Lending Club Case Study

Samarjeet Saurabh & Santhosh Talluri

Objective

- Objective: The aim of this case study is to apply Exploratory Data Analysis (EDA) techniques to a real-world problem, uncover meaningful insights, and present them in a business-focused manner through a presentation.
- Benefits of the Case Study:
  - Provides an understanding of how EDA is utilized in addressing real-world business challenges.
  - Develops a foundational knowledge of risk analytics within the banking and financial services sectors.
  - Demonstrates how data is leveraged to minimize financial losses when lending to clients.
  - Enhances comprehension of data visualization and the appropriate use of charts for real world data analysis.

Problem Statement

- Find out the driving factors of loan default from given loan data to minimize financial loss and improve lending business.

Approach

- Data Understanding : Load and read the data
- Data clean up and preparation process: Delete null columns and duplicate data, fixing null values, correcting data types and removing outliers.
- Draw Insights: conduct univariate analysis, bivariate analysis and summarize.

Understanding Data

Loading the DATA ¶

[3]:
```
loan_data = pd.read_csv('loan.csv')
```

[5]:
```
# Printing the data(first 5 rows)
loan_data.head()
```

[5]:

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment	grade	sub_grade	...	num_tl_90g_dpd_24m	num_tl_op_past_12m	pc
0	1077501	1296599	5000	5000	4975.0	36 months	10.65%	162.87	B	B2	...	NaN	NaN	
1	1077430	1314167	2500	2500	2500.0	60 months	15.27%	59.83	C	C4	...	NaN	NaN	
2	1077175	1313524	2400	2400	2400.0	36 months	15.96%	84.33	C	C5	...	NaN	NaN	
3	1076863	1277178	10000	10000	10000.0	36 months	13.49%	339.31	C	C1	...	NaN	NaN	
4	1075358	1311748	3000	3000	3000.0	60 months	12.69%	67.79	B	B5	...	NaN	NaN	

5 rows × 111 columns

- Displaying first 5 header rows for quick understanding

Understanding Data

```
[7]: # Basic infomation about the dataframe
     print(loan_data.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39717 entries, 0 to 39716
Columns: 111 entries, id to total_il_high_credit_limit
dtypes: float64(74), int64(13), object(24)
memory usage: 33.6+ MB
None
```

```
[9]: # Data types of each column
     print(loan_data.dtypes)

id                               int64
member_id                        int64
loan_amnt                        int64
funded_amnt                      int64
funded_amnt_inv                float64
                                ...
tax_liens                      float64
tot_hi_cred_lim                float64
total_bal_ex_mort              float64
total_bc_limit                 float64
total_il_high_credit_limit     float64
Length: 111, dtype: object
```

```
# Describing the dataframe
print(loan_data.describe())

                 id     member_id     loan_amnt   funded_amnt  \
count  3.971700e+04  3.971700e+04  39717.000000  39717.000000
mean   6.831319e+05  8.504636e+05  11219.443815  10947.713196
std    2.106941e+05  2.656783e+05   7456.670694   7187.238670
min    5.473400e+04  7.069900e+04    500.000000    500.000000
25%    5.162210e+05  6.667800e+05   5500.000000   5400.000000
50%    6.656650e+05  8.508120e+05  10000.000000   9600.000000
75%    8.377550e+05  1.047339e+06  15000.000000  15000.000000
max    1.077501e+06  1.314167e+06  35000.000000  35000.000000

       funded_amnt_inv   installment    annual_inc           dti  \
count     39717.000000  39717.000000  3.971700e+04  39717.000000
mean      10397.448868    324.561922  6.896893e+04     13.315130
std        7128.450439    208.874874  6.379377e+04      6.678594
min           0.000000     15.690000  4.000000e+03      0.000000
25%        5000.000000    167.020000  4.040400e+04      8.170000
50%        8975.000000    280.220000  5.900000e+04     13.400000
75%       14400.000000    430.780000  8.230000e+04     18.600000
max       35000.000000   1305.190000  6.000000e+06     29.990000
```

Data Info

- 39717 entries
- 111 columns.
- Float, Int and object data types
- Describing Stats on all columns

Understanding Data

```
# Columns in the dataframe
print(loan_data.columns)
```

```
Index(['id', 'member_id', 'loan_amnt', 'funded_amnt', 'funded_amnt_inv
       'term', 'int_rate', 'installment', 'grade', 'sub_grade',
       ...
       'num_tl_90g_dpd_24m', 'num_tl_op_past_12m', 'pct_tl_nvr_dlq',
       'percent_bc_gt_75', 'pub_rec_bankruptcies', 'tax_liens',
       'tot_hi_cred_lim', 'total_bal_ex_mort', 'total_bc_limit'
       'total_il_high_credit_limit'],
      dtype='object', length=111)
```

Basic information about the data

```
## Number of rows and columns
print('Number of Columns:',loan_data.shape[1])
print('Number of Rows:',loan_data.shape[0])

## Number of missing values
print('Number of missing values:',loan_data.isnull().sum().sum())

## Number of unique values
print('Number of unique values:',loan_data.nunique().sum())

## Number of duplicates
print('Number of duplicates:',loan_data.duplicated().sum())
```

```
Number of Columns: 111
Number of Rows: 39717
Number of missing values: 2263366
Number of unique values: 416800
Number of duplicates: 0
```

Observations:

- No duplicate values
- Null value columns are present

Analysing data set for cleaning

```
# checking null values in data set
print(loan_data.isnull().sum())
```

```
id                              0
member_id                       0
loan_amnt                       0
funded_amnt                     0
funded_amnt_inv                 0
                              ...
tax_liens                      39
tot_hi_cred_lim             39717
total_bal_ex_mort           39717
total_bc_limit              39717
total_il_high_credit_limit  39717
Length: 111, dtype: int64
```

```
# Checking null values percetange in descending order in given data set.
print((loan_data.isnull().sum()/loan_data.shape[0]*100).round(2).sort_values(ascending=False))
```

```
verification_status_joint    100.0
annual_inc_joint             100.0
mo_sin_old_rev_tl_op         100.0
mo_sin_old_il_acct           100.0
bc_util                      100.0
                              ...
delinq_amnt                    0.0
policy_code                    0.0
earliest_cr_line               0.0
delinq_2yrs                    0.0
id                             0.0
Length: 111, dtype: float64
```

Data Clean UP

```
# Removed all columns whose null values percentage is above 50%, as these columns will not impact on analysis
loan_data = loan_data.loc[:,loan_data.isnull().sum()/loan_data.shape[0]*100 <50]
# Shape of the dataframe after removing columns
print(loan_data.shape)
```

```
(39717, 54)
```

```
# Checking columns again for null value percentage
print((loan_data.isnull().sum()/loan_data.shape[0]*100).round(2).sort_values(ascending=False))
```

```
desc                          32.59
emp_title                      6.19
emp_length                     2.71
pub_rec_bankruptcies           1.75
last_pymnt_d                   0.18
collections_12_mths_ex_med     0.14
chargeoff_within_12_mths       0.14
revol_util                     0.13
tax_liens                      0.10
title                          0.03
last_credit_pull_d             0.01
total_rec_prncp                0.00
out_prncp                      0.00
```

Clean Up Activities :

- Deleted Null columns which have nulls above 50%
- Deleted irrelevant columns.
- Deleted unique value columns.
- After clean up data set has 25 columns.
- Listed final columns.

Removing the irrelevant columns

```
# Removing irrelevant columns which are calculated after loan is approved thus have no relevance to the analysis
loan_data=loan_data.drop(['revol_bal','out_prncp','out_prncp_inv','total_pymnt','total_pymnt_inv','total_rec_prncp','total_rec_int','total_rec_late_fee',
```

```
# Checking columns for irrelevant data which has no impact to analysis(having very few unqiue values)
print(loan_data.nunique().sort_values(ascending=True))
```

```
tax_liens                     1
delinq_amnt                   1
chargeoff_within_12_mths      1
acc_now_delinq                1
initial_list_status           1
collections_12_mths_ex_med    1
pymnt_plan                    1
application_type              1
policy_code                   1
term                          2
pub_rec_bankruptcies          3
loan_status                   3
verification_status           3
home_ownership                5
```

```
# Removing irrelevant columns which contain 1 unique value
loan_data = loan_data.loc[:,loan_data.nunique()>1]

# Shape of the dataframe after removing columns
print(loan_data.shape)
```

```
(39717, 25)
```

```
# Columns in the dataframe
print(loan_data.columns)
```

```
Index(['id', 'loan_amnt', 'term', 'int_rate', 'installment', 'grade',
       'sub_grade', 'emp_length', 'home_ownership', 'annual_inc',
       'verification_status', 'issue_d', 'loan_status', 'purpose', 'title',
       'addr_state', 'dti', 'delinq_2yrs', 'earliest_cr_line',
       'inq_last_6mths', 'open_acc', 'pub_rec', 'revol_util', 'total_acc',
       'pub_rec_bankruptcies'],
```

Data Clean Up

```
# Checking for missing values across the dataframe
print(loan_data.isnull().sum().sort_values(ascending=False))
```

```
emp_length              1075
pub_rec_bankruptcies     697
revol_util                50
title                     11
```

```
##  Fill null with Unknown to emp_length
loan_data["emp_length"].fillna("Unknown", inplace=True)
```

```
# Check emp_length count
loan_data.emp_length.value_counts()
```

```
emp_length
10+ years     8879
< 1 year      4583
2 years       4388
3 years       4095
4 years       3436
5 years       3282
1 year        3240
6 years       2229
7 years       1773
8 years       1479
9 years       1258
Unknown       1075
Name: count, dtype: int64
```

```
##  Fill null with Unknown to pub_rec_bankruptcies
loan_data["pub_rec_bankruptcies"].fillna("Unknown", inplace=True)
loan_data.pub_rec_bankruptcies.value_counts()
```

```
pub_rec_bankruptcies
0.0         37339
1.0          1674
Unknown       697
2.0             7
Name: count, dtype: int64
```

```
# Checking "revol_util" after removing null values, so we can handle missing values in original data
loan_data.revol_util=loan_data.revol_util.apply(lambda x:str(x).replace('%','')).astype('float').round(2)

print(loan_data['revol_util'].describe())
print(loan_data['revol_util'].median())
```

```
count    39667.000000
mean        48.832152
std         28.332634
min          0.000000
25%         25.400000
50%         49.300000
75%         72.400000
max         99.900000
Name: revol_util, dtype: float64
49.3
```

```
# Variation between mean and median is very close to each, so filling null values with the mean value.
loan_data['revol_util'].fillna("48.83%")
```

```
0       83.7
1        9.4
2       98.5
3       21.0
4       53.9
```

Missing values Treatment

- Emp_length filled with Unknown.
- pub_rec_bankruptcies filled with Unknown
- revol_util filled with mean

Data Clean Up

```
### converting data type to few columns.
loan_data.int_rate=loan_data.int_rate.apply(lambda x:str(x).replace('%','')).astype('float').round(2)
loan_data.revol_util=loan_data.revol_util.apply(lambda x:str(x).replace('%','')).astype('float').round(2)
loan_data['annual_inc'] = loan_data['annual_inc'].apply(lambda x: f"{x:.0f}").astype(int)
loan_data.term=loan_data.term.apply(lambda x: int(x.replace(' months',''))).astype(int)
```

```
loan_data.head(5)
```

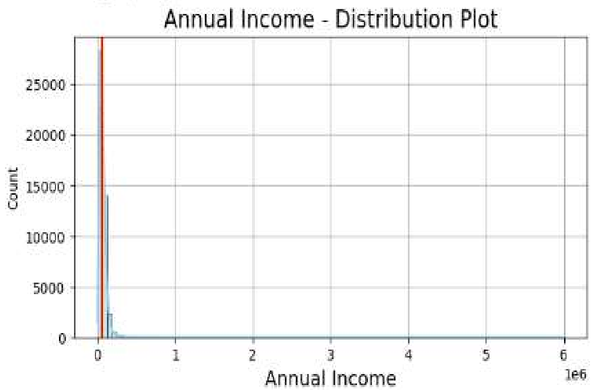
	id	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_length	home_ownership	annual_inc	...	addr_state	dti	delinq_2yrs	earliest_cr_line	inq
0	1077501	5000	36	10.65	162.87	B	B2	10+ years	RENT	24000	...	AZ	27.65	0	Jan-85	
1	1077430	2500	60	15.27	59.83	C	C4	< 1 year	RENT	30000	...	GA	1.00	0	Apr-99	
2	1077175	2400	36	15.96	84.33	C	C5	10+ years	RENT	12252	...	IL	8.72	0	Nov-01	
3	1076863	10000	36	13.49	339.31	C	C1	10+ years	RENT	49200	...	CA	20.00	0	Feb-96	
4	1075358	3000	60	12.69	67.79	B	B5	1 year	RENT	80000	...	OR	17.94	0	Jan-96	

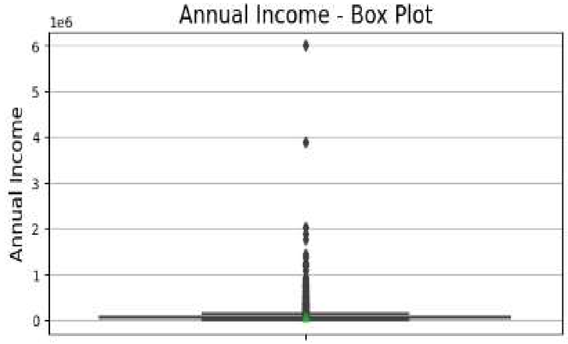
5 rows × 25 columns

Data Type conversion

- int_rate and revol_util columns converted to Float
- annual_inc and term converted to int

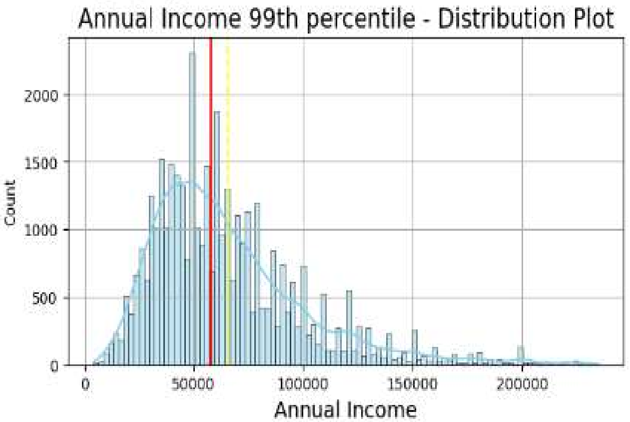
Data Clean UP

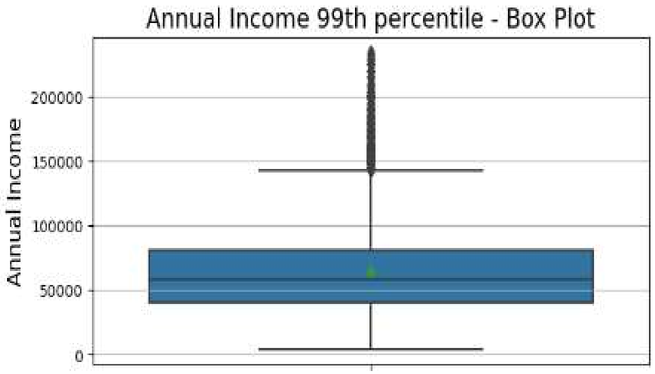





```
### As observed from the box plot annual_inc shows an exponential increase around the 99th percentile. Remove above the 99th percentile values.
loan_data = loan_data[loan_data.annual_inc<=np.percentile(loan_data.annual_inc,99)]
```

```
# Univariate analysis on "annual_inc" after treating outliers
print(loan_data['annual_inc'].describe())
```

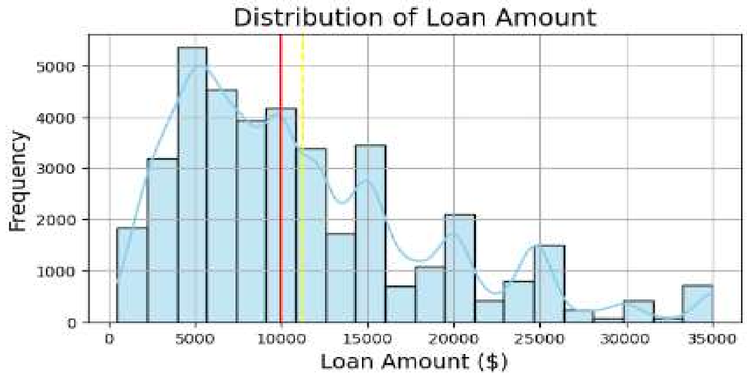




Outliers treatment

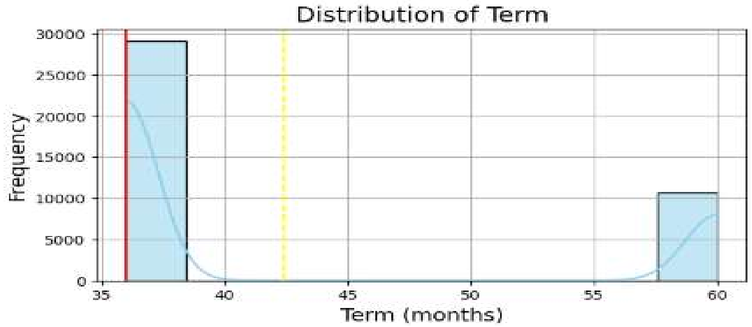
- As observed from the box plot annual_inc shows an exponential increase around the 99th percentile.
- Removed above the 99th percentile values.

Univariate Analysis



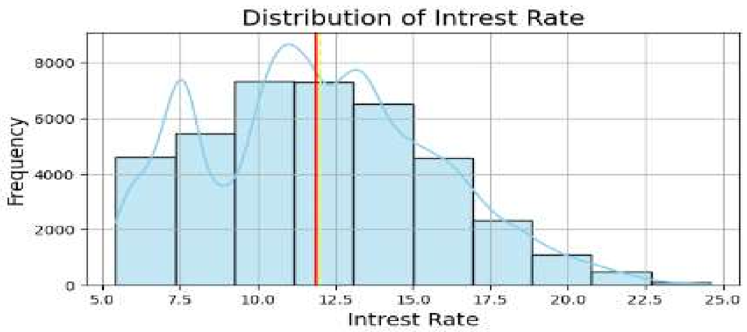
Loan Amount

- Most of the borrowers taken loan amounts between 5500 - 15000
- 99-95 percentile of loan amounts are below 30000



Term

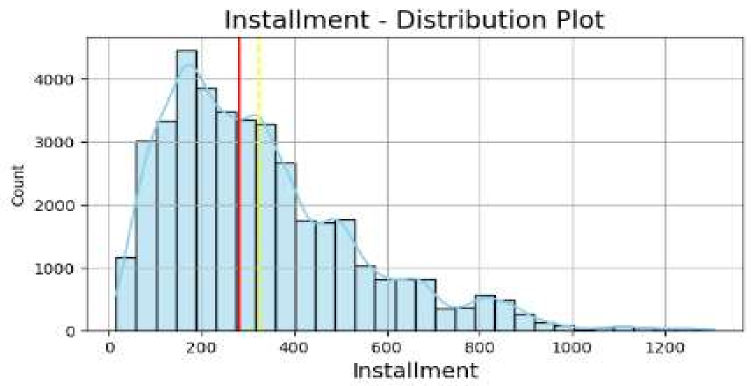
- 36 months term borrowers are more

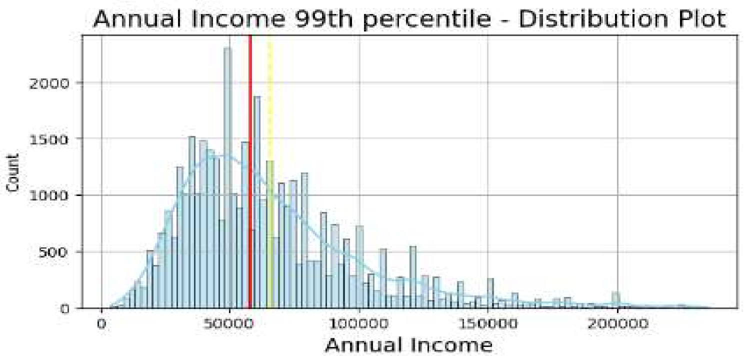


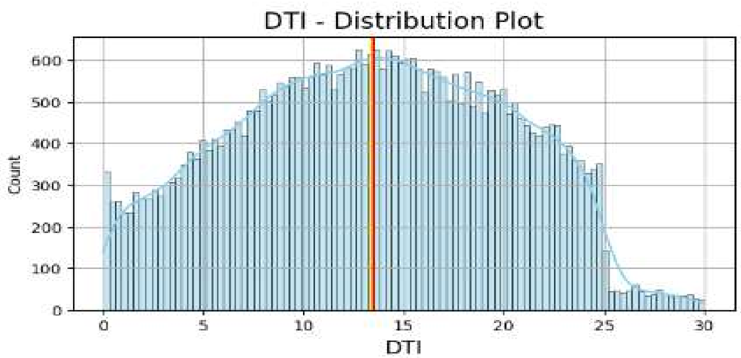
Interest Rate

- As interest rate increases from 14% number of borrowers are less.
- Majority of the borrowers intrest rate is between 9.25 to 14.59

Univariate Analysis







Installment

- **Most of the instalments are in between 167 to 430**
- **lowest installment is 15 and highest installment is 1305**
- high installment borrowers are few and low installment borrowers are high

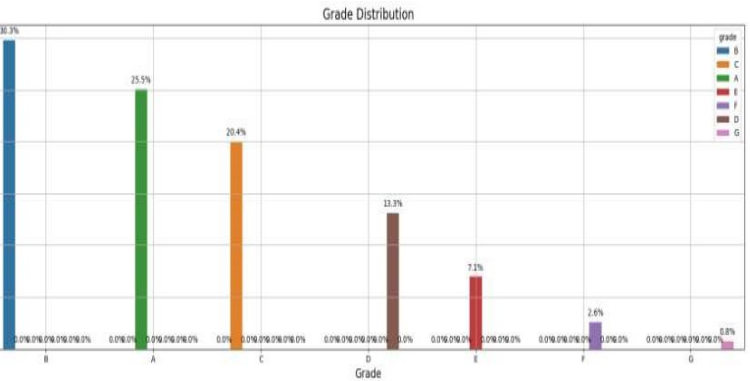
Annual Income

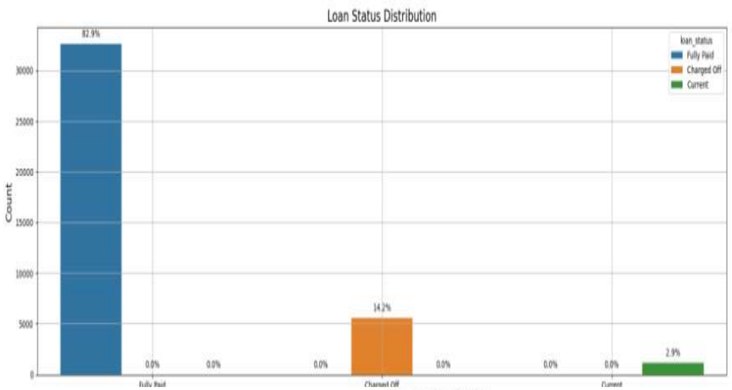
- **50000 thousand annual income borrowers are more with compare to other income borrowers.**
- **Most of the borrowers income is in between 4000 to 81000**

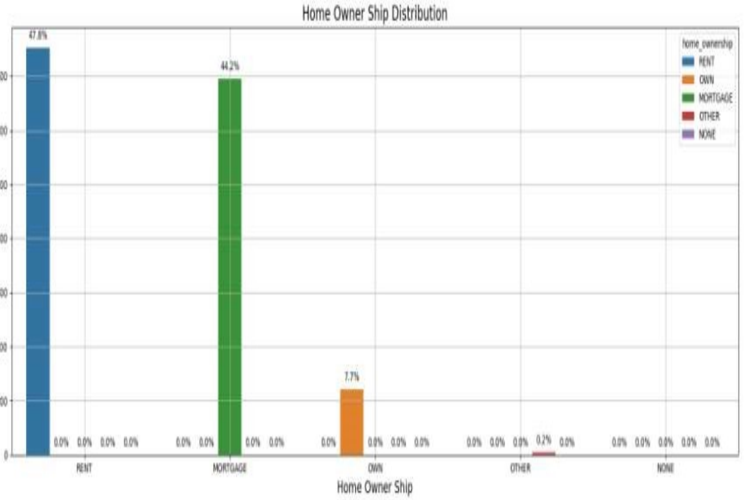
DTI

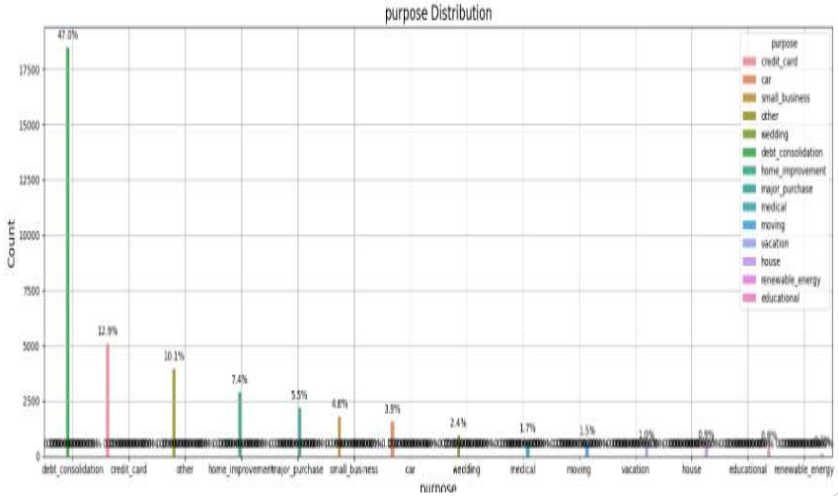
- **Average debt to income ratio is 13.37**
- **Most of the borrowers debt to income ration is in between 8.27 to 18.64**

Categorical Univariate Analysis





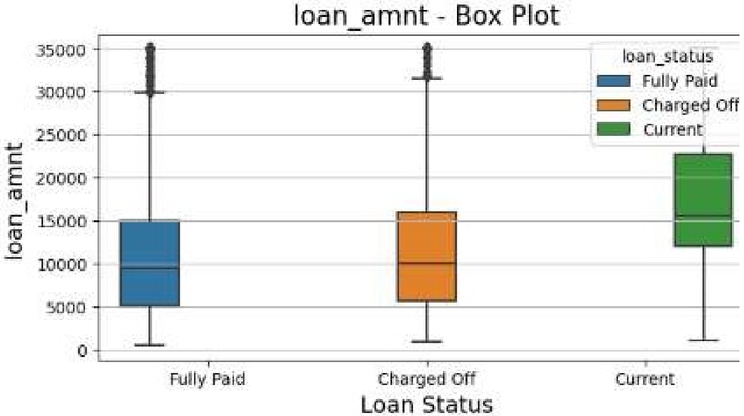


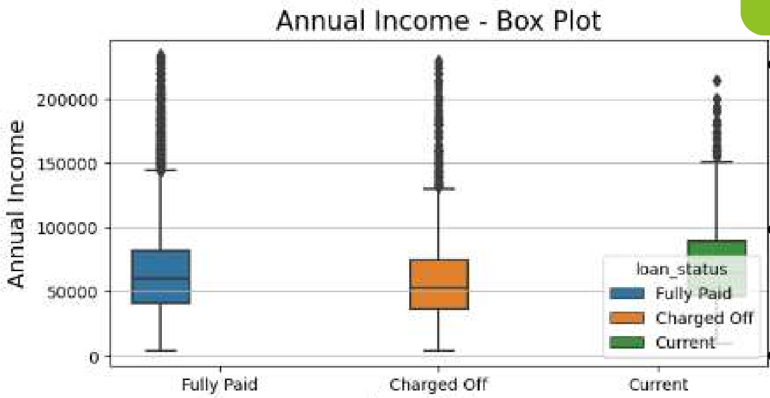


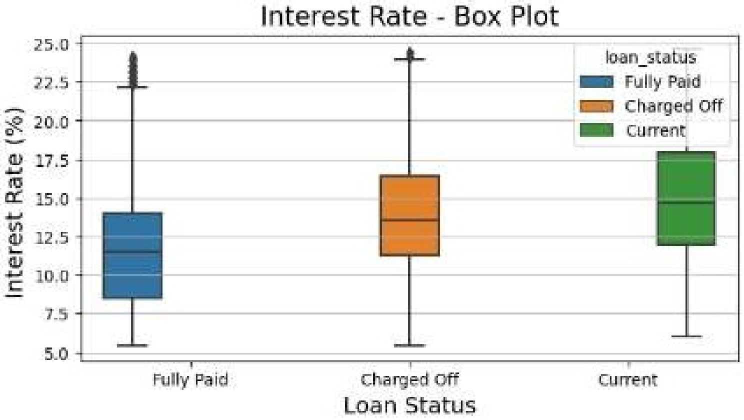
Insights

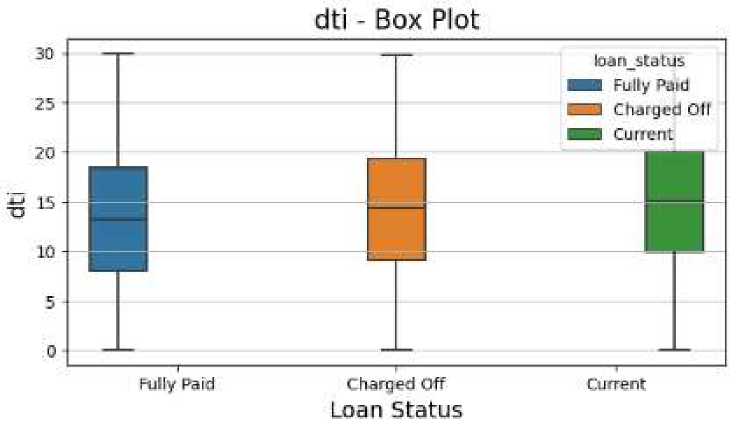
- B grade borrowers are more
- Rent borrowers are more
- 47% borrowers are taken loans for Debt consolidation
- 14.2% borrowers are defaulters

Bivariate analysis





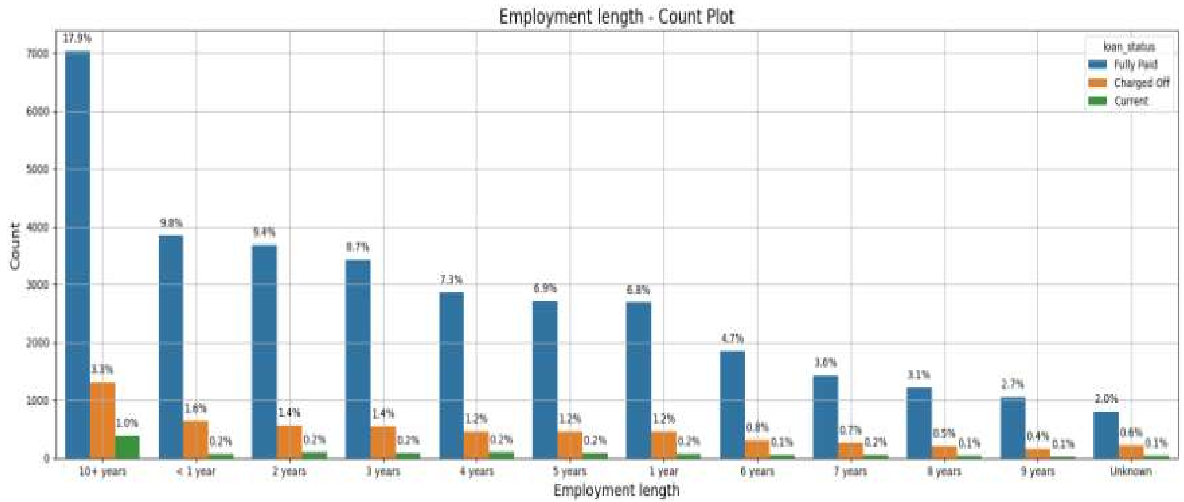


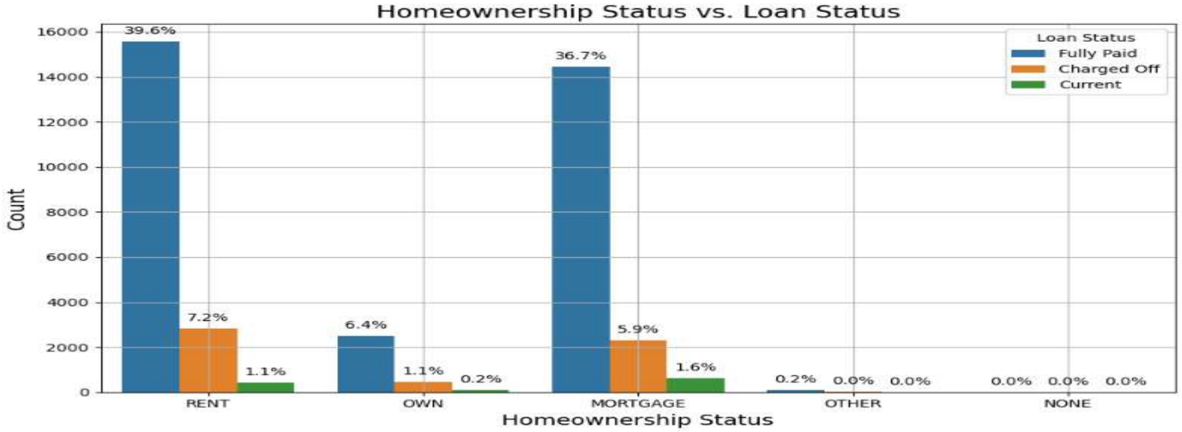


Insights

- Charged Off borrowers median compared to fully paid borrowers is high and risk is associated with higher loan amounts.
- Charged of 75th quartile is higher, require proper risk analysis for high loan amounts.
- As loan rate increasing from 14.5 number of applications are decreasing.
- Most of the borrowers interest rates are between 9.25 to 14.59
- Fully paid customers interest rates are low with compare to defaulters.
- If interest rate is high then there is probability to default loan
- Most of the borrowers salary is 60000
- Less salary borrowers are becoming defaulters and avg salary of charged off borrowers is less than fully paid borrowers
- Defaulted loan DTI is more when compared with fully paid loans.
- If the dti is more then there is chance to default loan.

Bivariate analysis

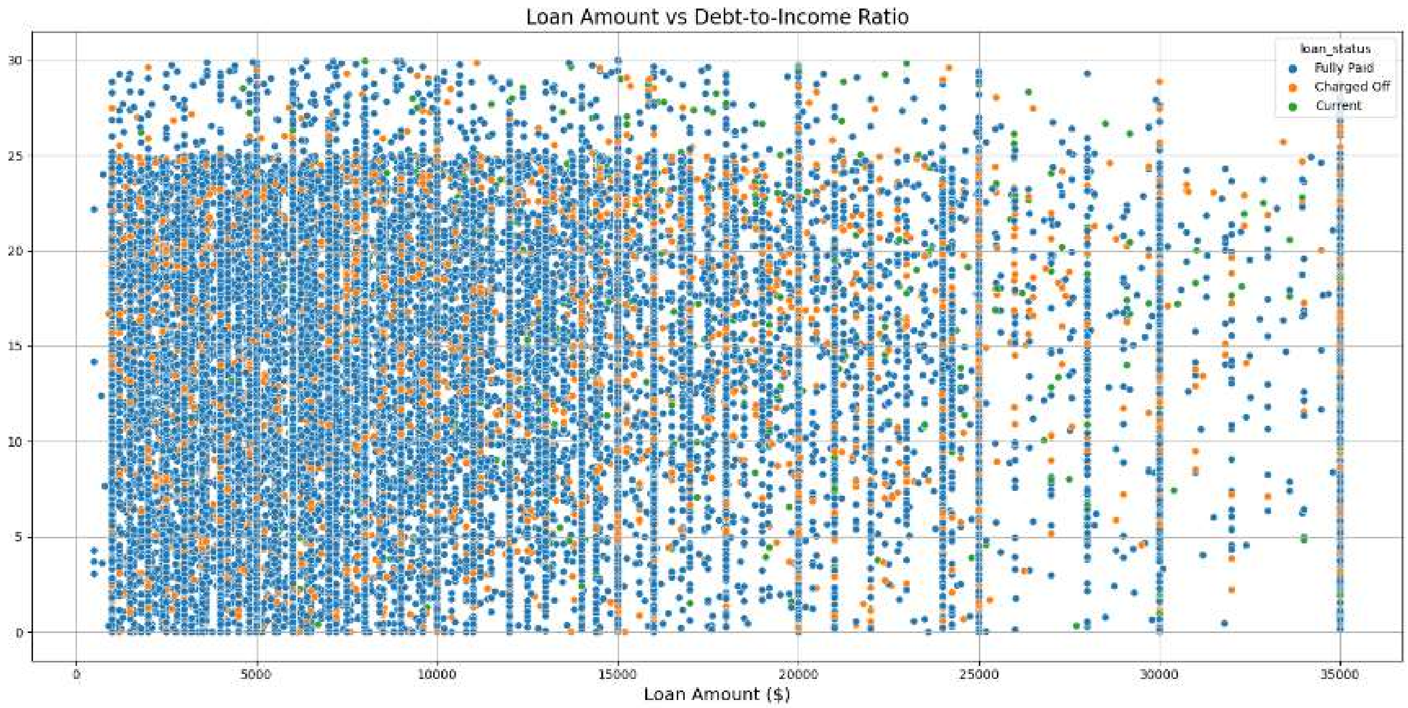




Insights

- 10+ years employee borrowers are high.
- 1 year to 9 years as experience increase number of borrowers are decreasing
- Rent and mortgage borrowers are more defaulters.

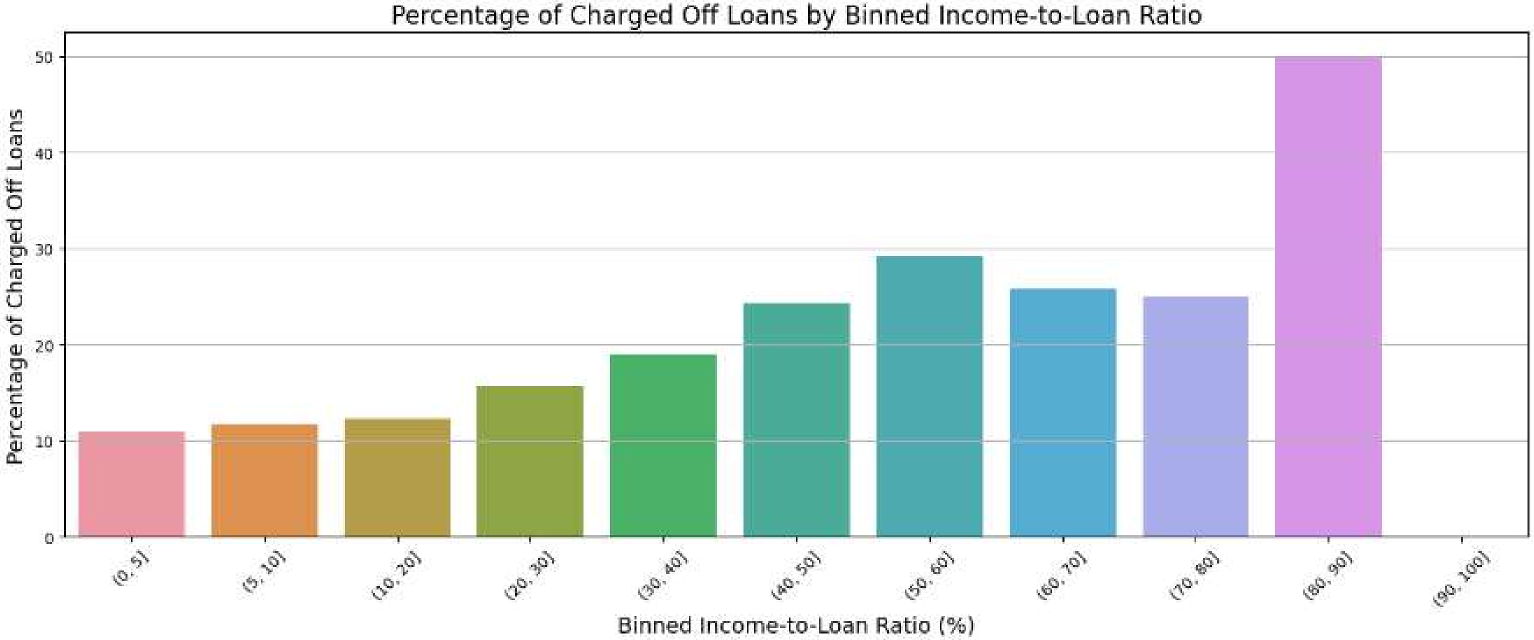
Bivariate analysis



Insights

- Chances of the loan being Charged Off increase as DTI increases.

Segment analysis



Insights

- Till loan amount less than 20% of annual income, loan charge off is low
- Loan amounts percentage of annual income increases loan charge off rate increase.

Key variables impacting Loan status

- DTI
- Interest Rates
- Loan Term
- Employment Length
- Home ownership
- Purpose
- Loan Grade

Summary

- Debt-to-income ratio (DTI) is positively correlated with loan default, higher DTI ratios increase the risk of default.
- Higher interest loans are likely to be charged off compared to fully paid loans, this is a potential risk associated with higher interest rates.
- The length of the loan term is increasing the defaulters, longer-term loans having higher default rates compared to shorter-term loans.
- Employment length increases likelihood of loan default decrease, longer employment tenure might reduce the risk of default.
- Home owner ship exhibiting lower default rates compared to rent and mortgage borrowers.
- Loan purpose impacts default rates, loans for debt consolidation having relatively lower default rates compared to others.
- Higher-grade loans are lower default rates.

Conclusion

- Using EDA techniques analysed given data set thoroughly.
- Identified key attributes which influence loan status to default.
- Loan amount, debt to income ratio, employment length and borrower behaviour are key factors which will impact loan status
- In future to mitigate and reduce financial risk lender requires more attention on high loan amount and high DTI applicants.