Loan Default Prediction

SHIMIL SHIJO

Introduction

Problem Statement

- The company specializes in lending loans to urban customers.
- Two risks associated with loan approval:
  1. Loss of Business: If the applicant is likely to repay, not approving the loan results in a loss of business.
  2. Financial Loss: If the applicant is likely to default, approving the loan may lead to financial loss.

Objective

- Identify patterns that indicate if a person is likely to default on a loan.

Actions based on Insights

- Accept the loan - Fully Paid / Current / Charged Off
- Reject the loan.

Dataset

Data Source:

- Loan Dataset(training and testing)
- Target Variable - loan_status

Data Type:

- The data includes a mix of categorical and numerical values

Variables:

- The dataset includes variables such as 'loan_amnt', 'term', 'int_rate', 'installment', 'grade', 'sub_grade', 'emp_title', 'emp_length', 'home_ownership', 'annual_inc', 'verification_status', 'issue_d', 'loan_status', 'purpose', 'title', 'dti', erliest_cr_line', 'open_acc', 'pub_rec', 'revol_bal', 'revol_util', 'total_acc', 'initial_list_status', 'application_type', 'mort_acc', 'pub_rec_bankruptcies', 'address'.

Data Size:

- Train Data - 316970 Records and 28 Columns
- Test Data - 79061 Records and 27 Columns(no target field)

EDA - Dataset Information

Interpretation :
Dataset contains combination of numeric and numeric fields

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 316970 entries, 0 to 316969
Data columns (total 24 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   loan_amnt             316970 non-null  float64
 1   term                  316970 non-null  int64  
 2   int_rate              316970 non-null  float64
 3   installment           316970 non-null  float64
 4   grade                 316970 non-null  int64  
 5   sub_grade             316970 non-null  int64  
 6   emp_length            316970 non-null  int64  
 7   home_ownership        316970 non-null  int64  
 8   annual_inc            316970 non-null  float64
 9   verification_status   316970 non-null  int64  
 10  issue_d               316970 non-null  int64  
 11  loan_status           316970 non-null  object 
 12  purpose               316970 non-null  int64  
 13  dti                   316970 non-null  float64
 14  earliest_cr_line      316970 non-null  int64  
 15  open_acc              316970 non-null  float64
 16  pub_rec               316970 non-null  float64
 17  revol_bal             316970 non-null  float64
 18  revol_util            316970 non-null  float64
 19  total_acc             316970 non-null  float64
 20  initial_list_status   316970 non-null  int64  
 21  application_type      316970 non-null  int64  
 22  mort_acc              316970 non-null  float64
 23  pub_rec_bankruptcies  316970 non-null  float64
dtypes: float64(12), int64(11), object(1)
memory usage: 58.0+ MB
```

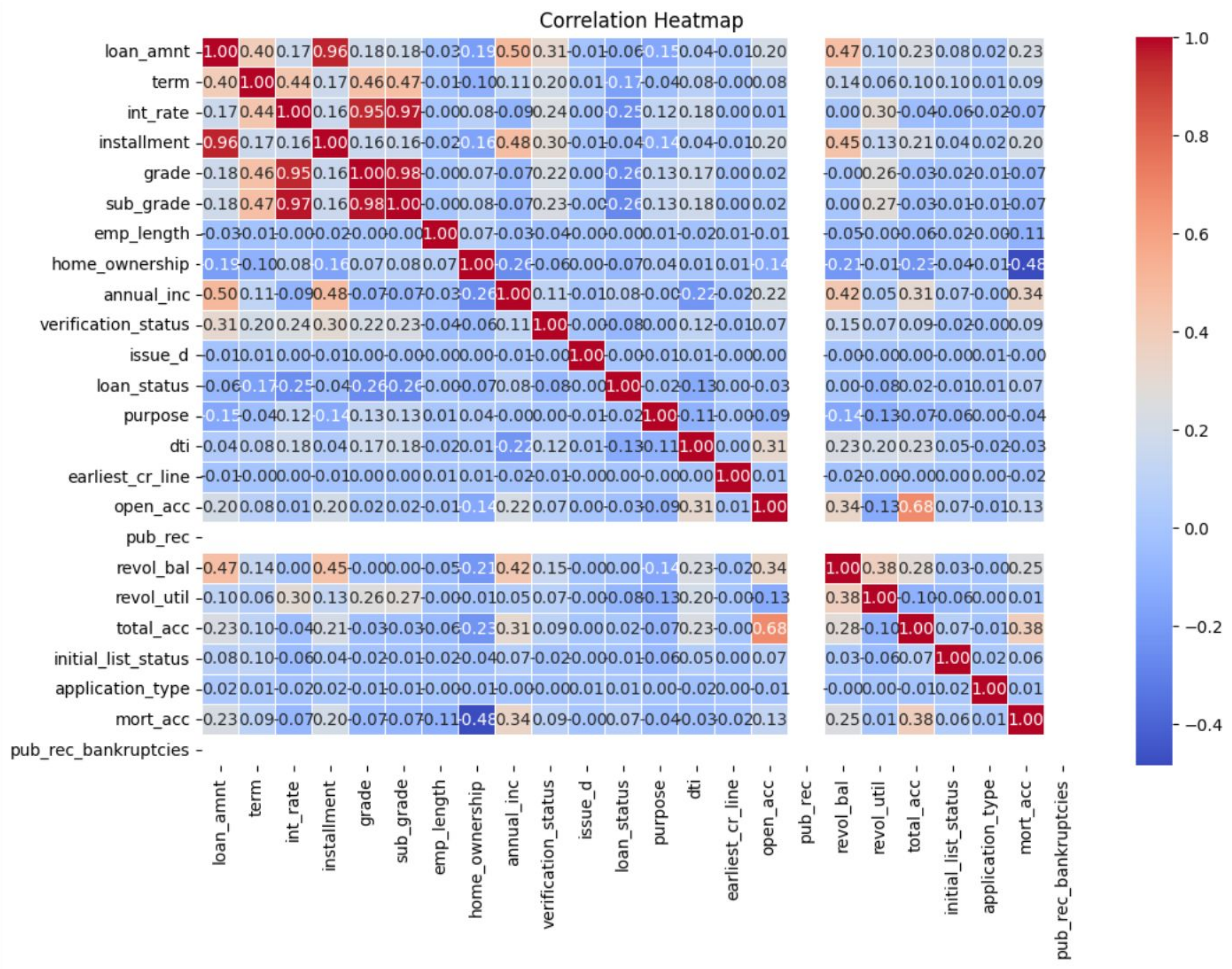
EDA - Descriptive Statistics

	loan_amnt	term	int_rate	installment	grade	sub_grade	emp_length	home_ownership	annual_inc
count	316970.000000	316970.000000	316970.000000	316970.000000	316970.000000	316970.000000	316970.000000	316970.000000	316970.000000
mean	14122.829369	0.238291	13.634503	428.424574	1.823154	11.088254	3.578121	2.900439	70988.746431
std	8354.792864	0.426039	4.454000	240.343783	1.334792	6.606825	3.157071	1.924452	34316.110309
min	500.000000	0.000000	5.320000	16.080000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	8000.000000	0.000000	10.490000	250.330000	1.000000	6.000000	1.000000	1.000000	45000.000000
50%	12000.000000	0.000000	13.330000	375.490000	2.000000	10.000000	2.000000	1.000000	64000.000000
75%	20000.000000	0.000000	16.550000	568.107500	3.000000	15.000000	6.000000	5.000000	90000.000000
max	38000.000000	1.000000	25.640000	1044.773750	6.000000	34.000000	10.000000	5.000000	157500.000000

verification_status	...	earliest_cr_line	open_acc	pub_rec	revol_bal	revol_util	total_acc	initial_list_status	application_type	mort_acc
316970.000000	...	316970.000000	316970.000000	316970.0	316970.000000	316970.000000	316970.000000	316970.000000	316970.000000	316970.000000
1.037622	...	358.121217	11.189706	0.0	14177.670685	53.797737	25.254081	0.399615	1.000375	1.703270
0.816623	...	199.283046	4.737067	0.0	10721.120735	24.397147	11.399251	0.489820	0.042738	1.928349
0.000000	...	0.000000	0.000000	0.0	0.000000	0.000000	2.000000	0.000000	0.000000	0.000000
0.000000	...	161.000000	8.000000	0.0	6028.000000	35.900000	17.000000	0.000000	1.000000	0.000000
1.000000	...	375.000000	10.000000	0.0	11183.000000	54.800000	24.000000	0.000000	1.000000	1.000000
2.000000	...	545.000000	14.000000	0.0	19639.000000	72.800000	32.000000	1.000000	1.000000	3.000000
2.000000	...	675.000000	23.000000	0.0	40055.500000	128.150000	54.500000	1.000000	2.000000	7.500000

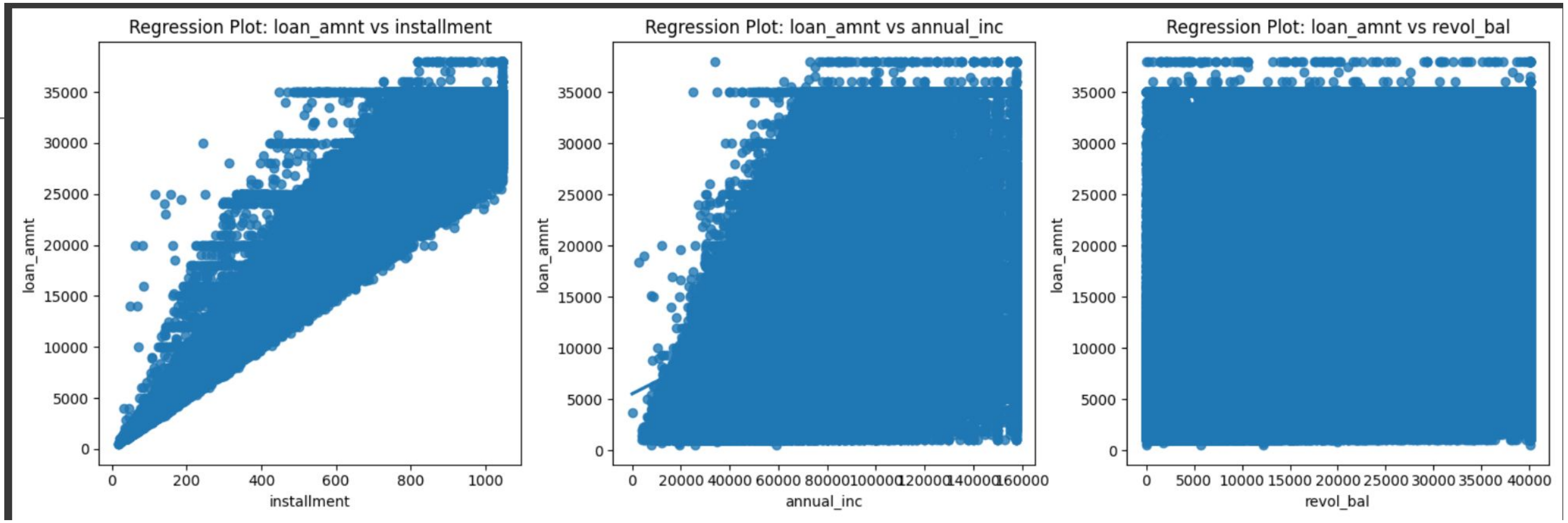
Interpretation :
Dataset need to be scaled since each field's mean values are entirely different

EDA - Heatmap



Interpretation :
Loan status is highly positively correlated to Installment and moderately positively correlated to Annual income and revol_bal

EDA - Regression Plot

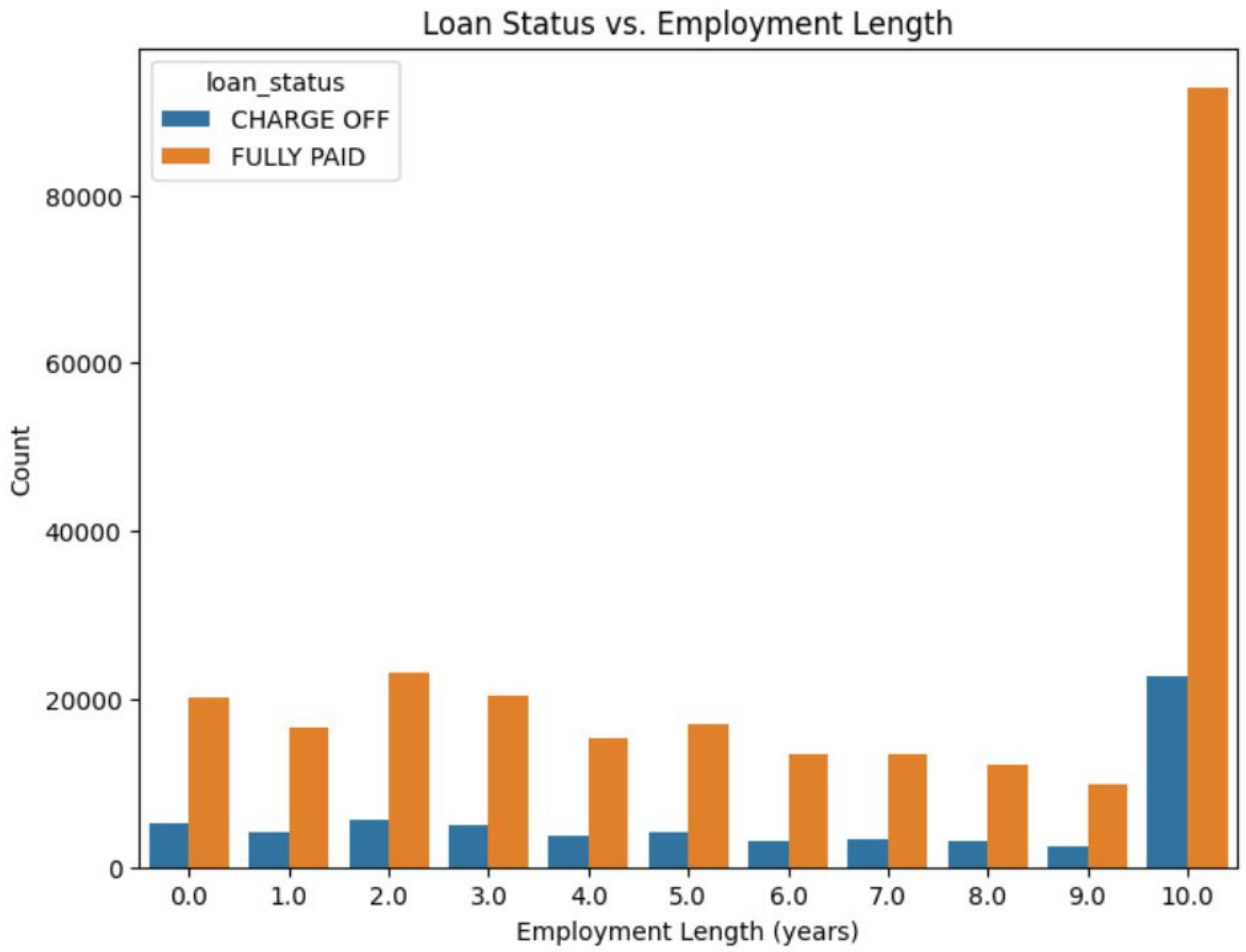


Interpretation :
Loan status is highly positively correlated to Installment and moderately positively correlated to Annual income and revol_bal

EDA - Annual Income Distribution

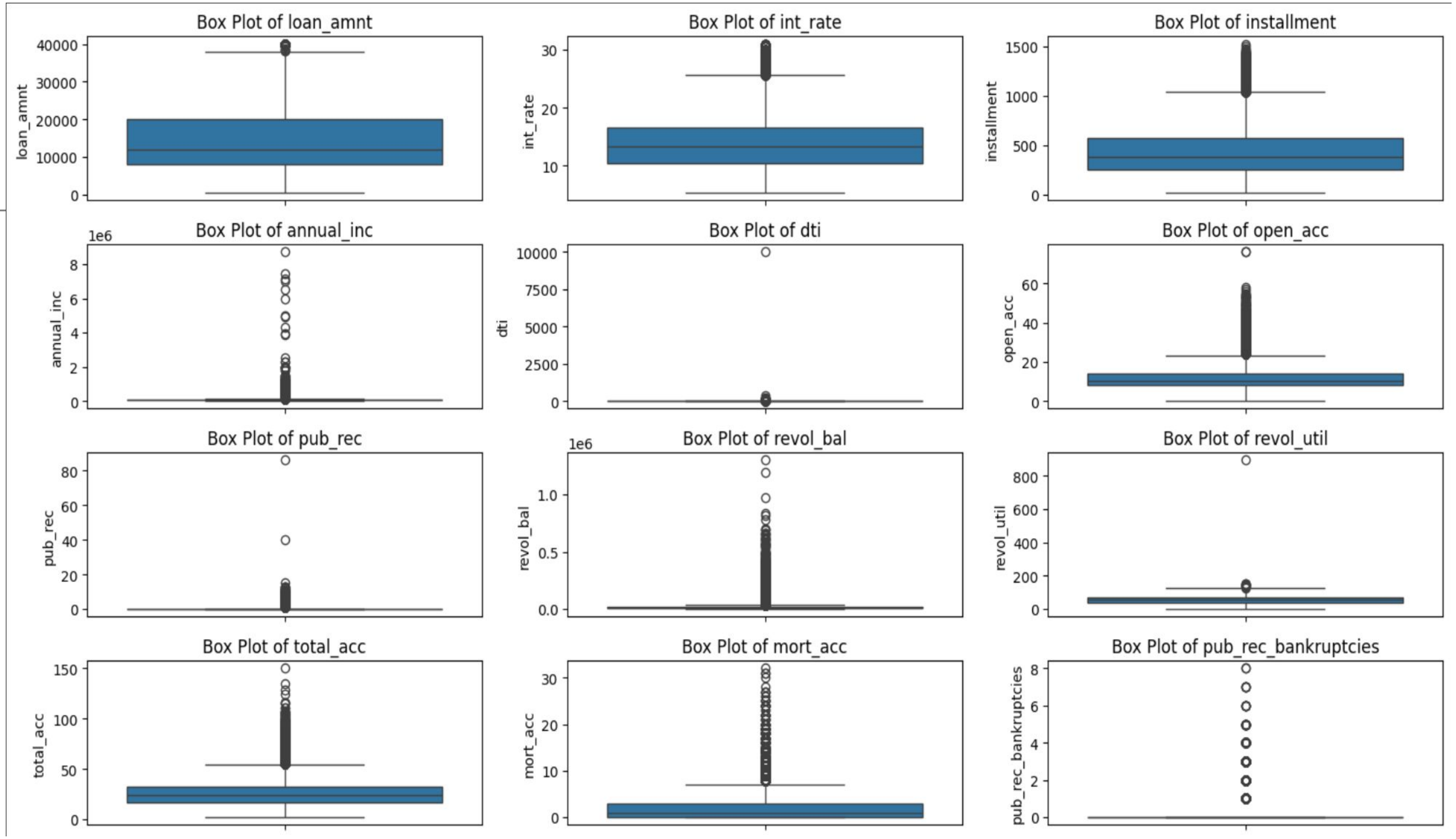
Interpretation :
Majority of annual income lies between 25000 to 75000

EDA - Bar Plot



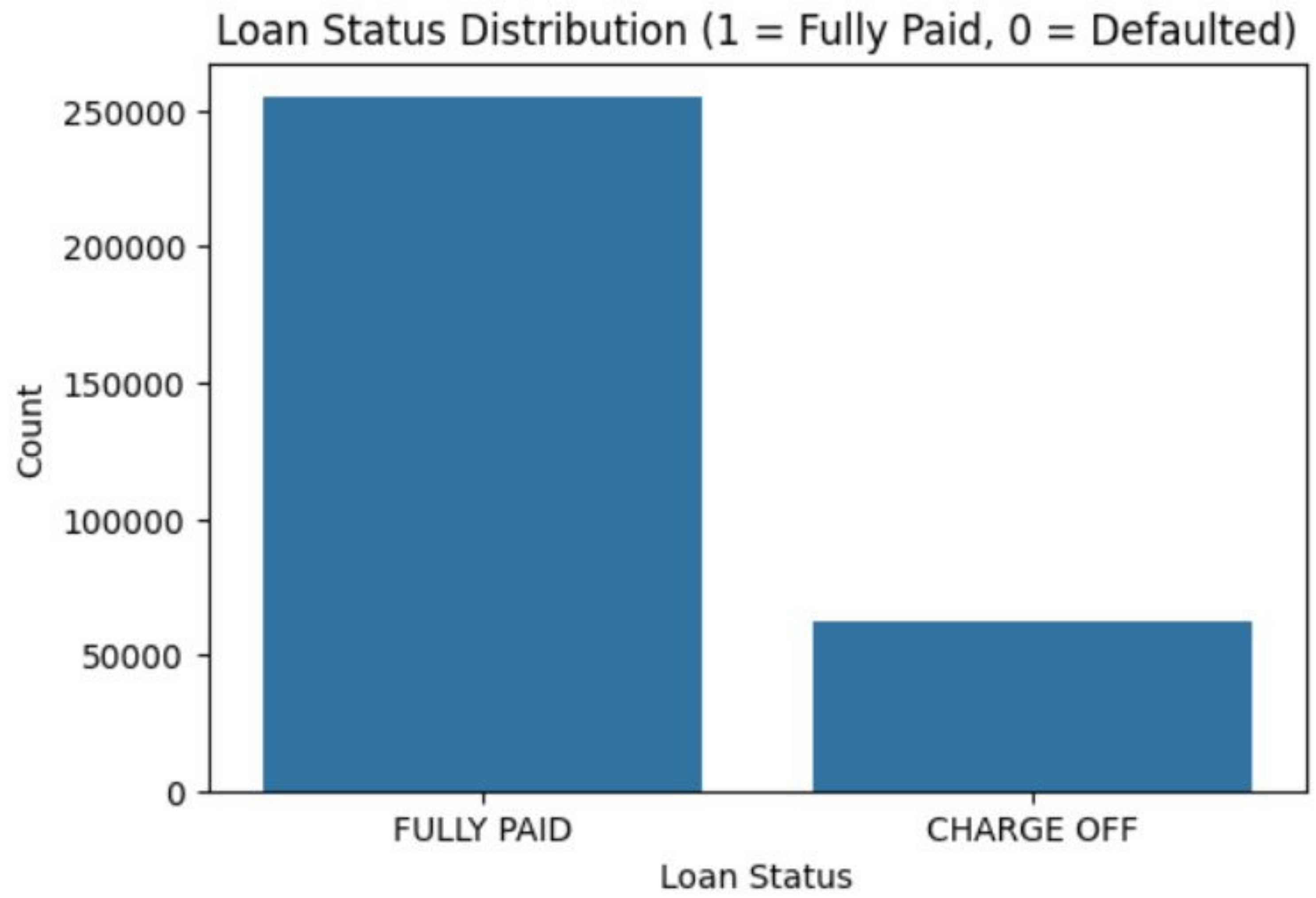
Interpretation :
Highest loan approval rate is for people with more than 10 years of experience

EDA - Box Plot



Interpretation :
Most of the data are in its IQR but there are many extreme values(outliers)

EDA - Dataset Balance



Interpretation :
Dataset is biased towards fully paid. This has to be balanced(using SMOTE)

EDA - Data Preprocessing: Cleaning

Remove Unnecessary fields

- Remove the columns 'Unnamed: 0', 'emp_title', 'title', 'address'

Handle Missing Values

- Imputation by median for 'revol_util','mort_acc','pub_rec_bankruptcies' and by mode 'emp_length'

Handle Duplicates

- There is no duplicates in the dataset

Handle outliers

- Impute outliers by capping(Winsorizing)

Encoding:

- Converted categorical features to numerical using Label Encoding.

Normalizing

- Perform StandardScaler to bring all values into the same scale

Handle Dataset Imbalane

Use SMOTE to handle dataset imbalance on loan_status

EDA - Data Preprocessing: Feature Engineering

Feature Engineering

- Create new feature: Loan-to-Income Ratio
- Convert 'term' column into numerical (36 months -> 36, 60 months -> 60)
- Convert 'emp_length' column into numerical
- Create a binary feature indicating whether the applicant owns a home

Methodology and Outcomes

Model Used	Detailed Steps	Outcome
Logistic Regression	• Data preprocessing • Train-Test Split • Build LogisticRegression Model with C=0.1 and solver as ‘liblinear’ • Prediction and Evaluation	• Classification Model to predict the a person is defaulted or not • Evaluation metrics - Classification Report(Accuracy,Precision,Recall,F1 Score) and Confusion Matrix
Decision Tree	• Data preprocessing • Train-Test Split • Build Model using DecisionTreeClassifie with parameters max_depth=10,min_samples_split=10 and min_samples_leaf=5 • Prediction and Evaluation	• Classification Model to predict the a person is defaulted or not • Evaluation metrics - Classification Report(Accuracy,Precision,Recall,F1 Score) and Confusion Matrix
Random Forest	• Data preprocessing • Train-Test Split • Build model using andomForestClassifie with values (n_estimators=200, max_depth=20,min_samples_split=10,min_samples_leaf=5, and random_state=42 • Prediction and Evaluation	• Classification Model to predict the a person is defaulted or not • Evaluation metrics - Classification Report(Accuracy,Precision,Recall,F1 Score) and Confusion Matrix

Results - Logistic Regression

Results(Generalized) - Without Handling Class Imbalance

```
Logistic Regression Accuracy: 0.8044452156355492
Logistic Regression Classification Report:
              precision    recall  f1-score   support

  CHARGE OFF       0.54      0.09      0.15     12558
  FULLY PAID       0.81      0.98      0.89     50836

    accuracy                           0.80     63394
   macro avg       0.68      0.53      0.52     63394
weighted avg       0.76      0.80      0.74     63394

Logistic Regression Confusion Matrix:
[[ 1068 11490]
 [  907 49929]]
```

- Accuracy: 80.44% → Higher than the previous models but primarily due to correctly predicting "FULLY PAID" cases.
- Recall for "CHARGE OFF" (0.09) → Extremely poor at identifying loan defaults, meaning it misclassifies most of them.
- Confusion Matrix → 11,490 false negatives indicate that the model labels most defaults as "FULLY PAID."

Results - Decision Tree

Results(Generalized) - Without Handling Class Imbalance

```
Decision Tree Accuracy: 0.7989872858630154
Decision Tree Classification Report:
              precision    recall  f1-score   support

  CHARGE OFF       0.46      0.08      0.14     12558
  FULLY PAID       0.81      0.98      0.89     50836

    accuracy                           0.80     63394
   macro avg       0.64      0.53      0.51     63394
weighted avg       0.74      0.80      0.74     63394

Decision Tree Confusion Matrix:
[[ 1026 11532]
 [ 1211 49625]]
```

- Accuracy: 79.89% ⟶ Similar to Logistic Regression, indicating that the model may be overfitting to the majority class.
- Recall for "CHARGE OFF" (0.08) ⟶ Even worse than Logistic Regression; it barely identifies any actual defaults.
- Confusion Matrix ⟶ 11,532 false negatives show the same class imbalance issue as Logistic Regression.

Results - Random Forest

Results(Generalized) - Without Handling Class Imbalance

```
Random Forest Accuracy: 0.8046029592705934
Random Forest Classification Report:
              precision    recall  f1-score   support

  CHARGE OFF       0.56      0.07      0.12     12558
  FULLY PAID       0.81      0.99      0.89     50836

    accuracy                           0.80     63394
   macro avg       0.68      0.53      0.50     63394
weighted avg       0.76      0.80      0.74     63394

Random Forest Confusion Matrix:
[[  831 11727]
 [  660 50176]]
```

- Accuracy: 80.46% → Slightly better than Decision Tree and Logistic Regression, but the issue persists.
- Recall for "CHARGE OFF" (0.07) → Lowest recall among all three models, meaning it hardly catches any actual loan defaults.
- Confusion Matrix → 11,727 false negatives indicate that this model, despite high accuracy, is not useful for identifying risky loans.

Results - Logistic Regression

Results(SMOTE) - After Handling Class Imbalance

```
Logistic Regression Accuracy: 0.6795280310439473
Logistic Regression Classification Report:
              precision    recall  f1-score   support

  CHARGE OFF       0.30      0.47      0.37     12558
  FULLY PAID       0.85      0.73      0.79     50836

    accuracy                           0.68     63394
   macro avg       0.57      0.60      0.58     63394
weighted avg       0.74      0.68      0.70     63394

Logistic Regression Confusion Matrix:
[[ 5870  6688]
 [13628 37208]]
```

- Accuracy: 67.95% ⟶ Performs moderately but struggles with the imbalanced dataset.
- Recall for "CHARGE OFF" (0.47) ⟶ Only 47% of actual "CHARGE OFF" cases are correctly classified, meaning it fails to detect many defaults.
- Confusion Matrix ⟶ High false negatives (13,628) indicate misclassification of "CHARGE OFF" cases as "FULLY PAID."

Results - Decision Tree

Results(SMOTE) - After Handling Class Imbalance

```
--------------------------------------------------------------------------
Decision Tree Accuracy: 0.7397387765403666
Decision Tree Classification Report:
              precision    recall  f1-score   support

  CHARGE OFF       0.32      0.29      0.31     12558
  FULLY PAID       0.83      0.85      0.84     50836

    accuracy                           0.74     63394
   macro avg       0.58      0.57      0.57     63394
weighted avg       0.73      0.74      0.73     63394

Decision Tree Confusion Matrix:
[[ 3627  8931]
 [ 7568 43268]]
```

- Accuracy: 73.97% → Improved performance over Logistic Regression.
- Recall for "CHARGE OFF" (0.29) → Worse recall than Logistic Regression, meaning it still struggles to detect loan defaults.
- Confusion Matrix → 8,931 false positives show that many "FULLY PAID" loans are misclassified as "CHARGE OFF."

Results - Random Forest

Results(SMOTE) - After Handling Class Imbalance

```
Random Forest Accuracy: 0.7795848187525634
Random Forest Classification Report:
              precision    recall  f1-score   support

  CHARGE OFF       0.40      0.23      0.29     12558
  FULLY PAID       0.83      0.92      0.87     50836

    accuracy                           0.78     63394
   macro avg       0.61      0.57      0.58     63394
weighted avg       0.74      0.78      0.75     63394

Random Forest Confusion Matrix:
[[ 2836  9722]
 [ 4251 46585]]
```

- Accuracy: 77.96% ⟶ Best-performing model among the three.
- Precision for "CHARGE OFF" (0.40) ⟶ Better than previous models, but recall is low (0.23), meaning many defaults are still missed.
- Confusion Matrix ⟶ 9,722 false positives and 4,251 false negatives indicate that while it improves accuracy, it still struggles with class imbalance.

Model Accuracy Comparison

Model	Without SMOTE	With SMOTE
Logistic Regression	80.44	67.95%
Decision Tree	79.90	73.97%
Random Forest	80.46	77.96%

Key Findings

EDA : Loan status exhibits a strong positive correlation with installment amounts

Random Forest outperformed over Logistic Regression and Decision tree with and without class balancing.

Without using SMOTE : Due to severe class imbalance, models are biased towards predicting 'FULLY PAID,' leading to high accuracy but poor recall for 'CHARGE OFF.'

Using SMOTE : Random Forest performs best overall Even Though accuracy is reduced compared to without SMOTE technique", recall and F1 score is improved and there is a better balance between "FULLY PAID" and "CHARGE OFF"

Conclusion

Performed Exploratory Data Analysis(EDA) to explore the dataset

Dataset is preprocessed by cleaning and feature engineering

Implemented 3 different classification models to evaluate the problem

Performed hyper parameter tuning and SMOTE analysis on the built models.

Performed predictions on test dataset

Reference

Code Link :
https://colab.research.google.com/drive/1K6ON_X0QjTtEDMavyt-aeVkuuEAno_KI?usp=sharing