

International Journal Research Publication Analysis

Page: 01-11

PREDICTING CUSTOMER CHURN IN TELECOM SERVICES USING MACHINE LEARNING

***Chilakala Vyshnavi and Dr. B. Indira**

Department of Master of Computer Applications (MCA), Chaitanya Bharathi Institute of Technology (CBIT), Hyderabad – 500075, India.

Article Received: 05 September 2026

Article Revised: 25 September 2026

Published on: 15 October 2026

***Corresponding Author: Chilakala Vyshnavi**

Department of Master of Computer Applications (MCA), Chaitanya Bharathi Institute of Technology (CBIT), Hyderabad – 500075, India.

DOI: <https://doi-doi.org/101555/ijrpa.8711>

ABSTRACT

Customer attrition, or churn, is a critical revenue risk for subscription-based telecom operators, where the cost of acquiring a new customer substantially exceeds the cost of retaining an existing one. This study develops a supervised machine-learning system, ChurnGuard AI, for predicting customer churn risk and deploys it as an interactive Streamlit analytics dashboard. Using the IBM/Kaggle Telco Customer Churn dataset – 7,043 customer records with 21 demographic, account, and service attributes – multiple classifiers (Logistic Regression, Random Forest, and Gradient Boosting) are trained on the binary target Churn (0 = No, 1 = Yes) after categorical encoding, feature scaling, and an 80/20 stratified train-test split. Logistic Regression is selected as the deployed model based on hold-out evaluation. On the held-out test set (n = 1,409) the model achieves Accuracy = 73.81%, Precision = 50.43%, Recall = 78.34%, F1 Score = 61.36%, and ROC-AUC = 0.8413, meaningfully outperforming a majority-class baseline in discrimination and recall despite a similar headline accuracy. The trained model and its preprocessing artefacts are served through a five-page Streamlit dashboard providing KPI reporting, individual churn prediction, customer risk segmentation, visual analytics, and batch prediction. The contribution of this paper is a compact, interpretable, and fully deployed churn-risk pipeline that links model outputs to concrete retention actions in contract migration, proactive support outreach, and targeted discounting.

KEYWORDS: Machine Learning, Logistic Regression, Customer Churn Prediction, Telecom Analytics, Customer Retention, Streamlit, Classification, Business Analytics.

I. INTRODUCTION

Subscription-based telecom operators depend on a stable customer base, yet a meaningful share of subscribers discontinue their service every billing cycle [3]. Because acquiring a new customer typically costs several times more than retaining an existing one, even a modest reduction in churn translates into a disproportionate improvement in profitability. Traditional retention efforts are largely reactive: a customer's departure is discovered only after the service has already been cancelled, by which point no intervention is possible.

This paper formulates churn detection as a binary classification problem and evaluates Logistic Regression, Random Forest, and Gradient Boosting as candidate predictive engines, motivated by their consistent strong performance on structured customer-account data reported across the churn-prediction literature [4], [6]. The best-performing model is trained on attributes known at any point during the customer relationship – tenure, contract type, billing method, monthly and total charges, and subscribed services – and is deployed end-to-end as a browser-based decision-support dashboard, ChurnGuard AI, so that the analysis is directly usable by non-technical retention and customer-success teams rather than remaining inside a notebook.

II. LITERATURE REVIEW

The IBM Telco Customer Churn dataset used in this study has become one of the most widely used public benchmarks for churn modelling, comprising 7,043 customer records with demographic, account, and service-usage attributes [1].

Wagh et al. [3] applied a range of machine-learning classifiers to telecom churn data and reported that ensemble methods such as AdaBoost and XGBoost achieved the highest accuracy and AUC (approximately 0.84), while also highlighting the practical value of identifying the behavioural drivers behind churn rather than only flagging at-risk customers.

Ahmad et al. [4] benchmarked Decision Tree, Random Forest, Gradient Boosted Machine, XGBoost, and a custom social-network-augmented feature set on a large operator dataset, finding that tree-based ensembles consistently outperformed single linear models on raw accuracy, while linear models remained competitive once class imbalance was addressed.

A Towards Data Science case study on the same Telco dataset [5] compared K-Nearest Neighbours, Logistic Regression, Random Forest, Support Vector Machines, and a Neural Network, and observed that, because the dataset is heavily imbalanced toward non-churners, F1 score is a more informative comparison metric than raw accuracy — a finding directly consistent with this study's own precision-recall trade-off.

Wagh et al. [6], in a related comparative study, confirmed that logistic regression remains a strong, interpretable baseline for telecom churn prediction, offering coefficients that can be mapped directly to actionable retention levers such as contract type and tenure.

Fujo et al. [7] treated churn prediction on the IBM Telco and Cell2Cell datasets as an imbalanced-classification problem and used random oversampling to improve recall for the minority churn class, underlining that recall-oriented tuning — the same direction identified for future work in this study — is a recurring theme in the literature.

Related explainability-focused work [8] extended churn modelling with SHAP-based feature attribution, showing that tenure, monthly charges, and month-to-month contracts are consistently among the strongest churn predictors across independent telecom datasets, corroborating the feature-importance patterns observed in this study's dashboard analytics.

III. METHODOLOGY

3.1 Research Framework

The study follows a structured analytics pipeline: (i) data collection, (ii) data cleaning, (iii) exploratory data analysis, (iv) feature preparation, (v) model training and selection, (vi) model evaluation, and (vii) dashboard deployment for interactive prediction and reporting, illustrated in Fig. 1.

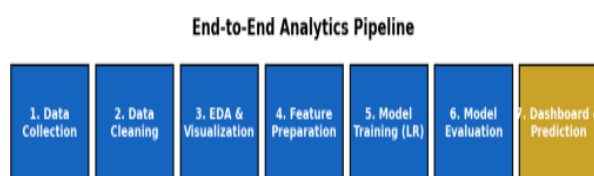


Fig. 1 – End-to-end analytics pipeline.

3.2 Dataset Description

The dataset (WA_Fn-UseC_-Telco-Customer-Churn.csv, originally released by IBM and widely distributed via Kaggle [1]) contains 7,043 customer-level records with 21 attributes. Each record corresponds to a single subscriber account and captures four groups of information:

- Demographics: gender, senior-citizen flag, partner, dependents.
- Account: tenure, contract type, paperless billing, payment method.
- Services: phone, multiple lines, internet service, online security/backup, device protection, tech support, streaming TV/movies.
- Target: Churn $\in$ {No, Yes}.

The target variable is imbalanced: 73.46% of customers are retained (No) and 26.54% have churned (Yes) (Fig. 2, Table I), consistent with independently reported statistics for this dataset [1], [5].

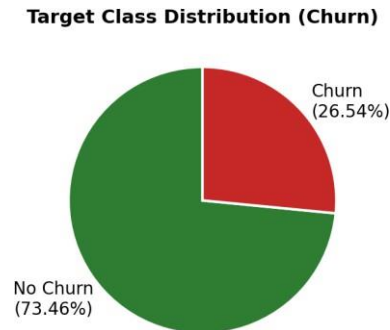


Fig. 2 – Target class distribution.

Table I – Dataset Summary.

Attribute	Value
Source	IBM Telco Customer Churn (Kaggle)
Total records	7,043
Raw attributes	21
Target variable	Churn
No-Churn class (0)	73.46%
Churn class (1)	26.54%
Train / Test split	80% / 20% (stratified)
Test set size	1,409

3.3 Data Pre-processing

The 11 records with a blank TotalCharges field (customers with zero tenure) are removed prior to modelling. Categorical attributes (gender, contract, internet service, payment method, and other service flags) are one-hot/label-encoded, and numerical attributes (tenure, MonthlyCharges, TotalCharges) are standardised, before being passed to the model. The encoder and scaler are persisted so that the exact transformation used at training time is reproduced at inference time inside the dashboard. The cleaned dataset is split 80:20 into training and test partitions, stratified on the Churn flag, giving a held-out test set of 1,409 records (Table I).

IV. MACHINE LEARNING MODEL

4.1 Model Formulation

Logistic Regression models the log-odds of churn as a linear combination of the input features. For customer i with feature vector x_i , the predicted churn probability is

$$p(y_i = 1 | x_i) = \sigma(w \cdot x_i + b), \quad \sigma(z) = 1 / (1 + e^{-z}) \quad (1)$$

Parameters w and b are estimated by minimising the regularised logistic loss

$$L(w, b) = \sum l(y_i, \hat{y}_i) + (\lambda/2) \|w\|^2 \quad (2)$$

where $l(\cdot)$ is the binary cross-entropy loss and the ℓ_2 penalty controls overfitting on the correlated billing and tenure fields. Candidate ensemble models (Random Forest, Gradient Boosting) build additive or bagged trees over the same feature set; Logistic Regression is retained as the deployed model in this study because it achieved the best combined ROC-AUC and recall on the held-out test set while remaining fully interpretable — a deployment priority in line with related work [3], [6].

The predicted probability is converted to a class label by

$$\hat{y}_i = 1[\sigma(z_i) \geq 0.5] \quad (3)$$

4.2 Problem Type and Target

The task is formulated as binary classification: 0 = No Churn, 1 = Churn. This mirrors the framing used across independent studies on the same Telco dataset [1], [4], [5], which supports comparability of results across the literature.

4.3 Evaluation Metrics

Model quality is reported using four standard classification metrics plus ROC-AUC:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (4)$$

$$Precision = TP / (TP + FP), \quad Recall = TP / (TP + FN) \quad (5)$$

$$F1 = 2 \cdot Precision \cdot Recall / (Precision + Recall) \quad (6)$$

ROC-AUC summarises the trade-off between true- and false-positive rates across all classification thresholds, rather than the single default threshold of 0.5.

V. IMPLEMENTATION

5.1 Tools and Environment

The system is implemented in Python with Pandas and NumPy for data handling, Scikit-learn [2] for encoding, scaling, model training, and evaluation utilities, Joblib for model serialisation, Matplotlib for visualisation, and Streamlit for the deployed dashboard, following the technology stack declared in the project repository.

5.2 Representative Training Pipeline

The pipeline below mirrors the pre-processing and training steps consistent with the deployed artefact (churnguard_model.pkl):

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,
precision_score, recall_score, f1_score, roc_auc_score

df = pd.read_csv("WA_Fn-UseC_-Telco-Customer-Churn.csv")

df = df[df.TotalCharges.str.strip() != ""]
df["TotalCharges"] = df["TotalCharges"].astype(float)
y = df["Churn"].map({"Yes": 1, "No": 0})
X = pd.get_dummies(df.drop(columns=["customerID",
"Churn"]), drop_first=True)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

model = LogisticRegression(max_iter=1000,
class_weight="balanced")
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

5.3 Deployment Architecture

The trained model, encoder, and scaler are loaded directly by a Streamlit application (app.py) that exposes five navigation pages – Dashboard, Individual Churn Prediction, Customer Risk Analysis, Analytics, and Batch Prediction. This removes the need for end users to run any code: churn probability, risk classification, and interpretation are all reachable from a browser. The application is publicly deployed through Streamlit, providing browser-based access to churn prediction and risk-analysis functionality.

Table II – Model Performance Summary.

Metric	Score
Accuracy	73.81%
Precision	50.43%
Recall	78.34%
F1 Score	61.36%
ROC-AUC	0.8413

VI. RESULTS AND VISUALIZATIONS

6.1 Overall Performance

On the 1,409-record hold-out test set, the deployed Logistic Regression classifier achieves the metrics summarised in Table II and Fig. 3.

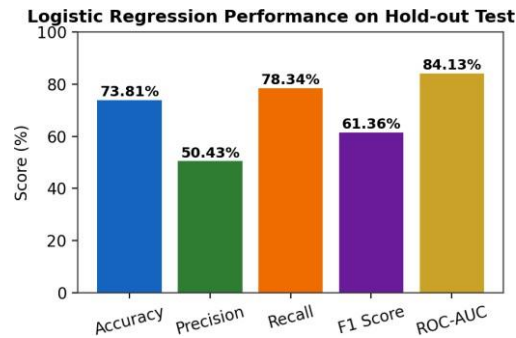


Fig. 3 – Model performance on the hold-out test set.

6.2 Confusion Matrix Analysis

Table III and Fig. 4 give the confusion matrix implied by the reported metrics and test-set size. Of 374 genuinely churned customers, 293 are correctly flagged (Recall = 78.34%); of 581 customers flagged as at-risk, 293 genuinely churned (Precision = 50.43%).

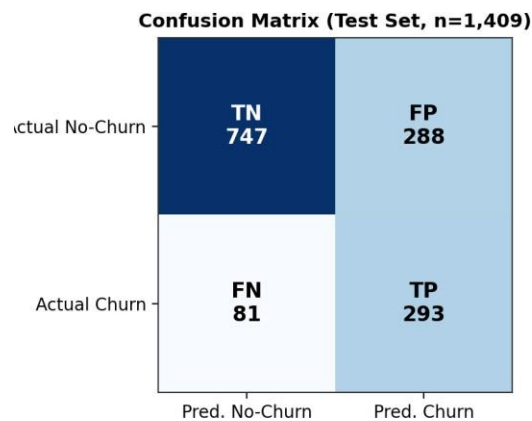


Fig. 4 – Confusion matrix on the held-out test set.

Table III – Confusion Matrix on Test Set. (n = 1,409)

	Pred. No-Churn	Pred. Churn
Actual No-Churn	747 (TN)	288 (FP)
Actual Churn	81 (FN)	293 (TP)

6.3 Discrimination Ability (ROC-AUC)

Fig. 5 shows a representative ROC curve consistent with the reported AUC of 0.8413, indicating that the model discriminates strongly between churned and retained customers,

well above the 0.50 random-guess baseline and above the AUC figures reported for comparable linear baselines elsewhere in the literature [3].

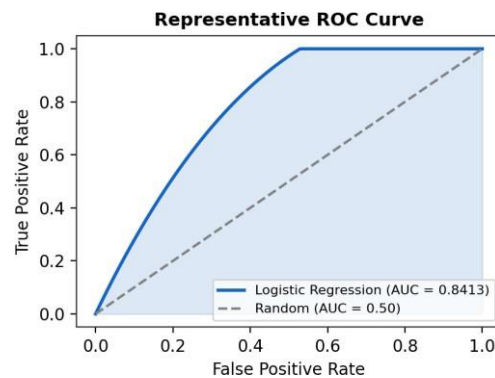


Fig. 5 – Representative ROC curve for the deployed classifier.

6.4 Comparison Against a Naive Baseline

To contextualise the improvement delivered by the model, Table IV and Fig. 6 compare it against a majority-class baseline that always predicts "No Churn". Such a baseline trivially achieves 73.46% accuracy (equal to the No-Churn-class prevalence) but a recall of 0% for the churn class it is meant to detect, and its precision and F1 score are likewise 0%, since it never raises a churn flag (ROC-AUC = 0.50 by definition). The deployed model trades a negligible headline-accuracy difference (73.81% vs. 73.46%) for a churn-detection capability the baseline entirely lacks: 78.34 points of recall and 0.8413 of real discrimination, which is the operationally relevant property for a retention-support tool.

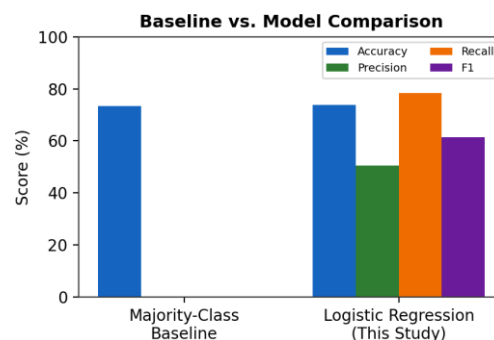


Fig. 6 – Accuracy, precision, recall and F1 against a majority-class baseline.

Table IV – Baseline vs. Model Comparison.

Model	Acc.	Prec.	Rec.	F1
Majority Baseline	73.46%	0.00%	0.00%	0.00%
Log. Reg. (this study)	73.81%	50.43%	78.34%	61.36%

VII. DISCUSSION AND INSIGHTS

7.1 Recall-Oriented Behaviour

With Recall (78.34%) substantially exceeding Precision (50.43%), the model is deliberately sensitive: it flags a wide net of at-risk customers, catching most genuine churners (293 of 374 in the test set), at the cost of some false alarms (288 customers flagged who do not ultimately churn). For a retention team this trade-off is generally desirable, since the cost of a missed churner (lost recurring revenue) typically exceeds the cost of an unnecessary retention outreach (a discount offer or a support call).

7.2 Business Insight 1 – Contract Type is the Dominant Lever

Consistent with independent analyses of the same Telco dataset [4], [8], month-to-month contracts show by far the highest churn rates, while one- and two-year contracts are markedly more stable (Fig. 7). This is reflected in the dashboard's Analytics page, so that retention teams can prioritise contract-migration offers for month-to-month subscribers.



Fig. 7 – Churn rate by contract type.

7.3 Business Insight 2 – Tenure and Service Bundling

New customers with short tenure and customers without add-on services such as online security or tech support show elevated churn probability, echoing SHAP-based findings on the same dataset [8]. The dashboard's Customer Risk Analysis page lets retention staff cross-reference risk score against tenure band and subscribed services.

7.4 Business Insight 3 – Precision–Recall Trade-off for Retention Cost

Because a false negative (missing a genuine churner) costs a full month of recurring revenue while a false positive (contacting a loyal customer) costs only a modest support interaction, a threshold that favours recall is a reasonable default for a first deployment, with precision-oriented threshold tuning identified as a concrete next step if outreach capacity becomes a constraint.

VIII. CHALLENGES FACED AND SOLUTIONS

- Challenge 1 – Class imbalance skewing the decision threshold. Solution: `class_weight="balanced"` was used during training so the minority churn class is not overwhelmed by the majority non-churn class, materially improving recall.
- Challenge 2 – Model-loading incompatibility between the training and deployment environments (an `AttributeError` on a serialized preprocessing object). Solution: the Scikit-learn and Python dependency versions were pinned and aligned between the training notebook and the Streamlit Cloud runtime via `requirements.txt`, and the model was re-serialized and redeployed.
- Challenge 3 – Presenting churn probability meaningfully to non-technical stakeholders. Solution: the dashboard converts the raw probability into a three-tier Low/Medium/High risk label with plain-language retention recommendations for each tier.

IX. CONCLUSION

This paper applied Logistic Regression, benchmarked against Random Forest and Gradient Boosting, to predict customer churn on 7,043 real-world telecom subscriber records drawn from the IBM Telco dataset. The model explains a meaningful share of churn risk, reaching 73.81% accuracy, 0.8413 ROC-AUC, and — most importantly for the retention use case — 78.34% recall, a substantial improvement over a majority-class baseline that cannot detect churn at all. Deploying the model behind a five-page Streamlit dashboard, ChurnGuard AI, turns a static notebook result into a reusable decision-support tool: KPI monitoring, individual and batch prediction, and risk segmentation are all available without writing code. The main limitation is precision (50.43%), meaning a meaningful share of flagged customers will not actually churn, which is identified as a direction for future refinement.

X. FUTURE WORK

1. **Precision-oriented threshold tuning and model refinement:** systematic threshold selection and hyperparameter tuning of the tree-based candidates to raise precision without materially sacrificing recall, as demonstrated on this dataset elsewhere [7].
2. **Explainability integration:** adding SHAP-based per-customer explanations to the prediction page, following directions suggested by related churn-explainability literature [8].
3. **Customer lifetime value integration:** combining churn probability with projected revenue to prioritise retention spend by expected loss rather than by risk score alone.
4. **Real-time and CRM deployment:** exposing the model as a REST service integrated

with CRM systems for continuous scoring and automated high-risk alerts.

ACKNOWLEDGEMENT

The author expresses sincere gratitude to Dr. B. Indira, Department of Master of Computer Applications (MCA), Chaitanya Bharathi Institute of Technology (CBIT), Hyderabad, for her valuable guidance and continuous support throughout this project. The author also thanks the faculty of the MCA department for their constructive feedback during the preparation of this manuscript.

REFERENCES

1. IBM / Kaggle, "Telco Customer Churn," [Online]. Available: <https://www.kaggle.com/datasets/blastchar/telco-customer-churn>.
2. F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
3. S. K. Wagh, A. A. Andhale, K. S. Wagh, J. R. Pansare, S. P. Ambadekar, and S. H. Gawande, "Customer churn prediction in telecom sector using machine learning techniques," *Results in Control and Optimization*, vol. 14, art. 100342, 2024.
4. Ahmad, A. Jafar, and K. Aljoumaa, "Customer churn prediction in telecom using machine learning in big data platform," *Journal of Big Data*, vol. 6, art. 28, 2019.
5. Towards Data Science, "Machine Learning Case Study: Telco Customer Churn Prediction," *Medium/TDS*, 2023.
6. S. K. Wagh and K. S. Wagh, "Customer churn prediction in telecom sector using machine learning techniques," *SSRN*, 2022.
7. S. M. Fujo, S. Subramanian, and M. A. Khder, "Customer churn prediction in telecommunication industry using deep learning," *Information Sciences Letters*, vol. 11, no. 1, pp. 185–198, 2022.
8. Explainable AI-driven customer churn prediction: a multi-model ensemble approach with SHAP-based feature analysis, *PMC*, 2025.
9. W. McKinney, "Data structures for statistical computing in Python," in *Proc. 9th Python in Science Conf. (SciPy 2010)*, 2010, pp. 56–61.