Credit Score Simulator – Predicting Serious Delinquency Risk

Ana Harringthon

Data Science Certificate
May 2025

Agenda

- Problem Statement
- Hypothesis
- Data and Tools Used
- Data Processing
- Exploratory Data Analysis (EDA)
- Feature Engineering
- Modeling
    - Hyperparameter Tuning
    - Threshold Optimization
    - Final Model & Evaluation
- Deployment & App Integration
- Summary & Key Learnings
- Q&A

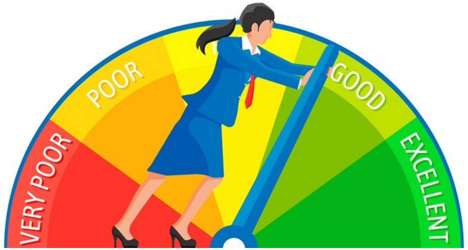
The Problem Statement

Credit scores are crucial in financial decisions, yet:

- Most people don't fully understand what affects their score.
- Few tools allow users to experiment with different input scenarios.
- Existing models often lack transparency and interactivity.

This project offers a **simple credit score simulator** that uses a public dataset to predict **serious delinquency risk** and map it to a credit score category.

Users can explore how selected inputs influence their credit classification.



Hypothesis

Certain financial patterns increase the likelihood of serious delinquency, especially:

- High credit utilization
- Low income
- Past delinquencies

The model predicts the SeriousDlqin2yrs outcome using:

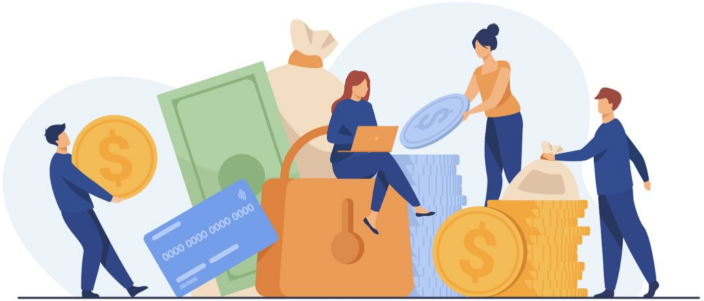
- Age
- Monthly income
- Delinquency history
- Credit utilization
- Debt ratio

➡️ This prediction is then mapped to a credit score category in the simulator.

Data and Tools Used

This project uses the **"Give Me Some Credit"** dataset from Kaggle, which includes anonymized data from credit applicants.

- 150,000+ entries
- Target variable: **SeriousDlqin2yrs**
- 11 features, including:
  - Monthly income
  - Age
  - Delinquency history
  - Credit utilization
  - Debt ratio
- Personal and sensitive details are excluded
- Limitations: No timestamps or detailed credit history



The dataset was used to train a model that predicts delinquency risk and supports the credit score simulation.

Data and Tools Used

Overview of the features included in the dataset:

Feature	Description
RevolvingUtilization	Ratio of total credit used to total available credit (max 1.0)
age	Age of the customer (in years)
NumberOfTime30-59DaysLate	Number of times the customer has been 30–59 days late on a payment
DebtRatio	Ratio of monthly debt payments to monthly income
MonthlyIncome	Reported monthly income
NumberOfOpenCreditLines	Number of open credit lines (e.g., credit cards, loans)
NumberOfTimes90DaysLate	Number of times the customer has been 90+ days late
NumberRealEstateLoans	Number of real estate loans or lines of credit
NumberOfTimes60-89DaysLate	Number of times the customer has been 60–89 days late
NumberOfDependents	Number of dependents claimed by the customer

Data and Tools Used

This project used a range of tools for data processing, modeling, and app development:

- Python – Core language for all development
  Pandas / NumPy – Data cleaning and manipulation
- Scikit-learn – Machine learning model training and evaluation
- Matplotlib / Seaborn – Data visualization
- Jupyter Notebook – Exploratory analysis and testing
- Streamlit – Building the interactive credit score simulator

These tools enabled the creation of a functional, end-to-end solution.

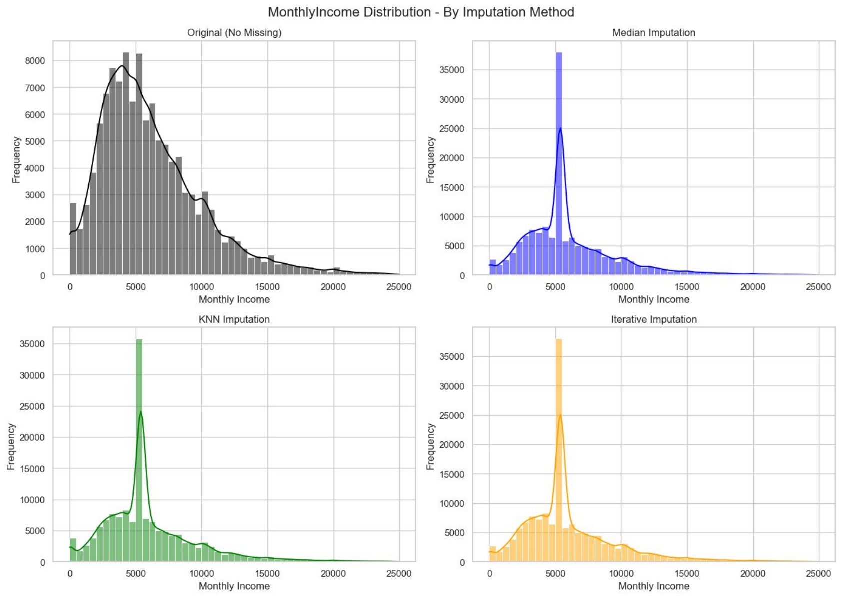
Data Processing

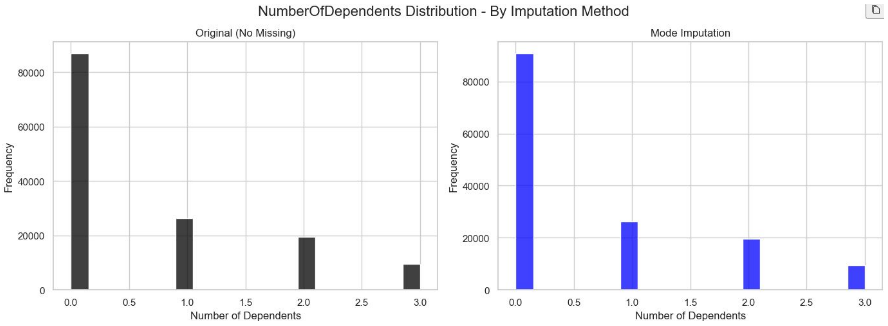
Data Types:

- All features were numerical, so no type conversion was required.

Missing Values:

- MonthlyIncome (~20%) → imputed with median.
- NumberOfDependents (~2.6%) → imputed with mode.
- Advanced methods like KNN and MICE were tested, but did not show meaningful improvement.
- Median and mode were selected for their simplicity and stability.





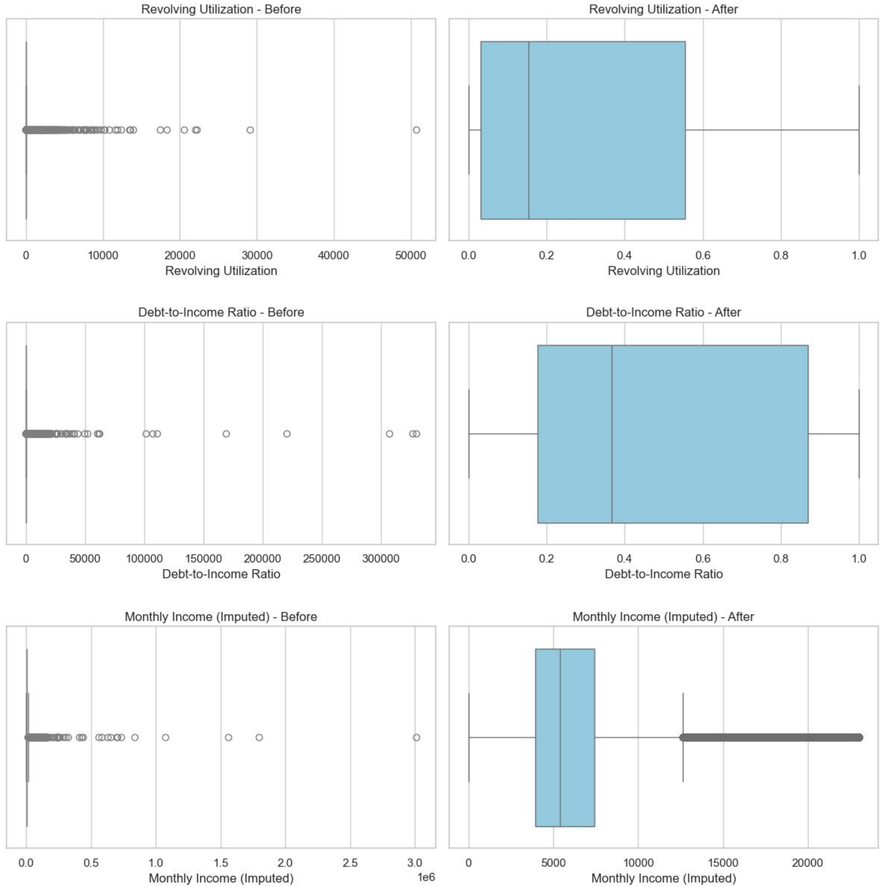
Data Processing

Outliers:

Extreme values were identified in several numeric features. To improve model stability and data quality, the following adjustments were made:

- Credit utilization and debt ratio capped at 100%
- Monthly income capped at the 99th percentile
- Delinquency variables with placeholder values (96, 98) were removed. (rare and non-representative records)
- Age capped between 18 and 80

Some count variables (e.g., credit lines, real estate loans) were reviewed but kept unchanged to preserve potential predictive value.



Exploratory Data Analysis

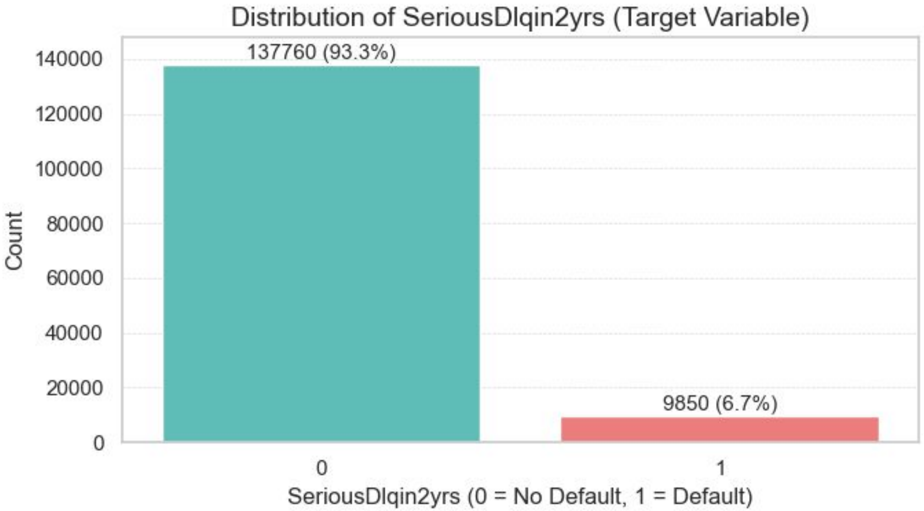
Understanding the Target

The target variable SeriousDlqin2yrs indicates whether a customer experienced serious delinquency (90+ days past due) within two years.

- 93.3% of customers did not default (class 0)
- 6.7% of customers defaulted (class 1)

This reveals a strong class imbalance, where defaults represent a small minority.

Techniques like resampling or using metrics like F1-score can help address this during model training.



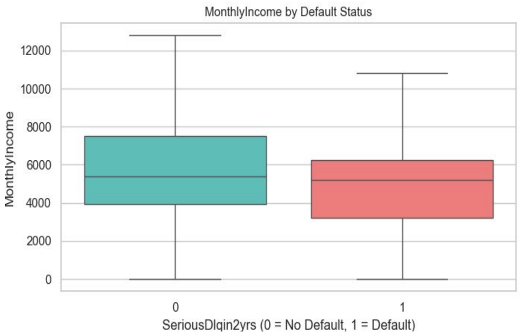
Exploratory Data Analysis

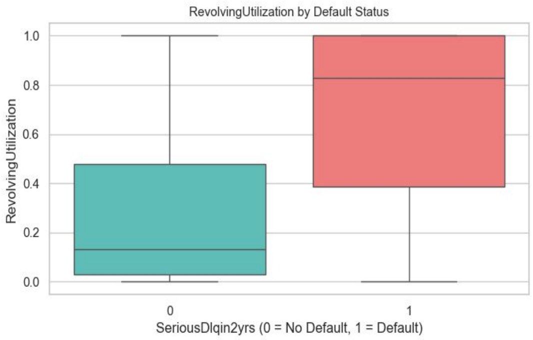
Default Risk Factors

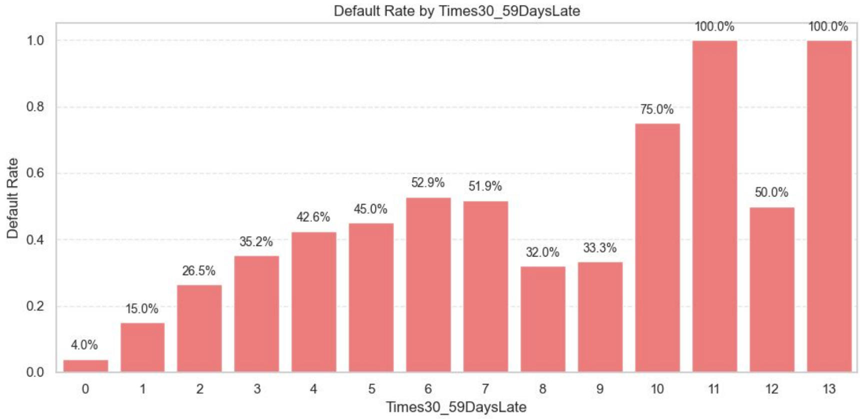
Exploration revealed clear patterns linked to higher default risk:

- Slightly **lower income levels** observed among defaulters**.**
- **High credit utilization** (close to 1.0)
- **Any history of late payments**
- **Younger individuals**

Even a single past due record strongly increases the likelihood of serious delinquency.





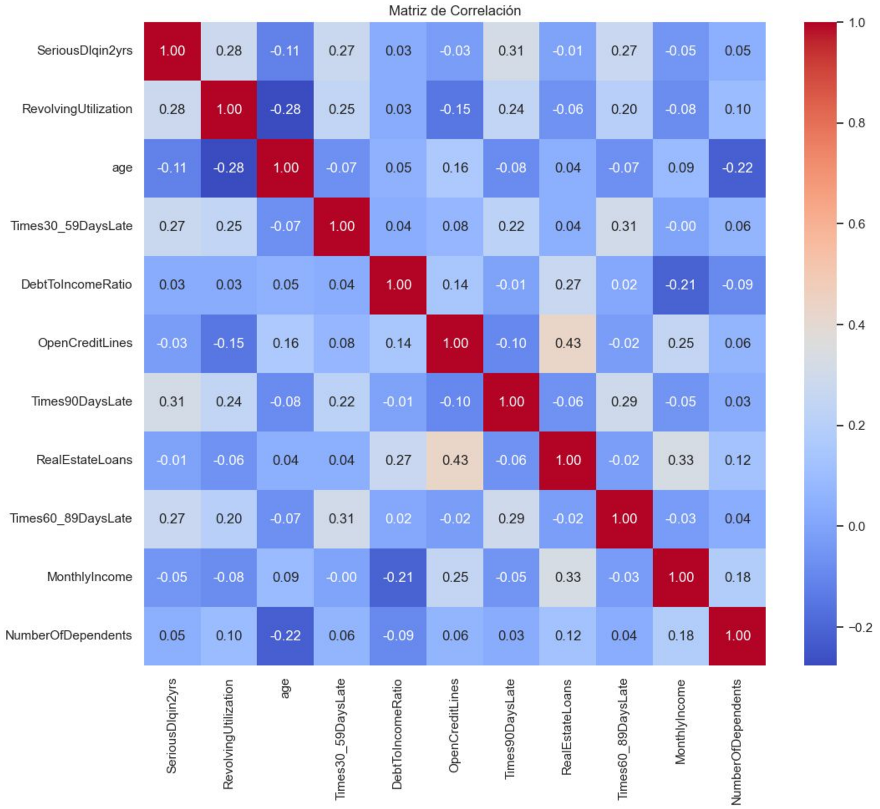


Exploratory Data Analysis

A correlation matrix was used to explore linear relationships between numerical variables.

- SeriousDlqin2yrs showed low correlations with most features
- The strongest relationships were with:
  - Times90DaysLate (0.31)
  - Times30_59DaysLate (0.27)
  - Times60_89DaysLate (0.27)
  - RevolvingUtilization (0.28)

These findings confirmed the relevance of delinquency and utilization features for modeling.



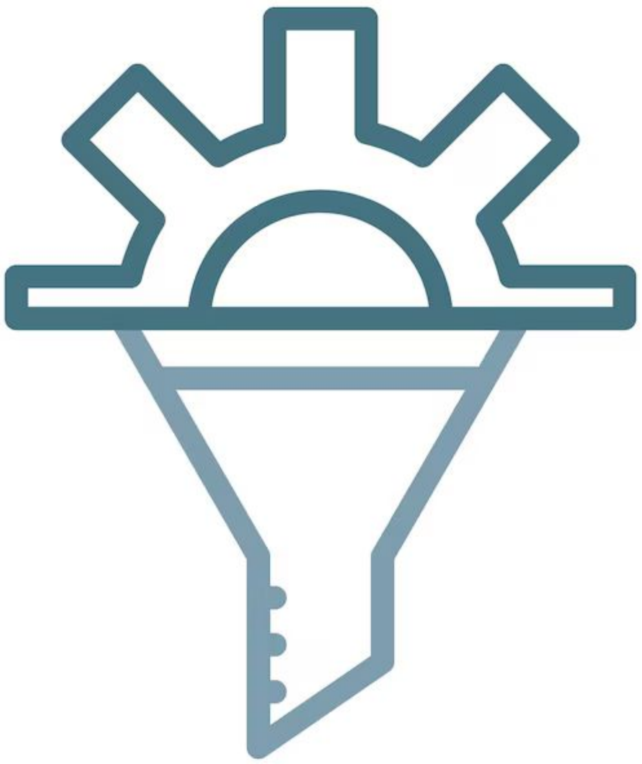
Feature Engineering

To better capture financial behavior and risk dynamics, new features were created based on domain knowledge and variable interactions:

Engineered Features:

- **TotalPastDue:** Combined late payment events (30–59, 60–89, 90+ days)
- **FinancialStressScore:** DebtRatio × RevolvingUtilization
- **CreditBurdenPerLine:** Income / (Open Credit Lines + 1)
- **AgeUtilizationRatio:** Age / RevolvingUtilization
- **IncomeAgeRatio**: Income / (Age + 1)
- **LinesPerYear:** Open Credit Lines / Age

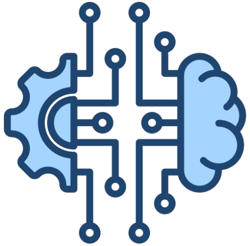
📌 These features helped represent more complex financial behaviors and improved the dataset's predictive potential.



Modeling

Models Evaluated

- Logistic Regression
- Random Forest
- XGBoost
- LightGBM

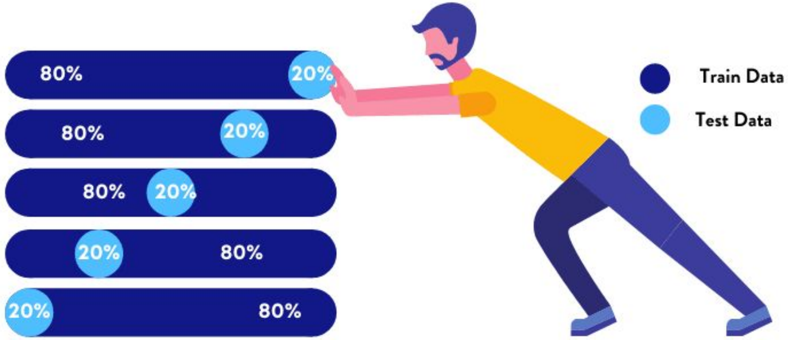


Validation Strategy

- 80/20 **train/test spli**t with **stratification**
- **10-fold stratified cross-validation**
- In each fold:
  - Features were scaled
  - SMOTE used to address class imbalance

Evaluation Metrics

- AUC
- F1-score
- Precision
- Recall



Modeling

Model Performance Comparison

Model	AUC (± std)	F1-score (± std)	Precision	Recall
Logistic Regression	0.854 ± 0.006	0.331 ± 0.005	0.213	**0.749**
Random Forest	0.831 ± 0.007	0.344 ± 0.020	0.417	0.293
XGBoost	0.845 ± 0.008	0.327 ± 0.021	0.462	0.253
LightGBM	**0.856 ± 0.008**	**0.366 ± 0.018**	**0.481**	0.296

📌 LightGBM showed the best overall performance, with the highest AUC, F1-score, and Precision. Logistic Regression had the highest Recall, but lower balance overall.

Modeling

Hyperparameter Tuning

To improve the top-performing model, LightGBM, hyperparameter tuning was performed using Optuna.

- **Optimization method:** Optuna — 100 trials
- **Objective:** Maximize AUC (10-fold stratified CV)
- Parameters explored:
  - n_estimators
  - learning_rate
  - max_depth
  - num_leaves
  - Other related parameters

- **Within each fold:**
  - Feature scaling
  - SMOTE applied for class balance
  - Data leakage prevention maintained

OPTUNA

```
LGBMClassifier

LGBMClassifier(colsample_bytree=0.6126463569356596,
               learning_rate=0.04095983868272451, max_depth=10,
               min_child_samples=18, n_estimators=73, num_leaves=29,
               random_state=42, subsample=0.8584287480250375)
```

Modeling

Threshold Optimization

To go beyond the default 0.5 threshold, a custom threshold was optimized to maximize recall, ensuring high-risk clients are not missed.

Strategy

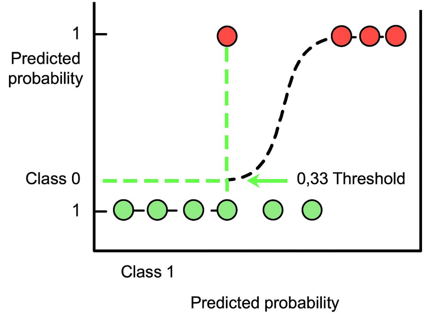
- A custom threshold was selected per fold during validation
- Thresholds tested: **0.01 to 0.99**
- For each fold:
  - Enforced **recall ≥ 0.7**
  - Chose threshold with highest precision among those

Validation Setup

- 10-fold Stratified Cross-Validation
- Tuned LightGBM model
- Probabilities predicted in each fold
- Final threshold = median of selected thresholds across folds

Final Threshold

- The final decision threshold was set to **0.33**
- Balances the need to **detect high-risk clients** while avoiding excessive false positives.

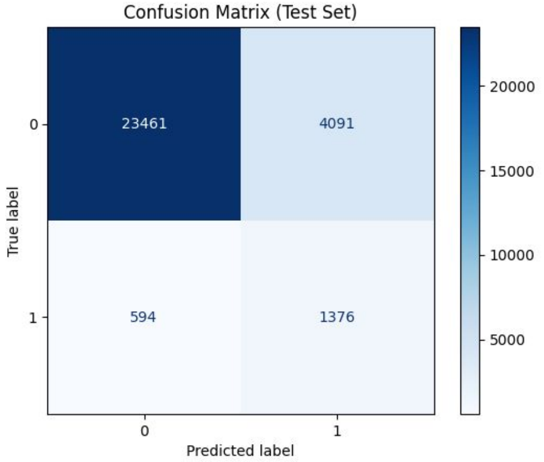


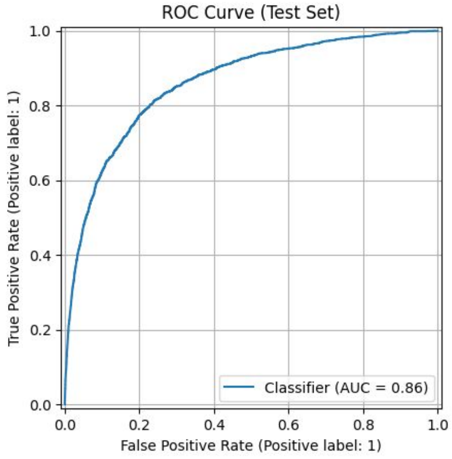
Modeling

Final Model Evaluation

The final **LightGBM** model was trained with optimized hyperparameters and evaluated on the unseen test set using the threshold **0.33**.

Metric	Value
AUC	0.8614
Accuracy	0.8413
Precision	0.2517
Recall	0.6985
F1 Score	0.3700



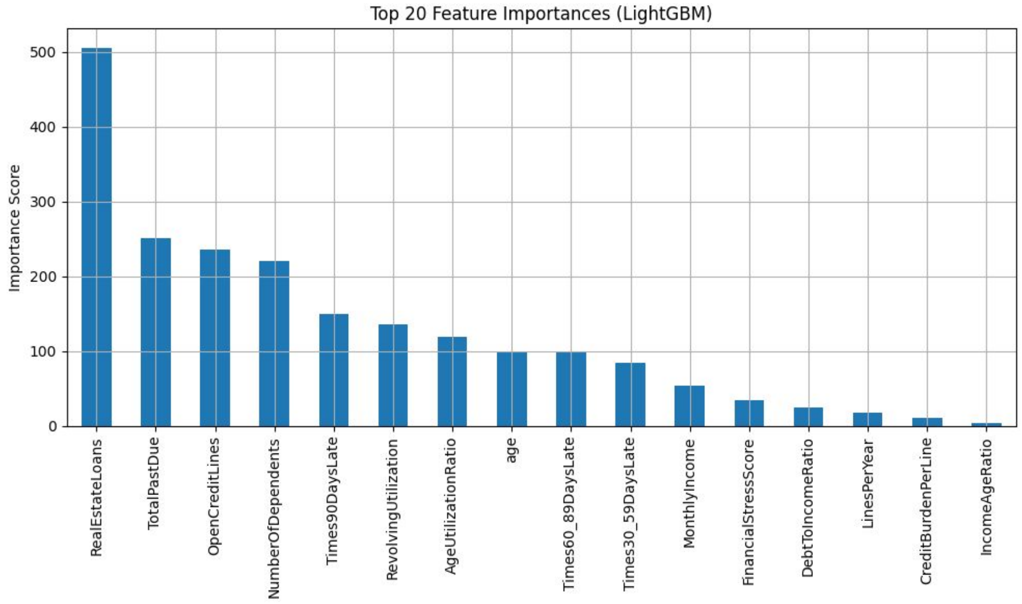


- **Confusion Matrix**
  1376 true positives (class 1) identified
  High recall shows success in identifying high-risk clients

- **ROC Curve**
  AUC = 0.86 → Strong ability to distinguish between default and non-default cases

Modeling

The top 20 features were ranked by their contribution to the final LightGBM model.



Key observations:

- RealEstateLoans, TotalPastDue, and OpenCreditLines were the most influential features
- Both original and engineered features contributed meaningfully (e.g., TotalPastDue, AgeUtilizationRatio)
- Delinquency-related variables (Times90DaysLate, etc.) remained relevant, confirming initial analysis

📌 Feature importance helps interpret how the model makes decisions and supports transparency for stakeholders.

Deployment & App Integration

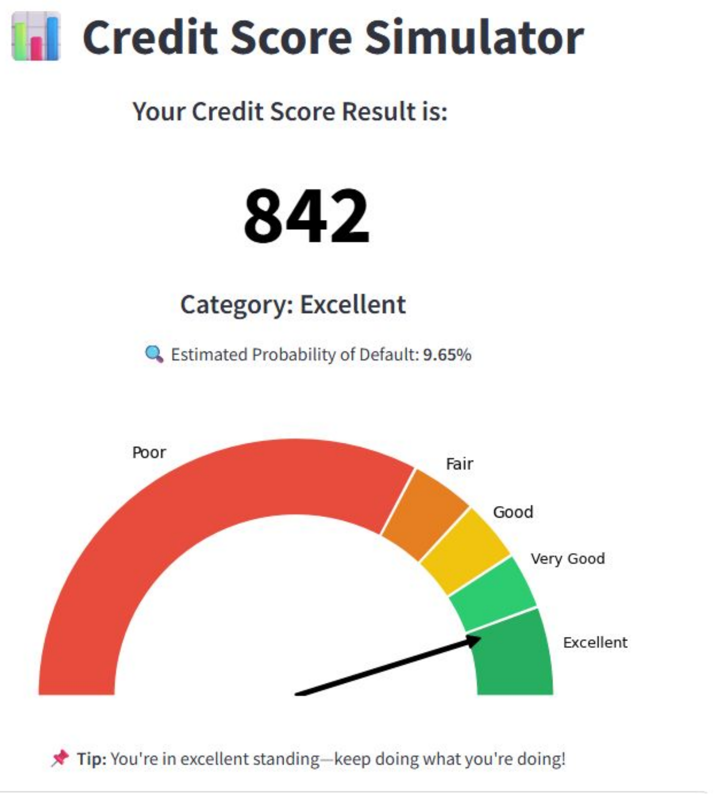
The trained **LightGBM** model was integrated into a **Streamlit-based web application** to simulate credit scores and provide decision support.

Deployment Pipeline

- Model saved with joblib
- Threshold (0.33) applied to prediction logic
- Input form allows users to test different scenarios
- Outputs: **score, risk category**, and a **general tip** based on the result

Application Features

- Interactive profile simulation
- Real-time risk prediction
- Credit score category mapping



Summary Conclusions

- A thorough EDA and preprocessing pipeline addressed class imbalance, outliers, and data quality issues, establishing a strong foundation for modeling.
- Custom feature engineering introduced variables that captured complex financial behavior, enhancing both model performance and interpretability.
- The final LightGBM model, tuned with Optuna and SMOTE, achieved strong performance (AUC = 0.86, Recall = 70.7%) in identifying customers at risk of serious delinquency.
- The decision to optimize for recall aligned with the business goal of minimizing missed defaulters, while accepting some false positives as a trade-off.
- Model outputs were transformed into a credit score (300–900) and grouped into risk categories (e.g., Poor, Excellent), offering a more flexible and intuitive way to communicate risk. This transformation is independent of the business-driven threshold used for binary classification.
- A fully functional Streamlit app was developed to simulate risk in real time, providing users with a score, category, and general recommendations for improvement.

Key Learnings

- Modeling decisions must align with business objectives.
- Feature engineering plays a critical role in model success.
- Comprehensive preprocessing is essential.
- Translating outputs into interpretable formats enhances communication.
- Interactivity increases impact.
- Machine learning development is iterative.

Thank You

Appendix

References

Dataset: *Give Me Some Credit* – Kaggle
https://www.kaggle.com/datasets/c/GiveMeSomeCredit

Documentation:

- Streamlit → https://docs.streamlit.io
- Scikit-learn → https://scikit-learn.org/stable/documentation.html
- Seaborn → https://seaborn.pydata.org
- Matplotlib → https://matplotlib.org

Additional resources:
Stack Overflow threads and official library documentation were consulted for troubleshooting and implementation examples.