LENDING CLUB CASE STUDY

Exploratory Data Analysis by Bhasa Mohapatra & Avik Kundu

Our Approach to the Case study

1. Define the Problem Statement and Objectives
2. Load and Understand the Dataset
3. Data Cleaning and Preparation
4. Exploratory Data Analysis (EDA)
5. Insights and Recommendations

Background- Lending Club Case Study

Background:

Consumer finance companies face challenges in balancing business growth with minimizing financial risk. Approving loans for likely defaulters leads to financial losses, while rejecting trustworthy applicants results in business loss. Risk analytics helps analyse historical data to predict defaults, enabling informed lending decisions and improved financial outcomes

Business Objective:

The primary goal of this case study is to leverage Exploratory Data Analysis (EDA) to identify key factors that influence loan defaults that can help the company to take better decision

Load and Understand the Dataset

Load Data:

Relevant libraries have been imported to read the CSV file and inspect on the data set to find overall information on the data set

Import relevant Libraries

```
[3]: import pandas as pd, numpy as np
     import seaborn as sns, matplotlib.pyplot as plt
     %matplotlib inline
```

Load and Inspect Data

```
[5]: ln = pd.read_csv('loan.csv', low_memory=False)
     ln.head()
```

```
ln.describe()
```

	id	member_id	loan_amnt	funded_amn
count	3.971700e+04	3.971700e+04	39717.000000	39717.00000
mean	6.831319e+05	8.504636e+05	11219.443815	10947.71319

```
[7]: ln.info(verbose = True)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39717 entries, 0 to 39716
Data columns (total 111 columns):
 #   Column                                  Dtype
---  ------                                  -----
 0   id                                      int64
```

<u>DATA AT A GLANCE :</u>

The Loan data set contains

- No of columns : 111
- Entries : 3971
- Minimum loan amount provide : 500
- Maximum loan amount provided: 35000
- Mean Interest Rate : 12.02%
- Mean dti : 13.3
- Max dti : 29.9

Load and Understand the Dataset cont...

Identifying relevant columns from the Data dictionary provided

The data dictionary was skimmed through to identify the relevant columns upon which the analysis is to be done.

ndex	Column	Type	Description
0	id	int64	A unique LC assigne
1	member_id	int64	A unique LC assigne
2	loan_amnt	int64	The listed amount c
3	funded_amnt	int64	The total amount cc
4	funded_amnt_inv	float64	The total amount cc
5	term	object	The number of payr
6	int_rate	object	Interest Rate on the
7	installment	float64	The monthly payme
8	grade	object	LC assigned loan gra
9	sub_grade	object	LC assigned loan sul
10	emp_title	object	The job title supplie
11	emp_length	object	Employment length
12	home_ownership	object	The home ownershi
13	annual_inc	float64	The self-reported ar
14	verification_status	object	Indicates if income
15	issue_d	object	The month which th
16	loan_status	object	Current status of th

18	url	object	U
19	desc	object	L
20	purpose	object	A
21	title	object	T
22	zip_code	object	T
23	addr_state	object	T
24	dti	float64	A
25	delinq_2yrs	int64	T
26	earliest_cr_line	object	T
27	inq_last_6mths	int64	T
28	mths_since_last_delinq	float64	T
29	mths_since_last_record	float64	T
30	open_acc	int64	T
31	pub_rec	int64	N
32	revol_bal	int64	T
33	revol_util	object	R
34	total_acc	int64	T
36	out_prncp	float64	F

9	37	out_prncp_inv	float64
0	38	total_pymnt	float64
1	39	total_pymnt_inv	float64
2	40	total_rec_prncp	float64
3	41	total_rec_int	float64
4	42	total_rec_late_fee	float64
5	43	recoveries	float64
6	44	collection_recovery_fee	float64
7	45	last_pymnt_d	object
8	46	last_pymnt_amnt	float64
9	47	next_pymnt_d	object
0	48	last_credit_pull_d	object
07	105	pub_rec_bankruptcies	float64

Columns highlighted in yellow being important columns for first hand assessment

Columns highlighted in Blue for further deeper analysis

Data Cleaning and Preparation

Handle Missing Values:

- Dropped off all the columns which had NULL
- Inspected for any rows with NULL values
- Identified columns with high amount of NULL values
- Removed the columns which were not relevant and were capturing only zero values

NULL COLUMNS AND ROWS

Data cleaning and Preparing

Find the columns with all 'NULL' Values

```
[9]: ln_null_columns =ln.columns[ln.isnull().all()]
     len(ln_null_columns)
     # INFERENCE : out of 111 columns 54 columns have NULL values >>
     #           we need to inspect the columns which all are not relevant and can drop them from the dataframe
[9]: 54
```

Find the rows with all 'NULL' Values

```
[11]: rows_all_null = ln[ln.isnull().all(axis=1)]
      len(rows_all_null)

      # INFERENCE : there are no rows with missing values
[11]: 0
```

DROPPING NULL COLUMNS

```
[17]: #dropping all the columns without any value
      ln = ln.drop(columns=ln_null_columns)

[19]: ln.info() # we are left with 57 columns after dropping 54 columns from the da

      <class 'pandas.core.frame.DataFrame'>
      RangeIndex: 39717 entries, 0 to 39716
      Data columns (total 57 columns):
       #   Column                Non-Null Count   Dtype
      ---  ------                --------------   -----
       0   id                    39717 non-null   int64
       1   member_id             39717 non-null   int64
       2   loan_amnt             39717 non-null   int64
       3   funded_amnt           39717 non-null   int64
       4   funded_amnt_inv       39717 non-null   float64
```

COLUMNS WITH HIGH NULL VALUES

```
[21]: ln.isnull().sum() # find any null in each column and check for irrelevance

      #INFERENCE : mths_since_last_delinq ,mths_since_last_record,next_pymnt_d have lot of NU

[21]: id                    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
      emp_title          2459
      emp_length         1075
      home_ownership        0
```

mths_since_last_delinq : 65%
mths_since_last_record : 93%
next_pymnt_d : 97%

DROPPING COLUMNS WITH ZERO VALUES

```
[23]: ln_zero_value_columns = ln.columns[(ln == 0).all()] # column with zero value and can be ignored

[25]: ln = ln.drop(columns=ln_zero_value_columns) #drop the columns with zero value
```

Post dropping the irrelevant columns we are left with 55 columns to work with

Data Cleaning and Preparation cont…

Analysing columns with one value:

- This is to check on any columns which are even though relevant covey a single value and message on which we necessary won't do much of analysis
- Post Analysis ,the columns were dropped

```
# Find columns with 1 unique value
unique_value_columns = [col for col in ln.columns if ln[col].nunique() == 1]
unique_values_table = pd.DataFrame({
    'Column Name': unique_value_columns,
    'Unique Value': [ln[col].unique()[0] for col in unique_value_columns]
})
unique_values_table

# INFERENCE: by studying the Data dictionary and the column value we understand
#              pymnt_plan is by default set as 'n'/NO => so we can drop this column
#              initial_list_status is set to 'f' => are from fractional loan program and can be dropped
#              collections_12_mths_ex_med is 0.0 => no collections done and hence can be ignored
#              all the policies are publicly available policies for the burrowers => we can drop the column
#              application_type are all INDIVIDUAL => so can be dropped
#              chargeoff_within_12_mths = 0.0 which means no defaults with in 12 months is none => irrelevant column
#              tax_liens is 0.0 => all are low risk [which is an internal assumption] and we can drop or ignore the column
```

OUR FINDINGS:

- **pymnt_plan is by default set as 'n'/NO** : we can drop this column
- **initial_list_status is set to 'f'** : all entries are from fractional loan program and can be dropped
- **collections_12_mths_ex_med is 0.0** : no collections done and hence can be ignored
- **all the policies are publicly available policies for the burrowers** : we can drop the column
- **application_type are all INDIVIDUAL** : can be dropped
- **chargeoff_within_12_mths = 0.0 :** which means no defaults with in 12 months is none => irrelevant column
- **tax_liens is 0.0 :** all are low risk [which is an internal assumption] and we can drop or ignore the column

7 more columns were dropped from the data set and left with 48 columns to work with

Data Cleaning and Preparation cont…

- **Changing data types and extracting derived columns**
- **Creating subset of the data for analysis**

  Figuring out the defaulters list and creating a separate data frame to do further analysis

Changing data types and extracting derived columns

1. Interest rate from object to float removing % from data

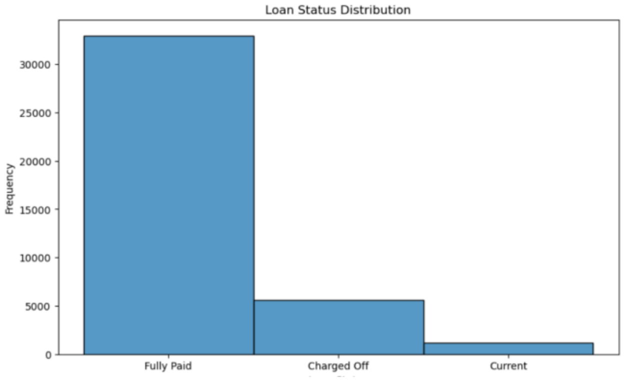
```
# Changing the data type of interest rate column
ln['int_rate'] = ln['int_rate'].str.replace('%','', regex=False).astype(float)
```

2. Changing issue date from object to date type and extracting 'Year' column from the same

```
#### Change 'issue_d' column to date type
ln['issue_d'].value_counts()
ln['issue_d'] = pd.to_datetime(ln['issue_d'], format='%b-%y')
```

```
# Extract Year column from the loan issue date
ln['loan_issue_Year'] = ln['issue_d'].dt.year
```

Creating subset of the data for analysis




```
# Check for the count of Loan status of all the customers
ln['loan_status'].value_counts()
# INFERENCE : 5627 are defaulters in the data set which is 14.2% of the dataset
```

```
loan_status
Fully Paid      32950
Charged Off      5627
Current          1140
```

Creating a separate defaulters list for further analysis

```
# create a separate data frame of defaulter list to analyse further
ln_d =ln[ln['loan_status']=='Charged Off']
```

```
ln_d.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5627 entries, 1 to 39688
Data columns (total 48 columns):
```

A new DF with 5627 entries were created

Data Cleaning and Preparation cont…

Analysing the defaulters’ data set created

1. Check for NULL value columns

```
ln_d.isnull().sum() #Check for any columns with only NULL Values in defaulters lis
```

```
id                       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
emp_title              484
emp_length             228
home_ownership           0
annual_inc               0
verification_status      0
issue_d                  0
loan_status              0
url                      0
desc                  1802
```

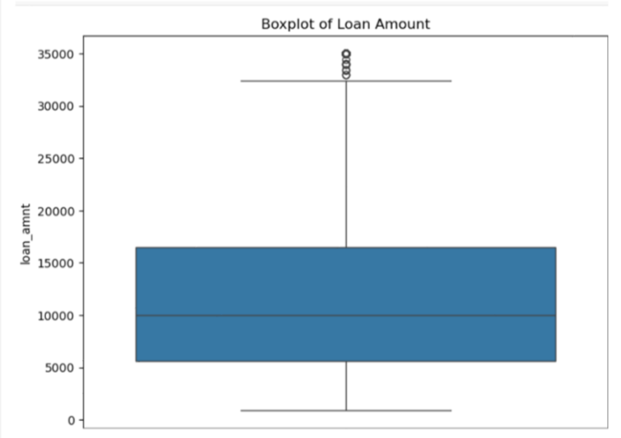
2. Find Duplicate Rows information

Find any duplicate row information in the defaulters data set

```
#Find any duplicate rows
ln_d.duplicated().sum()
```

```
0
```

3. Check for outliers



4. Find outlier Data

```
# Define function to find outlier
def detect_outliers_iqr(data, column):
    Q1 = data[column].quantile(0.25)
    Q3 = data[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Identifying outliers
    outliers = data[(data[column] < lower_bound) | (data[column] > upper_bound)]
    return outliers

# Find the outliers for Loan Amount coulmn
ln_amt_o = detect_outliers_iqr(ln_d, 'loan_amnt')
ln_amt_o['loan_amnt'].value_counts()
```

```
loan_amnt
35000    150
33000      2
34000      2
33950      2
33425      1
34475      1
33500      1
```

5. Issuance Year of the outlier Loan Amount

```
# Find in which year the oulier loans were issues
ln_amt_o['loan_issue_Year'].value_counts()
```

```
loan_issue_Year
2011    159
```

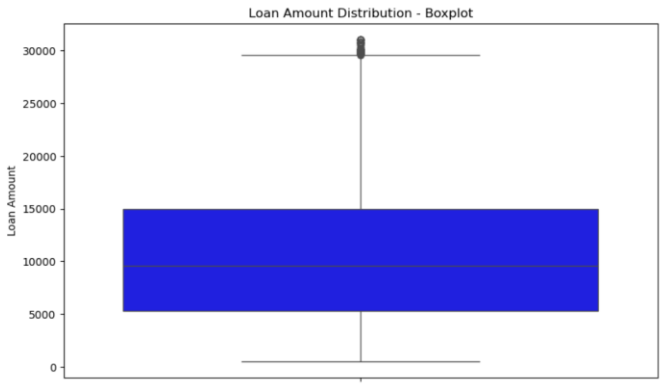
6. Drop the outliers

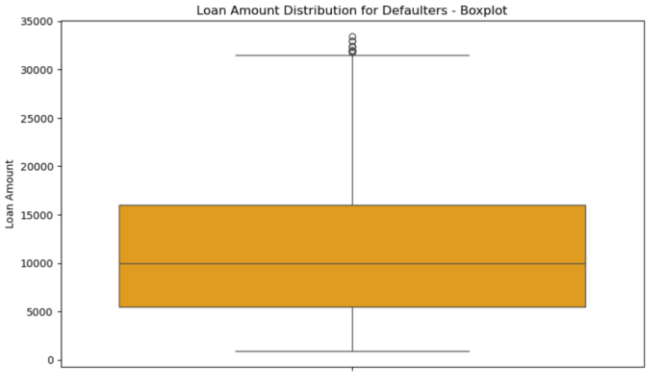
```
# Drop the outliers
ln_d = ln_d[~((ln_d['loan_amnt'] >= 33500) & (ln_d['loan_amnt'] <= 35000))]
```

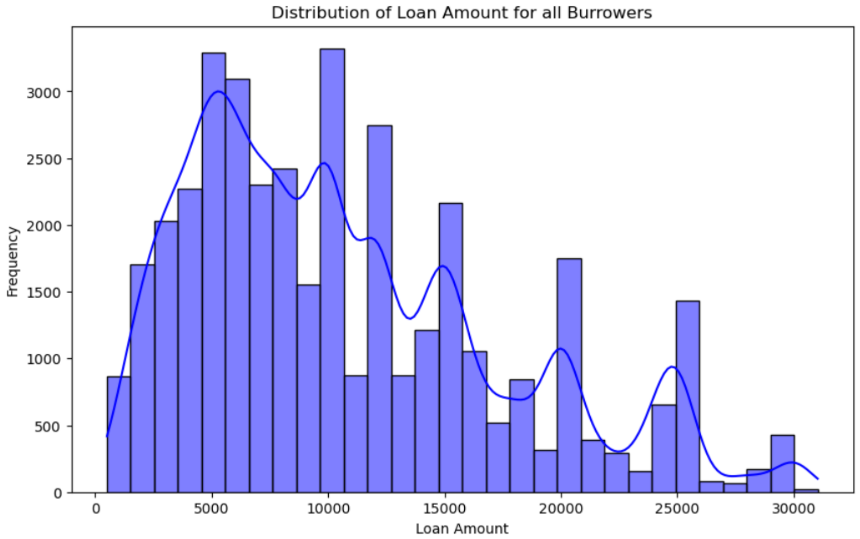
An attempt to find if the outliers are old records. However, seems the outliers loan amounts were given during recent time and the defaulters are higher during 2011

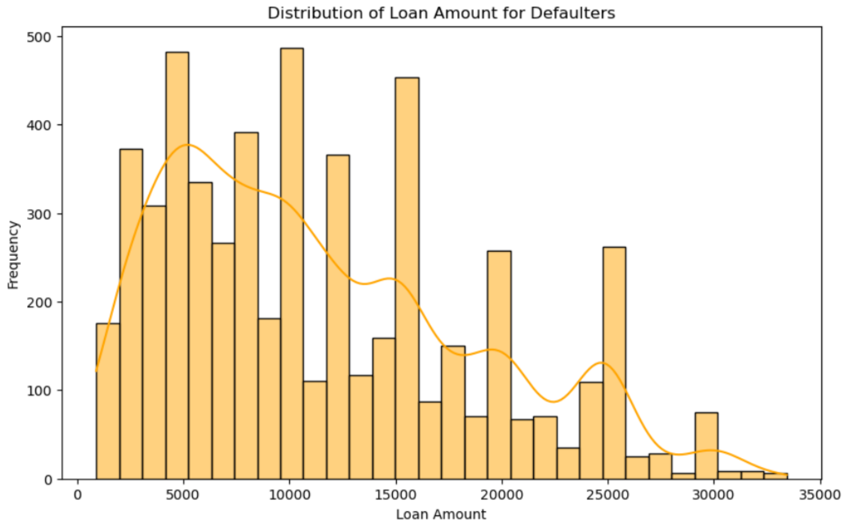
Exploratory Data Analysis (EDA)

Univariate Analysis on Loan Amount









OBSERVATION

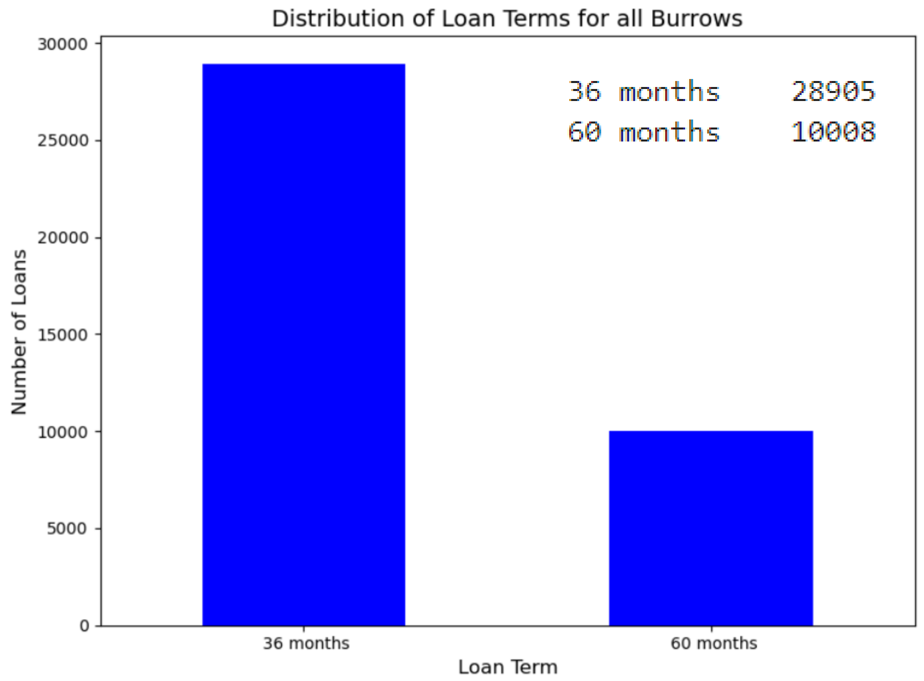
- The typical loan size is around 10,735, with a median of 9,600
- There's a fair amount of variability (standard deviation of 6719.4) and most loans fall within the 5,500 to 15,000 range
- Both the distributions are skewed to left

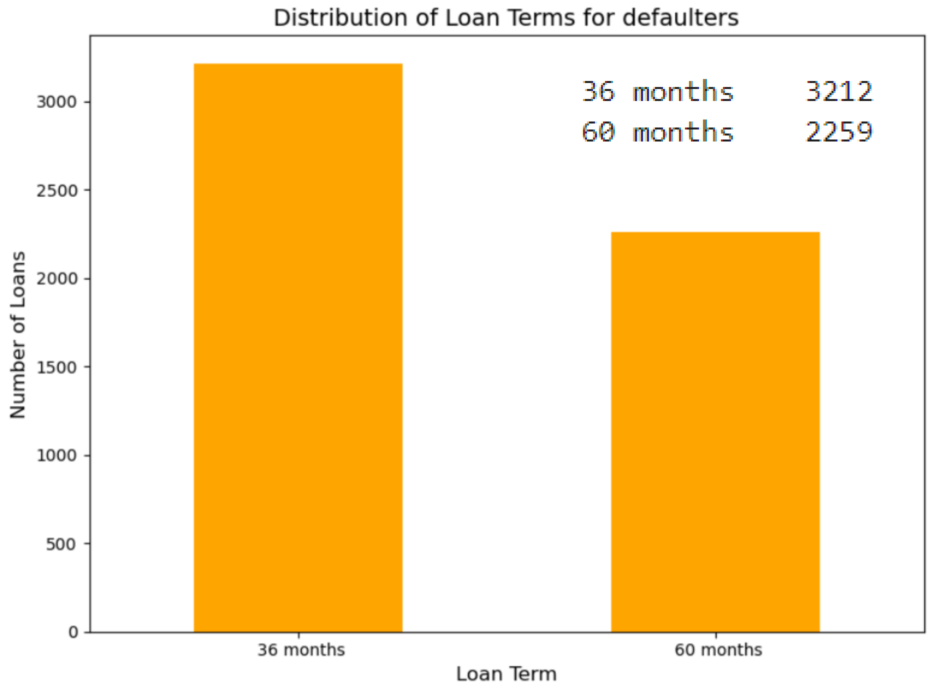
INFERENCE:

The defaulters are majorly in the low loan amount value ranging from 5,000 to 16,000

Exploratory Data Analysis (EDA)

Univariate Analysis on Loan terms





OBSERVATION

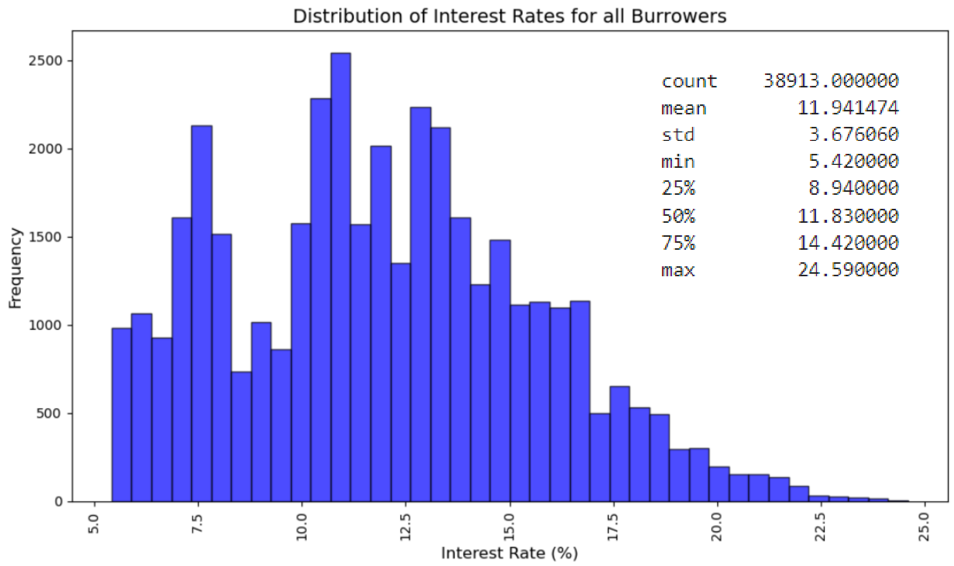
- Over all 14% defaulters from the burrowers
- 11.11 % defaulters in 36 months loan term
- 22.5 % defaulters in 60 months loan term

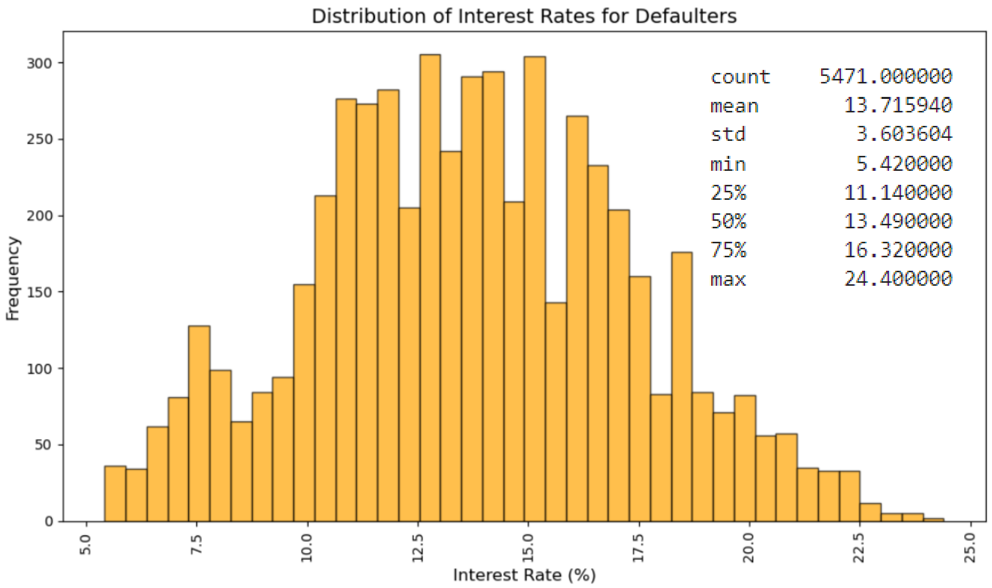
INFERENCE:

Though number of defaulters in short payment term are high but the percentage of defaulters in long term in terms of burrows is higher

Exploratory Data Analysis (EDA)

Univariate Analysis on Interest rates





INFERENCE :

Compared to the total data set which is left skewed , Defaulters show a normal distribution in the frequency of interest rate with maximum defaulters provided with 13.4% interest rate

Exploratory Data Analysis (EDA)

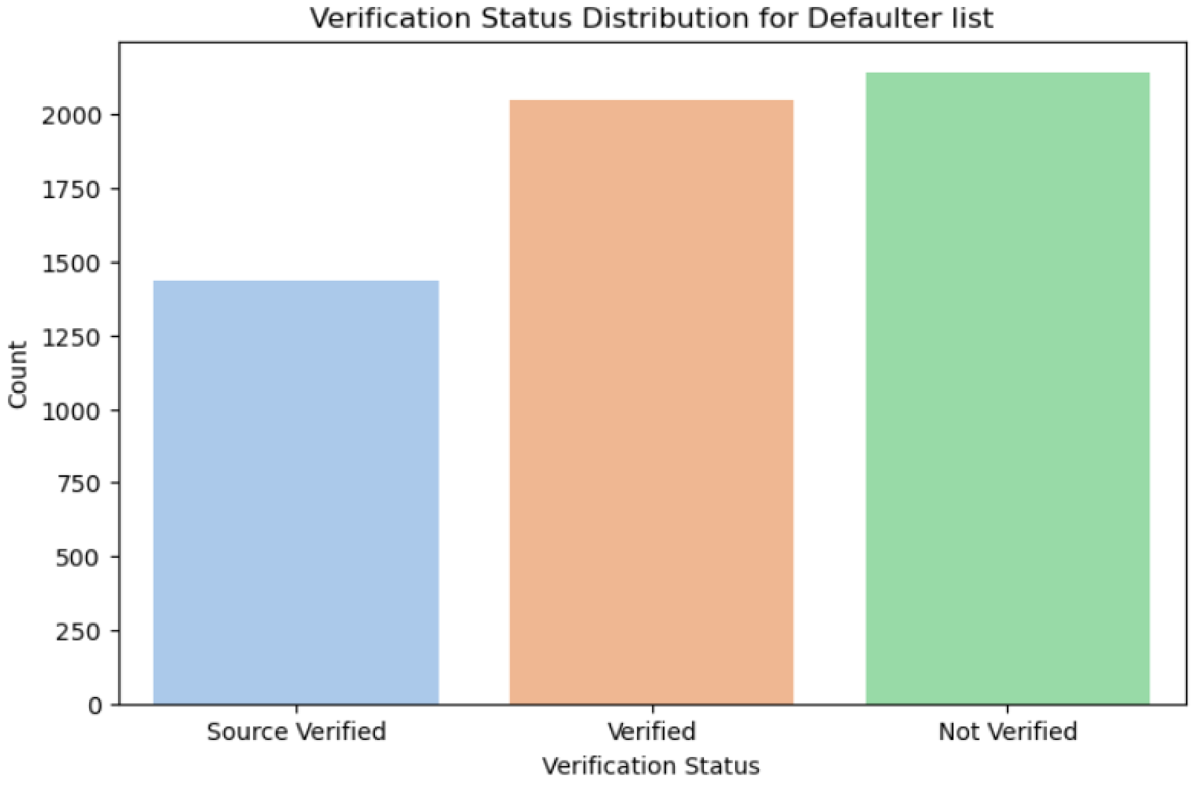
Univariate Analysis on verification status

Interestingly, the defaulter count is high for cases where the source income was verified and LC had verified the burrowers

INFERENCE :

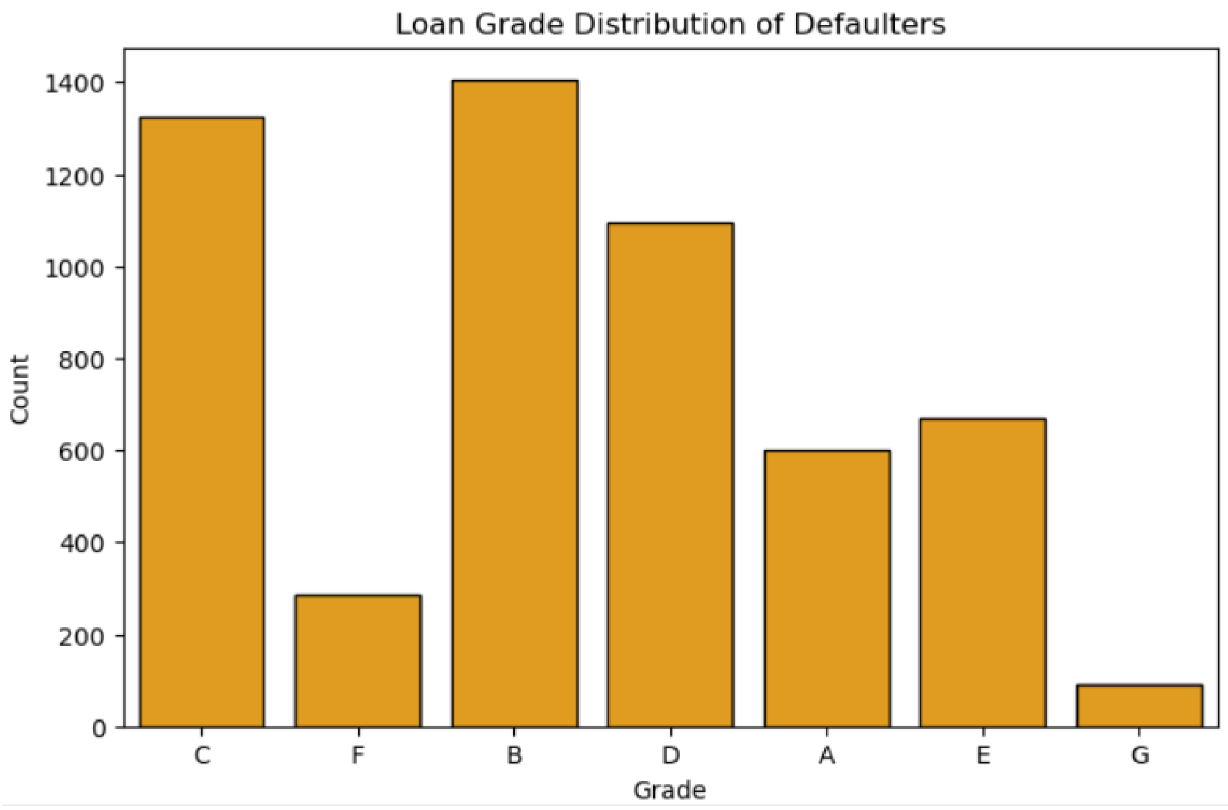
62% of the defaulters were verified and still charged off

The verification process to be looked into



Exploratory Data Analysis (EDA)

Univariate Analysis on Loan grade assigned by LC



OUTPUT

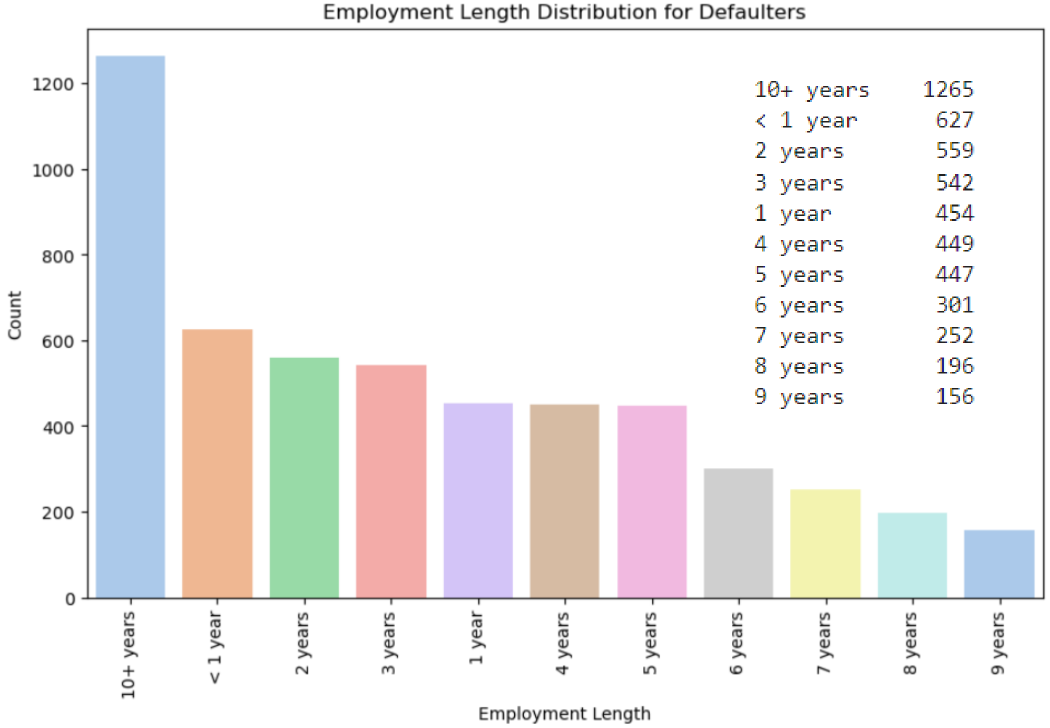
B	1405
C	1325
D	1093
E	671
A	601
F	285
G	91

INFERENCE :
Surprisingly, maximum defaulters are from the low risk graded by Lending club [B,C and D]

The grading system should be revisited

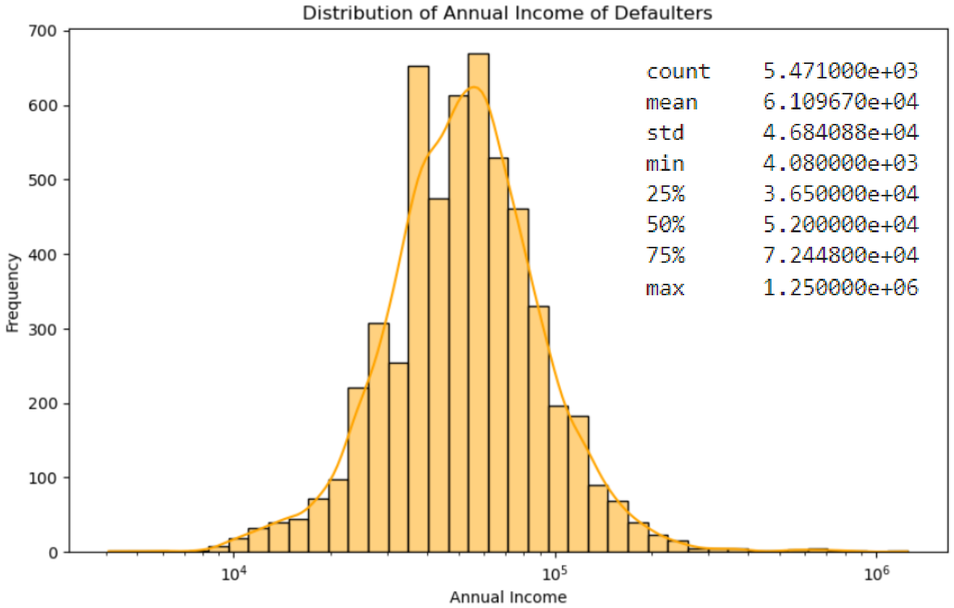
Exploratory Data Analysis (EDA)

Univariate Analysis on Employee Length and Annual income



OBSERVATION :
For employees who have a constant income and are working for more than 10 Years are defaulting more

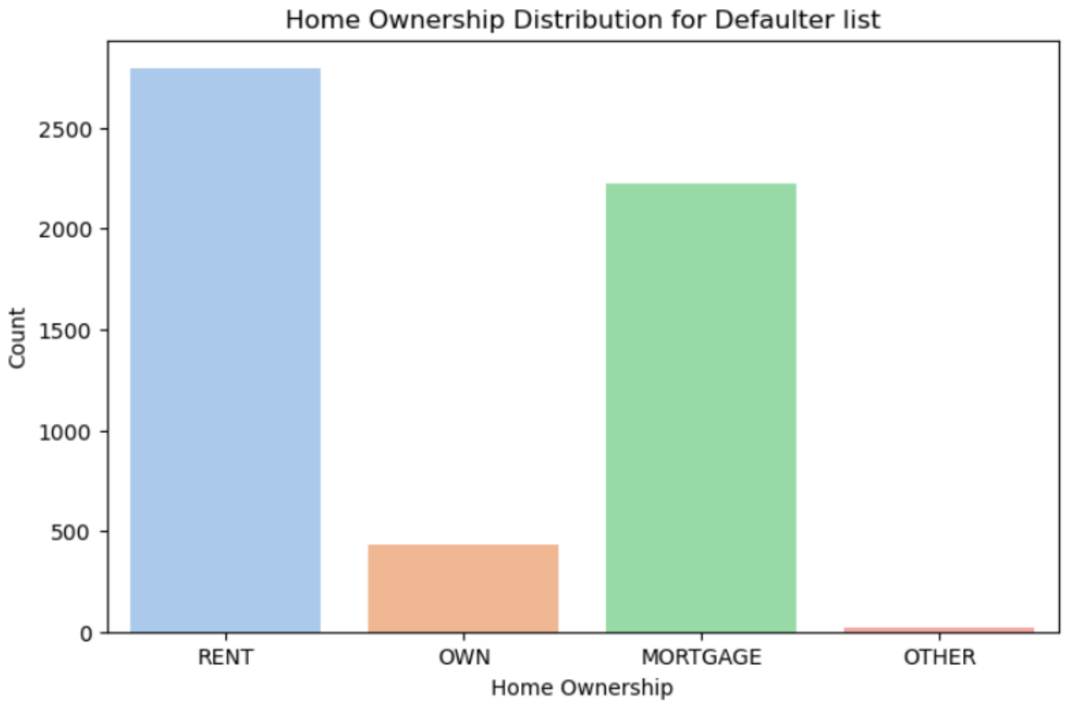
For employees with less than 1 year to 3 years employment are also defaulting more.



OBSERVATION :
A normal distribution of defaulters where the defaulters are higher in the range of 36500 to 73000 income slab

Exploratory Data Analysis (EDA)

Univariate Analysis on House ownership

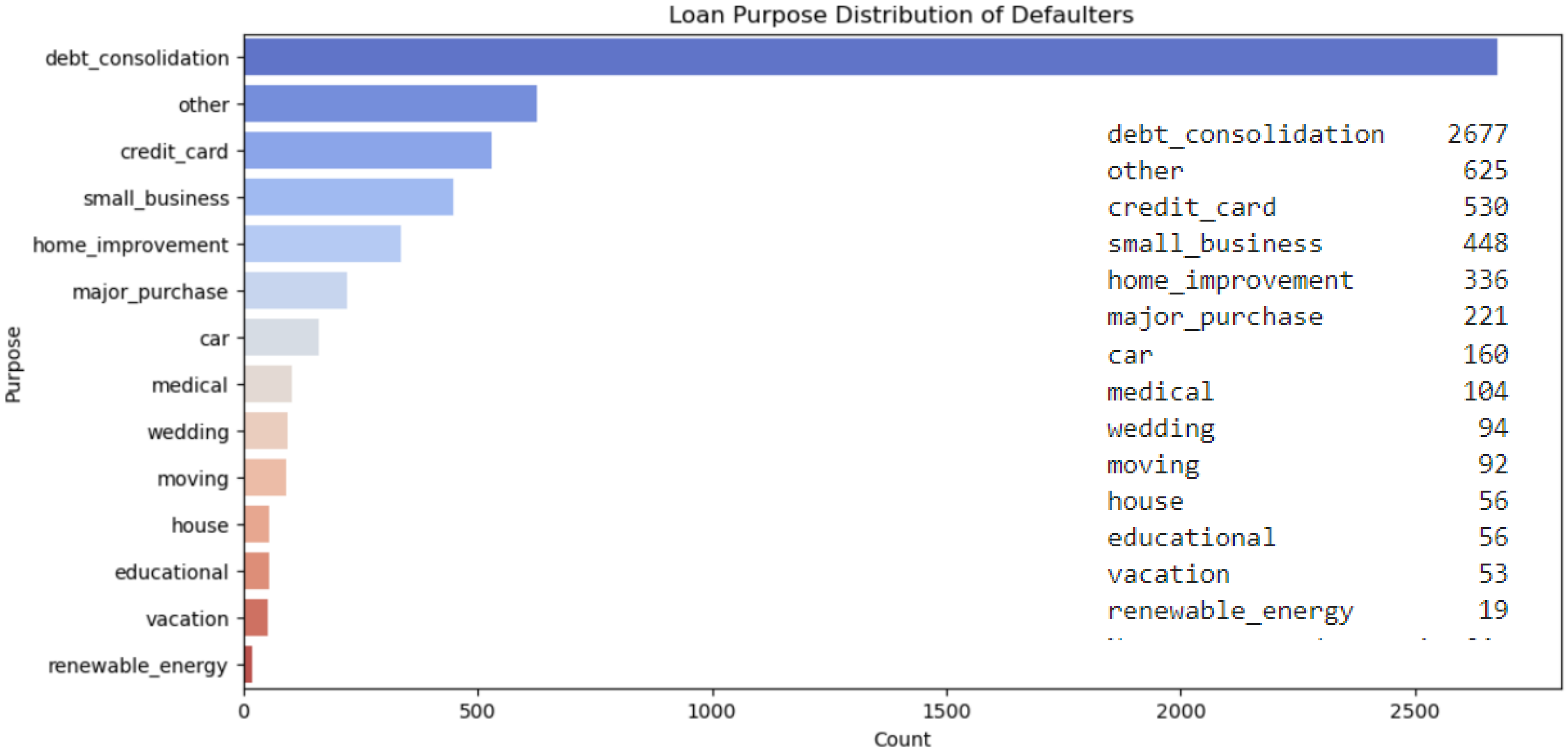


RENT	2795
MORTGAGE	2223
OWN	435
OTHER	18

OBSERVATION :
Burrowers who live in rented house and owners who have mortgaged their house seem to default more

Exploratory Data Analysis (EDA)

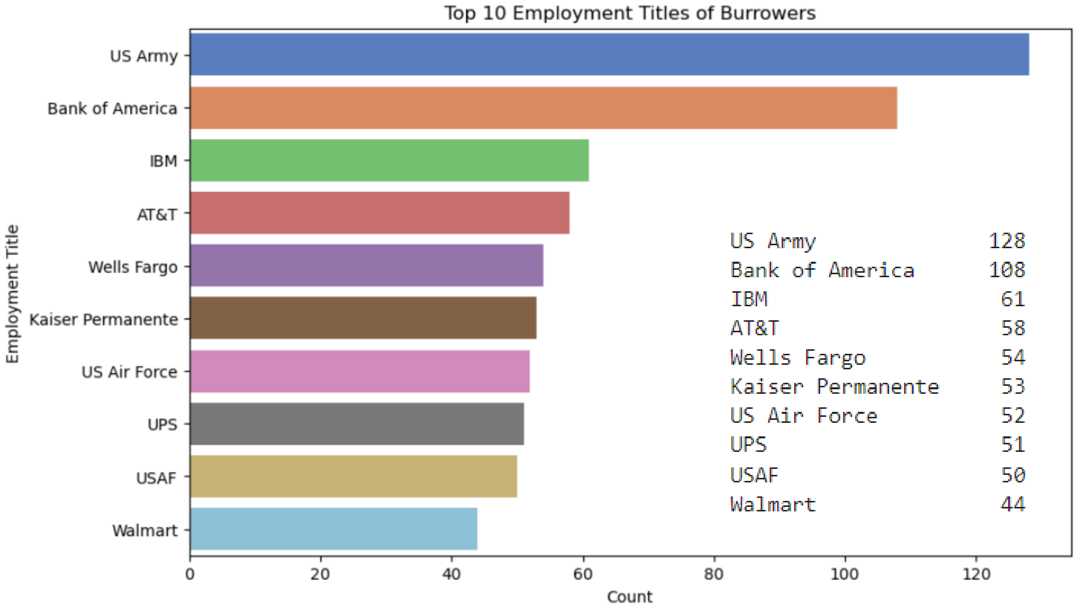
Univariate Analysis on loan purpose

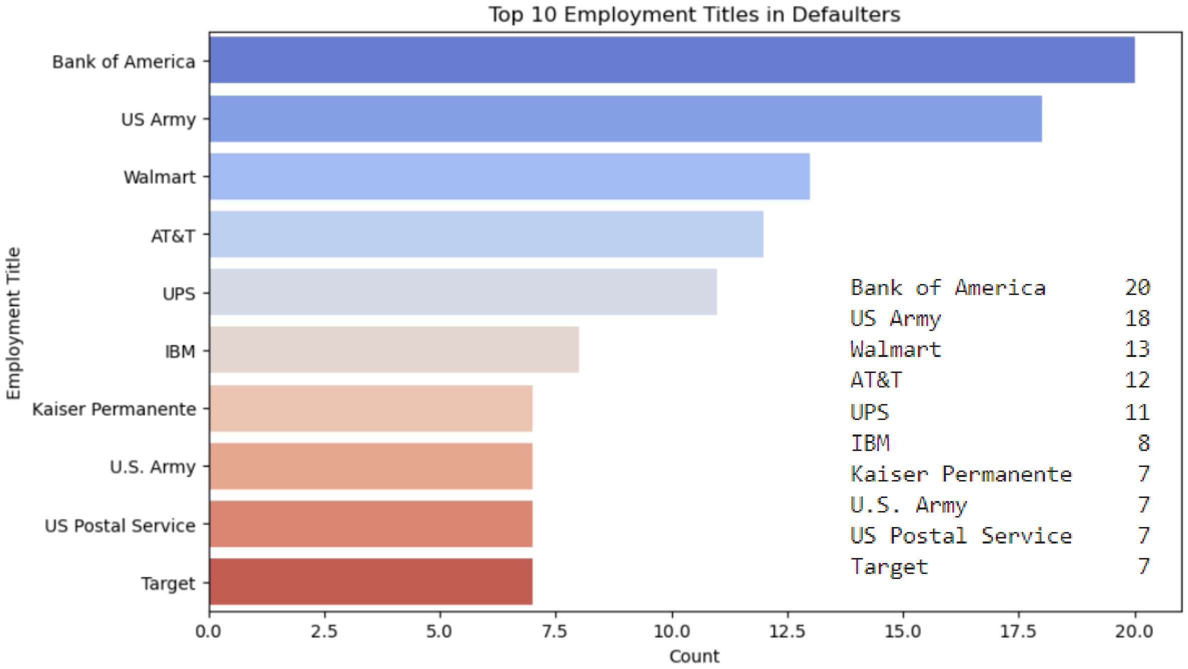


OBSERVATION :
Clearly shows people who are taking the loan for consolidating their debts are defaulting, followed by Others ,credit card ,small businesses and home improvement purpose

Exploratory Data Analysis (EDA)

Univariate Analysis on loan purpose



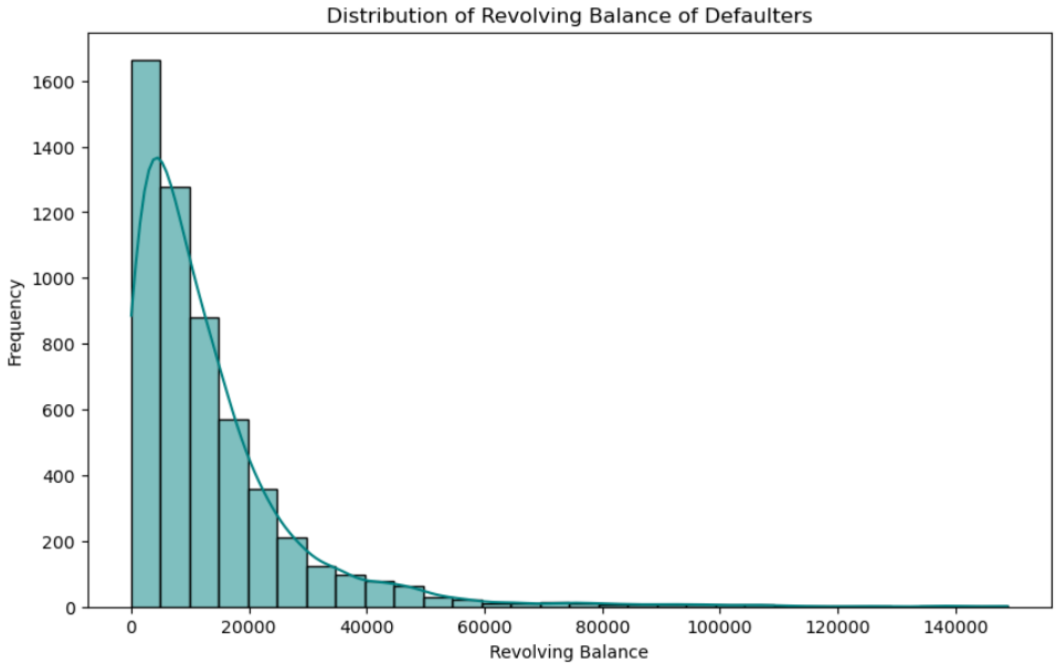


Employment Title	Defaulter %
Walmart	29.5%
AT&T	20.6%
UPS	21.5%
Bank of America	18.5%
US Army	19.5%

OBSERVATION :
Walmart, AT&T and UPS are top employer list with high defaulters

Exploratory Data Analysis (EDA)

Univariate Analysis on Revolving Balance

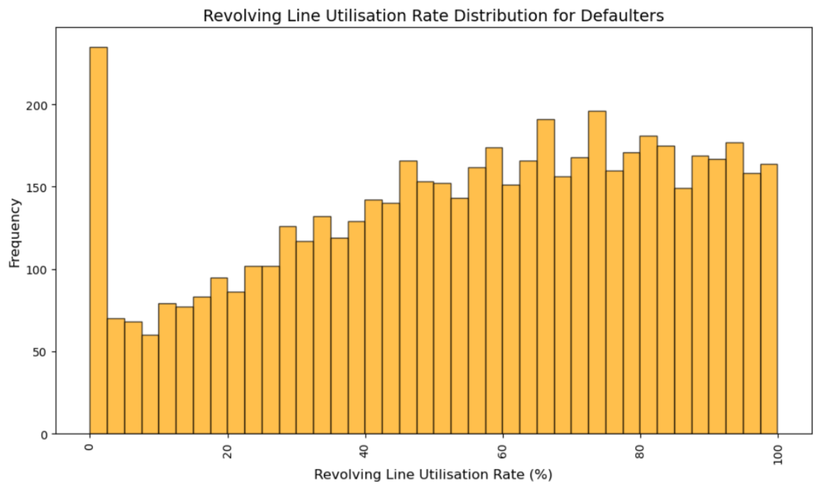


OBSERVATION :
the high default frequency to low revolving balance shows poor financial management or payment difficulties which might be a risk

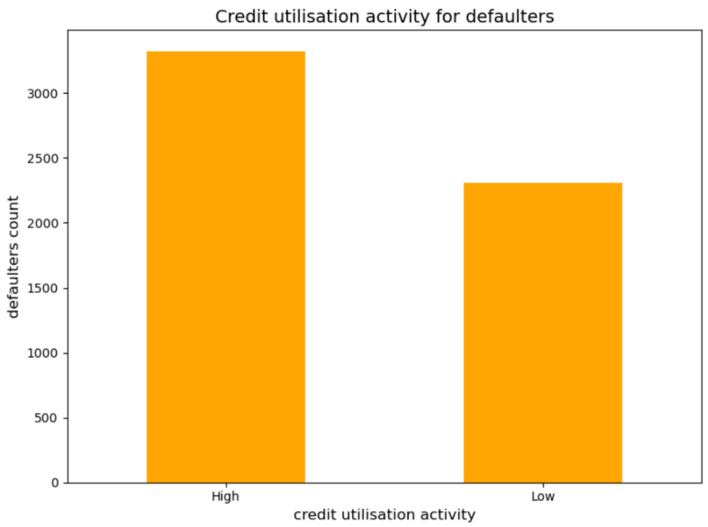
Exploratory Data Analysis (EDA)

Univariate Analysis on Revolving Line Utilisation Rate

To Analyse the Credit Utilisation we derived a categorical column as 'reol_util_catg'




```python
# Add a credit utilisation category into the defaulters data set 'High' if revolving utilisation rate is more than 50%
ln_d['reol_util_catg'] = ln_d['revol_util'].apply(lambda x: 'High' if x > 50 else 'Low')
```

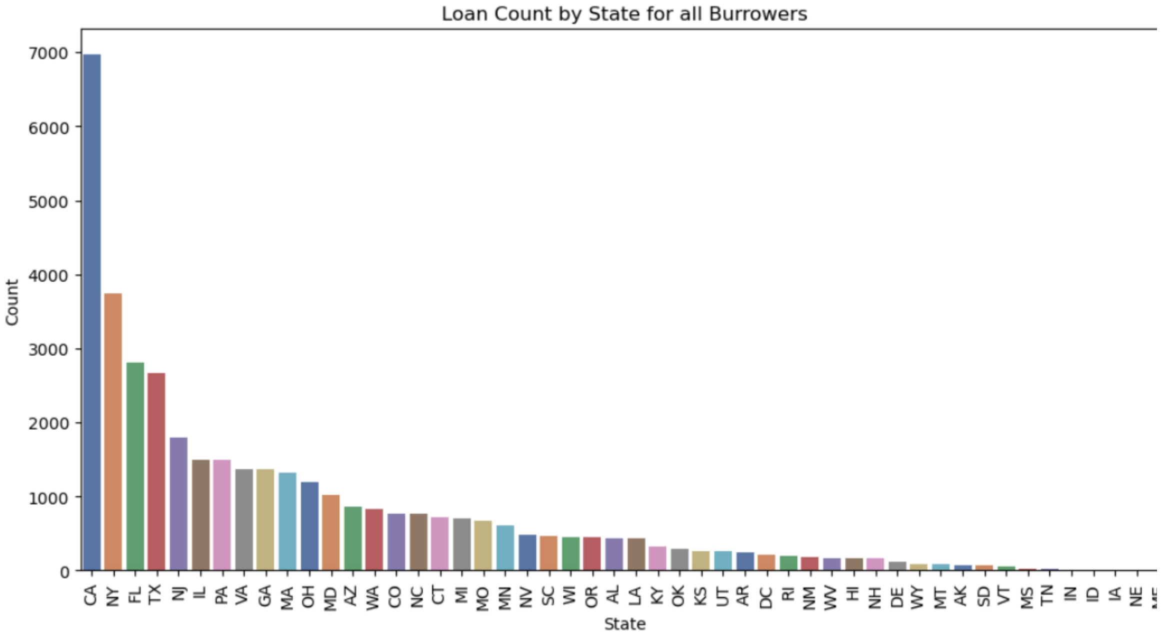


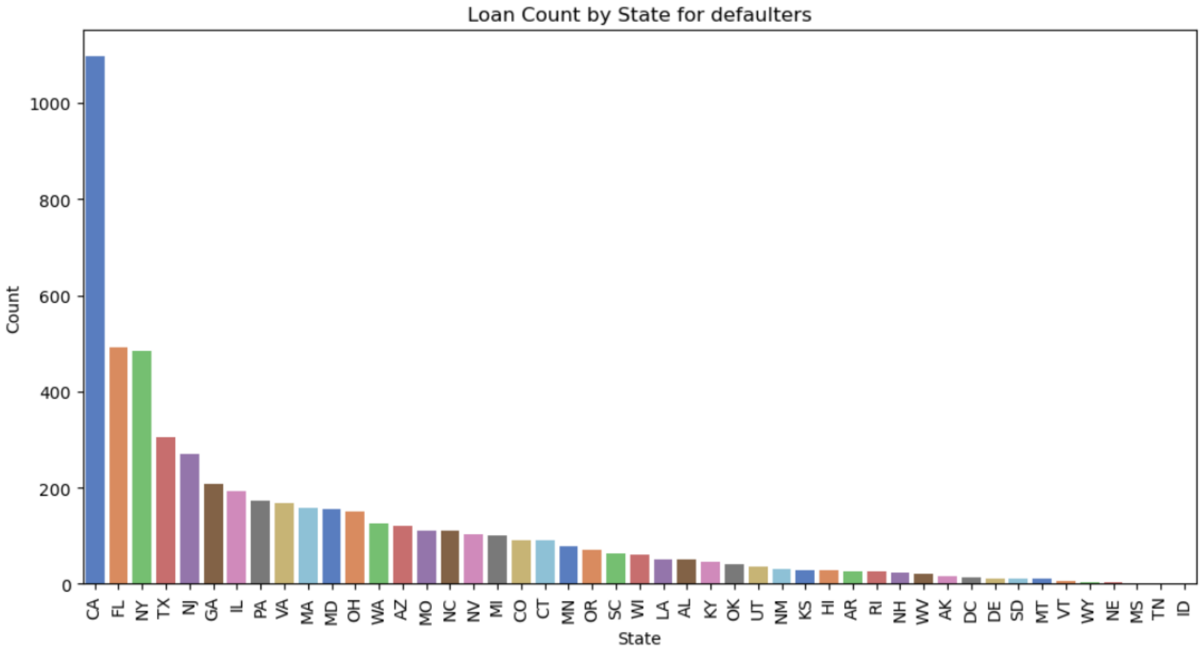
INFERENCE :
Borrowers with high credit utilisation activity are defaulting more, though there is no substantial difference in the count

ACTION :
May be the threshold for credit utilisation can be lowered to keep a check

Exploratory Data Analysis (EDA)

Univariate Analysis on State



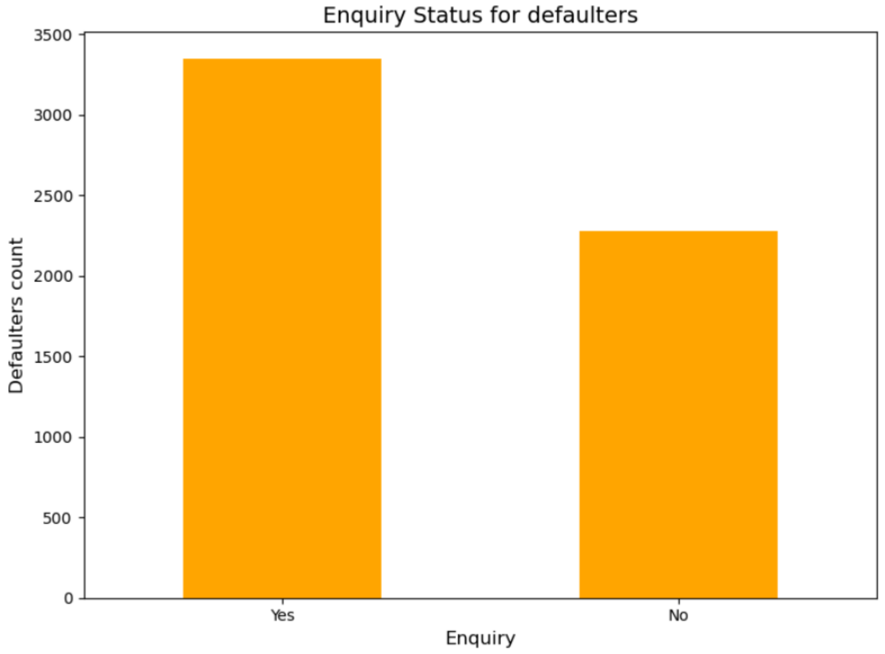


OBSERVATION :
California ,New York and Florida top chart of defaulters from a state view

Exploratory Data Analysis (EDA)

Univariate Analysis on Inquiry status

To Analyse the inq_last_6mths on defaulters we derived a categorical column as ‘Enquiry status’

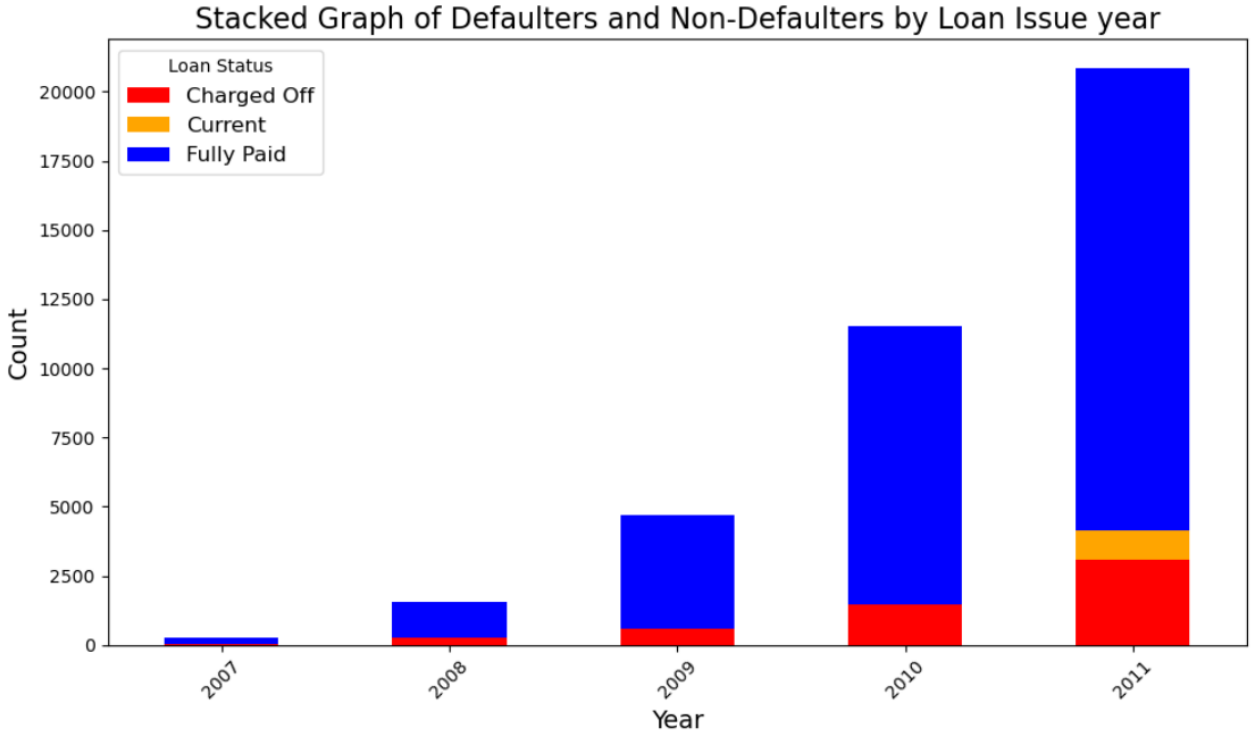



```
#Derive column for Enquiry status
ln_d['Enquiry_status'] = ln_d['inq_last_6mths'].apply(lambda x: 'Yes' if x > 0 else 'No')
```

INFERENCE : Defaulting increases with enquiry. But still cannot derive a direct relation

Exploratory Data Analysis (EDA)

Univariate Analysis on Year of issuance to defaulters count



PIVOT TABLE WITH DERIVED COLUMN FOR DEFAULT PERCENTAGE

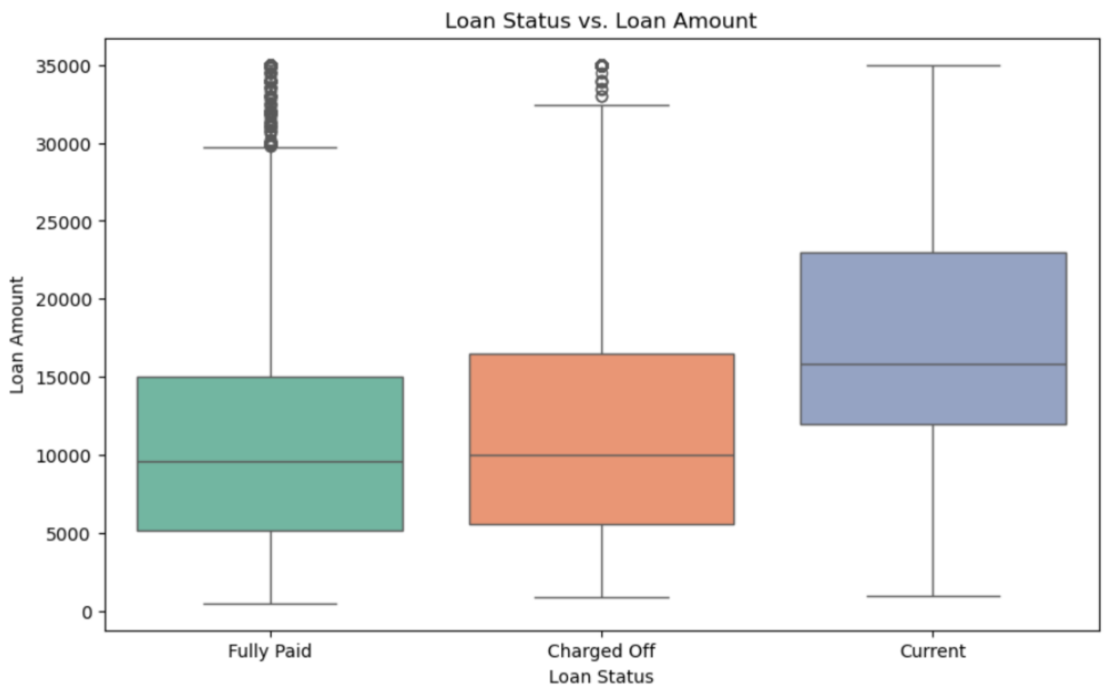
loan_status	Charged Off	Current	Fully Paid	Defaulters_perc
loan_issue_Year				
2007	45	0	206	17.928287
2008	247	0	1315	15.813060
2009	594	0	4122	12.595420
2010	1485	0	10047	12.877211
2011	3084	1051	16717	14.789948

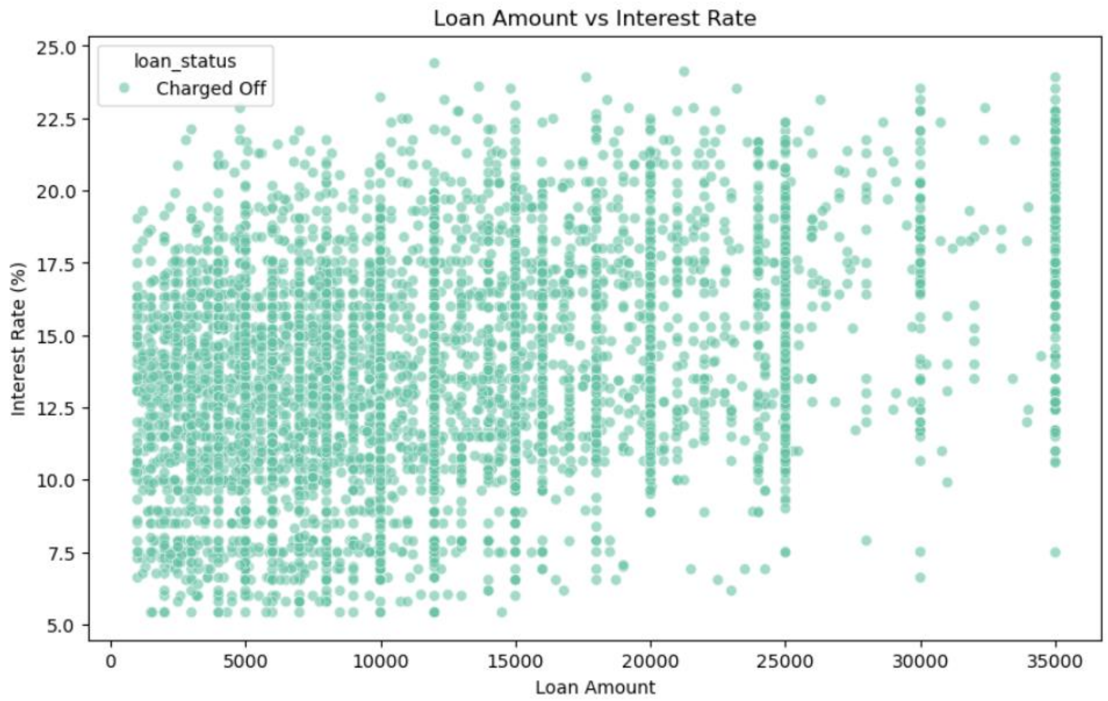
OBSERVATION :

- *The default percentage decreased from 2007 till 2010 but there is slight increase in the 2011*
- *There also a sudden surge in number of burrowers in 2011 compared to last 4 years*

Exploratory Data Analysis (EDA)

Bivariant Analysis on loan amount vs interest rate and loan amount vs loan status



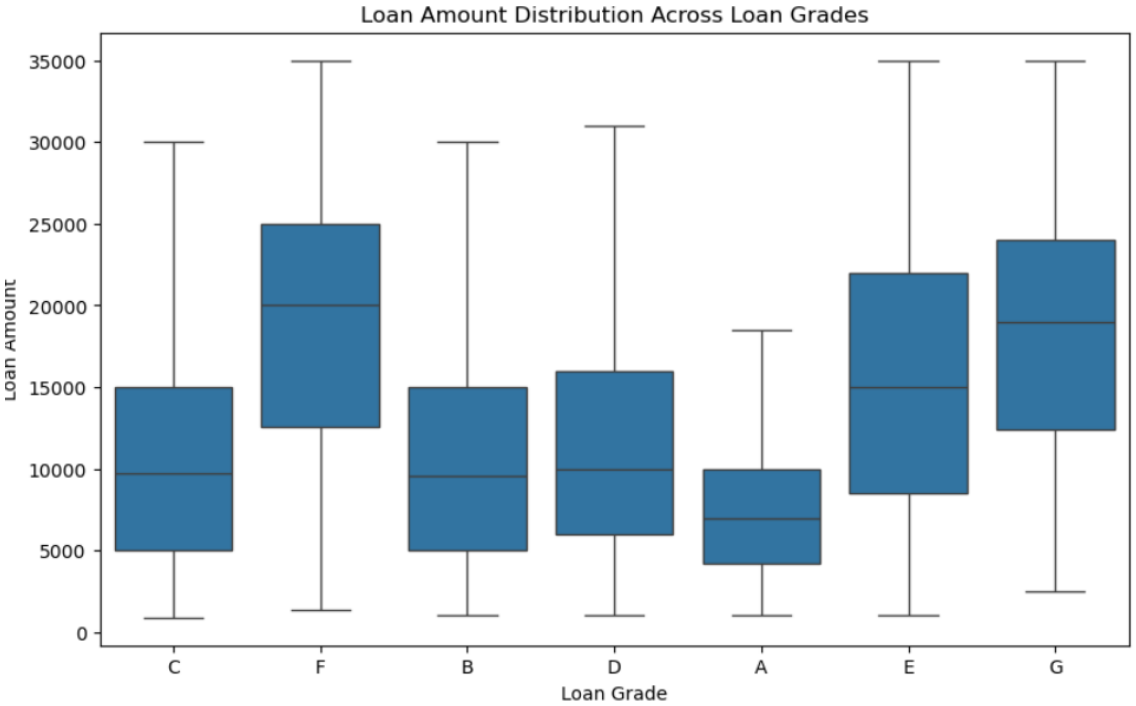


OBSERVATION :

- *The loan amount defaults are in between 5,000 – 15,000*
- *Defaulters are concentrated between interest rate with in 10%- 18% with low loan amount upto 10,000*

Exploratory Data Analysis (EDA)

Bivariant Analysis on loan amount vs loan grade



OBSERVATION :

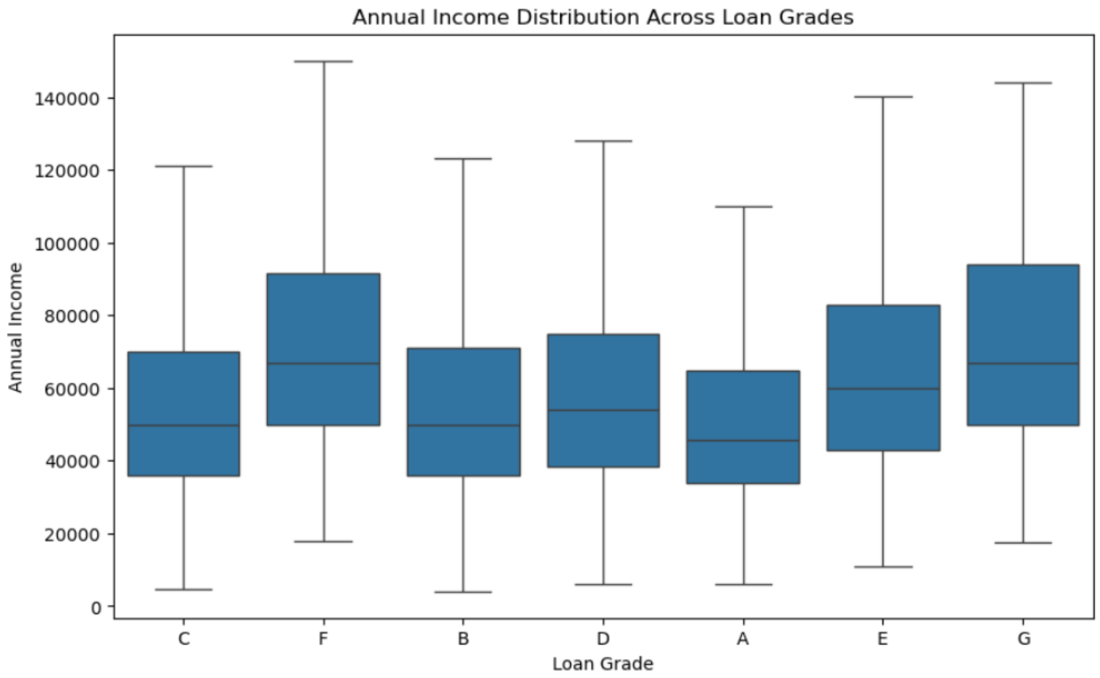
- *Loan grades like F and G have the highest median loan amounts, indicating that riskier loans (as F and G are lower grades) tend to be larger in size*
- *Grades A and B have lower median loan amounts, showing that borrowers with good credit scores tend to take smaller loans*
- *Variance in loan amounts increases as the grades go from A to G, indicating more inconsistency in loan sizes for higher-risk grades*

Recommendations:

1. For high-risk grades like F and G, stricter lending conditions should be applied, such as:
    1. Higher interest rates.
    2. Lower loan caps to mitigate potential risks of default.
2. For lower-risk grades like A, flexible loan policies could be extended, as these borrowers are less likely to default

Exploratory Data Analysis (EDA)

Bivariant Analysis on Annual Income vs loan grade



OBSERVATION :

- *Grades F and G have the highest median annual incomes, shows borrowers with riskier loans often have higher incomes*

- *Grade A has the lowest median annual income, shows lower-risk borrowers are typically in moderate-income brackets*

Exploratory Data Analysis (EDA)

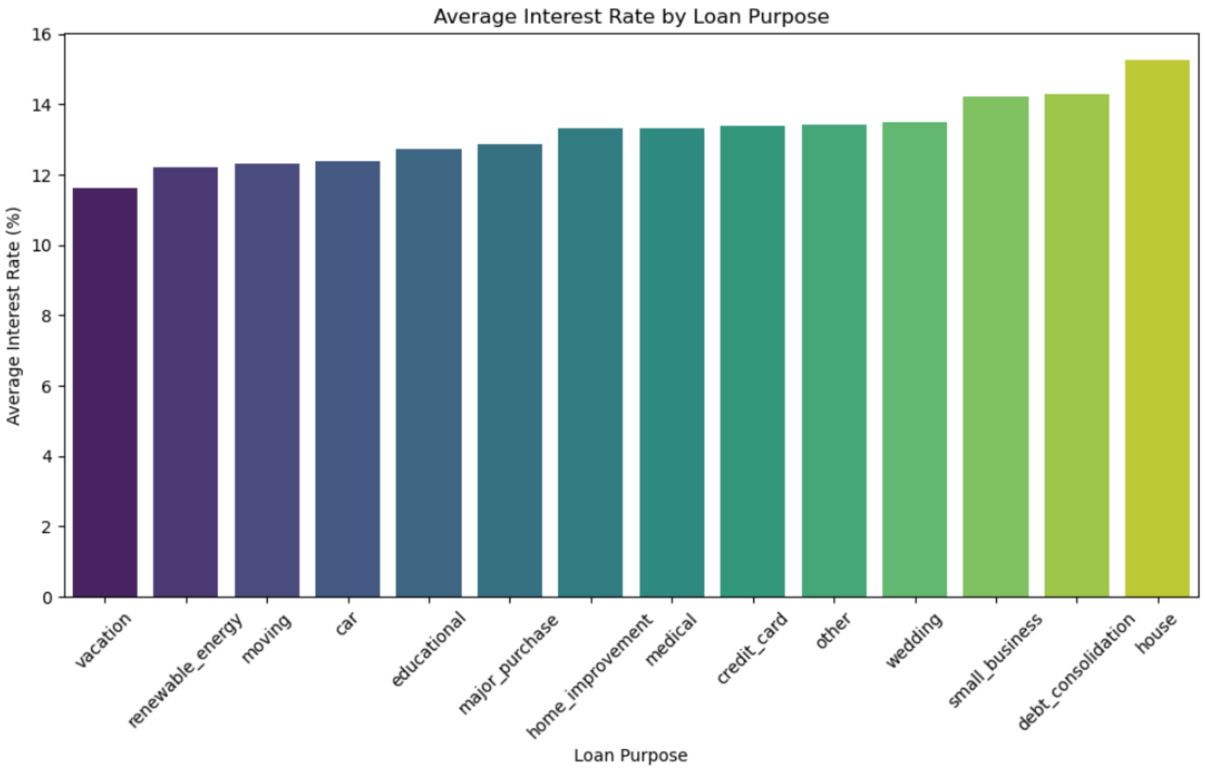
Bivariant Analysis on Annual Income vs loan amount

OBSERVATION :

- *Lower loan amount with lower income burrowers tend to default more*

Exploratory Data Analysis (EDA)

Bivariant Analysis on Avg interest rate vs Loan purpose

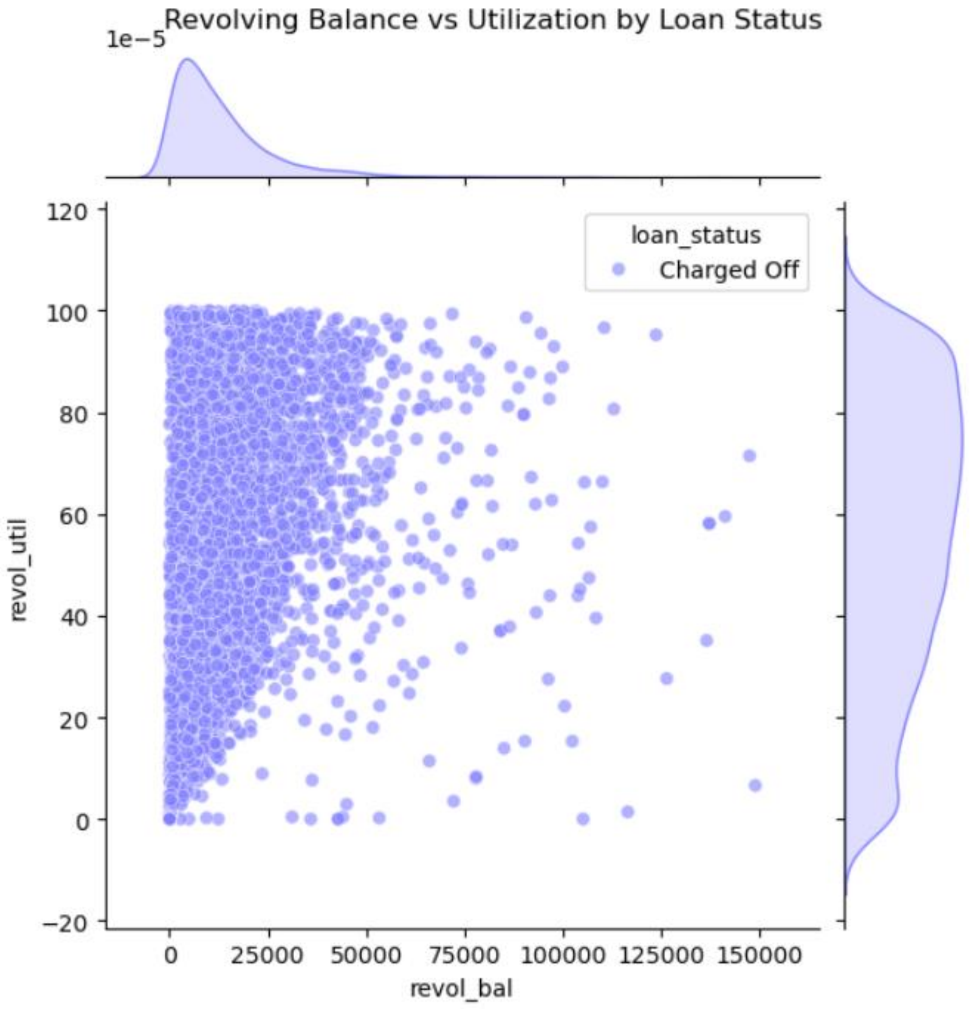


OBSERVATION :

- *Considering the univariate analysis on loan purpose, debt consolidation defaulters were highest. I might be because of the high interest rate*

Exploratory Data Analysis (EDA)

Bivariant Analysis on Revolving Balance vs Utilization by Loan Status

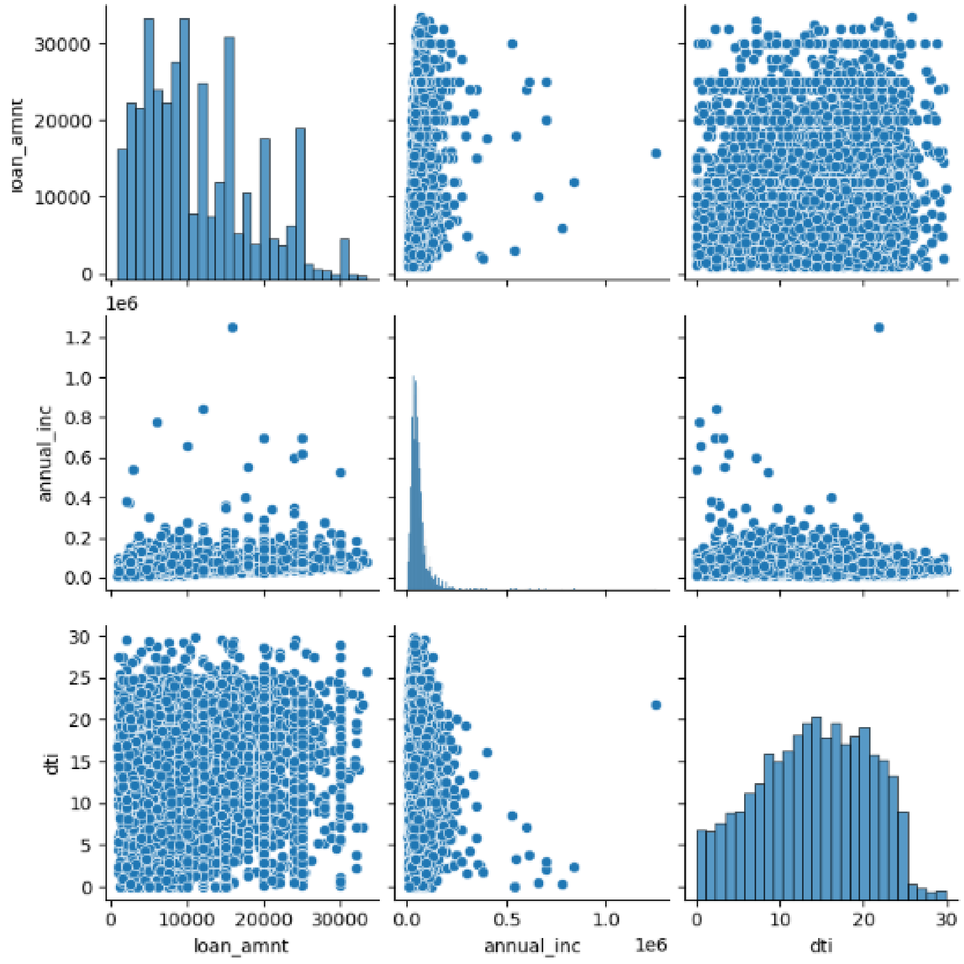


OBSERVATION :

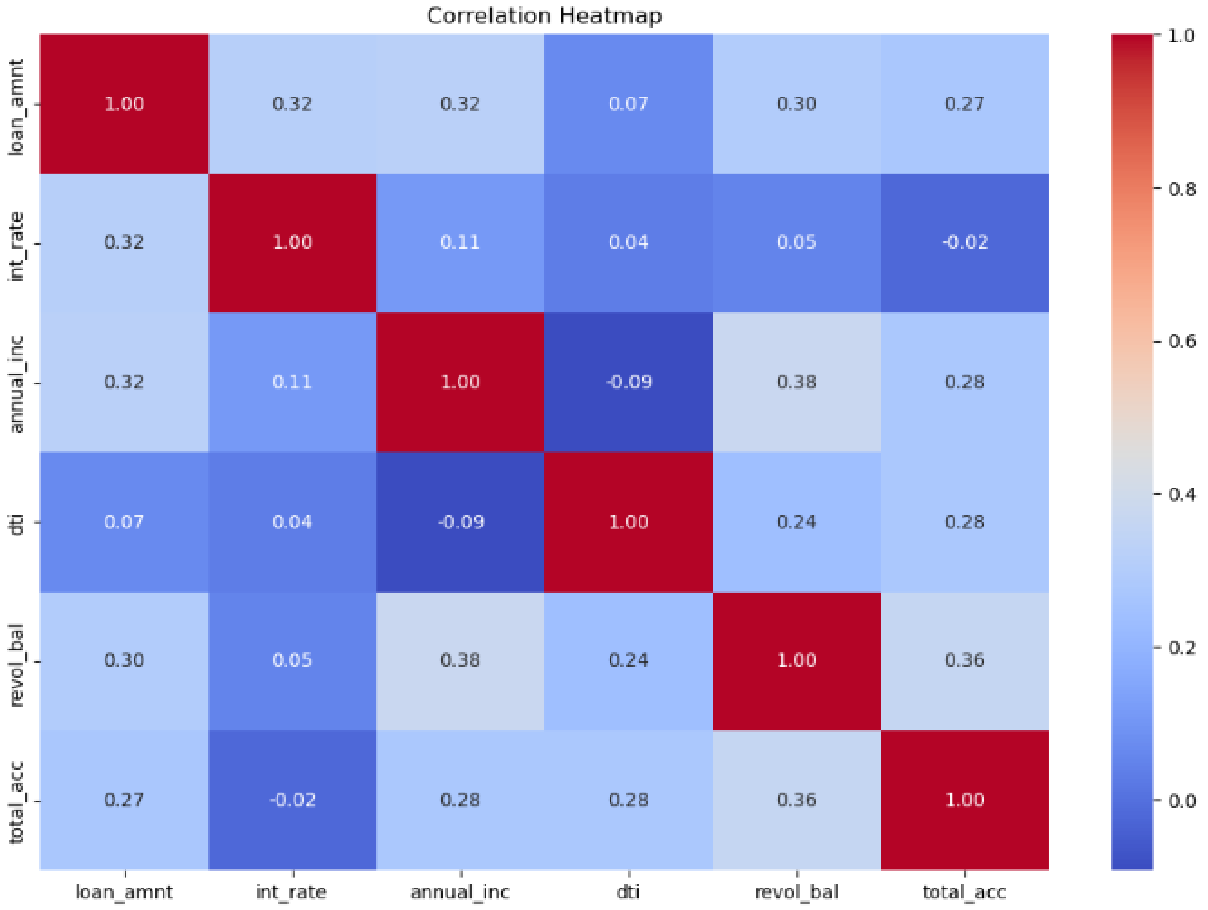
- *higher revolving utilization >30% with low revolving balance is risky*

Exploratory Data Analysis (EDA)

PAIR PLOT ON loan amount, Annual Income and DTI



Correlation Heat Map



Recommendations

- **Loan Amount**: Risk increases with smaller loan amounts
- **Loan Term**: Long-term loans are riskier; shorter terms are preferable
- **Interest Rate**: 10%-18% range requires stricter monitoring
- **Annual Income**: Borrowers in $36,500–$73,000 should be scrutinized more
- **Employment Details**: Evaluate both tenure and stability; 1–3 years and 10+ years are risky
- **Housing Status**: Renters and those with mortgages default more
- **Loan Purpose**: Debt consolidation, small businesses, and credit card loans need rigorous checks
- **Verification Status**: Verification should go beyond income validation
- **Credit Utilization**: High utilization (>30%) combined with low revolving balances is risky
- **Revolving Balance**: Low balances suggest financial instability
- **Grade** : high-risk grades like F and G should have stricter lending conditions should be applied