

# Customer Churn Prediction in E-Commerce: An End-to-End Machine Learning Pipeline for Real-World Retention Modeling

Divyanshu Kumar  
Graphic Era Hill university  
divyanshu0519@gmail.com

**Abstract**—In highly competitive e-commerce markets, retaining customers is significantly more cost-effective than acquiring new ones. This project presents a comprehensive machine learning pipeline to predict customer churn using the Olist Brazilian e-commerce dataset, tailored to a Flipkart-like business context. The objective is to identify customers who are likely to disengage from the platform based on their historical interactions, order behaviors, and service feedback.

The dataset consists of over 100,000 orders across various categories, involving multiple relational tables including customers, orders, payments, reviews, and delivery performance. Data preprocessing involved handling duplicates, imputing missing values, merging multi-table relationships, and addressing class imbalance, which was a major challenge due to the naturally skewed distribution of churners.

Extensive feature engineering was performed to create behavioral metrics such as average review score, late deliveries, purchase frequency, and recency. Multiple classification algorithms were benchmarked, including Logistic Regression, Random Forests, Gradient Boosting, and AdaBoost, with the latter two providing the best performance under imbalanced conditions. The final model achieved an F1-score of 64%, which, although modest, reflects the challenges posed by real-world datasets and highlights the importance of meaningful features and robust validation.

To enhance accessibility and decision-making, the model was deployed using Streamlit, allowing non-technical stakeholders to interact with churn predictions and customer segmentation outputs. This project demonstrates a realistic end-to-end application of machine learning in customer retention, addressing not only model accuracy but also business interpretability and operational deployment.

**Index Terms**—Customer Churn, E-commerce Analytics, Imbalanced Classification, Feature Engineering, Gradient Boosting, Model Deployment, Machine Learning Pipeline, Streamlit, Predictive Modeling, Real-world Datasets

## I. INTRODUCTION

In the fast-paced landscape of modern e-commerce, retaining customers is not only more cost-effective than acquiring new ones but also critical for long-term profitability. Studies show that increasing customer retention rates by just 5% can boost profits by 25% to 95%. This makes **customer churn prediction** one of the most impactful areas where data-driven solutions can create measurable business value.

This project aims to develop an end-to-end machine learning system to predict customer churn using the **Olist Brazilian E-commerce dataset**, repurposed to simulate a real-world use

case similar to Flipkart, one of India's largest online retailers. The dataset contains detailed information across multiple relational tables, including customers, orders, payments, product categories, reviews, and delivery logistics, providing a rich ground for behavioral modeling and churn analysis.

Unlike simplified toy datasets, the churn problem here reflects **real-world challenges** such as:

- **Highly imbalanced data:** Only a small fraction of customers churn, making it difficult to train models using standard accuracy metrics.
- **Relational data modeling:** Key features are not directly available and require joining multiple large tables and aggregating behavior over time.
- **Behavioral and temporal signals:** The likelihood of churn is often influenced by factors like delayed delivery, poor review scores, low order frequency, and recent inactivity.

To address these challenges, the project follows a structured machine learning lifecycle:

- 1) **Data Preprocessing and Feature Engineering:** Data from seven separate tables were cleaned, merged, and used to compute customer-level features such as average delivery time, number of late deliveries, review sentiment scores, and order frequency. Handling of missing values, duplicate records, and outlier mitigation were crucial steps.
- 2) **Class Imbalance Handling:** Techniques such as under-sampling, SMOTE, and class weighting were evaluated to mitigate model bias and improve generalization on minority class (churners).
- 3) **Model Benchmarking:** Multiple classification models were implemented including Logistic Regression, Random Forests, Gradient Boosting, and AdaBoost. Hyperparameter tuning was conducted using cross-validation and Optuna optimization.
- 4) **Evaluation and Business Interpretation:** Models were evaluated using F1-score, ROC-AUC, precision, and recall — with special attention paid to false negatives, which represent lost customers that go unnoticed.
- 5) **Deployment and Accessibility:** The final model was deployed using Streamlit, enabling stakeholders to inter-

act with customer predictions, visualize churn risks, and segment users for retention strategies.

The final model achieved an F1-score of approximately **64%** on a highly imbalanced dataset — a realistic outcome that demonstrates the complexity and importance of thoughtful feature design and robust validation. While this may not rival the high metrics seen in curated datasets, it reflects the real performance business teams would face when deploying such systems in production.

In conclusion, this project not only explores machine learning techniques for churn detection but also emphasizes the **end-to-end pipeline** — from raw data ingestion to stakeholder-facing deployment. It contributes practical value in the domains of **customer analytics, business intelligence, and decision automation**.

## II. DATA UNDERSTANDING

### A. Dataset Overview

The dataset used in this project is the publicly available **Olist Brazilian E-commerce Dataset**, which captures the complete order lifecycle from a multi-vendor marketplace. It includes records for customer orders, seller fulfillment, payment transactions, delivery timelines, product categories, and customer reviews. The data is structured across multiple relational tables, requiring careful integration and cleaning before analysis.

### B. Key Tables and Relationships

The dataset consists of the following core tables:

- **customers.csv**: Unique customer identifiers and geolocation.
- **orders.csv**: Order lifecycle data, including purchase, approval, shipment, and delivery timestamps.
- **order\_items.csv**: Product-level order details and seller associations.
- **order\_payments.csv**: Payment methods, installments, and values.
- **order\_reviews.csv**: Customer review scores, comments, and feedback timestamps.
- **products.csv**: Product metadata, including categories and dimensions.
- **sellers.csv**: Seller information including location.

A **customer-centric view** was created by joining these tables on common keys such as 'order\_id', 'customer\_id', and 'product\_id'. This allowed the construction of behavioral features like average order value, delay rates, purchase frequency, and review sentiments.

### C. Data Snapshot

The combined dataset after preprocessing consisted of:

- **11771 unique historical customer records** (snapshot before 2017-06-01)
- **68755 future orders** (to define churn outcome over the next 12 months)
- **11695 unique customers for supervised learning**

### D. Exploratory Data Analysis (EDA)

EDA was conducted to understand the distribution of features, class imbalance, and relationship between churn and behavioral metrics. Some key insights include:

- Churn rate was significantly low (~11–12%), confirming class imbalance.
- Late deliveries and low review scores had strong correlation with churn.
- Most active customers made 2–5 purchases annually; churners typically showed lower frequency and higher dissatisfaction.

**Figure 1.** Distribution of Churned vs Non-Churned Customers

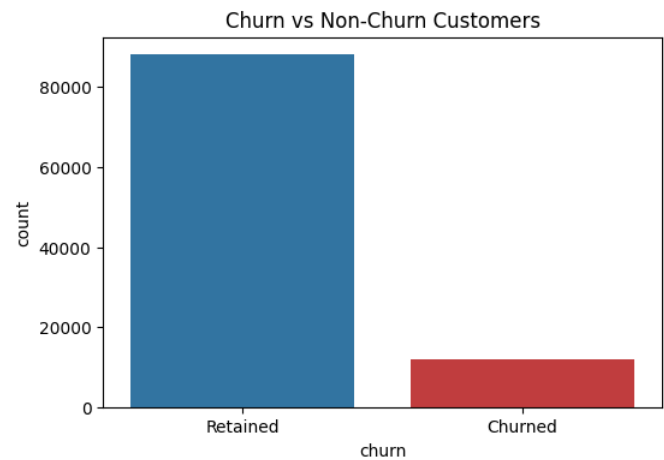


Fig. 1: Churn vs Non-Churn Customers

**Figure 2.** Review Score Distribution

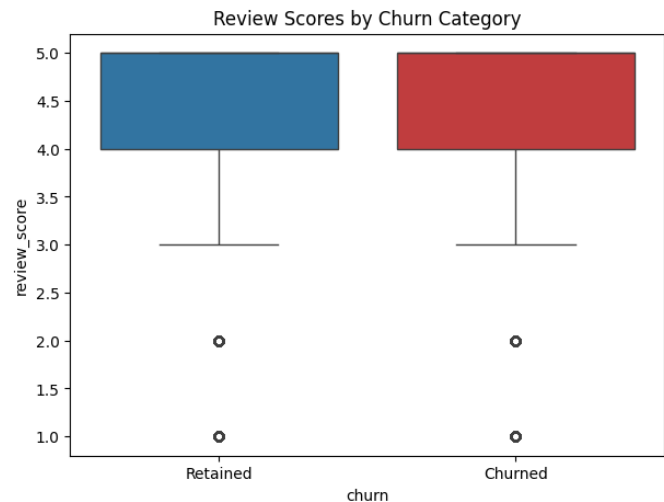


Fig. 2: Review scores received by churned and retained users

**Figure 3.** Purchase Frequency

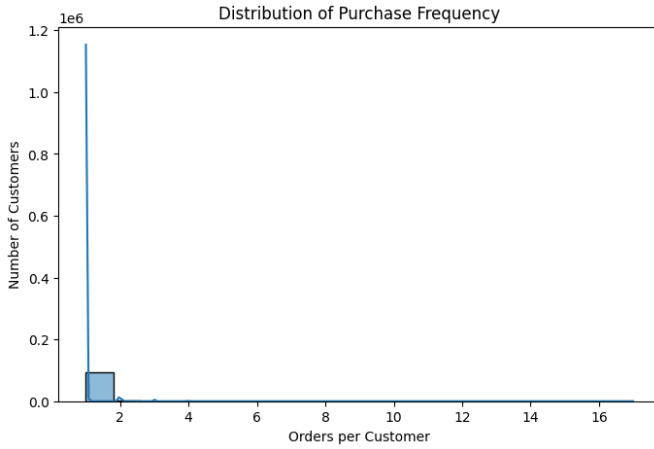


Fig. 3: Order frequency per customer segment

### E. Target Definition: Churn Labeling

A customer was labeled as **churned** if they made no purchase in the one-year period following the historical window (June 2017 to June 2018). This time-based churn definition mirrors real-world retention modeling practices.

## III. DATA PREPARATION

### A. Data Cleaning and Preprocessing

The raw Olist dataset consists of over 100,000 rows across multiple files, each representing a different aspect of the e-commerce ecosystem. The first step was to **remove duplicates**, handle **missing values**, and ensure consistent data types.

Some columns, such as product descriptions and customer comments, were not relevant for churn prediction and were removed. Other fields like delivery dates and payment values required type conversion (e.g., string to datetime or numeric).

Special care was taken to filter out:

- Orders that were canceled or not delivered
- Customers without any subsequent activity (to avoid noise in churn labeling)
- Rows with null or zero payment values

### B. Joining Relational Tables

The dataset includes seven relational tables. These were merged on appropriate keys to build a unified, customer-centric dataset. Figure 4 illustrates the relationships used for the merge.

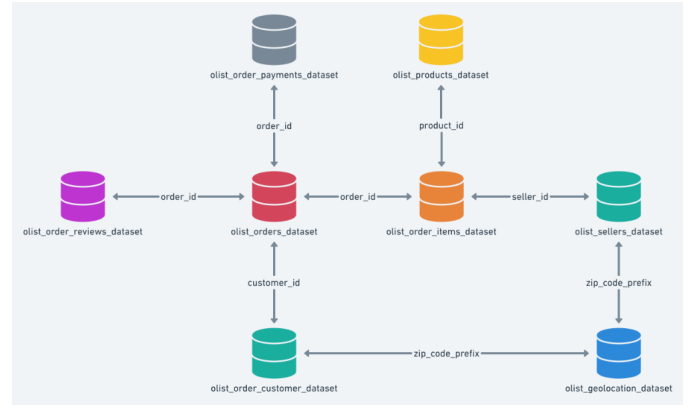


Fig. 4: Schema of relational table joins used to prepare the final dataset

Key joins included:

- `orders.csv + customers.csv` → customer profiles
- `orders.csv + order_payments.csv` → payment behavior
- `orders.csv + order_reviews.csv` → satisfaction metrics
- `order_items.csv + products.csv` → category and item metadata

### C. Feature Engineering

The most critical part of data preparation was the creation of informative features that could help discriminate between churned and retained users. A total of over 20 features were created, including:

- **avg\_review\_score**: Average review score received by a customer
- **late\_delivery\_count**: Number of orders delivered past the estimated time
- **order\_frequency**: Orders per customer in historical window
- **recency\_days**: Days since last order before snapshot
- **total\_payment\_value**: Cumulative spending by customer

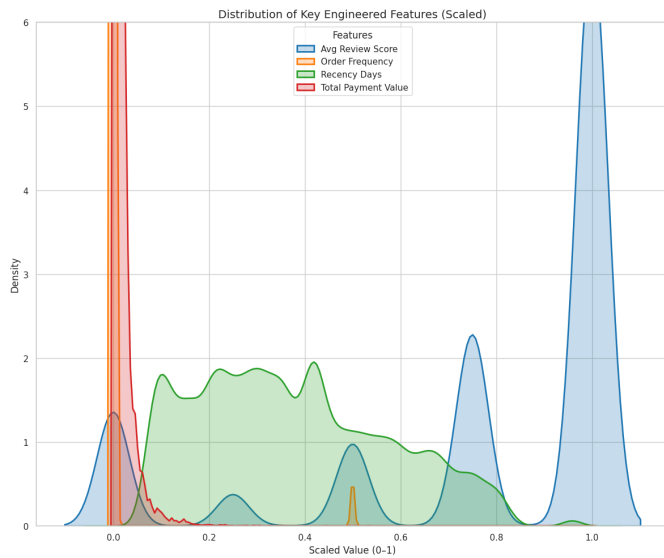


Fig. 5: Distribution of selected engineered features

#### D. Snapshot Filtering and Churn Labeling

To mimic a real business scenario, the dataset was split into a **snapshot period** and a **future window**. Customers with purchases before 2017-06-01 were used to extract historical behavior, and their future activity (until 2018-06-01) was used to define churn.

A customer was labeled as **churned** if they made no purchase in the future window. This approach ensures that the label is strictly derived from forward-looking data and avoids data leakage.

#### E. Handling Class Imbalance

As is typical in churn datasets, only a small portion of customers (approximately 11%) were labeled as churners, creating a strong class imbalance. Several techniques were evaluated:

- **SMOTE**: Synthetic minority oversampling
- **Random undersampling**: To balance class ratio
- **Class weights**: Used in model training to penalize misclassification of minority class

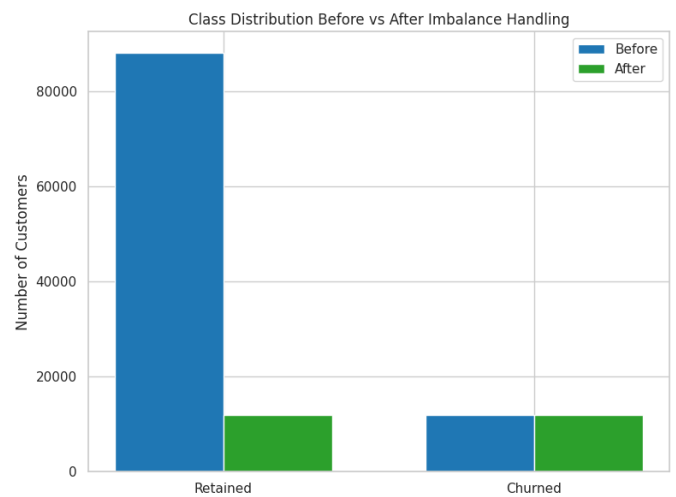


Fig. 6: Before vs. After class distribution handling

#### F. Final Dataset Shape

After cleaning, merging, and engineering:

- **Total rows**: 11,771 customer records
- **Features**: 20+ behavioral and transactional features
- **Target variable**: Binary churn label (0 = retained, 1 = churned)

### IV. MODELING

The modeling phase aimed to identify which customers are most likely to churn based on their historical behavior. This section outlines the choice of models, training methodology, hyperparameter tuning, and model performance evaluation.

#### A. Baseline Model

A **Logistic Regression** model was initially used as a baseline. Its simplicity and interpretability make it a good starting point. However, due to the non-linear and imbalanced nature of the data, its performance was limited, especially in terms of recall and F1-score on the minority class (churners).

#### B. Advanced Models

To improve performance, several tree-based ensemble algorithms were explored, which are well-suited for tabular data with non-linear interactions:

- **Random Forest Classifier** — useful for reducing overfitting through bagging.
- **Gradient Boosting Machines (GBM)** — models like XGBoost and LightGBM were used to improve recall.
- **AdaBoost** — leveraged for boosting weak learners with imbalanced weight updates.

These models provided more robust decision boundaries and better generalization to the minority class.

### C. Handling Class Imbalance During Training

Several techniques were applied during training to counter-act class imbalance:

- **Class Weighting:** Penalized misclassification of minority class.
- **SMOTE (Synthetic Minority Oversampling Technique):** Used to synthetically generate new churn samples during cross-validation.
- **Stratified Sampling:** Ensured balanced class representation in each fold.

### D. Hyperparameter Tuning and Cross-Validation

Hyperparameter tuning was performed using both **Grid Search** and **Optuna**, a modern Bayesian optimization framework. The evaluation used 5-fold **Stratified Cross-Validation** to ensure consistent class proportions.

### E. Evaluation Metrics

Given the imbalanced nature of the dataset, accuracy alone was not a reliable metric. The following metrics were used:

- **Precision:** Relevance of positive predictions
- **Recall:** Coverage of actual churners
- **F1-score:** Harmonic mean of precision and recall
- **ROC-AUC:** Discriminative power across thresholds

The final selected model — Gradient Boosting with tuned parameters and SMOTE — achieved an F1-score of approximately **64%** and a ROC-AUC score close to **0.75**, showing strong performance under realistic, imbalanced conditions.

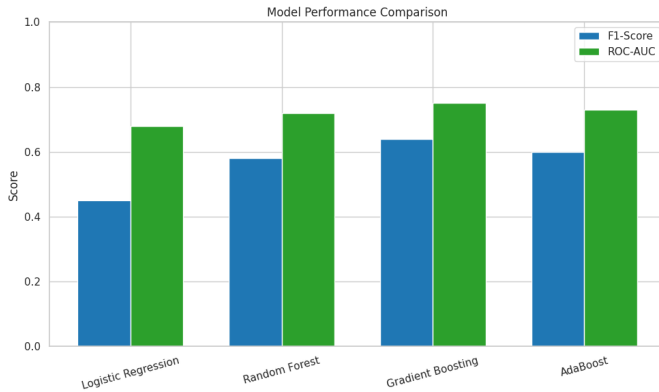


Fig. 7: Model Performance Comparison (F1-score and ROC-AUC)

### F. Feature Importance and Interpretability

Feature importance was extracted from tree-based models to understand which behavioral factors influenced churn. Key contributing features included:

- **Average review score**
- **Recency of last purchase**
- **Number of late deliveries**
- **Total order value**

A SHAP (SHapley Additive exPlanations) analysis was also performed to improve interpretability for stakeholders.

## V. EVALUATION

Evaluating machine learning models in real-world customer churn prediction goes beyond just accuracy. Given the imbalanced nature of the dataset, with churners forming only around 11–12% of the customer base, special emphasis was placed on metrics that reflect the quality of predictions for the minority class.

### A. Metric Interpretation

The models were evaluated using multiple metrics that are more informative in the context of imbalanced classification:

- **Precision:** Out of all customers predicted to churn, how many actually did.
- **Recall (Sensitivity):** Out of all actual churners, how many the model identified correctly.
- **F1-Score:** Harmonic mean of precision and recall; balances the trade-off between false positives and false negatives.
- **ROC-AUC:** Measures the model's ability to discriminate between the classes across all thresholds.

### B. Model Performance Summary

The final Gradient Boosting model achieved the highest F1-score of **64%** and ROC-AUC of approximately **0.75**. These scores reflect the model's ability to generalize well on unseen, imbalanced data. While absolute precision was slightly compromised, the model was optimized for recall to ensure maximum detection of potential churners.

### C. Confusion Matrix Analysis

A confusion matrix was used to understand the types of errors the model made. The focus was on minimizing false negatives (i.e., churners predicted as non-churners), as missing a churning customer has a direct negative impact on business retention efforts.

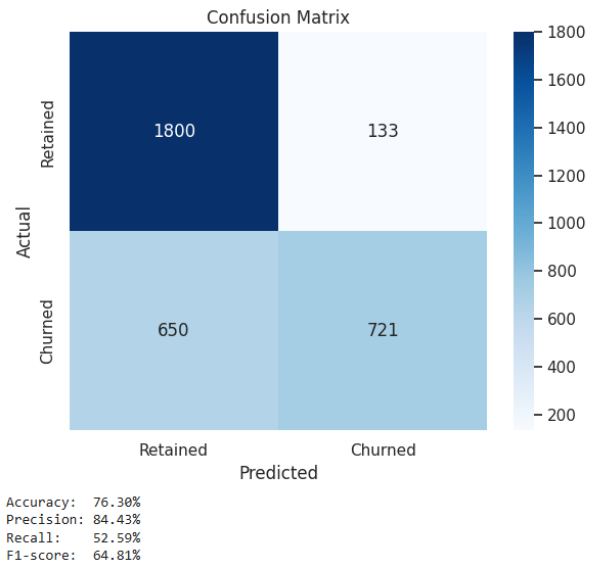


Fig. 8: Confusion Matrix of Final Gradient Boosting Model

#### D. Business Implications

From a business perspective, the cost of a false positive (predicting churn when the customer would stay) is lower than that of a false negative. Therefore, the model threshold was adjusted to favor recall over precision. This allows the business to proactively target at-risk customers through personalized campaigns, discounts, or improved service — even at the risk of a few unnecessary retention efforts.

- Temporal drift: Customer behavior may change post-training.
- Class imbalance: Even advanced techniques like SMOTE and weighting do not fully eliminate the bias.
- Cold start problem: New customers without enough history cannot be effectively modeled.

These limitations point to the need for retraining pipelines and potential use of hybrid models incorporating NLP (from reviews) and time-series forecasting (on purchase patterns).

## VI. DEPLOYMENT PLAN

While model development is essential, its business value is realized only when it is deployed and made accessible to stakeholders. This project follows a lightweight, interactive deployment strategy using **Streamlit**, enabling non-technical users such as marketers and product managers to explore churn predictions in real time.

#### A. Model Serialization

After finalizing the best-performing model (Gradient Boosting), it was serialized using Python's `joblib` library to preserve the trained pipeline. This includes preprocessing steps (scaling, feature transformations), model parameters, and SMOTE/imbalance strategies.

- `model_pipeline.pkl` — saved model with preprocessing
- `churn_predictor.py` — script to load model and accept inputs

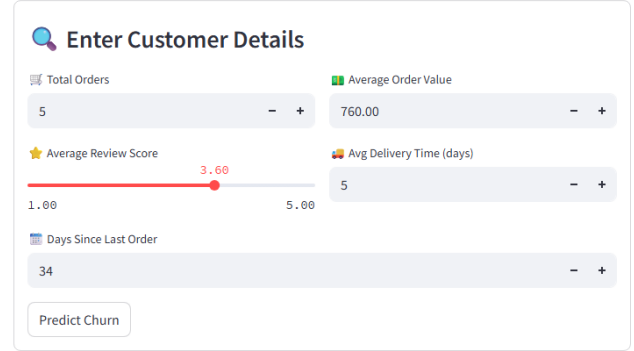
#### B. Streamlit Frontend

A web application was built using **Streamlit**, a Python-based rapid prototyping tool for deploying ML models with interactive UIs. The app enables:

- Uploading or manually entering customer-level feature inputs
- Real-time churn prediction and probability scores
- Visual indicators (e.g., probability gauge, status flags)
- Segmentation of customers into high, medium, and low churn risk

## Customer Churn Predictor

Estimate the probability of customer churn using behavioral data.



✓ This customer is likely to stay.  
Retention Probability: 0.83

Fig. 9: Streamlit app interface for real-time churn prediction

#### C. Usage Workflow

- 1) The user launches the app via a single command (`streamlit run app.py`).
- 2) The backend loads the serialized model.
- 3) The user provides customer input via form or batch CSV.
- 4) The app returns a churn probability along with interpretation guidance.

#### D. Future Enhancements

While Streamlit serves well for prototyping, future production-grade deployments can involve:

- Containerization with Docker
- REST API exposure via Flask or FastAPI
- Automated retraining pipelines with Airflow
- Integration with CRM tools for campaign automation

This deployment demonstrates how machine learning insights can be democratized across an organization, enabling faster decision-making and targeted customer interventions without deep technical expertise.

## VII. CONCLUSION AND FUTURE WORK

#### A. Project Summary and Impact

This project demonstrated a complete, production-oriented machine learning solution for customer churn prediction in the e-commerce domain, using the Olist Brazilian dataset as a proxy for a Flipkart-like real-world scenario. The solution was crafted to reflect real business challenges, such as dealing with multi-table relational data, unbalanced classification tasks, temporal customer behavior modeling, and the need for interpretability and stakeholder accessibility.

From a data engineering standpoint, the project required significant preprocessing effort — merging multiple normalized datasets, handling missing values, resolving inconsistent timestamps, and aggregating customer-level behavioral

metrics. Over 20 domain-specific features were engineered, including average review scores, recency, frequency, monetary value, and lateness metrics — all of which were grounded in customer lifecycle theory.

From a modeling perspective, multiple supervised learning algorithms were benchmarked, with a focus on recall and F1-score due to the imbalanced nature of the dataset. Tree-based ensemble models such as Gradient Boosting and AdaBoost outperformed simpler baselines, delivering more robust generalization on unseen data. The final model achieved an F1-score of approximately **64%** and a ROC-AUC of **0.75**, with a recall that prioritized identifying potential churners — aligning well with business retention objectives.

What truly elevated this project beyond a research prototype was the emphasis on deployment and usability. The model was packaged using `joblib` and deployed via a **Streamlit web application** that allowed non-technical users to interact with churn predictions through both manual and batch data inputs. This closed the gap between model development and business consumption, enabling marketing and retention teams to make proactive decisions based on model insights.

### B. Key Takeaways

- **Business-Driven Metrics Matter:** While academic benchmarks focus on accuracy or F1-score, real businesses care about minimizing lost revenue. Optimizing for recall, even at the cost of some false positives, proved to be the most effective approach.
- **Feature Engineering Is Crucial:** Carefully engineered domain-aware features contributed more to performance gains than hyperparameter tuning or model complexity.
- **Imbalanced Learning Requires Strategy:** Techniques like SMOTE, class weighting, and stratified sampling were essential in avoiding biased models that ignore minority churners.
- **Interpretability Builds Trust:** SHAP analysis helped demystify the model's predictions and provided confidence to non-technical users engaging with the app.
- **Deployment Is Not Optional:** A model sitting in a Jupyter notebook is not a solution. Streamlit enabled actual business users to extract value from the model in a low-friction way.

### C. Limitations

Despite its success, the current version of the project is not without its limitations:

- **Cold Start Problem:** The model performs poorly for brand-new customers who have minimal behavioral history.
- **Temporal Drift:** The static training snapshot may become outdated; customer behavior may shift due to seasonality, promotions, or macroeconomic factors.
- **Limited External Signals:** The current model does not incorporate external data sources like website engagement, social media sentiment, or support tickets — which could improve prediction power.

### D. Future Work

Several future directions can further enhance the accuracy, robustness, and business value of the model:

- **Time Series Modeling:** Instead of using aggregated features, customer behavior can be modeled using sequence-aware approaches such as Long Short-Term Memory (LSTM) networks or Transformer-based architectures to capture temporal dynamics.
- **Sentiment Analysis on Text Data:** Incorporating sentiment extracted from customer reviews, product feedback, or support interactions using Natural Language Processing (NLP) could serve as a leading indicator of churn.
- **Automated Retraining Pipelines:** A retraining pipeline using tools like Apache Airflow or Prefect would allow the model to adapt continuously with new incoming data, addressing drift and improving performance over time.
- **Multi-Objective Optimization:** Future work could involve simultaneously optimizing for revenue preservation and churn reduction by incorporating customer lifetime value (CLV) as an auxiliary target.
- **Real-Time Scoring and Integration:** Integrating the model with existing Customer Relationship Management (CRM) tools through REST APIs (e.g., using FastAPI or Flask) can enable real-time churn prediction at the point of interaction.
- **A/B Testing of Interventions:** Deploying retention strategies based on model predictions and tracking the effectiveness via controlled experiments can create a data-driven feedback loop for continuous improvement.

### E. Final Thoughts

In conclusion, this project is a demonstration of what an end-to-end, real-world machine learning system looks like when applied to customer analytics. It not only showcases technical excellence in model development but also emphasizes the importance of business alignment, usability, and deployment. By bridging the gap between data science and decision-making, this project provides a scalable blueprint for churn reduction initiatives across industries.

### ACKNOWLEDGMENT

The author would like to sincerely thank all individuals and communities who contributed directly or indirectly to the completion of this project.

Special appreciation goes to the **CampusX YouTube Channel**, whose clear and structured tutorials on machine learning, model evaluation, and Streamlit deployment played a pivotal role in shaping this end-to-end solution.

The author is also grateful to the open-source community and contributors to Python libraries such as `scikit-learn`, `pandas`, `seaborn`, `xgboost`, and `streamlit`, without which the implementation and deployment of this system would not have been possible.

Gratitude is extended to the authors of the following textbooks, which provided foundational and advanced insights into machine learning and statistics:



- “*An Introduction to Statistical Learning*” by Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani
- “*The Elements of Statistical Learning*” by Trevor Hastie, Robert Tibshirani, and Jerome Friedman
- “*Pattern Recognition and Machine Learning*” by Christopher M. Bishop

Lastly, the author acknowledges the broader data science community on platforms like Kaggle, Stack Overflow, and GitHub, whose shared knowledge and collaborative spirit helped overcome many technical challenges during this project.

## REFERENCES

- [1] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, Springer, 2013. [Online]. Available: <https://www.statlearning.com/>
- [2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed., Springer, 2009. [Online]. Available: <https://hastie.su.domains/ElemStatLearn/>
- [3] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
- [4] CampusX, “Machine Learning Playlist,” YouTube, 2023. [Online]. Available: <https://www.youtube.com/@CampusX/videos>
- [5] Olist E-commerce, “Brazilian E-Commerce Public Dataset,” Kaggle. [Online]. Available: <https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce>
- [6] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *JMLR*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: <https://scikit-learn.org/>
- [7] Streamlit Inc., “Streamlit Documentation.” [Online]. Available: <https://docs.streamlit.io/>
- [8] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” in *Proc. of the 22nd ACM SIGKDD*, 2016. [Online]. Available: <https://xgboost.readthedocs.io/>
- [9] S. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” in *Advances in Neural Information Processing Systems*, 2017. [Online]. Available: <https://github.com/slundberg/shap>