Report on Loan Default Prediction and Credit Risk Modelling: Analysis and Insights for Company ABC

Data Understanding

The dataset we used for this analysis consists of 6,000 rows and 21 columns, containing a mix of both categorical and numerical data. It includes various features related to loan applications and borrower details, which are crucial for evaluating credit risk. One of the key columns, **Total Collateral Value**, had 62% missing data, which we addressed during the preprocessing phase to ensure a clean dataset for analysis.

To prepare the data for modelling, we handled missing values, encoded categorical variables, and scaled numerical features as needed. This made the dataset ready for analysis using machine learning techniques. We used **Python** and its **machine learning** libraries to build and evaluate the model, helping us gain insights into credit risk and loan default prediction.

Objective And Outcome

1. Identify the top 5 insights from the data that are most relevant to explaining the default trends. These insights will help the company understand the factors influencing loan defaults

➢ USED CHI SQUARE TEST TO FIND THE SIGNIFICANCE WITH THE TARGET VARIABLE.

- **Fresh Top Up and Loan Defaults**: The "fresh top up" column is significantly related to the target variable "good_bad_loan," suggesting that loans with fresh top-up amounts may have a higher probability of default. This indicates that additional loan amounts could be linked to higher financial strain, increasing the likelihood of default.

- **Impact of the COVID Period**: The COVID period also shows significance with the "good_bad_loan" variable, implying that loans taken or managed during the pandemic are more prone to defaults. This could be due to economic instability, job losses, or other financial stressors during that time.

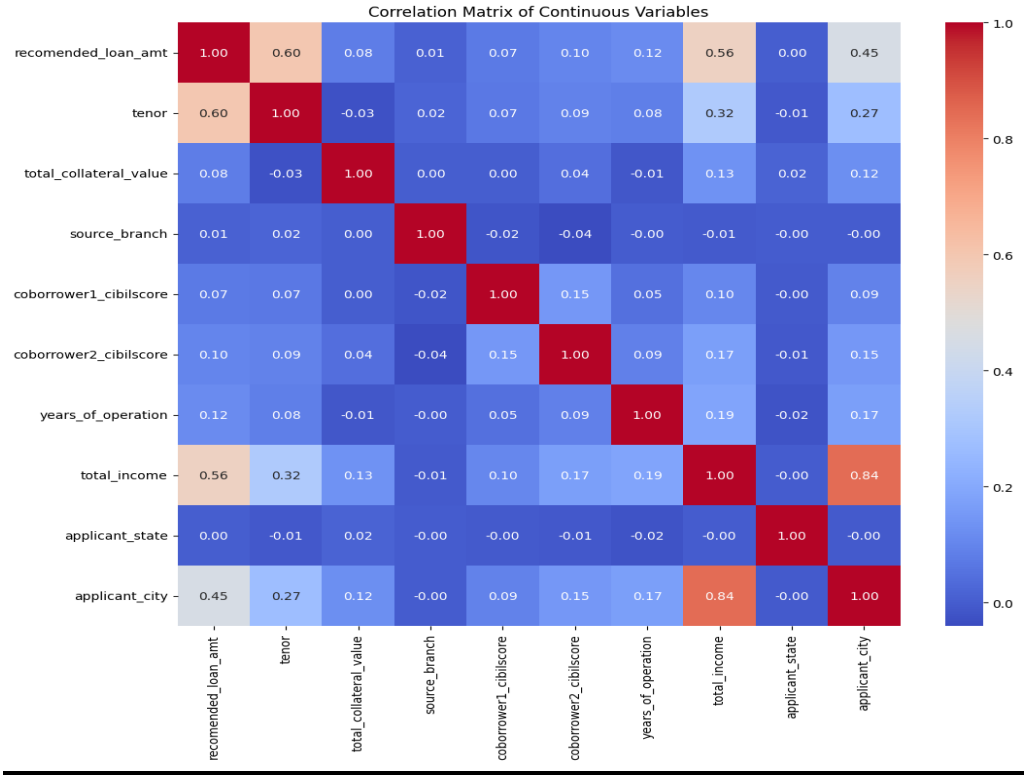
```
Column: fresh_topup
Chi2: 5.3709
P-value: 0.0205
Significant relationship with good_bad_loan

Column: covid_period_default_
Chi2: 304.7628
P-value: 0.0000
Significant relationship with good_bad_loan
```

➢ USED CORRELATION MATRIX TO FIND OUT THE RELATION BETWEEN CONTINOUS VARIABLES.

• **Recommended Loan Amount and Loan Tenure (60% correlation)**:
This moderate positive correlation suggests that, generally, as the loan tenure increases, the recommended loan amount also tends to increase. This makes sense because longer tenures often accommodate higher loan amounts, as they spread the repayment over a longer period. It also indicates that loan amounts and tenures are somewhat interdependent, and larger loans are typically recommended for longer terms.

• **Recommended Loan Amount and Total Income (56% correlation)**:
This moderate positive correlation shows that borrowers with higher total incomes tend to be recommended higher loan amounts. Lenders might base the recommended loan amount on an applicant's income to ensure that the borrower can repay the loan. However, it's important to assess if the loan amount is proportionate to the borrower's income to avoid overburdening borrowers, which could lead to defaults.

• **Applicant's City and Total Income (84% correlation)**:
A strong positive correlation here indicates that an applicant's income is highly influenced by their geographic location. Applicants from certain cities may have higher average incomes, perhaps due to local economic conditions, industries, or living costs. This could suggest that income levels are not uniform across the dataset, and applicants in different cities may have varying financial capabilities, influencing their ability to repay loans.



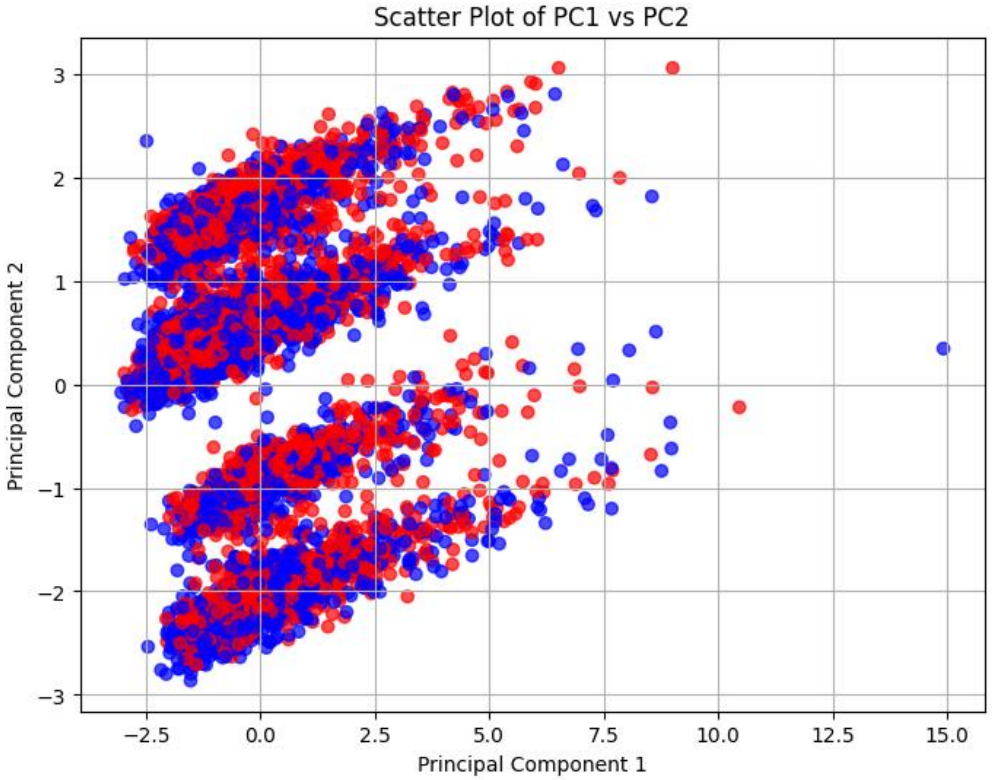
- **The RFE (Recursive Feature Elimination) method has been applied to identify the most important features by ranking them**

The following features are most influential:

Feature	Ranking
coborrower2_designation_ordinal	1
recomended_loan_amt	1
tenor	1
total_collateral_value	1
coborrower1_designation_ordinal	1
source_branch	1
total_income	1
applicant_city	1
coborrower1_cibilscore	1
coborrower2_cibilscore	1

- **CHECKING FOR LINEAR RELATIONSHIP**

The scatter plot of PC1 vs PC2 shows significant overlapping between the two classes, indicating a lack of linearity in the data. The heavily overlapping pattern suggests the principal components are not effectively separating the classes, which means the relationship between the variables does not have a linear structure.



2. Develop a credit risk model that assigns a risk score to each new loan application. This model will help the company determine whether to approve or reject an application based on historical trends and patterns observed in the data.

I started developing the credit risk model using a **logistic regression** approach, which gave me an **F1 score of 64%**. To see if I could improve the results, I tried using a **Random Forest** model, but it only reached **59% accuracy**. Despite trying some feature engineering techniques, I didn't see any significant improvements.

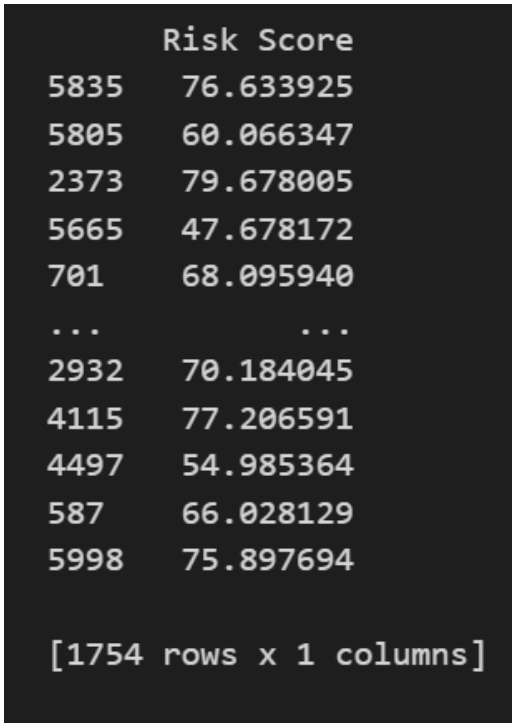
As I was working through this, I began to notice some potential **overlap between the variables** in my data, which could be affecting the model's performance. To dig deeper, I decided to apply **Principal Component Analysis (PCA)** to check for any relationships between the variables. This analysis revealed that there was indeed **overlap** and **no significant linearity** between variables, which could be limiting the model's ability to improve.

After considering these findings, I decided to stick with the **logistic regression model** because it offered clear insights into the data and its results. I then focused on using **odds ratio** and **credit risk scores** to better interpret and assign risk to loan applications. While accuracy improvements were limited by the nature of the data, this approach allowed me to create a model that was both practical and meaningful in evaluating credit risk.

This is the code used and logic behind it.

```
def calculate_risk_score(input_data, model, features):
    # Ensure input data has the same columns as the features used in the model
    input_data = input_data[features]

    # Add the constant (intercept) to the input data for prediction
    input_data_sm = sm.add_constant(input_data)

    # Calculate the log odds (predicted logit)
    log_odds = model.predict(input_data_sm)

    # Calculate the odds ratio (exponentiate the log odds)
    odds_ratio = np.exp(log_odds)

    # Normalize the odds ratio to range [0, 100]
    # Min-Max scaling of odds ratio to [0, 100]
    min_odds = odds_ratio.min()
    max_odds = odds_ratio.max()
    risk_scores = 100 * (odds_ratio - min_odds) / (max_odds - min_odds)

    # Return the risk scores
    return risk_scores
```

Given below is the output where we can see risk score of test set.



	Risk Score
5835	76.633925
5805	60.066347
2373	79.678005
5665	47.678172
701	68.095940
...	...
2932	70.184045
4115	77.206591
4497	54.985364
587	66.028129
5998	75.897694

[1754 rows x 1 columns]

METHODOLOGY

Data Cleaning and Preprocessing

The initial phase of the analysis involved data preparation to ensure the dataset was suitable for statistical analysis and machine learning. The following steps were undertaken:

1. **Renaming Columns**: Renamed columns for better understanding and interpretability.
2. **Handling Missing Values**: Identified missing values and filled them with zero.
3. **Converting Categorical Variables**: Transformed categorical variables into numerical representations.
4. **Removing Unnecessary Columns**: Removed irrelevant columns to streamline the dataset.

Encoding and Feature Transformation

Following preprocessing, categorical variables were encoded to create dummy variables. This transformation was necessary for handling categorical data in the models.

Model Development and Evaluation

To develop a predictive model, the dataset was split into training and testing subsets, with 70% allocated to training and 30% to testing. The following models were evaluated:

Logistic Regression

1. **Initial Model**: A logistic regression model was fitted to the data.The F1 score obtained was **64%**.
2. **Multicollinearity Check**: Variance Inflation Factor (VIF) analysis was conducted to check for multicollinearity among features.
3. **Feature Engineering**: Tried improving model performance by creating new features, but it did not lead to a significant increase in accuracy.

Random Forest

1. **Model Performance**: A Random Forest model was also evaluated but yielded an accuracy of only **59%**.
2. **Decision: Although Random Forest was tried, its performance was not satisfactory. Logistic regression was chosen instead because it performed better in comparison.**

Calculating Credit risk

I calculated the credit risk score by first determining the **log-odds** using the logistic regression model. The log-odds represent the predicted likelihood of a loan default based on various factors. Then, I converted these log-odds to **odds ratios** by exponentiating the values, which provided a clearer understanding of the relative risk of default for each loan application. Finally, I used the odds ratios to derive the **credit risk score**, which was normalized to a scale of 0 to 100, making it easier to interpret and assign risk to different loan applications.

Conclusion

The analysis faced challenges in achieving high predictive accuracy due to overlapping and nonlinear relationships among variables. Despite this, logistic regression proved to be an effective model, with an F1 score of 64%, indicating a balanced prediction of loan defaults. The use of odds ratios and credit risk scoring, normalized to a scale of 0-100 (with higher scores indicating higher risk), provided valuable insights, helping to interpret the influence of different factors on credit risk. While the model's accuracy could be improved, these methods offered a clear understanding of the key drivers behind loan defaults, enhancing the decision-making process in lending.