Importing Libraries

```
import pandas as pd
import numpy as np

import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
%matplotlib inline

pd.set_option('display.max_columns', None)

from scipy import stats
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import (accuracy_score, confusion_matrix,
                             roc_curve, auc, ConfusionMatrixDisplay,
                             f1_score, recall_score,
                             precision_score, precision_recall_curve,
                             average_precision_score, classification_report)
from statsmodels.stats.outliers_influence import variance_inflation_factor
from imblearn.over_sampling import SMOTE

import warnings
warnings.filterwarnings("ignore")


df = pd.read_csv('https://drive.usercontent.google.com/download?id=1ZPYj7CZCfxntE8p2Lze_4
df.head()
```

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_title	emp_length
0	10000.0	36 months	11.44	329.48	B	B4	Marketing	10+ years
1	8000.0	36 months	11.99	265.68	B	B5	Credit analyst	4 years
2	15600.0	36 months	10.49	506.97	B	B3	Statistician	< 1 year
3	7200.0	36 months	6.49	220.65	A	A2	Client Advocate	6 years
4	24375.0	60 months	17.27	609.33	C	C5	Destiny Management Inc.	9 years

```
df.shape
```

```
(396030, 27)
```

```
df.describe()
```

	loan_amnt	int_rate	installment	annual_inc	dti	o
count	396030.000000	396030.000000	396030.000000	3.960300e+05	396030.000000	396030
mean	14113.888089	13.639400	431.849698	7.420318e+04	17.379514	1
std	8357.441341	4.472157	250.727790	6.163762e+04	18.019092	5
min	500.000000	5.320000	16.080000	0.000000e+00	0.000000	0
25%	8000.000000	10.490000	250.330000	4.500000e+04	11.280000	8
50%	12000.000000	13.330000	375.430000	6.400000e+04	16.910000	10
75%	20000.000000	16.490000	567.300000	9.000000e+04	22.980000	14
max	40000.000000	30.990000	1533.810000	8.706582e+06	9999.000000	90

1. Nearly 80% of the loans have a term of 36 months.
2. The majority of loans (30%) are graded as B, followed by C, A, and D respectively.
3. For 50% of cases, the type of home ownership is mortgage.
4. The loan status target variable is biased towards fully-paid loans, with defaulters accounting for approximately 25% of fully-paid instances.

5. Approximately 85% of applicants do not have a public record or have not filed for bankruptcy.
6. Nearly all applicants (99%) have applied under the 'individual' application type. -The most common purpose for taking out loans is debt consolidation, accounting for 55%, followed by 20% for credit card purposes.

Data Cleaning

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 396030 entries, 0 to 396029
Data columns (total 27 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   loan_amnt             396030 non-null  float64
 1   term                  396030 non-null  object 
 2   int_rate              396030 non-null  float64
 3   installment           396030 non-null  float64
 4   grade                 396030 non-null  object 
 5   sub_grade             396030 non-null  object 
 6   emp_title             373103 non-null  object 
 7   emp_length            377729 non-null  object 
 8   home_ownership        396030 non-null  object 
 9   annual_inc            396030 non-null  float64
 10  verification_status   396030 non-null  object 
 11  issue_d               396030 non-null  object 
 12  loan_status           396030 non-null  object 
 13  purpose               396030 non-null  object 
 14  title                 394274 non-null  object 
 15  dti                   396030 non-null  float64
 16  earliest_cr_line      396030 non-null  object 
 17  open_acc              396030 non-null  float64
 18  pub_rec               396030 non-null  float64
 19  revol_bal             396030 non-null  float64
 20  revol_util            395754 non-null  float64
 21  total_acc             396030 non-null  float64
 22  initial_list_status   396030 non-null  object 
 23  application_type      396030 non-null  object 
 24  mort_acc              358235 non-null  float64
 25  pub_rec_bankruptcies  395495 non-null  float64
 26  address               396030 non-null  object 
dtypes: float64(12), object(15)
memory usage: 81.6+ MB
```

Checking Column Datatypes

```
cat_cols = df.select_dtypes(include=['object']).columns
cat_cols
```

```
Index(['term', 'grade', 'sub_grade', 'emp_title', 'emp_length',
       'home_ownership', 'verification_status', 'issue_d', 'loan_status',
```

```
             'purpose', 'title', 'earliest_cr_line', 'initial_list_status',
             'application_type', 'address'],
            dtype='object')
```

```
for col in cat_cols:
    print(f"No of unique values in {col}: {df[col].nunique()}")
```

```
No of unique values in term: 2
No of unique values in grade: 7
No of unique values in sub_grade: 35
No of unique values in emp_title: 173105
No of unique values in emp_length: 11
No of unique values in home_ownership: 6
No of unique values in verification_status: 3
No of unique values in issue_d: 115
No of unique values in loan_status: 2
No of unique values in purpose: 14
No of unique values in title: 48816
No of unique values in earliest_cr_line: 684
No of unique values in initial_list_status: 2
No of unique values in application_type: 3
No of unique values in address: 393700
```

```
df['earliest_cr_line'] = pd.to_datetime(df['earliest_cr_line'])
df['issue_d'] = pd.to_datetime(df['issue_d'])
```

```
d = {'10+ years':10, '4 years':4, '< 1 year':0, '6 years':6, '9 years':9, '7 years':7, '8
df['emp_length'] = df['emp_length'].replace(d)
```

```
cat_cols = ['term', 'grade','sub_grade','home_ownership',
            'verification_status','loan_status','purpose',
            'initial_list_status','application_type']

df[cat_cols] = df[cat_cols].astype('category')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 396030 entries, 0 to 396029
Data columns (total 27 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   loan_amnt             396030 non-null  float64
 1   term                  396030 non-null  category
 2   int_rate              396030 non-null  float64
 3   installment           396030 non-null  float64
 4   grade                 396030 non-null  category
 5   sub_grade             396030 non-null  category
 6   emp_title             373103 non-null  object
 7   emp_length            377729 non-null  float64
 8   home_ownership        396030 non-null  category
 9   annual_inc            396030 non-null  float64
 10  verification_status   396030 non-null  category
```

```
 11  issue_d               396030 non-null  datetime64[ns]
 12  loan_status           396030 non-null  category
 13  purpose               396030 non-null  category
 14  title                 394274 non-null  object
 15  dti                   396030 non-null  float64
 16  earliest_cr_line      396030 non-null  datetime64[ns]
 17  open_acc              396030 non-null  float64
 18  pub_rec               396030 non-null  float64
 19  revol_bal             396030 non-null  float64
 20  revol_util            395754 non-null  float64
 21  total_acc             396030 non-null  float64
 22  initial_list_status   396030 non-null  category
 23  application_type      396030 non-null  category
 24  mort_acc              358235 non-null  float64
 25  pub_rec_bankruptcies  395495 non-null  float64
 26  address               396030 non-null  object
dtypes: category(9), datetime64[ns](2), float64(13), object(3)
memory usage: 57.8+ MB
```

Check for Duplicate Values

```
df.duplicated().sum()
```

```
0
```

Handling Missing Values

```
df.isna().sum()
```

```
loan_amnt                 0
term                      0
int_rate                  0
installment               0
grade                     0
sub_grade                 0
emp_title             22927
emp_length            18301
home_ownership            0
annual_inc                0
verification_status       0
issue_d                   0
loan_status               0
purpose                   0
title                  1756
dti                       0
earliest_cr_line          0
open_acc                  0
pub_rec                   0
revol_bal                 0
revol_util              276
total_acc                 0
initial_list_status       0
application_type          0
mort_acc              37795
pub_rec_bankruptcies    535
```

```
address                   0
dtype: int64
```

```
fill_values = {'title': 'Unknown', 'emp_title': 'Unknown'}
df.fillna(value=fill_values, inplace=True)
```

```
df['mort_acc'].fillna(df.groupby('total_acc')['mort_acc'].transform('mean'), inplace=True
```

```
df.isna().sum()
```

```
loan_amnt                 0
term                      0
int_rate                  0
installment               0
grade                     0
sub_grade                 0
emp_title                 0
emp_length            18301
home_ownership            0
annual_inc                0
verification_status       0
issue_d                   0
loan_status               0
purpose                   0
title                     0
dti                       0
earliest_cr_line          0
open_acc                  0
pub_rec                   0
revol_bal                 0
revol_util              276
total_acc                 0
initial_list_status       0
application_type          0
mort_acc                  0
pub_rec_bankruptcies    535
address                   0
dtype: int64
```

```
from sklearn.impute import KNNImputer
knn_imputer = KNNImputer(n_neighbors=5)
df['emp_length'] = knn_imputer.fit_transform(df['emp_length'].values.reshape(-1, 1)).rave
df['revol_util'] = knn_imputer.fit_transform(df['revol_util'].values.reshape(-1, 1)).rave
df['pub_rec_bankruptcies'] = knn_imputer.fit_transform(df['pub_rec_bankruptcies'].values.
```

```
df.isna().sum()
```

```
loan_amnt             0
term                  0
int_rate              0
installment           0
grade                 0
sub_grade             0
emp_title             0
emp_length            0
```

```
home_ownership          0
annual_inc              0
verification_status     0
issue_d                 0
loan_status             0
purpose                 0
title                   0
dti                     0
earliest_cr_line        0
open_acc                0
pub_rec                 0
revol_bal               0
revol_util              0
total_acc               0
initial_list_status     0
application_type        0
mort_acc                0
pub_rec_bankruptcies    0
address                 0
dtype: int64
```

```
df.shape
```

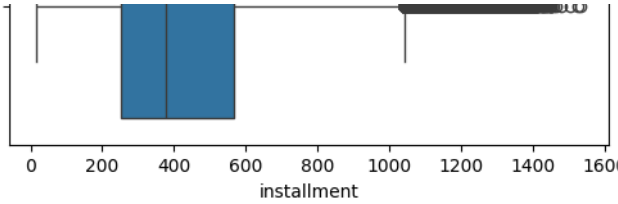
```
(396030, 27)
```

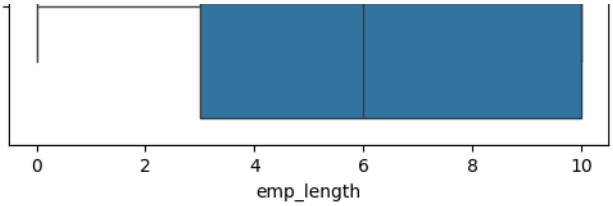
Outlier Treatment

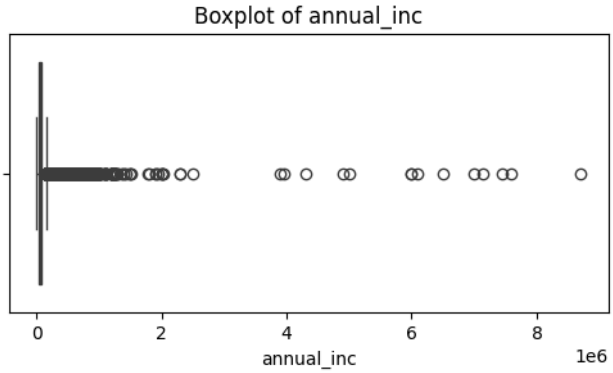
```
num_cols = df.select_dtypes(include='number').columns
num_cols
```

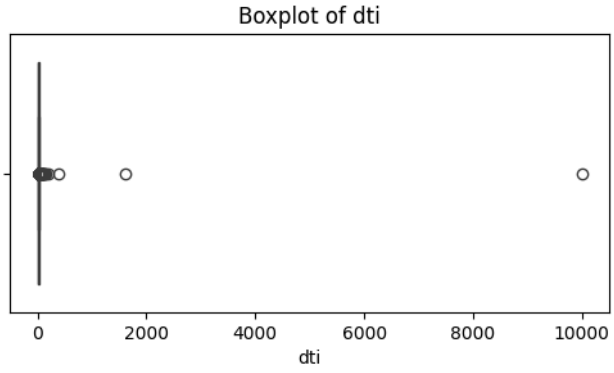
```
Index(['loan_amnt', 'int_rate', 'installment', 'emp_length', 'annual_inc',
       'dti', 'open_acc', 'pub_rec', 'revol_bal', 'revol_util', 'total_acc',
       'mort_acc', 'pub_rec_bankruptcies'],
      dtype='object')
```

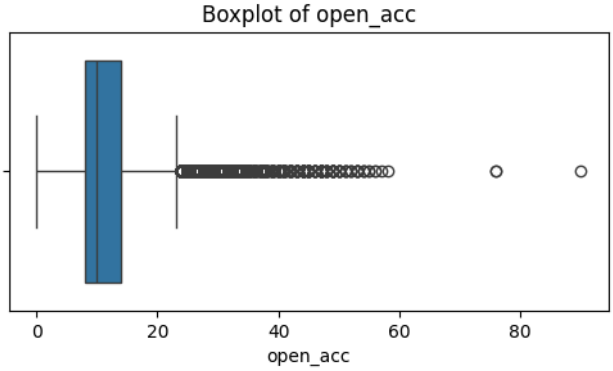
```
fig = plt.figure(figsize=(10,21))
i=1
for col in num_cols:
  ax = plt.subplot(7,2,i)
  sns.boxplot(x=df[col])
  plt.title(f'Boxplot of {col}')
  i += 1

plt.tight_layout()
plt.show()
```

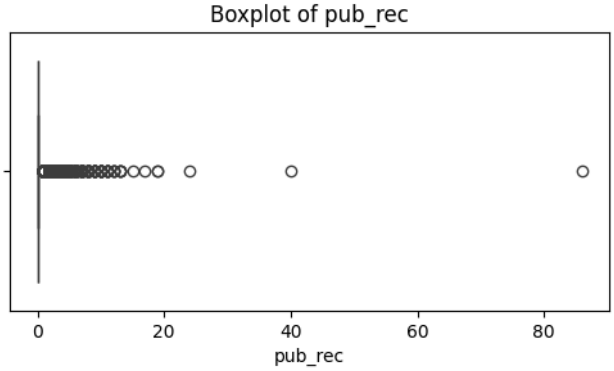

0
200
400
600
800
1000
1200
1400
1600
installment

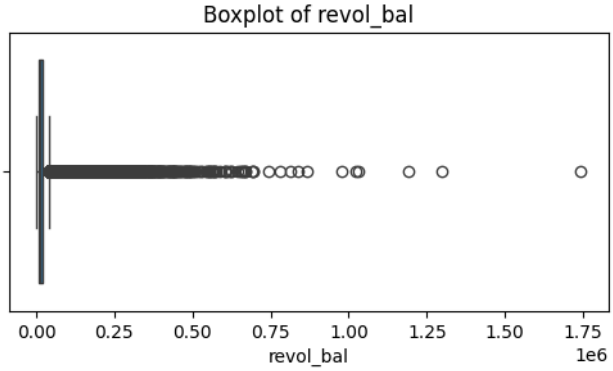

0
2
4
6
8
10
emp_length


Boxplot of annual_inc
0
2
4
6
8
annual_inc
1e6

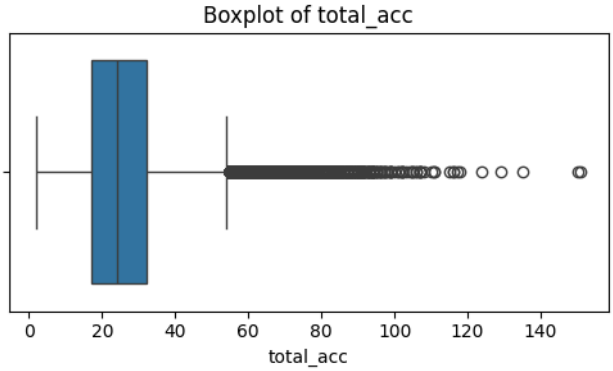

Boxplot of dti
0
2000
4000
6000
8000
10000
dti

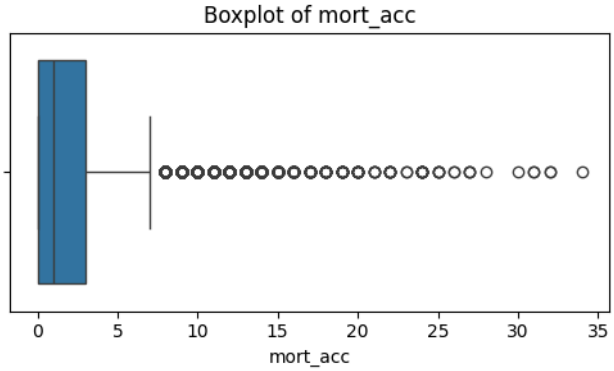

Boxplot of open_acc
0
20
40
60
80
open_acc

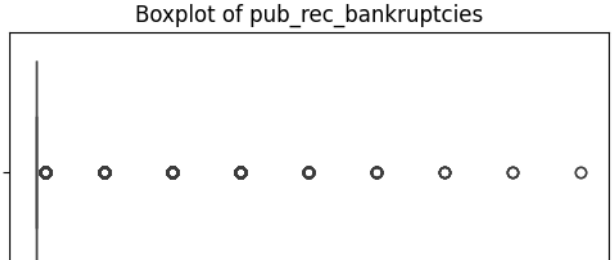

Boxplot of pub_rec
0
20
40
60
80
pub_rec


Boxplot of revol_bal
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
revol_bal
1e6


Boxplot of revol_util
0
200
400
600
800
revol_util


Boxplot of total_acc
0
20
40
60
80
100
120
140
total_acc


Boxplot of mort_acc
0
5
10
15
20
25
30
35
mort_acc


Boxplot of pub_rec_bankruptcies

```
df['pub_rec_bankruptcies'] = np.where(df['pub_rec_bankruptcies']>0,'yes','no')
df['pub_rec'] = np.where(df['pub_rec']>0,'yes','no')
df[['pub_rec_bankruptcies','pub_rec']] = df[['pub_rec_bankruptcies','pub_rec']].astype('c
```

```
num_cols = df.select_dtypes(include='number').columns
num_cols
```

```
Index(['loan_amnt', 'int_rate', 'installment', 'emp_length', 'annual_inc',
       'dti', 'open_acc', 'revol_bal', 'revol_util', 'total_acc', 'mort_acc'],
      dtype='object')
```

```
for col in num_cols:
  mean = df[col].mean()
  std = df[col].std()
  upper = mean + (3*std)
  df = df[~(df[col] > upper)]
```

```
df.shape
```

```
(368778, 27)
```

Feature Engineering

```
df['address'].sample(10)
```

```
352370         00077 Jason Lock\r\nVelasquezville, NM 70466
1076      146 Jacob Junction Suite 603\r\nCynthiafort, K...
289831       7018 Newman Vista\r\nWest Brianhaven, WI 22690
22417          753 Caitlin Vista\r\nMackenzietown, NM 29597
226175    9489 Cynthia Groves Suite 704\r\nThomasshire, ...
298743                   PSC 2135, Box 2351\r\nAPO AA 05113
262229                         USS Williams\r\nFPO AP 29597
253339                           USNV Lewis\r\nFPO AP 29597
291914    724 Thomas Hill Apt. 320\r\nAustinhaven, GA 05113
112117    246 Emily Crossroad Apt. 926\r\nSouth Austinha...
Name: address, dtype: object
```

```
df[['state', 'zip_code']] = df['address'].apply(lambda x: pd.Series([x[-8:-6], x[-5:]]))
```

```
df.drop(["address"], axis = 1, inplace=True)
```

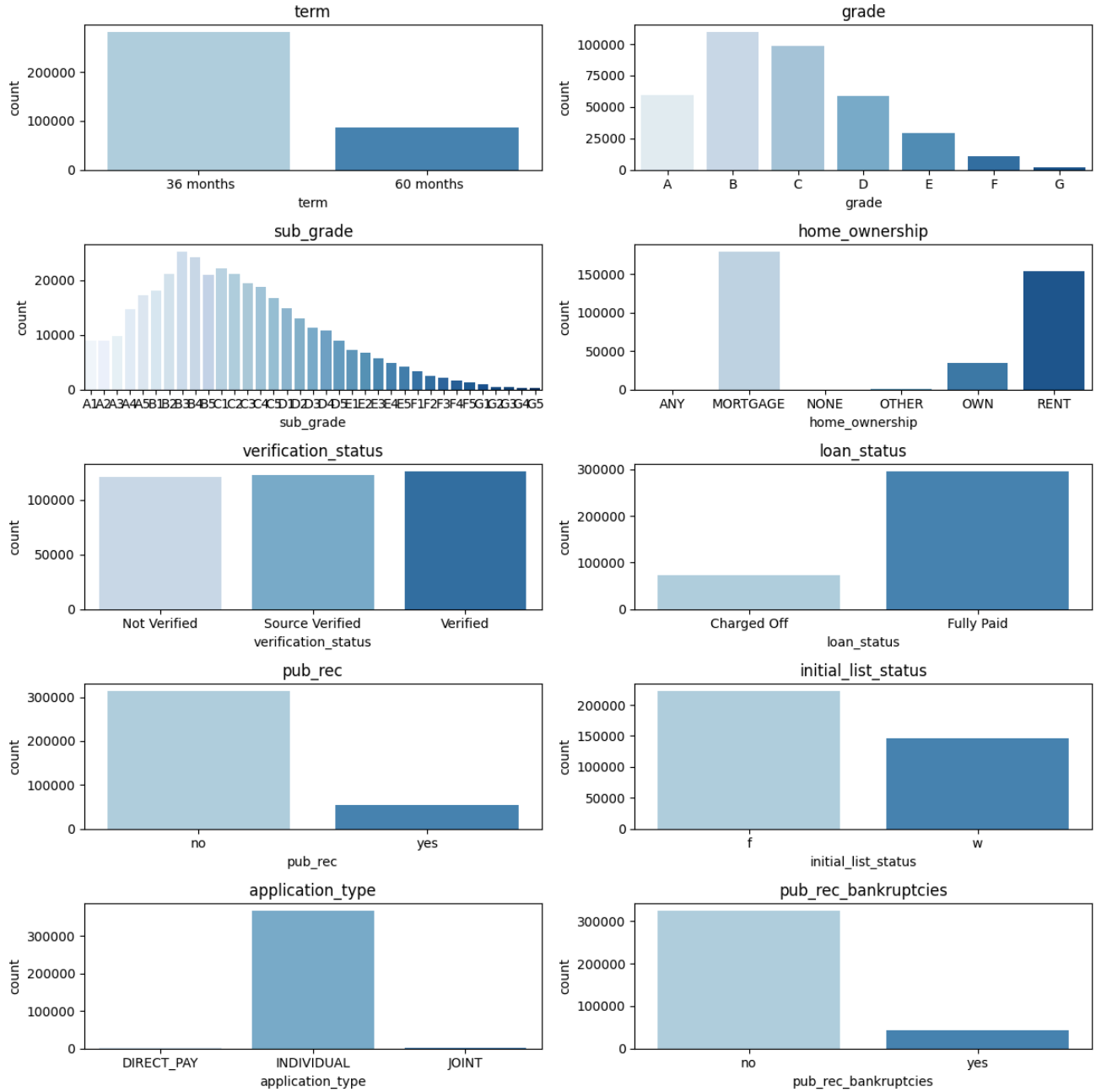
```
df.zip_code.nunique()
```

```
10
```

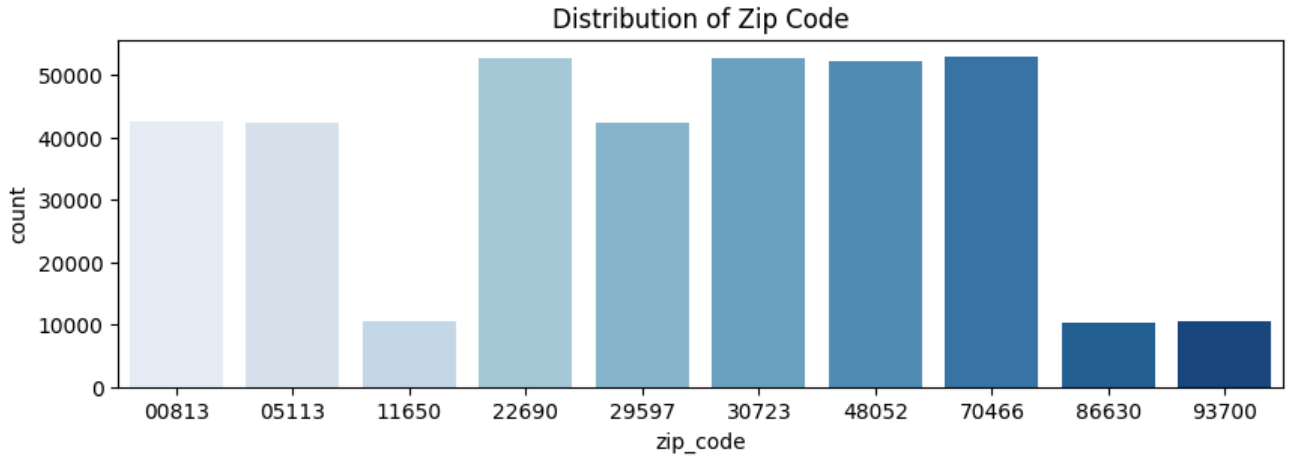
```
df['zip_code'] = df['zip_code'].astype('category')
```

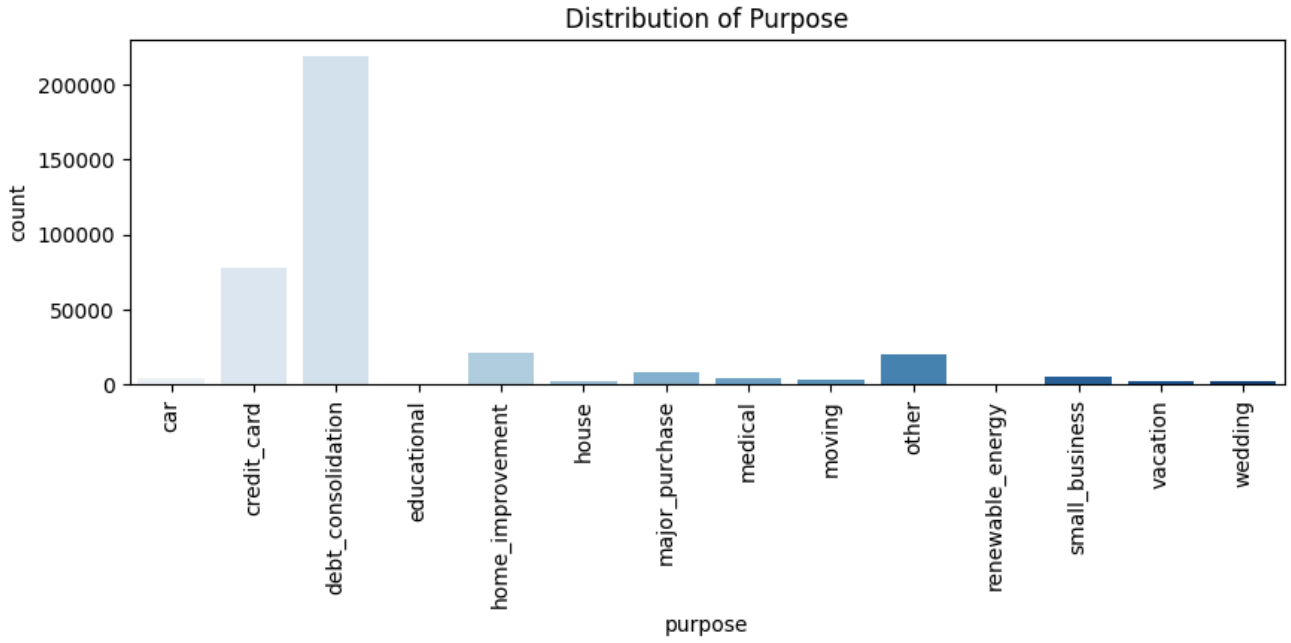
Exploratory Data Analysis

```
df.drop(columns=['installment'], inplace=True)


plot = ['term', 'grade', 'sub_grade', 'home_ownership', 'verification_status',
       'loan_status', 'pub_rec', 'initial_list_status',
       'application_type', 'pub_rec_bankruptcies']

plt.figure(figsize=(12,12))
i=1
for col in plot:
  ax=plt.subplot(5,2,i)
  sns.countplot(x=df[col], palette='Blues')
  plt.title(f'{col}')
  i += 1

plt.tight_layout()
plt.show()
```

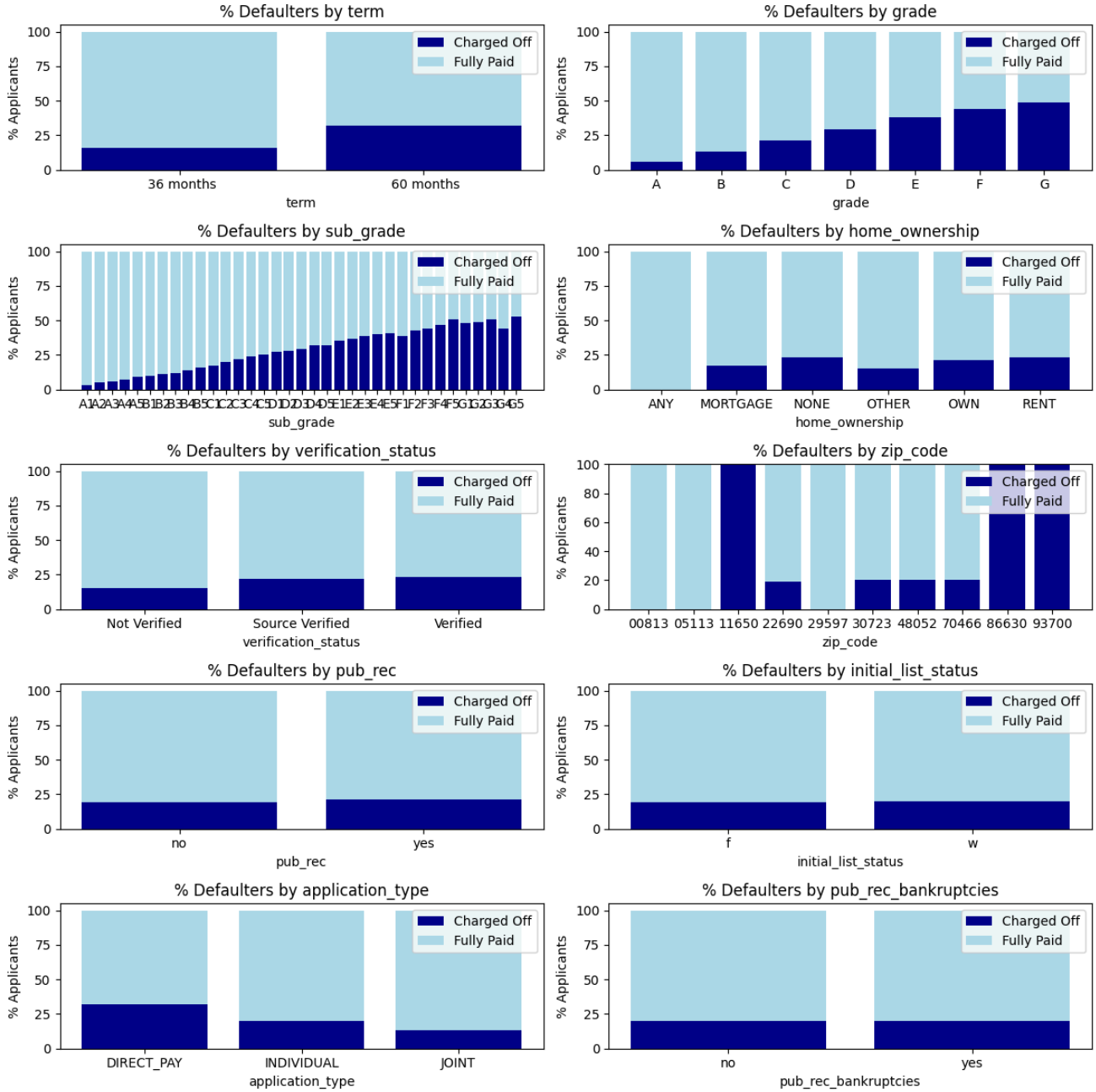

term
count
200000
100000
0
36 months
60 months
term
grade
count
100000
75000
50000
25000
0
A
B
C
D
E
F
G
grade
sub_grade
count
20000
10000
0
A1 A2 A3 A4 A5 B1 B2 B3 B4 B5 C1 C2 C3 C4 C5 D1 D2 D3 D4 D5 E1 E2 E3 E4 E5 F1 F2 F3 F4 F5 G1 G2 G3 G4 G5
sub_grade
home_ownership
count
150000
100000
50000
0
ANY
MORTGAGE
NONE
OTHER
OWN
RENT
home_ownership
verification_status
count
100000
50000
0
Not Verified
Source Verified
Verified
verification_status
loan_status
count
300000
200000
100000
0
Charged Off
Fully Paid
loan_status
pub_rec
count
300000
200000
100000
0
no
yes
pub_rec
initial_list_status
count
200000
150000
100000
50000
0
f
w
initial_list_status
application_type
count
300000
200000
100000
0
DIRECT_PAY
INDIVIDUAL
JOINT
application_type
pub_rec_bankruptcies
count
300000
200000
100000
0
no
yes
pub_rec_bankruptcies

```
plt.figure(figsize=(10,3))
sns.countplot(x=df['zip_code'], palette='Blues')
plt.title('Distribution of Zip Code')

plt.figure(figsize=(10,3))
sns.countplot(x=df['purpose'], palette='Blues')
plt.xticks(rotation=90)
plt.title('Distribution of Purpose')

plt.show()
```

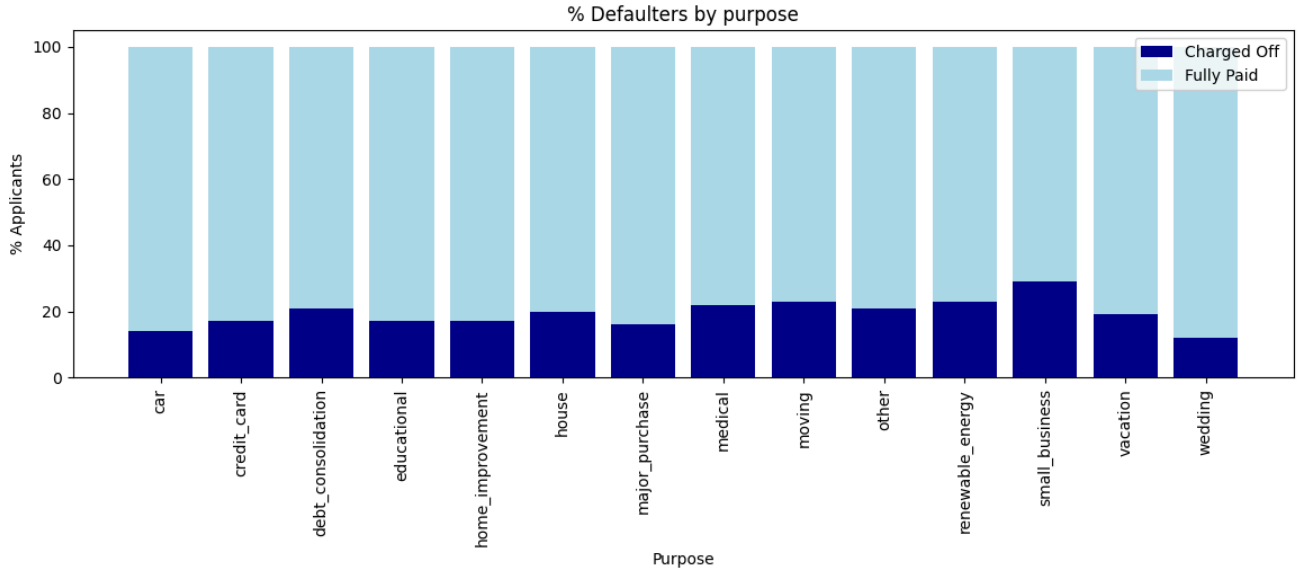

Distribution of Zip Code
count
50000
40000
30000
20000
10000
0
00813
05113
11650
22690
29597
30723
48052
70466
86630
93700
zip_code

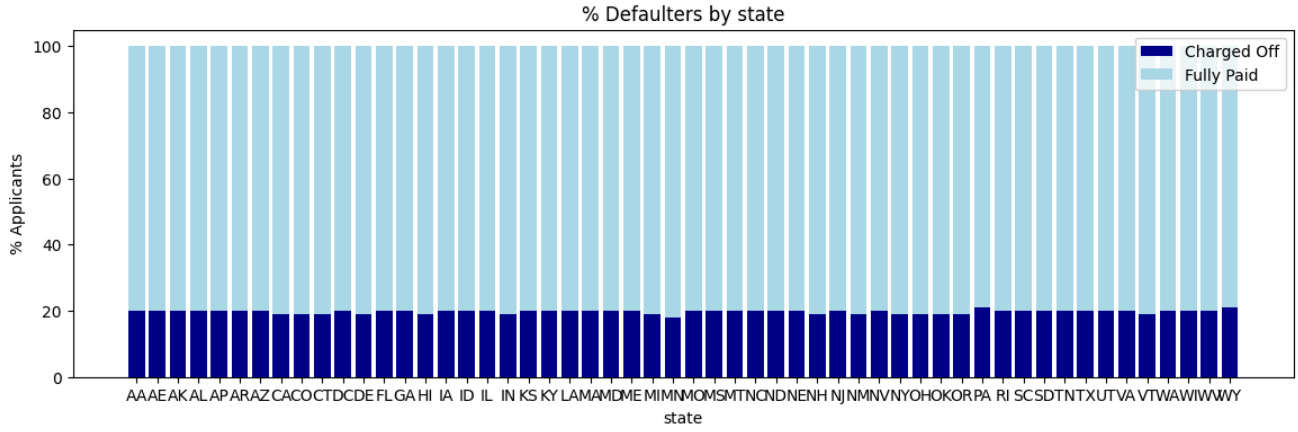

Distribution of Purpose
count
200000
150000
100000
50000
0
car
credit_card
debt_consolidation
educational
home_improvement
house
major_purchase
medical
moving
other
renewable_energy
small_business
vacation
wedding
purpose

```
plot = ['term', 'grade', 'sub_grade', 'home_ownership', 'verification_status',
       'zip_code', 'pub_rec', 'initial_list_status',
       'application_type', 'pub_rec_bankruptcies']

plt.figure(figsize=(12,12))
i=1
for col in plot:
  ax=plt.subplot(5,2,i)

  data = pd.crosstab(df[col], df['loan_status'], normalize='index').round(2)*100
  data.reset_index(inplace=True)

  plt.bar(data[col],data['Charged Off'], color='#00008b')
  plt.bar(data[col],data['Fully Paid'], color='#add8e6', bottom=data['Charged Off'])
  plt.xlabel(f'{col}')
  plt.ylabel('% Applicants')
  plt.title(f'% Defaulters by {col}')
  plt.legend(['Charged Off','Fully Paid'])
  i += 1

plt.tight_layout()
plt.show()
```

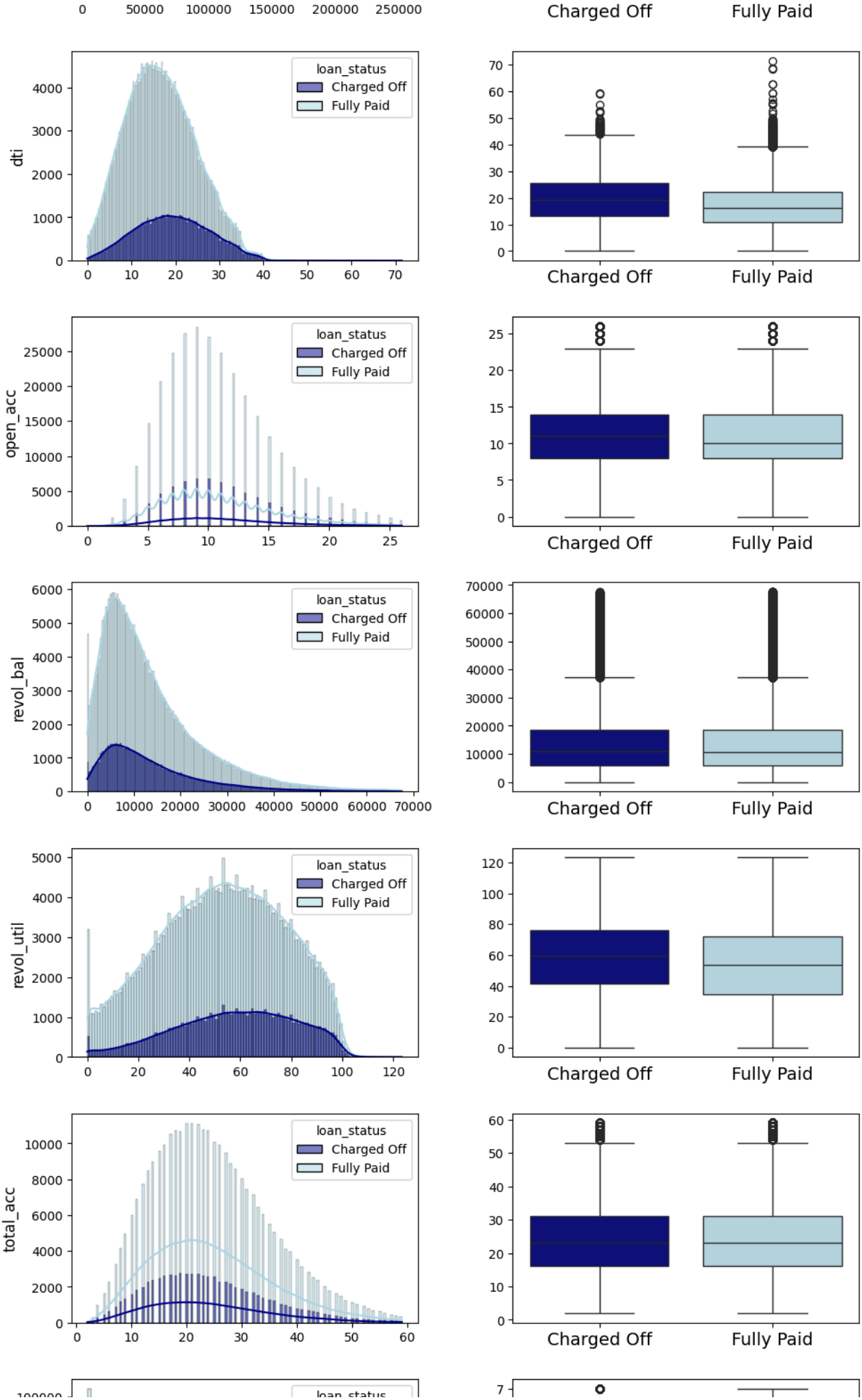

% Defaulters by term
Charged Off
Fully Paid
% Applicants
36 months
60 months
term
% Defaulters by grade
A
B
C
D
E
F
G
grade
% Defaulters by sub_grade
sub_grade
% Defaulters by home_ownership
ANY
MORTGAGE
NONE
OTHER
OWN
RENT
home_ownership
% Defaulters by verification_status
Not Verified
Source Verified
Verified
verification_status
% Defaulters by zip_code
00813 05113 11650 22690 29597 30723 48052 70466 86630 93700
zip_code
% Defaulters by pub_rec
no
yes
pub_rec
% Defaulters by initial_list_status
f
w
initial_list_status
% Defaulters by application_type
DIRECT_PAY
INDIVIDUAL
JOINT
application_type
% Defaulters by pub_rec_bankruptcies
pub_rec_bankruptcies

```
purpose = pd.crosstab(df['purpose'], df['loan_status'], normalize='index').round(2)*100
purpose.reset_index(inplace=True)

plt.figure(figsize=(14,4))
plt.bar(purpose['purpose'],purpose['Charged Off'], color='#00008b')
plt.bar(purpose['purpose'],purpose['Fully Paid'], color='#add8e6', bottom=purpose['Charge
plt.xlabel('Purpose')
plt.ylabel('% Applicants')
plt.title('% Defaulters by purpose')
plt.legend(['Charged Off','Fully Paid'])
plt.xticks(rotation=90)
plt.show()

state = pd.crosstab(df['state'], df['loan_status'], normalize='index').round(2)*100
state.reset_index(inplace=True)

plt.figure(figsize=(14,4))
plt.bar(state['state'],state['Charged Off'], color='#00008b')
plt.bar(state['state'],state['Fully Paid'], color='#add8e6', bottom=state['Charged Off'])
plt.xlabel('state')
plt.ylabel('% Applicants')
plt.title('% Defaulters by state')
plt.legend(['Charged Off','Fully Paid'])
plt.show()
```


% Defaulters by purpose
Charged Off
Fully Paid
% Applicants
0
20
40
60
80
100
car
credit_card
debt_consolidation
educational
home_improvement
house
major_purchase
medical
moving
other
renewable_energy
small_business
vacation
wedding
Purpose


% Defaulters by state
Charged Off
Fully Paid
% Applicants
0
20
40
60
80
100
AA AE AK AL AP AR AZ CA CO CT DC DE FL GA HI IA ID IL IN KS KY LA MA MD ME MI MN MO MS MT NC ND NE NH NJ NM NV NY OH OK OR PA RI SC SD TN TX UT VA VT WA WI WV WY
state

```
num_cols = df.select_dtypes(include='number').columns

fig, ax = plt.subplots(10,2,figsize=(10,30))
i=0
color_dict = {'Fully Paid': matplotlib.colors.to_rgba('#add8e6', 0.5),
              'Charged Off': matplotlib.colors.to_rgba('#00008b', 1)}
for col in num_cols:
    sns.histplot(data=df, x=col, hue='loan_status', ax=ax[i, 0], legend=True,
                 palette=color_dict, kde=True, fill=True)
    sns.boxplot(data=df, y=col, x='loan_status', ax=ax[i,1],
                palette=('#00008b', '#add8e6'))
    ax[i,0].set_ylabel(col, fontsize=12)
    ax[i,0].set_xlabel(' ')
    ax[i,1].set_xlabel(' ')
    ax[i,1].set_ylabel(' ')
    ax[i,1].xaxis.set_tick_params(labelsize=14)
    i += 1

plt.tight_layout()
plt.show()
```


0 50000 100000 150000 200000 250000
Charged Off
Fully Paid
dti
open_acc
revol_bal
revol_util
total_acc
loan_status
Charged Off
Fully Paid

```
df.drop(columns=['initial_list_status','state',
                 'emp_title', 'title','earliest_cr_line',
                 'issue_d','sub_grade'], inplace=True)
```