Lending Club Case Study

Renganayaki S

Balla Mallibabu

Agenda

- ✓ Problem Statement
- ✓ Exploratory Data Analysis
    - ✓ Data Loan & Understanding
    - ✓ Data Cleaning
    - ✓ Segmentation
    - ✓ Univariate Analysis
    - ✓ Bivariate Analysis
- ✓ Inferences

Problem Statement

- This Finance company is the largest online loan marketplace, facilitating personal loans, business loans, and financing of medical procedures. Borrowers can easily access lower interest rate loans through a fast online interface.
- The finance company aims to minimize losses by identifying risky loan applicants. Approving a risky applicant may lead to defaults and financial loss, while rejecting a reliable applicant results in lost business.
- Using Exploratory Data Analysis (EDA), the goal is to analyze past loan data to identify key factors that predict loan defaults, enabling better decision-making to reduce credit risk and improve portfolio management..

Data Load & Understanding

- Data set contains 39717 records and 111 columns
- Need to perform data clean as data contains many columns with null values
- Also, need to convert data type for few columns and drop the rows which are not required

```
loan_raw_data = pd.read_csv("loan.csv", sep=",")
loan_raw_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
```

```
C:\Users\81008015\AppData\Local\Temp\ipykernel_15732\1758096309.py:1: DtypeWarning: Columns (47) ha
type option on import or set low_memory=False.
  loan_raw_data = pd.read_csv("loan.csv", sep=",")
```

```
loan_raw_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
```

Data Cleaning

Fix Rows:

- Loan Raw data not having, Summary Rows (Total, sub total rows), Extra rows (Column number indicator rows, blank rows, section indicator rows)

Fix Columns:

- Identified 56 Columns with complete null values

```
In [23]:  # Identify columns with >80% null values

          unnecessary_Columns = loan_raw_data.columns[(loan_raw_data.isnull().sum()/loan_raw_data.shape[0])*100 > 80]
```

```
In [30]:  # All the below mentioned columns are unnecessary for anlaysis

          print(unnecessary_Columns.size )
          print(unnecessary_Columns)

          56
          Index(['mths_since_last_record', 'next_pymnt_d', 'mths_since_last_major_derog',
                 'annual_inc_joint', 'dti_joint', 'verification_status_joint',
                 'tot_coll_amt', 'tot_cur_bal', 'open_acc_6m', 'open_il_6m',
                 'open_il_12m', '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_fi', 'total_cu_tl', 'inq_last_12m',
                 'acc_open_past_24mths', 'avg_cur_bal', 'bc_open_to_buy', 'bc_util',
                 'mo_sin_old_il_acct', 'mo_sin_old_rev_tl_op', 'mo_sin_rcnt_rev_tl_op',
                 'mo_sin_rcnt_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_pd',
                 '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_12m', 'pct_tl_nvr_dlq', 'percent_bc_gt_75',
                 'tot_hi_cred_lim', 'total_bal_ex_mort', 'total_bc_limit',
                 'total_il_high_credit_limit'],
                dtype='object')
```

```
In [83]:  # Drop all the unnecessary columns which has more >80% null values

          # loan_cleaned_data = loan_raw_data.dropna(thresh=loan_raw_data.shape[0] * 0.2, axis=1)

          loan_cleaned_data = loan_raw_data.drop(unnecessary_Columns, axis=1)
```

Data Cleaning

Fix Columns:

- Data Set share after dropping Null columns (39717 rows, 54 columns)
- Identified 9 columns which contains sinlge non-null value which will not be use full for any analysis, dropping this columns. Data Shape (46 columns)

In [87]:

```
#Identify the columns which contain a single unique value across all records
loan_cleaned_data.loc[:, loan_cleaned_data.nunique() == 1]
```

Out[87]:

	pymnt_plan	initial_list_status	collections_12_mths_ex_med	policy_code	application_type	acc_now_delinq	chargeoff_within_12_mths	delinq_amnt
0	n	f	0.0	1	INDIVIDUAL	0	0.0	0
1	n	f	0.0	1	INDIVIDUAL	0	0.0	0
2	n	f	0.0	1	INDIVIDUAL	0	0.0	0
3	n	f	0.0	1	INDIVIDUAL	0	0.0	0
4	n	f	0.0	1	INDIVIDUAL	0	0.0	0
...	...	...	...	...	...	...	...	...
39712	n	f	NaN	1	INDIVIDUAL	0	NaN	0
39713	n	f	NaN	1	INDIVIDUAL	0	NaN	0
39714	n	f	NaN	1	INDIVIDUAL	0	NaN	0
39715	n	f	NaN	1	INDIVIDUAL	0	NaN	0
39716	n	f	NaN	1	INDIVIDUAL	0	NaN	0

39717 rows × 8 columns

In [88]:

```
Idnentified 9 columns which contains sinlge non-null value which will not be use full for any analysis - dropping this column
an_cleaned_data_1 = loan_cleaned_data.loc[:, loan_cleaned_data.nunique() > 1]
```

In [89]:

```
loan_cleaned_data_1.shape
```

Out[89]:

```
(39717, 46)
```

Data Cleaning

Dropping below columns as no helpful to derive insights
1) desc - Data provided by browser (it has various details like loan Amount requested etc..)
but all are in unstructured format and with 33% null values so not useful for analysis
2) mths_since_last_delinq: This column has around 65% null values
loan_cleaned_data_1 = loan_cleaned_data_1.drop(['desc','mths_since_last_delinq'], axis=1)
3) ID and Member_ID columns are index columns, which are not useful in analyis, hence dropping these columns
4) Dropping additional columns which are not derive much insights

```
Name: member_id, Length: 39717, dtype: int64>

In [54]: # Dropping columns  desc and mths_since_last_delinq due to below reasons
         # 1) desc - Data provided by browser (it has various details like loan Amount requested etc..)
         #           but all are in unstructured format and with 33% null values so not useful for analysis
         # 2) mths_since_last_delinq: This column has around 65% null values
         loan_cleaned_data_1 = loan_cleaned_data_1.drop(['desc','mths_since_last_delinq'], axis=1)
         # ID and Member_ID columns are index columns, which are not useful in analyis, hence dropping these columns
         loan_cleaned_data_1 = loan_cleaned_data_1.drop(['id','member_id'], axis=1)

In [55]: loan_cleaned_data_1.shape

Out[55]: (39717, 42)
```

```
[80]: # Dropping below columns as data is not suitable for analysis
      DrppingColumns = ['emp_title', 'url', 'title','out_prncp','out_prncp_inv','collection_recovery_fee','pub_rec',
                        'zip_code','total_acc','total_rec_late_fee','recoveries','revol_bal','revol_util', 'installment',
                        'last_credit_pull_d','last_pymnt_amnt','last_pymnt_d','total_pymnt','total_pymnt_inv','total_rec_int',
                        'total_rec_prncp']

[81]: loan_cleaned_data_1 = loan_cleaned_data_1.drop(DrppingColumns, axis=1)

[82]: loan_cleaned_data_1.shape

Out[82]: (39717, 21)
```

Data Cleaning

1) No Duplicate rows exist
2) No Rows present with more than 5 null values
3) Drop the Rows with loan_status as 'Current' as Current: Applicant is in the process of paying the instalments, i.e. the tenure of the loan is not yet completed. These candidates are not labelled as 'defaulted'.
4) Removed records with NA values

```
In [124]:
# Drop the Rows with Loan_status as 'Current'  as
# Current: Applicant is in the process of paying the instalments, i.e. the tenure of the Loan is not yet completed.
# These candidates are not Labelled as 'defaulted'.

loan_cleaned_data_1[ loan_cleaned_data_1.loan_status == 'Current']
```

Out[124]:

loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	grade	sub_grade	emp_length	home_ownership	annual_inc	...	issue_d	loan_status	purpos

0 rows × 21 columns

```
In [125]:
#dopping 1140 rcords which are having Loan satus as 'Current' as those candiates are not Lebelled as 'Dafaulter'

loan_cleaned_data_1 =loan_cleaned_data_1[~(loan_cleaned_data_1.loan_status == 'Current')]
```

```
In [126]:
loan_cleaned_data_1.shape
```

Out[126]: (34034, 21)

```
In [127]:
loan_cleaned_data_1 = loan_cleaned_data_1[loan_cleaned_data_1['emp_length'].notna()]
```

```
In [93]:
loan_cleaned_data_1 = loan_cleaned_data_1[loan_cleaned_data_1['pub_rec_bankruptcies'].notna()]
```

```
In [94]:
loan_cleaned_data_1.shape
```

Out[94]: (36847, 21)

Data Cleaning

1) Data standardization
   1) Int_Rate, Loan_amnt, funded_amnt, issue_d, emp_length columns data is standardized by changing data type, converting to valid data formats as in below

```
In [128]:  #Standardize column data types

           #interest rate - is object -- should be converted to float by removing the %
           loan_cleaned_data_1['int_rate'] = pd.to_numeric(loan_cleaned_data_1['int_rate'].replace('%', '', regex=True))

In [129]:  loan_cleaned_data_1['loan_amnt'] = loan_cleaned_data_1['loan_amnt'].astype(float)
           loan_cleaned_data_1['funded_amnt'] = loan_cleaned_data_1['funded_amnt'].astype(float)

In [130]:  #Convert issue_d in to proper date format
           loan_cleaned_data_1['issue_d'] = pd.to_datetime(loan_cleaned_data_1['issue_d'],format='%b-%y')
           loan_cleaned_data_1['issue_d']

Out[130]:  0       2011-12-01
           1       2011-12-01
           2       2011-12-01
           3       2011-12-01
           5       2011-12-01
                      ...
           39562   2007-11-01
           39573   2007-11-01
           39623   2007-10-01
           39666   2007-08-01
           39680   2007-08-01
           Name: issue_d, Length: 34034, dtype: datetime64[ns]

In [131]:  #update Emp_Length column by

           loan_cleaned_data_1['emp_length'].replace('years', '', regex = True)

Out[131]:  0        10
           1         1
           2        10
           3        10
           5         3
                    ..
           39562     1
           39573     3
           39623     8
           39666     2
```

Data Segmentation

1) Data variables divided in to 3 sections as below

Categorical Variables
addr_state
earliest_cr_line
emp_length
grade
home_ownership
issue_d
loan_status
purpose
sub_grade
term
verification_status
pub_rec_bankruptcies

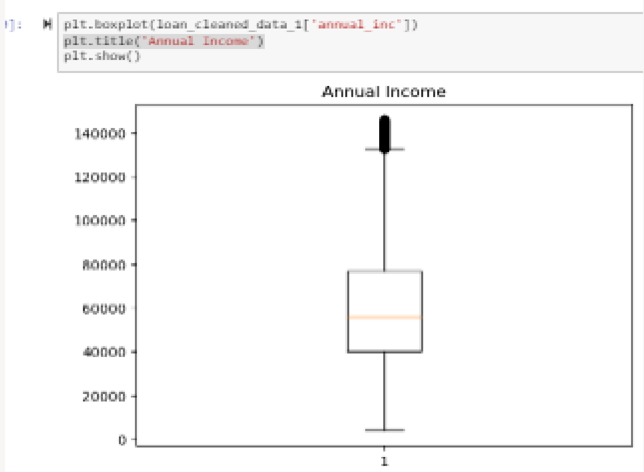
Numerical Variables
annual_inc
delinq_2yrs
dti
funded_amnt
inq_last_6mths
int_rate
loan_amnt
open_acc
funded_amnt_inv

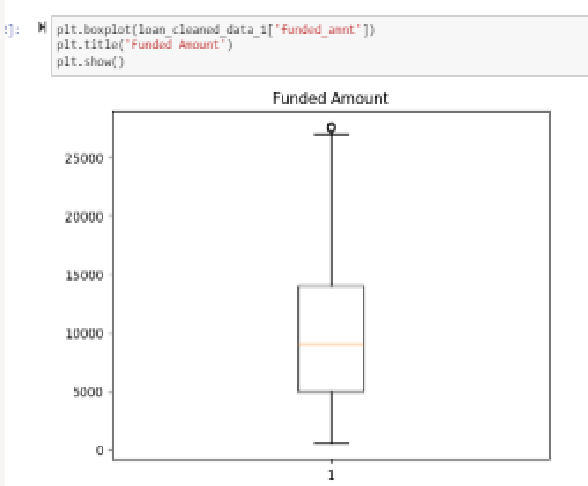
Additional Columns (with complete Null Values / singel values / imporper data)				
acc_now_delinq	total_rev_hi_lim	dti_joint	member_id	pub_rec
acc_open_past_24mths	mths_since_last_record	emp_title	num_tl_30dpd	all_util
tot_hi_cred_lim	percent_bc_gt_75	bc_util	inq_fi	pymnt_plan
mths_since_last_delinq	num_tl_op_past_12m	avg_cur_bal	il_util	tax_liens
mths_since_recent_bc	num_tl_120dpd_2m	next_pymnt_d	open_acc_6m	title
mths_since_recent_inq	last_fico_range_low	bc_open_to_buy	open_il_12m	tot_coll_amt
mths_since_recent_revol_delinq	mo_sin_old_il_acct	num_actv_bc_tl	open_il_24m	tot_cur_bal
num_accts_ever_120_pd	mo_sin_old_rev_tl_op	num_actv_rev_tl	open_il_6m	total_bal_il
chargeoff_within_12_mths	mo_sin_rcnt_rev_tl_op	num_bc_sats	open_rv_12m	total_bc_limit
collection_recovery_fee	mo_sin_rcnt_tl	num_bc_tl	open_rv_24m	total_cu_tl
collections_12_mths_ex_med	mort_acc	num_il_tl	out_prncp	desc
num_tl_90g_dpd_24m	annual_inc_joint	num_op_rev_tl	out_prncp_inv	url
total_il_high_credit_limit	total_bal_ex_mort	num_rev_accts	pct_tl_nvr_dlq	installment
mths_since_recent_bc_dlq	mths_since_last_record	application_type	inq_last_12m	id
num_rev_tl_bal_gt_0	mths_since_rcnt_il	num_sats	policy_code	
verification_status_joint	fico_range_high	fico_range_low	max_bal_bc	
last_credit_pull_d	last_pymnt_amnt	last_pymnt_d	initial_list_status	
mths_since_last_major_derog	last_fico_range_high	zip_code	delinq_amnt	

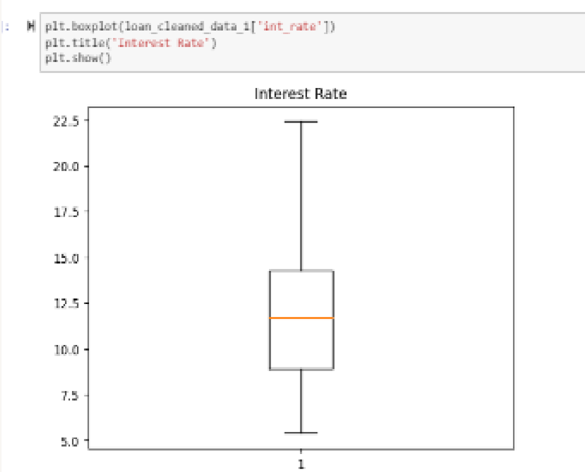
Additional variables includes (Variables with full null values, and columns which contains only single value, index columns, and extra columns which are not helping much in deriving any insights

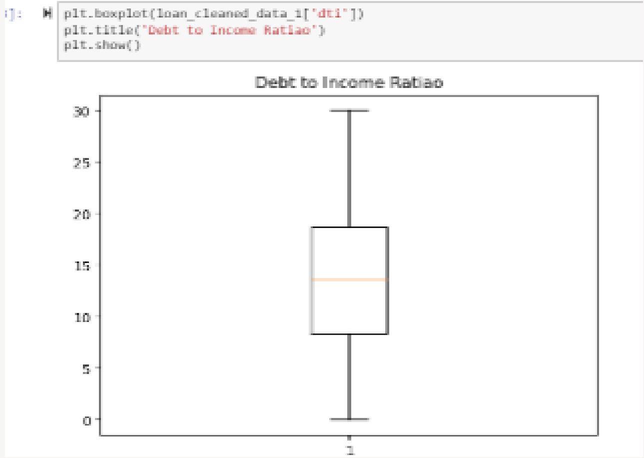
Outlier Analysis using Box Plot

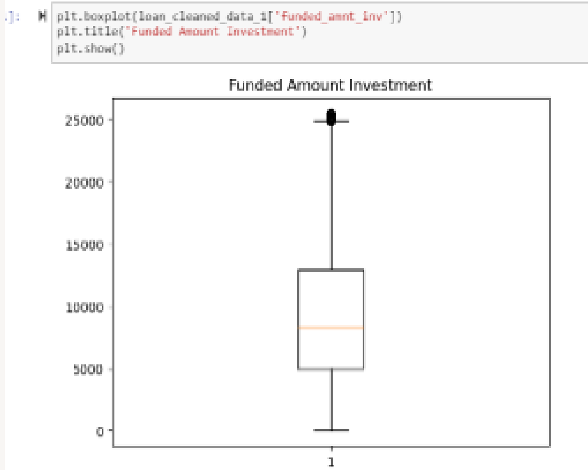
1) Identified Outlier in below columns
    1) Int_Rate, Loan_amnt, funded_amnt, issue_d, emp_length columns data is standardized by changing data type, converting to valid data formats as in below except DTI.

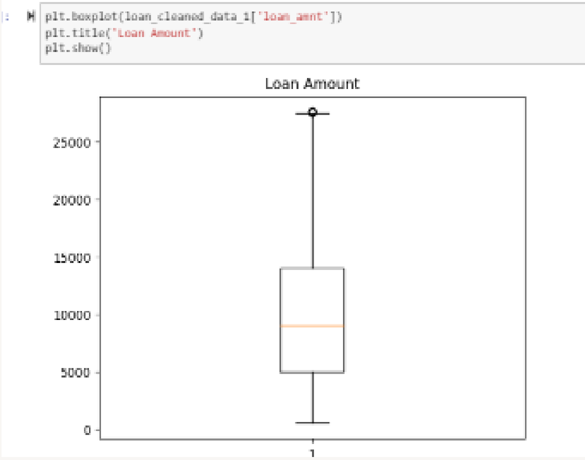








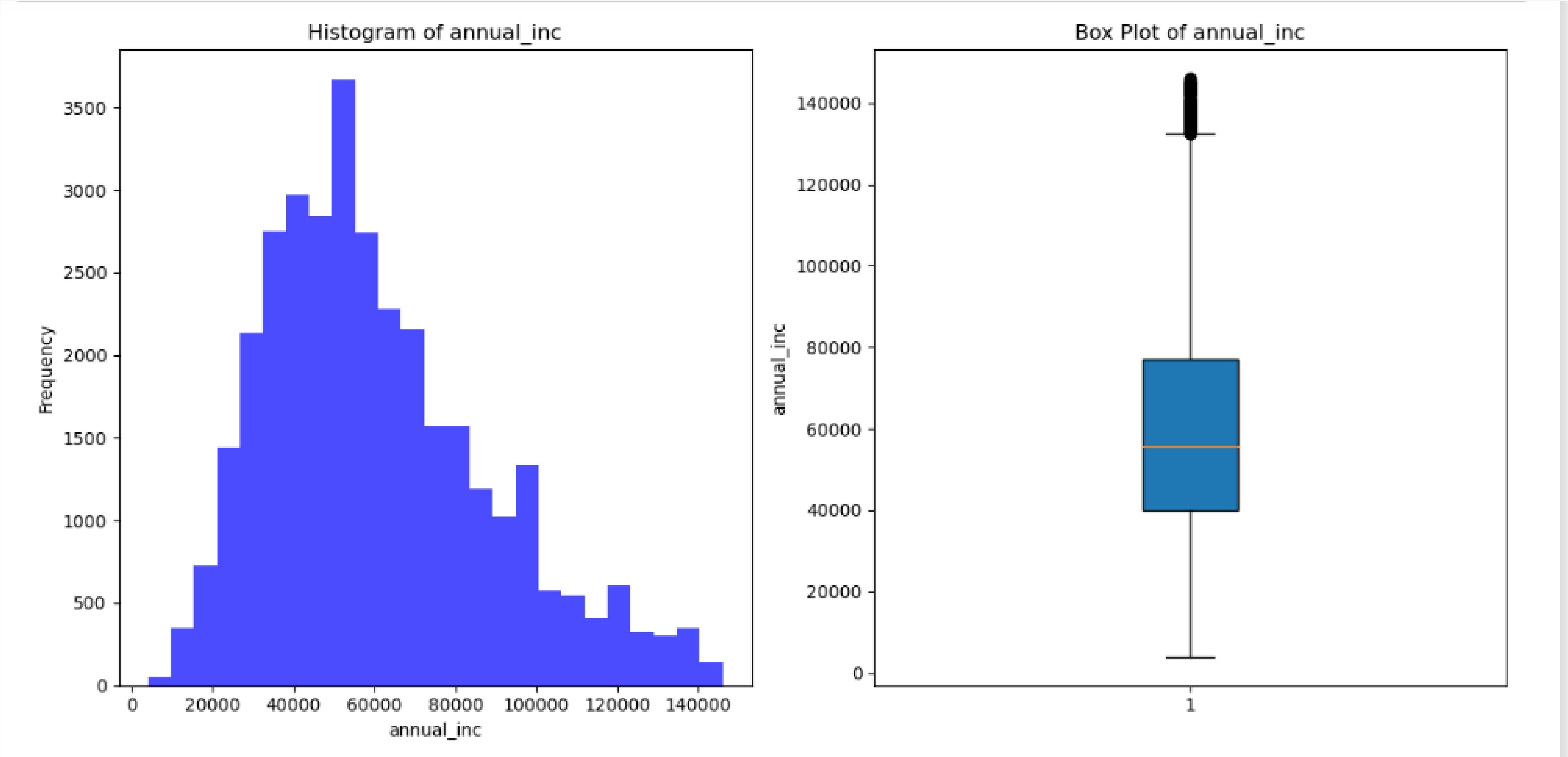




Univariate Analysis

Annual Income Variable observations:

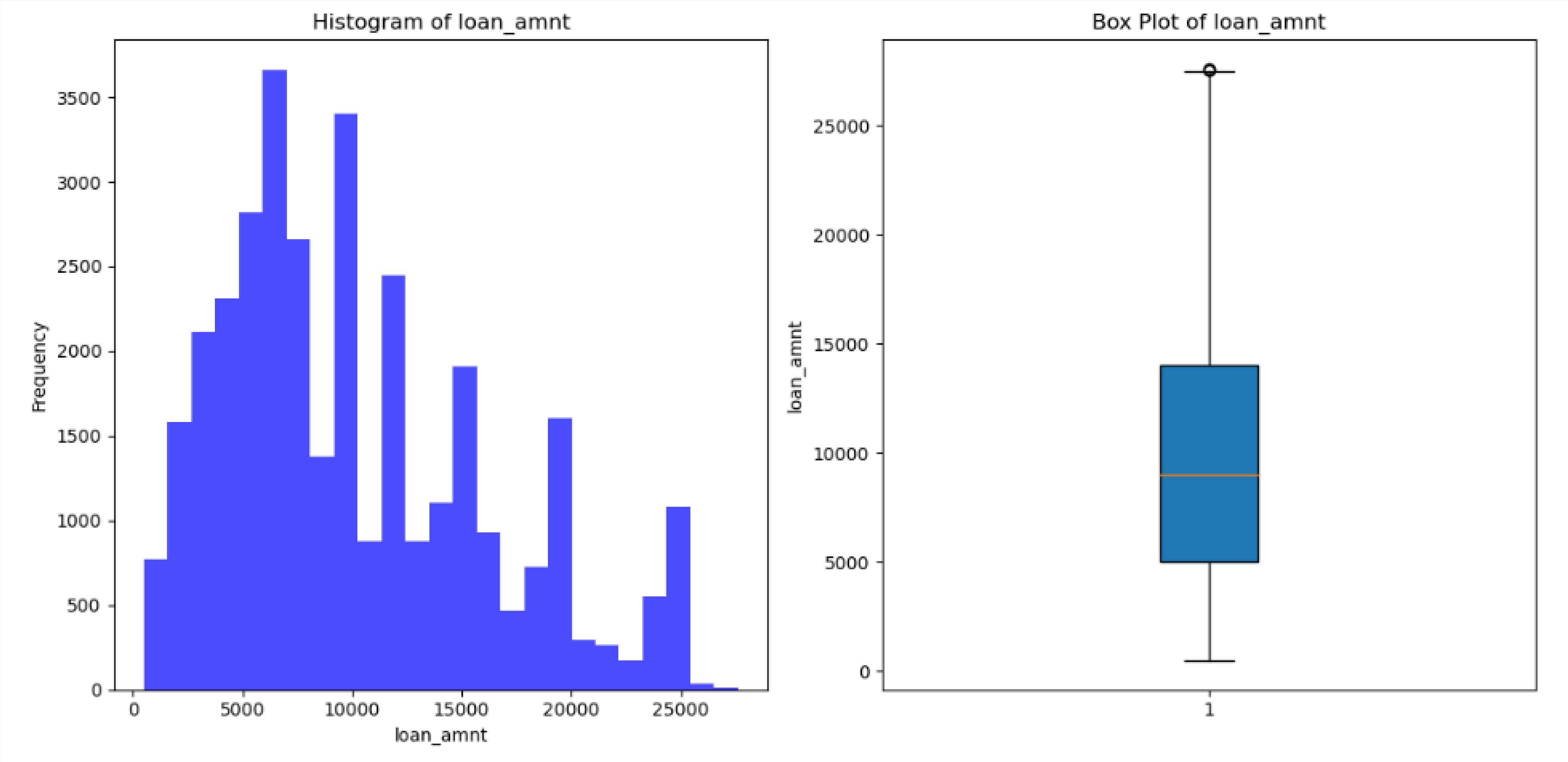
- Annual Income of Most applicants is in between 40000-77000 as per IQR
- Average Annual Income: 60880



Univariate Analysis

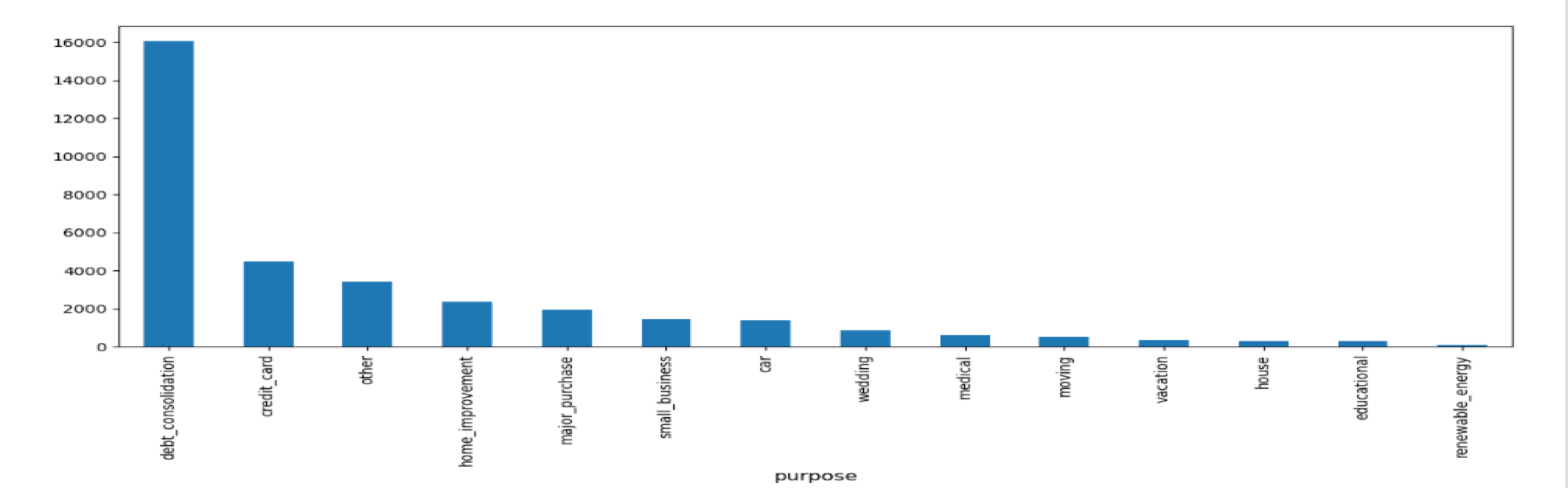
Loan Amount Variable observations:

- Loan Amount of Most applicants is in between 5K to 14K as per IQR
- Average Loan Amount: ~10K



Univariate Analysis (Categorical Variables)

- Major portion of the loan applicants are
  - From mortgage or rental homes under home ownership
  - From 36 month loan term
  - Most applicants verification status is **not Verified**
  - Most loan applicants are from CALIFORNIA
  - Major loan applicants purpose is ‘Debt Consolidation’
  - Most applicants are either below below 1 year or 10+ years employee length
  - Majority portion is from Grade A and Grade B



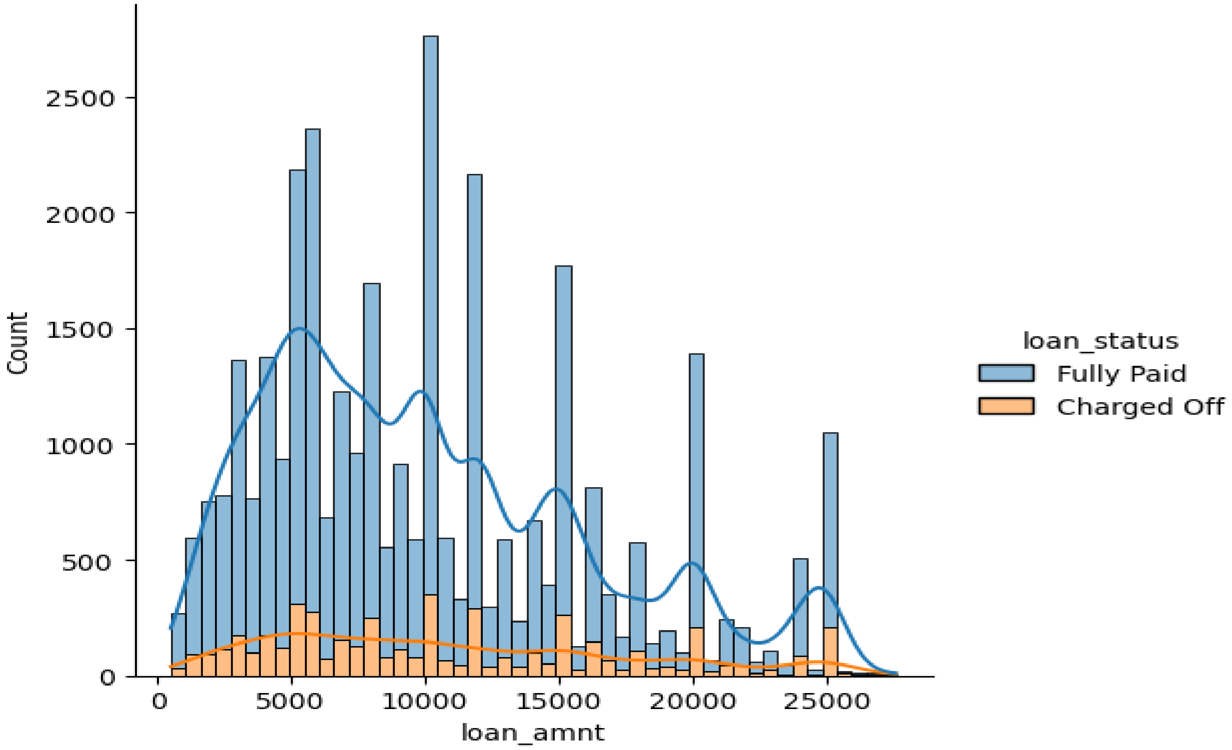
Bivariate Analysis

Loan Amount versus Loan Status analysis:

- people who take loan between 5000 and 10000 are more prone to be default

```
# Analysis of Loan status ang grade
sns.displot(data=loan_cleaned_data_1, x="loan_amnt", hue="loan_status", multiple="stack", kde=True)
```

```
93]: <seaborn.axisgrid.FacetGrid at 0x1daed9b8250>
```

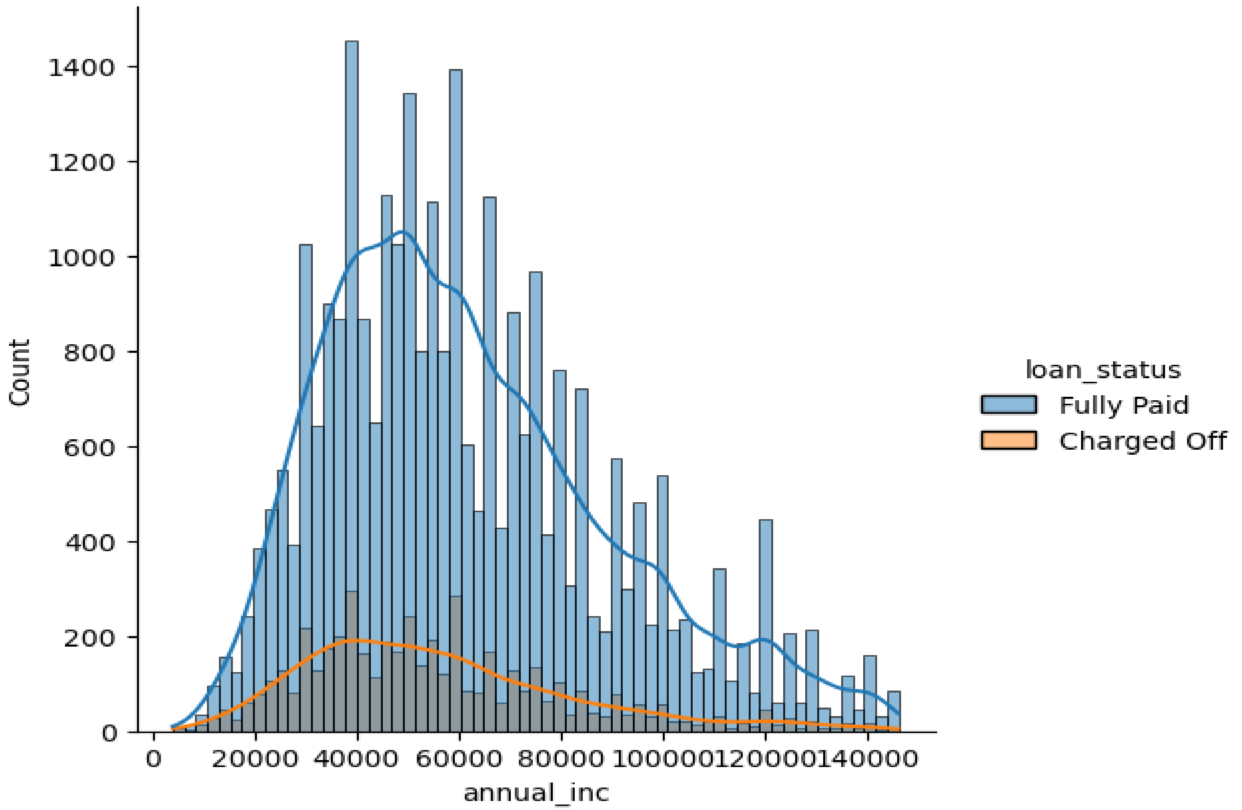


Bivariate Analysis

Annual Income versus Loan Status analysis:
- people who has annual incomes between 30K to 60k are more prone to be defaulters

```
#find correlation between  annual income and loan_status
sns.displot(data=loan_cleaned_data_1, x='annual_inc',hue='loan_status', kde=True)
# does this tell us that people who  have annual come less than 80000 are more   prone to be defaulters ?
```

```
: <seaborn.axisgrid.FacetGrid at 0x1daecf11490>
```

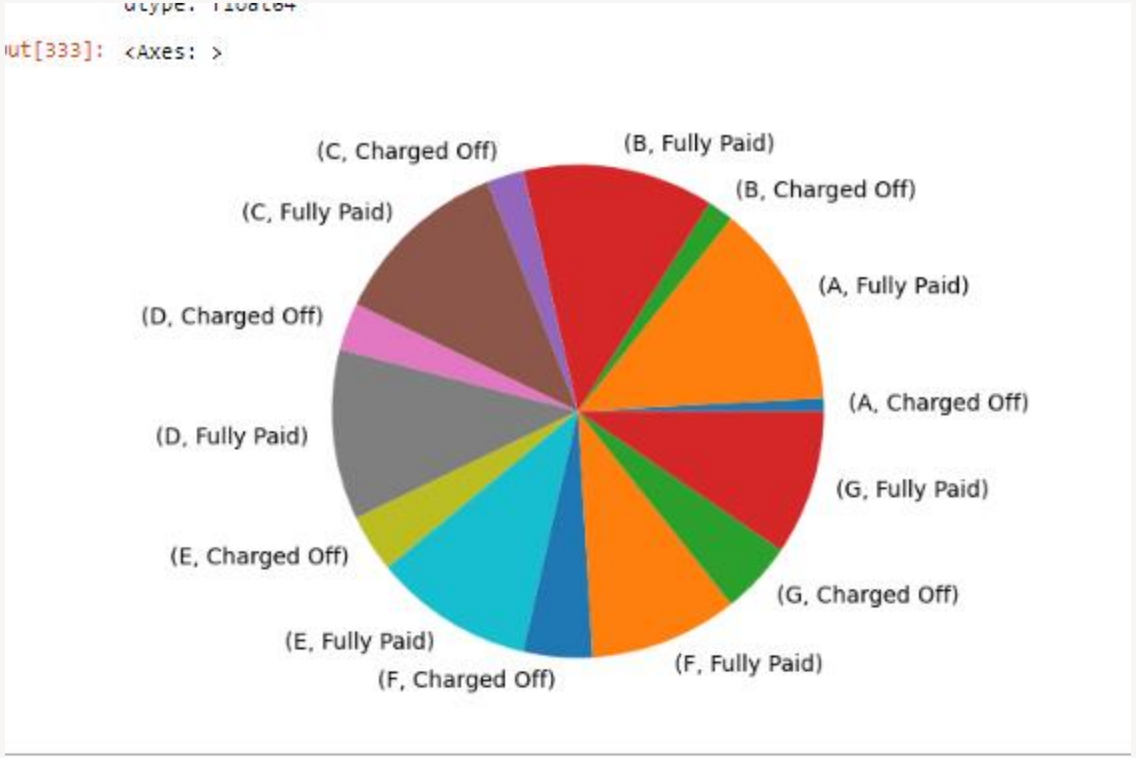


Bivariate Analysis

Visualizing data to understand correlation of Grade with defaulters
Inference : The percentage of defaulters increases with grade A to G

```
grade  loan_status
A      Charged Off     5.953560
       Fully Paid     94.046440
B      Charged Off    12.009592
       Fully Paid     87.990408
C      Charged Off    17.100000
       Fully Paid     82.900000
D      Charged Off    21.713903
       Fully Paid     78.286097
E      Charged Off    26.516220
       Fully Paid     73.483780
F      Charged Off    31.750339
       Fully Paid     68.249661
G      Charged Off    33.333333
       Fully Paid     66.666667
dtype: float64

]: <Axes: >
```

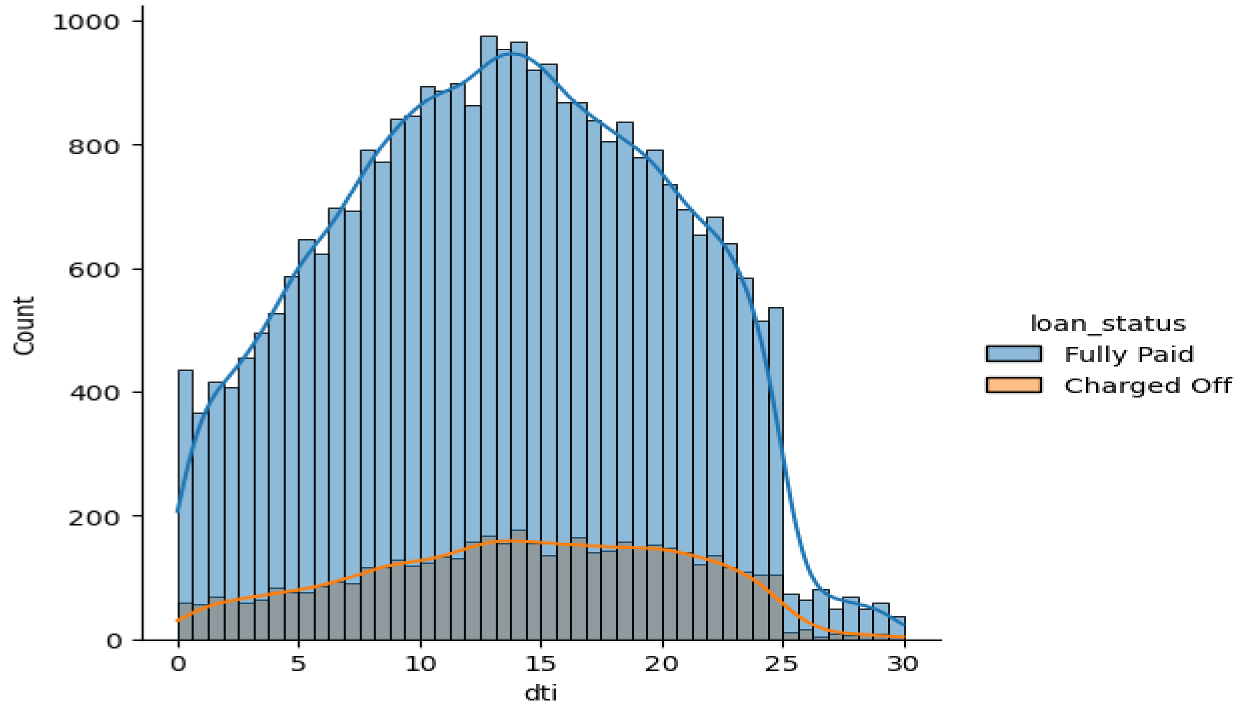


Bivariate Analysis

DTI versus Loan Status analysis:
- people who has DTI between 8 to 20 are more prone to be defaulters

```python
sns.displot(data=loan_cleaned_data_1, x='dti',hue='loan_status', kde=True)
# Inference : customers with dti between 10 and 20 are more likely to default
```

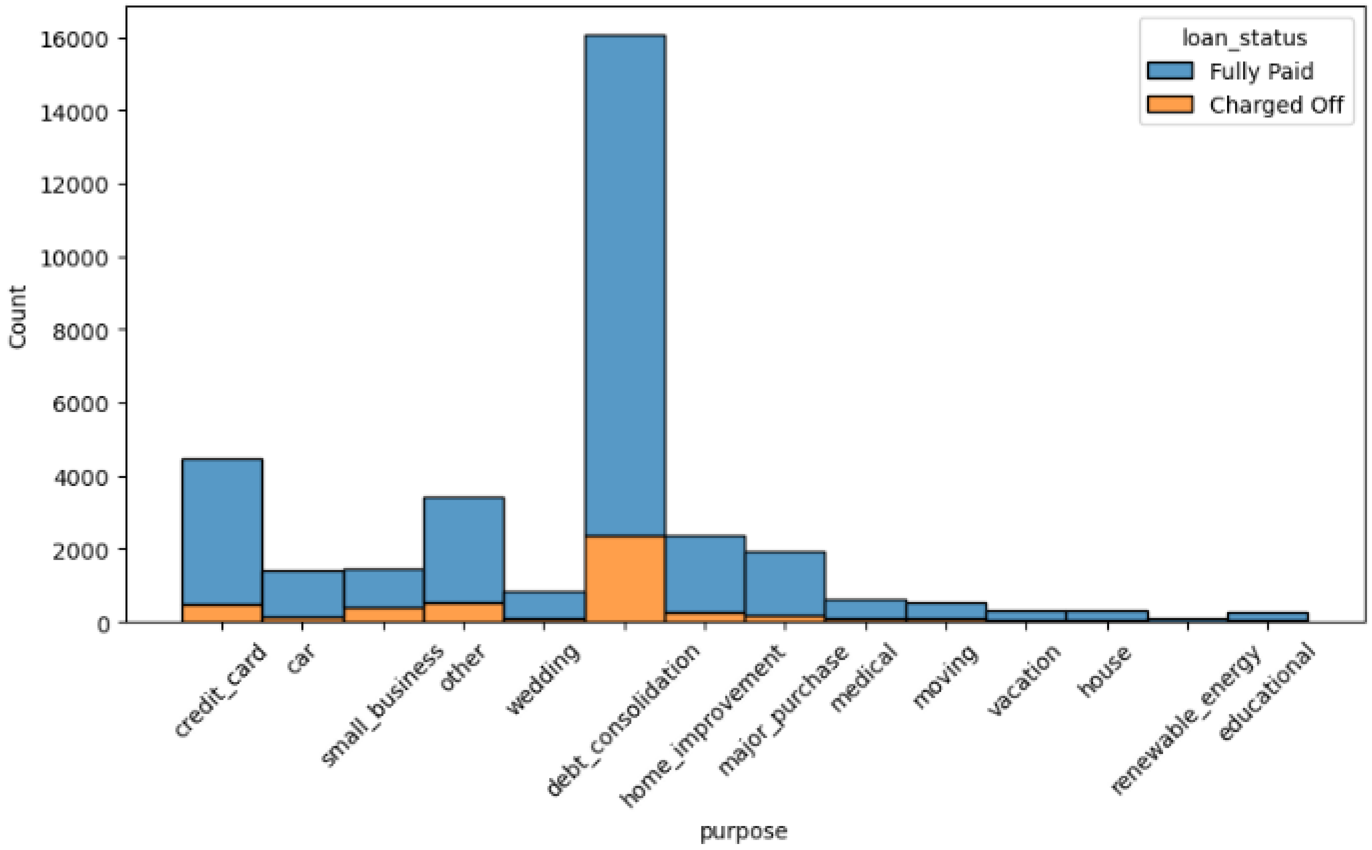
]: <seaborn.axisgrid.FacetGrid at 0x1daecbe7090>



Bivariate Analysis

Loan purpose versus Loan Status analysis:
- people with loan purpose has ‘debt consolidation’ are more prone to be defaulters

```
plt.figure(figsize=(10,5))
ax  =  sns.histplot(data=loan_cleaned_data_1, x="purpose" , hue='loan_status',  multiple='stack')
ax.tick_params(axis='x', labelrotation=45)
plt.show()
#debt consolidation seems to be the maximum purpose for taking loan and has more defaulters
```

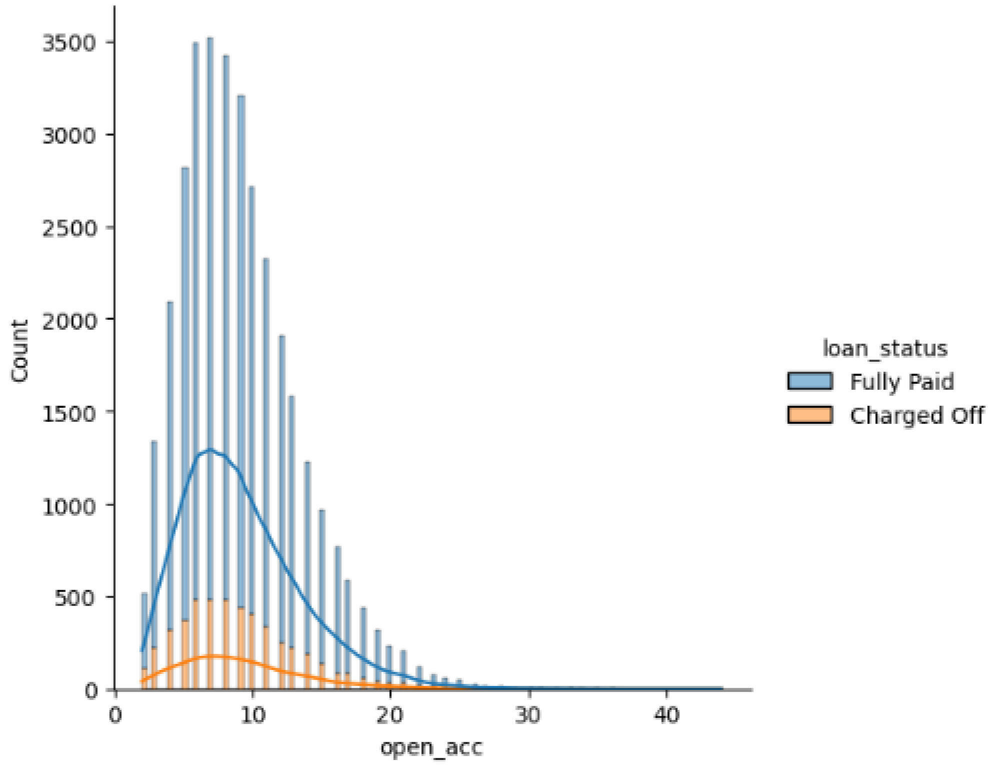


Bivariate Analysis

Open Accounts versus Loan Status analysis:
- Inference: customers with open accounts between 5 and 10 have more defaulters

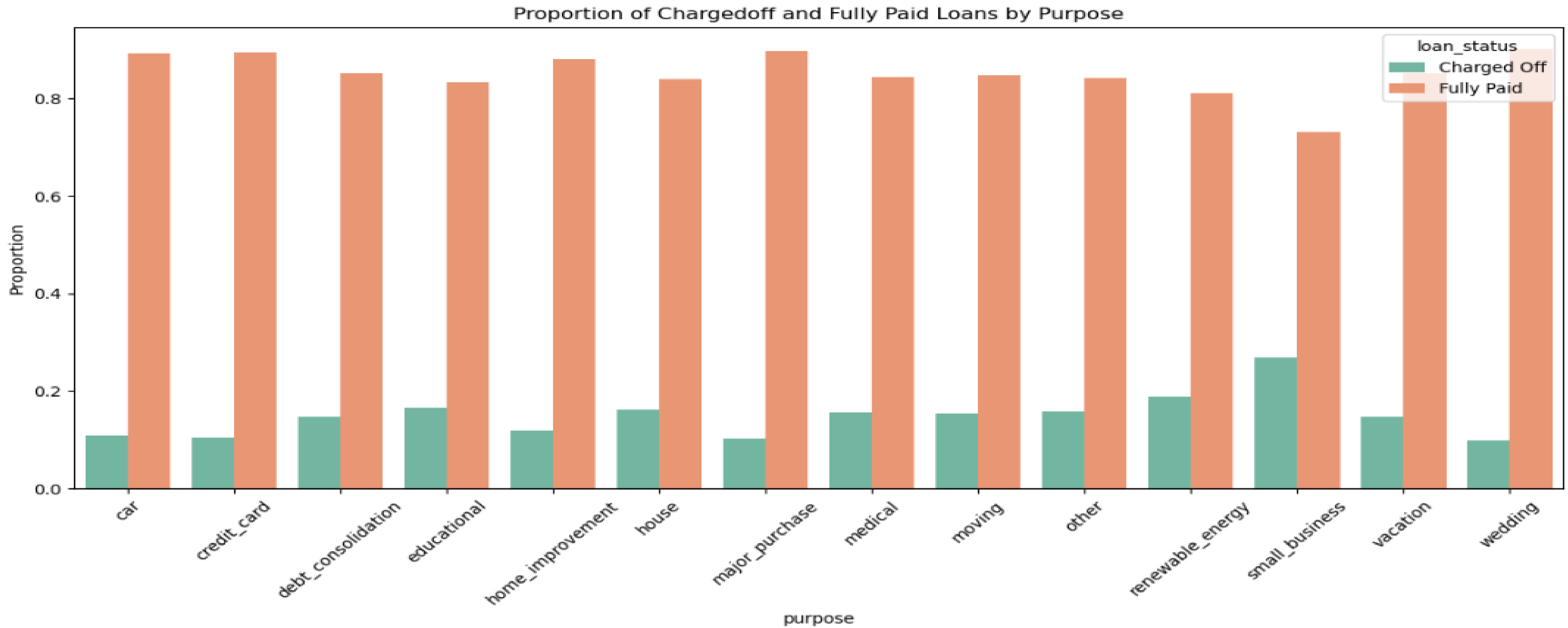
In [351]:
```
sns.displot(data=loan_cleaned_data_1, x="open_acc" , hue='loan_status',  multiple='stack', kde=True)
# customers with open accounts between 5 and 10 have more defaulters
```

Out[351]:
```
<seaborn.axisgrid.FacetGrid at 0x1daed463010>
```



Bivariate Analysis

Loan purpose versus Loan Status analysis:
- Inference: Loans with purpose as ‘Small business’ are more prone to be defaulters



Inferences

- Customers with dti between 8 and 20 are more likely to default
- Borrowers with annual income between 30k to 65k are more likely to default and higher annual income are less likely to default.
- Grade and Interest rates are correlated. Higher number of loans have been given to Grade A and Grade B, and the percentage of defaulters increases with grade A to G. Higher grade will have higher interest rates and the percentage of defaulters increases with grade A to G.
- customers with open accounts between 5 and 10 have more defaulters
- Loan amounts, Funded amount are highly correlated and people will less than 5k loan amount are less prone to be defaulters
- Debt consolidation seems to be the maximum purpose for taking loan and has more defaulters but if we look at proportion, Loans with purpose as ‘Small business’ are more prone to be defaulters