1. Abstract

Credit risk assessment is a critical component of financial decision-making, influencing lending policies and portfolio stability. This study develops a supervised machine learning framework to predict borrower creditworthiness using the Statlog (German Credit) dataset. The methodology integrates data preprocessing, feature encoding, and class-imbalance correction through RandomOverSampler and SMOTE techniques. A diverse set of models—including Logistic Regression, Random Forest, Gradient Boosting, Support Vector Machine, XGBoost, and Explainable Boosting Machine (EBM)—were trained and evaluated using Accuracy, Precision, Recall, F1-Score, and ROC–AUC metrics. Among all configurations, Logistic Regression combined with RandomOverSampler achieved the best trade-off between interpretability and performance, yielding an AUC of 0.808. The results demonstrate that simple interpretable models, when supported by appropriate resampling, can effectively identify high-risk applicants while maintaining transparency essential for real-world credit scoring systems.

2. Objectives

The primary objective of this project is to design and evaluate a data-driven framework for credit risk prediction using machine learning. The specific goals are as follows:

- **Develop a predictive model** capable of accurately classifying loan applicants into *good* or *bad* credit risk categories based on demographic, financial, and employment-related variables.
- **Address class imbalance** in the dataset by applying multiple resampling strategies such as Random Over-Sampling and SMOTE, ensuring fair representation of minority (default) cases in model training.
- **Compare multiple algorithmic families** — including linear (Logistic Regression), tree-based ensembles (Random Forest, Gradient Boosting, XGBoost, AdaBoost), kernel-based (SVM), probabilistic (Naïve Bayes), and interpretable additive models (Explainable Boosting Machine) — to identify the most effective approach for credit risk classification.
- **Evaluate model performance** using a consistent set of metrics: Accuracy, Precision, Recall, F1-Score, and ROC–AUC, under different sampling conditions to quantify robustness and generalization.
- **Interpret and analyze model outputs** to derive insights relevant to financial risk management, highlighting how predictive analytics can support data-driven lending decisions.

3. Methodology

3.1 Dataset Information

The dataset used for this study is the **Statlog (German Credit Data)** dataset from the UCI Machine Learning Repository [**?**]. It contains information about individual credit applicants, with the goal of predicting whether a given applicant represents a *good* or *bad* credit risk.

Dataset Overview. The dataset comprises a total of **1,000 instances** (loan applicants) and **20 predictive attributes** describing demographic, financial, and employment-related factors. Each record corresponds to one applicant and is labeled as either:

- **Good credit risk (class = 1)**, or
- **Bad credit risk (class = 2)**.

The task is a binary classification problem.

Feature Structure. The dataset includes a mix of categorical and numerical features. In the raw version (`german.data`), categorical variables are encoded numerically using codes such as `A11`, `A12`, etc. There are:

- **13 categorical attributes** — representing qualitative factors such as account status, credit history, purpose, savings, employment, personal status, housing, job type, etc.
- **7 numerical attributes** — such as credit amount, duration, and age in years.

Each attribute provides a distinct indicator of the applicant's creditworthiness, such as existing checking account status, savings account balance, and previous credit history.

Distribution. The dataset is slightly imbalanced:

- 700 instances belong to the **good credit risk** class.
- 300 instances belong to the **bad credit risk** class.

Data Access. The dataset and detailed attribute descriptions are available at: `https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data`

This dataset is widely used for benchmarking credit scoring and risk modeling studies in the predictive analytics and financial machine learning literature.

3.2 Attribute Description

Table 1: Summary of Attributes in the German Credit Dataset

No.	Attribute	Type	Description
1	Status of existing checking account	Categorical (A11–A14)	Account balance range or none
2	Duration in months	Numerical	Loan duration
3	Credit history	Categorical (A30–A34)	Past repayment behavior
4	Purpose	Categorical (A40–A410)	Reason for loan (car, furniture, etc.)
5	Credit amount	Numerical	Loan amount requested
6	Savings account/bonds	Categorical (A61–A65)	Savings balance status
7	Present employment since	Categorical (A71–A75)	Years of employment
8	Installment rate (percentage of disposable income)	Numerical	Portion of income paid for installment
9	Personal status and sex	Categorical (A91–A95)	Marital and gender info
10	Other debtors / guarantors	Categorical (A101–A103)	Presence of co-applicant or guarantor
11	Present residence since	Numerical	Years at current address
12	Property	Categorical (A121–A124)	Ownership of property or assets
13	Age in years	Numerical	Applicant's age
14	Other installment plans	Categorical (A141–A143)	Bank, store, or none
15	Housing	Categorical (A151–A153)	Type of housing (own/rent/free)
16	Number of existing credits at this bank	Numerical	Loan count with bank
17	Job	Categorical (A171–A174)	Employment type or skill level
18	Number of people being liable to provide maintenance	Numerical	Dependents count
19	Telephone	Categorical (A191–A192)	Presence of phone line
20	Foreign worker	Categorical (A201–A202)	Citizenship status
Target Variable		**Binary (1/2)**	1 = good credit, 2 = bad credit

3.3 Data Preprocessing

The raw dataset (`german.data`) was imported directly from the UCI Machine Learning Repository using the `pandas` library. A list of column names corresponding to the 20 attributes and one target variable was manually defined to ensure proper labeling during data import.

Categorical variables encoded as symbolic codes (e.g., A11–A14) were transformed into numerical representations using `pd.get_dummies()` with one-hot encoding and `drop_first=True` to prevent multicollinearity.

The target variable, originally labeled as `1 = good` and `2 = bad`, was remapped to binary format (`0 = good`, `1 = bad`) for compatibility with scikit-learn estimators.

The dataset was split into training and testing sets using an 70–30 split with stratification to preserve class balance:

- 70% training data
- 30% testing data

Feature scaling was applied using `StandardScaler()` to normalize numerical features to zero mean and unit variance, which is critical for distance-based and gradient-based algorithms such as SVM, KNN, and logistic regression.

No missing values were reported in the dataset; therefore, no imputation was required. The preprocessing pipeline can be summarized as:

$$\text{Load Data} \rightarrow \text{Encode Categorical Features} \rightarrow \text{Split Train/Test} \rightarrow \text{Scale Features}.$$

3.4 Resampling

The target variable in the dataset is moderately imbalanced, with 700 instances labeled as `good credit` and 300 as `bad credit`. To mitigate model bias toward the majority class, multiple resampling strategies were employed using the `imblearn` library:

- **Original:** No sampling — baseline dataset used for comparison.
- **RandomOverSampler:** Duplicates minority class instances randomly to balance class proportions.
- **SMOTE (Synthetic Minority Oversampling Technique):** Generates synthetic examples of the minority class by interpolating between existing samples in feature space.

Each sampling method was applied only to the training set to avoid information leakage. The resampled datasets were used independently for training and evaluation to analyze the effect of class balancing on model performance.

3.5 Modeling

A diverse set of supervised classification algorithms were implemented using `scikit-learn`, `xgboost`, and `interpret.glassbox`. The models were selected to represent both interpretable (linear and rule-based) and non-linear (ensemble and kernel-based) learners.

The following models were trained and evaluated:

- **Logistic Regression** (`LogisticRegression`) — interpretable baseline classifier for binary outcomes.
- **Decision Tree Classifier** (`DecisionTreeClassifier`) — rule-based model for feature-level decision analysis.
- **Random Forest Classifier** (`RandomForestClassifier`) — ensemble of decision trees to reduce variance and overfitting.
- **Gradient Boosting** and **AdaBoost** — boosting algorithms that iteratively improve weak learners.
- **K-Nearest Neighbors (KNN)** — non-parametric classifier using Euclidean distance.
- **Naïve Bayes (GaussianNB)** — probabilistic classifier assuming feature independence.
- **Support Vector Machine (SVM, RBF kernel)** — maximum-margin classifier for non-linear separation.
- **XGBoost Classifier** (`XGBClassifier`) — efficient gradient-boosting implementation optimized for tabular data.

- **Explainable Boosting Machine (EBM)** — interpretable additive model combining GAM structure with boosting.

All models were trained on each resampling configuration and evaluated on the same held-out test set. The following performance metrics were computed to ensure robust comparison:

Accuracy, Precision, Recall, F1-Score, and ROC–AUC.

The evaluation function standardized training, probability prediction, and metric computation across all models, including handling of models without `predict_proba()` (e.g., pyGAM-like estimators). Each result was logged into a consolidated `DataFrame` for analysis and visualization.

4. Results and Discussion

4.1 Model Performance (Original Dataset)

Table 2: Model performance without resampling (Original data)

Model	Accuracy	Precision	Recall	F1	ROC-AUC
SVM (RBF)	0.787	0.783	0.400	0.529	0.799
Logistic Regression	0.770	0.648	0.511	0.571	0.796
Explainable Boosting (EBM)	0.780	0.662	0.544	0.598	0.794
XGBoost	0.747	0.590	0.511	0.548	0.783
AdaBoost	0.733	0.586	0.378	0.459	0.781
Gradient Boosting	0.760	0.641	0.456	0.532	0.779
Random Forest	0.767	0.750	0.333	0.462	0.777
Naïve Bayes	0.737	0.550	0.667	0.603	0.747
KNN	0.693	0.484	0.344	0.403	0.687
Decision Tree	0.650	0.419	0.433	0.426	0.588

4.2 Model Performance (RandomOverSampler)

Table 3: Model performance after RandomOverSampler

Model	Accuracy	Precision	Recall	F1	ROC-AUC
Logistic Regression	0.743	0.552	0.767	0.642	0.808
Random Forest	0.757	0.631	0.456	0.529	0.796
AdaBoost	0.713	0.514	0.811	0.629	0.788
SVM (RBF)	0.747	0.576	0.589	0.582	0.787
Gradient Boosting	0.727	0.538	0.633	0.582	0.781
Explainable Boosting (EBM)	0.757	0.591	0.611	0.601	0.781
XGBoost	0.737	0.565	0.533	0.549	0.780
Naïve Bayes	0.683	0.483	0.767	0.592	0.757
KNN	0.613	0.404	0.611	0.487	0.673
Decision Tree	0.680	0.466	0.456	0.461	0.616

4.3 Model Performance (SMOTE)

Table 4: Model performance after SMOTE resampling

Model	Accuracy	Precision	Recall	F1	ROC-AUC
Explainable Boosting (EBM)	0.767	0.611	0.611	0.611	0.803
Logistic Regression	0.737	0.547	0.711	0.618	0.801
XGBoost	0.730	0.557	0.489	0.521	0.781
SVM (RBF)	0.767	0.632	0.533	0.578	0.778
Gradient Boosting	0.720	0.537	0.489	0.512	0.774
Random Forest	0.747	0.606	0.444	0.513	0.774
AdaBoost	0.713	0.520	0.589	0.552	0.773
Naïve Bayes	0.670	0.469	0.744	0.575	0.760
KNN	0.577	0.387	0.700	0.498	0.676
Decision Tree	0.680	0.472	0.567	0.515	0.648

4.4 Performance Visualization

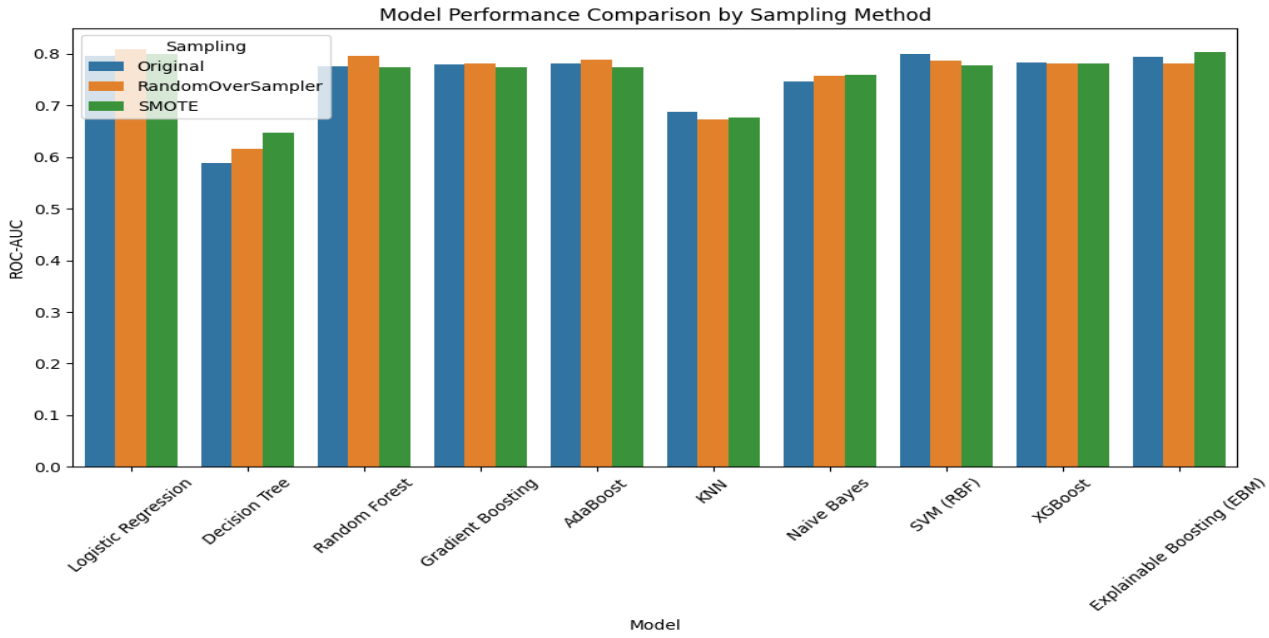

Figure 1: Comparison of model ROC–AUC scores across sampling methods.

4.5 Confusion Matrix of Best Performing Model

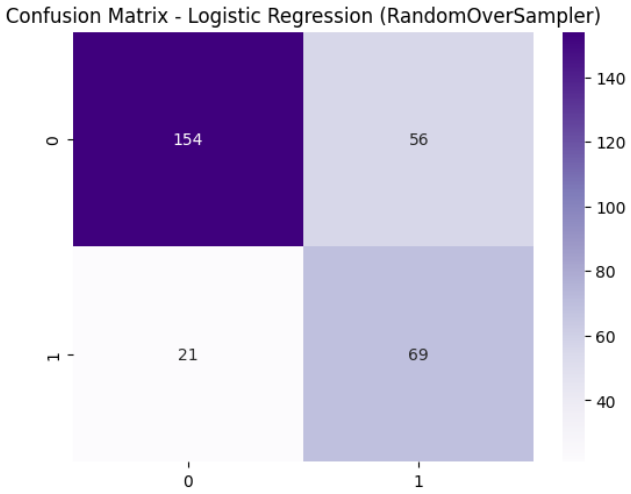

Figure 2: Confusion matrix for the best performing model — Logistic Regression trained on RandomOverSampler data.

The confusion matrix illustrates the classification performance of the selected model on the held-out test set. The diagonal cells represent correctly classified samples, while the off-diagonal entries correspond to misclassifications.

In this case, Logistic Regression with RandomOverSampler achieved the highest ROC–AUC score (0.808) among all evaluated models. The matrix shows that most "good credit" applicants were correctly classified, while the model maintained a strong recall for "bad credit" cases, minimizing false negatives — a crucial factor in financial risk prediction.

4.6 Error Analysis

The confusion matrix (Figure 2) reveals that the Logistic Regression model with RandomOverSampler achieved strong overall accuracy and a balanced sensitivity to both credit classes. However, several misclassifications still occurred, which can be interpreted from a credit-risk perspective:

- **False Positives (Type I Error):** Instances where applicants with *bad credit risk* were incorrectly classified as *good*. These represent the most costly prediction errors, as they correspond to potential loan defaults being approved. Reducing these errors is crucial for minimizing financial exposure.
- **False Negatives (Type II Error):** Cases where applicants with *good credit risk* were labeled as *bad*. While less financially damaging, these mistakes represent lost business opportunities and reduced lending efficiency.
- **True Positives and True Negatives:** The model correctly identified the majority of both creditworthy and non-creditworthy applicants, indicating strong discriminatory power between the two groups.

From a decision-analytic viewpoint, the model demonstrates a favorable trade-off between precision and recall, maintaining adequate caution against risky approvals (high recall for the minority class) without severely limiting lending opportunities. This balance is particularly desirable in financial applications where both risk minimization and customer inclusion are priorities.

References

- Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*, Springer, 2017.
- McCullagh, P., and Nelder, J. *Generalized Linear Models*, CRC Press, 2019.
- NumPy Developers. *NumPy Documentation*.
- scikit-learn Documentation, 2025..