MACHINE LEARNING

Project Report
German Bank Loan

Introduction

In the banking sector, managing loan defaulters is a critical challenge for financial institutions. A German bank is grappling with this issue and seeks to develop a machine learning model to predict whether a customer will default on their loan. To achieve this, the bank has provided a dataset containing historical information on customers who have taken loans. For each customer, this dataset includes various features such as employment duration, existing loan count, savings balance, percentage of income, age, and more.

The problem context revolves around building a machine learning model to predict loan defaulters for a German bank. The goal is to analyze this data and develop a predictive model that can accurately determine whether a customer is likely to default on their loan based on these features. This model will help the bank identify customers who are likely to default on their loans, enabling them to take proactive measures to mitigate risks and optimize their lending practices. By analyzing various customer attributes such as employment duration, existing loan count, and savings balance, the bank seeks to enhance its decision-making process and improve overall financial stability.

In the pursuit of developing a robust predictive model, several intriguing questions emerge. Firstly, what are the key factors that contribute to loan default among customers? Through exploratory data analysis (EDA), I aim to uncover patterns and correlations within the dataset to identify these influential factors. Additionally, how do different customer segments, such as age groups, employment types, and credit history, vary in their likelihood of defaulting on loans? By segmenting the data and conducting comparative analyses, I hope to gain insights into the differential impact of these factors. Furthermore, how can machine learning algorithms be effectively applied to predict loan defaults with a high degree of accuracy? Through model development and evaluation, I aim to identify the most suitable approach that balances predictive performance with interpretability, ensuring practical utility for the bank's decision-making processes.

Data Columns Context

The data set has 17 columns and 1000 rows. Columns are described below and each row is a customer.

```
checking_balance - Amount of money available in account of customers
months_loan_duration - Duration since loan taken
credit_history - credit history of each customers
purpose - Purpose why loan has been taken
amount - Amount of loan taken
savings_balance - Balance in account
employment_duration - Duration of employment
percent_of_income - Percentage of monthly income
years_at_residence - Duration of current residence
age - Age of customer
other_credit - Any other credits taken
housing- Type of housing, rent or own
existing_loans_count - Existing count of loans
job - Job type
dependents - Any dependents on customer
phone - Having phone or not
default - Default status (Target column)
```

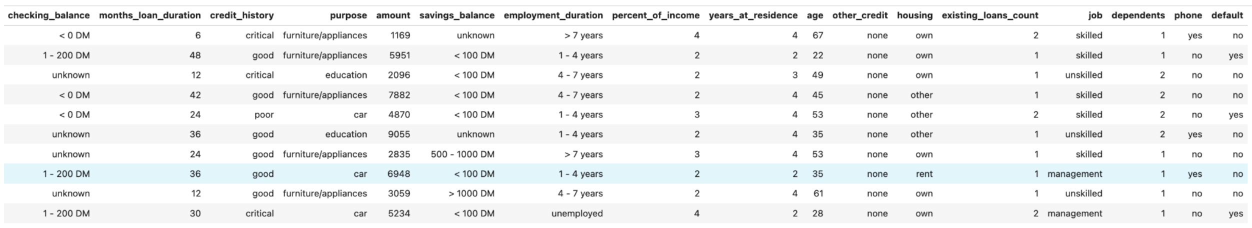
Objectives

The objective of this final project is to create a comprehensive and reproducible project that showcases the entire process of developing and evaluating machine learning models for predicting loan defaults in the banking financial sector. The project will be designed to be easily understood and replicable, with a focus on clear documentation and thorough evaluation metrics.

Methods and Materials

Data Description

The data set has 17 columns and 1000 rows. Columns are described below and each row is a customer.



checking_balance	months_loan_duration	credit_history	purpose	amount	savings_balance	employment_duration	percent_of_income	years_at_residence	age	other_credit	housing	existing_loans_count	job	dependents	phone	default
< 0 DM	6	critical	furniture/appliances	1169	unknown	> 7 years	4	4	67	none	own	2	skilled	1	yes	no
1 - 200 DM	48	good	furniture/appliances	5951	< 100 DM	1 - 4 years	2	2	22	none	own	1	skilled	1	no	yes
unknown	12	critical	education	2096	< 100 DM	4 - 7 years	2	3	49	none	own	1	unskilled	2	no	no
< 0 DM	42	good	furniture/appliances	7882	< 100 DM	4 - 7 years	2	4	45	none	other	1	skilled	2	no	no
< 0 DM	24	poor	car	4870	< 100 DM	1 - 4 years	3	4	53	none	other	2	skilled	2	no	yes
unknown	36	good	education	9055	unknown	1 - 4 years	2	4	35	none	other	1	unskilled	2	yes	no
unknown	24	good	furniture/appliances	2835	500 - 1000 DM	> 7 years	3	4	53	none	own	1	skilled	1	no	no
1 - 200 DM	36	good	car	6948	< 100 DM	1 - 4 years	2	2	35	none	rent	1	management	1	yes	no
unknown	12	good	furniture/appliances	3059	> 1000 DM	4 - 7 years	2	4	61	none	own	1	unskilled	1	no	no
1 - 200 DM	30	critical	car	5234	< 100 DM	unemployed	4	2	28	none	own	2	management	1	no	yes

```
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   checking_balance      1000 non-null   object
 1   months_loan_duration  1000 non-null   int64
 2   credit_history        1000 non-null   object
 3   purpose               1000 non-null   object
 4   amount                1000 non-null   int64
 5   savings_balance       1000 non-null   object
 6   employment_duration   1000 non-null   object
 7   percent_of_income     1000 non-null   int64
 8   years_at_residence    1000 non-null   int64
 9   age                   1000 non-null   int64
 10  other_credit          1000 non-null   object
 11  housing               1000 non-null   object
 12  existing_loans_count  1000 non-null   int64
 13  job                   1000 non-null   object
 14  dependents            1000 non-null   int64
 15  phone                 1000 non-null   object
 16  default               1000 non-null   object
```

```
Unique values for checking_balance: ['< 0 DM' '1 - 200 DM' 'unknown' '> 200 DM']
Unique values for credit_history: ['critical' 'good' 'poor' 'perfect' 'very good']
Unique values for purpose: ['furniture/appliances' 'education' 'car' 'business' 'renovations' 'car0']
Unique values for savings_balance: ['unknown' '< 100 DM' '500 - 1000 DM' '> 1000 DM' '100 - 500 DM']
Unique values for employment_duration: ['> 7 years' '1 - 4 years' '4 - 7 years' 'unemployed' '< 1 year']
Unique values for other_credit: ['none' 'bank' 'store']
Unique values for housing: ['own' 'other' 'rent']
Unique values for job: ['skilled' 'unskilled' 'management' 'unemployed']
Unique values for phone: ['yes' 'no']
Unique values for default: ['no' 'yes']
```

Here DM stands for Deutsche Mark.

In the data, I see that amount, dependents, existing_loans_count, months_loan_duration, percent_of_income, years_at_residence, age are numerical columns. Among categorical columns, credit_history, and employment_duration are ordinal while checking_balance, purpose, savings_balance, other_credit, housing, job, phone, and default are nominal. Note that some of the columns have 'unknown', 'other', 'none' or 'unemployed' as a category and hence cannot be considered as an ordinal variable even though other categories within the column have a hierarchical order. For `employment_duration' column, the 'unemployed' column can be considered as the ordinal categorical column.

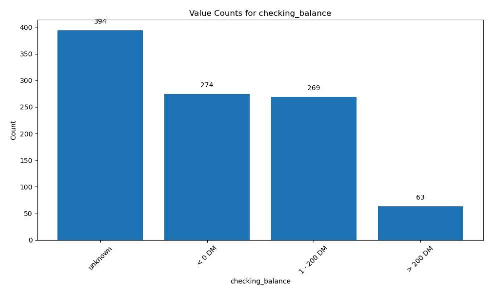
Data Cleaning

In the data cleaning process, I identified and corrected inconsistencies in column values, specifically renaming 'car0' to 'car' in the purpose column to ensure consistency. Additionally, I found that some categorical columns contained 'unknown' values, which I treated as a separate category, ensuring that no missing values were present in the dataset. For the phone and default columns, I encoded the value by zero and one, respectively, because the target is the target variable and I need to understand the correlation between the target variable (default) and any other variables.

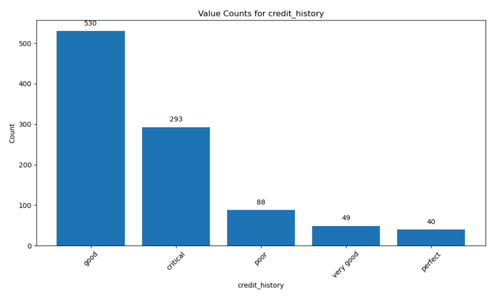
Exploratory Data Analysis

1. Univariate Analysis

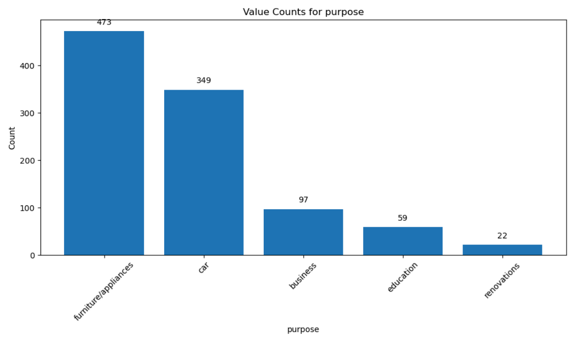
In the given data, the number of defaulted customers are only less than one-third of the total customers. This may give rise to imbalanced dataset problem, where the target class has an uneven distribution of observations.



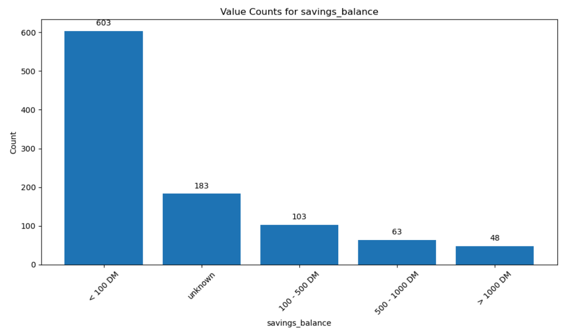
Value Counts for checking_balance
Count
0
50
100
150
200
250
300
350
400
394
274
269
63
unknown
< 0 DM
1 - 200 DM
> 200 DM
checking_balance



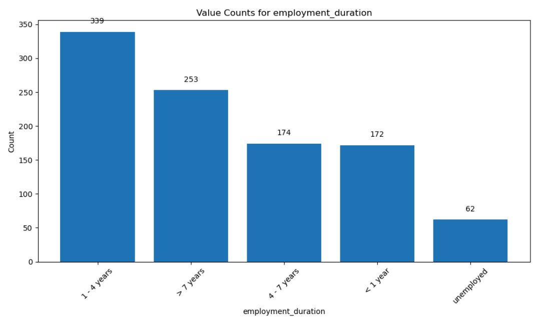
Value Counts for credit_history
Count
0
100
200
300
400
500
530
293
88
49
40
good
critical
poor
very good
perfect
credit_history



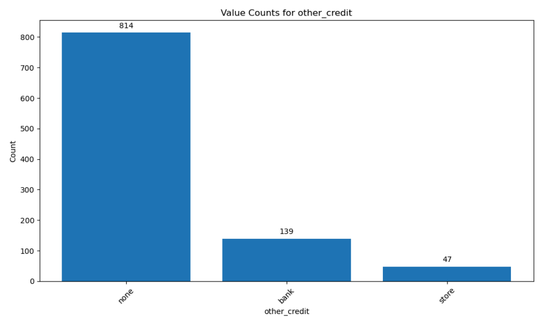
Value Counts for purpose
Count
0
100
200
300
400
473
349
97
59
22
furniture/appliances
car
business
education
renovations
purpose



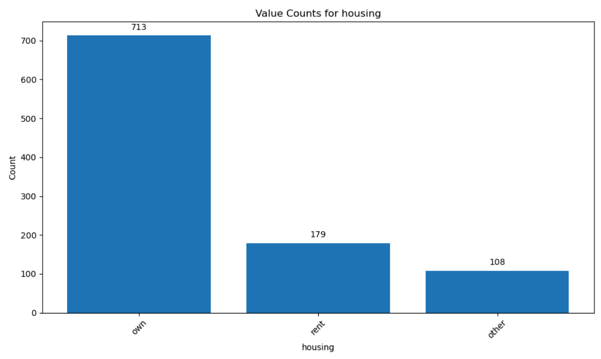
Value Counts for savings_balance
Count
0
100
200
300
400
500
600
603
183
103
63
48
< 100 DM
unknown
100 - 500 DM
500 - 1000 DM
> 1000 DM
savings_balance



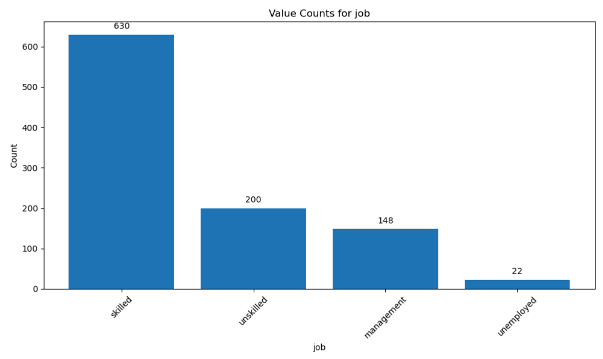
Value Counts for employment_duration
Count
0
50
100
150
200
250
300
350
339
253
174
172
62
1 - 4 years
> 7 years
4 - 7 years
< 1 year
unemployed
employment_duration



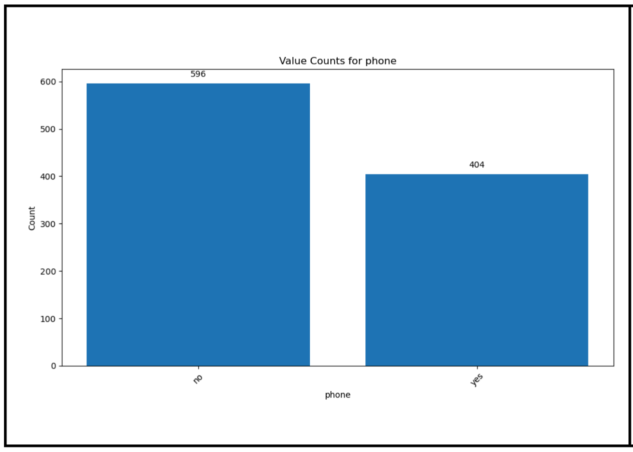
Value Counts for other_credit
Count
0
100
200
300
400
500
600
700
800
814
139
47
none
bank
store
other_credit

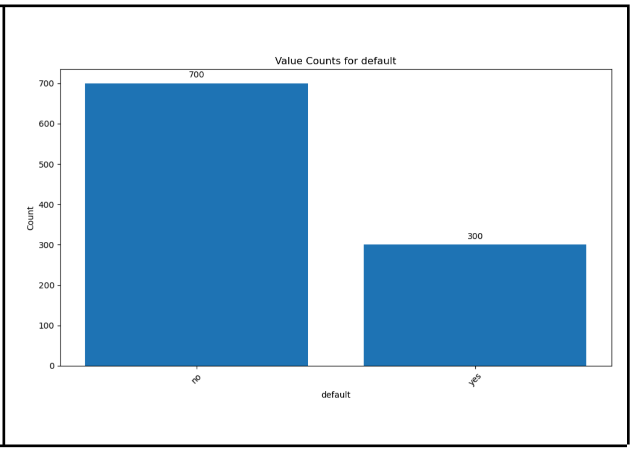


Value Counts for housing
Count
0
100
200
300
400
500
600
700
713
179
108
own
rent
other
housing



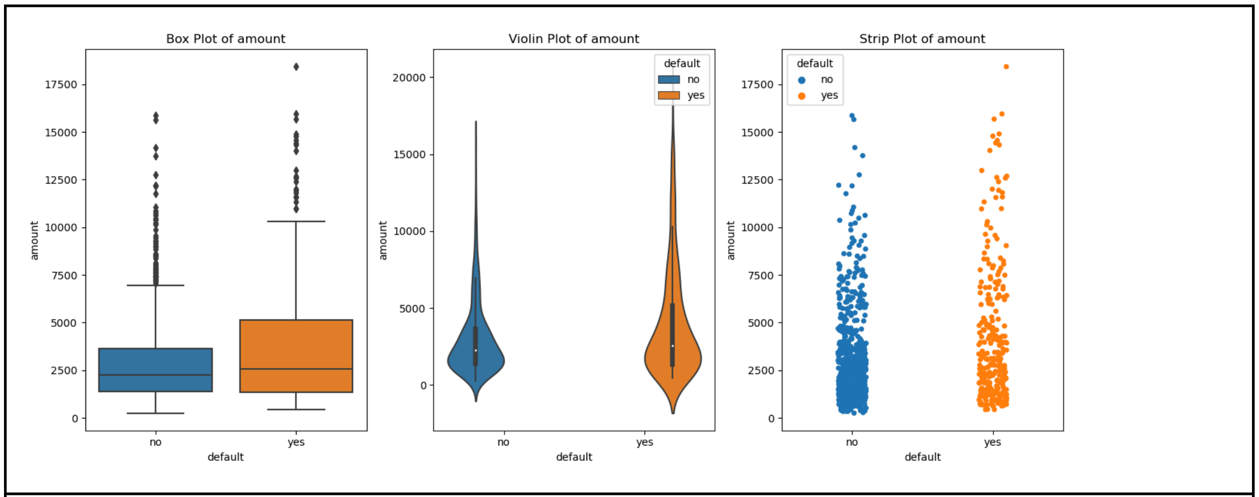
Value Counts for job
Count
0
100
200
300
400
500
600
630
200
148
22
skilled
unskilled
management
unemployed
job

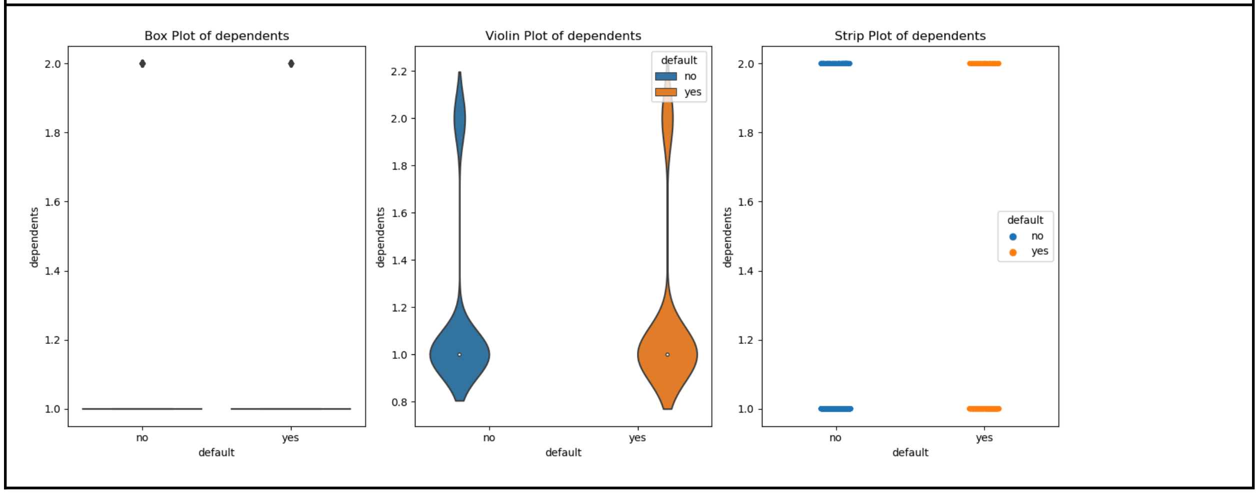


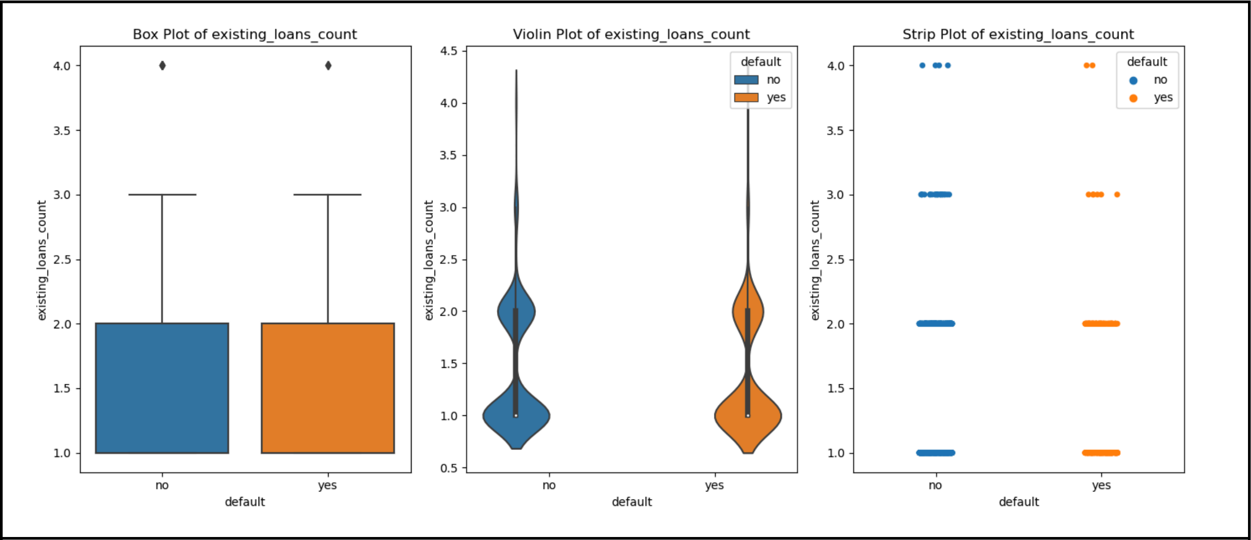


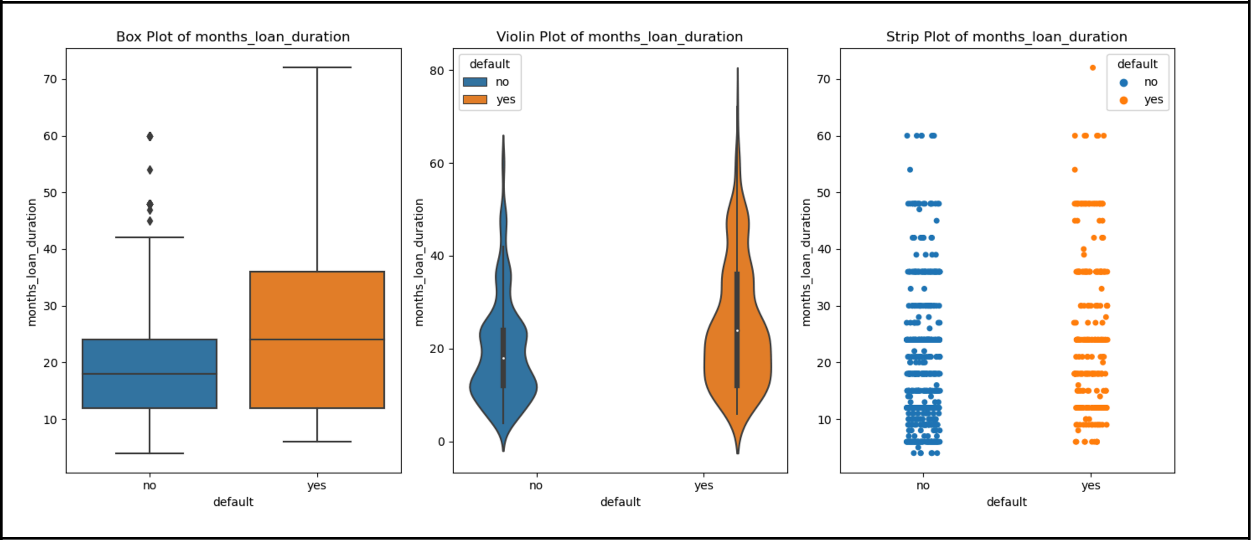
2. Multivariate Analysis

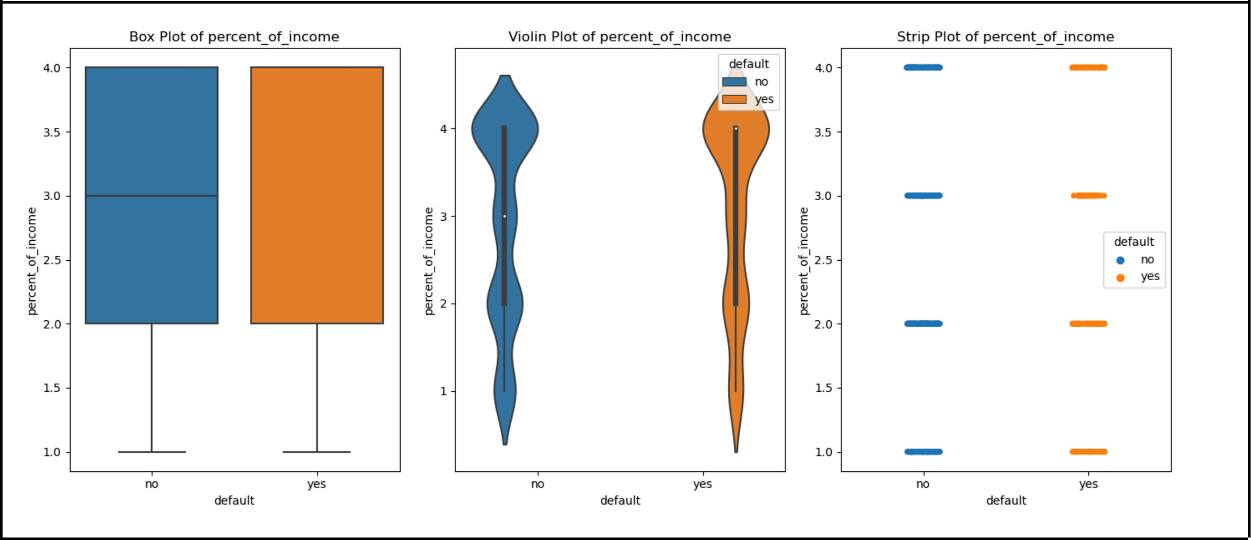
I have also plotted the box plot, violin plot and strip plot to identify the relation if any between the variables for default status.

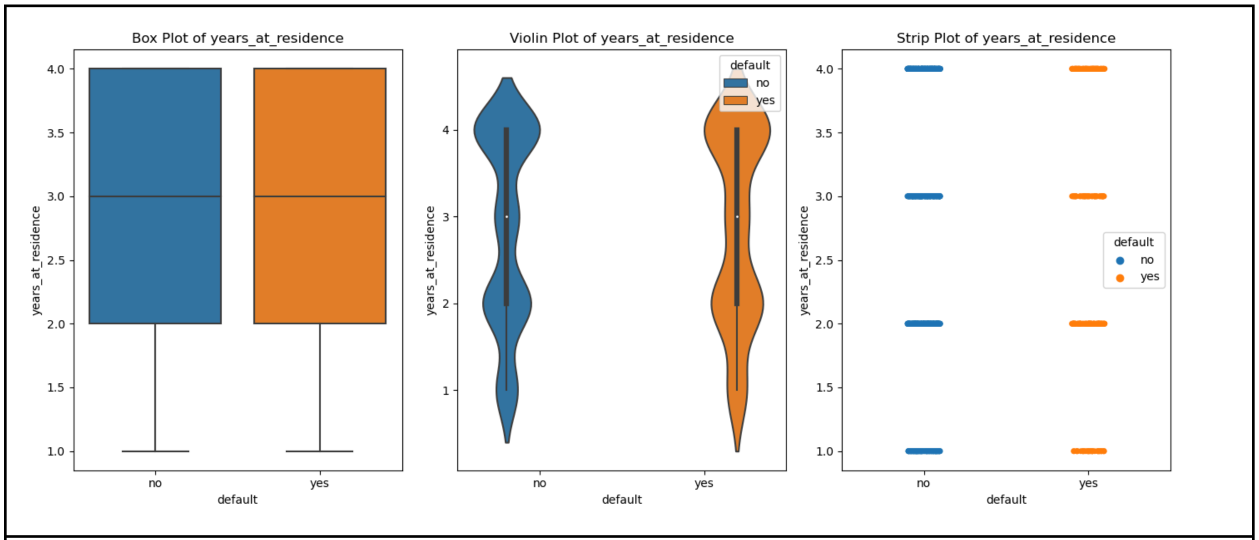


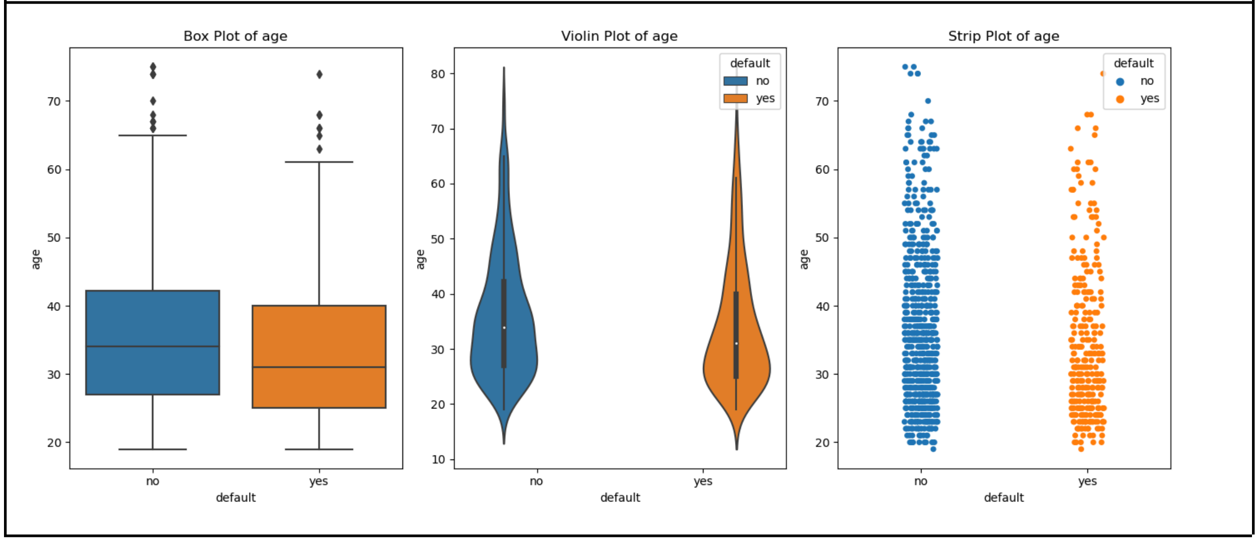



Box Plot of existing_loans_count
Violin Plot of existing_loans_count
Strip Plot of existing_loans_count
existing_loans_count
default
no
yes

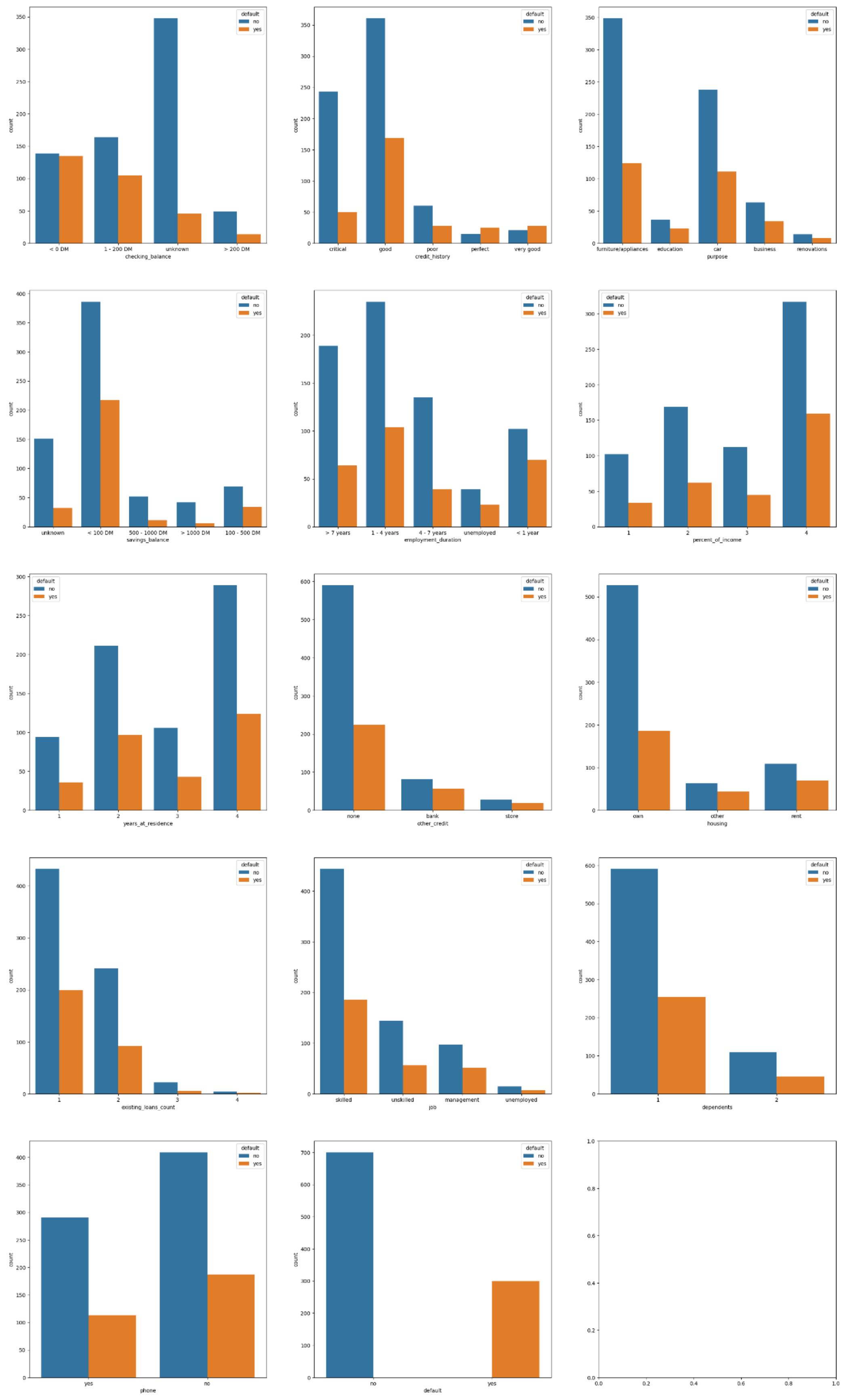

Box Plot of months_loan_duration
Violin Plot of months_loan_duration
Strip Plot of months_loan_duration
months_loan_duration
default
no
yes


Box Plot of percent_of_income
Violin Plot of percent_of_income
Strip Plot of percent_of_income
percent_of_income
default
no
yes

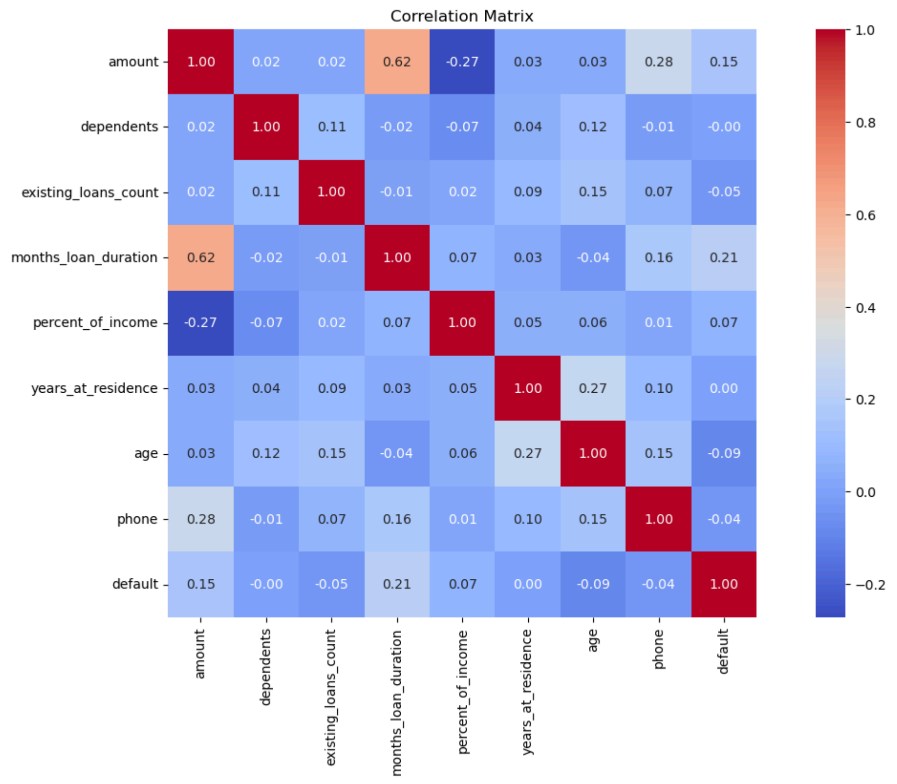




For categorical variables, the count plot can be made as below. This shows the variance and any pattern in data concerning default status.

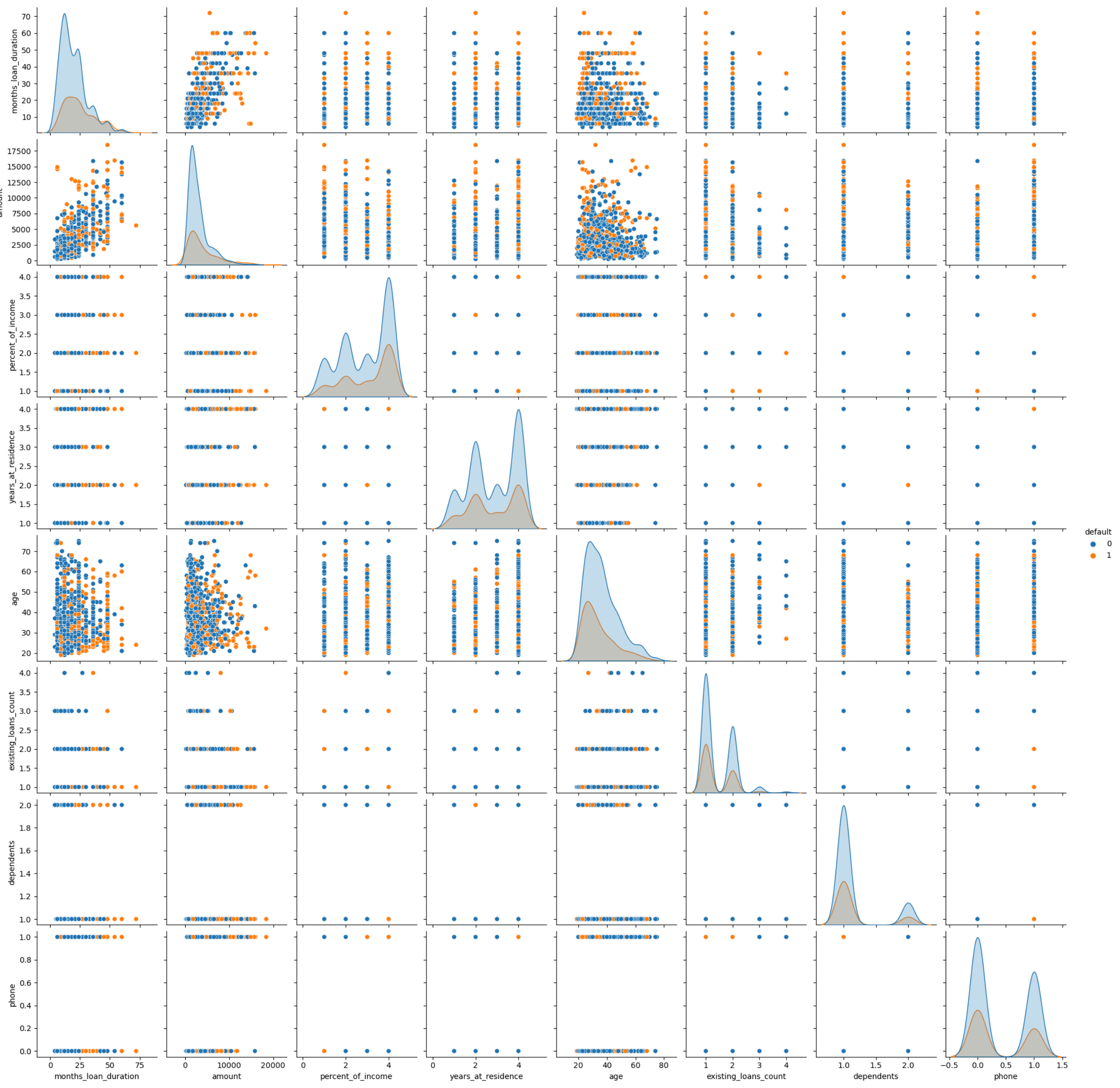

default
no
yes
count
< 0 DM
1 - 200 DM
unknown
> 200 DM
checking_balance
critical
good
poor
perfect
very good
credit_history
furniture/appliances
education
car
business
renovations
purpose
unknown
< 100 DM
500 - 1000 DM
> 1000 DM
100 - 500 DM
savings_balance
> 7 years
1 - 4 years
4 - 7 years
unemployed
< 1 year
employment_duration
percent_of_income
years_at_residence
none
bank
store
other_credit
own
other
rent
housing
existing_loans_count
skilled
unskilled
management
unemployed
job
dependents
yes
no
phone
no
yes
default

Correlation Matrix for numerical columns

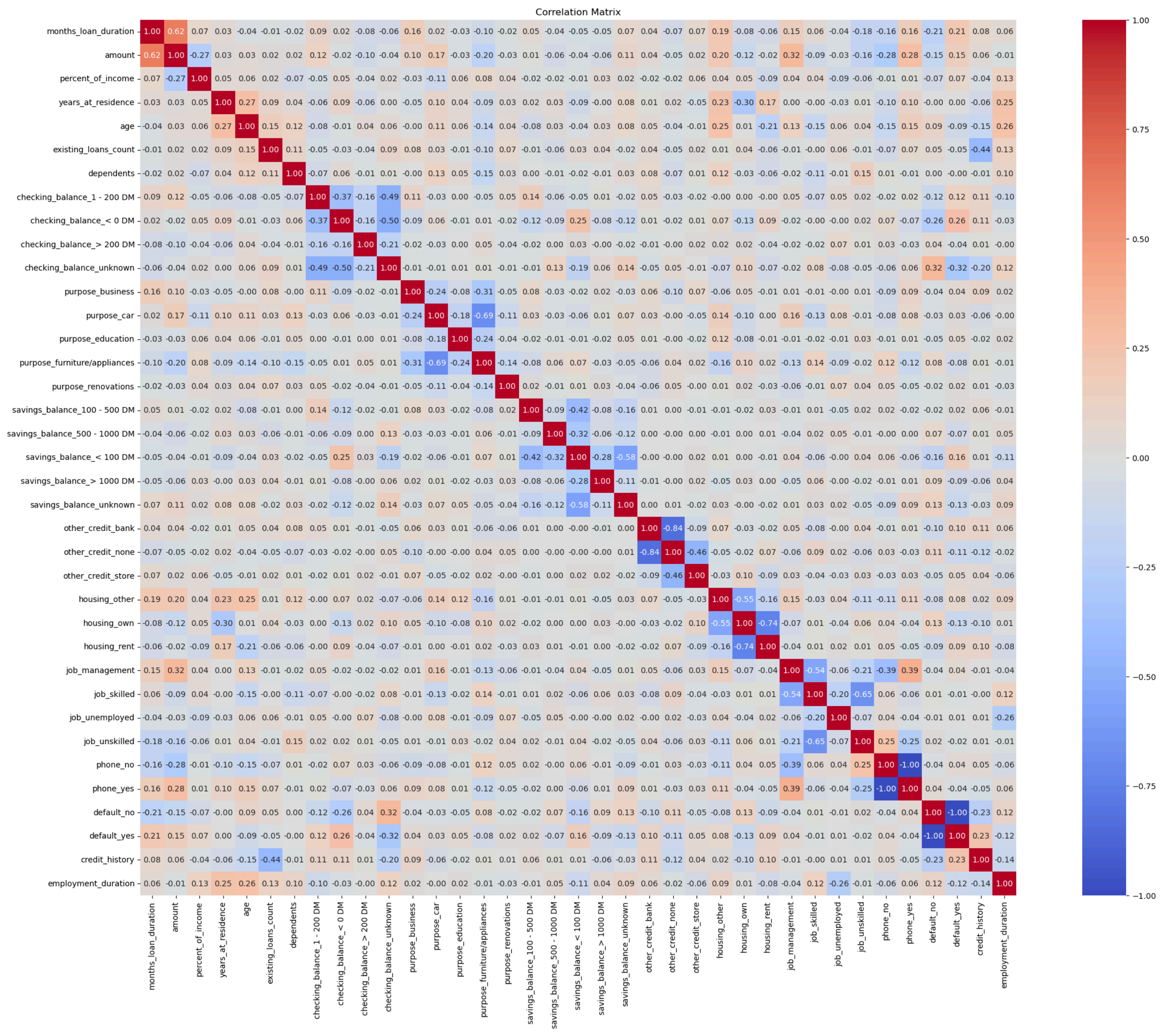


I can see a significant correlation for very few variables. I can see that the duration of the loan and amount are positively correlated (0.62), while the percentage of income is negatively correlated weakly(-0.27). For our target columns, I can see that except for dependences and years of residence, all other features are weekly correlated. Also, note that I have added phone and default status by encoding yes and no with one and zero respectively.

In analyzing the dataset, several patterns and trends have emerged regarding the characteristics of defaulters and non-defaulters. Defaulters tend to have a lower mean and variance in age distribution compared to non-defaulters. The loan amount for non-defaulters is typically around 2000, while the average loan duration is higher for defaulters. Interestingly, customers with a checking balance of '< 0 DM' show an equal distribution between defaulters and non-defaulters, indicating that default is more likely when customers have less money available in their checking accounts. Regarding credit history, customers with a 'perfect' or 'very good' credit history category have a higher proportion of defaulters, which is counterintuitive. For savings balance, there is a lower



default rate for customers with a very high savings balance (> 500 DM). Additionally, customers with an employment duration of less than 1 year have a higher default rate, and in general, the number of defaulters decreases with the number of years of employment, although this relationship is not purely linear. The percentage of income also plays a role, with customers who have a higher percentage of income (categories 1-4) more likely to default. Customers without any other credit have the least number of defaulters, while having more existing loans increases the likelihood of default, albeit with a small correlation. Customers who own their own house tend to have fewer defaulters, and skilled customers tend to take more loans. The number of dependents does not show a clear relationship with default rates. Interestingly, having a phone appears to reduce the number of defaulters, as indicated by both the plot and the correlation matrix. Finally, the dataset exhibits class imbalance, with the default class being significantly less balanced, highlighting the need to address class imbalance issues when developing predictive models. The same conclusion can be seen from below correlation matrix also. Note that here all categorical columns are encoded using one hot encoding.



Model Training and Hyperparameter Tuning

As for preprocessing the data, I followed several steps to prepare the data and train the models effectively. First, I used One hot encoding for categorical variables, ensuring that for each original column with k enum values, only k-1 columns were kept to increase interpretability and to reduce the effect of too many features (curse of dimensionality). I then split the data into a 20% test set and the remaining data for training the models. Additionally, I scaled the data separately for the training and test sets to remove any potential information leakage and improve the efficiency of the models.

In the process of building predictive models for loan default prediction, I explored several machine learning algorithms, including Logistic Regression, Linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Random Forest, Gradient Boosting, AdaBoost, XGBoost, and Catboost. I employed hyperparameter tuning using GridSearchCV and cross-validation on the training set to optimise their performance. The objective was to identify the best hyperparameters for each model while focusing on the 'recall' metric to minimise false negatives and improve the identification of potential loan defaulters.

Following hyperparameter tuning, I fitted each model with the optimal hyperparameters on the entire training set and evaluated their performance on both the training and testing sets. The evaluation metrics included precision, recall, and F1-score for each class, providing insights into the models' ability to correctly predict loan defaults. Additionally, I visualized confusion matrices to gain a deeper understanding of the model's predictions, including true positives, true negatives, false positives, and false negatives, further refining their performance assessment.

Results

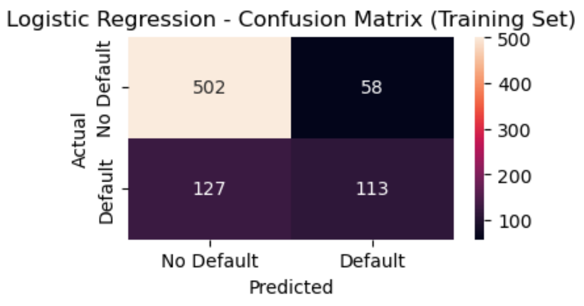
1. Logistic Regression

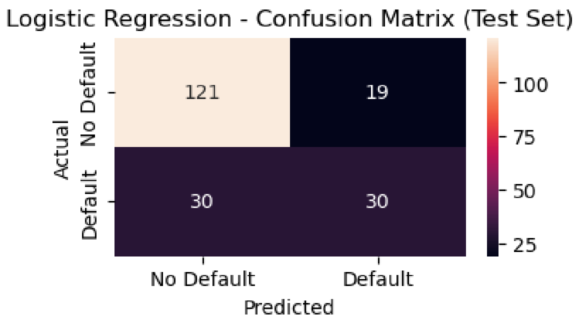
Logistic regression doesn’t have any hyperparameters, hence I directly trained the model with training data.

Training Set Performance	Test Set Performance
Accuracy: 0.76875	Accuracy: 0.755

```
              precision    recall  f1-score   support

           0       0.80      0.90      0.84       560
           1       0.66      0.47      0.55       240

    accuracy                           0.77       800
   macro avg       0.73      0.68      0.70       800
weighted avg       0.76      0.77      0.76       800
```

```
              precision    recall  f1-score   support

           0       0.80      0.86      0.83       140
           1       0.61      0.50      0.55        60

    accuracy                           0.76       200
   macro avg       0.71      0.68      0.69       200
weighted avg       0.74      0.76      0.75       200
```





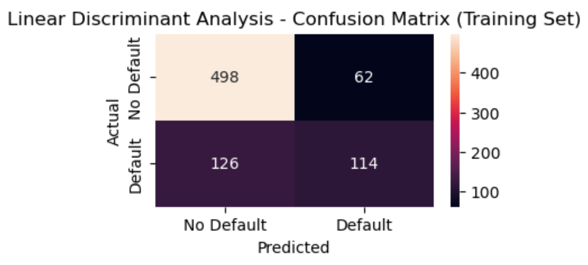
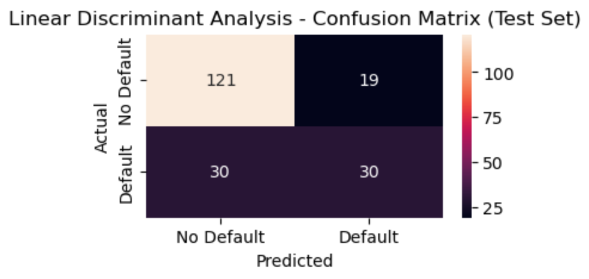
2. Linear discriminant analysis (LDA)

The coefficients for LDA are

```
{'months_loan_duration': 0.387637893319331855, 'amount':
0.23664033553123354, 'percent_of_income': 0.3345350531030353,
'years_at_residence': 0.011917164546244398, 'age': 0.018502508664861 33,
'existing_loans_count': 0.1057075280323715, 'dependents':
0.021889183125162 22, 'checking_balance_1 - 200 DM': 0.49732603055386115,
'checking_balance_< 0 DM': 0.720083831863462, 'checking_balance_> 200 DM':
0.19820170087286543, 'purpose_business': -0.20252406831318143,
'purpose_car': -0.22703634764793837, 'purpose_education':
0.11999153381662542, 'purpose_furniture/appliances': -0.33849761571699 64,
'savings_balance_100 - 500 DM': 0.24349078612134467, 'savings_balance_500 -
1000 DM': 0.08861645057251245, 'savings_balance_< 100 DM':
0.46462608917962955, 'savings_balance_> 1000 DM': -0.0030552684599530703,
'other_credit_bank': 0.2880587452611497, 'other_credit_store':
0.05940624182226663, 'housing_own': -0.0384436287307671 35, 'housing_rent':
```

```
0.2288607930691368, 'job_management': 0.04236474704832353, 'job_skilled':
0.1061136850556989, 'job_unskilled': 0.04249077837089162, 'credit_history':
0.42246562294865664, 'employment_duration': -0.24494620034461712, 'phone':
-0.15532218358506392}
```

Linear Discriminant Analysis (LDA) is a statistical method used for classification tasks. In LDA, the coefficients represent the weights assigned to each feature when predicting the class of a sample. In this case, the coefficients indicate the impact of each feature on the classification decision for loan default prediction. A positive coefficient suggests that an increase in that feature's value is associated with a higher likelihood of default, while a negative coefficient suggests the opposite. For example, a higher value for 'checking_balance_< 0 DM' has a strong positive impact on the prediction of default, indicating that customers with lower checking account balances are more likely to default. Conversely, 'credit_history' has a positive coefficient, suggesting that customers with a favourable credit history are more likely to default (I noticed the same in EDA which may indicate some issue in the data or mechanism by which credit history is calculated). Other features, such as 'housing_own' and 'purpose_car', have negative coefficients, indicating that owning a house or taking a loan for a car purchase is associated with a lower likelihood of default.

Training Set Performance	**Test Set Performance**
Accuracy: 0.765	Accuracy: 0.755
	

```
              precision    recall  f1-score   support

           0       0.80      0.89      0.84       560
           1       0.65      0.47      0.55       240

    accuracy                           0.77       800
   macro avg       0.72      0.68      0.69       800
weighted avg       0.75      0.77      0.75       800
```

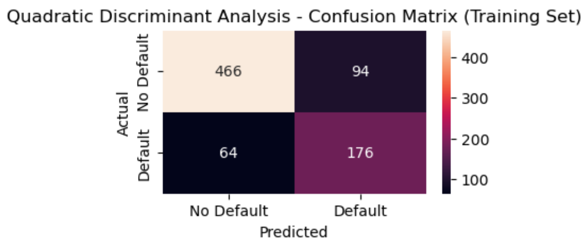
```
              precision    recall  f1-score   support

           0       0.80      0.86      0.83       140
           1       0.61      0.50      0.55        60

    accuracy                           0.76       200
   macro avg       0.71      0.68      0.69       200
weighted avg       0.74      0.76      0.75       200
```

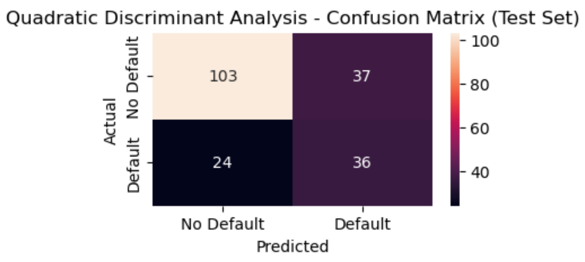
3. Quadratic discriminant analysis (QDA)

Training Set Performance	Test Set Performance
Accuracy: 0.8025	Accuracy: 0.695

```
              precision    recall  f1-score   support

           0       0.88      0.83      0.86       560
           1       0.65      0.73      0.69       240

    accuracy                           0.80       800
   macro avg       0.77      0.78      0.77       800
weighted avg       0.81      0.80      0.81       800
```

```
              precision    recall  f1-score   support

           0       0.81      0.74      0.77       140
           1       0.49      0.60      0.54        60

    accuracy                           0.69       200
   macro avg       0.65      0.67      0.66       200
weighted avg       0.72      0.69      0.70       200
```



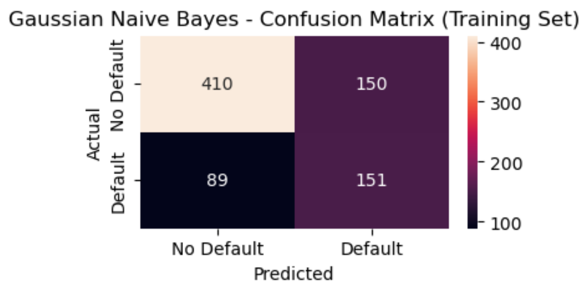
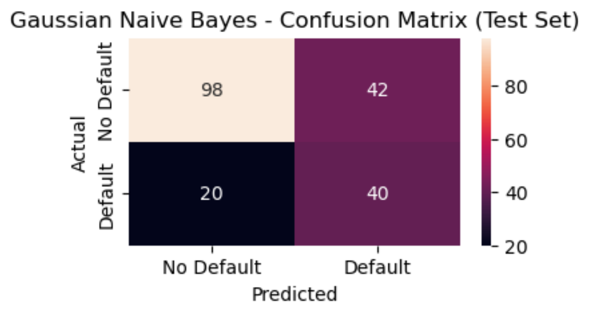


4. Gaussian Naive Bayes

Training Set Performance	Test Set Performance
Accuracy: 0.70125	Accuracy: 0.69

```
              precision    recall  f1-score   support

           0       0.82      0.73      0.77       560
           1       0.50      0.63      0.56       240

    accuracy                           0.70       800
   macro avg       0.66      0.68      0.67       800
weighted avg       0.73      0.70      0.71       800
```

```
              precision    recall  f1-score   support

           0       0.83      0.70      0.76       140
           1       0.49      0.67      0.56        60

    accuracy                           0.69       200
   macro avg       0.66      0.68      0.66       200
weighted avg       0.73      0.69      0.70       200
```

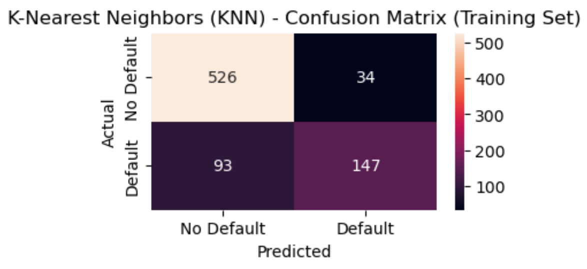
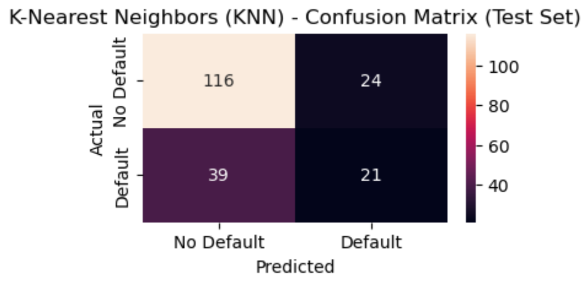
	

5. K-Nearest Neighbors

Training Set Performance	Test Set Performance
Accuracy: 0.84125	Accuracy: 0.685

```
              precision    recall  f1-score   support

           0       0.85      0.94      0.89       560
           1       0.81      0.61      0.70       240

    accuracy                           0.84       800
   macro avg       0.83      0.78      0.80       800
weighted avg       0.84      0.84      0.83       800
```

```
              precision    recall  f1-score   support

           0       0.75      0.83      0.79       140
           1       0.47      0.35      0.40        60

    accuracy                           0.69       200
   macro avg       0.61      0.59      0.59       200
weighted avg       0.66      0.69      0.67       200
```

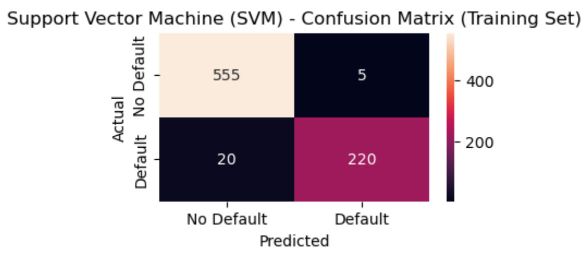
	

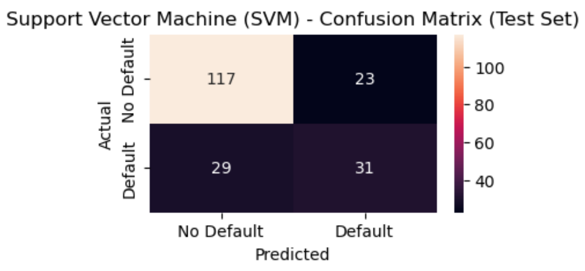
6. Support Vector Machines

Training Set Performance	Test Set Performance
Accuracy: 0.96875	Accuracy: 0.74

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       560
           1       0.98      0.92      0.95       240

    accuracy                           0.97       800
   macro avg       0.97      0.95      0.96       800
weighted avg       0.97      0.97      0.97       800
```

```
              precision    recall  f1-score   support

           0       0.80      0.84      0.82       140
           1       0.57      0.52      0.54        60

    accuracy                           0.74       200
   macro avg       0.69      0.68      0.68       200
weighted avg       0.73      0.74      0.74       200
```



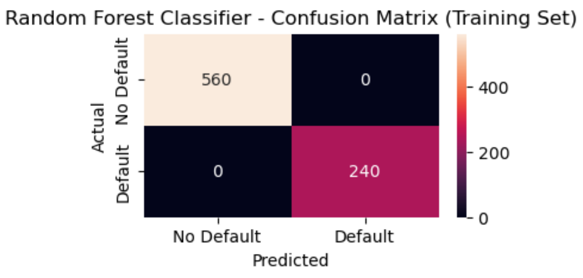


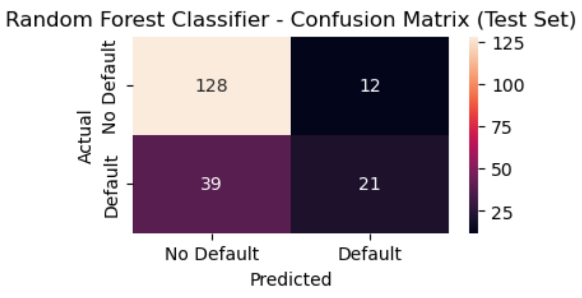
7. Random Forest

Training Set Performance	Test Set Performance
Accuracy: 1.0	Accuracy: 0.745

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       560
           1       1.00      1.00      1.00       240

    accuracy                           1.00       800
   macro avg       1.00      1.00      1.00       800
weighted avg       1.00      1.00      1.00       800
```

```
              precision    recall  f1-score   support

           0       0.77      0.91      0.83       140
           1       0.64      0.35      0.45        60

    accuracy                           0.74       200
   macro avg       0.70      0.63      0.64       200
weighted avg       0.73      0.74      0.72       200
```



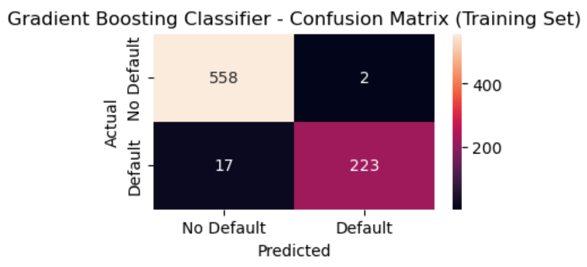
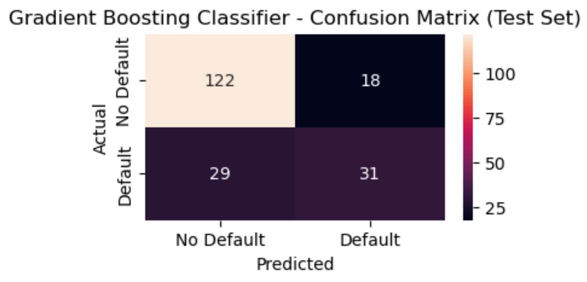


8. Gradient Boosting

Training Set Performance	Test Set Performance
Accuracy: 0.97625	Accuracy: 0.765

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       560
           1       0.99      0.93      0.96       240

    accuracy                           0.98       800
   macro avg       0.98      0.96      0.97       800
weighted avg       0.98      0.98      0.98       800
```

```
              precision    recall  f1-score   support

           0       0.81      0.87      0.84       140
           1       0.63      0.52      0.57        60

    accuracy                           0.77       200
   macro avg       0.72      0.69      0.70       200
weighted avg       0.76      0.77      0.76       200
```

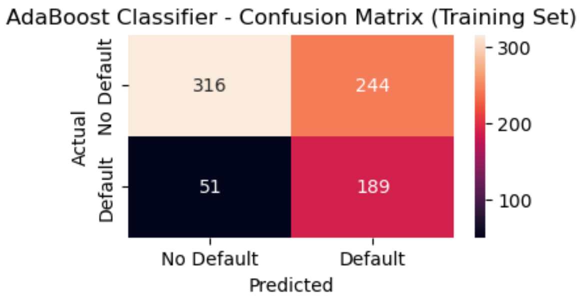
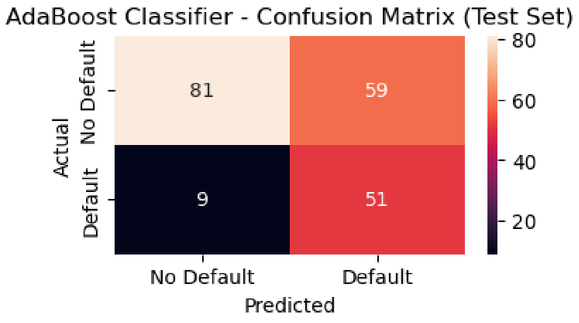
	

9. AdaBoost

Training Set Performance	Test Set Performance
Accuracy: 0.63125	Accuracy: 0.66

```
              precision    recall  f1-score   support

           0       0.86      0.56      0.68       560
           1       0.44      0.79      0.56       240

    accuracy                           0.63       800
   macro avg       0.65      0.68      0.62       800
weighted avg       0.73      0.63      0.65       800
```

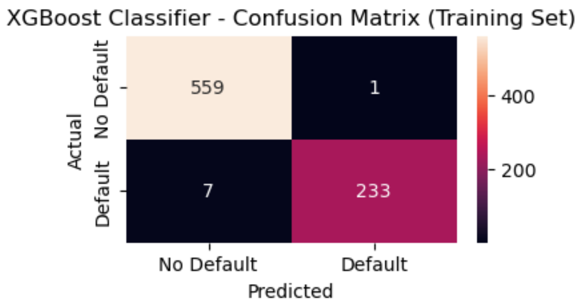
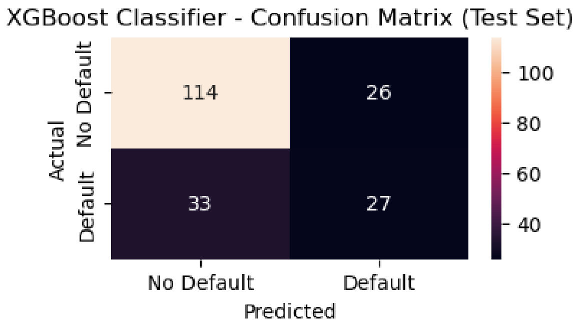
```
              precision    recall  f1-score   support

           0       0.90      0.58      0.70       140
           1       0.46      0.85      0.60        60

    accuracy                           0.66       200
   macro avg       0.68      0.71      0.65       200
weighted avg       0.77      0.66      0.67       200
```

10. XGBoost

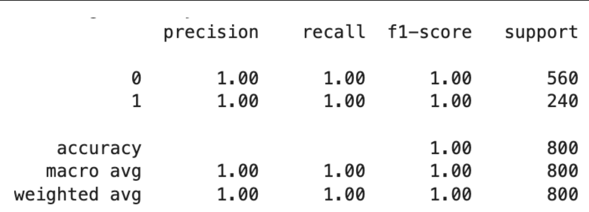
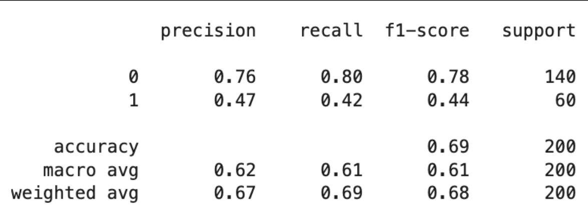
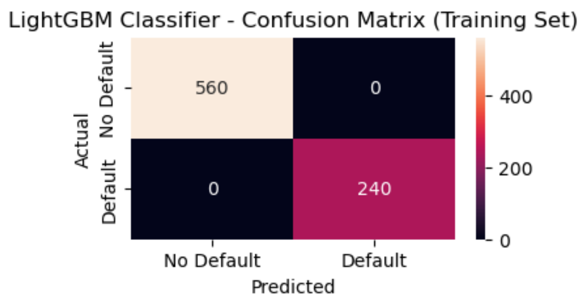
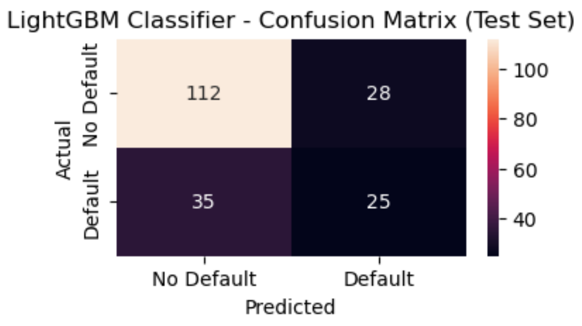
Training Set Performance	**Test Set Performance**
Accuracy: 0.99	Accuracy: 0.705

`              precision    recall  f1-score   support`
`           0       0.99      1.00      0.99       560`
`           1       1.00      0.97      0.98       240`
`    accuracy                           0.99       800`
`   macro avg       0.99      0.98      0.99       800`
`weighted avg       0.99      0.99      0.99       800`
	`              precision    recall  f1-score   support`
	`           0       0.78      0.81      0.79       140`
	`           1       0.51      0.45      0.48        60`
	`    accuracy                           0.70       200`
	`   macro avg       0.64      0.63      0.64       200`
	`weighted avg       0.70      0.70      0.70       200`

11. LightGBM

Training Set Performance	**Test Set Performance**
Accuracy: 1.0	Accuracy: 0.685

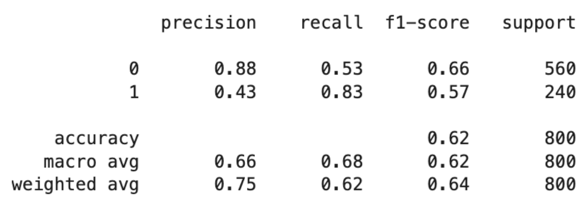
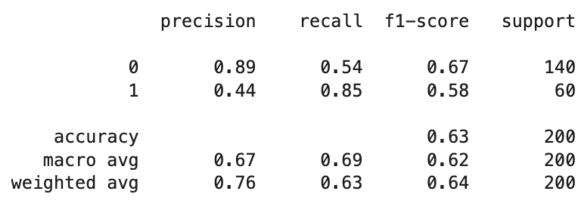
	
	

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       560
           1       1.00      1.00      1.00       240

    accuracy                           1.00       800
   macro avg       1.00      1.00      1.00       800
weighted avg       1.00      1.00      1.00       800
```

```
              precision    recall  f1-score   support

           0       0.76      0.80      0.78       140
           1       0.47      0.42      0.44        60

    accuracy                           0.69       200
   macro avg       0.62      0.61      0.61       200
weighted avg       0.67      0.69      0.68       200
```

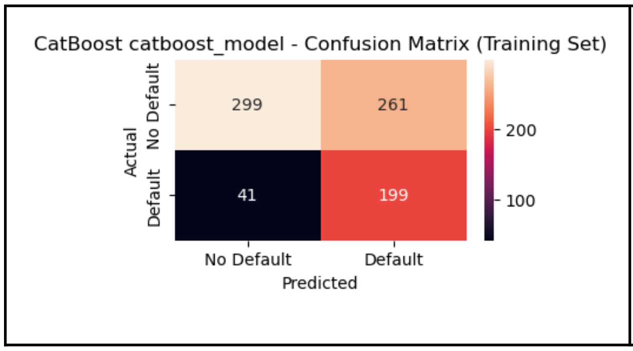
12. Catboost

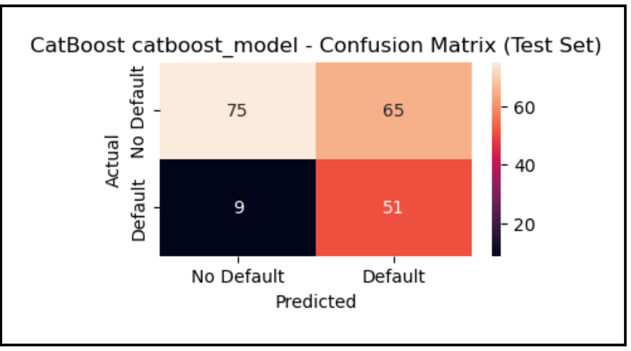
For Catboost, the categorical variables need to be used as such without the one-hot encoding to increase the performance. Catboost performs better with Categorical variables than with numerical variables. Catboost model uses scale_pos_weight to handle Imbalanced Data. This parameter adjusts the cost of misclassifying positive examples. A good default value is the ratio of negative to positive samples in the training dataset. This could be a reason why this model shows much better recall than other models and even notably higher accuracy in the test set than the training set.

Training Set Performance	**Test Set Performance**
Accuracy: 0.6225	Accuracy: 0.63
	

```
              precision    recall  f1-score   support

           0       0.88      0.53      0.66       560
           1       0.43      0.83      0.57       240

    accuracy                           0.62       800
   macro avg       0.66      0.68      0.62       800
weighted avg       0.75      0.62      0.64       800
```

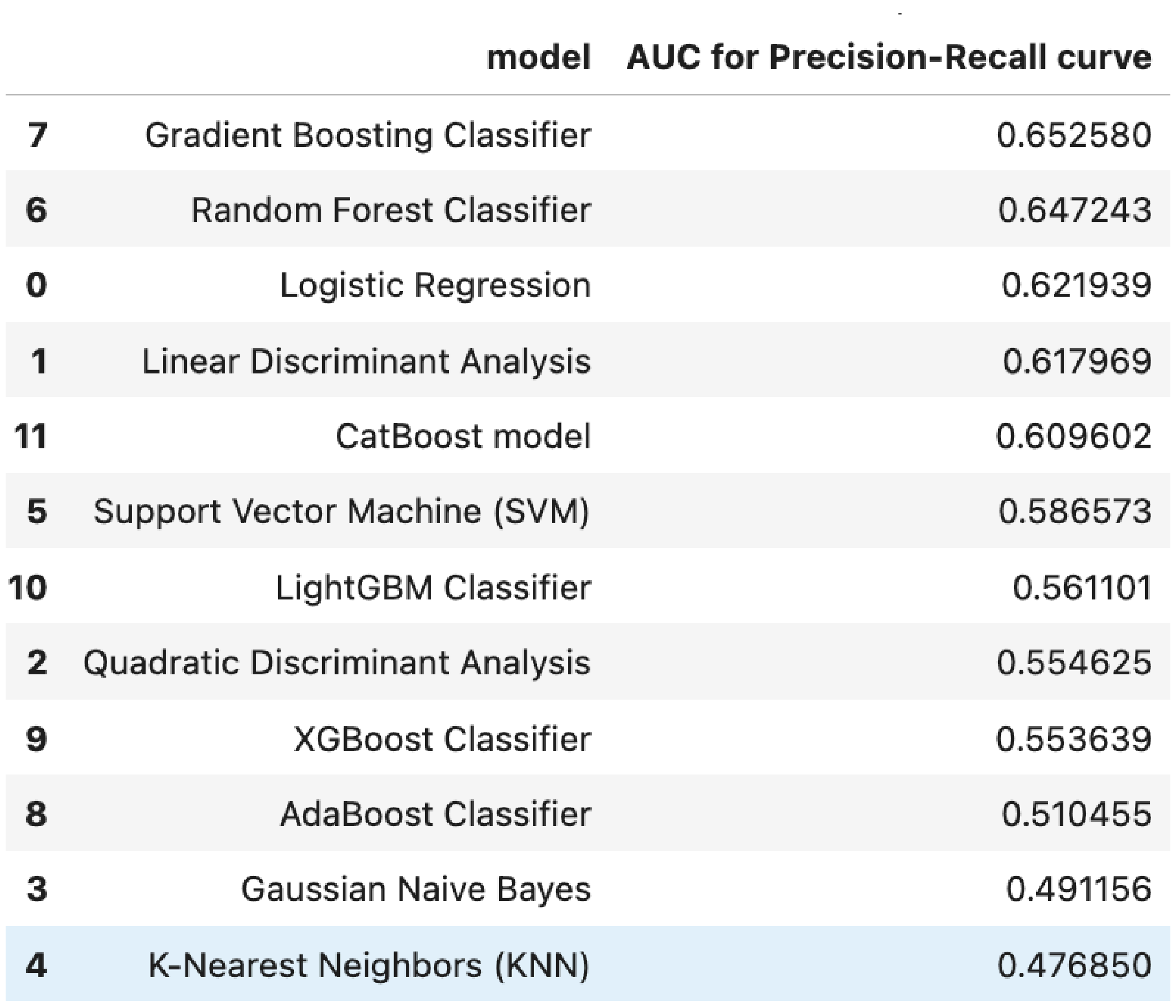
```
              precision    recall  f1-score   support

           0       0.89      0.54      0.67       140
           1       0.44      0.85      0.58        60

    accuracy                           0.63       200
   macro avg       0.67      0.69      0.62       200
weighted avg       0.76      0.63      0.64       200
```



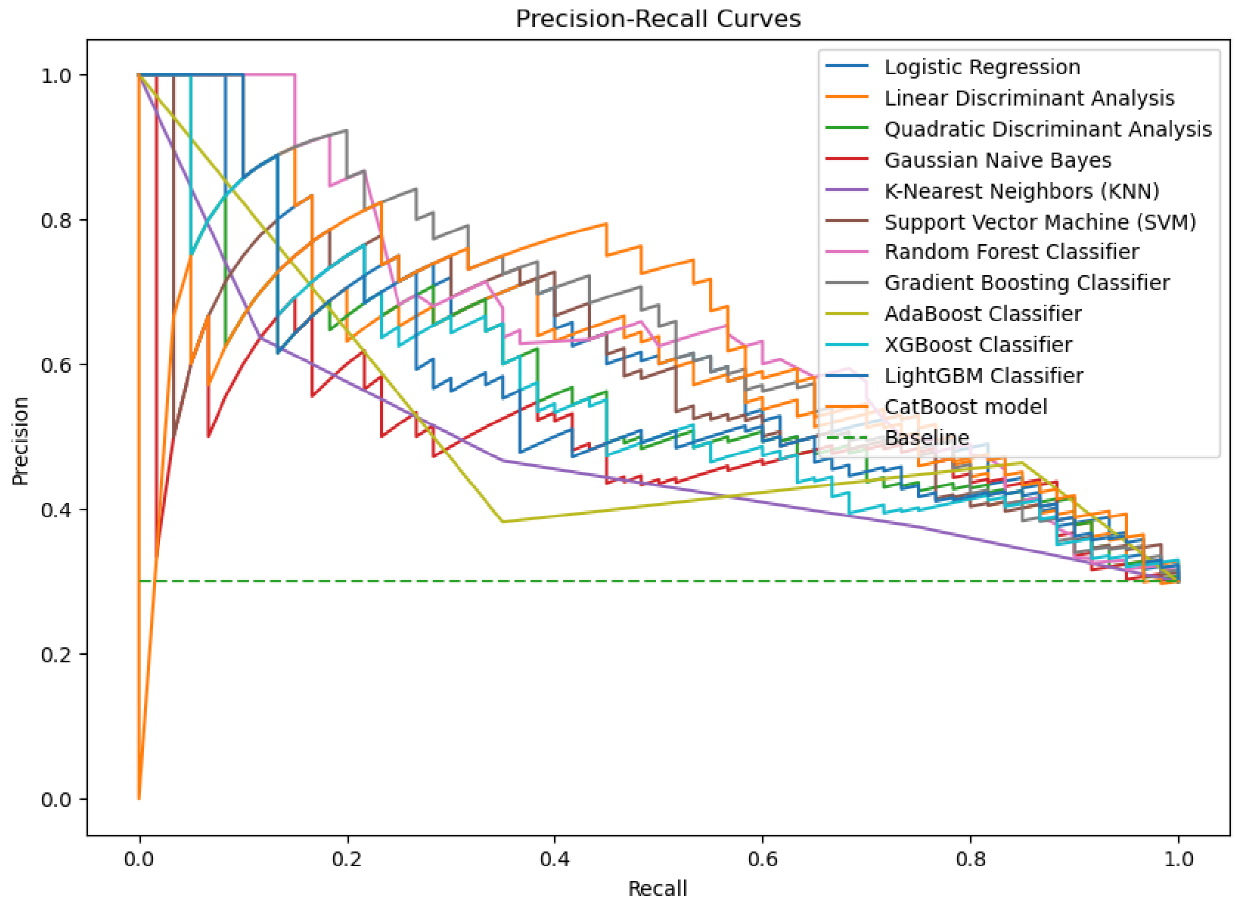


Selecting the Final Model

Given the imbalanced nature of the 'default' column, with significantly more 'no' (0) values than 'yes' (1) values, calculating the Area Under the Curve (AUC) for Precision-Recall (PR) curves is a sensible approach for model evaluation. PR curves are often more informative than ROC curves for imbalanced datasets, especially when the positive class is the minority class, as in this case. By considering varying class probability decision thresholds, this approach provides a more nuanced view of the model's performance, particularly its ability to identify defaulters while minimising false positives correctly.

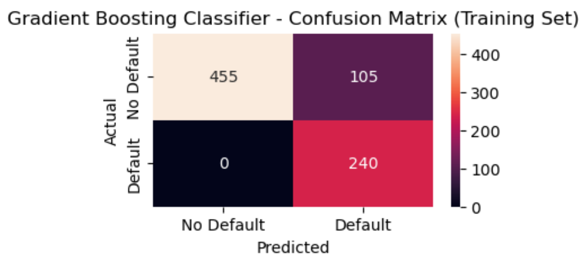


	model	AUC for Precision-Recall curve
7	Gradient Boosting Classifier	0.652580
6	Random Forest Classifier	0.647243
0	Logistic Regression	0.621939
1	Linear Discriminant Analysis	0.617969
11	CatBoost model	0.609602
5	Support Vector Machine (SVM)	0.586573
10	LightGBM Classifier	0.561101
2	Quadratic Discriminant Analysis	0.554625
9	XGBoost Classifier	0.553639
8	AdaBoost Classifier	0.510455
3	Gaussian Naive Bayes	0.491156
4	K-Nearest Neighbors (KNN)	0.476850



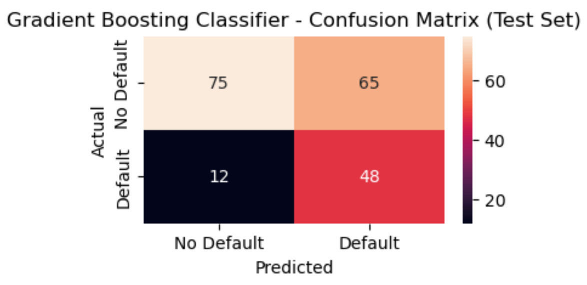
Training Set Performance(Final Model)

```
              precision    recall  f1-score   support

           0       1.00      0.81      0.90       560
           1       0.70      1.00      0.82       240

    accuracy                           0.87       800
   macro avg       0.85      0.91      0.86       800
weighted avg       0.91      0.87      0.87       800
```



Test Set Performance(Final Model)

```
              precision    recall  f1-score   support

           0       0.86      0.54      0.66       140
           1       0.42      0.80      0.55        60

    accuracy                           0.61       200
   macro avg       0.64      0.67      0.61       200
weighted avg       0.73      0.61      0.63       200
```



Discussion

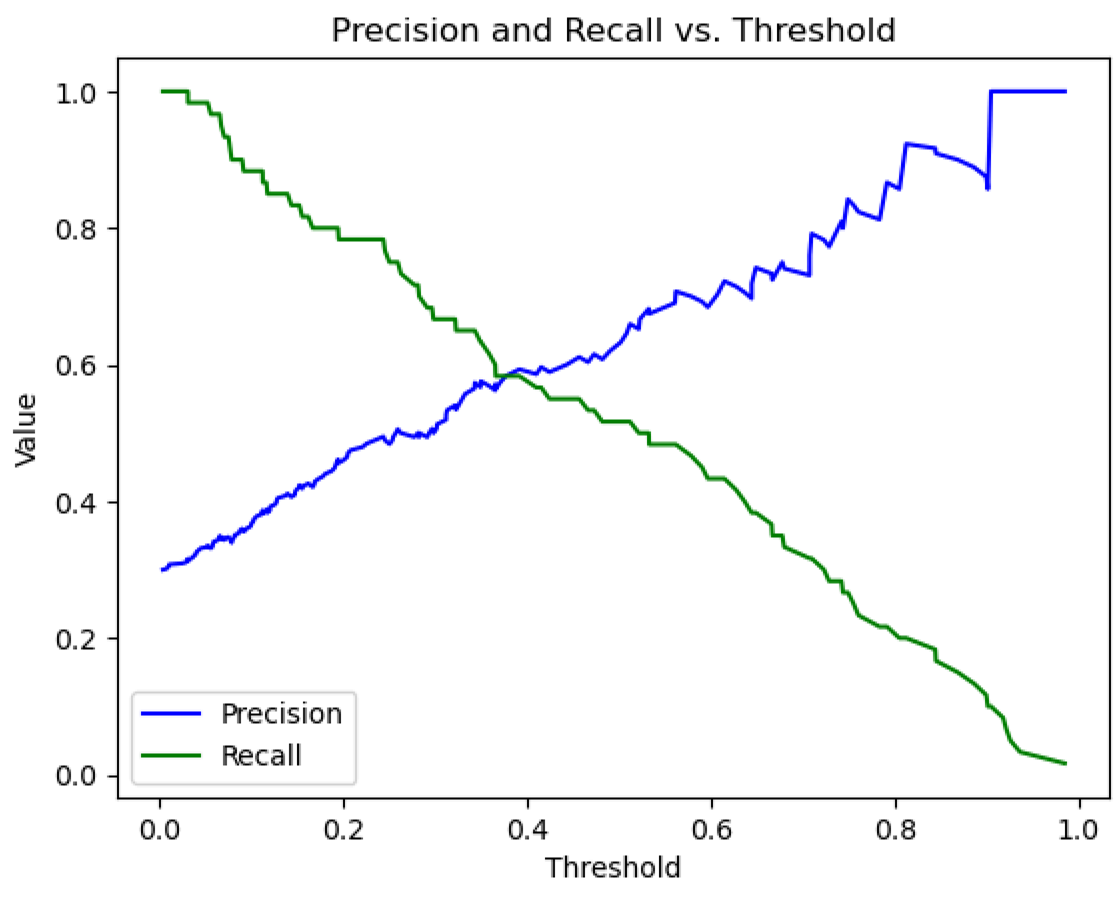
The analysis of the coefficients for LDA reveals several key insights into the factors influencing loan default. A checking balance of less than 200 DM indicates a higher likelihood of default, whereas a savings balance exceeding 1000 DM suggests a lower risk. However, a savings balance below 100 DM may increase the risk of default. Positive coefficients for credit history and various loan purposes like business, car, and furniture/appliances indicate lower default probabilities. Longer employment durations and owning a house are associated with a higher likelihood of loan repayment. These findings highlight the multifaceted nature of loan default determinants, emphasizing the importance of considering various factors in predicting default risk. Most factors have a small effect and please note that I have only mentioned the top and bottom few. This same observation matches with the EDA we have done in the first step.

We can see that LDA performed better than QDA. This might be because the covariance matrix seems to be close to each other. Another reason might be because of high dimensional data with comparatively less training data resulting in QDA underfitting.

Based on the evaluation of various machine learning models for predicting loan default, the Gradient Boosting Classifier emerged as the top performer with an AUC score of 0.652580 for the Precision-Recall curve. This indicates its superior ability to balance precision and recall, crucial for a reliable loan default prediction system. Following closely behind, the Random Forest Classifier and Logistic Regression also demonstrated strong performance, with AUC scores of 0.647243 and 0.621939 respectively. These models exhibit promising potential for accurately identifying customers at risk of defaulting on loans. Other models which have performed quite well are LDA and Catboost which have AUC scores of 0.617969 and 0.609602 respectively.

In contrast, the Gaussian Naive Bayes and K-Nearest Neighbors (KNN) models performed the least effectively, with AUC scores below 0.5, suggesting that these models are not best for this specific prediction task. The moderate performers, including Support Vector Machine (SVM), LightGBM Classifier, Quadratic Discriminant Analysis, XGBoost Classifier, and AdaBoost Classifier, achieved AUC scores ranging from 0.554625 to 0.586573. While these models showed some capability in predicting loan defaults, they fell short compared to the top performers.

Based on these findings, the Gradient Boosting Classifier stands out as the most suitable model for predicting loan defaults in this dataset. Further refinement and optimization of this model could potentially enhance its performance and reliability, providing financial institutions with a powerful tool for assessing and managing loan default risks.



The Precision and recall vs threshold for Gradient Boosting Classifier

To improve the Gradient Boosting Classifier model for loan default prediction, I selected a threshold of 0.1664 to achieve a desired recall of 0.80. Also note that the F1-score maximises at a threshold of 0.34 which is not too far away from the selected threshold to reach 80% recall for class ‘yes’ (0). This adjustment maximized the model's ability to correctly identify default cases, reducing the risk of false negatives. By prioritizing recall, the model ensures that fewer instances of actual defaults are missed, ultimately minimizing the bank's risk. This approach maintains a reasonable level of precision, making the model a robust tool for identifying customers at high risk of default.

However, several avenues for further improvement and exploration exist. The small size of the dataset and class imbalance could potentially limit the generalizability of the model to larger populations. Future efforts should focus on collecting more extensive and diverse datasets to enhance the model's robustness and applicability. Additionally, incorporating advanced feature engineering techniques like PCA to reduce dimensionality and external data sources could further enrich the predictive power of the model and uncover hidden patterns within the data.

Conclusions

In conclusion, this project represents a significant step towards developing a robust and reliable model for loan default prediction in the banking sector. The thorough exploration of various machine learning models and meticulous tuning of hyperparameters has led to the identification of Gradient Boosting as a highly effective tool for this task. By prioritizing recall and achieving a balance between precision and recall through custom thresholding, the model demonstrates a strong ability to identify potential loan defaulters while keeping false negatives low.

Furthermore, while Gradient Boosting excels in predictive performance, its complexity limits interpretability. Exploring simpler models like Logistic Regression, which also performed well in this project, could provide more insights into the factors influencing loan default predictions. Overall, this project underscores the importance of leveraging machine learning in financial risk assessment and highlights the potential for further advancements in the field to improve the accuracy and reliability of loan default prediction models.

By researching and evaluating various models, including both complex ensemble methods like Gradient Boosting and simpler models like Logistic Regression, I have gained valuable insights into the trade-offs between model complexity and performance. This research has highlighted the need to match the model to the specific use case, as each model performs differently based on the characteristics of the data and the objectives of the task. It has also demonstrated that there is no one-size-fits-all solution in machine learning, emphasizing the need for careful consideration and experimentation to identify the best-performing model for a given problem. Overall, this project has been a valuable learning experience, providing insights into the complexities of model selection, hyperparameter tuning, preprocessing the data and the importance of balancing performance with interpretability in machine learning applications.