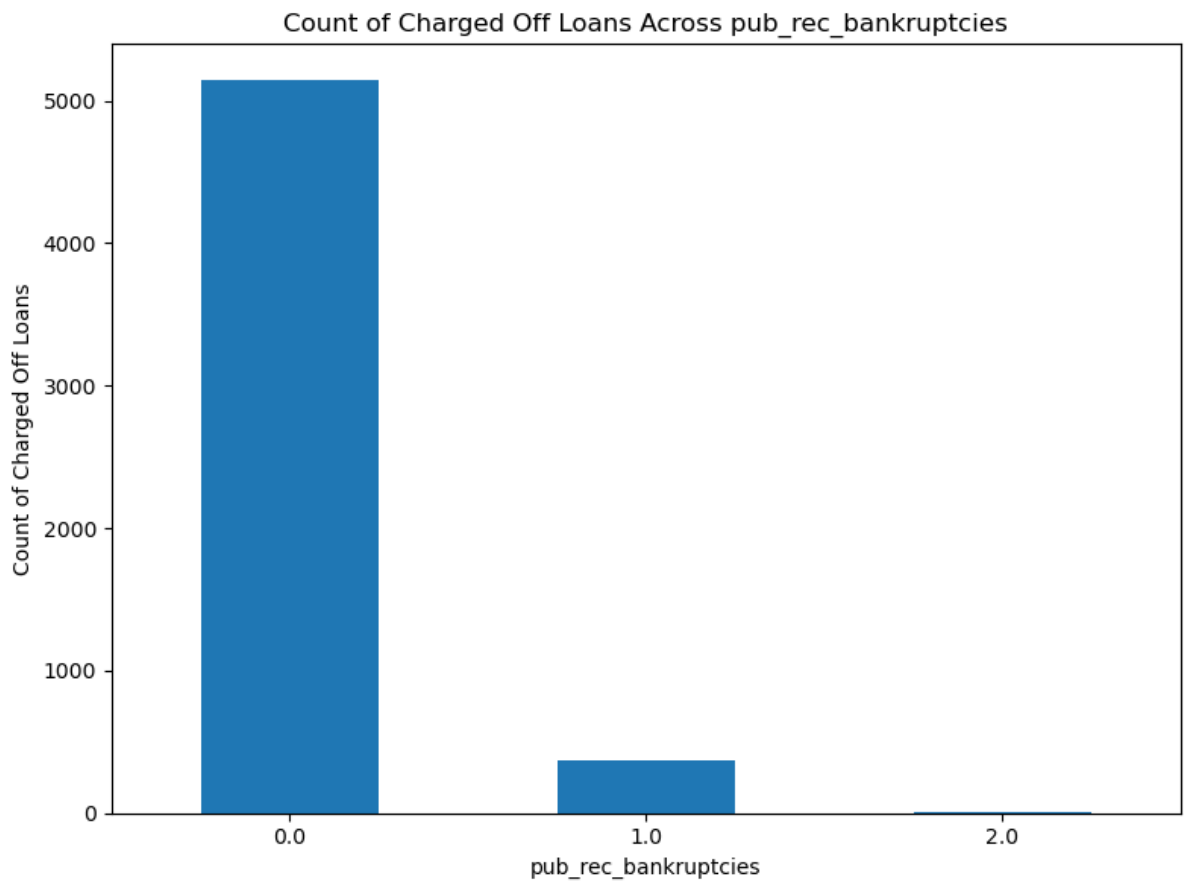
```

In [163... column='pub_rec_bankruptcies'

cross_tab = pd.crosstab(d2[column], d2['loan_status'])
charged_off_counts = cross_tab['Charged Off']

# Plot the bar chart
charged_off_counts.plot(kind='bar', figsize=(8, 6))
plt.title(f'Count of Charged Off Loans Across {column}')
plt.xlabel(column)
plt.ylabel('Count of Charged Off Loans')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()

```



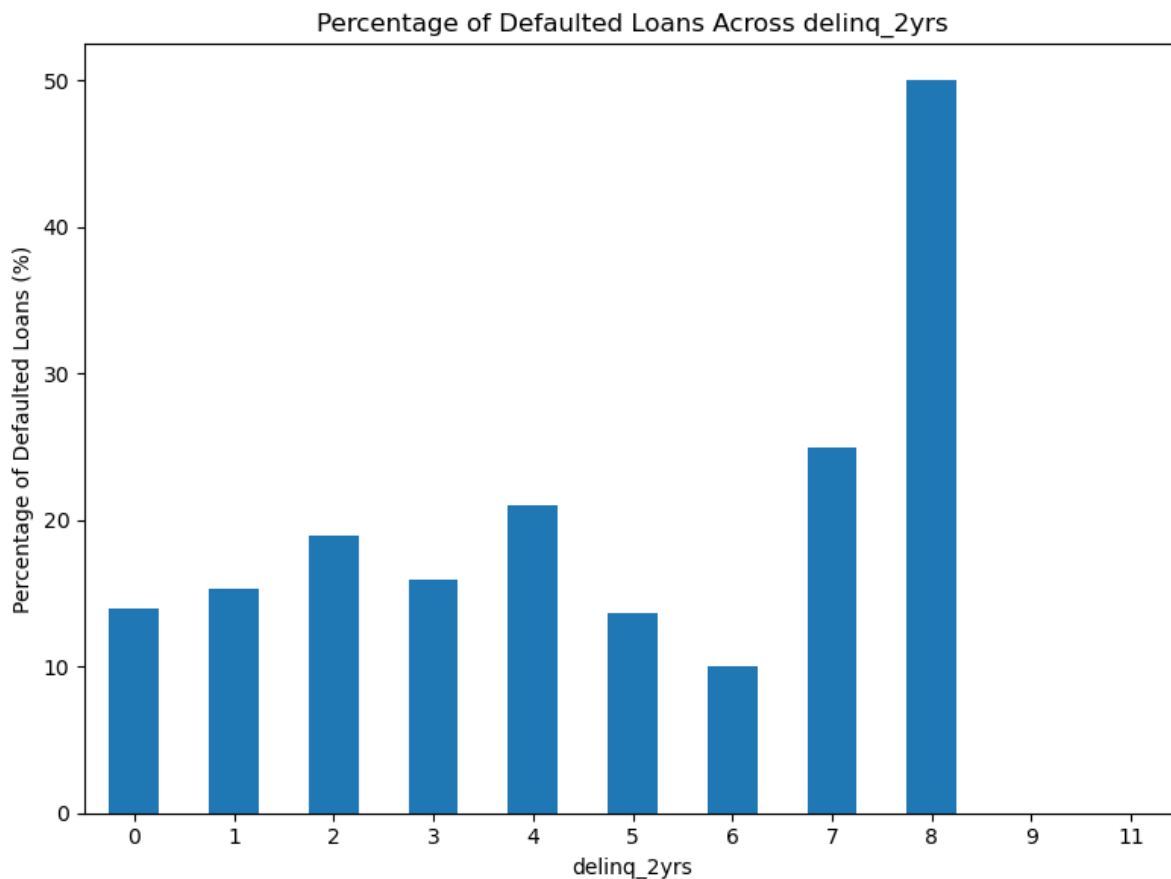
```
In [164... d2.delinq_2yrs.value_counts()
```

```
Out[164]: 0      35405
          1       3303
          2        687
          3        220
          4         62
          5         22
          6         10
          7          4
          8          2
          9          1
         11          1
          Name: delinq_2yrs, dtype: int64
```

```
In [169... column = 'delinq_2yrs'

# Create cross-tabulation
cross_tab = pd.crosstab(d2[column], d2['loan_status'])
percentage_defaulted = (cross_tab['Charged Off'] / cross_tab.sum(axis=1)) *

# Plot the bar chart
percentage_defaulted.plot(kind='bar', figsize=(8, 6))
plt.title(f'Percentage of Defaulted Loans Across {column}')
plt.xlabel(column)
plt.ylabel('Percentage of Defaulted Loans (%)')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



```
In [171... column = 'loan_amnt'

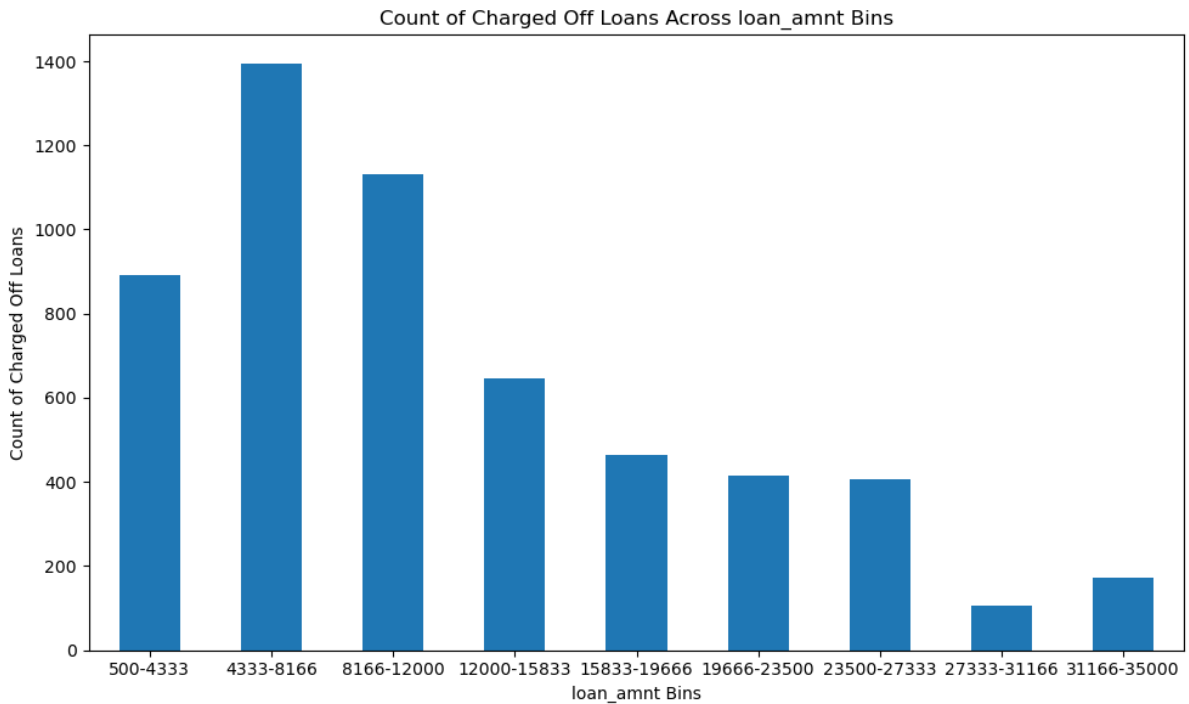
# Define bin edges
bin_edges = np.linspace(d2[column].min(), d2[column].max(), num=10)

# Create bins and labels
bin_labels = [f'{int(bin_edges[i])}-{int(bin_edges[i+1])}' for i in range(len(bin_edges)-1)]

# Assign each value to a bin
d2[f'{column}_bins'] = pd.cut(d2[column], bins=bin_edges, labels=bin_labels, include_lowest=True)

# Create cross-tabulation
cross_tab = pd.crosstab(d2[f'{column}_bins'], d2['loan_status'])
charged_off_counts = cross_tab['Charged Off']

# Plot the bar chart
charged_off_counts.plot(kind='bar', figsize=(10, 6))
plt.title(f'Count of Charged Off Loans Across {column} Bins')
plt.xlabel(f'{column} Bins')
plt.ylabel('Count of Charged Off Loans')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



## General Information

- Provide general information about your project here. This project involves an exploratory data analysis (EDA) of a dataset from a consumer finance company specializing in lending various types of loans to urban customers. The aim is to identify patterns and factors that indicate whether a person is likely to default on a loan. Through EDA, we seek to gain insights into the attributes of loan applicants that influence loan repayment tendencies.
- What is the background of your project? The company operates as an online loan marketplace, providing personal, business, and medical procedure loans. Risk assessment is crucial for lenders to minimize credit loss. By understanding the factors that correlate with loan default, the company can make informed decisions on loan approval, interest rates, and loan amounts.
- What is the business problem that your project is trying to solve? The primary business problem addressed by this project is the identification of attributes that significantly influence loan default. The company wants to minimize financial losses caused by defaults and enhance its risk assessment process. The project aims to provide insights that allow the company to take actions such as adjusting loan terms, interest rates, and loan approval criteria.
- What is the dataset that is being used? The dataset used in this project contains information about past loan applicants and whether they defaulted on their loans. It includes various attributes related to the applicants' financial situation, credit history, employment details, loan details, and more. The data includes both numerical and categorical variables, which will be analyzed to uncover patterns and trends that correlate with loan default. The analysis involves exploring the relationships between these attributes and loan repayment behavior to identify strong indicators of loan default.

# Conclusions

- Conclusion 1 from the analysis Optimal Loan Amount Allocation: The analysis suggests that loans with lower amounts, ranging from 500to1583, have a higher likelihood of default. On the other hand, loans with higher amounts, specifically from 15833to35000, exhibit fewer defaults. To mitigate risk, the company could consider focusing on larger loans with more comprehensive
- Conclusion 2 from the analysis Employment Stability Matters: The analysis indicates that individuals who have experienced 7 to 8 instances of default in the past two years have a higher tendency to default on loans. While employment length itself may not be the driving factor, this observation underscores the significance of evaluating applicants' credit history and repayment behavior. To enhance risk assessment, the company could incorporate more detailed credit history analysis, including the frequency of previous loan defaults, late payments, or delinquencies. This would provide a more accurate picture of an applicant's repayment habits and help make more informed lending decisions. Additionally, the company could offer financial education and counseling to applicants with a history of defaults to improve their financial management skills and increase their chances of successful loan repayment.
- Conclusion 3 from the analysis Debt-to-Income Ratio Threshold: Individuals with Debt-to-Income (DTI) ratios between 15 and 25 are more likely to default. This range might indicate a financial burden that affects repayment capabilities. The company could consider implementing a stricter DTI threshold for loan approval or offering financial counseling services to borrowers within this range, potentially reducing default rates.
- Conclusion 4 from the analysis Regional Risk Profiling: The analysis indicates that borrowers from certain regions such as CA, NY, FL, TX, NJ, and others tend to default more. To address this regional risk variation, the company could fine-tune its lending strategies for specific states. This might involve adjusting loan terms, interest rates, or requiring additional documentation for applicants from higher-risk regions.

## Technologies Used

**Pandas:** Pandas is a fundamental library for data manipulation and analysis. It provides data structures and functions to efficiently manipulate large datasets, clean and preprocess data, and perform various data transformations.

**Matplotlib:** Matplotlib is a popular plotting library that allows you to create a wide range of static, interactive, and animated visualizations. It's often used to generate various types of charts and graphs to visualize data patterns and relationships.

**Seaborn:** Seaborn is built on top of Matplotlib and provides a higher-level interface for creating visually appealing statistical visualizations. It offers a variety of functions to generate informative and aesthetically pleasing plots.



NumPy: NumPy is a library for numerical computing in Python. It provides support for arrays, mathematical functions, linear algebra operations, and other numerical computations. NumPy is often used in conjunction with Pandas for data manipulation and analysis.

## Acknowledgements

Give credit here.

- This project was inspired by... Upgrad
- References if any...

Pandas:

Pandas Documentation: <https://pandas.pydata.org/docs/> "Python for Data Analysis" book by Wes McKinney: <https://wesmckinney.com/pages/book.html> Matplotlib:

Matplotlib Documentation: <https://matplotlib.org/stable/contents.html> Matplotlib Tutorials: <https://matplotlib.org/stable/tutorials/index.html> Seaborn:

Seaborn Documentation: <https://seaborn.pydata.org/documentation.html> Seaborn Gallery: <https://seaborn.pydata.org/examples/index.html> NumPy:

NumPy Documentation: <https://numpy.org/doc/stable/> NumPy Quickstart Tutorial: <https://numpy.org/doc/stable/user/quickstart.html>

- This project was based on [this tutorial](#).

## Contact

Created by [github.com/lijink14] - feel free to contact me!

In [ ]: