IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

TELECOM CUSTOMER CHURN PREDICTION AND RETENTION SYSTEM USING MACHINE LEARNING

Ranjith Kumar K R[1], Nishan J N[2], Prem Kumar BR[3], Roshan R[4], B. C. Hemapriya[5]

Student, Dept. of CSE, East West College of Engineering, Bengaluru, India[1,2,3,4]

Project Guide, Dept. of CSE, East West College of Engineering, Bengaluru, India[5]

Abstract: Customer churn, the loss of subscribers who stop using a service, reduces revenue and increases acquisition costs in the competitive telecom industry. Retention teams have limited capacity, so identifying customers at elevated risk can help them prioritize human review. This paper presents a machine-learning framework for telecom churn prediction and retention support. It uses customer attributes such as tenure, contract type, monthly charges, payment method, and subscribed services to estimate churn probability. The workflow validates and prepares data, separates training, validation, and test records, applies categorical encoding and numerical imputation within leakage-safe pipelines, compares Logistic Regression, Decision Tree, Random Forest, and XGBoost, and selects a classification threshold on validation data. Precision, recall, F1-score, ROC-AUC, PR-AUC, and confusion-matrix counts are reported so that performance on the minority churn class remains visible. The measured experiment uses the public IBM Telco Customer Churn sample (7,043 records) with a fixed stratified 60/20/20 split and seed 42. XGBoost was selected by validation F1 at threshold 0.358; on the held-out test portion it achieved ROC-AUC 0.832, precision 0.554, recall 0.682, and F1-score 0.612. This is a benchmark evaluation, not evidence of a deployed dashboard or of retention actions preventing churn. External and time-based validation remain necessary before operational use.

Keywords: Customer Churn, Telecom, Machine Learning, Classification, Random Forest, XGBoost, SMOTE, Class Imbalance, Retention Analytics, Decision Support.

I. INTRODUCTION

Telecommunication services such as mobile, broadband, and voice have become essential to individuals and organizations. Because customers can compare plans, prices, and service quality across providers and can switch with little effort, retaining existing customers is a central business activity. Churn may be voluntary, when a customer chooses a competitor, or involuntary, when a service is disconnected for reasons such as non-payment. Acquiring a new customer typically costs more than retaining an existing one, so even a modest reduction in churn can affect revenue noticeably.

Traditional retention practices rely on complaints, manual review, or simple rules. These methods identify some dissatisfied customers but may miss patterns that are spread across many attributes. Supervised machine learning can examine many variables together and estimate which customers resemble earlier customers who left. However, a churn model is useful only when it is evaluated correctly: churn datasets are usually imbalanced, so accuracy alone can be misleading, and common mistakes such as resampling before the train-test split can produce inflated results [15].

Churn Guard addresses this requirement through a pipeline-based workflow. Customer records are validated and prepared with transformations fitted only on training data, four candidate classifiers are compared under the same protocol, and the selected model outputs a churn probability that is mapped to a documented risk band. A dashboard presents these estimates with suggested review categories, and the final retention decision remains with the business team.

This work contributes a reproducible benchmark workflow that connects churn prediction with reviewable retention support. It does not claim live deployment, causal explanation of churn, or that a retention action will prevent a customer from leaving.

II. LITERATURE SURVEY

Churn prediction is commonly framed as supervised binary classification on historical customer records. The methods relevant to this project include linear models, tree-based models, ensemble methods, and techniques for class imbalance,

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.1511199

evaluation, and explanation. Selected foundational studies are summarized in Table I; their methodological contributions inform the project, while their validation settings are not treated as directly comparable telecom results.

A. Logistic Regression as a Baseline

Logistic Regression models the log-odds of the positive class as a linear function of the features and converts the result to a probability with the sigmoid function [9]. It trains quickly, and its coefficients can be read as odds ratios, which makes it a common explainable baseline [1], [2]. Its limitation is that nonlinear interactions are captured only if added as features, and its coefficients are sensitive to scaling and correlated predictors.

B. Decision Trees

Classification trees split the training data using feature conditions chosen to reduce an impurity measure such as the Gini index [8]. The resulting rules are easy to explain to non-technical users, but a single deep tree is unstable and tends to overfit unless depth and leaf size are controlled or the tree is pruned.

C. Ensemble Methods: Random Forest and Gradient Boosting

Random Forest averages many decorrelated trees trained on bootstrap samples and random feature subsets, which reduces variance [6]. Gradient boosting builds trees sequentially so that each tree corrects the errors of the previous ones [7]; XGBoost is a widely used, regularized implementation with efficient tree learning [4]. These methods often perform strongly on tabular data but have more hyperparameters and require careful validation.

D. Class Imbalance, Leakage, and Evaluation

When churners are a minority, a classifier can obtain high accuracy while missing most of them. Class weighting and SMOTE, which synthesizes minority examples by interpolating between neighbouring minority points [5], are common remedies, and their implementation for pipelines is available in imbalanced-learn [16]. Resampling must be applied only to training data; otherwise information leaks into evaluation [15]. Evaluation should use precision, recall, F1-score, ROC analysis [10], and, for rare positives, the precision-recall curve [11], with metric definitions as in [13].

E. Explainability and Decision Support

Explanation techniques such as permutation importance and SHAP values [12] describe how a model uses its features. They describe model behavior, not the causes of churn. Data-science practice also stresses connecting predictions to business costs and decisions [14], which motivates the risk-band and human-review design of Churn Guard.

III. COMPARATIVE ANALYSIS AND RESEARCH GAP

Table I is provided for the studies reviewed by the team. Each row should state the paper's principal technical emphasis, how it was validated, its relevance to Churn Guard, and the difference or limitation relative to this project. Results reported in other papers must not be treated as results of this project.

TABLE I COMPARISON OF REVIEWED STUDIES WITH CHURN GUARD

Study	Main focus and methods	Validation described in source	Relevance to Churn Guard	Difference / limitation relative to this project
T. Chen and C. Guestrin, 2016	Regularized, scalable gradient-boosted trees	Algorithm paper with empirical benchmark experiments	Provides the boosting method used as one candidate classifier	Method study; its benchmarks are not a telecom-specific result
Study	**Main focus and methods**	**Validation described in source**	**Relevance to Churn Guard**	**Difference / limitation relative to this project**
L. Breiman, 2001	Random Forest ensemble learning	Method paper with empirical classification examples	Motivates the bagged tree ensemble baseline	Not evaluated on the current IBM churn split

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

N. V. Chawla et al., 2002	Synthetic minority oversampling (SMOTE)	Method paper with classification experiments	Motivates trainingonly minority sampling	Does not establish that resampling improves this project's validation results
T. Saito and M. Rehmsmeier, 2015	Precision-recall evaluation under class imbalance	Comparative evaluation across imbalanced datasets	Supports reporting PR-AUC alongside ROC-AUC	General evaluation study, not a telecom deployment study
Churn Guard (this work)	Validated data, leakage-safe pipeline, LR/DT/RF/XGBoost comparison, threshold selection, risk dashboard	Held-out test set; crossvalidation on training data	Integrates modelling with a reviewable decisionsupport output	Single dataset; random split; no live deployment

A. Research Gap

Many churn-prediction demonstrations concentrate on a single accuracy figure and provide limited support for the next operational step. A score is less useful if analysts cannot identify the customer, understand the risk category, or review an appropriate follow-up. A second common limitation is insufficient attention to leakage and imbalance: resampling before the split, using future information as a feature, or tuning repeatedly on the test set can give misleadingly high results [15]. Churn Guard addresses these concerns through a pipeline that fits all transformations on training data, a fixed evaluation protocol, and a dashboard that presents risk estimates for human review. The project does not claim to solve causal attribution, realtime deployment, or uplift modelling.

B. Objectives

1. Define the churn target and identify the customer attributes available for prediction.
2. Prepare the data with validation, imputation, encoding, and a stratified split, applying any resampling only to training data.
3. Develop and compare Logistic Regression, Decision Tree, Random Forest, and XGBoost using precision, recall, F1score, ROC-AUC, and PR-AUC.
4. Build a dashboard that shows churn probability, predicted class, and risk band for each customer.
5. Map risk groups to suggested retention review categories while keeping the final decision with the business team.

IV. PROPOSED SYSTEM AND METHODOLOGY

A. System Overview

Churn Guard follows a layered architecture (Fig. 1): a data input layer, a validation and preprocessing layer, a model development layer, a model store, a prediction service, and a presentation layer. The saved model and its preprocessing pipeline are stored together so that prediction uses exactly the transformations learned during training.

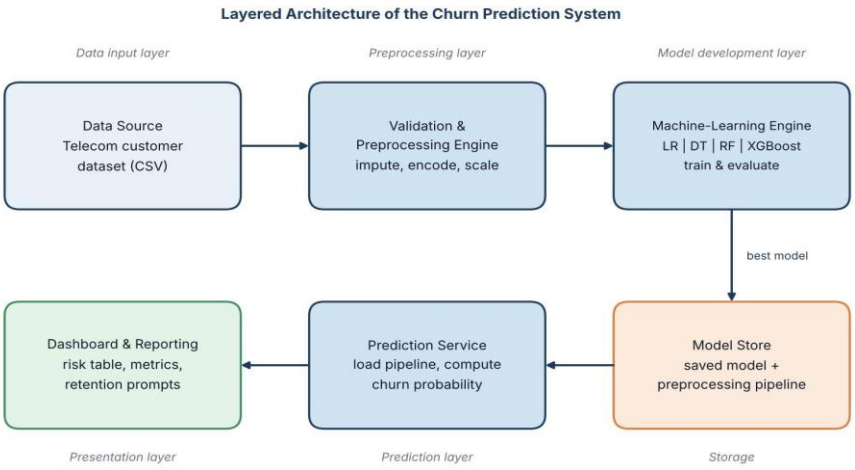

Fig. 1 Layered architecture of the Churn Guard system.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

B. Dataset and Target Definition

The experiments use the IBM Telco Customer Churn sample, obtained from IBM's public GitHub repository on 24 November 2026. The file contains 7,043 customer records and 21 columns, including the customer identifier and churn label. The target column is Churn, where Yes is the positive class and denotes a customer recorded as having left; the source file does not provide a dated observation window. Churn represents 26.54% of records (1,869 customers). The customer identifier is excluded from model training. Eleven blank TotalCharges entries are converted to missing numeric values and imputed using training data only. The fields used are summarized in Table II.

TABLE II DATASET FIELDS USED IN THE EXPERIMENTS

Field group	Example fields	Type	Handling
Customer tenure	tenure (months)	Numerical	Median imputation; scaling for Logistic Regression
Contract and billing	Contract, PaperlessBilling, PaymentMethod	Categorical	Most-frequent imputation; one-hot encoding
Services	PhoneService, MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies	Categorical	Most-frequent imputation; one-hot encoding
Charges	MonthlyCharges, TotalCharges	Numerical	Convert TotalCharges to numeric; median imputation
Target	Churn (Yes/No)	Binary	Yes mapped to 1 (churn); No mapped to 0 (stay)

Exploratory analysis examines the class distribution, missing values, duplicates, and feature distributions (Fig. 2). Observed associations, such as higher churn in a category, are descriptive and are not treated as causes.

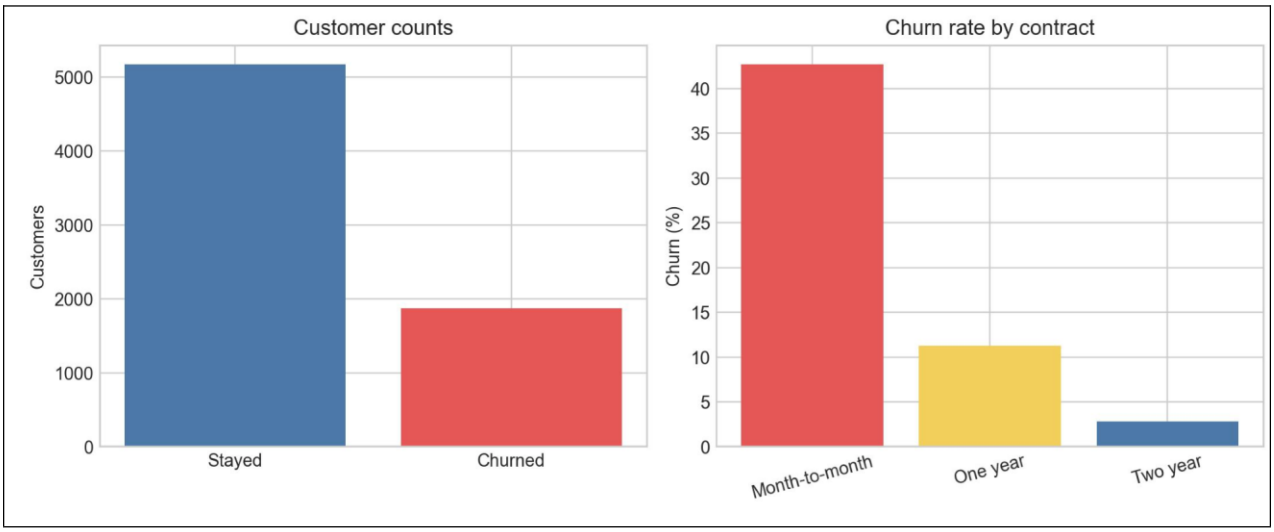

Fig. 2. Class distribution and churn rate by category.

The data are split with stratification into 60% training, 20% validation, and 20% held-out test records using random seed 42. Numerical features are imputed with the training median and standardized; categorical features are imputed with the training-set most frequent value and one-hot encoded with unseen categories ignored. Every preprocessing step is fitted inside the training pipeline. Validation is used for threshold selection, and the test set is evaluated once after the selected model and threshold are fixed. The customer identifier is removed before any split-based modelling.

The dataset is first split into training and test portions using stratification so that the churn ratio is preserved. Numerical features are imputed with the median and, for Logistic Regression, standardized; categorical features are imputed with the most frequent value and one-hot encoded with unseen categories ignored. All transformations are fitted on the training

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

portion only and are packaged with the classifier in a single pipeline, so the same transformations are applied to validation, test, and new records. Fitting an imputer on the full data, or applying SMOTE before the split, would allow test information to influence training and is therefore avoided.

The Random Forest imbalance sensitivity experiment compares no resampling, class-weighted training, and SMOTENC on the validation set. SMOTE-NC treats categorical fields as categorical during synthetic sampling and is applied only to the training partition; validation examples are never resampled. The displayed precision, recall, and F1-score use a threshold selected on the same validation predictions, so this table is exploratory and is not an independent estimate of treatment benefit. The four-model comparison uses the unweighted training objective for consistent baseline comparison.

Two treatments are compared with no treatment: class weighting, which increases the penalty for errors on churners, and SMOTE, which creates synthetic minority samples $x_{new} = x + \delta(x_{nn} - x)$ with δ drawn uniformly from [0, 1] [5]. SMOTE is applied only inside the training portion of each cross-validation fold. Because ordinary SMOTE on one-hot encoded data can produce fractional category values, a mixed-data variant or encoding-aware procedure is used where categorical features dominate. The treatment finally used was [class weights / SMOTE / none], chosen on validation metrics.

C. Candidate Models

Logistic Regression estimates the probability of churn as

Logistic Regression estimates churn probability as $P(\text{churn} = 1 \mid x) = 1 / (1 + \exp(-(w \cdot x + b)))$ and uses regularized log loss. The Decision Tree uses Gini impurity, $G = 1 - \Sigma\, p_k^2$, with maximum depth and minimum leaf size limiting complexity. Random Forest averages bootstrap trees trained on random feature subsets [6]. XGBoost adds regularized trees sequentially, $F_m(x) = F_{(m-1)}(x) + \eta\, f_m(x)$ [4], [7]. Fixed parameter settings used in this run are listed in Table III; they were specified before the held-out test evaluation rather than tuned against the test set.

and is fitted by minimizing the regularized log loss, with the regularization strength C tuned on validation data. The Decision Tree uses Gini impurity, $G = 1 - \Sigma\, p_k^2$, with maximum depth and minimum leaf size limiting complexity. Random Forest averages many such trees trained on bootstrap samples [6]. XGBoost minimizes a regularized objective by adding trees sequentially, $F_m(x) = F_{(m-1)}(x) + \eta\, f_m(x)$, where η is the learning rate [4], [7]. The hyperparameters selected for each model are listed in Table III.

TABLE III HYPERPARAMETERS SELECTED FOR EACH MODEL

Model	Parameter settings	Values used	Imbalance setting
Logistic Regression	C, penalty, solver, iterations	C=1.0; L2; liblinear; max_iter=2,000	None
Decision Tree	criterion, max depth, minimum leaf size	Gini; depth=6; min_samples_leaf=10	None
Random Forest	trees, depth, max features, minimum leaf size	300 trees; unrestricted depth; sqrt features; min_samples_leaf=2	None
XGBoost	learning rate, trees, depth, subsampling, column sampling	0.05; 200 trees; depth=3; subsample=0.90; colsample_bytree=0.90; reg_lambda=1	None

Five-fold stratified cross-validation is run on the 60% training portion with random seed 42. The complete preprocessing and classifier pipeline is fitted within each fold. Mean and sample standard deviation are reported for precision, recall, F1-score, ROC-AUC, and PR-AUC; threshold-dependent cross-validation scores use 0.50 for consistent fold comparison. Candidate models are selected using validation F1 after choosing each model's threshold to maximize validation F1. This threshold is frozen before the held-out test set is evaluated once

Models are compared with stratified k-fold cross-validation on the training portion, with k = [5], a fixed random seed of [value], and the complete preprocessing-and-model pipeline fitted inside each fold. Mean and standard deviation of

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

validation scores are recorded. The selection metric is [PR-AUC / F1 / ROC-AUC]. The decision threshold is chosen on validation predictions according to a documented objective ([maximum F1 / target recall / minimum cost]) and is fixed before the held-out test set is evaluated, once, for the selected model

For the positive churn class, accuracy = (TP + TN) / (TP + TN + FP + FN); precision = TP / (TP + FP); recall = TP / (TP + FN); and F1-score = 2 × (precision × recall) / (precision + recall) [13]. Here TP is a churner correctly predicted to churn; TN is a non-churner correctly predicted to stay; FP is a non-churner incorrectly predicted to churn; and FN is a churner incorrectly predicted to stay. These formulas are appropriate for the binary target when Yes is designated positive. ROC-AUC and PRAUC use predicted probabilities [10], [11].

For the churn class, accuracy is (TP + TN) / (TP + TN + FP + FN); precision is TP / (TP + FP); recall is TP / (TP + FN); and F1-score is 2PR / (P + R) [13]. ROC-AUC is computed from predicted probabilities [10], and PR-AUC is reported because churners are a minority [11]. Lift at the top 10% of ranked customers is also reported: the churn rate in that group divided by the overall churn rate.

The proposed dashboard would accept a customer file, apply a saved preprocessing and prediction pipeline, and show estimated churn probability, predicted class, and risk band for each record. Suggested review categories may include servicequality review, billing clarification, plan suitability consultation, contract information, or no immediate action. These are prompts for staff; the system should not send messages or offer discounts automatically. Figure 7 is an illustrative wireframe, not a screenshot of a running application.

The dashboard accepts a customer file, applies the saved pipeline, and shows each customer's estimated probability, predicted class, and risk band (low, medium, high; boundaries defined in project configuration and not presented as industry standards). A detail view lists selected attributes and suggested review categories such as service-quality review, billing clarification, plan suitability consultation, contract information, loyalty-team review, or no immediate action. These are prompts for staff; the system does not send messages or offer discounts automatically.

V. USE CASES AND TEST SCENARIOS

The principal use case is a telecom analyst who uploads a customer file, reviews predicted risk, filters records, and records a human-reviewed follow-up. The modelling workflow was executed for this paper, but no runnable dashboard or service implementation was supplied for functional testing; Table IV distinguishes measured workflow evidence from application tests that remain unverified.

TABLE IV FUNCTIONAL TEST SCENARIOS

Scenario	Input	Expected behavior	Recorded result	Status
Valid dataset	File with all required columns	Data read; summary displayed	Not run: runnable application was not supplied.	Not verified
Missing target column	Training file without churn label	Clear error; training does not start	Not run: runnable application was not supplied.	Not verified
Missing feature values	Blank values in supported columns	Configured imputation applied	Not run: runnable application was not supplied.	Not verified
Unsupported file	Wrong file extension	File rejected with a clear message	Not run: runnable application was not supplied.	Not verified
Saved model loading	Restart and load pipeline	Loads without schema mismatch	Not run: runnable application was not supplied.	Not verified
Single prediction	One valid customer record	Probability in [0, 1] and a class	Not run: runnable application was not supplied.	Not verified

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

Batch prediction	Several valid records	One result per record; row order preserved	Not run: runnable application was not supplied.	Not verified
Reproducibility	Same split and seed	Consistent results	Seeded experiment executed; repeatoutput equality not separately audited.	Partially verified

VI. RESULTS AND BASELINE COMPARISON

Table V reports the measured results on the held-out test set. Each candidate's threshold was selected on validation data by maximum F1 and then held fixed. Logistic Regression is the baseline. Results are from a reproducible re-run on the cited public benchmark and should not be interpreted as results from a live operator system.

TABLE V TEST-SET PERFORMANCE OF THE CANDIDATE MODELS

Model	Accuracy	Precision	Recall	F1-score	ROC-AUC	PR-AUC
Logistic Regression (baseline)	0.787	0.614	0.532	0.570	0.832	0.622
Decision Tree	0.752	0.526	0.668	0.589	0.804	0.599
Random Forest	0.754	0.528	0.703	0.603	0.821	0.614
XGBoost	0.770	0.554	0.682	0.612	0.832	0.638

Selected model: XGBoost; validation-selected threshold: 0.358; imbalance treatment in the four-model comparison: none; random seed: 42; package versions: scikit-learn 1.7.2, XGBoost 3.1.1, imbalanced-learn 0.14.0, pandas 3.0.1.

Selected threshold: [value]; imbalance treatment: [treatment]; random seed: [value]; software versions: [Python, scikit-learn, XGBoost].

TABLE VI CROSS-VALIDATION RESULTS (MEAN ± STANDARD DEVIATION, VALIDATION FOLDS)

Model	Precision	Recall	F1-score	ROC-AUC	PR-AUC
Logistic Regression	0.655 ± 0.045	0.547 ± 0.034	0.596 ± 0.033	0.845 ± 0.021	0.663 ± 0.049
Decision Tree	0.629 ± 0.033	0.519 ± 0.030	0.568 ± 0.016	0.818 ± 0.016	0.605 ± 0.029
Random Forest	0.671 ± 0.025	0.504 ± 0.033	0.575 ± 0.021	0.841 ± 0.015	0.647 ± 0.029
Model	**Precision**	**Recall**	**F1-score**	**ROC-AUC**	**PR-AUC**
XGBoost	0.669 ± 0.030	0.533 ± 0.037	0.593 ± 0.029	0.847 ± 0.017	0.672 ± 0.029

The selected model's test-set confusion matrix and candidate ROC curves are shown in Figs. 4 and 5; the corresponding counts are given in Table VII.

The confusion matrix and ROC curves of the selected model are shown in Fig. 4 and Fig. 5, and its confusion-matrix counts are given in Table VII.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

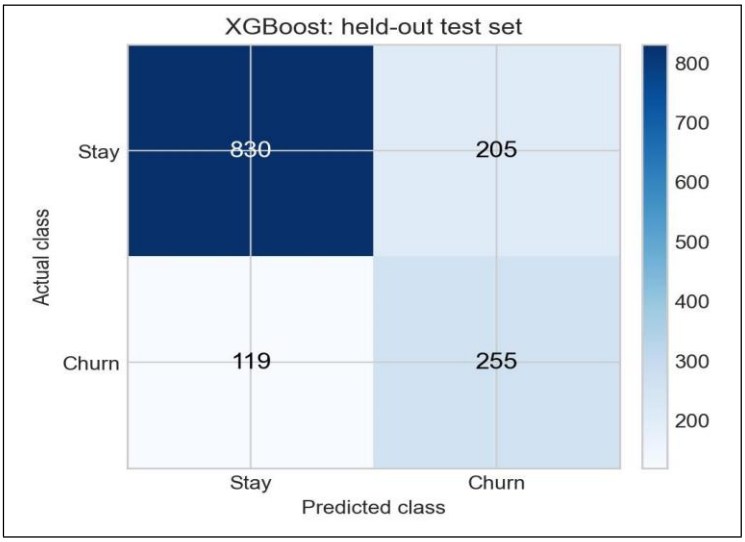

Fig. 4 Confusion matrix of the selected model (test set).

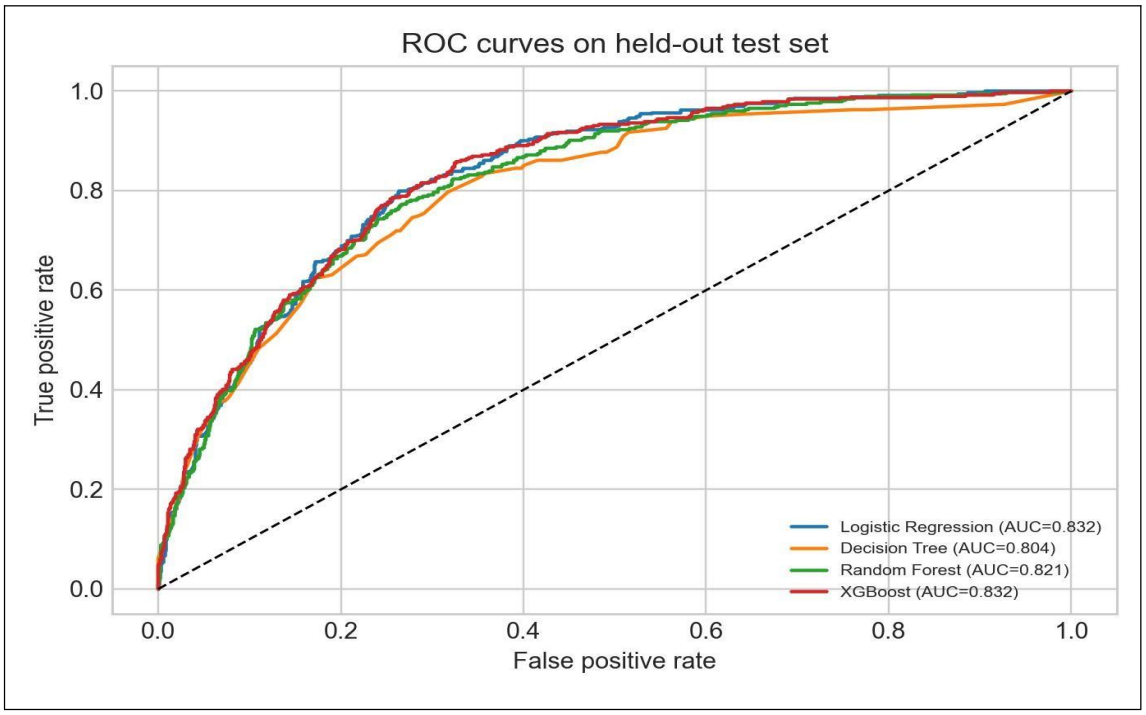

Fig. 5 ROC curves of the candidate models (test set).

TABLE VII CONFUSION MATRIX COUNTS OF THE SELECTED MODEL

	Predicted: Churn	**Predicted: Stay**
Actual: Churn	TP = 255	FN = 119
Actual: Stay	FP = 205	TN = 830

XGBoost had the highest validation F1-score (0.660) and was selected at threshold 0.358. On the held-out test set it achieved F1 0.612, ROC-AUC 0.832, and PR-AUC 0.638. Logistic Regression, evaluated at its own validation-selected threshold, achieved test F1 0.570, ROC-AUC 0.832, and PR-AUC 0.622. The observed test F1 difference is +0.041; this is a single split and is not a significance test. Five-fold training-set cross-validation gives XGBoost ROC-AUC 0.847 ± 0.017 and PR-AUC 0.672 ± 0.029, which provides context on fold variation.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

[Describe which model gave the best validation and test scores, by how much it exceeded the Logistic Regression baseline, and whether the difference is larger than the fold-to-fold variation.]

Table VIII compares no resampling, class weights, and SMOTE-NC for the Random Forest on validation data. The threshold for each treatment is independently selected to maximize validation F1. The test results in Table V remain the pre-specified four-model comparison and do not represent a post hoc test-set selection among these treatments.

Table VIII compares no treatment, class weights, and SMOTE for the selected model on validation data.

TABLE VIII EFFECT OF IMBALANCE TREATMENT (VALIDATION DATA)

Treatment	Precision	Recall	F1-score	PR-AUC
None	0.567	0.738	0.641	0.654
Class weights	0.583	0.711	0.641	0.655
SMOTE-NC (training only)	0.554	0.735	0.632	0.629

No resampling and class-weighted training produced similar validation F1 in this run. SMOTE-NC altered the precisionrecall balance; the validation estimates are exploratory because the treatment and threshold are compared on the same validation partition. A future evaluation should choose the treatment within nested or cross-validated training data and confirm it on a separate later-period sample.

The threshold was selected on validation predictions to maximize F1, giving 0.358 for XGBoost. At that threshold, validation precision was 0.594 and recall was 0.743. This balances false alarms and missed churners equally in the F1 objective; an operator should replace this objective with an explicit contact-capacity or expected-cost policy before deployment (Fig. 6).

[Describe the threshold objective, the chosen value, and the validation precision and recall at that value (Fig. 6).]

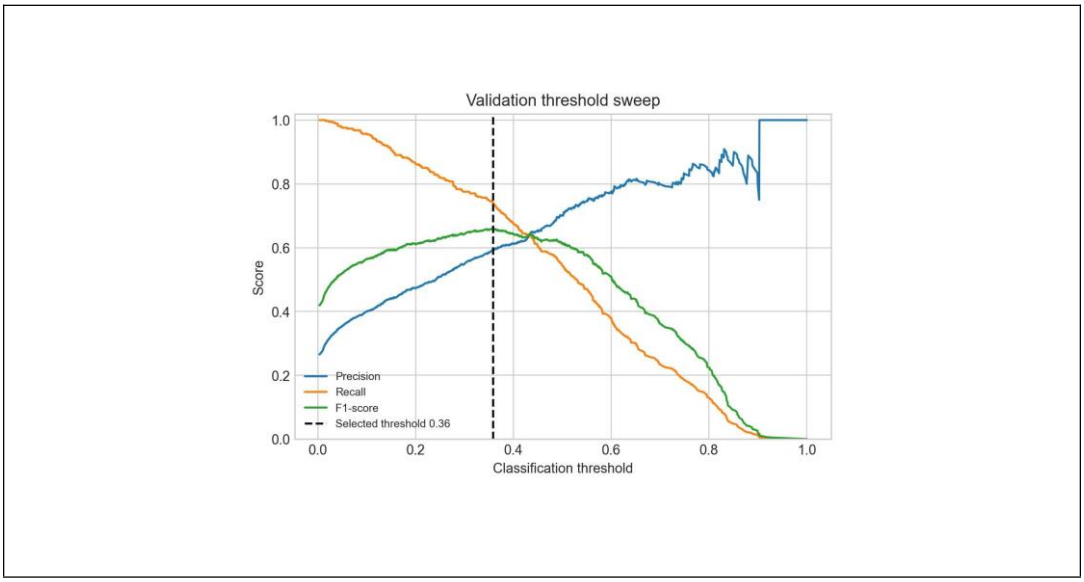

Fig. 6 Precision, recall, and F1-score against the classification threshold (validation data).

Table IX lists the leading input fields by permutation importance on the validation partition, measured as the decrease in F1 when a field is shuffled (five repeats; threshold frozen at the selected value). These values describe model behavior on this sample and are not causal findings.

Table IX lists the most influential features by permutation importance. These describe model behavior and are not causal findings.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511199

TABLE IX MOST INFLUENTIAL FEATURES (PERMUTATION IMPORTANCE)

Rank	Feature	Importance	Interpretation
1	Contract	0.149 ± 0.016	Contract commitment is associated with model predictions.
2	tenure	0.080 ± 0.017	Customer tenure is associated with model predictions.
3	PaymentMethod	0.029 ± 0.005	Payment method is associated with model predictions.
4	MonthlyCharges	0.028 ± 0.012	Monthly charges are associated with model predictions.
5	InternetService	0.024 ± 0.009	Internet service type is associated with model predictions.

TABLE X SEGMENT-LEVEL PERFORMANCE (TEST SET)

Segment	Records	Churn rate	Precision	Recall
Month-to-month contract	786	41.2%	0.554	0.778
One year contract	284	12.7%	0.600	0.083
Two year contract	339	4.1%	0.000	0.000
Tenure 0–12 months	431	46.4%	0.610	0.860
Tenure 13–48 months	522	24.3%	0.458	0.559
Segment	**Records**	**Churn rate**	**Precision**	**Recall**
Tenure 49+ months	456	10.3%	0.522	0.255

Figure 7 presents a proposed dashboard wireframe with a summary area, risk-review queue, and staff-review prompts. It is an illustrative interface concept; it is not evidence that a functional application was implemented or tested. The example rows are synthetic labels and contain no personal identifiers.

Fig. 7 Proposed Churn Guard dashboard wireframe (illustrative; not a live application screenshot).

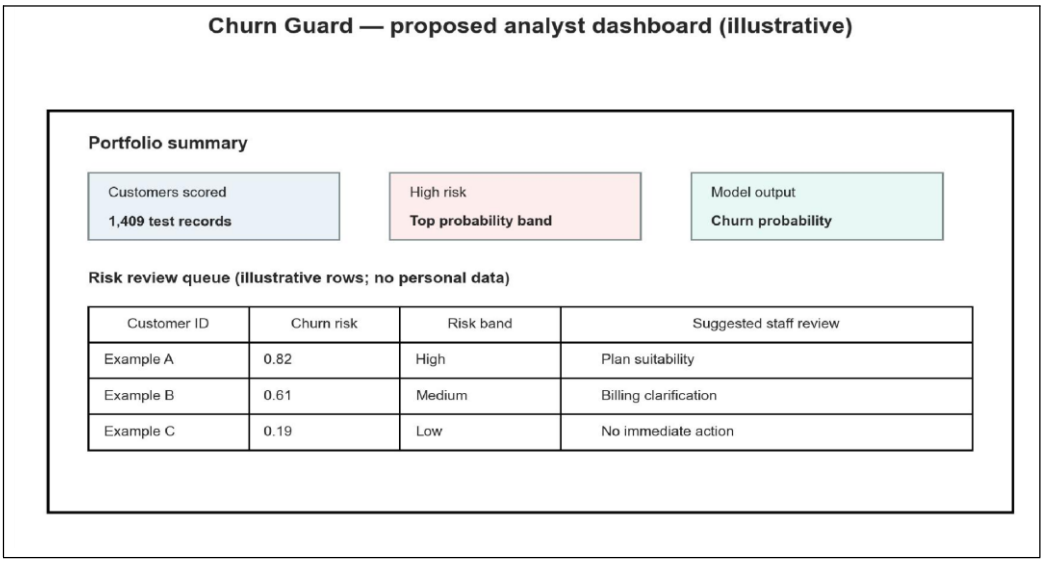

Fig. 7 Churn Guard dashboard showing risk table, filters, and customer detail.

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511199

Across the held-out test set, XGBoost achieved the strongest F1-score (0.612) among the compared fixed pipelines; Logistic Regression had the highest accuracy (0.787), illustrating why accuracy alone does not express churner detection. For the selected model, precision was 0.554, recall 0.682, ROC-AUC 0.832, and PR-AUC 0.638. This supports a limited statement about discrimination and thresholded classification on this split of the IBM sample. It does not establish generalization to another operator, time period, or the effectiveness of retention actions.

XGBoost yielded the highest validation F1 and the highest held-out test F1 in this single benchmark split, while Logistic Regression remained competitive and had higher test accuracy. The selected model's F1 gain over the baseline was +0.041, a modest difference that should be weighed against ensemble complexity and checked with repeated seeds or temporal validation. The permutation ranking placed contract type and tenure at the top, followed by payment method, monthly charges, and internet service; these associations reflect the dataset and model, not causes of churn.

Accuracy alone would understate the trade-off on the imbalanced target. At the frozen threshold 0.358, the selected model identified 68.2% of test churners and 55.4% of its churn alerts were correct. This operating point was chosen to maximize F1 rather than to meet an operator's contact budget. The leading permutation features were Contract, tenure, PaymentMethod; they describe associations in historical records, not reasons why an individual customer leaves [12].

Accuracy alone would have been misleading on this imbalanced target; the precision, recall, F1-score, and PR-AUC values show [comment] [11], [13]. The chosen threshold of [value] reflects [business objective], and the dashboard's risk bands allow retention teams to match contact volume to capacity. The most influential features ([list]) agree with [or differ from] common expectations about contract type and tenure, but they describe associations in historical records and not reasons why individual customers leave [12].

The scope of the contribution is limited. Churn Guard evaluates one dataset with a random split, does not use time-ordered validation, and does not measure the business effect of retention actions. Measuring that effect would require a controlled pilot in which a randomly chosen subset of high-risk customers receives an action and the remainder serves as a control group.

VII. LIMITATIONS AND FUTURE WORK

- Validate on a later time period, an external dataset, or a live pilot, because a random split may not represent future deployment and the model may drift.
- Add probability calibration and threshold optimization based on explicit contact costs and customer value.
- Report explanations with SHAP or permutation importance in the dashboard, presented as model behavior and not as causes.
- Perform fairness checks across customer groups and document findings in a model card.
- Repeat experiments across multiple seeds and report mean, standard deviation, and worst-case performance.
- Evaluate uplift modelling and survival analysis to target by expected benefit and timing of action.
- Integrate with a CRM platform only after privacy, security, and business approval.

VIII. CONCLUSION

This paper presented a reproducible machine-learning workflow for telecom churn prediction and retention support. It combines validation, leakage-safe preprocessing, comparison of Logistic Regression, Decision Tree, Random Forest, and XGBoost, validation-based threshold selection, and a proposed dashboard wireframe for human review.

On the held-out IBM benchmark test set, XGBoost recorded ROC-AUC 0.832, precision 0.554, recall 0.682, and F1-score 0.612; Logistic Regression recorded F1-score 0.570 at its own validation-selected threshold. These measurements describe one fixed random split and do not demonstrate performance on future customers, another operator, or a live service. Timebased validation, calibration, fairness checks, and a controlled retention pilot are required before operational claims can be made.

ACKNOWLEDGMENT

The authors thank the Department of Computer Science and Engineering, East West College of Engineering, and the project guide for their guidance and support.

Funding Source: Not stated in the supplied manuscript; authors to confirm.

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.1511199

Data Availability: The experiment uses IBM's public Telco Customer Churn sample, cited in Reference [17]. The dataset contains customer-level records and should be used subject to the source repository's terms. The reported measurements were generated by the reproducible run described in this paper.

Authors' Contribution: [To be completed and approved by the authors.] All authors should review and approve the final manuscript.

Use of AI and AI-Assisted Technologies: AI-assisted tools were used to support language editing and document formatting.

The authors remain responsible for verifying the technical content, experimental setup, and references.

Conflict of Interest: The authors declare no conflict of interest.

Figures 2 and 4–6 were generated from the benchmark experiment reported here. Figure 7 is an illustrative wireframe. Authors should verify the source and reuse permissions for any retained figures before submission.

DECLARATIONS

Funding: Not stated in the supplied manuscript; authors to confirm.

Data availability: IBM Telco Customer Churn sample; see Reference [17].

Conflict of interest: The authors declare no conflict of interest.

Author contributions: The authors contributed to the project development, experiments, evaluation, and preparation of the manuscript.

REFERENCES

[1]. T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed. New York, NY, USA: Springer, 2009.
[2]. C. M. Bishop, Pattern Recognition and Machine Learning. New York, NY, USA: Springer, 2006.
[3]. F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
[4]. T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.
[5]. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.
[6]. L. Breiman, “Random forests,” Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.
[7]. J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” Ann. Statist., vol. 29, no. 5, pp. 1189– 1232, 2001.
[8]. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and Regression Trees. Belmont, CA, USA: Wadsworth, 1984.
[9]. D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, Applied Logistic Regression, 3rd ed. Hoboken, NJ, USA: Wiley, 2013. [10] T. Fawcett, “An introduction to ROC analysis,” Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.
[10]. T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets,” PLoS ONE, vol. 10, no. 3, Art. no. e0118432, 2015.
[11]. S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 4765–4774.
[12]. D. M. W. Powers, “Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation,” J. Mach. Learn. Technol., vol. 2, no. 1, pp. 37–63, 2011.
[13]. F. Provost and T. Fawcett, Data Science for Business. Sebastopol, CA, USA: O’Reilly Media, 2013.
[14]. S. Kaufman, S. Rosset, C. Perlich, and O. Stitelman, “Leakage in data mining: Formulation, detection, and avoidance,” ACM Trans. Knowl. Discov. Data, vol. 6, no. 4, Art. no. 15, 2012.
[15]. G. Lemaître, F. Nogueira, and C. K. Aridas, “Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning,” J. Mach. Learn. Res., vol. 18, no. 17, pp. 1–5, 2017.