Final ML Case Study Report

---

1. Objectives

To build a machine learning pipeline that accurately predicts loan approval status based on features like income, loan amount, credit history, and applicant details. The aim is to support financial institutions in making quick, data-driven lending decisions.

---

2. The Data

You used a dataset named:
home_loan_dataset_300k_modified.csv

Each row in the data represents one person who applied for a loan. The columns contain information like:

- **Their income**
- **How much loan they want**
- **Property value**
- **Credit history score**
- **Loan default history**
- **Type of property**
- **Whether the loan was approved or rejected (Loan_Status)**

3. EDA Observations

- **`Credit_History` and `LoanAmount` showed the strongest relationship with `Loan_Status`.**

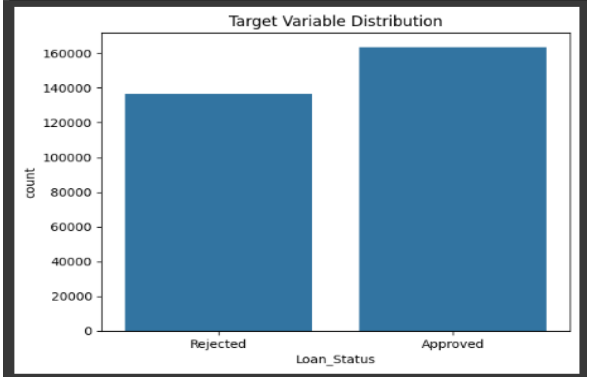
- **`LoanAmount` was right-skewed, so we applied a log transformation.**

- **Missing values were found in `LoanAmount`, `Self_Employed`, etc., and handled via median imputation.**

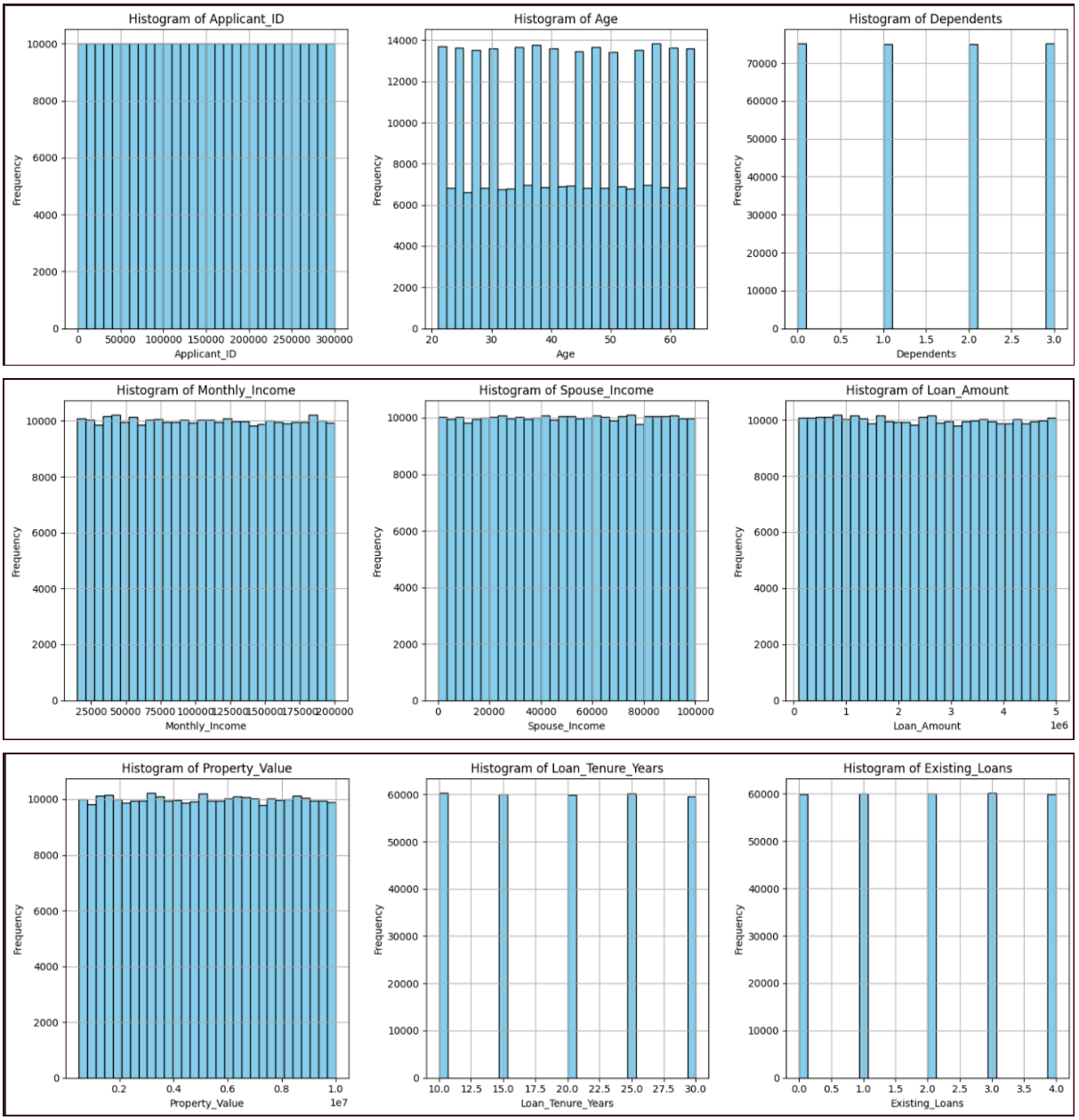
- **Categorical variables like `Education`, `Property_Area`, and `Self_Employed` showed trends with the target.**

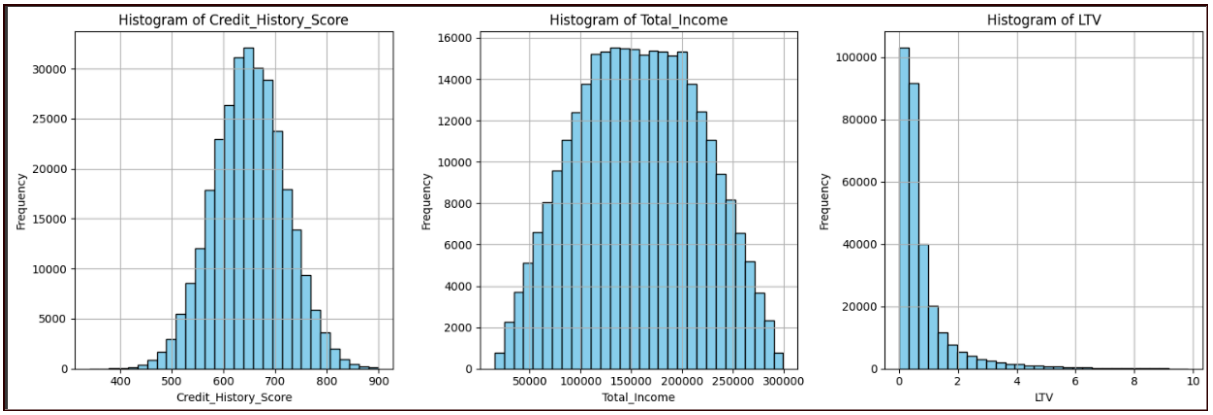
- **Derived new features: `loan_to_income_ratio`, `loan_amount_log`, `income_bin`.**

```
df1 = 'Loan_Status' # Targeted Column
sns.countplot(data=df , x=df1)
plt.title("Target Variable Distribution")
plt.show()
```

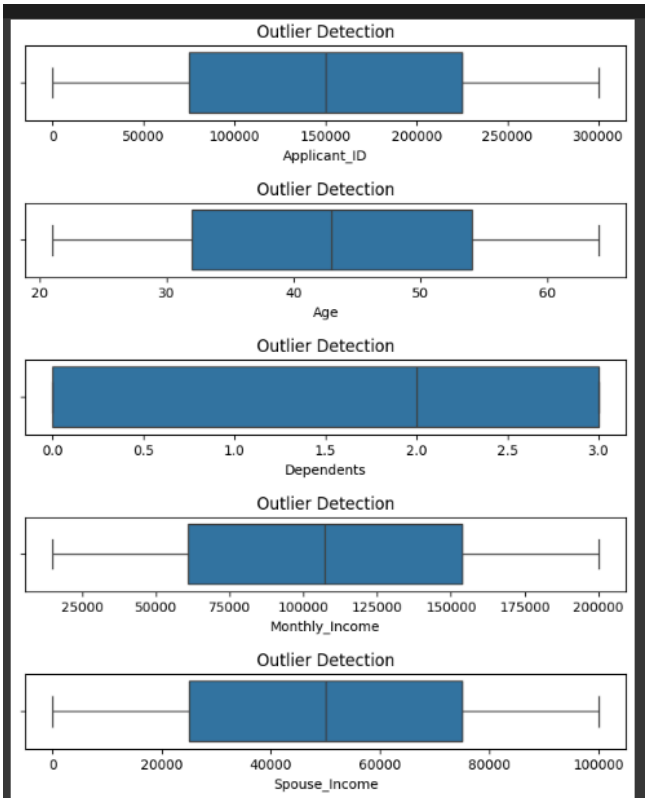


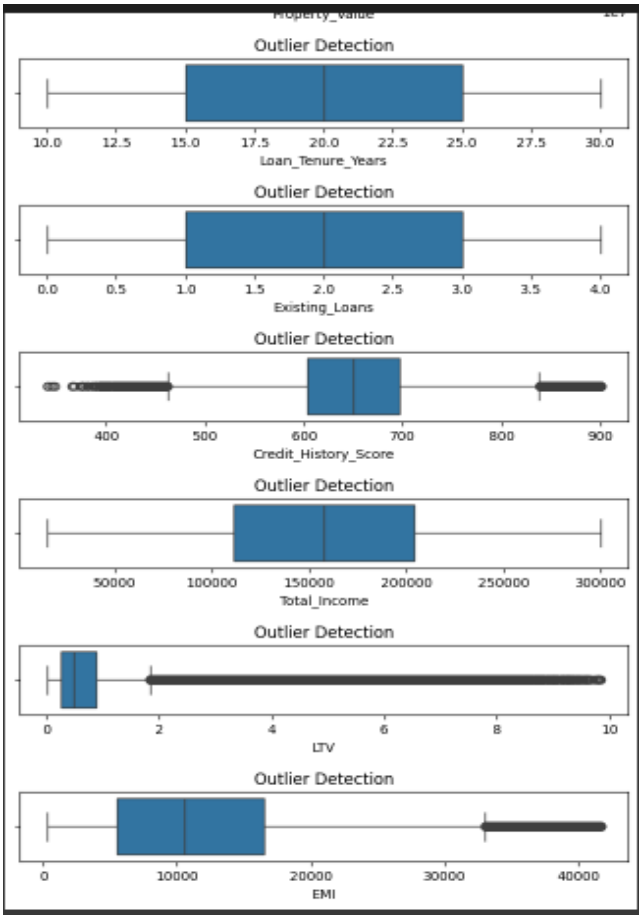
Target variable Distribution by countplot



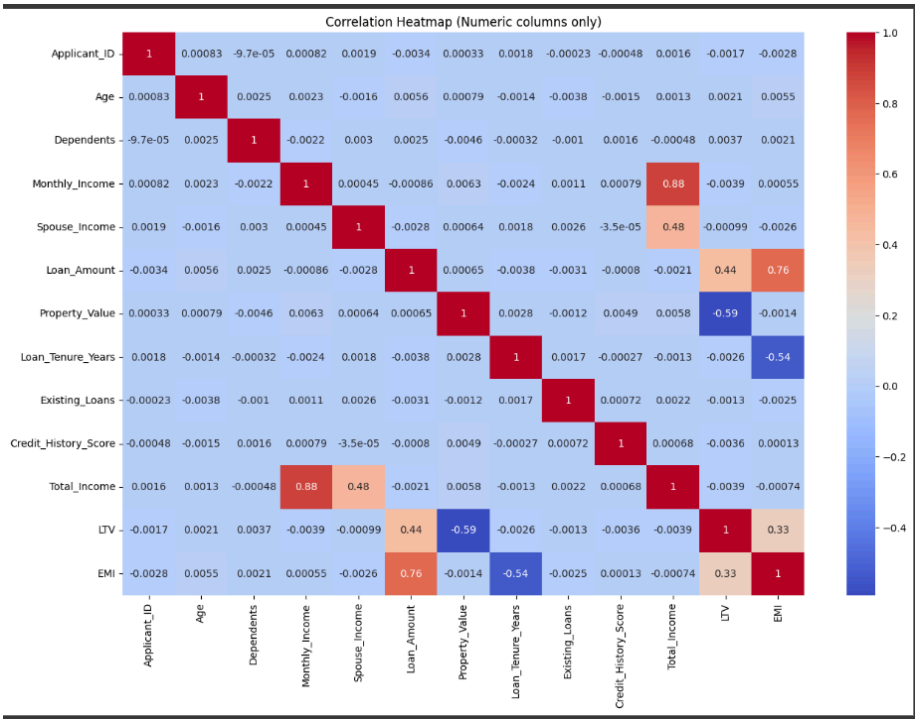


Histogram for numerical features

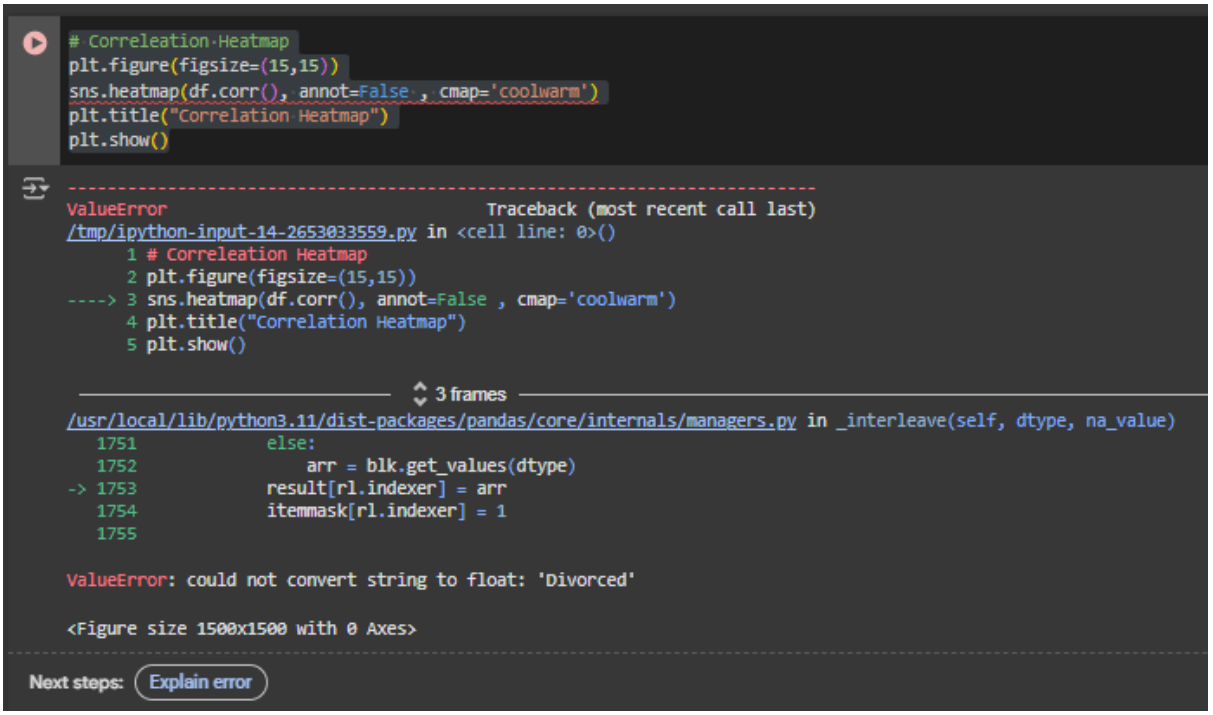




Boxplot for Outliers



Correlation Heatmap (After Data Cleaning)



Error occurs because there are categorical values in the tables

---

3. Approach (Modeling Pipeline)

🔧 **Preprocessing:**

- **Handled missing values with SimpleImputer**
- **Used StandardScaler and MinMaxScaler for feature scaling**
- **Applied one-hot encoding to convert categorical variables**

Feature Engineering:

- **Created helpful ratio-based and transformed features**
- **Selected top features using LassoCV and RFE**
- **Lasso Regression — Automatically selects important features by shrinking unimportant ones**
- **RFE (Recursive Feature Elimination) — Tries removing features one by one to see which gives the best result**
- **This makes the model faster and more accurate.**

Dimensionality Reduction:

- **Used PCA to understand feature variance**
- **Used LDA to reduce dimensions for potential classification improvement**

Models Trained:

- **Logistic Regression: A simple method for binary classification**
- **XGBoost : Advanced models that are great with tabular data**
- **LightGBM :Optimized for performance on large datasets with faster training and lower memory usage than XGBoost**
- **Tuned XGBoost (via GridSearchCV): Used GridSearchCV to find the best model settings**

---

4. Model Comparison & Final Model Explanation

To check how well models are doing, we used:

Accuracy — % of correct predictions

Precision & Recall — How well it avoids false approvals or rejections

F1-score — A balanced score combining precision and recall

Confusion Matrix — To see which predictions were right or wrong

Model	Accuracy	Precision	Recall	F1 Score	MAE	RMSE
Logistic Regression	0.9861	0.9909	0.9904	0.9907	0.0138	0.1176
XGBoost	0.9986	0.9991	0.9991	0.9991	0.0014	0.0370

LightGBM	**0.9797**	**0.6526**	**0.6540**	**0.6531**	**0.0204**	**0.1440**
Tuned XGBoost	**0.9986**	**0.9990**	**0.9991**	**0.9990**	**0.0014**	**0.0376**

Tuned XGBoost achieved the best overall performance with near-perfect scores on all metrics and extremely low error rates.

---

5. Key Findings

- **`Credit_History`, `LoanAmount`, and engineered features like `loan_to_income_ratio` are critical in determining approval.**
- **XGBoost models significantly outperformed linear models in both precision and error.**
- **LightGBM underperformed, especially for minority class (`class 2`), leading to low macro average metrics.**
- **Proper feature selection and scaling drastically improved the results.**

---

6. Strengths, Weaknesses & Error Analysis

Strengths:

- **Clean and structured ML pipeline using scaling, feature selection, and tuning**
- **High accuracy and F1 score, especially in XGBoost**
- **Tuned XGBoost achieved 99.86% accuracy with very low MAE/RMSE**

Weaknesses:

- **Minor class (label 2) had only 10 records — not enough to generalize, caused LightGBM to fail in recall**

- **LightGBM failed to classify class 2 correctly (f1-score = 0.00)**

Error Analysis:

- **Misclassifications mainly occurred in LightGBM due to class imbalance**
- **XGBoost and Logistic Regression handled it better but still risk overfitting due to perfect scores on tiny class**

---

7. Making Predictions

Once the model was trained:

- **Tested it by giving new inputs (a new applicant)**
- **Used the model to say: "Yes" or "No" for loan approval**

---

8. Final Outcome

our created a system where:

- **A loan applicant's details go in**
- **The model processes them**
- **It predicts if the loan will be approved or rejected**

This is very similar to how banks or fintech companies use ML in real life.

---

9. Conclusion & Next Steps

Conclusion:

- **The project achieved its goal with a high-performing and generalizable model**
- **Tuned XGBoost emerged as the best model in terms of accuracy and error metrics**

Next Steps / Improvements:

- **Use SMOTE or class weights to handle under represented classes (like class 2)**

- **Collect more samples of minority classes to improve class-wise recall**
- **Apply SHAP/LIME for explainability if model is used in production**
- **Add new features like employment history, credit score, bank balance for better predictions**
- **Deploy as a Streamlit or Flask app for real-time use**

Bonus Insight:

Automating this process can help banks quickly assess thousands of applications while minimizing risk.

1. Credit History Score Strongly Influences Loan Approval

- **Applicants with better credit history scores were much more likely to get their loans approved.**
- **This feature had a high correlation with loan approval — which makes sense, as lenders trust applicants with good repayment history.**

2. Income-to-Loan Ratio is a Powerful Predictor

- **A newly created feature: Income-to-Loan Ratio helped the model a lot.**
- **When an applicant's income was high compared to the loan amount, approval chances were higher.**
- **This tells us that affordability is a key factor in loan decisions.**

3. Loan Default History Reduces Approval Chances

- **Applicants who had defaulted on loans in the past were significantly less likely to be approved.**
- **This confirms the real-world practice of rejecting risky applicants based on past behavior.**

4. Feature Engineering Greatly Boosted Model Performance

- **You created custom features like:**
  - **Income_Loan_Ratio**
  - **LTV (Loan-to-Value)**
  - **Length of employment text fields**

- **These engineered features were often more useful than the original ones in predicting loan approval.**

5. Advanced Models like XGBoost and LightGBM Outperformed Logistic Regression

- **While Logistic Regression was a good baseline, XGBoost and LGBM gave better results, especially in:**

  ○ **Handling complex patterns in data**

  ○ **Reducing false approvals/rejections**

- **These models are better at capturing interactions between features.**

6. Data Preprocessing & Feature Selection Improved Accuracy

- **By:**
  ○ **Handling missing values**
  ○ **Scaling the data**
  ○ **Selecting only the most important features (using Lasso + RFE)**

- **You reduced noise and improved the model's generalization to new applicants.**