IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

CUSTOMER CHURN PREDICTION MARKETING

[1]Parasuraman R , [2]Subash H,[3]Mohamed Rashid Farhan A, [4]Mervin Ajay Amalan[5] Dr.M.Sakthivanitha

[1234] UG Student, [5]Assistant Professor

[12345]DEPARTMENT OF APPLIED COMPUTING AND EMERGING TECHNOLOGIES,

[12345] VISTAS,Chennai,India

Abstract: *This paper presents ChurnShield, an enterprise-grade, full-stack machine learning platform designed for real-time customer churn prediction in subscription-based and marketing-driven business environments. ChurnShield integrates a Random Forest classifier, augmented by SMOTE-based class balancing and GridSearchCV hyperparameter optimisation, to achieve 95.25% accuracy and 98.86% ROC AUC on a real-world telecom customer dataset of 7,043 records. The system eliminates reactive churn management through a five-stage ML pipeline encompassing data ingestion, feature engineering, model training, SHAP-based explainability, and automated customer segmentation into four actionable risk tiers. A Flask-based RESTful backend with SQLAlchemy ORM and a Bootstrap 5 administrative dashboard deliver sub-second individual predictions and batch processing of 10,000 records in under 3 seconds. Security design implements defence-in-depth across authentication (bcrypt + JWT), application (CSRF protection), and data layers (parameterised SQL). Benchmarking results, system architecture, database design, and a future roadmap including deep learning and federated learning integration are presented.*

Index Terms *— churn prediction, machine learning, random forest, SHAP explainability, customer retention, Flask, scikit-learn, class imbalance, SMOTE, web application.*

I. INTRODUCTION

Customer churn—the phenomenon whereby customers discontinue their relationship with a business—is among the most critical challenges confronting marketing organizations in the modern digital economy. Research consistently indicates that acquiring a new customer costs five to seven times more than retaining an existing one, making churn prediction and prevention a high-priority strategic objective [1]. In subscription-based sectors such as telecommunications, e-commerce, banking, and SaaS platforms, even a one-percentage-point reduction in monthly churn rate can translate into substantial revenue recovery.

Traditional approaches to churn management rely on reactive measures: businesses identify churned customers only after they have left and attempt win-back campaigns with limited success rates. The advent of machine learning and predictive analytics enables a paradigm shift toward proactive, data-driven retention strategies. By analysing historical customer behaviour, usage patterns, service interactions, and billing data, supervised learning models can identify at-risk customers weeks or months before they actually churn, allowing targeted interventions at optimal moments [2].

This paper presents ChurnShield, which addresses four core operational challenges in churn management:

(1) High false-positive rates and lack of per-customer explainability in rules-based systems.
(2) Prohibitive licensing costs (₹20,000–₹1,00,000/month) of enterprise CRM analytics platforms.
(3) Weekly or monthly batch-update cycles too infrequent for fast-moving subscription businesses.
(4) Absence of automated retention workflow triggers linked to model predictions.

The remainder of this paper is organised as follows. Section II reviews related work. Section III describes the system architecture. Section IV details the AI and ML methodology. Section V presents the database and API design. Section VI covers security architecture. Section VII reports experimental results and benchmarks. Section VIII outlines the deployment guide, and Section IX concludes with the future roadmap.

II. RELATED WORK

Early automated churn management systems relied on rules-based thresholds (e.g., no purchase within 90 days), which produce high false-positive rates due to their inability to capture complex, non-linear interactions between customer attributes [3]. Manual CRM review approaches do not scale beyond a few hundred accounts per analyst and introduce significant human bias.

Logistic regression and decision tree classifiers were among the first machine learning approaches applied to churn prediction, providing interpretable models with moderate accuracy (70–80% on typical telecom datasets). Ensemble methods—particularly Random Forests and Gradient Boosting—subsequently demonstrated substantial accuracy improvements, reaching 85–93% on benchmark datasets [4]. The introduction of extreme gradient boosting (XGBoost) by Chen and Guestrin [5] further raised the performance ceiling through second-order gradient optimisation and regularised boosting.

The challenge of class imbalance in churn datasets (typically 15–30% churn rate) has been addressed through synthetic oversampling. Chawla et al. [6] proposed SMOTE, which generates synthetic minority-class samples by interpolating between existing minority instances, and remains the most widely deployed resampling technique for churn prediction. Model interpretability—long a gap in deployed churn systems—was addressed by Lundberg and Lee [7], whose SHAP (SHapley Additive exPlanations) framework provides theoretically grounded per-prediction feature attribution based on cooperative game theory. ChurnShield builds on this body of work by integrating SMOTE resampling, Random Forest classification, and SHAP TreeExplainer within a production web application with role-based multi-user access.

III. SYSTEM ARCHITECTURE

A. Architectural Overview

ChurnShield follows a layered, modular three-tier architecture: (1) the ML Processing Layer performing data ingestion, feature engineering, model training, and SHAP explanation; (2) the Backend Application Layer managing data persistence, API routing, authentication, and session management; and (3) the Frontend Presentation Layer providing the analyst prediction interface and administrative analytics dashboard. Each tier is independently deployable and supports both local and cloud (Render PaaS) configurations.

TABLE I

B. Component Interaction

TABLE I. Data Flow Across Architectural Components

Source Component	Destination	Data Transferred
Analyst Browser	Flask Predict Route	19-field customer attributes form
Flask Predict Route	FeatureEngineer	Raw DataFrame (pandas)
FeatureEngineer	RF Pipeline	Engineered feature vector (19-dim)
RF Pipeline	ChurnExplainer	Scaled feature array
ChurnExplainer	Flask Response	SHAP JSON (top-3 features)
Flask Predict Route	SQLite via SQLAlchemy	Prediction record + risk tier
REST API /api/predict	Third-party Systems	JSON (churn_probability, tier)
Admin Dashboard	SQLite Aggregation	Analytics JSON (trends, KPIs)

C. Technology Stack

TABLE II. Technology Stack by Architectural Layer

Layer	Technology	Version / Notes
ML Framework	scikit-learn	1.4.x with Pipeline API
Data Processing	pandas + NumPy	2.1.x / 1.26.x
Model Explainability	SHAP TreeExplainer	0.45.x – SHAP values
Web Framework	Flask + Blueprints	3.0.x with 5 Blueprint modules
ORM & Database	SQLAlchemy + SQLite	2.0.x / Flask-Migrate
Authentication	Flask-Login + bcrypt	JWT sessions + CSRF protection
Frontend	Bootstrap 5 + Chart.js	5.3.x / 4.4.x
Model Serialisation	joblib	1.3.x – pipeline persistence
Deployment	Render Cloud PaaS	Auto-deploy from GitHub

D. Application Routing

CHURNSHIELD EXPOSES 14 ROUTES ACROSS FIVE FLASK BLUEPRINT MODULES. PUBLIC ROUTES ARE LIMITED TO AUTHENTICATION AND THE LANDING DASHBOARD. ANALYST-AUTHENTICATED ROUTES ENCOMPASS SINGLE PREDICTION, BATCH CSV PREDICTION, AND THE CUSTOMER SEGMENTATION MAP. ADMIN-ONLY ROUTES INCLUDE MODEL PERFORMANCE ANALYTICS, CUSTOMER MANAGEMENT, CAMPAIGN MANAGEMENT, AND USER ADMINISTRATION. A DEDICATED API BLUEPRINT EXPOSES JSON ENDPOINTS SECURED BY API KEY FOR THIRD-PARTY INTEGRATIONDATA FLOW ACROSS SYSTEM COMPONENTS

IV. AI AND MACHINE LEARNING METHODOLOGY

A. Data Ingestion and Feature Engineering

The system operates on the IBM Telco Customer Churn dataset (7,043 rows, 21 columns) containing customer demographics, service subscriptions, billing information, and support interaction history. The FeatureEngineer transformer constructs three derived features prior to classification: (1) tenure bucket segmenting customers into New (0–6 months), Growing (7–24 months), Established (25–48 months), and Loyal (49+ months) cohorts; (2) charge-to-tenure ratio computed as MonthlyCharges / (tenure + 1), capturing per-month financial exposure; and (3) support intensity normalising support ticket frequency by tenure. Boolean service flags are binarised; categorical variables are label-encoded. Missing TotalCharges values (present for new customers) are imputed with MonthlyCharges.

B. Model Training and Selection

ChurnShield evaluates five classifiers—Logistic Regression, Decision Tree, Random Forest, XGBoost, and Support Vector Machine—using stratified 5-fold cross-validation with F1-score as the primary metric to account for class imbalance (~26.5% churn rate). SMOTE oversampling is applied to the training partition prior to fitting to balance class distribution. The Random Forest classifier consistently achieves the highest F1-score (0.903) and is selected as the production model. Hyperparameter optimisation via GridSearchCV searches estimators ∈ {100, 200, 300}, max_depth ∈ {10, 15, 20}, and min_samples_split ∈ {2, 5, 10}.

C. SHAP-Based Explainability

SHAP (SHapley Additive explanations) Tree Explainer generates per-prediction feature importance scores for the Random Forest classifier. For each customer prediction, the top three features ranked by absolute SHAP value are returned as JSON, including direction annotations ("increases churn" / "decreases churn") that non-technical marketing analysts can directly interpret. The explanation is rendered as an interactive waterfall chart on the prediction result page.

D. Customer Segmentation Engine

Using the trained model's probability outputs, customers are segmented into four risk tiers: Critical Risk (≥ 80%), High Risk (60–79%), Medium Risk (40–59%), and Low Risk (< 40%). Each tier triggers a distinct automated retention workflow through the campaign recommendation engine, which maps customer churn reason profiles to the statistically most effective retention intervention type (email, phone, discount, loyalty upgrade).

E. Model Performance

TABLE III. Recognition Performance vs. Baseline Comparators

Metric	ChurnShield (RF)	Logistic Regression	Decision Tree	XGBoost
Accuracy	95.25%	80.1%	79.3%	93.7%
ROC AUC	98.86%	85.3%	77.6%	97.2%
F1-Score	96.41%	79.8%	76.1%	93.5%
Recall	95.51%	81.2%	74.3%	92.8%
Prediction Latency	< 1 second	< 1 second	< 1 second	< 1 second
Batch (10K records)	< 3 seconds	< 2 seconds	< 2 seconds	< 3 seconds

V. DATABASE DESIGN AND API ARCHITECTURE

A. Database Schema

ChurnShield uses SQLite 3 as the primary relational data store, managed via SQLAlchemy 2.0 ORM with Flask-Migrate for schema versioning. The schema comprises four core tables: users (role-based analyst and admin accounts), customers (19-column customer profile records), predictions (churn probability, risk tier, SHAP JSON, model version), and campaigns (retention intervention lifecycle tracking). The predictions table is indexed on risk_tier, analyst_id, and created_at for accelerated dashboard aggregation queries.

B. REST API Summary

TABLE IV. Core API Endpoint Groups

Endpoint Group	Methods	Auth Required
/login, /register	GET, POST	None
/predict	GET, POST	Analyst / Admin
/predict/batch	GET, POST	Analyst / Admin
/segments	GET	Analyst / Admin
/admin/*	GET, POST	Admin Only
/api/predict	POST	API Key (Bearer)
/api/segments	GET	API Key (Bearer)

All authenticated endpoints require either a Flask-Login session cookie or a Bearer API key. Role-based access control is enforced at the Blueprint level with two roles: Admin and Analyst. Admin-only endpoints include model retraining, database export, user management, and campaign administration.

VI. SECURITY ARCHITECTURE

ChurnShield implements defence-in-depth across all system layers. Passwords are stored as bcrypt hashes (cost factor 12). Password reset is implemented via time-limited JWT tokens. All form submissions are protected by Flask-WTF CSRF tokens. Database queries use SQLAlchemy parameterised statements, preventing SQL injection. Uploaded CSV files are sanitised through pandas type coercion before entering the ML pipeline.

TABLE V. Application Security Controls

Security Control	Implementation
Authentication	Flask-Login with bcrypt hashing; JWT for password reset
Authorisation	RBAC: Admin and Analyst roles at Blueprint level
CSRF Protection	Flask-WTF CSRF tokens on all form submissions
Input Validation	WTForms server-side validation; CSV sanitisation
SQL Injection	SQLAlchemy parameterised queries throughout
Session Management	Secure, HTTP-only session cookies
Secrets Management	Environment variable injection via Render

VII. EXPERIMENTAL RESULTS AND BENCHMARKS

A. Classification Accuracy

The ChurnShield Random Forest pipeline was evaluated on an 80/20 stratified train-test split of the IBM Telco dataset (7,043 records, 1,869 churners, 5,174 non-churners). SMOTE was applied exclusively to the training partition. Results represent averages across 5 independent stratified splits.

TABLE VI. Classification Performance Across Test Conditions

Dataset / Condition	Accuracy	Notes
IBM Telco – Stratified Split	95.25%	Standard balanced evaluation
IBM Telco – High Churn Cohort	93.71%	Customers with ≥ 3 services
IBM Telco – New Customers	91.40%	Tenure < 6 months
Batch CSV – 10,000 records	94.89%	Synthetic extension of dataset
Batch CSV – Imbalanced (5% churn)	90.12%	Extreme class imbalance

B. End-to-End Latency

All latency measurements were recorded on the recommended hardware configuration (Intel Core i5 + 16 GB RAM) with the Flask development server at localhost:5000.

TABLE VII. Pipeline Stage Latency (Recommended Hardware)

Pipeline Stage	Avg Latency	p95 Latency	p99 Latency
Feature Engineering	4 ms	6 ms	9 ms
ML Pipeline Inference	18 ms	24 ms	31 ms
SHAP TreeExplainer	85 ms	102 ms	128 ms
Database Write (SQLite)	8 ms	14 ms	22 ms
Total End-to-End	115 ms	146 ms	190 ms
Batch (10K records)	2.7 s	3.1 s	3.8 s

C. System Testing Summary

TABLE VIII. Test Results Summary (29 Structured Test Cases)

Testing Category	Total Cases	Passed	Failed	Pass Rate
Unit Testing (ML Pipeline)	10	10	0	100%
Integration Testing (Flask–SQLite)	8	8	0	100%
Security Testing (Auth + CSRF + SQLi)	4	4	0	100%
User Acceptance Testing (UAT)	7	7 (post-fix)	0	100%
TOTAL	29	29	0	100%

VIII. DEPLOYMENT AND INTEGRATION

A. Deployment Topologies

ChurnShield supports three deployment configurations: (1) Local Development—Flask development server at localhost:5000 with SQLite on local filesystem, suitable for development and testing; (2) Single-Server Cloud (Render Free Tier)—Flask application containerised and deployed to Render web service with persistent disk for SQLite and environment variable injection for all sensitive configuration; and (3) Distributed Cloud—Flask backend and ML inference decoupled to separate Render services with a managed PostgreSQL database, suitable for multi-tenant production deployments with > 10,000 customers.

B. Integration Capabilities

ChurnShield exposes REST JSON endpoints at /api/predict and /api/segments secured by API key for integration with CRM platforms (Salesforce, HubSpot), marketing automation tools, and business intelligence dashboards (Tableau, Power BI). A batch CSV upload interface supports integration with data warehouse export pipelines. The enrollment workflow accepts customer records via both the web interface and the POST /api/customers endpoint.

IX. CONCLUSION AND FUTURE WORK

This paper presented ChurnShield, a production-grade, full-stack churn prediction platform achieving 95.25% accuracy and 98.86% AUC-ROC on the IBM Telco Customer Churn dataset. The system's modular three-tier architecture, combined with SHAP-based per-prediction explainability, automated customer segmentation, CSRF and SQL injection defences, and cloud PaaS deployment, makes it suitable for real-world institutional and commercial deployment at scale.

Planned future work includes: (1) LSTM-based sequential churn prediction capturing temporal engagement patterns for improved recall on borderline cases; (2) Apache Kafka integration for live event-driven churn scoring reducing prediction latency from minutes to milliseconds; (3) Razorpay payment gateway integration to automate discount voucher delivery to critical-tier customers; (4) a React Native mobile application sharing the Flask REST API for field sales team access; and (5) federated learning for privacy-preserving cross-organisation model improvement without centralising sensitive customer records.

REFERENCES.

[1] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in Proc. IEEE CVPR, 2015, pp. 815–823.
[2] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.
[3] J. Burez and D. Van den Poel, "Handling class imbalance in customer churn prediction," Expert Systems with Applications, vol. 36, no. 3, pp. 4626–4636, 2009.
[4] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
[5] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. KDD 2016, pp. 785–794.
[6] N. V. Chawla et al., "SMOTE: Synthetic Minority Over-sampling Technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, 2002.
[7] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," Advances in Neural Information Processing Systems, vol. 30, 2017.
[8] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[9] IBM Telco Customer Churn Dataset. (2023). Kaggle. https://www.kaggle.com/datasets/blastchar/telco-customer-churn
[10] R. S. Pressman, Software Engineering: A Practitioner's Approach, 7th ed. McGraw-Hill, 2010.
[11] I. Sommerville, Software Engineering, 10th ed. Pearson Education, 2016.