Credit Default Prediction

Arka Roy

20-06-2024

Abstract

This report presents the methodology and results of a project aimed at predicting credit default using the UCI credit default dataset. Various machine learning models were applied, including Logistic Regression, Random Forest, and Boosting algorithms, with and without resampling techniques such as SMOTE, oversampling, and undersampling.

1 Introduction

Credit default prediction is crucial for financial institutions to manage risk and make informed lending decisions. This study uses the UCI credit default dataset consisting of 30,000 users with various features such as credit amount, demographic information, and payment history to predict whether a client will default on their payment next month.

2 Dataset

The dataset consists of the following primary columns:

- **ID**: ID of each client
- **LIMIT_BAL**: Amount of given credit in NT dollars (includes individual and family/supplementary credit)
- **SEX**: Gender (1=male, 2=female)
- **EDUCATION**: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown)
- **MARRIAGE**: Marital status (1=married, 2=single, 3=others)
- **AGE**: Age in years
- **PAY_0 to PAY_6**: Repayment status from April to September 2005
- **BILL_AMT1 to BILL_AMT6**: Amount of bill statement from April to September 2005
- **PAY_AMT1 to PAY_AMT6**: Amount of previous payment from April to September 2005
- **default.payment.next.month**: Default payment (1=yes, 0=no)

3 Methodology

3.1 Feature Engineering

Three new features were created:

- **TOTAL_BILL**: Sum of BILL_AMT1 to BILL_AMT6
- **TOTAL_PAY**: Sum of PAY_AMT1 to PAY_AMT6
- **PAY_TO_BAL_RATIO**: Ratio of TOTAL_PAY to TOTAL_BILL

3.2 Data Splitting

The data was split into 70% training and 30% testing sets.

3.3 Resampling Techniques

Class imbalance is a common issue in credit default datasets. To address this, the following resampling techniques were used:

- **SMOTE (Synthetic Minority Over-sampling Technique)**: Generates synthetic samples for the minority class by interpolating between existing minority samples.
- **Oversampling**: Randomly replicates minority class samples to balance the dataset.
- **Undersampling**: Randomly removes majority class samples to balance the dataset.

3.4 Modeling

Various models were applied to the dataset:

3.4.1 Logistic Regression (LGR)

Logistic Regression is a linear model used for binary classification problems. The model was trained with scaled data and feature importance was analyzed to select significant features.

3.4.2 Random Forest Classifier (RFC)

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes for classification. It is robust to overfitting and provides feature importance scores.

3.4.3 Boosting Models

Boosting algorithms combine multiple weak learners to form a strong learner. The following boosting models were used:

- **AdaBoost**: Adjusts the weights of incorrectly classified instances so that subsequent classifiers focus more on difficult cases.
- **XGBoost**: An optimized distributed gradient boosting library designed to be highly efficient and flexible. It uses a more regularized model formalization to control overfitting.

For each model, different resampling techniques were applied to handle class imbalance.

4 Results

4.1 Model Performance

The performance of different models with various resampling techniques is summarized below:

5 Discussion

The results demonstrate the impact of different resampling techniques on model performance. Here are the key observations:

- **Accuracy**: XGBoost and AdaBoost without resampling achieved the highest accuracy, indicating these models are effective at correctly predicting both default and non-default cases.
- **Precision**: Precision was highest for AdaBoost without resampling, suggesting it had fewer false positives.
- **Recall**: Models with undersampling generally had better recall, meaning they were better at identifying actual default cases, although at the cost of higher false positives.

Model	Accuracy	Precision	F1 Score	Recall	AUC
Adaboost	0.8167	0.6621	0.4416	0.3313	0.7749
Adaboost with SMOTE	0.7478	0.4409	0.4965	0.5682	0.7454
Adaboost with oversampling	0.7572	0.4590	0.5249	0.6131	0.7733
Adaboost with undersampling	0.7458	0.4434	0.5212	0.6321	0.7633
XGBoost	0.8173	0.6415	0.4731	0.3747	0.7662
XGBoost with SMOTE	0.7615	0.4580	0.4737	0.4905	0.7343
XGBoost with oversampling	0.7640	0.4674	0.5107	0.5628	0.7577
XGBoost with undersampling	0.7083	0.3992	0.4971	0.6588	0.7642
Random Forest with SMOTE	0.7805	0.4984	0.4837	0.4699	0.7459
Random Forest with oversampling	0.8073	0.5803	0.4952	0.4318	0.7561
Random Forest with undersampling	0.7400	0.4353	0.5158	0.6329	0.7665
Logistic Regression with SMOTE	0.6688	0.3624	0.4720	0.6763	-

Table 1: Model Performance Comparison

- **F1 Score**: XGBoost with oversampling had a relatively balanced F1 Score, indicating a good trade-off between precision and recall.
- **AUC**: The area under the ROC curve was fairly consistent across models, with the highest values observed for AdaBoost and XGBoost without resampling.

6 Conclusion

This study explored various machine learning models and resampling techniques to predict credit default. Among the models tested, XGBoost and AdaBoost without resampling yielded the highest accuracy. However, models with resampling techniques, particularly undersampling, showed better recall, indicating they were better at identifying default cases. Further optimization and feature engineering could improve these results. Future work may involve exploring other algorithms, refining feature selection, and tuning model hyperparameters to enhance performance.