Proceedings of 14th IOE Graduate Conference
Peer Reviewed
Year: 2023 Month: December Volume: 14
ISSN: 2350-8914 (Online), 2350-8906 (Print)

Enhancing Default Prediction using Boosting based ML Techniques

Bijaya Bhatta [a], Arun K. Timalsina [b], Niranjan Shrestha [c]

[a, b, c] *Department of Electronics and Computer Engineering, Pulchowk Campus, IOE, Tribhuvan University, Nepal*

[a] 078msdsa002.bijaya@pcampus.edu.np, [b] t.arun@pcampus.edu.np, [c] 079mscsk013.niranjan@pcampus.edu.np

Abstract

When granting a loan, there is always some sort of credit risk. The likelihood of suffering an inconvenience as a result of the borrower's inability to make payments on loans or honour contractual obligations is known as credit risk. In order to determine whether a borrower is qualified to receive a loan or not, loan default probability is predicted using machine learning algorithm. If borrower does not default is predicted then it is eligible to be granted a loan. In this paper, data pre-processing, exploratory data analysis and model training using two baseline models Logistic Regression, Gradient Boosting and three Boosting based models XGBoost, LightGBM and CatBoost algorithm was done. There were less default cases in dataset causing class imbalance problem. It was solved by oversampling using SMOTE technique. After performing oversampling, it was found that the performance was increased of all the models. This is especially evident in the recall metric, which improves for all models by maximum of 17-20%. There is improvement in precision by 5 to 6% in all the models. F1-Score of models increased by 11 to 14%. LightGBM followed by CatBoost was the best performer among all with the highest F1-score 96%, accuracy 96% and AUC score 99. It was found Boosting algorithms showed better performance than baseline algorithms after oversampling. EDA gave insights that loan taken for venture was least defaulted among all covering 11.9%, loans taken for medical and debt consolidation reasons got defaulted the most which covered 45% of total loan default.

Keywords

Credit Risk, EDA, Logistic Regression(LR), Gradient Boosting (GB), LightGBM, CatBoost, SMOTE, ROC Curve

1. Introduction

People usually take loans from their known ones but due to lack of trust, formal agreements people chose to issue loans from lending companies and banks. When a bank gets a request for a loan it is necessary to make a wise decision on approving or rejecting the loan. Accepting a loan which isn't likely to be paid back or not issuing a capable loan borrower, both are risk cases. Traditionally, loan approval systems used to be manual and it would take months to verify a customer's details and make a decision. There used to be delays in processing and service delivery would also be late. By using digital lending, an intelligent agent will do the overall processing and give a decision whether to allow or reject a loan application. The paperwork will be reduced, efficient record keeping as well as the privacy of the customers would be increased. In this paper two baseline machine learning **(ML)** models namely, Logistic Regression **(LR)**, Gradient Boosting **(GB)** and three boosting models: Extreme Gradient Boosting **(XGB)**, LightGBM **(LGBM)** and CatBoost (**CB**) are used, enhanced by Synthetic Minority Oversampling Technique **(SMOTE)** and compared to find best model that can predict loan default more accurately.

2. Related Works

Previous works like [1, 2, 3, 4, 5, 6, 7, 8, 9] explored various machine learning approaches for loan default prediction. Machine Learning models: Random Forests [5, 1, 2, 9], Gradient Boosting [5], Decision Trees [5, 1, 9], Support Vector Machine [5], K nearest neighbor [5], Logistic Regression [5, 6, 1, 9] etc. were used to predict loan eligibility.

[5] used loan eligible dataset from kaggle website [10] having 13 features and 367 cases. SMOTE technique to handle class imbalance problems, one hot encoding to convert categorical variables into binary form, normalization, and Exploratory Data Analysis **(EDA)** was done. Accuracy was found out 80% in LR, 93.33% in K-Nearest Neighbour, 84.44% in Support Vector Machine, 91.11% in Decision Tree, 95.55% in Random Forests **(RF)** and 93.33% while using GB.

[7] leveraged the lender information dataset on Tianchi platform using the LightGBM model with a final AUC value of around 0.73. [11] used convolutional neural networks and the LightGBM algorithm to create a prediction model. First, features were extracted from the original loan data and a new feature matrix was created using the convolutional neural network's superior feature extraction capability. Second, the LightGBM model was constructed by using the updated feature matrix as input data and adjusting the LightGBM algorithm's parameters via grid search.

[2] used dataset from lending club using RF and XGBoosting classifier. Among 142 features with 2,72,401 records in raw data set only 26 features were selected to train the model. EDA found out grade A and grade B were the most popular loans favoured by the lender, Mortgage was highest count of ownership i.e most borrowers have house loan. Accuracy, precision, recall and f1-score was similar for both which is 0.97, 0.98, 0.96 and 0.97 respectively. Based on AUC, RF scored 0.99 and XGBoosting scored 0.97.

[12] used Catboost algorithm to predict the probabilty of loan default along with document verification module. System was made to classify the customers data into four categories like excellent credit, good credit, bad credit and low credit and as

per users data also provided personalized loan recommendations to the approved applicants.

3. Dataset

The data was taken from credit bureau simulating credit data from kaggle website [13] whose author was Lao Tse. It has 12 features and around 32k records. The attributes are listed in Table 1.

Table 1: Data Attributes

SN	Feature Name	Description
1	person_age	age of the customer
2	person_income	annual income
3	person_home_ownership	home ownership status
4	person_emp_length	years of employment
5	loan_intent	intent of the loan
6	loan_grade	grade of the loan
7	loan_amnt	total loan amount
8	loan_int_rate	interest rate
9	loan_status	default or not
10	loan_percent_income	percent income
11	cb_person_default_on_file	historical default
12	cb_person_cred_hist_length	credit history length

4. Methodology

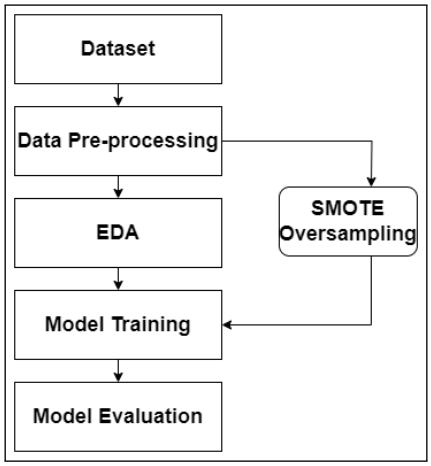

Figure 1: Block Diagram of Methodology

4.1 Data Pre-processing

The missing values, outliers, irrelevant and duplicate data were removed. The outliers were removed. Since categorical data can't be trained by ML model they were identified and changed into category type and encoded into integer form. The dataset had 165 duplicate data which was later removed. All data with a person's age more than 100 was removed. A person with an annual income of five million was screwing the data, so it was removed as an outlier. Person-emp-length and loan-int-rate columns had missing values. Person employment length had only few missing values and since it didn't have high correlation (only -0.083) with loan default status, missing values were dropped for this column. Loan interest rate depended on the type of loan one is getting so its missing values were imputed by the median value of interest rate of respective loan grade. Multicollinearity was removed as it would create bias between the variables.

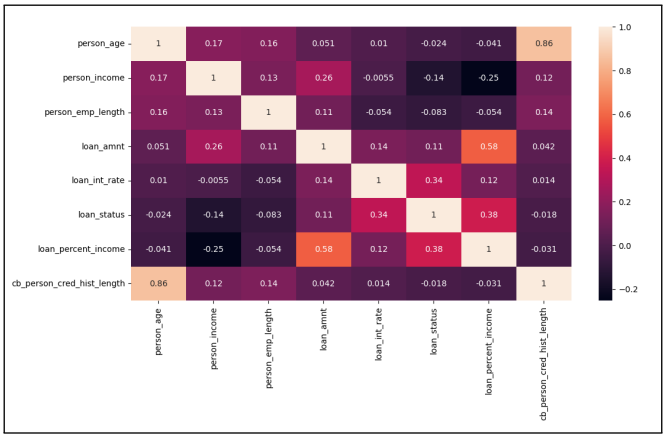

Figure 2: Correlation of all variables

4.2 Exploratory Data Analysis (EDA)

EDA studies dependent and independent variables using normal distribution, probability density function, correlation, univariate and bivariate and multivariate analysis. It gives insights to viewers which may not be visible in data values but clearly expressed in graphs and pictorial representations. It was found that loans taken for medical and debt consolidation reasons got defaulted the most which covered 45% of total loan default. Loan taken for venture was least defaulted among all covering 11.9%. The people with the age 23 years take the highest amount of loans. About 76% of total loans were taken by people within the age group 20 and 30. It was observed that almost all borrowers default regardless of their age. About 18% of borrowers had a historical record of default on file.

Table 2: Loan Grade Performance

Loan Grade	No. of Cases	No. of Defaults	Percentage
A	10301	985	9.6%
B	10124	1616	16.0%
C	6303	1280	20.3%
D	3550	2087	58.8%
E	951	611	64.2%
F	236	166	70.3%
G	64	63	98.4%

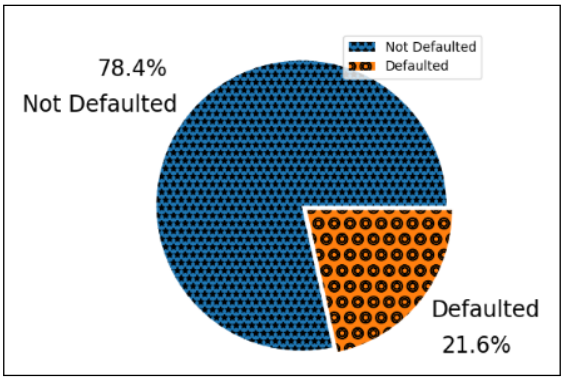

Figure 3: Default and Non-default cases

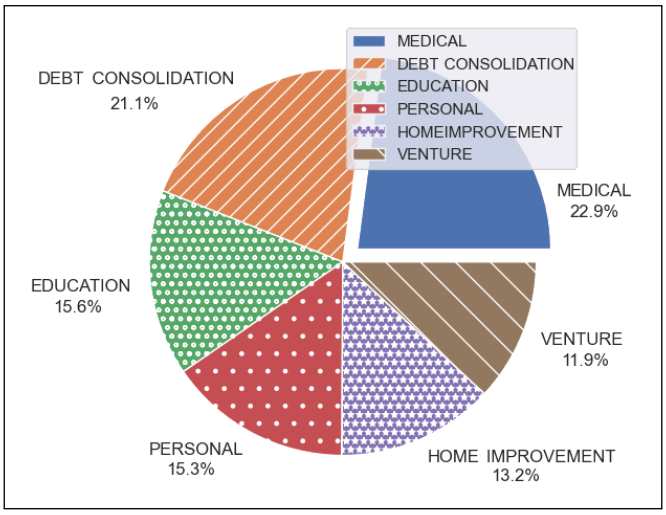

Figure 4: Loan default cases based on Loan intend

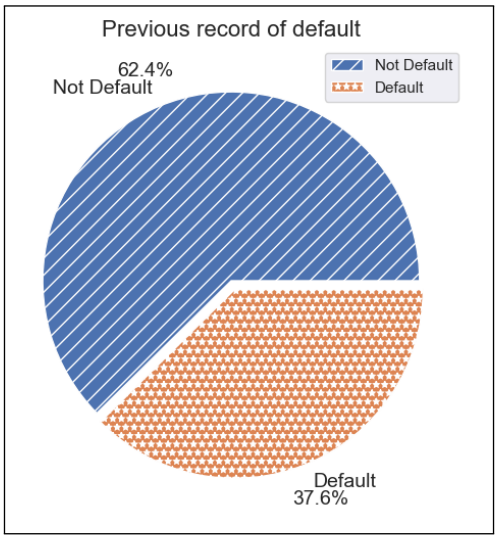

Figure 5: Loan default v/s. Previous default cases

5. Oversampling by SMOTE

Generally, people who default on loan will be less in number than the people who haven't defaulted.The dataset is hence imbalanced in nature and either undersampling of majority class or oversampling of minority class should be done. The paper chose to perform oversampling by SMOTE technique. SMOTE refers to Synthetic Minority Oversampling technique used to perform oversampling of minority class samples [14]. SMOTE selects examples in the feature space, which are close to each other, draws a line between the examples, and then creates a new sample at a location along the line. The goal of oversampling is to mitigate the effects of class imbalance, where the minority class has significantly fewer samples than the majority class. By increasing the number of samples in the minority class, the imbalance is reduced, making the dataset more balanced overall. SMOTE has several parameters that can be adjusted to control the behavior of the algorithm. These parameters are: (sampling_strategy= auto): resample all classes but not the majority class

(k=5): the number of nearest neighbors to consider when generating synthetic examples. A higher value of k will lead to more synthetic examples being generated, but it may also increase the risk of overfitting. A lower value of k will lead to fewer synthetic examples being generated, but it may also reduce the effectiveness of SMOTE in balancing the dataset.

(random_state = None): This parameter controls the random seed used by SMOTE. Setting a specific random seed will ensure that the same synthetic examples are generated each time the algorithm is run.

(out_spread = 0.5): This parameter controls the amount of spread to use for interpolation.

(ratio of synthetic to real examples= 2:1) : This ratio determines how many synthetic examples are created for each real example.

6. Model Training

The dataset was divided into features and target in 70:30 and features were scaled using scikit-learn standard scale. The models were divided into two categories:

- **Baseline models** (Logistic Regression, Gradient Boosting)
- **Boosting Based models** (Extreme Gradient Boosting, LightGBM, CatBoost)

6.1 Baseline Model

Logistic Regression (LR) and Gradient Boosting Classifier (GB) were considered as two baseline models in this paper due to their simplicity.

6.1.1 Logistic Regression

Logistic Regression (LR) is well suited for this task as it is simple, interpretative and computationally efficient. It works well with linearly separable data and provides probabilistic predictions useful in estimating likelihood of default for each loan application. Assume, a set of observation is given by $(x_{11}, x_{12}, ..., x_{1k}, y_1), ...(x_{n1}, x_{n2}, ..., x_{nk}, y_n)$, where n is the number of observations, x_{ik}(i = 1,..., n) are the predictor variables, y_i(i=1,..., n) is the response variable. Then the probability for classifying bad and good samples is given by:

$$p(Y = 1 \mid x) = \frac{\exp(b_0 + b_1x_1 + b_2x_2 + \cdots + b_kx_k)}{1 + \exp(b_0 + b_1x_1 + b_2x_2 + \cdots + b_kx_k)} \quad (1)$$

$$p(Y = 0 \mid x) = \frac{1}{1 + \exp(b_0 + b_1x_1 + b_2x_2 + \cdots + b_kx_k)} \quad (2)$$

where b_i (i = 0, ..., k) are the model coefficients.
Denote $p = \frac{\exp(z)}{1+\exp(z)}$, where z = $b_0 + b_1x_1 + b_2x_2 + ... + b_kx_k$
and we have: $\ln\left(\frac{p}{1-p}\right) = z$

Odds ratio, or the ratio of the chance of events happening compared to the probability of them not happening, is the name for the function of transformation on the left side of the formula known as the logit function. Logistic regression thus converts a number of real number values to the interval [0,1] and calculates the likelihood that an event will occur. When a crucial number (such as 0.5) is established, it can determine whether or not an event happens [5].

6.1.2 Gradient Boosting

Gradient Boosting (GB) is an ensemble boosting technique that combines the predictions of several weak learners, often decision trees, to create a powerful predictive model [5]. Initialise a group of decision trees first. It works by calculating the ensemble's initial predictions by averaging or adding the predictions of each weak learner separately. Determining the residuals, also known as errors, between the target value in the training data and the initial predictions. Instead of using the original goal values, fit a weak learner to the residuals. Finding a tree that can fix the mistakes created by the ensemble's present forecasts is the objective. This brand-new tree is known as the "base learner." Predictions made by the freshly trained base learner are added to or subtracted from the ensemble's predictions to update them. The residuals are reduced as a result of these updates, thus bringing the ensemble's forecasts closer to the actual target values. Repeat until a stopping requirement is satisfied or for a predetermined amount of iterations. A new base learner gets trained on the most recent residuals at each iteration, while ensemble's predictions get revised as a result. The total of all of the initial predictions plus the predictions that were given by every base learner at each iteration constitutes the overall prediction for the Gradient Boosting ensemble. Gradient Boosting is suitable for loan default prediction as it can handle non-linear relationships, handle missing values and is robust to outliers. Hyperparameter tuning is done by max-depth=[1,2,3,4,5], n_estimators=[100,200,300,400,500], max-leaf_nodes=[2,5,10,20,30,40,50] by using randomized searchCV.

6.2 Boosting Based Models

Boosting is a machine learning ensemble technique used to improve the predictive performance of models. In boosting, multiple weak learners (simple models that perform slightly better than random chance) are combined to create a strong learner. The idea is to iteratively train new models that focus on instances that previous models have misclassified. Boosting works by sequentially fitting new models to the residuals, or errors, of the previous models. Each new model pays more attention to the instances that were misclassified by the previous models. This process continues until a predefined number of weak learners have been created, or until a threshold in performance is achieved. Boosting models can be highly effective for loan default prediction due to their ability to handle complex relationships in the data and to minimize prediction errors [15]. Extreme Gradient Boosting (XGB), LightGBM (LGBM) and CatBoost (CB) are examples of boosting based models.

6.2.1 Extreme Gradient Boosting

Extreme Gradient Boosting (XGB) is a gradient boosting algorithm, which means it builds a sequence of decision trees to make predictions[9]. Each tree is trained to correct the errors of the previous trees, and the final prediction is a weighted sum of the predictions from all of the trees[2]. A unique distributed weighted quantile sketch technique was introduced with a theoretical guarantee that can handle weighted data. The fundamental concept was to find splits to provide a data structure that may be used for merge and prune operations which retained a particular degree of accuracy [3]. The logistic loss function is used in it which measures the difference between the predicted probability of default and the actual default indicator. A lower loss function value indicates that the model is making more accurate predictions. Regularization is a technique used to prevent overfitting, which is when a model learns the training data too well and is unable to generalize to new data.

$$L(y, f(x)) = -y\log(f(x)) - (1-y)\log(1-f(x))$$

where L represents the loss function y denotes the actual default indicator (0 for no default, 1 for default) f(x) symbolizes the predicted loan default probability. The gradient indicates the direction in which the model should adjust its predictions to minimize the loss function. It is computed with respect to the model parameters, which in the case of XGBoost, are the split values at each node of the decision trees.

$$g_i = \frac{\partial L}{\partial f(x_i)} = \frac{-y\_i}{f(x_i)} + \frac{(1-y_i)}{(1-f(x_i))}$$

where: g_i represents the gradient for the i_{th} instance L denotes the loss function y_{n1} symbolizes the actual default indicator for the i^{th} instance $f(x_i)$ represents the predicted loan default probability for the i^{th} instance. The core of the gradient boosting algorithm is the gradient descent optimization technique. Gradient descent works by iteratively updating the model parameters in the direction of the negative gradient of the loss function. This means that the model is constantly being adjusted to minimize the loss, which is the difference between the predicted values and the actual values. XGBoost's ability to handle complex relationships, prevent overfitting, provide feature importance, and be robust to outliers makes it a valuable tool for loan default prediction. The used parameters are max_depth=5, alpha=5, learning rate=1, n_estimators=1000.

6.2.2 LightGBM

Microsoft introduced the LightGBM (Light Gradient Boosting Machine) method in 2017 in response to the XGBoost algorithm's high computational complexity plus lengthy execution times when dealing with large amounts of data [7]. To increase accuracy and decrease complexity, this method combines a number of fundamental algorithms. It is highly effective for loan default prediction due to speed, accuracy, ability to handle large datasets and ability to learn complex relationships between features and the target variable. It is significantly faster than other gradient boosting algorithms, such as XGBoost due to its use of a novel feature importance calculation method and a more efficient gradient-based one-sided sampling (GOSS) algorithm[11]. GOSS preserves the data with big gradients and filter out the samples with tiny gradients, lowering computing costs and information gain. Through the use of a histogram-based decision tree technique, traversal samples are converted into traversal histograms by LightGBM. By using the histogram difference, it lowers computational complexity as well as memory use [16].

LightGBM uses logistic loss function, L1 and L2 regularization technique, histogram based tree learning to build its decision trees and Early stopping is used to prevent overfitting.

The depth parameter, "tree depth," is defined and used to regulate the tree's level of complexity. The ideal quantity of tree depth guarantees the optimal performance of the base-learner and captures the characteristics of the training sets [17]. Parameters used are: learning rate= 0.1, num_leaves= 31, min_child_samples=5, min_child_weight= 0.001, min_split_gain=0.0, n_estimators=100, reg_alpha=0.0.

6.2.3 CatBoost

CatBoost (CB) is a gradient boosting algorithm. It is an open-source gradient boosting library that uses decision trees as its base learners [12]. The application of ordered boosting, a permutation-driven substitute for the traditional approach, and a novel technique for handling categorical characteristics are two significant algorithmic innovations included in CatBoost. Both methods were developed to combat a prediction shift brought on by a particular type of target leakage that exists in all gradient boosting algorithm implementations that are in use today [18]. CatBoost is known for its speed, accuracy, and ability to handle large datasets. In this paper, number of iterations= 100,learning rate= 1, loss function= cross entropy was used. Its unique features include:

- Ordered target encoding: CatBoost automatically encodes categorical features based on their order of importance, which can improve the performance of the model.
- Symmetric weighted quantile sketch: CatBoost uses a data structure called a symmetric weighted quantile sketch to efficiently handle missing values.
- Gradient-based One-sided Sampling (GOSS): CatBoost selects data instances based on their gradients, prioritizing instances with larger gradients, which are more informative for learning.
- Exclusive Feature Bundling (EFB): CatBoost groups mutually exclusive features together, reducing overfitting and improving model interpretability.
- Histogram-based Tree Learning: CatBoost utilizes histograms to represent features, enabling efficient tree learning and reducing computation time.
- Early Stopping and Leaf-wise Tree Construction: Early stopping prevents overfitting by stopping tree growth when a stopping criterion is met. Leaf-wise tree construction builds trees one leaf at a time, further improving efficiency.

7. Parameters used to Build Models

Some parameters of machine learning algorithms are:

- num_leaves: This parameter controls the complexity of the tree model. Its higher value leads to deeper trees, which can capture more complex relationships in the data and if the value is too high, it can lead to overfitting.
- max_depth: This parameter limits the maximum depth of the decision trees. A higher value of max_depth allows the trees to capture more complex patterns but can lead to overfitting.
- learning_rate: The learning rate determines the step size taken during gradient descent optimization. A higher learning rate speeds up training but increases the risk of overfitting. Conversely, a lower learning rate slows down training but improves generalization ability.
- min_child_samples: This parameter specifies the minimum number of samples required in a node to split. A higher value of min_child_samples prevents overfitting by avoiding splitting nodes with too few data points.
- n_estimators: This parameter specifies the number of boosting rounds, representing the number of decision trees to be built in the ensemble model. A larger number of boosting rounds generally improves accuracy but increases training time. It is a crucial hyperparameter that controls the complexity of the model and its ability to capture complex patterns in the data.
- num_of_boost_round: This parameter specifies the number of boosting rounds, representing the number of decision trees to be built in the ensemble model. A larger number of boosting rounds generally improves accuracy but increases training time.
- bagging_fraction: This parameter determines the proportion of training data used for each tree split. A lower value of bagging_fraction reduces the correlation between trees and can help prevent overfitting.
- lambda_l1: This parameter controls the L1 regularization, which penalizes large model coefficients. Higher values of lambda_l1 lead to sparser models with fewer features.

8. Model Evaluation

The metrics used to evaluate performance of models are:

- **Accuracy:** Accuracy is the fraction of all predictions that are correct. It is calculated as follows:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Predictions}}$$

  Accuracy is a simple metric, but it can be misleading if the dataset is imbalanced, i.e., if there are many more negative examples than positive examples.
- **Precision:** Precision is the fraction of predicted defaults that are actually defaults. It is calculated as follows:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

  A high precision value indicates that the model is good at avoiding false positives, i.e., predicting a default when there is none.
- **Recall:** Recall is the fraction of actual defaults that are predicted correctly. It is calculated as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

  A high recall value indicates that the model is good at identifying true defaults.

- **F1-score:** The F1-score is a weighted average of precision and recall. It is calculated as follows:

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1-score is a useful metric for evaluating models when both precision and recall are important.

It is important to assess the model's performance on both the training and validation/test datasets to ensure it generalizes well to unseen data. Cross-validation is a technique for evaluating the performance of a model on unseen data [19]. It involves splitting the dataset into multiple folds and training the model on each fold separately. The performance of the model is then evaluated on the remaining folds. This helps to reduce the risk of overfitting, which is when a model learns the training data too well and is unable to generalize to new data. The paper employed 5 kfold and 10 kfold cross validation.

9. Experimental Results

9.1 Baseline Models

The baseline models were trained on loan default dataset for classification purpose of predicting whether there will be loan default or not. The result is shown in Table 3.

Table 3: Performance Metrics of Baseline Models

Model	Validation	Accuracy	Precision	Recall	F1-Score
LR	70:30 split	85	87	95	91
	5fold	84.6	72.5	47.15	57.13
	10fold	84.5	72.46	47.04	57.02
GB	70:30 split	92	94	69	79
	5fold	92.47	94.05	69.83	80.15
	10fold	92.54	94.18	70.08	80.34

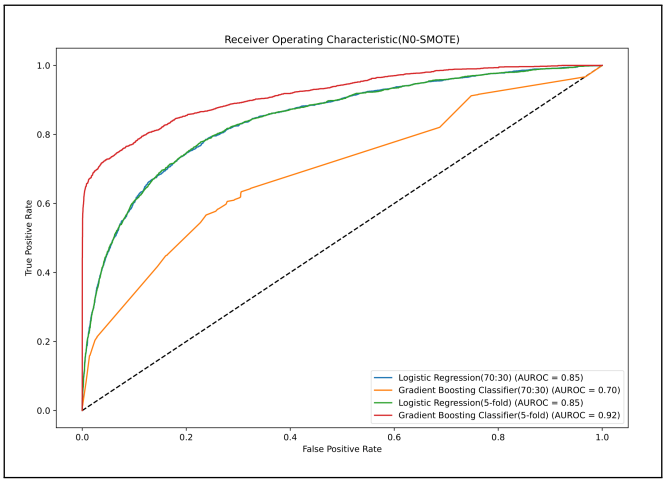

Figure 6: ROC Curve of Baseline Models

Area Under Curve (AUC) values are shown in Table 4.

Table 4: AUC score of Baseline Models

Model	Cross Validation	AUC Score
LR	70:30 split	85
	5fold	85
	10fold	85.10
GB	70:30 split	70
	5fold	92
	10fold	92.20

9.2 Boosting Based Models

The boosting based models were trained on loan default dataset for classification purpose of predicting whether there will be loan default or not. The result is shown in Table 5.

Table 5: Performance Metrics of Boosting Based Models

Model	Validation	Accuracy	Precision	Recall	F1-Score
XGB	70:30 split	93	91	74	82
	5fold	93.297	93.01	74.86	82.94
	10fold	93.207	92.75	74.67	82.72
LGB	70:30 split	94	97	72	82
	5fold	93.66	97.45	72.82	83.35
	10fold	93.68	97.62	72.78	83.37
CB	70:30 split	93	94	73	82
	5fold	93.31	93.77	74.24	82.86
	10fold	93.25	93.64	74.09	82.70

According to Table 5, the accuracy has increased for boosting models than baseline models. Also, precision score has increased but recall score has decreased comparatively. As f1-score is harmonic mean of precision and recall, it has also decreased. This is not due to model incompetency but due to imbalance nature of data. Loan eligible dataset is unbalanced in nature because the cases of loan defaults were less than non-default cases which is true in reality too. Cases of default will be few however it may have huge impact on overall banking system. Therefore, SMOTE oversampling technique was used to solve class imbalance problem and oversample minority class. Balanced dataset was created and address the class imbalance issue in the dataset, which could have made it difficult for the models to learn from the minority class (i.e., the default class).

Table 6: AUC score of Boosting Based Models

Model	Cross Validation	AUC Score
XGB	70:30 split	94.00
	5fold	92
	10fold	92.41
LGB	70:30 split	94.00
	5fold	93.00
	10fold	93.24
CB	70:30 split	94
	5fold	92
	10fold	92.32

Area Under Curve (AUC) values are shown in Table 6 which shows LightBGM has highest value 94. The ROC curve of all the models in Figure 7 also shows the same.

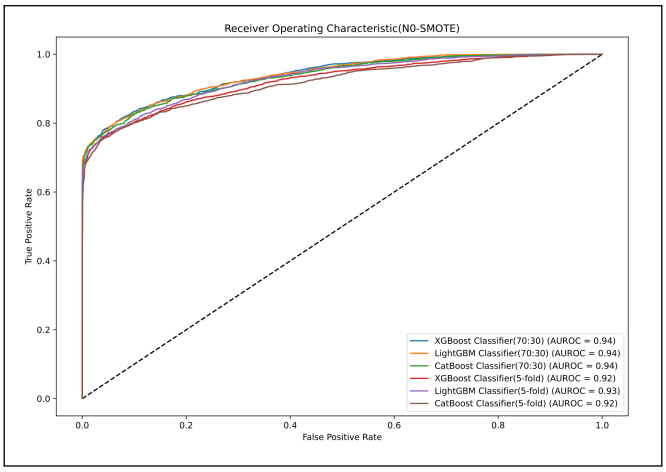

Figure 7: ROC Curve of Boosting Based Models

9.3 Boosting with Oversampling

SMOTE was applied to generate balanced dataset of minority class of default and majority class of non-default cases [20]. This balanced dataset was used in Boosting based models and the models were train to classify default and non-default cases. The effect of oversampling upon Boosting based models is shown below.

Table 7: Performance Metrics of Boosting Based Models using SMOTE

Model	Validation	Accuracy	Precision	Recall	F1-Score
XGB	70:30 split	94	96	91	93
	5fold	95.60	98.07	93.03	95.48
	10fold	95.60	98	93.08	95.48
LGB	70:30 split	96	99	92	96
	5fold	95.74	99.41	92.02	95.57
	10fold	95.76	99.45	92.02	95.59
CB	70:30 split	96	99	93	96
	5fold	95.51	98.43	92.50	95.37
	10fold	95.5	98.25	92.63	95.35

Table 8: AUC score of Boosting Based Models using SMOTE

Model	Cross Validation	AUC Score
XGB	70:30 split	98.00
	5fold	98
	10fold	98.10
LGB	70:30 split	99
	5fold	98
	10fold	98.32
CB	70:30 split	98
	5fold	98
	10fold	98.20

The results after oversampling with SMOTE as in Table 7 showed a significant improvement in the performance of all of the models. This is especially evident in the recall metric, which increases for all models by maximum of 17 to 20%. There is improvement in precision by 5 to 6% in overall models. F1-Score of models increased by 11 to 14%. There is consistency between accuracy, precision, recall and f1-score of all models after oversampling. The values are converging towards 100% unlike before using SMOTE. The results with SMOTE oversampling are better than the results without SMOTE oversampling. This indicates that the improvement in performance with SMOTE oversampling is not due to overfitting. Overall, the results show that SMOTE oversampling is an effective technique for improving the performance of machine learning models for loan default prediction, especially in the case of imbalanced datasets. The AUC value of LightGBM was 99 which was the highest showing it as best performer as shown in Table 8. The ROC curve in Figure 8 shows the same.

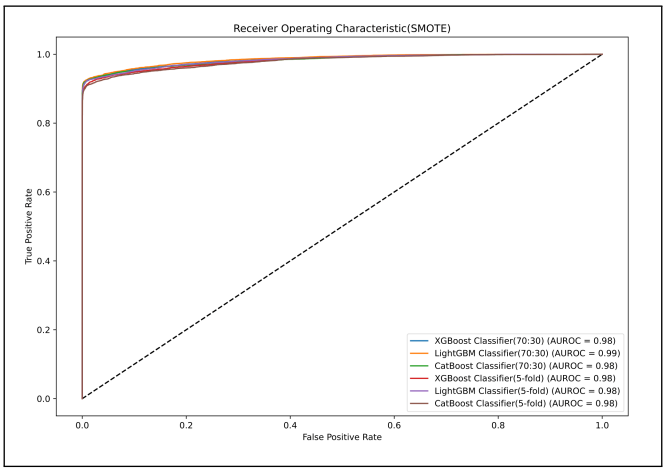

Figure 8: ROC Curve of Boosting Models with SMOTE

10. Conclusion

In this paper, machine learning models were trained on loan default dataset to predict loan default probability. Exploratory Data Analysis was done to know more about the nature of loan default cases and relation of loan default with other attributes like type of loan taken. Due to imbalance nature of data, SMOTE technique was used for oversampling minority class. The results before and after oversampling on boosting based models was computed. It was validated using hold out method on 70:30 split, 5 k-fold and 10 k-fold cross validation. The Boosting methods showed better performance than the baseline models. This performance was again enhanced by oversampling. Hence, by leveraging boosting models for loan default prediction, financial institutions can make more informed decisions about lending, reduce the risk of defaults, and optimize their loan portfolio management strategies.

References

[1] Hossam Meshref. Predicting loan approval of bank direct marketing data using ensemble machine learning algorithms. *International Journal of Circuits, Systems and Signal Processing*, 14:914–922, 2020.

[2] A.K. Sharma, LH. Li, and R. Ahmad. Default risk prediction using random forest and xgboosting classifier. *International Conference on Security and Information Technologies with AI, Internet Computing and Big-data Applications, 2021.*

[3] Shiv S.J D ept, S. Sreenivasa Murthy, and Krishnaprasad Challuru. Credit risk analysis using machine learning

techniques. *2018 Fourteenth International Conference on Information Processing (ICINPRO)*, pages 1–5, 2018.

[4] Yu Li. Credit risk prediction based on machine learning methods. *2019 14th International Conference on Computer Science & Education (ICCSE)*, pages 1011–1013, 2019.

[5] J. C. N. Nguemaleu U. E. Orji, C. H. Ugwuishiwu and P. N. Ugwuanyi. Machine learning models for predicting bank loan eligibility. *IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON), Lagos, Nigeria, 2022, pp. 1-5, doi: 10.1109/NIGERCON54645.2022.9803172*, 2022.

[6] Mohammad Ahmad Sheikh, Amit Kumar Goel, and Tapas Kumar. An approach for prediction of loan approval using machine learning algorithm. *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pages 490–494, 2020.

[7] Xingzhe Dong. Loan default prediction based on machine learning (lightgbm model). *BCP Business & Management*, 25:457–468, 08 2022.

[8] Yuelin Wang, Yihan Zhang, Yan Lu, and Xinran Yu. A comparative assessment of credit risk model based on machine learning——a case study of bank loan data. *Procedia Computer Science*, 174:141–149, 2020. Elsevier.

[9] Yuran Zhou. Loan default prediction based on machine learning methods. *International Conference on Big Data Economy and Information Management, BDEIM 2022, December 2-3, 2022, Zhengzhou, China*, 6 2023. European Union Digital Library.

[10] https://www.kaggle.com/datasets/vikasukani/loan-eligible-dataset. Accessed on: 04/09/2023.

[11] Qiliang Zhu, Wenhao Ding, Mingsen Xiang, Mengzhen Hu, and Ning Zhang. Loan default prediction based on convolutional neural network and lightgbm. *International Journal of Data Warehousing and Mining*, 19:1–16, 01 2023.

[12] Sujoy Barua, Divya Gavandi, Pooja Sangle, Leena Shinde, and Jyoti Ramteke. Swindle: Predicting the probability of loan defaults using catboost algorithm. In *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1710–1715, 2021.

[13] https://www.kaggle.com/datasets/laotse/credit-risk-dataset. Accessed on: 04/09/2023.

[14] Ahmed Saad Hussein, Tianrui Li, Chubato Wondaferaw Yohannese, and Kamal Bashir. A-smote: A new preprocessing approach for highly imbalanced datasets by improving smote. *International Journal of Computational Intelligence Systems*, 12:1412–1422, 2019.

[15] https://www.geeksforgeeks.org/bagging-vsboosting-in-machine-learning.

[16] Sumin Fan. Design and implementation of a personal loan default prediction platform based on lightgbm model. In *2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA)*, pages 1232–1236, 2023.

[17] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[18] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[19] https://www.geeksforgeeks.org/cross-validation-machine-learning/. Accessed on: 04/09/2023.

[20] https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/. Accessed on: 04/09/2023.