CHURN ANALYSIS OF BANKING CUSTOMER

BITS F464 - Machine Learning

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

HYDERABAD CAMPUS

Under Supervision of

Prof. Paresh Saxena

Submitted by:

Name	ID	Email
Arihant Rai	2021B3A72507H	f20212507@hyderabad.bits-pilani.ac.in
Shubham Birla	2021B3A72965H	f20212965@hyderabad.bits-pilani.ac.in
Kinjal Vardia	2021B3A72579H	f20212579@hyderabad.bits-pilani.ac.in
Deepanshu Garg	2021B3A72758H	f20212758@hyderabad.bits-pilani.ac.in
Utkarsh Singhal	2022A7PS1334H	f20221334@hyderabad.bits-pilani.ac.in

1 Introduction

Banking Customer Churn Analysis is an increasingly vital topic in the financial sector, driven by the need for banks to retain their customer base in a highly competitive market. Customer churn, or attrition, refers to the phenomenon where clients terminate their relationship with a bank, either by closing accounts or discontinuing the use of banking services. This loss can have significant financial and reputational impacts, making customer retention a top priority for modern banks

With the advancement of data science, machine learning (ML) techniques have become powerful tools for predicting which customers are at risk of leaving. By analyzing vast and diverse customer data—including demographics, account activity, product usage, and behavioral patterns—ML models can identify subtle signals and trends that precede churn events. These predictive insights allow banks to intervene proactively, offering targeted incentives or improving services to retain valuable customers.

Recent research demonstrates the effectiveness of various ML algorithms such as logistic regression, support vector machines, random forests, and gradient boosting (XGBoost) in churn prediction tasks. These models are evaluated using metrics like accuracy, sensitivity, specificity, and area under the curve (AUC) to determine their predictive power. Additionally, data preprocessing, feature engineering, and techniques to address class imbalance (such as SMOTE) are essential steps to ensure robust and reliable predictions

2 Paper Implementation and Results

2.1 Abstract

The paper "Investigating customer churn in banking: a machine learning approach and visualization app for data science and management" was written by Pahul Preet Singh, Fahim Islam Anik, Rahul Senapati, Arnav Sinha, Nazmus Sakib, and Eklas Hossain

This paper implements and critically evaluates the machine learning models presented in the original study by Singh et al. (2024), which focuses on predicting bank customer churn using logistic regression, SVM, random forest, and XGBoost. We replicate the study using the same Kaggle dataset, reapply preprocessing steps, handle class imbalance using SMOTE, and reproduce evaluation metrics. Additionally, we assess the effectiveness of the models in real-world deployment and extend the original work with our own insights into model tuning and visualization.

Aspect	**Description**
Dataset	- 10,000 records with 13 meaningful customer attributes from Kaggle's bank churn dataset.
Target Variable	- `Exited` (binary classification: 0 = retained, 1 = churned).
Initial Preprocessing	- Removed non-predictive columns: `RowNumber`, `CustomerID`, `Surname`. - Encoded categorical variables (`Geography`, `Gender`) using label encoding/one-hot encoding.
Feature Engineering	- `TenureByAge` = Tenure / Age - `BalanceSalaryRatio` = Balance / EstimatedSalary - `CreditScoreGivenAge` = CreditScore / Age
Models Implemented	- Logistic Regression, Support Vector Machine (SVM), Random Forest, and XGBoost.
Model Tuning	- Hyperparameter tuning via `GridSearchCV` with stratified 5-fold cross-validation.
Handling Class Imbalance	- Used SMOTE (Synthetic Minority Oversampling Technique) to address imbalance in the `Exited` variable.
Evaluation Metrics	- Accuracy - Sensitivity (Recall) - Specificity - Area Under Curve (AUC) - F1 Score

2.2 Paper Implemntation without SMOTE'd data

Model	Accuracy
Primal Logistic Regression	0.8156
Polynomial Logistic Regression (Degree = 2)	0.8554
SVM (RBF Kernel)	0.8519
SVM (Polynomial Kernel)	0.8545
Random Forest Classifier	0.8629
XGBoost Classifier	0.8628

Table 2: Model Accuracy Comparison

In this implementation, we evaluated six different machine learning models to predict customer churn based on accuracy, which reflects the proportion of correct predictions made by each model. The Primal Logistic Regression model served as the baseline with an accuracy of 0.8156, while Polynomial Logistic Regression (Degree = 2) yielded the highest accuracy of 0.8554 among the models. The SVM (RBF Kernel) achieved an accuracy of 0.8519, followed closely by SVM (Polynomial Kernel) at 0.8545. Both SVM models demonstrated strong performance, though they were slightly outperformed by the logistic regression models.

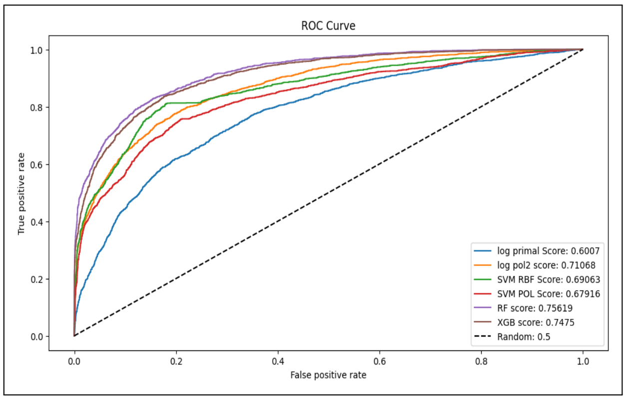

Figure 1: ROC Curve

The Random Forest Classifier and XGBoost Classifier were the top performers, both reaching an accuracy of approximately 0.8629 and 0.8628, respectively. These models proved to be the most effective for predicting customer churn, showcasing the power of ensemble methods in handling complex, high-dimensional datasets. The results indicate that Random Forest and XGBoost are the best choices for churn prediction in this dataset, while models like Polynomial Logistic Regression also offer competitive performance in terms of accuracy and interpretability.

2.3 Paper Implemntation with SMOTE'd data

SMOTE (Synthetic Minority Oversampling Technique) To address the class imbalance in my dataset, where 80% of the instances belonged to retained and only 20% to exited, I applied the SMOTE (Synthetic Minority Over-sampling Technique) algorithm. SMOTE works by generating synthetic samples for the minority class through interpolation between existing minority instances, effectively increasing its representation. This helped in balancing the dataset to a 50-50 distribution between the two classes. As a result, the model's performance improved by reducing its bias toward the majority class and enabling better generalization.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.91	0.88	0.89	1607
1	0.55	0.63	0.59	389
Accuracy		0.83		1996
Macro avg	0.73	0.75	0.74	1996
Weighted avg	0.84	0.83	0.83	1996

Table 3: Random Forest Classification Report on SMOTE'd Data

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.91	0.85	0.89	1607
1	0.52	0.66	0.58	389
Accuracy		0.83		1996
Macro avg	0.71	0.75	0.74	1996
Weighted avg	0.83	0.81	0.82	1996

Table 4: XGBoost Classification Report on SMOTE'd Data

3 Applied Improvements in the paper

3.1 Neural Network

We normalize all input features with StandardScaler to center and scale the data, which accelerates convergence. Our Keras Sequential model comprises three dense layers of 128, 64, and 32 units, each followed by BatchNormalization and Dropout (40%, 40%, and 30%) to stabilize learning and reduce overfitting. We compile with the Adam optimizer at a 1×10^{3} learning rate, using binary cross-entropy loss and AUC as the primary metric. Training runs for up to 100 epochs (batch size 32) on the resampled training set, with EarlyStopping (patience=10) and ReduceLROnPlateau (factor=0.5, patience=5) to prevent overtraining and adjust the learning rate. Finally, we evaluate on the held-out test set by computing predicted probabilities, applying a 0.5 threshold for class labels, and reporting both a detailed classification report and the test AUC.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.9017	0.7760	0.8341	1607
1	0.4127	0.6504	0.5050	389
Accuracy		0.7515		1996
Macro avg	0.6572	0.7132	0.6696	1996
Weighted avg	0.8064	0.7515	0.7700	1996

Table 5: Neural Net Classification Report

3.2 Neural Network - Improved

This updated network increases model capacity with four hidden layers (256→128→64→32 units) and applies L2 regularization on each to curb overfitting. It also incorporates class-weighting to emphasize the rare churn cases and uses a lower learning rate (5×10) for more stable convergence. Training now optimizes recall via EarlyStopping and ReduceLROnPlateau, ensuring we prioritize catching churners. After fitting, we compute an optimal decision threshold from the precision–recall curve to maximize F score rather than assuming 0.5. Finally, we visualize training metrics (loss, AUC, recall) and estimate feature importance through input–output gradients to interpret which variables drive churn predictions.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.8846	0.8780	0.8813	1607
1	0.5112	0.5270	0.5190	389
Accuracy	0.8096			1996
Macro avg	0.6979	0.7025	0.7002	1996
Weighted avg	0.8119	0.8096	0.8107	1996

Table 6: Improved Neural Network Classification Report

3.3 LightGBM Model

LightGBM is a fast, efficient gradient boosting framework optimized for large datasets, using histogram-based methods for speed and lower memory usage. It excels in handling categorical features and supports GPU acceleration, making it ideal for structured ML tasks like classification and regression.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.90	0.91	0.91	1607
1	0.62	0.57	0.59	389
Accuracy	0.85			1996
Macro avg	0.76	0.74	0.75	1996
Weighted avg	0.84	0.85	0.85	1996

Table 7: LightGBM Classification Report

3.4 Optimization Using RandomizedSearchCV

To enhance model performance and reduce computation time, RandomizedSearchCV was employed for hyperparameter tuning instead of GridSearchCV. Unlike GridSearchCV, which exhaustively tries all parameter combinations, RandomizedSearchCV selects a fixed number of random combinations from the parameter space. This allows for a broader search over the hyperparameter space in less time, especially beneficial when dealing with large datasets or complex models.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.8871	0.9340	0.9100	1607
1	0.6513	0.5090	0.5714	389
Accuracy	0.8512			1996
Macro avg	0.7692	0.7215	0.7407	1996
Weighted avg	0.8412	0.8512	0.8440	1996

Table 8: Classification Report for Optimization Using RandomizedSearchCV

3.5 CATBOOST

To address the issue of class imbalance, the CatBoostClassifier was trained on a SMOTE-resampled dataset. CatBoost, a gradient boosting algorithm that efficiently handles categorical features and missing values, was configured with 500 iterations, a learning rate of 0.1, and a tree depth of 6. The model incorporated early stopping based on performance on a validation set to mitigate overfitting. Training was conducted on the synthetically balanced data produced by SMOTE, while evaluation was performed on the original test set to ensure unbiased assessment. Model performance was assessed using classification metrics and the ROC AUC score. The results indicated that integrating SMOTE with CatBoost led to improved predictive performance on the minority class, validating the effectiveness of this approach for imbalanced classification problems.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.9021	0.9179	0.9099	1607
1	0.6343	0.5887	0.6107	389
Accuracy	0.8537			1996
Macro avg	0.7682	0.7533	0.7603	1996

Table 9: CatBoost Classification Report

3.6 Bayesian Optimization with Optuna

To further enhance model performance, Bayesian optimization was implemented using the Optuna framework. Unlike grid or random search, Bayesian optimization builds a probabilistic model of the objective function and uses it to select hyperparameter combinations more intelligently. It balances exploration of new regions with exploitation of known good ones, often converging to optimal solutions faster and more efficiently.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.8819	0.9708	0.9242	1607
1	0.7930	0.4627	0.5844	389
Accuracy	0.8717			1996
Macro avg	0.8374	0.7167	0.7543	1996
Weighted avg	0.8645	0.8717	0.8580	1996

Table 10: Classification Report for Bayesian Optimization with Optuna

4 Conclusion

The study on banking customer churn analysis demonstrated the effectiveness of various machine learning models in predicting customer attrition. Among the evaluated models, Random Forest and XGBoost emerged as the top performers, achieving accuracies of approximately 0.8629 and 0.8628, respectively, on the original imbalanced dataset. The application of SMOTE significantly improved model performance by addressing class imbalance, enhancing the ability to generalize and predict minority class instances.

Further improvements were explored through advanced techniques such as neural networks, LightGBM, CatBoost, and hyperparameter optimization using RandomizedSearchCV and Bayesian optimization with Optuna. The neural network, with its layered architecture and regularization techniques, achieved competitive results, while LightGBM and CatBoost showcased their efficiency and robustness in handling structured data. Bayesian optimization further refined model performance, achieving an accuracy of 0.8717, highlighting the value of intelligent hyperparameter tuning.

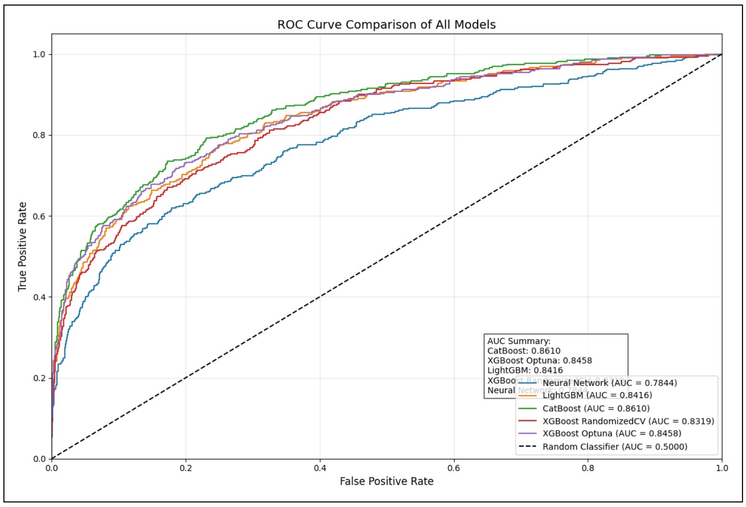

Figure 2: ROC Curve

Overall, the study underscores the importance of selecting appropriate models and techniques tailored to the dataset's characteristics. The combination of ensemble methods, resampling techniques, and optimization frameworks provides a comprehensive approach to churn prediction, enabling banks to proactively retain customers and mitigate financial losses. These insights contribute to the growing body of research on customer churn analysis, offering practical solutions for real-world applications in the banking sector.

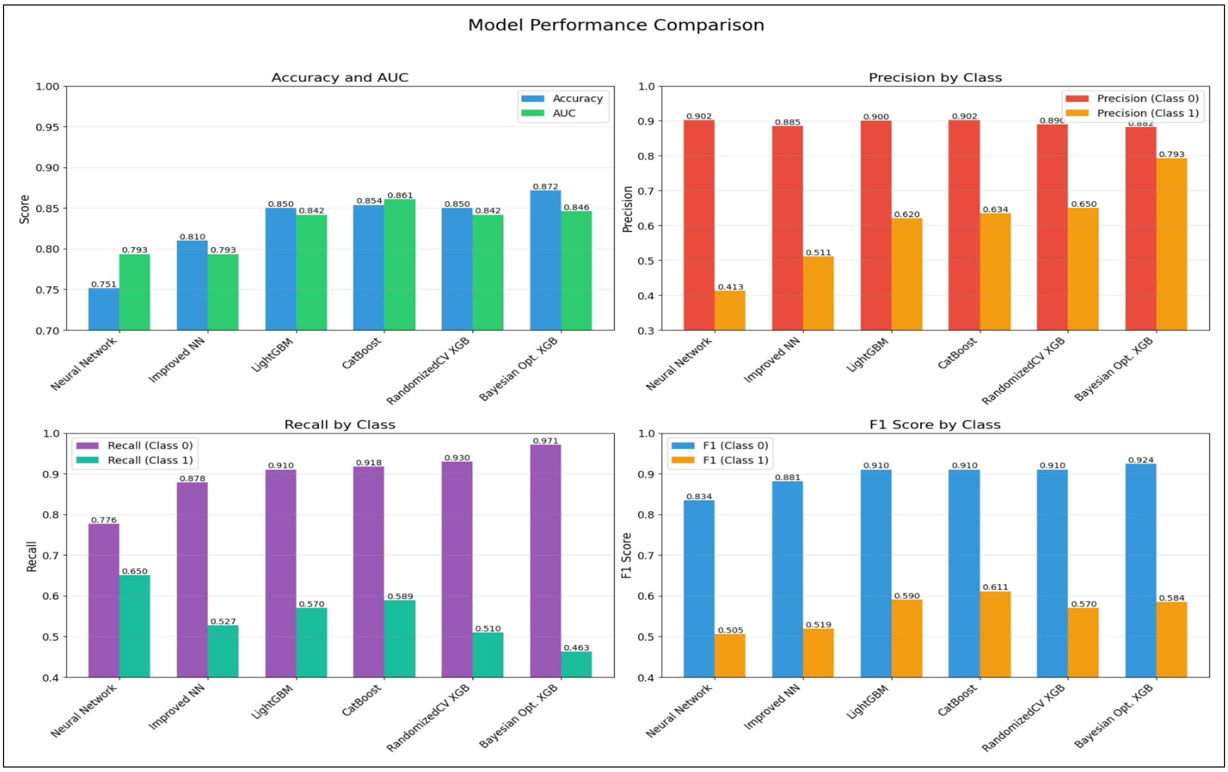

Figure 3: Model Performace Comparison

5 Contributions

Group Member	**Responsibilities**
Arihant	Responsible for making improvements, conducting experiments, and contributing to model analysis and fine-tuning.
Shubham	Worked on the paper implementation, particularly in coding algorithms and training models. Also responsible for writing the conclusion of the report.
Kinjal	Focused on making improvements and running experiments, including hyperparameter tuning, testing different models, and contributing to analysis.
Deepanshu	Was responsible for fine-tuning models and analyzing performance metrics to optimize results. He collaborated with the team on various experiments to improve model accuracy and efficiency.
Utkarsh	Led the paper implementation by setting up the framework, coding models, testing, and performing evaluations. Contributed to the documentation of results.

Table 11: Group Member Task Distribution in the Project