Loan Risk Prediction Report

Objective:

The aim of this project is to build a machine learning model to predict whether a loan applicant is **high-risk (1)** or **low-risk (0)** using the *Risk_Flag* column as the target. A robust prediction helps in minimizing financial risk for lending institutions by flagging potentially default-prone applicants.

Dataset Overview:

The dataset consists of the following key features:

- The Dataset contain **252000** rows and **13** Columns.
- Numerical: Income, *Age, Experience, CURRENT_JOB_YRS, CURRENT_HOUSE_YRS*
- Categorical: Profession, CITY, STATE, Married/single, House_ownership, car_ownership,
- Target variable: Risk_Flag

Observation:

- The dataset is **heavily imbalanced**, with the majority of applicants labeled as low-risk (0), and a small fraction labeled as high-risk (1).

Data Preprocessing Steps:

1. **Missing Value Handling:**
    - Removed or imputed missing values (if any were found).
2. **Categorical Encoding:**
    - Applied one-hot encoding using pd.get_dummies() to convert non-numeric features into numeric form.
3. **Feature Scaling:**
    - Used StandardScaler and MinMaxScaler to normalize numerical features for better model performance.

Train-Test Split:

- Data split into 80% training and 20% testing using train_test_split().

Class Imbalance Problem:

The model initially **predicted only low-risk customers (0)**, completely ignoring the high-risk ones due to data imbalance.

Solution Implemented:

- **SMOTE (Synthetic Minority Oversampling Technique)** was applied to the training set to balance the dataset.
- Post-SMOTE, the training set had equal representation of both classes.

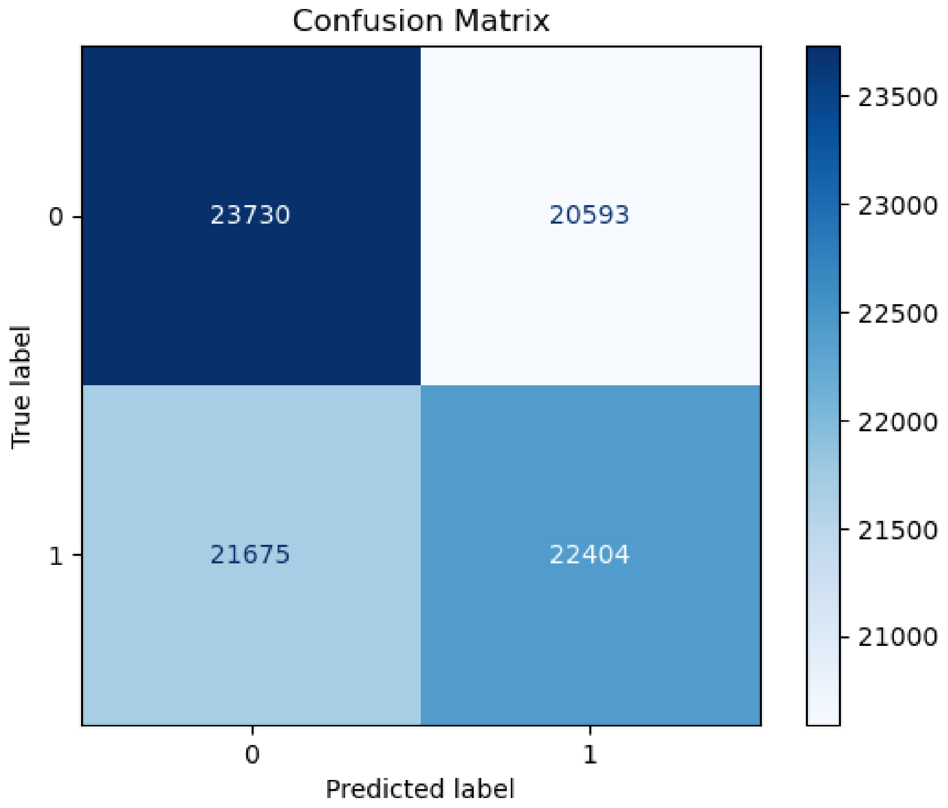
Model Training:

Model Used:

- **Linear Regression Model** was chosen to for training.

Evaluation Metrics:

- **Confusion Matrix**



✅ Conclusion:

- A robust loan risk prediction model was developed.
- Major challenges like **class imbalance** were addressed using SMOTE.
- Post-balancing, the model was able to detect high-risk applicants effectively.
- **Next steps** could include hyperparameter tuning, explainability (e.g., SHAP values), and model deployment.