COMP 9130 - Mini Project 2: Regression Challenge

Group 8

Mini Project 1 Report

Binger Yu
A01003660
gyu42@my.bcit.ca

Yansong Jia
A01473470
yjia16@my.bcit.ca

Supervisor: *Dr. Michal Aibin*

School of Computing and Academic Studies
Master of Science in Applied Computing
British Columbia Institute of Technology
Burnaby, British Columbia, Canada

Date: January 28, 2026

Contents

1 Introduction

Customer churn is a critical challenge for subscription-based industries such as telecommunications. Retaining existing customers is generally more cost-effective than acquiring new ones, making churn prediction a valuable business task. This project aims to build a classification system that predicts whether a telecom customer will churn based on demographic characteristics, subscribed services, and billing information.

The target variable, *Churn*, is binary, indicating whether a customer leaves the company (Yes) or remains (No). Due to the imbalanced nature of the dataset, traditional accuracy is not sufficient for evaluation. Instead, this project emphasizes precision, recall, F1-score, and ROC-AUC to better capture model performance on minority-class churners.

2 Data Exploration

2.1 Dataset Overview

The Telco Customer Churn dataset, sourced from IBM Sample Data on Kaggle, contains 7,043 customer records with 20 input features. These features include demographic information, service usage, contract details, and billing-related attributes.

2.2 Target Distribution

The dataset is imbalanced, with approximately 73% non-churned customers and 27% churned customers. This imbalance motivates the use of specialized evaluation metrics and imbalance-handling techniques.

2.3 Data Quality

During exploration, the `TotalCharges` feature was found to be stored as a string and contained missing values. This column was converted to numeric format, and rows with missing values were removed. Correlation analysis and visual inspection revealed no extreme outliers requiring removal.

3 Methodology

3.1 Data Preprocessing

The following preprocessing steps were applied:

- Conversion of numeric columns stored as strings
- Handling of missing values
- One-hot encoding of categorical features
- Feature scaling for numerical variables
- 80/20 train-test split using stratification to preserve class distribution

Preprocessing was applied as part of the modeling pipeline to avoid data leakage and ensure reproducibility.

3.2 Models

Three classification models were trained and evaluated:

- Logistic Regression (baseline)
- Random Forest
- LightGBM

3.3 Imbalanced Data Handling

Given the imbalanced target distribution, multiple strategies were explored:

- Class weighting using `class_weight='balanced'`
- SMOTE oversampling applied **only to the training data**
- Threshold adjustment for probability-based predictions

Models trained with and without imbalance handling were compared to assess performance differences.

3.4 Hyperparameter Tuning

GridSearchCV was applied to the LightGBM model to optimize performance. At least three hyperparameters were explored, and the F1-score was used as the scoring metric to balance precision and recall.

4 Results

4.1 Model Performance

Table I summarizes the performance of all evaluated models.

TABLE I
Model Performance Comparison

Model	Precision	Recall	F1-score	ROC-AUC
Logistic Regression	0.49	0.79	0.61	0.84
Random Forest	0.63	0.49	0.55	0.82
LightGBM	0.52	0.77	0.62	0.83
LightGBM (Tuned)	**0.51**	**0.81**	**0.63**	**0.84**

Confusion matrices and ROC curves were generated for all models to further evaluate classification performance.

4.2 Confusion Matrix Analysis

Figure 1 shows the confusion matrix for the tuned LightGBM model on the test set. The model demonstrates strong recall for the churn class, indicating effective identification of customers likely to leave. From a business perspective, this is desirable because false negatives (missed churners) are more costly than false positives.

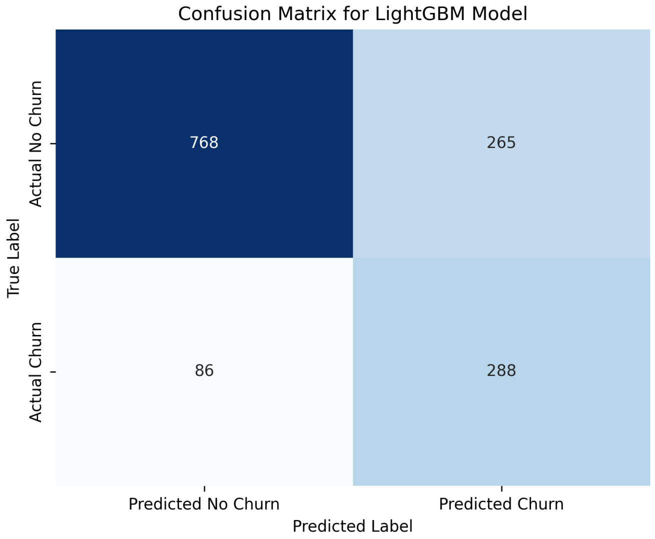

Fig. 1. Confusion matrix for the tuned LightGBM model on the test set.

4.3 ROC Curve Comparison

Figure 2 compares the ROC curves of all evaluated models. The tuned LightGBM model achieves the highest ROC-AUC, indicating superior ability to distinguish between churned and non-churned customers across different classification thresholds.

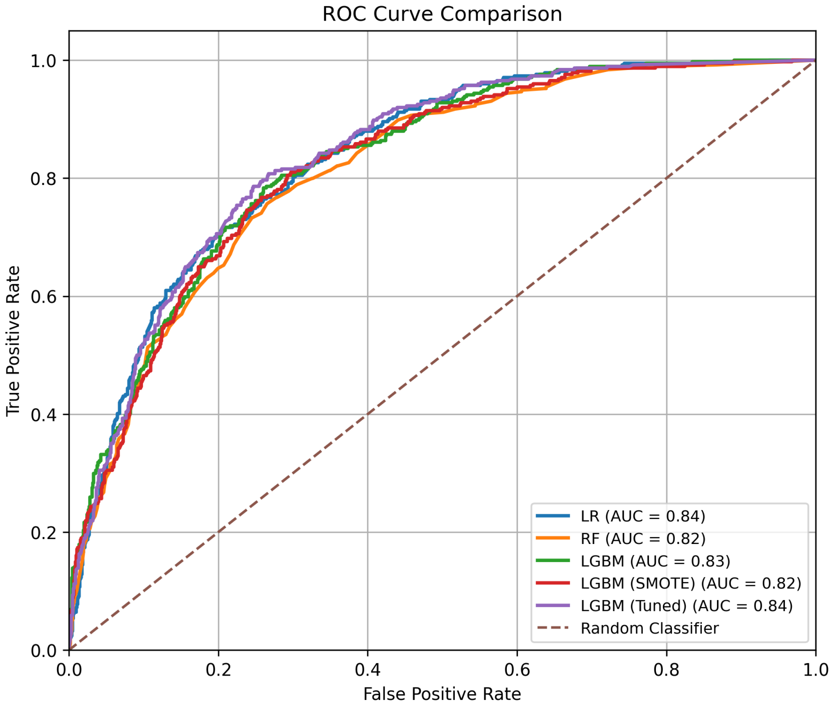

Fig. 2. ROC curve comparison across evaluated classifiers.

5 Discussion

The tuned LightGBM model achieved the best overall performance, particularly in terms of recall and ROC-AUC. From a business perspective, false negatives (missed churners) are more costly than false positives, as they represent lost customers without intervention. Therefore, models with higher recall for the churn class are preferred.

While Random Forest achieved higher precision, it suffered from low recall, making it less suitable for churn detection. Logistic Regression provided a strong baseline but was outperformed by LightGBM.

6 Conclusion

This project demonstrated the importance of appropriate evaluation metrics and imbalance-handling techniques when addressing real-world classification problems. By comparing multiple models and strategies, the tuned LightGBM classifier was selected as the final model due to its strong balance between recall and overall discriminative ability.

Future work may include feature importance analysis, cost-sensitive learning, or deployment of the model as a real-time churn prediction system.

7 Team Contributions

This project was completed collaboratively by both team members, with clear division of responsibilities throughout the development process.

Binger Yu was responsible for repository setup and project organization, including GitHub structure, environment configuration, and documentation. Binger conducted data preprocessing and exploratory data analysis in `01_exploration.ipynb`, addressed data quality issues, fixed implementation warnings, and contributed to report writing and final polishing.

Yansong Jia focused on model development and evaluation in `02_modeling.ipynb`. This included implementing multiple classifiers, handling class imbalance using class weighting, SMOTE, and threshold adjustment, performing hyperparameter tuning with GridSearchCV, generating evaluation metrics and visualizations, and interpreting model performance.