Credit Card Default Prediction Based on XGBoost

Xinyang Liu*
College of Aerospace and Civil Engineering, Harbin Engineering University, Harbin, China
2022029108@hrbeu.edu.cn

Siyu Wang
College of Aerospace and Civil Engineering, Harbin Engineering University, Harbin, China
2022029107@hrbeu.edu.cn

ABSTRACT

We introduce a credit card default prediction model that utilizes XGBoost and RandomUnderSampler to tackle data imbalance issues. We examined extensive borrower data employing various machine learning techniques, such as random forest, logistic regression, AdaBoost, XGBoost, LightGBM, and support vector machine. XGBoost showed superior performance in accuracy and ROC AUC score. To enhance model performance, we applied SMOTE and RandomUnderSampler for data resampling. Through extensive exploratory data analysis and preprocessing, combined with hyper-parameter tuning using GridSearchCV, we developed a high-performing model. This model achieved a ROC AUC score of 0.7882, a precision rate of 21.67%, a recall rate of 79.14%, and an F1 score of 33.85%. Key features influencing credit card default prediction were identified as RevolvingUtilizationOfUnsecuredLines, DebtRatio, and MonthlyIncome. The use of feature importance and SHAP values offered valuable insights into how the model makes its decisions. Additionally, we generated default probabilities to identify high-risk customers. This model serves as an efficient tool for banks and financial institutions to manage credit card default risk, blending theoretical advancements with practical applicability.

CCS CONCEPTS

• **CCS Concept**; • **Information systems** → Information systems applications; Data mining;

KEYWORDS

XGBoost, Credit Card Default Prediction, Data Imbalance, Machine Learning, Risk Management

ACM Reference Format:
Xinyang Liu* and Siyu Wang. 2024. Credit Card Default Prediction Based on XGBoost. In *2024 International Conference on Machine Intelligence and Digital Applications (MIDA 2024), May 30, 31, 2024, Ningbo, China.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3662739.3669913

Machine learning has recently become a key technology across multiple domains within computer science, demonstrating significant advancements in pattern recognition, data mining, and predictive analytics [1]. One of its critical applications in the financial industry is credit card default prediction, which involves using borrowers' historical and personal information to estimate the probability of future default. This task presents significant challenges because of the large and complex nature of the datasets involved, as well as the prevalent issue of data imbalance [2].

Credit card default prediction is crucial for risk management and decision-making processes within financial institutions, as accurate predictions can significantly reduce financial risks and improve credit granting processes[3]. Historically, machine learning algorithms like random forest, logistic regression and support vector machine have been utilized for this purpose [4]. While these methods perform adequately on standard datasets, their effectiveness diminishes when confronted with imbalanced data, a frequent characteristic in credit card default datasets. Additionally, many existing methods lack sufficient interpretability and prediction accuracy.

To tackle these issues, this study introduces a credit card default prediction model leveraging XGBoost and RandomUnderSampler [5]. XGBoost is a powerful and scalable gradient boosting algorithm, has demonstrated superior performance in various machine learning tasks, particularly in handling structured or tabular data. The RandomUnderSampler technique is employed to mitigate the data imbalance issue, ensuring that the model does not become biased towards the majority class [6].

Our approach is structured in several key steps: first, exploratory data analysis is conducted to comprehend the fundamental features of the dataset and identify potential data imbalance issues. Subsequently, the XGBoost algorithm is utilized as the primary classifier to enhance prediction accuracy through its robust gradient boosting mechanism. Data imbalance is then addressed using the RandomUnderSampler technique. Finally, SHAP values [7] are applied for feature importance analysis to enhance the interpretability of the model.

The test outcomes confirm our model's superiority in key performance indicators like precision, recall, F1 score, and ROC AUC, underscoring its efficacy in predicting credit card defaults.

The main contributions of our study include the development of an innovative credit card default prediction model that effectively addresses data imbalance and enhances prediction accuracy through integrated learning and undersampling techniques. Additionally, the model's interpretability is significantly improved by leveraging SHAP value analysis, offering a powerful tool for financial institutions' risk management and decision-making processes.

1 RELATED WORK

1.1 Traditional Statistical Methods

Initial investigations concentrated on employing conventional statistical techniques, including logistic regression and linear discriminant analysis [8]. These methods are superior in understanding and interpreting models, but have limited performance when dealing with large and complex datasets. In contrast, our study employs

MIDA 2024, May 30, 31, 2024, Ningbo, China

ACM ISBN 979-8-4007-1814-4/24/05
https://doi.org/10.1145/3662739.3669913

the XGBoost algorithm, which improves the ability to handle large-scale datasets and has better recognition of nonlinear relationships.

1.2 Machine Learning Methods

As ML technology advanced, algorithms [9] are beginning to be applied to default prediction. These methods can effectively handle nonlinear data, but they still face challenges in dealing with unbalanced data. This study addresses the data imbalance problem by combining XGBoost and RandomUnderSampler, which is a key point often overlooked in previous studies.

1.3 Integrated Learning and Unbalanced Data Processing

Integration learning methods, such as AdaBoost [10] and XGBoost [11], are favored for the high accuracy they exhibit on multiple classification problems. However, most research on integrated learning methods has not adequately addressed the problem of data imbalance. By combining XGBoost and RandomUnderSampler techniques, our approach increases classification accuracy and enhances the model's capacity to deal with imbalanced data.

1.4 Deep Learning Methods

Neural networks show significant potential in predicting credit card defaults. Although they excel at processing large-scale data, they generally lack interpretability. While this study does not directly employ deep learning methods, it takes new steps in improving model interpretability, an often missing piece of deep learning methods.

1.5 Feature Importance and Model Interpretability

Some study [12] provides a comprehensive overview of these interpretability methods. As model complexity increases, researchers have begun to employ tools such as SHAP values [13] and LIME [14] to explain decisions in complex models. The implementation of advanced algorithms for evaluating credit risk is analyzed, while emphasizing the importance of model interpretability [15]. While these methods have improved model transparency, few studies have combined these methods with machine learning models for credit card default prediction. Our research addresses this gap by using SHAP values to improve model interpretability and offer insights into the critical factors influencing credit card default prediction.

To summarize, this study builds on existing research on credit card default prediction by introducing a combination of integrated learning and data imbalance processing with a special emphasis on model interpretability, resulting in advances in both prediction accuracy and model transparency.

2 METHODS

2.1 Data Loading

The dataset we use is stored in the file "cs-training.csv" and includes information about the borrowers and whether they have defaulted on credit card payments. The dataset includes attributes such as borrower's age, income, number of dependents and other financial indicators.

2.2 Exploratory Data Analysis

In this section, we performed several tasks, including: (1) Checking data types and null values. (2) Obtaining descriptive statistics. (3) Plotting the distribution of defaults. It can be seen that 10026 or 6.68% of the customers in the dataset have serious defaults (SeriousDlqin2yrs). Correlations between features are explored using heatmaps. We found that the features 'NumberOfTimes90DaysLate', 'NumberOfTimes60-89DaysPastDueNotWorse' and 'NumberOfTimes30-59DaysPastDueNotWorse' were highly correlated, so we combined them into one new feature named 'TotalDelinquencies90DaysLate'. Visualize the distribution of age and monthly income. Analyze the relationship between age/monthly income and defaults.

2.3 Data Preprocessing

Data preprocessing steps include: (1) Deleting unnecessary columns, such as customer ID. (2) Create a composite feature called "TotalDelinquencies90DaysLate" by merging related columns. (3) Use StandardScaler to process missing values and normalized numeric features. (4) Dividing the dataset into training, validation, and test subsets.

2.4 Model Training and Experimentation

In this section, we trained and assessed the performance of multiple machine learning algorithms, including: (1) Logistic Regression, (2) XGBoost Classifier, (3) Random Forest Classifier, (4) AdaBoost Classifier, (5) LightGBM Classifier, and (6) SVC.

To facilitate this, we developed a processing pipeline for each model that incorporated both preprocessing steps and the respective classifiers. The models were trained using the training dataset and evaluated using the validation dataset. The evaluation metrics employed included accuracy, ROC AUC score, and F1 score.

Among all the models, the XGBoost classifier exhibited the highest performance in terms of both accuracy and ROC AUC score, leading us to select it as the preferred model.

2.5 Hyperparametric Tuning

We performed Since our XGBoost model still performs poorly, we decided to use SMOTE [16] and RandomUnderSampler [17] for oversampling and undersampling because the dataset suffers from severe imbalance balancing problems.

3 EXPERIMENT

The dataset used in this study is "cs-training.csv," which contains borrower information such as age, income, and indebtedness. We evaluated the models using metrics like accuracy, ROC AUC score, and F1 score and compared the performance with baseline models.

3.1 Exploratory Data Analysis

Initially, we conducted an exploratory data analysis to grasp the fundamental attributes of the dataset. We looked at data types, null values and generated descriptive statistics. In addition, we plotted the distribution of different features and heatmaps to explore the relationship between the features. Among them, 'NumberOfTimes90DaysLate', 'NumberOfTimes60-89DaysPastDueNotWorse'

	count	mean	std	min	25%	50%	75%	max
CustomerID	150000.000000	75000.500000	43301.414527	1.000000	37500.750000	75000.500000	112500.250000	150000.000000
SeriousDlqin2yrs	150000.000000	0.066840	0.249746	0.000000	0.000000	0.000000	0.000000	1.000000
RevolvingUtilizationOfUnsecuredLines	150000.000000	6.048438	249.755371	0.000000	0.029867	0.154181	0.559046	50708.000000
age	150000.000000	52.295207	14.771866	0.000000	41.000000	52.000000	63.000000	109.000000
NumberOfTime30-59DaysPastDueNotWorse	150000.000000	0.421033	4.192781	0.000000	0.000000	0.000000	0.000000	98.000000
DebtRatio	150000.000000	353.005076	2037.818523	0.000000	0.175074	0.366508	0.868254	329664.000000
MonthlyIncome	120269.000000	6670.221237	14384.674215	0.000000	3400.000000	5400.000000	8249.000000	3008750.000000
NumberOfOpenCreditLinesAndLoans	150000.000000	8.452760	5.145951	0.000000	5.000000	8.000000	11.000000	58.000000
NumberOfTimes90DaysLate	150000.000000	0.265973	4.169304	0.000000	0.000000	0.000000	0.000000	98.000000
NumberRealEstateLoansOrLines	150000.000000	1.018240	1.129771	0.000000	0.000000	1.000000	2.000000	54.000000
NumberOfTime60-89DaysPastDueNotWorse	150000.000000	0.240387	4.155179	0.000000	0.000000	0.000000	0.000000	98.000000
NumberOfDependents	146076.000000	0.757222	1.115086	0.000000	0.000000	0.000000	1.000000	20.000000

Figure 1: Descriptive stastistics.

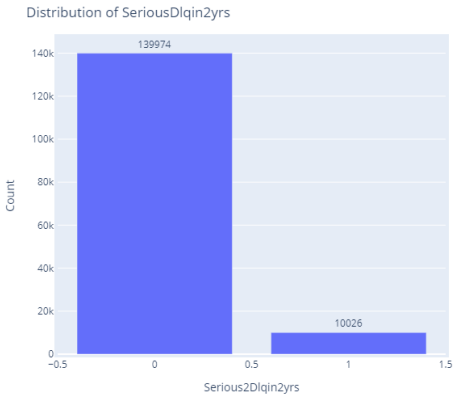

Figure 2: Distribution of SeriousDlqin2yrs.

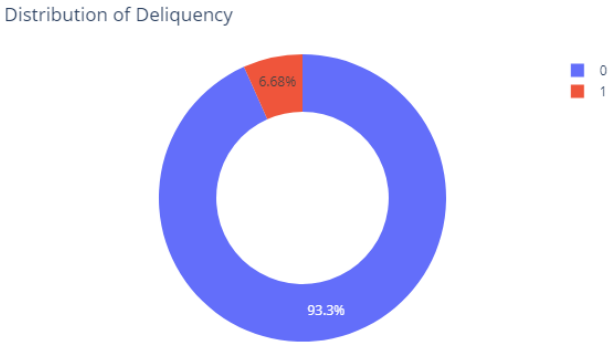

Figure 3: Distribution of Deliquency.

'NumberOfTimes30-59DaysPastDueNotWorse' are highly correlated, so we merged these three into a new feature ' TotalDelinquencies90DaysLate'.

We then also plotted the distributions of age and monthly income and analyzed the relationship between these two characteristics and default. The summary statistics of the dataset are displayed in Figure 1. Distribution of deliquency as shown in Figure 2 and Figure 3. Exploring the correlation between features using heat maps is shown in Figure 4. 'NumberOfTimes90DaysLate', 'NumberOfTime60-89DaysPastDueNotWorse' 'NumberOfTime30-59DaysPastDueNotWorse' are all highly correlated, we will aggregate into a single feature 'TotalDelinquencies90DaysLate'.

Figure 5 visualizes the distribution of age and monthly income and includes an analysis of the relationship between age/monthly income and delinquent debt.

3.2 Data Preprocessing

In our study, data preprocessing is an essential step to clean and transform the raw data into a format suitable for model learning. In this study, our data preprocessing steps are as follows:

First, we imported the pandas library and loaded our dataset with its read_csv() function, which is stored in the path 'Data/cs-training.csv'. Then, to facilitate the processing and understanding of the data, we modified the column names of the data, in which, the original meaningless 'Unnamed: 0' column was renamed to 'CustomerID'. In the feature engineering phase, we created a composite feature 'TotalDelinquencies90DaysLate', which is a combination of 'NumberOfTimes90DaysLate', 'NumberOfTime60-89DaysPastDueNotWorse' and ' NumberOfTime30-59DaysPastDueNotWorse' obtained from the sum of these three features. We do this step to extract the shared information of these three features to enhance the subsequent data analysis and modeling. Next, we removed these three features to prevent multicollinearity issues during the model's learning process. In addition, we also deleted the 'CustomerID' column, this is because 'CustomerID' is not a meaningful feature and cannot provide useful information for our model.

Subsequently, we partitioned the dataset into training and testing subsets. In this step, we first created the feature set, which consists of all the columns in the dataset except the 'SeriousDlqin2yrs' column, and the target set, which is the 'SeriousDlqin2yrs' column. We then split the dataset, allocating 20% of the data to the test set and the remaining 80% to the training set. Afterward, we further divided the training set, designating 25% of it as the validation set.

Finally, we pre-processed the numerical features. The preprocessing step consisted of two parts: first, we populated the missing values of the features with SimpleInputer, using a strategy that used the median of the features; then, we normalized the features with StandardScaler, resulting in each feature having a mean of 0 and a standard deviation of 1. These preprocessing steps guaranteed the data's quality and facilitated effective model learning.

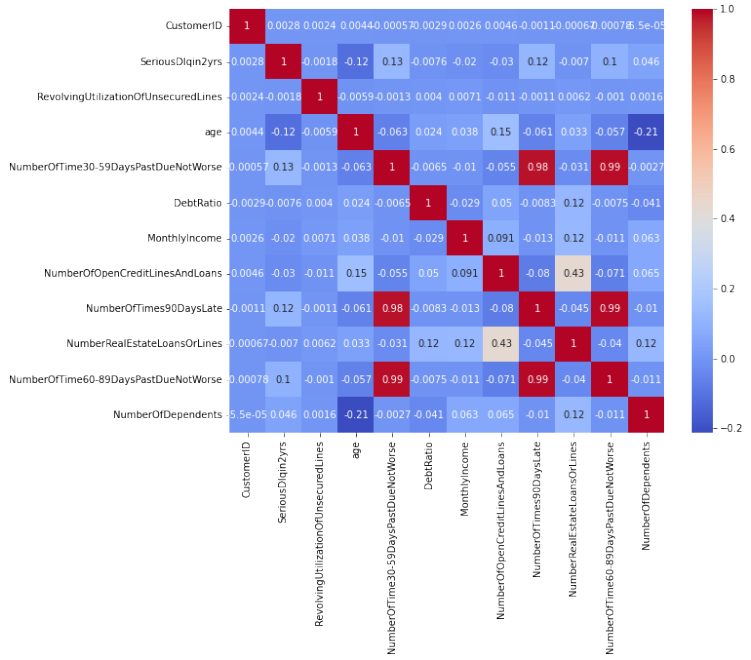

Figure 4: Heat map of correlation between features.

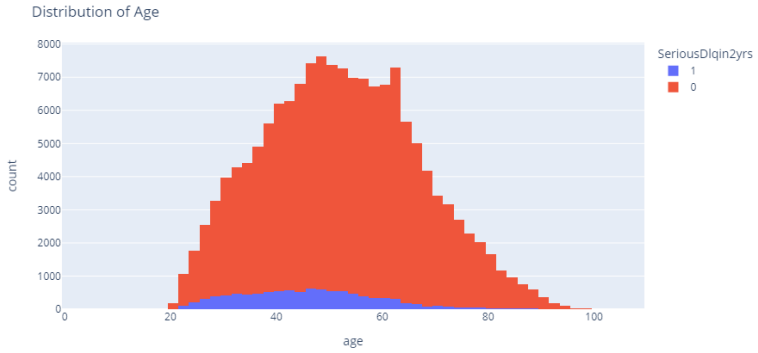

Figure 5: Distribution of Age.

3.3 Model Training and Experimentation

In this stage of the research, several machine learning algorithms were utilized to train and assess the preprocessed data. The ensemble of classifiers comprised Logistic Regression, Random Forest, Support Vector Classifier (SVC) [18], AdaBoost [19], XGBoost, and LightGBM [20] . All these classifiers are instantiated with a set random state of 42 to ensure consistency in the results, except for LightGBM Classifier which is set at 32.

Each classifier is included in a dictionary (named classifiers) with its respective name as the key. This approach allows for the easy management of multiple models, facilitating subsequent iteration over them for model training and validation.

To efficiently preprocess the data and fit each model, a pipeline is utilized, which links the predefined data preprocessing steps and the classifier together. This pipeline is fitted with the training data. Afterwards, predictions are made on the validation set, which are then used to compute evaluation metrics including accuracy, ROC AUC, recall, and F1 score.

These metrics for each model are stored in a Pandas DataFrame, providing a structured format for comparison. It is important to note that each metric provides a unique perspective on the performance of the models, and thus, the final model selection should consider a balance among these metrics rather than relying on a single one.

The process triggers warnings with XGBoost and LightGBM indicating changes in the evaluation metrics and selection of multithreading, respectively, which are normal notifications providing information about the internal operations of these algorithms.

Table 1: Model performance metrics.

Model	Accuracy	ROC AUC	F1 Score
XGBClassifier	0.9351	0.5938	0.2969
LGBMClassifier	0.9348	0.5848	0.2735
AdaBoostClassifier	0.9338	0.5823	0.2692
RandomForestClassifier	0.9330	0.5769	0.2528
LogisticRegression	0.9308	0.5077	0.0317
SVC	0.9308	0.5041	0.0170

Upon executing the code, we obtain a table of results showing the performance metrics for each model. This tabular format allows for clear visual comparison of the performance metrics across different models. The results reveal differences in the performance of various classifiers on the validation set.

Finally, the results DataFrame is ordered by the Accuracy metric in descending sequence. This arrangement highlights which model achieved the highest accuracy on the validation set. However, selecting the model with the top accuracy doesn't necessarily mean it's the optimal choice. As previously mentioned, the choice of the final model should consider a balance among all the performance metrics, as well as the specific requirements and constraints of the problem at hand.

Of all the algorithms tested, the XGBoost classifier delivered the highest performance regarding accuracy and ROC AUC scores, so we chose it as our final model. The correctness results for all models are shown in Table 1.

3.4 Hyperparameter Tuning

To enhance the machine learning models' performance, we applied hyperparameter tuning to the XGBoost classifier using GridSearchCV. Despite this effort, the improvements in the model's performance were minimal.

Recognizing the severe class imbalance in the dataset, which could impede the model's accuracy in predicting the minority class, we explored various resampling methods to address this problem. We specifically implemented the Synthetic Minority Over-sampling Technique (SMOTE) and Random Under Sampling (RUS).

The preprocess_data function was created for preprocessing and resampling the data. It accepts the training, validation, and testing data, along with a list of numerical features and the desired sampling method ('smote' or 'undersampler').

In the preprocessing steps, missing values in numerical features are imputed using the median strategy, and the numerical features are standardized using StandardScaler. The sampling strategy is then applied to the preprocessed training data, with the choice between SMOTE and RandomUnderSampler based on the input sampling method.

Following the preprocessing and resampling steps, the data is transformed back to DataFrame format for easier manipulation and analysis in subsequent steps.

To assess the effectiveness of the models on the resampled data, we defined a distinct function called evaluate_performance. This function predicts the target variable using the provided model and dataset, then calculates various performance metrics such as accuracy, recall, precision, F1 score, and ROC AUC score. Additionally, it generates a confusion matrix and a classification report to offer a more detailed understanding of the model's performance.

Using these preprocessing, resampling, and evaluation steps, we experimented with the SMOTE and RandomUnderSampler methods, seeking to address the class imbalance issue and thereby improve the models' performance.

3.4.1 XGBoostClassifier. For the XGBoostClassifier results, we display the key performance metrics using bar charts, while using the Confusion Matrix and Classification Report to provide a more detailed view of performance. The validation score is 0.8614. The hyperparameters are listed in Table 2. Evaluation on Validation set as shown as Table 3. Classification performance as shown as Table 4.

3.4.2 SMOTE. After SMOTE oversampling, we found the best hyperparameters and obtained better results on the validation set. Validation Score is 0.9841. The hyperparameters are listed in Table 5. Evaluation on Validation set as shown as Table 6. Classification performance as shown as Table 7.

3.4.3 Random Under Sampler. After Random Under Sampler undersampling, we also found a different set of optimal hyperparameters and obtained better results on the validation set. Although the accuracy has decreased, the recall, precision, F1 score and ROC AUC score have improved. Validation Score is 0.8593. The hyperparameters are listed in Table 8. Evaluation on Validation set as shown as Table 9. Classification performance as shown as Table 10.

Overall, our experimental results show that the XGBoost classifier performs best in dealing with this problem, however, due to the imbalance of the dataset, sampling the data is necessary. After the sampling process, the performance of the model improved significantly.

4 RESULTS AND ANALYSIS

The outcomes of the model training and hyperparameter tuning are compiled in a table, detailing the model names, accuracy, ROC AUC scores, and F1 scores. The XGBoost model, paired with RandomUnderSampler, attained the highest ROC AUC score of 0.7882, along with an accuracy of 78.54%, a precision of 21.53%, a recall of 79.14%, and an F1 score of 33.85%.

4.1 Model Evaluation

During the evaluation phase, we initially extracted the predicted probabilities for the target class (class 1) from the tuned XGBoost

Table 2: Hyperparameter values.

Hyperparameter	Value
colsample_bytree	0.8
gamma	0.1
learning_rate	0.1
max_depth	4
n_estimators	100
subsample	1.0

Table 3: Validation set performance.

Accuracy	Recall	Precision	F1 score	ROC AUC score
0.9350	0.1691	0.6154	0.2654	0.5806

Table 4: Classification performance.

	Precision	Recall	F1-Score	Support
0	0.94	0.99	0.97	27919
1	0.62	0.17	0.27	2081
Accuracy			0.94	30000
Macro Avg	0.78	0.58	0.62	30000
Weighted Avg	0.92	0.94	0.92	30000

Table 5: Hyperparameter values.

Hyperparameter	Value
colsample_bytree	1.0
gamma	0.2
learning_rate	0.1
max_depth	5
n_estimators	300
subsample	0.8

model. To illustrate the model's performance, we created a Receiver Operating Characteristic (ROC) curve, which demonstrates the trade-off between the true positive rate (TPR) and the false positive rate (FPR) at various thresholds. Furthermore, we computed the Area Under the Curve (AUC) to encapsulate the model's performance across all classification thresholds.

Table 6: Validation set performance.

Accuracy	Recall	Precision	F1 score	ROC AUC score
0.9287	0.3133	0.4791	0.3788	0.6440

Table 7: Classification performance.

	Precision	Recall	F1-Score	Support
0	0.95	0.97	0.96	27919
1	0.48	0.31	0.38	2081
Accuracy			0.93	30000
Macro Avg	0.71	0.64	0.67	30000
Weighted Avg	0.92	0.93	0.92	30000

Table 8: Hyperparameter values.

Hyperparameter	Value
colsample_bytree	0.8
gamma	0.2
learning_rate	0.1
max_depth	3
n_estimators	100
subsample	0.8

Table 9: Validation set performance.

Accuracy	Recall	Precision	F1 score	ROC AUC score
0.7866	0.7943	0.2167	0.3405	0.7902

Table 10: Classification performance.

	Precision	Recall	F1-Score	Support
0	0.98	0.79	0.87	27919
1	0.22	0.79	0.34	2081
Accuracy			0.79	30000
Macro Avg	0.60	0.79	0.61	30000
Weighted Avg	0.93	0.79	0.84	30000

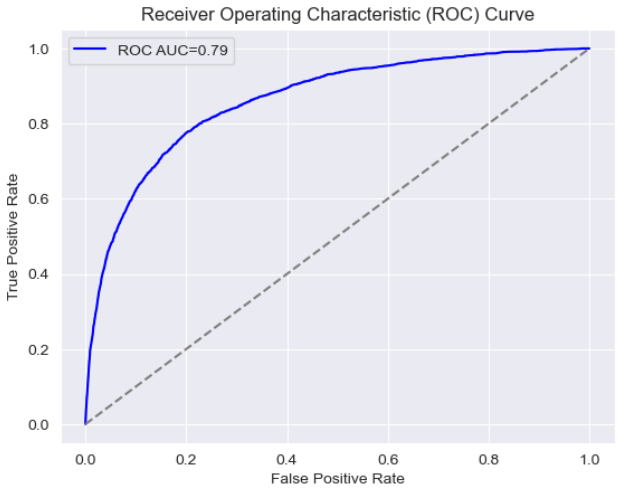

Figure 6: Receiver Operating Characteristic (ROC) Curve.

We generated the ROC curve using seaborn's lineplot function, plotting FPR values on the x-axis and TPR values on the y-axis. A line representing random guessing was added for comparison. The AUC value, which reflects the model's capacity to differentiate between classes, was computed and displayed in the ROC curve legend (as shown in Figure 6).

We then created a confusion matrix for the XGBoost model to analyze its classification performance in more detail. This matrix offers an in-depth look at the model's effectiveness by displaying the counts of true positives, false positives, true negatives, and false negatives. We visualized the confusion matrix using a seaborn heatmap, placing actual labels on the y-axis and predicted labels on the x-axis (as shown in Figure 7).

We calculated the overall accuracy of the XGBoost model, which indicates the proportion of correct predictions, and displayed it in the title of the confusion matrix heatmap.

In conclusion, the evaluation of the XGBoost model demonstrated its good capability in distinguishing between classes, though improvements can still be made, especially in accurately classifying the minority class. The insights gained from this evaluation are crucial for guiding further model tuning and selection efforts.

4.2 Model Interpretation

We performed feature importance analysis on the XGBoost model and identified the most important features for predicting credit card defaults, including "RevolvingUtilizationOfUnsecuredLines," "DebtRatio," and "MonthlyIncome" (as shown in Figure 8). Additionally, we calculated the SHAP values (as shown in Figure 9) and plotted the SHAP value distributions (as shown in Figure 10).

5 CONCLUSION

In this project, we have developed credit card default prediction models using various machine learning algorithms. The XGBoost model sampled using RandomUnderSampler achieved the best performance with relative ROC AUC scores as well as reasonable accuracy, delinquency rates and F1 scores.

Feature-emphasized analysis shows that features such as "RevolvingUtilizationOfUnsecuredLines", "DebtRatio", and "MonthlyIncome" play a crucial role in predicting credit card defaults. Credit card companies can use this information to identify customers at

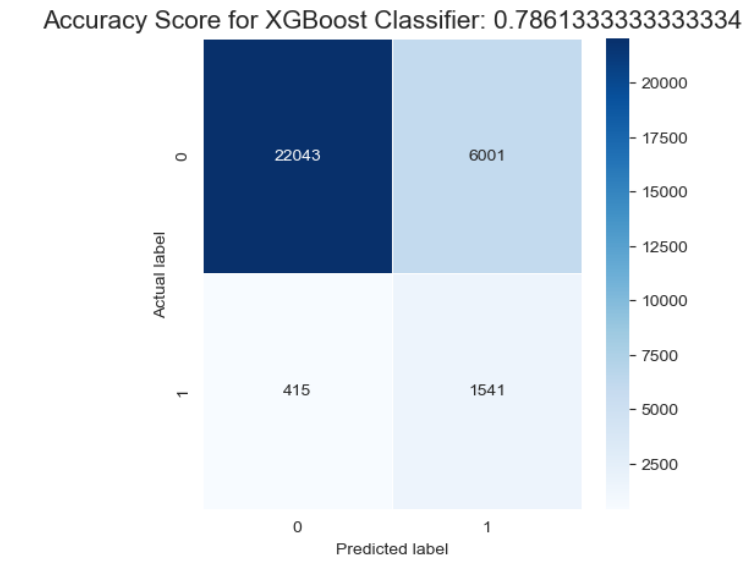

Figure 7: Confusion Matrix.

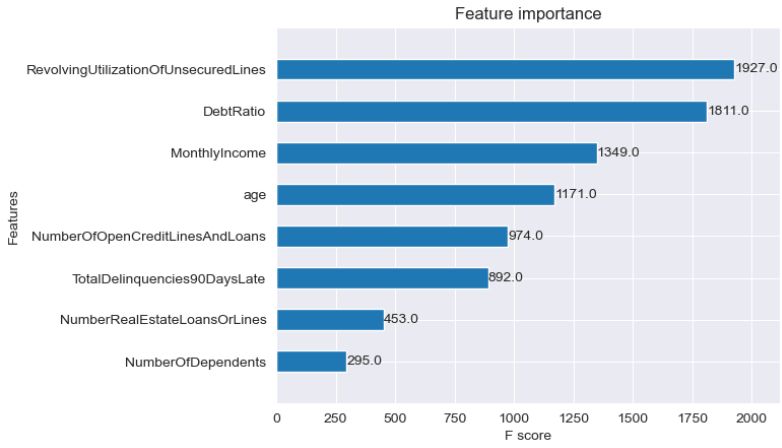

Figure 8: Feature Importance.

higher risk of default and take appropriate measures to reduce the risk.

Overall, the project provides credit card companies with a valuable tool to make informed decisions about customer creditworthiness, enabling them to effectively manage their portfolios and reduce default risk.

REFERENCES

[1] Kim, H., Cho, H., & Ryu, D. (2020). Corporate default predictions using machine learning: Literature review. Sustainability, 12(16), 6325.

[2] Ciampi, F., Giannozzi, A., Marzi, G., & Altman, E. I. (2021). Rethinking SME default prediction: a systematic literature review and future perspectives. Scientometrics, 126, 2141-2188.

[3] Butaru, F., Chen, Q., Clark, B., Das, S., Lo, A. W. *et al.* (2016). Risk and risk management in the credit card industry. Journal of Banking & Finance, 72, 218-239.

[4] Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. European Journal of Operational Research, 247(1), 124-136.

[5] Chang, Y. C., Chang, K. H., & Wu, G. J. (2018). Application of eXtreme gradient boosting trees in the construction of credit risk assessment models for financial institutions. Applied Soft Computing, 73, 914-920.

[6] Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. International Journal of Emerging Technology and Advanced Engineering, 2(4), 42-47.

[7] Lundberg, S. M., Erion, G. G., & Lee, S. I. (2018). Consistent Individualized Feature Attribution for Tree Ensembles. methods, 5(13), 25.

[8] Bellotti, T., & Crook, J. (2009). Credit scoring with macroeconomic variables using survival analysis. Journal of the Operational Research Society, 60(12), 1699-1707.

[9] Barboza, F., Kimura, H., & Altman, E. (2017). Machine learning models and bankruptcy prediction. Expert Systems with Applications, 83, 405-417.

[10] Liang, S., Wang, L., Zhang, L., & Wu, Y. (2018). Research on recognition of nine kinds of fine gestures based on adaptive AdaBoost algorithm and multi-feature combination. Ieee Access, 7, 3235-3246.

[11] Yan, Z., Chen, H., Dong, X., Zhou, K., & Xu, Z. (2022). Research on prediction of multi-class theft crimes by an optimized decomposition and fusion method based on XGBoost. Expert Systems with Applications, 207, 117943.

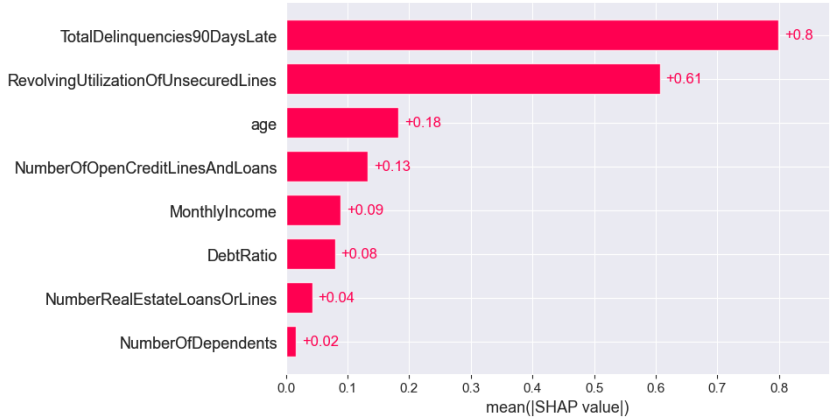

Figure 9: Calculate SHAP values.

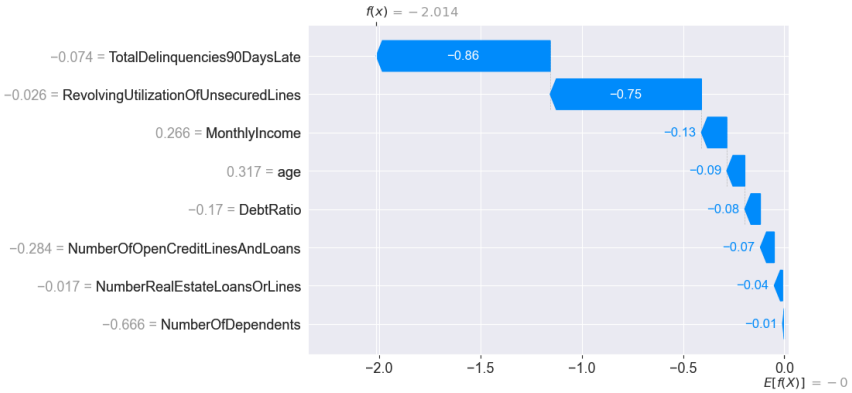

Figure 10: Plot the SHAP values.

[12] Du, M., Liu, N., & Hu, X. (2019). Techniques for interpretable machine learning. Communications of the ACM, 63(1), 68-77.
[13] Lundberg, S. M., & Lee, S. I. (2017, December). A unified approach to interpreting model predictions. In Proceedings of the 31st International Conference on Neural Information Processing Systems (pp. 4768-4777).
[14] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, August). "Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1135-1144).
[15] Khandani, A. E., Kim, A. J., & Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. Journal of Banking & Finance, 34(11), 2767-2787.
[16] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.
[17] Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD explorations newsletter, 6(1), 20-29.
[18] Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery, 2(2), 121-167.
[19] Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55(1), 119-139.
[20] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W. *et al.* (2017, December). LightGBM: a highly efficient gradient boosting decision tree. In Proceedings of the 31st International Conference on Neural Information Processing Systems (pp. 3149-3157).