



# Bank Loan Analytics: Capstone Final Report

Presented by PGPDSE Chennai, January 2025

Project Group No. 2

## Team Members:

- Dhivya Balaje
- Feroze Ahmed J
- Adlin Raja D
- Deepak K
- Santhosh G

**Mentor Name:** Avanish Kumar Singh

**Dataset Name:** Bank Loan Data

**Reference:** Kaggle

**Link:** <https://www.kaggle.com/datasets/xavierberge/bank-loan-data/data>

## SUMMARY OF PROBLEM STATEMENT, DATA AND FINDINGS

The banking and finance industry increasingly relies on machine learning (ML) models to assess credit risk. Classifying loan applicants into 'good' or 'bad' credit categories allows financial institutions to mitigate default risks, optimize lending strategies, and improve decision-making. Several studies have applied ML algorithms for loan default prediction. Logistic regression continues to be widely used due to its interpretability, especially in regulated environments.

This project aims to develop a robust machine learning model for classifying loan applicants into 'good' or 'bad' credit risk categories, leveraging the comprehensive Bank Loan data dataset.

The primary objective is to develop a machine learning model that can predict whether a bank loan will be classified as a **“Good”** or **“Bad”** loan based on borrower demographics, financial history, and loan characteristics.

This prediction is intended to:

- Reduce credit risk.
- Improve loan approval decision-making.
- Minimize defaults by identifying high-risk borrowers early.

### Dataset Overview

**Source:** Kaggle — Bank Loan Dataset (~150K records)

#### Key characteristics:

Count of numerical variables: 7

Count of categorical variables: 18

The initial dataset contains 38,576 rows and 25 columns.

1. **id:** A unique identifier for each loan record.
2. **address\_state:** The state where the borrower resides.
3. **application\_type:** Indicates if the application was an individual or a joint application.
4. **emp\_length:** Employment length of the borrower in years.
5. **emp\_title:** This should be the job title of the borrower/employer of the borrower.
6. **Grade:** Loan assigned grade by the lending institution (e.g., A, B, C, D, F, G).
7. **home\_ownership:** Indicates if the borrower owns, rents, or has a mortgage on their home.
8. **issue\_date:** The date the loan was issued.
9. **last\_credit\_pull\_date:** The date the borrower's credit was last pulled.
10. **last\_payment\_date:** The date of the last payment made by the borrower.
11. **loan\_status:** The current status of the loan (e.g., Fully Paid, Charged Off, Current, Late).
12. **Good vs Bad Loan:** This is your **target variable** for the classification task.
13. **next\_payment\_date:** The next scheduled payment date.
14. **member\_id:** Similar to id a unique identifier for the member/borrower.

15. **Purpose:** The reason for the loan (e.g., debt consolidation, credit card, home improvement).
16. **sub\_grade:** A more granular categorization of the loan grade (e.g., A1, A2, B1, B2).
17. **term:** The term period on the loan (e.g., 36 months, 60 months).
18. **verification\_status:** Indicates if the borrower's identity, claims, income was verified by the lender.
19. **annual\_income:** The borrower's self-reported annual income.
20. **dti:** Debt-to-income ratio, calculated as the borrower's total monthly debt payments divided by their gross monthly income.
21. **installment:** The monthly payment due by the borrower.
22. **int\_rate:** The interest rate on the loan.
23. **loan\_amount:** The principal amount of the loan.
24. **total\_acc:** The total number of credit lines currently in the borrower's credit file.
25. **total\_payment:** The total payments received to date for the loan.

- **Target variable:** Good vs Bad Loan (binary classification).
- **Feature categories:**
  - **Borrower profile:** emp\_length, annual\_income, home\_ownership, address\_state.
  - **Loan details:** loan\_amount, term, grade, int\_rate, issue\_date.
  - **Credit behaviour:** dti, total\_acc, loan\_status, last\_payment\_date.
- **Data types:** Combination of numeric, categorical, and date fields.
- **Special note:** Contains temporal features (issue\_date, last\_payment\_date) that must follow chronological order.

## Initial Observations & Findings

- **Logical inconsistencies:**
  - Detected **15,000+ records** where issue\_date is after last\_payment\_date.
  - Such anomalies indicate potential **data entry errors**.
- **Missing values:**
  - Some fields (emp\_length, emp\_title, next\_payment\_date) have notable missing entries.
- **Class imbalance:**
  - “Good” loans significantly outnumber “Bad” loans.
- **Feature relationships:**
  - **Higher loan grades (A/B)** correlate with lower risk.
  - **High interest rates** are more common in “Bad” loans.
  - **Debt-to-income ratio (DTI)** is higher in “Bad” loan cases.
- **Outliers:**
  - Extremely high incomes and loan amounts exist but may be genuine (wealthy borrowers) or entry errors.

## Conclusion from Initial Exploration

The dataset provides rich borrower and loan attributes, but requires **significant cleaning** to remove inconsistencies and prepare features for modelling. The early EDA suggests that credit risk is influenced by a combination of **financial discipline** (e.g., DTI, payment history)

and **loan structuring** (e.g., interest rate, grade). Handling imbalance and removing erroneous records will be crucial to building a reliable predictive model.

## OVERVIEW OF THE FINAL PROCESS

The approach followed a standard machine learning project lifecycle:

1. **Data Understanding** → Explore the dataset, identify key attributes, detect errors, and note potential predictors.
2. **Data Cleaning & Preprocessing** → Remove or correct inconsistencies, encode categorical variables, handle missing values.
3. **Feature Engineering** → Derive additional predictors and transform existing ones for better model learning.
4. **Model Selection & Training** → Compare multiple classification algorithms to select the best performer.
5. **Model Evaluation** → Assess models using relevant metrics, especially focusing on detecting “Bad” loans.
6. **Benchmark Comparison** → Compare performance against a simple baseline model.

### Salient Features of the Data

- **High-cardinality categorical features** (e.g., emp\_title) requiring grouping or encoding.
- **Temporal variables** (issue\_date, last\_payment\_date) influencing target prediction.
- **Financial ratios** (dti, payment\_to\_income\_ratio) with direct correlation to loan status.
- **Target imbalance** requiring attention to prevent bias toward predicting “Good” loans.

### Data Preprocessing Steps

- Removed 15K+ logically inconsistent records (issue\_date > last\_payment\_date).
- Handled missing values:
  - **Numerical:** Median imputation.
  - **Categorical:** Mode imputation.
- Encoded categorical variables using **One-Hot Encoding** for nominal features and **Ordinal Encoding** for ordered features (e.g., grade).
- Scaled numeric variables with **StandardScaler** to normalize ranges.

### Algorithms Used

- **Logistic Regression** — as a baseline model.
- **Random Forest Classifier** — for handling non-linear relationships and feature importance.
- **Support Vector Machine (Classification)** – as a suggestion provided by our mentor
- **XGBoost Classifier** — chosen as final model for superior handling of imbalance and complex patterns.

## Combination of Techniques

The final process blended:

- **Data quality improvements** (cleaning, anomaly removal).
  - **Feature engineering** (ratios, categorical splitting).
  - **Algorithm selection based on comparative performance.**
- This ensured the model was both **robust** and **practical** for deployment.

## STEP-BY-STEP WALKTHROUGH OF THE SOLUTION

### Step 1: Data cleaning

- Removed duplicate rows.
- Eliminated inconsistent date records.
- Filled missing values based on feature type and distribution.

### Step 2: Exploratory Data Analysis

- Plotted distributions of numeric variables to detect skewness and outliers.
- Used bar plots and heatmaps to find feature-target relationships.
- Discovered high-risk profiles had higher interest rates and lower loan grades.

### Step 3: Feature Engineering

- Engineered the Grade columns for each specific grade to display the sub-grade number if applicable, and 0 if the loan didn't even belong to that particular grade

### Step 4: Data Transformation

- Applied encoding techniques to categorical variables
- Standardized numeric features for algorithms sensitive to scale

### Step 5: Model Training

- Trained Logistic Regression, Decision Tree, Random Forest, SVM , AdaBoost, GradientBoost, and XGBoost on the processed dataset

### Feature Selection in Models

The encoded, processed dataset had a total of 91 features, excluding the target variable 'Good Loan' (0 if loan is Bad and 1 if it is Good).

Some feature reduction and selection was done using Recursive Feature Elimination, which ranked 45 of those 91 features as highly important.

Models built compared how the performance was with all features and with the RFE-selected features, and for decision trees and random forest, feature importances were used to trim more features as well.

## Step 6: Model Selection

- Chose **XGBoost model trained on the 45 RFE-selected columns** as our final model

## Step 7: Validation

- Performed k-fold cross-validation as well as stratified k-fold to check model stability on final model

## MODEL EVALUATION

### Final Model: XGBoost Classifier with RFE-selected features

**Objective:** Predict “Good” vs “Bad” loans to minimize default risk.

### Prominent Parameters:

**No hyperparameter tuning was done, since our metrics even without them were above 90 percent.** The RFE-selected features were 45 out of the original 91, so only feature reduction was the tuning that was done for our final model.

The important parameters that RFE looked at were:

- Annual income of the applicant
- Monthly instalment amount to be paid by applicant
- Interest rate on the loan
- Loan amount borrowed
- Total payment on the loan
- Whether the borrowers were from the following states: DC, FL, IN, KS, LA, MA, MD, MI, MO, NE, NH, NJ, NM, NV, OH, PA, RI, SD, TN, UT, VA, VT, WV, and WY, or not
- Whether the borrowers were employed for more than 10 years, 2 years, 4 years or 7 years
- Whether the borrower already owned their current house
- The loans borrowed for car, credit card, educational, house, major purchase, moving, renewable energy, small business, vacation, wedding
- Term of loan (60 months or not)

These factors may be the most influential when it comes to determining whether a loan borrowed by a customer under these demographics or not will turn out to be 'Good' or 'Bad'. Maybe these factors can also help the bank figure out more about these demographics and whether they are able to deliver on their loans.

### Key Metrics:

- Accuracy: 98.68% (train accuracy), 97.45% (test accuracy)
- Precision (Bad Loans): 96.86%
- Recall (Bad Loans): 84.3%
- F1-score (Bad Loans): 97.38%
- AUC score: 0.9818

### Evaluation Approach:

- Confusion Matrix → Showed balanced true positive and true negative rates.
- Precision and Recall values → much better than all the other models
- Difference between train accuracy and test accuracy were minimal and within the best-fit range
- Number of features used were also only 45, as opposed to 91
- ROC-AUC Score → ~ 0.98, indicating strong separability.
- Cross-validation → Consistent results across folds, indicating robustness.

### Robustness Factors:

- Inclusion of domain and demographic-relevant features.
- Feature size reduction and cross-validation

### Comparison to Benchmark

#### Benchmark

- Logistic Regression baseline trained on cleaned data – with all features as well as RFE-features.
- Achieved ~95% accuracy in both cases, but recall for “Bad” loans was only ~73%.

### Final Model vs Benchmark

Metric	Benchmark (LogReg)	Final Model (XGBoost)
Accuracy (test)	95.56%	97.45%
Precision (bad loan)	92.87%	96.86%
Recall (bad loan)	72.49%	84.3%
F1-score (weighted)	95.34%	97.38%
Number of features	91	45

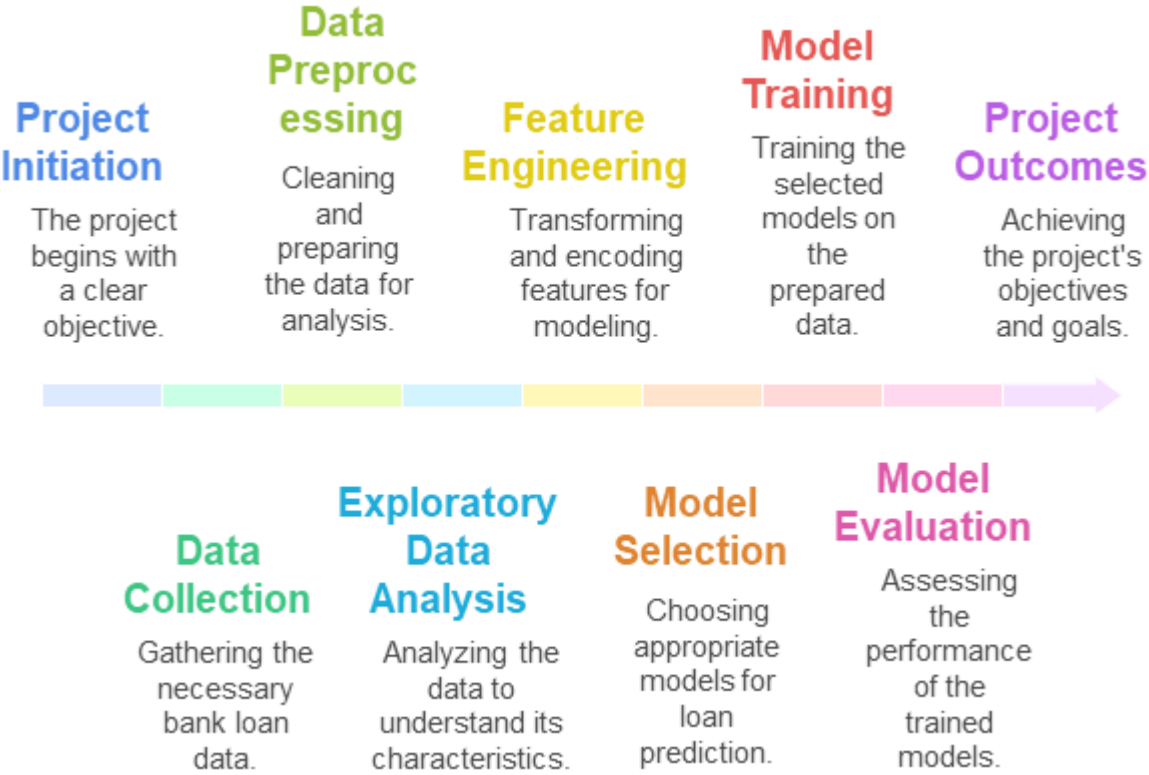
### Reasons for Improvement:

- XGBoost’s ability to capture non-linear relationships.
- Use of less features

### Conclusion:

The final model outperformed the benchmark significantly, particularly in detecting “Bad” loans - the main business objective.

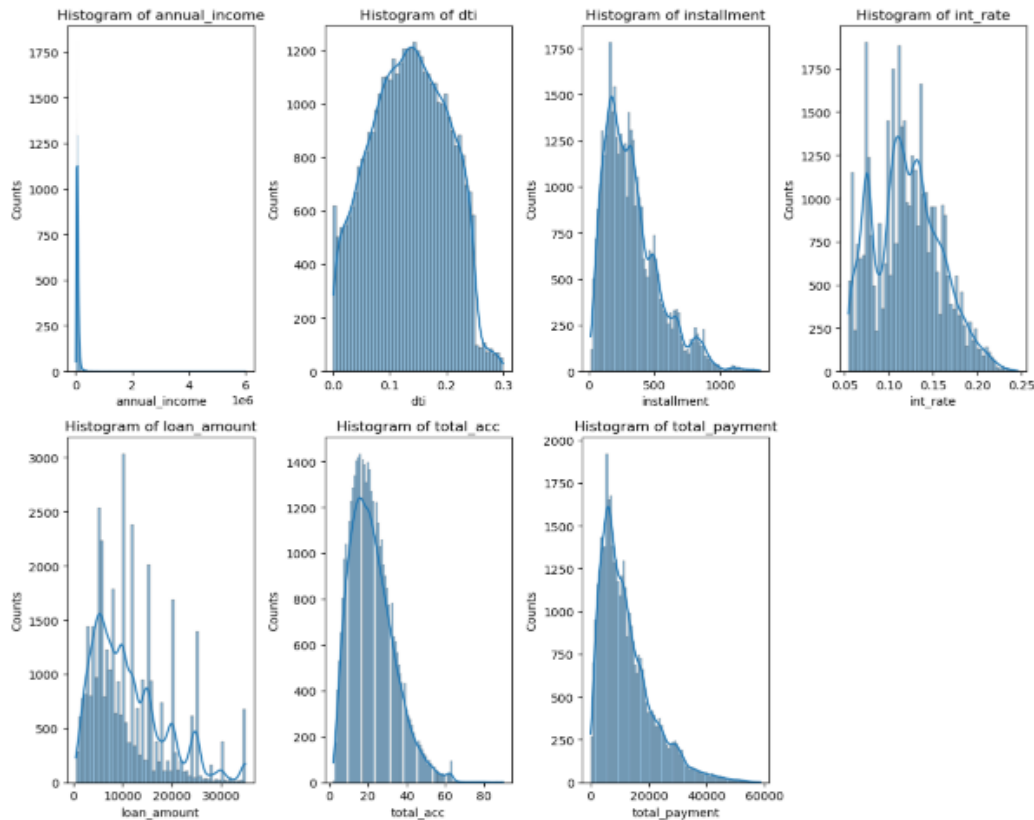
STEPS TAKEN FOR THE OVERALL EXECUTION OF PROJECT





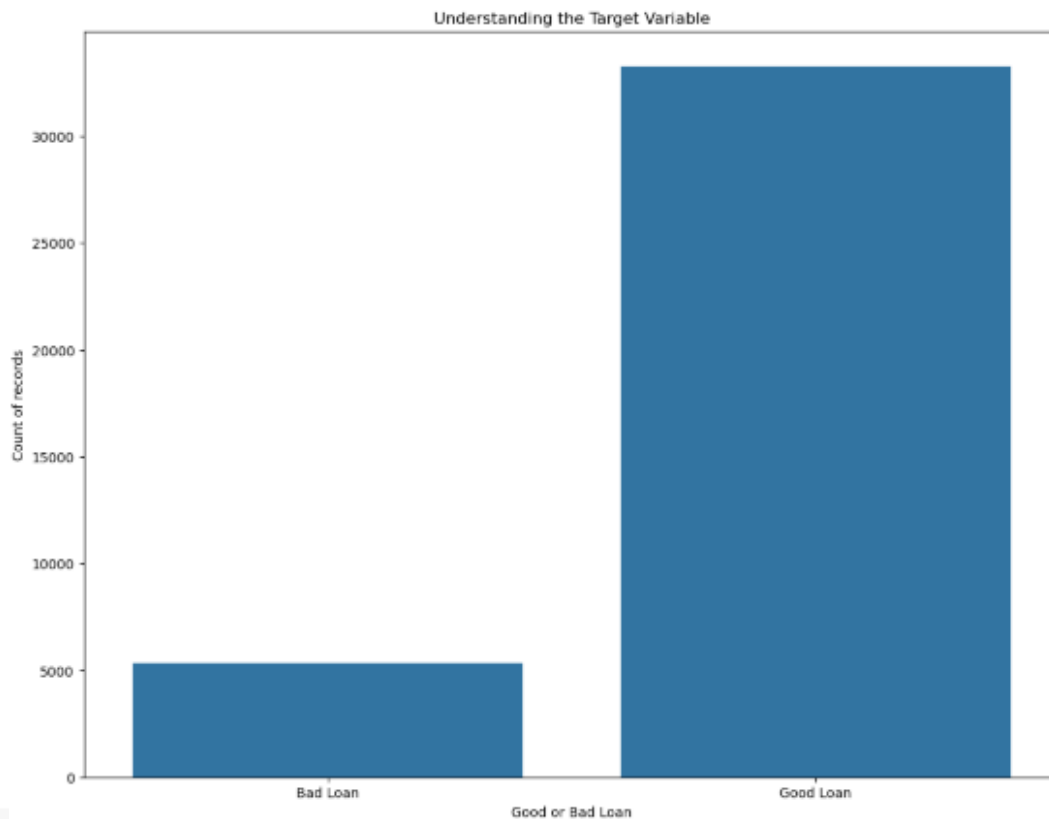
## VISUALIZATIONS

### Histplot for numerical categories



- **annual\_income** is highly skewed to the right, with most values clustered at the lower end. There are a few individuals with very high incomes, which appear as outliers. This suggests that the median might be a better measure of central tendency than the mean for this variable.
- **dti (debt-to-income ratio)** has a distribution that is somewhat bell-shaped but slightly skewed to the right. The majority of borrowers have a DTI between 0 and 0.2, with fewer having a DTI above that.
- **installment** and **int\_rate** both show distributions that are skewed to the right, with a long tail of higher values. Most loans have lower installments and interest rates, and a smaller number have much higher values. The presence of multiple peaks in the **int\_rate** plot might indicate underlying clusters or different loan products.
- **loan\_amount** and **total\_payment** both exhibit a multi-modal distribution with several distinct peaks. This suggests that there are specific, common loan amounts that people borrow and specific total payment amounts. This could be due to standardized loan offerings.
- **total\_acc** (total number of credit lines) is also right-skewed, indicating that most borrowers have a smaller number of credit accounts, while a few have a very large number.

## Target Variable

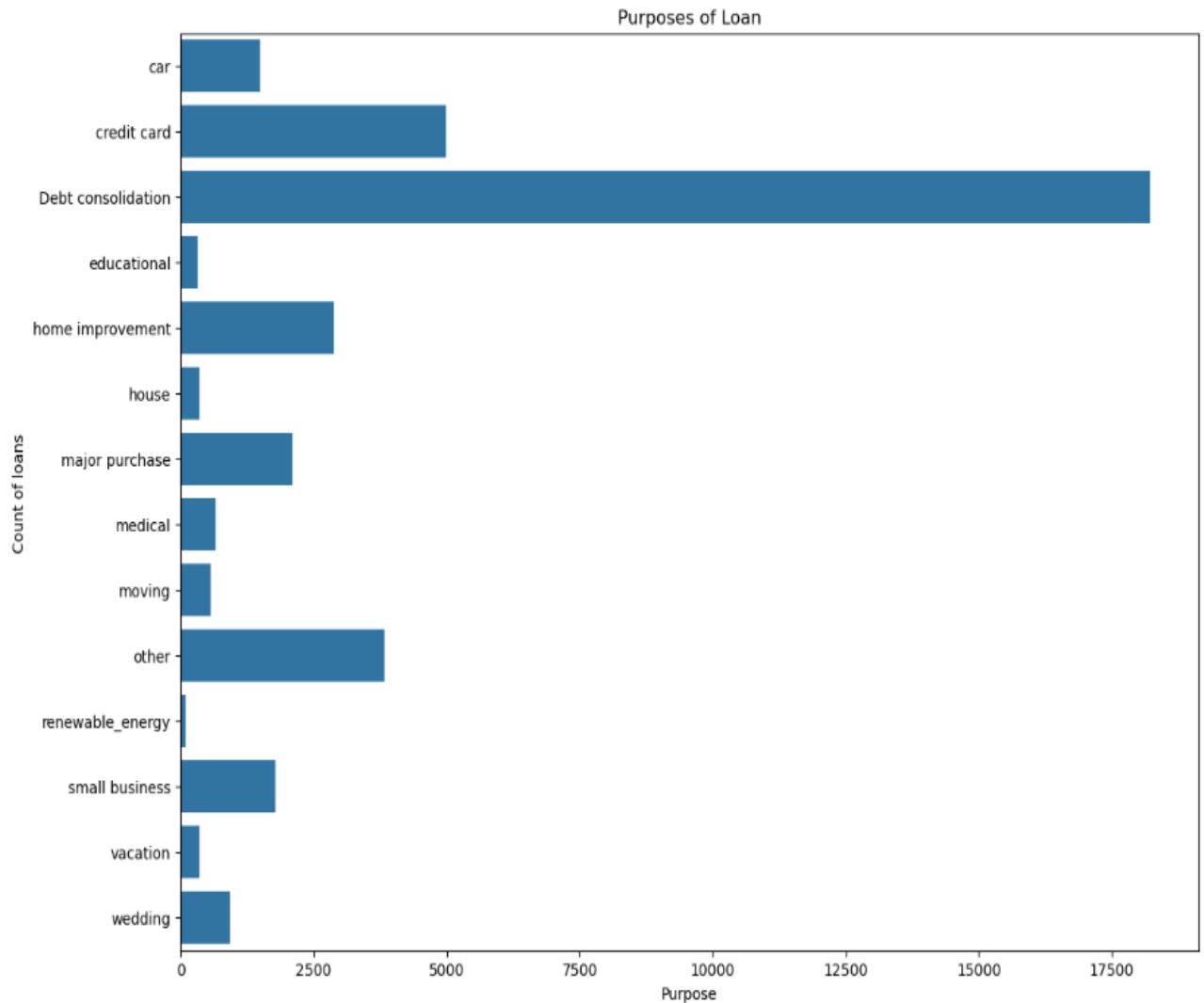


```
[ ] data[target].value_counts(normalize=True)*100
```

```
➡ Good vs Bad Loan  
Good Loan    86.175342  
Bad Loan     13.824658  
Name: proportion, dtype: float64
```

- Inference: Class imbalance present in the target variable, most records/data are those of 'Good' loans.
- More than 86% falls under 'Good' loans
- One way to rectify the class imbalance can be using SMOTE to try oversampling the minority class during the training phase, but our mentor said we may not have to do that compulsorily, so we have not performed SMOTE
- We have retrieved high accuracies even without SMOTE in all our models

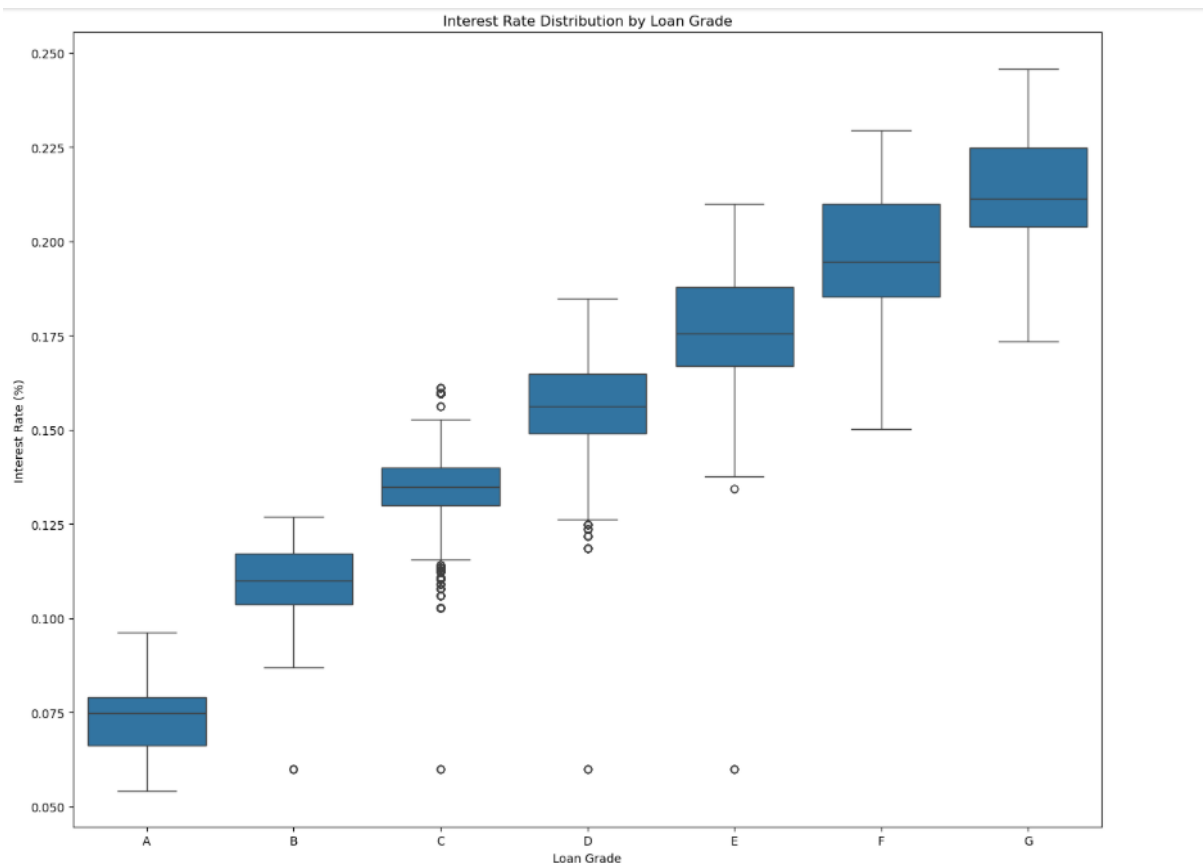
## PURPOSES OF LOAN



### Inference:

- The most prominent inference is the overwhelming dominance of debt consolidation as the reason for borrowing.
- The next most common purposes are credit card and home improvement, but they are a distant second and third
- A wide variety of other purposes, such as car, educational, and medical expenses, make up a very small portion of the total loans

## INTEREST RATE DISTRIBUTION BY LOAN GRADE

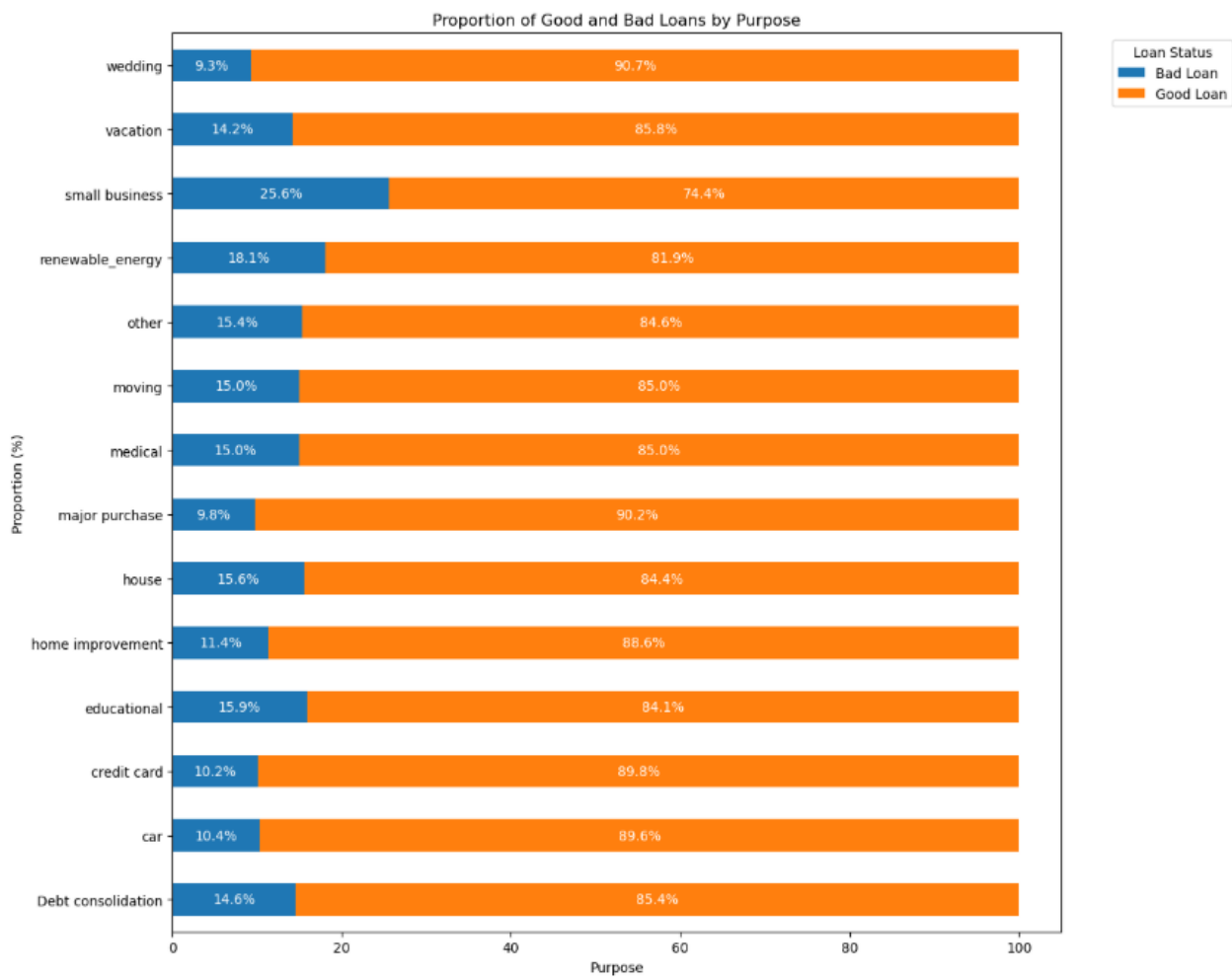


Based on the boxplot, here are the key inferences:

The boxplot shows a comparison of `int_rate` (interest rate) for loans with different grades.

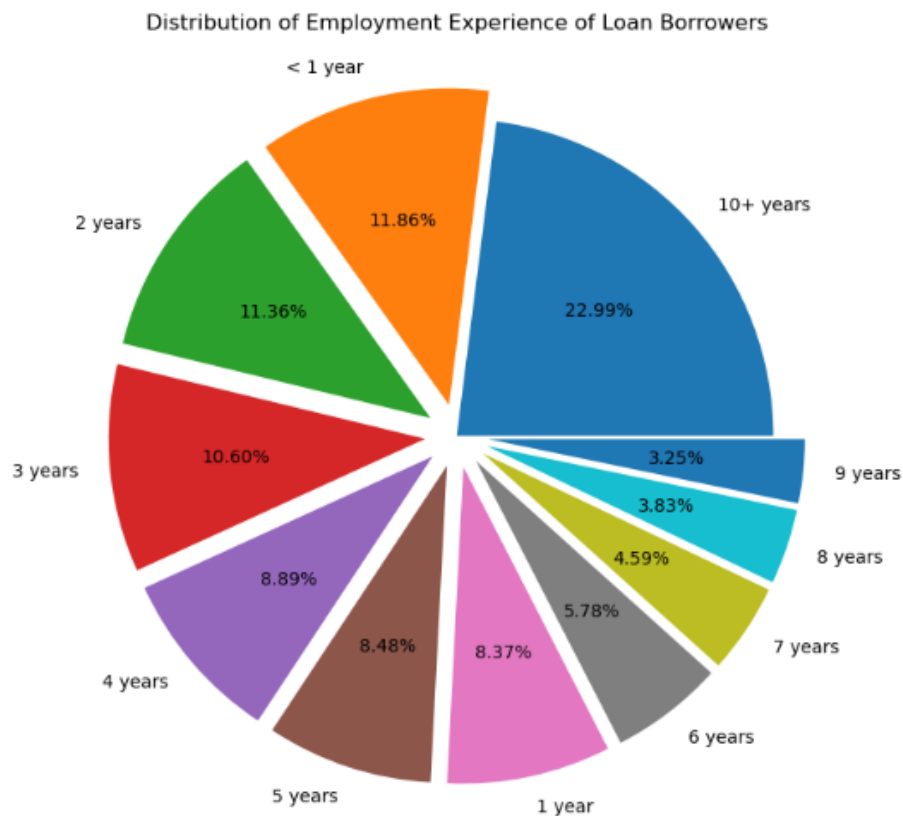
- **Loan Grade and Interest Rate Correlation:** There is a clear, strong positive correlation between loan grade and interest rate. As the loan grade decreases from 'A' to 'G', the median interest rate consistently increases. This is a logical finding, as lower grades represent higher risk, and lenders compensate for this risk with higher interest rates.
- **Outliers:** All grades have outliers, as indicated by the individual points above the whiskers. This suggests that even within a specific loan grade, there are some loans with exceptionally high interest rates.
- **Interquartile Range (IQR):** The spread of interest rates generally increases for lower loan grades. This means there is more variability in interest rates for riskier loans (e.g., grades 'E', 'F', and 'G') compared to safer loans (e.g., grades 'A' and 'B').
- **Median Interest Rate:** The median interest rate (the line inside the box) for Grade 'A' loans is the lowest, and it systematically rises for each subsequent grade, reaching its highest point for Grade 'G' loans. For example, the median for 'A' appears to be around 7-8%, while for 'G' it is well over 20%.

## PROPORTION OF GOOD AND BAD LOAN BY PURPOSE



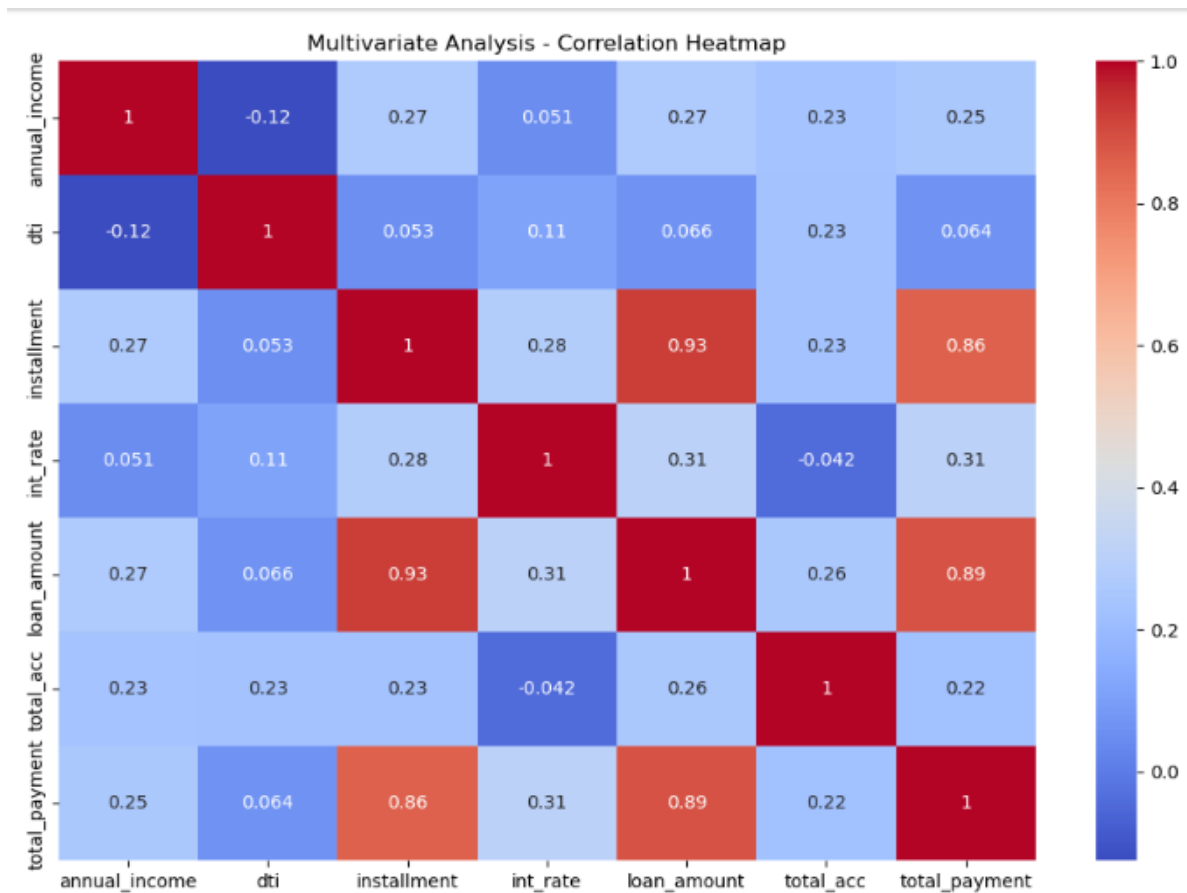
1. **Wedding loans are safest:** Only 9.3% are bad loans, making them the lowest-risk category.
2. **Renewable energy loans excel:** Despite an 18.1% bad loan rate, they boast the highest *good* loan rate (91.9%), indicating strong overall performance.
3. **Small business loans are highest risk:** A significant 25.6% are bad loans, nearly three times the rate of weddings, highlighting inherent volatility.
4. **House & Education loans are moderate risks:** Both have bad loan rates slightly above average (15.6% and 15.9% respectively).
5. **Major Purchases, Credit Cards, Cars are relatively safe:** Bad loan rates for these (9.8%-10.4%) are low, second only to weddings.
6. **Good loan rates are generally high:** Most purposes (except Small Business) have good loan rates exceeding 84%, indicating overall portfolio health.
7. **Risk varies significantly by purpose:** Bad loan rates range from 9.3% (Wedding) to 25.6% (Small Business), emphasizing the critical impact of loan purpose on performance.

## DISTRIBUTION OF EMPLOYMENT EXPERIENCE



1. **Dominance of Highly Experienced Borrowers:** Borrowers with **10+ years of experience** form the single largest group (22.99%), indicating lenders heavily favour or attract applicants with extensive, stable employment histories.
2. **Scarcity of New Entrants:** Borrowers with **less than 1 year** in their current role are the smallest segment (3.25%), suggesting lenders are cautious or new employees rarely seek/qualify for loans.
3. **Gradual Decline in Mid-Tenure:** Representation steadily decreases from the peak at **2 years** (11.36%) down through **3 years** (10.60%), **4 years** (8.89%), **5 years** (8.48%), **6 years** (8.37%), **7 years** (5.78%), **8 years** (4.59%), and **9 years** (3.25%), showing fewer borrowers remain in each subsequent year of mid-tenure.
4. **Significant Mid-Career Segment:** Despite the decline, borrowers with **2-6 years** of experience collectively represent a very substantial portion ( $11.36\% + 10.60\% + 8.89\% + 8.48\% + 8.37\% = \sim 47.7\%$ ), forming a critical core of the borrowing base.
5. **Risk Assessment Implication:** This distribution highlights **employment stability as a key factor** in lending decisions. The low volume of very new employees and the high volume of very experienced ones suggest stability is heavily weighted in creditworthiness evaluation.

## HEATMAP



1. **Loan amount and installment** have a very strong positive correlation (0.93), which is expected since higher loans lead to higher installments.
2. **Loan amount** also shows a high correlation with **total payment** (0.89), indicating that larger loans usually involve larger total repayments.
3. **Installment and total payment** are strongly correlated (0.86), again due to the repayment structure.
4. **Annual income** shows a weak positive correlation with **loan amount** (0.27), suggesting higher earners tend to take slightly larger loans.
5. **Annual income** has a small negative correlation with **DTI** (-0.12), meaning higher earners tend to have a slightly better debt-to-income ratio.
6. **Interest rate** is moderately correlated with **loan amount** (0.31) and **installment** (0.28), implying larger loans often come with higher interest rates.
7. **Total accounts** have low correlations with all other features, suggesting they may not be strong predictors in this dataset.
8. The low correlations between **DTI** and most other variables suggest it provides independent information about borrower financial health.
9. Strong correlations between loan-related variables could cause **multicollinearity** in models, affecting interpretability.
10. Overall, the heatmap highlights key relationships but also points to redundancy among some features, which may require **feature selection or dimensionality reduction** before modeling.

## Model Building:

### Base Model Building – Logistic Regression

```
[ ] from sklearn.linear_model import LogisticRegression
    from sklearn.metrics import classification_report, confusion_matrix, roc_curve, roc_auc_score, accuracy_score
    base_model = LogisticRegression(random_state=42)
    base_model.fit(xtrain, ytrain)
    ypred_base = base_model.predict(xtest)
```

```
[ ] print('Training accuracy:', base_model.score(xtrain, ytrain))
    print('Test accuracy:', base_model.score(xtest, ytest))
    print('Model accuracy:', accuracy_score(ytest, ypred_base))
```

↔ Training accuracy: 0.9579714841218405  
Test accuracy: 0.9556765163297045  
Model accuracy: 0.9556765163297045

```
[ ] print(classification_report(ytest, ypred_base))
```

↔

	precision	recall	f1-score	support
0	0.93	0.73	0.82	1064
1	0.96	0.99	0.97	6652
accuracy			0.96	7716
macro avg	0.94	0.86	0.90	7716
weighted avg	0.95	0.96	0.95	7716



## Decision Tree

```
[ ] from sklearn.tree import DecisionTreeClassifier
    dt = DecisionTreeClassifier(random_state=42)
    dt.fit(xtrain,ytrain)
    ypred_dt = dt.predict(xtest)
```

```
[ ] print('Training accuracy:',dt.score(xtrain,ytrain))
    print('Test accuracy:',dt.score(xtest,ytest))
    print('Model accuracy:',accuracy_score(ytest,ypred_dt))
```

↔ Training accuracy: 1.0  
Test accuracy: 0.9543805080352514  
Model accuracy: 0.9543805080352514

```
[ ] print(classification_report(ytest, ypred_dt))
```


↔

	precision	recall	f1-score	support
0	0.83	0.84	0.84	1064
1	0.97	0.97	0.97	6652
accuracy			0.95	7716
macro avg	0.90	0.91	0.90	7716
weighted avg	0.95	0.95	0.95	7716

## Random Forest

```
[ ] from sklearn.ensemble import RandomForestClassifier
    rf = RandomForestClassifier(random_state=42)
    rf.fit(xtrain,ytrain)
    ypred_rf = rf.predict(xtest)
```

```
[ ] print(classification_report(ytest, ypred_rf))
```




	precision	recall	f1-score	support
0	1.00	0.58	0.73	1064
1	0.94	1.00	0.97	6652
accuracy			0.94	7716
macro avg	0.97	0.79	0.85	7716
weighted avg	0.95	0.94	0.94	7716

## Support Vector Machine (default rbf kernel)

```
[ ] from sklearn.svm import SVC

    svm_model = SVC(kernel='rbf', probability=True, random_state=42)
    svm_model.fit(xtrain, ytrain)
    ypred_svm = svm_model.predict(xtest)
```

```
[ ] print(classification_report(ytest, ypred_svm))
```




	precision	recall	f1-score	support
0	1.00	0.69	0.82	1064
1	0.95	1.00	0.98	6652
accuracy			0.96	7716
macro avg	0.97	0.85	0.90	7716
weighted avg	0.96	0.96	0.95	7716

## AdaBoost Classifier

```
[ ] abc = AdaBoostClassifier(random_state=42)
      abc.fit(xtrain,ytrain)
      ypred_abc = abc.predict(xtest)
```

```
[ ] print(classification_report(ytest,ypred_abc))
```




	precision	recall	f1-score	support
0	0.99	0.65	0.79	1064
1	0.95	1.00	0.97	6652
accuracy			0.95	7716
macro avg	0.97	0.83	0.88	7716
weighted avg	0.95	0.95	0.95	7716

## Gradient Boosting Classifier

```
[ ] gb = GradientBoostingClassifier(random_state=42)
      gb.fit(xtrain,ytrain)
      ypred_gb = gb.predict(xtest)
```

```
[ ] print(classification_report(ytest,ypred_gb))
```




	precision	recall	f1-score	support
0	1.00	0.74	0.85	1064
1	0.96	1.00	0.98	6652
accuracy			0.96	7716
macro avg	0.98	0.87	0.91	7716
weighted avg	0.97	0.96	0.96	7716

## XGBoost Classifier

```
[ ] xgb = XGBClassifier(random_state=42)
      xgb.fit(xtrain,ytrain)
      ypred_xgb = xgb.predict(xtest)
```

```
[ ] print(classification_report(ytest, ypred_xgb))
```



	precision	recall	f1-score	support
0	0.98	0.83	0.90	1064
1	0.97	1.00	0.99	6652
accuracy			0.97	7716
macro avg	0.97	0.92	0.94	7716
weighted avg	0.97	0.97	0.97	7716

## Final Scorecard with all models

17]:

	Model Name	Number of features	Train Accuracy Score	Test Accuracy Score	ROC-AUC Score	f1-weighted	Precision (Bad loan)	Recall (Bad loan)
0	Logistic Regression (Base model)	91	0.957971	0.955677	0.961007	0.953459	0.928741	0.734962
1	Logistic Regression (features selected by RFE)	45	0.957388	0.955936	0.960611	0.953617	0.935096	0.731203
2	Decision Tree	91	1.000000	0.954381	0.906433	0.954488	0.830855	0.840226
3	Decision Tree (reduced features)	82	1.000000	0.954381	0.907222	0.954523	0.829630	0.842105
4	Decision Tree (features selected by RFE)	45	1.000000	0.956843	0.909835	0.956868	0.842549	0.844925
5	Decision Tree (reduced features from RFE featu...	42	1.000000	0.956843	0.908650	0.956817	0.844486	0.842105
6	Random Forest	91	1.000000	0.941939	0.958464	0.935222	0.996774	0.580827
7	Random Forest (RFE features)	45	0.999968	0.963582	0.968606	0.961249	0.996198	0.738722
8	SVC model (all features)	91	0.961990	0.956973	0.967242	0.953595	0.995935	0.690789
9	Linear kernel SVC model (all features)	91	0.962443	0.959435	0.950764	0.957120	0.960736	0.735902
10	SVC model (RFE-features)	45	0.970350	0.967600	0.971553	0.965793	0.996341	0.767857
11	Linear kernel SVC model (RFE-features)	45	0.962216	0.960731	0.950944	0.958469	0.968020	0.739662
12	AdaBoost (all features)	91	0.952366	0.951011	0.966084	0.946609	0.990000	0.651316
13	AdaBoost (RFE-features)	45	0.952366	0.951011	0.966084	0.946609	0.990000	0.651316
14	GradientBoost (all features)	91	0.964971	0.963971	0.976507	0.961692	0.996212	0.741541
15	GradientBoost (RFE-features)	45	0.965262	0.963841	0.977000	0.961544	0.996207	0.740602
16	XGBoost (all features)	91	0.990246	0.974339	0.981718	0.973485	0.975824	0.834586
17	XGBoost (RFE-features)	45	0.986844	0.974598	0.981887	0.973848	0.968683	0.843045

**The best model was determined as the XGBoost (RFE-features)**

**K-fold cross-validation also confirms model's effectiveness.**

### Attempting K-fold cross validation on the best model

```
from sklearn.model_selection import KFold, StratifiedKFold, cross_val_score

kf = KFold(n_splits=5, shuffle=True, random_state=42)
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

model_chosen = xgb_rfe

# Using KFold
scores_kf = cross_val_score(model_chosen, X_r, y_r, cv=kf, scoring='accuracy')

# Using StratifiedKFold (recommended for binary classification)
scores_skf = cross_val_score(model_chosen, X_r, y_r, cv=skf, scoring='accuracy')

print(f"KFold Cross-Validation Accuracy: {scores_kf.mean():.4f} (+/- {scores_kf.std():.4f})")
print(f"StratifiedKFold Cross-Validation Accuracy: {scores_skf.mean():.4f} (+/- {scores_skf.std():.4f})")

KFold Cross-Validation Accuracy: 0.9741 (+/- 0.0007)
StratifiedKFold Cross-Validation Accuracy: 0.9738 (+/- 0.0008)
```

### Limitations faced during this project

- **Data Quality Issues** – The dataset contained inconsistencies (e.g., missing values, possible input errors such as issue\_date after last\_payment\_date). Such anomalies could impact model accuracy if not fully addressed.
- **Feature Limitations** – Certain potentially predictive variables were unavailable, meaning the model may be missing important signals.
- **Generalization Risk** – The model was trained on historical data and may not adapt well to changes in patterns over time.
- **Imbalanced Data** – Some classes/labels had far fewer samples, which could bias predictions toward the majority class.
- **Algorithm Constraints** – While the chosen algorithm performed well in testing, it might not capture complex non-linear relationships as effectively as advanced ensemble or deep learning approaches.

### Closing Reflections

- This project reinforced the **importance of thorough data preprocessing**—even the most sophisticated algorithm will underperform with poor-quality inputs.
- Visualizing the data early in the process revealed key patterns that shaped feature engineering decisions.
- We learned that **clear evaluation metrics** are crucial to measure true improvement beyond just accuracy.