Churn Prediction for Sumauto Advertisers

Zrive Applied Data Science

Final project with Komorebi AI

Authors: Alejandro Marcos, Álvaro Ortega, Eduardo Ezponda and Maick Mosquera

Mentor: Guillermo Santamaría

Presentation date: Madrid, 22/05/2025

Index

1. Introduction and problem statement

1.1 Business context

Sumauto is a digital group that operates several classified advertisement portals specialized in the sale and rental of vehicles. Its business model revolves around attracting online traffic and directing it toward listings published by various merchants across its platforms. By optimizing user engagement and maximizing visibility for vehicle listings, Sumauto serves as a crucial intermediary between car dealerships and potential buyers or renters.

As part of a collaboration between Zrive and the Madrid-based company Komorebi, a real-world data science project was proposed to Zrive's students from the first-semester 2025 cohort. The focus of this project was a business case previously addressed by Komorebi for an external client, Sumauto. However, the initial approach to the project had been limited in scope and lacked depth, prompting a renewed opportunity for the Zrive students to carry out a more comprehensive analysis.

This analysis was undertaken by a dedicated team of Zrive students, including Alejandro Marcos, Álvaro Ortega, Maick Mosquera and Eduardo Ezponda. The core objective of this initiative was to develop a predictive model capable of identifying customer churn based on the performance and behavioural patterns of advertisers on Sumauto's platforms. The analysis was conducted using anonymized and scaled metrics to ensure data privacy and safeguard sensitive information.

1.2 Definition of churn in this context

In this project, churn refers to the definitive loss of an advertiser from the Sumauto platform. There are two main ways in which churn is identified:

1. Explicit Churn via Withdrawals

Advertiser churn is first determined using a dedicated withdrawal dataset. A withdrawal is considered definitive if it meets all the following conditions:

- *withdrawal_type* is *'TOTAL'*
- *withdrawal_status* is not *'Denegada'*
- *withdrawal_reason* is not one of the following:
    - *'Upselling-cambio de contrato'*
    - *'Cambio a Bundle Online'*
    - *'Cambio de Contrato/propuesta/producto'*

These rules filter out commercial or administrative changes that don't represent real customer loss.

2. Implicit Churn via Inactivity

In addition to explicit withdrawals, some advertisers stop using the platform without formally cancelling their contract. These cases are identified using a second source: the advertiser dataset, which includes a field called *contrato_churn_date*.

A custom procedure checks for advertisers who:

- Have no explicit churn recorded
- Do have a non-null *contrato_churn_date*
- And were still active in the month prior to that date

If these conditions are met, churn is registered in the previous active month. This method ensures we capture users who silently abandon the platform. Once churn is detected, all activity beyond that point is disregarded to maintain data consistency.

1.3 Project objectives

The primary goal of this project is to develop a predictive model capable of identifying advertiser churn one month in advance, using a combination of performance indicators and behavioural metrics. Since the outcome involves predicting a binary event, this is fundamentally a classification problem.

Churn is defined based on historical withdrawal events and patterns of inactivity, as previously described. However, it's important to note that the churn date present in the data (*contrato_churn_date*) does not always imply a definitive cancellation, as some users may renew their contracts. For this reason, the prediction target is carefully adjusted to anticipate churn with a one-month lead time, allowing for proactive business decisions.

Example:

Advertiser_zrive_id has a definitive withdrawal in 202303, so that churn = 1 on 202302.

advertiser_zrive_id	period_int	churn
1	202301	0.0
1	202302	1.0

1.3.1 Methodology

The project was carried out with complete flexibility in terms of analytical approach and implementation. This included:

- Defining the modelling strategy, such as:
  - Selecting and engineering key features
  - Creating derived metrics from raw data
  - Handling missing values or imbalances

- Choosing machine learning algorithms freely, based on experimentation and model performance.

The only constraint was to justify all technical decisions rigorously and to validate the model results using appropriate evaluation metrics.

1.3.2 Expected deliverables

The outcome of the project includes a comprehensive report containing:

1. The methodology used for data processing, modelling, and evaluation.
2. The results obtained, including key performance indicators of the predictive model.
3. Business recommendations based on the analytical findings.
4. Additional proposals or complementary developments that may enhance churn prevention strategies.

This project is intended not only to provide a working predictive model but also to offer valuable insights into the underlying dynamics of churn at Sumauto, ultimately supporting data-driven decisions for customer retention.

1.4 Stakeholders and Zrive collaboration

This project was made possible through a collaborative initiative between Zrive Applied Data Science and Komorebi, a company based in Madrid. The business case was originally provided by Sumauto, a group of vehicle classifieds portals, and Komorebi served as the link between industry needs and the student team.

Throughout the project, we received academic supervision from Guillermo Santamaría, a lead instructor at Zrive. Guillermo is currently a Machine Learning Engineer at Meta and has over nine years of experience in the field. He previously worked at companies such as Fintonic, Cabify, and Farfetch, contributing to the development of real-world machine learning systems. His guidance ensured that our approach remained both technically rigorous and aligned with best practices in the industry.

2. Exploratory Data Analysis (EDA)

2.1 Data overview

1. *zrive_dim_advertiser*: Advertiser general and contractual information

This dimension table stores static and longitudinal data about each advertiser on the platform. Each row corresponds to a unique advertiser identified by *advertiser_zrive_id*.

Key fields:

- *advertiser_group_id*: Identifies if the advertiser belongs to a larger business group or franchise.

Contractual fields (important for understanding churn dynamics):

- *min_start_contrato_date*: Date of the first ever contract signed by the advertiser and used to compute their tenure on the platform.
- *max_start_contrato_nuevo_date*: Start date of the latest contract classified as "new". This helps detect returning users who reactivated after previously churning.
- *contrato_churn_date*: End date of the latest contract. A null value indicates an ongoing contract without a defined end. However, if this date is in the future, it does not guarantee an upcoming churn, as many contracts are later renewed.

- *province_id / advertiser_province*: Geographic location of the advertiser, which may influence market behavior.
- *updated_at*: Timestamp of the last data update for this advertiser.

Note: Advertisers can churn and return multiple times. These fields help identify such contract cycles and are fundamental to the modelling process for both explicit and implicit churn.

2. *zrive_fct_monthly_snapshot_advertiser*: Advertiser monthly metrics

This fact table captures monthly activity snapshots for each advertiser. It is key for modelling behavioural signals over time.

Primary keys:

- *advertiser_zrive_id*
- *period_int*

These define the granularity: one row per advertiser per month.

Key features (metrics):

- Ad volume and activity:
    - *monthly_contracted_ads, monthly_published_ads, monthly_unique_published_ads, monthly_total_distinct_ads*
    - Breakdown of premium ads: *monthly_oro_ads, monthly_plata_ads, monthly_destacados_ads, monthly_pepitas_ads*

- Engagement metrics:
    - *monthly_shows_ads*, *monthly_visits_ads*, *monthly_leads*
    - *monthly_total_calls, monthly_total_emails, monthly_total_phone_views*

- Revenue signals:
    - *monthly_total_invoice*: billing
    - *monthly_avg_ad_price*: average price of posted ads

- Derived engagement (unique):
    - *monthly_unique_calls, monthly_unique_emails, monthly_unique_leads*: valuable to understand advertiser content attractiveness across regions

- Contract status:
    - *has_active_contract*: binary indicator of whether the advertiser had an active proposal during the month. It is often used to align contract periods with actual activity.

Importance: This dataset captures time-series behaviour, which is crucial for detecting signals of drop-off, change in engagement, or sudden decline before a churn.

3. *zrive_advertiser_withdrawals*: *Advertiser churn records*

This table logs advertiser withdrawals (both temporary and permanent) and is the main source for explicit churn labelling.

Main columns:

- *withdrawal_id*: Unique ID for the withdrawal event.
- *advertiser_zrive_id*: Links to the advertiser.
- *withdrawal_creation_date*: Date the cancellation was initiated.
- *withdrawal_type*: Type of withdrawal.
- *withdrawal_status*: Shows if the withdrawal was finalized, denied, or retracted.
- *withdrawal_reason*: Provides qualitative insight into the rationale behind the churn. Useful for post-hoc analysis or segmentation.

3. Data preprocessing and feature engineering

The dataset was constructed to predict advertiser churn, focusing specifically on forecasting the first churn event per user. This goal strongly influenced the preprocessing and filtering steps applied.

The primary source was the *zrive_fct_monthly_snapshot_advertiser* table, which consolidates monthly snapshots of advertiser activity and contract status.

To define the churn target, explicit churns were identified by merging the dataset with advertiser withdrawals. These represent users who actively cancelled their contracts through a withdrawal within a given month.

Additionally, implicit churns were inferred from the *contrato_churn_date* field in the advertiser data. These correspond to users whose contracts ended without a formal withdrawal. In such cases, churn was assigned to the month prior to the contract end date, but only if the user had activity during that month, indicating they stopped using the service by simply not renewing.

After assigning churn events, all rows occurring after a user's first churn were removed. This ensures the dataset only includes behaviour leading up to the initial churn, which aligns with the model's objective.

Advertisers with no registered churn and whose last observed activity occurred before the final period were excluded, as they likely represent churned users without an explicit label.

Besides, during the data exploration phase, it was observed that several features contained a high proportion of null values (approximately 40% of all rows in some cases). Among these was *monthly_distinct_ads*, along with others. Multiple approaches were considered to handle the missing data, such as imputation or computation of derived values. However, given the extent of the null values and the risk of introducing bias or distortion, the most robust solution was to remove the affected columns entirely.

Finally, rows were dropped where advertisers had no active contract and did not publish any ads, as they do not provide meaningful signals for churn prediction.

These steps resulted in a cleaned and temporally consistent dataset tailored to train models that accurately predict the first occurrence of advertiser churn.

3.1 Creation of features

This project involved the engineering of three main types of features: temporal features, ratio-based indicators, and aggregated statistics over recent months. These features were designed to enrich the dataset with contextual and behavioural information, thereby enhancing its predictive potential.

3.1.1 Temporal features

These features describe the relationship between the advertiser's timeline and their contractual activity.

Feature name	**Description**	**Purpose**
tenure	Number of months since the advertiser’s first recorded activity or contract	Captures customer seniority or how long they have been active

months_since_last_contract	Number of months since the last new contract was signed	Measures recency of contractual engagement
has_renewed	Binary feature (0 or 1) indicating if the advertiser	Acts as a proxy for advertiser engagement

3.1.2 Ratio features

These features reflect performance efficiency, engagement, and economic effectiveness. They are derived from existing columns in the dataset.

a) Ad ratios

Feature name	**Formula**	**Description**
ratio_published_contracted	Published ads / Contracted ads	Indicates whether advertisers are using their ad inventory
ratio_unique_published	Unique published ads / Published ads	Measures content repetition or uniqueness.
ratio_premium_ads	Premium ads / Published ads	Reflects the advertiser's investment in premium ad formats

Premium ads include: oro, plata, destacados, and pepitas.

b) Engagement ratios

Feature name	Formula	Description
leads_per_published_ad	Leads / Published ads	Shows how effective each published ad is in generating leads
leads_per_premium_ad	Leads / Premium ads	Evaluates performance of paid premium ad formats
visits_per_published_ad	Visits / Published ads	Measures average visibility of each ad
leads_per_visit	Leads / Visits	Indicates conversion rate from visit to lead
leads_per_shows	Leads / Shows	Evaluates lead generation based on total ad impressions

c) Economic ratios

Feature name	Description	Purpose
invoice_per_published_ad	Total invoice / Published ads	Revenue contribution per ad
invoice_per_lead	Total invoice / Leads	Monetization efficiency of each lead

3.1.3 Aggregated statistics

These features summarize recent trends and variations using a 3-month rolling window.

Feature name	Description
monthly_leads_3_months_mean	Mean of monthly leads over the past 3 months
leads_per_visit_3_months_mean	Average lead-to-visit ratio over the past 3 months
invoice_per_lead_3_months_mean	Revenue per lead trend over recent months
invoice_per_lead_3_months_mean_delta	Difference between the current value and its 3-month mean (trend change)

Note: The delta features measure how much a feature deviates from its recent average, capturing short-term fluctuations or anomalies.

After executing the full pipeline, the final dataset includes:

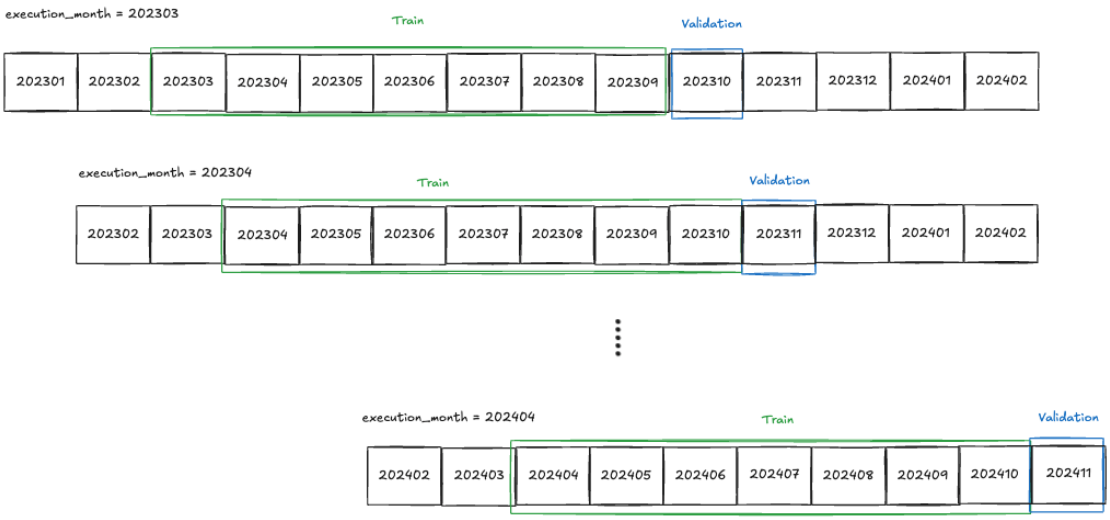
- Enriched time-aware customer information.
- Detailed performance and monetization ratios.
- Smoothed historical trends with deviation metrics.

4. Methodology and evaluation

4.1 Sliding window evaluation scheme

In this project, we use a sliding window evaluation strategy to assess model performance over time, which is well-suited for time-dependent datasets with evolving data distributions.

For each evaluation cycle, the model is trained on a rolling window of six consecutive months starting from a given execution month and validated on the following month. This process is repeated in a loop, advancing the window one month at a time, until the last available month in the dataset is reached as the final validation period.



Time series and temporally structured data are not independent and identically distributed as their distributions shift over time, trends change, and patterns evolve. This dynamic nature can lead models to overfit recent conditions and perform poorly when applied to future data. Additionally, models may fail to generalize to unseen timeframes, particularly if they are trained and tested using random or mixed time splits that ignore temporal structure.

To address these challenges, a sliding window evaluation method offers significant advantages. It respects the chronological order of data, avoiding information leakage from the future, and simulates real-world deployment by training models on

past data to predict future outcomes. This approach also helps detect concept drift, as it reveals whether model performance degrades as the data evolves. Moreover, by progressively moving the training and validation window forward in time, it enables evaluation across multiple periods, providing a more robust and realistic picture of model performance.

Sliding window evaluation is especially recommended in temporal problems where the target variable or features are seasonal or subject to trend shifts. It is also useful when there is a risk of overfitting to short-term patterns, or when there is a need to understand how well a model adapts over time and whether frequent retraining is necessary.

4.2 Evaluation metrics

We evaluate the churn prediction model using both technical metrics and business-oriented metrics to ensure its real-world value.

4.2.1 Technical metrics

a) Log loss

Used during training to assess the quality of predicted probabilities. Penalizes incorrect confident predictions, encouraging well-calibrated outputs.

b) AUC-ROC and AUC-PR

Both are threshold-independent metrics.

- AUC-ROC indicates the model's general ability to distinguish churn vs. no churn.
- AUC-PR is more informative in imbalanced datasets, like churn, where most customers don't churn in each month. It better reflects model performance on the minority class.

These metrics are essential to understand the model's overall behaviour, but don’t directly inform business decisions.

4.2.2 Business metrics

a) ROI

To reflect the actual economic impact, we use custom business metrics based on these assumptions:

- Losing a client costs **€1,000**.
- A retention action costs **€100**.
- **70%** of the time, a correct action keeps the client.

We simulate different thresholds to calculate:

- Total action costs (true + false positives).
- Savings from retained churners.
- Net profit and ROI.

4.3 Threshold selection: Business trade-offs

Once the model is trained, decisions depend on a specific threshold. This is critical because:

- Lower thresholds increase recall (we catch more churners) but lead to more false positives, triggering unnecessary retention actions.
- Higher thresholds reduce action costs but risk missing churners, which is more expensive.

Choosing the right threshold requires knowing whether the company prefers:

- To capture more true churners at the risk of acting unnecessarily.
- Or to minimize false alarms, even if that means losing more customers.

This is where business priorities must guide model usage.

4.4 Tested models

We started with a baseline approach: assigning all customers a churn probability equal to the churn rate observed in the training months. This naive method gave us a reference point to evaluate any model against.

4.4.1 Logistic Regression

We tested both L2 (Ridge) and L1 (Lasso) regularization. Lasso was particularly useful for feature selection, allowing us to identify the most relevant variables. We retrained the models using only the top features and re-tested both penalties.

While logistic regression models consistently outperformed the baseline, the improvement in performance was limited.

4.4.2 Random Forest

Random Forest models outperformed both logistic regression and the baseline, and they showed good temporal generalization when tested with sliding window validation. We monitored the mean and standard deviation of both AUC and error across time, confirming stable performance.

However, the gain over simpler linear models was modest, and considering the computational cost and complexity, we questioned whether it was truly worth it. Moreover, based on feedback from our tutor, Guillermo Santamaría, we considered Random Forest somewhat outdated for this type of problem.

4.4.3 Final model selection: XGBoost

Following Guillermo's recommendation, we explored gradient boosting, a family of models that builds predictors sequentially, where each new model corrects the errors of the previous one. This approach allows the model to focus on difficult-to-predict cases, often resulting in strong performance on complex datasets.

We chose XGBoost for its efficiency, flexibility, and proven success in structured data problems.

We started hyperparameter tuning with a low learning rate, which helps the model learn gradually and reduces the risk of overfitting. After experimentation, we achieved a model with strong predictive power, good generalization across time, and robustness to overfitting, clearly outperforming the other tested approaches.

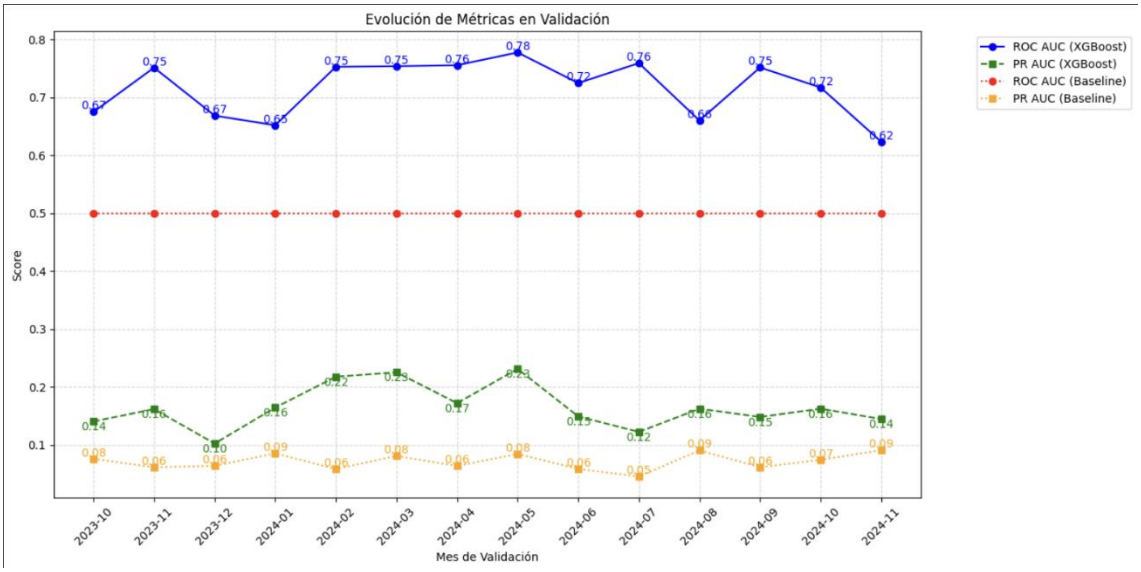
This combination of performance, stability, and scalability made XGBoost the best choice for our churn prediction task.

5. Results and analysis

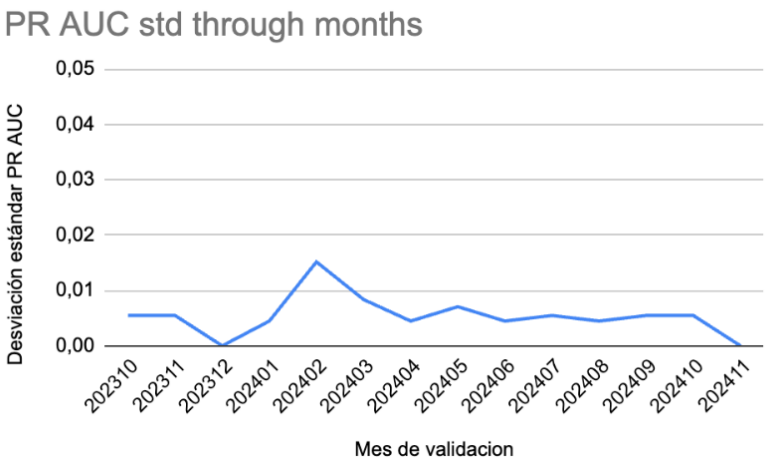
5.1 Evaluation of metrics through validation months

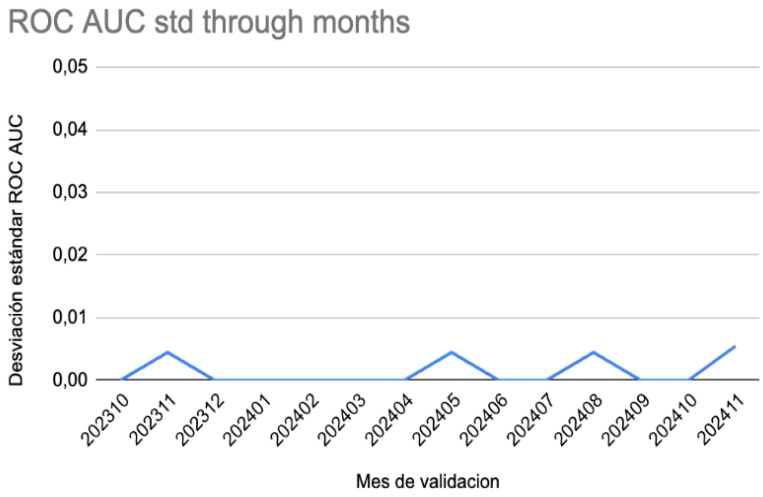
To assess the value of our model, we compared it to a baseline that assumes every customer has a constant churn probability equal to the churn rate in the training period. While this naive predictor yields an AUC-ROC of 0.5, as it cannot differentiate between churn and non-churn cases, our XGBoost model consistently outperformed it, reaching a mean ROC-AUC of 0.72 and PR-AUC of 0.18 across the validation months. Since ROC-AUC measures a model's ability to distinguish between classes, a score of

0.5 reflects complete inability to separate them, thus validating the baseline's simplicity.



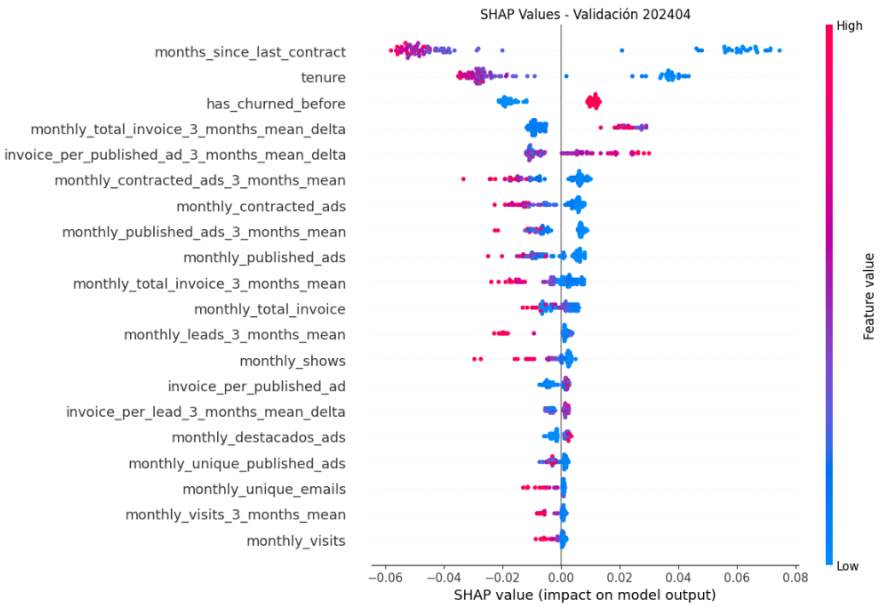
Model performance varied over time, which is expected in temporal problems where data distribution naturally shifts month to month. To ensure this variation wasn't due to model instability, we retrained the final XGBoost model five times with different random seeds and computed the standard deviation of ROC-AUC and PR-AUC for each validation window. Both metrics showed a low standard deviation (0.05), reinforcing that the model was stable and not sensitive to random initialization.





We followed a progressive modelling approach, starting from simple linear models. We first tested logistic regression with both L2 (ridge) and L1 (lasso) regularization. Using Lasso, we also extracted feature importance, selecting the top features and retraining models on this reduced set. While these models outperformed the baseline (which assumed that every customer has a churn probability equal to the churn rate in the training months), the improvement was modest and not sufficient given the problem's complexity.

We then moved to Random Forest, which provided more flexibility and better predictive performance. To assess model behaviour, we visualized precision-recall (PR) and ROC curves, along with SHAP values to interpret feature importance. SHAP values provide a unified framework to explain the contribution of each feature to individual predictions, based on cooperative game theory. This allowed us to understand not only which features were important globally, but also how they influenced specific predictions. Despite these insights, the overall performance was still not significantly better than the best logistic models, and the added complexity and computational cost made us question whether the gain was worth it.



Additionally, feedback from our tutor, Guillermo Santamaría, suggested that Random Forest is currently considered outdated for problems of this nature.

Following that advice, we adopted XGBoost, a modern and powerful gradient boosting algorithm. We started with a low learning rate to prevent overfitting and allow the model to build knowledge gradually. To monitor the learning dynamics, we plotted boxplots of the log loss across training and validation sets for different numbers of trees, treating each box as a summary of log loss across the validation months. These boxplots served as learning curves, revealing how the model's performance evolved with more boosting rounds. XGBoost consistently outperformed all previous models and showed strong generalization over time, becoming our final choice.

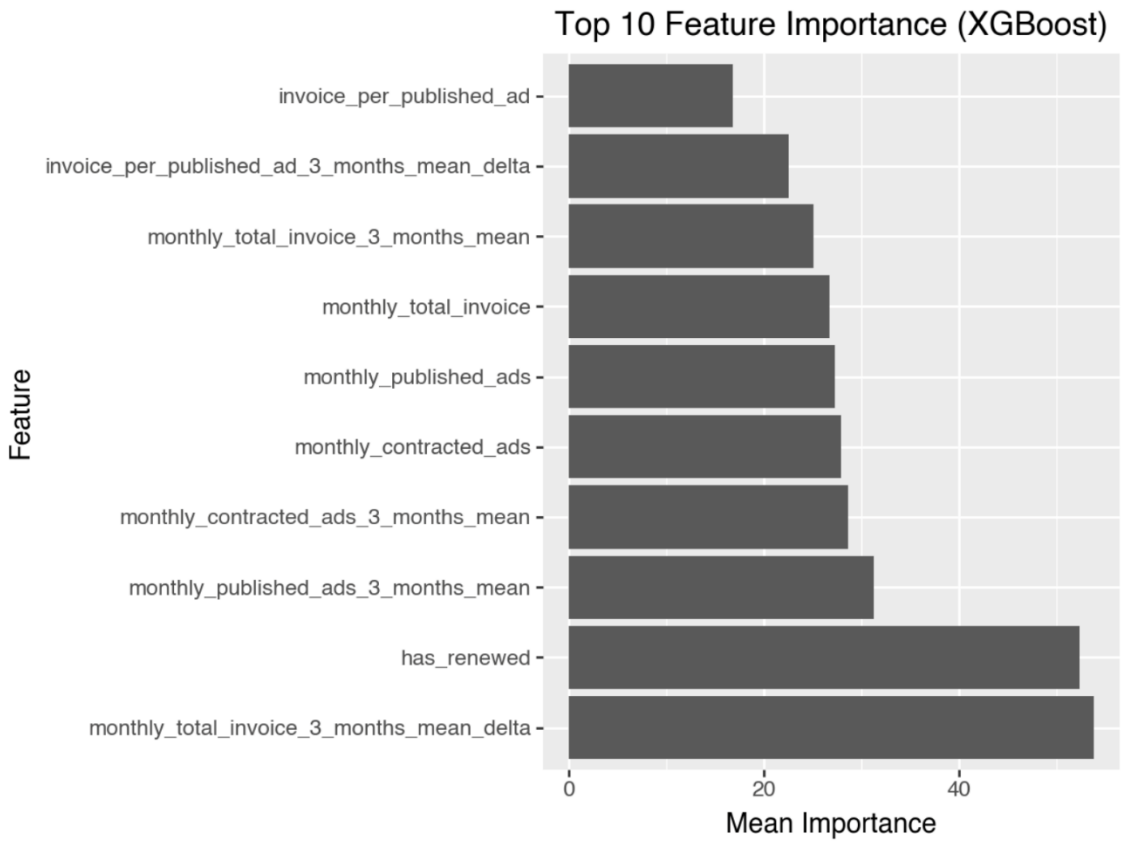
Additionally, we validated how performance was affected by the temporal distance between the training window and the validation month. We observed that the farther the validation month was from the training period, the higher the log loss became. This behaviour is expected in time-dependent data, as underlying patterns and distributions change over time—a phenomenon known as concept drift. The model gradually becomes less effective at capturing patterns it was not trained on, reinforcing the importance of frequent retraining or adaptation in temporal problems.

5.2 Feature importance analysis

Regarding feature importance, the initial XGBoost model highlighted two dominant features:

- *tenure*: the number of months the customer has been with the company.
- *months_since_last_contract*: time since the customer last signed a contract.

Both features seemed highly predictive, but also raised red flags about possible data leakage, as they could indirectly capture information about future behaviour or churn labels. After removing these two variables, the importance scores across remaining features were more evenly distributed and aligned better with business intuition. Although this reduction slightly decreased overall performance, we preferred the safer, more interpretable version of the model, which avoids relying on potentially misleading information.



5.3 Business metrics evaluation and threshold trade-offs

To complement traditional performance metrics, we evaluated the model using business-oriented KPIs such as net profit and return on investment (ROI). While the values used in this analysis are not real, they serve as indicative estimates for decision-making.

We made the following assumptions:

- If a client churns, the company incurs a cost of **€1000**.
- Taking a retention action (e.g., a promotion) has a cost of **€100**.
- When we act on a churn prediction, **70%** of those clients stay thanks to the intervention.

This creates a clear trade-off when deciding what probability threshold to use to classify a customer as a churn risk.

- A **low** threshold (close to 0) means almost all clients will be predicted as churners. This ensures we capture most true positives, but at the cost of acting unnecessarily on many false positives. As a result, we spend heavily on retention actions that may not have been needed, leading to high costs and potentially negative ROI.
- A **high** threshold (0.3) is more conservative. We only act on the most confident churn predictions. This reduces false positives and unnecessary spending but increases the risk of missing true churners (false negatives), which results in lost clients and missed opportunities to retain them.

In our specific case, model predictions fall mostly between 0 and 0.3. As a result, setting the threshold above 0.3 leads to all predictions being classified as 0, meaning the company takes no action at all. This causes net profit and ROI to drop to zero, as there is no investment and no return.

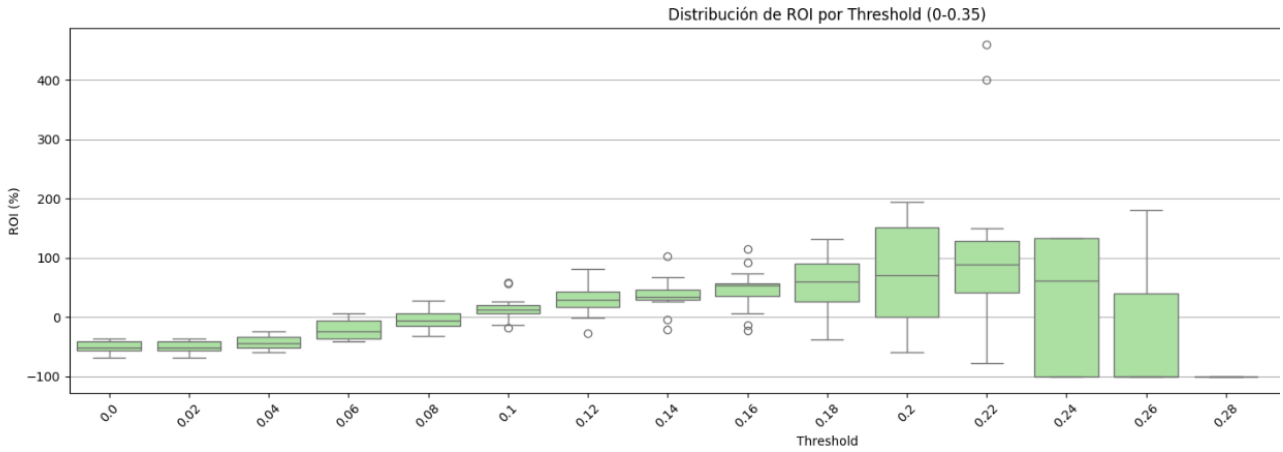
5.4 Analysis of ROI by threshold

We visualized ROI across different thresholds using boxplots that aggregate results across validation months.

- Extremely **low** thresholds (**0.01**) result in consistently **negative ROI**. This is because the model flags nearly everyone as a churner, causing excessive and often unnecessary spending.

- As the threshold increases, we begin to see **positive ROI** values, especially around **0.14 to 0.16**. These thresholds strike a favourable balance: we correctly identify a significant portion of churners without overspending on clients who wouldn't have left. Additionally, the boxplots at these thresholds show low dispersion, meaning the performance is stable month over month, which is essential for risk-averse business decisions.

- Interestingly, at slightly higher thresholds like **0.2**, the expected ROI may be higher in some months, but the dispersion increases significantly. This indicates that while some months yield higher profits, others can lead to negative outcomes—increasing the business risk and month-to-month unpredictability.



6. Conclusions and recommendations

6.1 Key findings

Throughout the analysis, several consistent churn patterns were identified. Two features stood out due to their high importance: user tenure and the time since the last contract (*months_since_last_contract*). The model consistently showed that the longer a user had been active on the platform or without recent contract modifications, the lower their probability of churn. This suggests that long-term users tend to be more stable, and that retention improves over time. Additionally, the model detected meaningful patterns in user activity and engagement variables, which could be leveraged to trigger earlier and more targeted retention actions.

6.2 Model limitations

One notable limitation is that the current model focuses only on first-time churn. Clients who leave and later return (and churn again) are not explicitly modelled, which could limit the generalizability of predictions in more complex churn scenarios.

Furthermore, some input variables such as *tenure* or *months_since_last_contract* raised concerns about potential data leakage, as they had disproportionately high importance in early versions of the model.

Although these features were later removed from the final model to be cautious, it is still possible that their high importance was not due to data leakage, but because they were capturing real patterns in how contracts work. For example, some contracts might include discounts that expire after a certain number of months, which could naturally increase the chance of churn at that point. So, even though we excluded these variables to avoid any risk, they might have been reflecting real business behaviour.

Therefore, we recommend a closer collaboration with business stakeholders to validate whether such contractual mechanisms exist and assess whether these features should be reintroduced.

6.3 Recommendations for implementation

- **Suggested threshold for deployment:** Based on ROI analyses across thresholds, values in the range of **0.14–0.16** consistently delivered **positive** and **stable** returns, with lower month-to-month variability. While other thresholds (**0.2**) might lead to **higher** profits in some months, they also involve greater **risk** and **volatility**. This makes **0.14–0.16** a **conservative** yet effective choice for initial deployment.

- However, this decision should ultimately be discussed with Sumauto's executive team, as the choice of threshold depends on business risk appetite. Some teams may prefer a more aggressive strategy that maximizes potential return, while others may opt for stability even at the cost of slightly lower gains. The final decision should weigh not only the data science perspective but also the company's strategic and financial priorities.

- **Suggested Actions:** For clients predicted as high-risk, we recommend testing personalized retention strategies, such as tailored promotions or proactive account management. One particularly relevant insight from the model is that the feature *monthly_total_invoice_3_months_mean_delta*, which measures the change in invoice amount compared to the average of the previous three months—was consistently ranked as the most important predictor. The only features that ranked higher, *tenure* and *months_since_last_contract*, were excluded from the final model to avoid potential data leakage. This highlights that unexpected changes in pricing or billing may be a strong driver of churn, and addressing these proactively could help improve retention. The cost/benefit assumptions used in this report can be further refined once real campaign data becomes available.

6.4 Future work

- **Include richer context from existing datasets**: For instance, the withdrawals dataset could be explored further to extract reasons for churn, number of previous cancellations, or behavioural patterns leading to disengagement. Currently, we rely on a single binary variable (*has_renewed*), which only implicitly captures the occurrence of a previous cancellation and the subsequent signing of a new contract. While this provides some signal, it lacks

the granularity needed to fully understand churn dynamics or recurring churn behaviour.

- **Experiment with alternative modelling approaches**: While tree-based models like XGBoost performed best in this case, future iterations could explore ensemble models, time-series-aware architectures, or semi-supervised techniques to leverage the structure in customer history more effectively.

- **Refine business metrics**: The ROI assumptions used here are based on indicative values. Incorporating actual cost and conversion rates from Sumauto's marketing and sales departments would significantly improve the accuracy and trustworthiness of these analyses.

- **Re-examine contractual mechanics**: As noted earlier, a deeper understanding of how client contracts evolve over time—particularly whether promotions or pricing changes occur after a fixed number of months—could help explain why variables like *tenure* and *months_since_last_contract* appeared so predictive in earlier models. If these are genuine business patterns rather than artifacts, these features could be reinstated into future models with confidence.

7. Appendices

- Technical details: GitHub repository

The repository's organizational structure is detailed in its README file.