Bank Marketing Campaign Predictive Analysis

Juan Felipe Caicedo Castro AI Engineer, Case Study IC7

February 2026

Dataset: Bank Marketing Dataset (UCI), Portuguese Bank
Records: 41,188 clients   Features: 20 input variables
Task: Binary classification, term deposit subscription

Contents

1. Business Context and Project Objective

1.1. Background

The financial services sector relies heavily on direct marketing campaigns to cross-sell and up-sell products to existing customers. A term deposit (also known as a *certificate of deposit*) is a fixed-term savings instrument in which a client locks in a sum of money for a predetermined period at a guaranteed interest rate. For a commercial bank, term deposits represent a low-cost, stable source of funding, making their acquisition a critical operational priority.

The dataset for this case study originates from a series of direct marketing campaigns—conducted exclusively via outbound telephone calls—of a Portuguese banking institution. Each row corresponds to one contact attempt with a client, and the outcome variable y records whether that client ultimately subscribed to a term deposit (`yes` or `no`).

Business Problem Statement

"Given historical data about clients, economic conditions, and previous campaign interactions, can we predict which clients are most likely to subscribe to a term deposit, enabling the bank to focus its outreach resources on the highest-probability prospects?"

1.2. Strategic Motivation

Traditional "spray-and-pray" telephone campaigns are expensive, both in agent time and in customer experience. Without a data-driven prioritization mechanism, marketing teams must contact large, unfiltered lists of clients, resulting in:

- Low conversion rates, only $\approx 11\%$ of contacts result in a subscription (see Section 2), meaning roughly 9 out of every 10 calls generate no revenue.
- High cost per acquisition, each unproductive call incurs an agent time, telephony costs, and potential customer churn due to unwanted contacts.
- Missed opportunities, without ranking, high-probability clients may be contacted last or not at all before campaign budgets are exhausted.

A predictive model that ranks clients by their subscription probability directly addresses all three pain points, turning a cost center into a strategic asset.

1.3. Project Scope and Objectives

This project covers the full analytical lifecycle from raw data to a deployable recommendation, structured around four pillars:

1. Exploratory Data Analysis (EDA): Characterize the 41,188-record dataset, understand feature distributions, identify class imbalance, and derive actionable business insights from the data alone.
2. Predictive Modeling: Build, compare, and tune machine learning classification models (Logistic Regression, Random Forest, XGBoost) that rank clients by their likelihood of subscribing. Model selection prioritizes *Recall* and *F1-Score* on the minority class given the severe class imbalance.

3. Cloud Deployment Architecture: Propose a scalable, cost-efficient AWS serverless infrastructure that supports the full ML lifecycle, from data ingestion and retraining to real-time inference via a REST API.
4. Generative AI Strategy: Assess whether and how Generative AI (LLMs) can augment the predictive solution, distinguishing between tasks better suited to classical ML and those where GenAI adds genuine value.

1.4. Dataset Overview

Table 1: High-level dataset characteristics

Attribute	Detail
Source	UCI Machine Learning Repository, Bank Marketing Dataset
Institution	Portuguese commercial bank
Time period	May 2008 – November 2010
Total records	41,188
Input features	20 (client demographics, campaign, socioeconomic context)
Target variable	`y`, binary: `yes` / `no`
Positive class rate	11.27% (`yes`)
Missing values	None (explicit nulls); "`unknown`" present in 6 features

The 20 input variables are organized into three conceptual groups:

1.4.1. Client Attributes (Demographics)

Variables describing the socioeconomic profile of the individual: `age`, `job`, `marital`, `education`, `default`, `housing`, `loan`.

1.4.2. Campaign Contact Variables

Variables describing the current and historical contact history: `contact`, `month`, `day_of_week`, `duration` (excluded, see below), `campaign`, `pdays`, `previous`, `poutcome`.

1.4.3. Socioeconomic Indicators

Macroeconomic context variables at the time of contact: `emp.var.rate`, `cons.price.idx`, `cons.conf.idx`, `euribor3m`, `nr.employed`.

//

Critical Note

The variable `duration` (duration of the last phone call in seconds) is not available before a call is made and therefore would constitute *data leakage* in a real production system. A model trained with `duration` achieves near-perfect accuracy ($\approx 99\%$) but is entirely useless operationally, since the goal is to decide *whom to call before the call happens*. This feature was excluded from all predictive models in this study.

1.5. Success Metrics

Given the severe class imbalance ($\approx 9:1$ negative-to-positive ratio), accuracy is a misleading metric, a naïve model that always predicts "no" would achieve 88.7% accuracy while identifying zero potential subscribers.

The evaluation framework therefore prioritizes:

- Recall (Sensitivity), the fraction of actual subscribers correctly identified. Missing a potential subscriber is a lost revenue opportunity.
- Precision, the fraction of predicted subscribers who actually subscribe. Low precision means wasted call-center resources.
- F1-Score (minority class), the harmonic mean of Precision and Recall; the primary optimization target as it balances both concerns.
- ROC-AUC, the model's overall ability to rank clients correctly, useful for generating a priority call list at any operating threshold.

2. Exploratory Data Analysis

2.1. Dataset Structure and Missing Values

The dataset contains 41,188 rows and 21 columns (20 input features + 1 target variable `y`). No explicit `null` or `NaN` values were detected; however, six categorical variables contain the category `"unknown"`, which was treated as a separate valid category during modeling.

Table 2: Count of `"unknown"` values per categorical feature

Feature	Unknown count
`default`	8,597
`education`	1,731
`housing`	990
`loan`	990
`job`	330
`marital`	80

2.2. Target Variable, Class Imbalance

The outcome variable `y` is severely imbalanced: only 11.27% of contacts result in a term deposit subscription. This imbalance is a central challenge that drives every modeling decision in this study, from metric selection (F1-Score / Recall over Accuracy) to the use of `scale_pos_weight` and `class_weight="balanced"`.

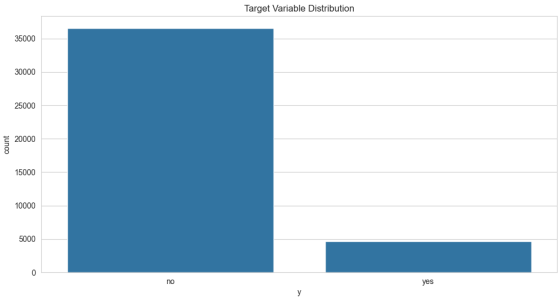


Figure 1: Target variable distribution: 88.73% “no” vs. 11.27% “yes”

2.3. Numerical Features

2.3.1. Descriptive Statistics

The dataset contains ten numerical variables spanning client demographics (`age`), campaign activity (`campaign`, `pdays`, `previous`), and macroeconomic conditions (`emp.var.rate`, `cons.price.idx`, `cons.conf.idx`, `euribor3m`, `nr.employed`).

2.3.2. Correlation Matrix

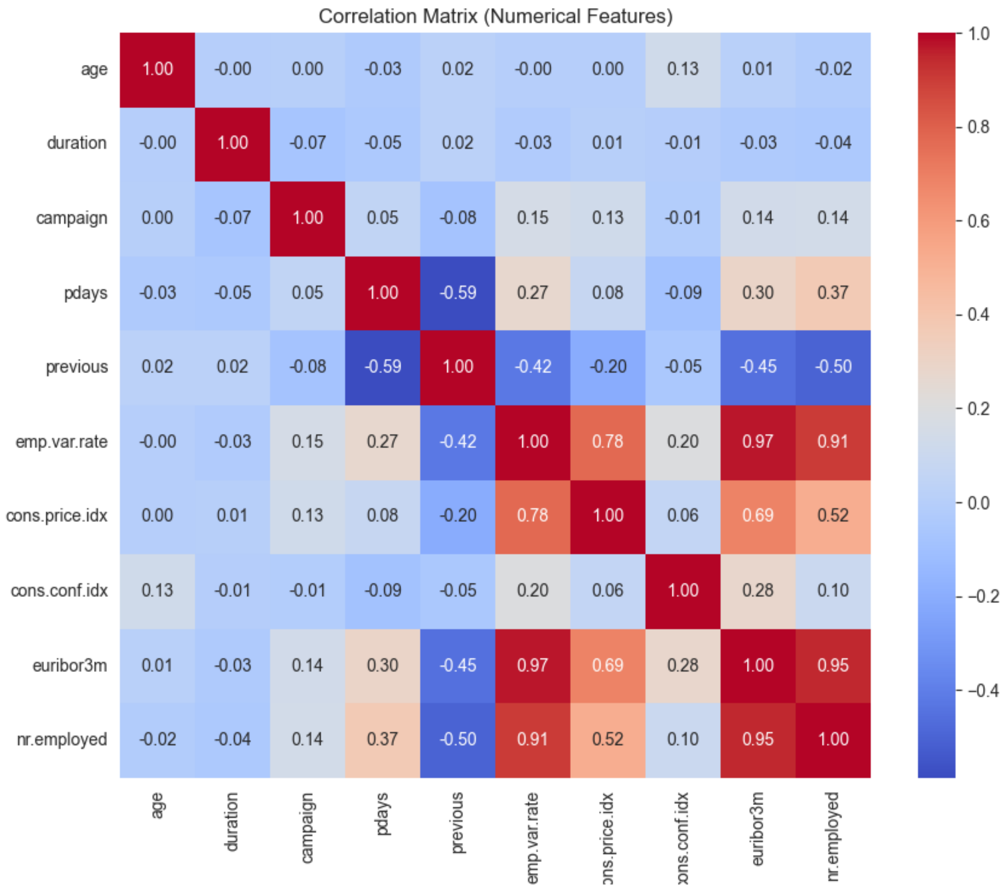
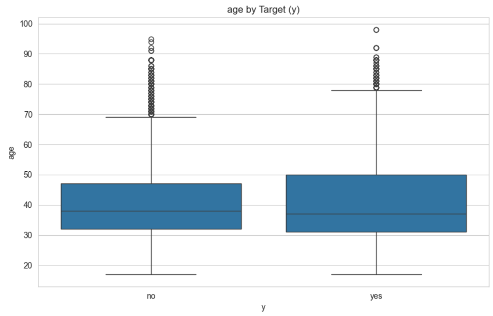


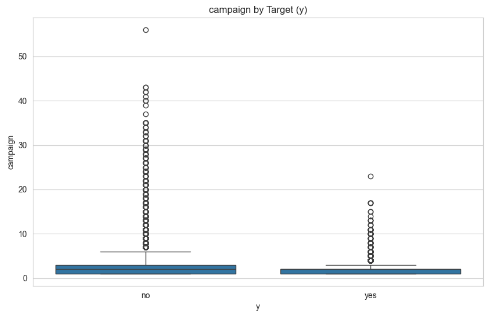
Figure 2: Pearson correlation matrix of all numerical features. Note the strong mutual correlation among the four macroeconomic indicators (`euribor3m`, `nr.employed`, `emp.var.rate`, `cons.price.idx`).

2.3.3. Boxplots, Numerical Features vs. Target

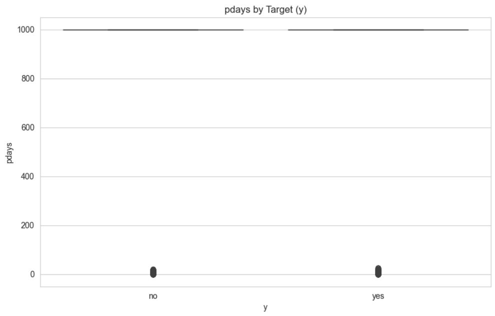
The following boxplots compare each numerical feature's distribution for clients who subscribed (`yes`) and those who did not (`no`).



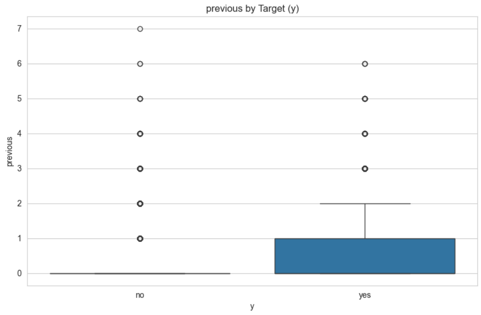
(a) `age`, older clients (retired segment) show slightly higher subscription rates.



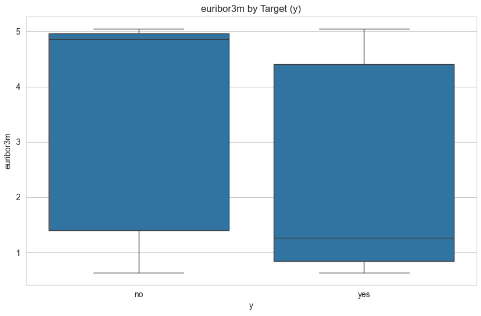
(b) `campaign`, fewer contact attempts correlate with higher success; after 3+ calls, conversion drops.



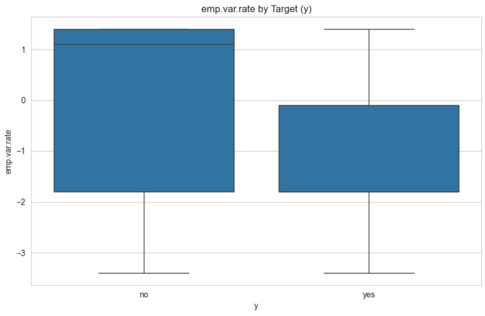
(a) `pdays`, most clients were never previously contacted (999); those contacted recently are more likely to subscribe.



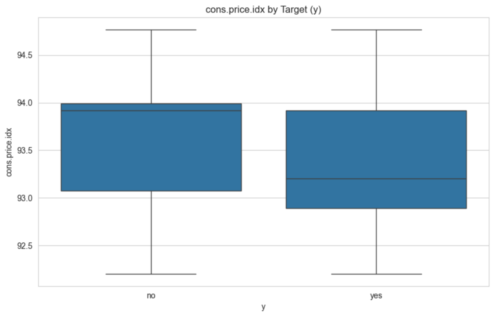
(b) `previous`, clients with prior contacts show higher subscription rates.



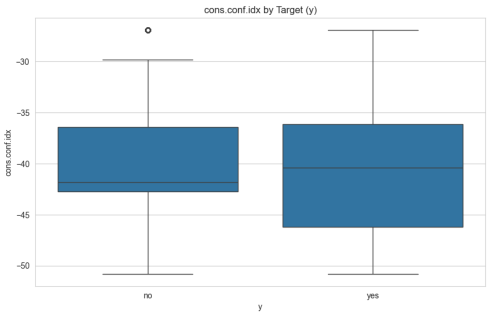
(a) `euribor3m`, key economic driver. Subscribers are concentrated in low-rate environments.



(b) `emp.var.rate`, negative employment variation (economic slowdown) correlates with higher subscription.



(a) `cons.price.idx`, subscribers tend to appear when the consumer price index is lower.



(b) `cons.conf.idx`, consumer confidence is slightly lower for subscribers, possibly reflecting defensive savings behavior.

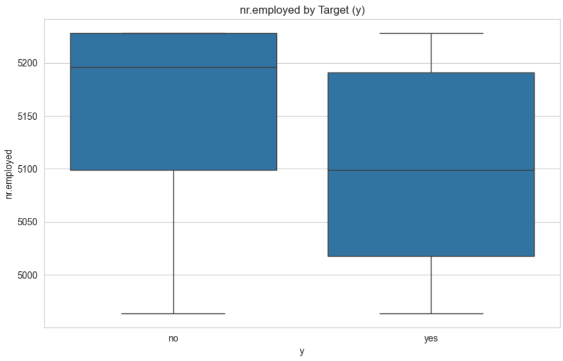


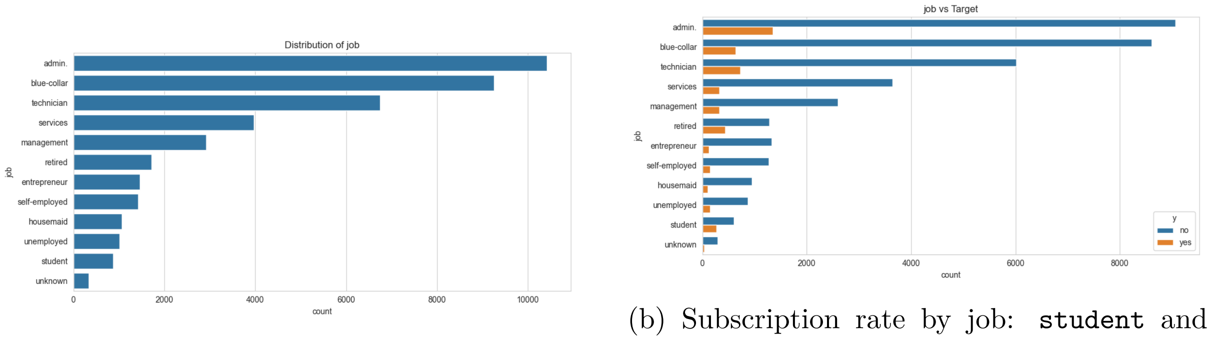
Figure 7: `nr.employed`, lower workforce numbers (recessionary periods) align with higher term deposit uptake.

Macroeconomic Insight

The four indicators `euribor3m`, `nr.employed`, `emp.var.rate`, and `cons.price.idx` are highly intercorrelated and collectively capture the macroeconomic climate. Clients are significantly more likely to subscribe during periods of *low interest rates* and *economic uncertainty*, a classic “flight to safety” behavior where fixed-term deposits become attractive.

2.4. Categorical Features

2.4.1. Job Type

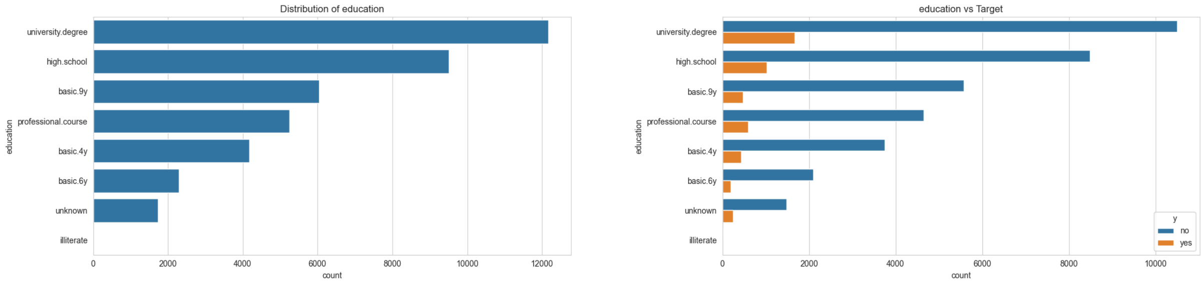


(a) Job distribution: `admin.` and `blue-collar` are the dominant categories.

(b) Subscription rate by job: `student` and `retired` show the highest conversion rates despite lower volume.

Figure 8: `job`, distribution and subscription rate

2.4.2. Education Level

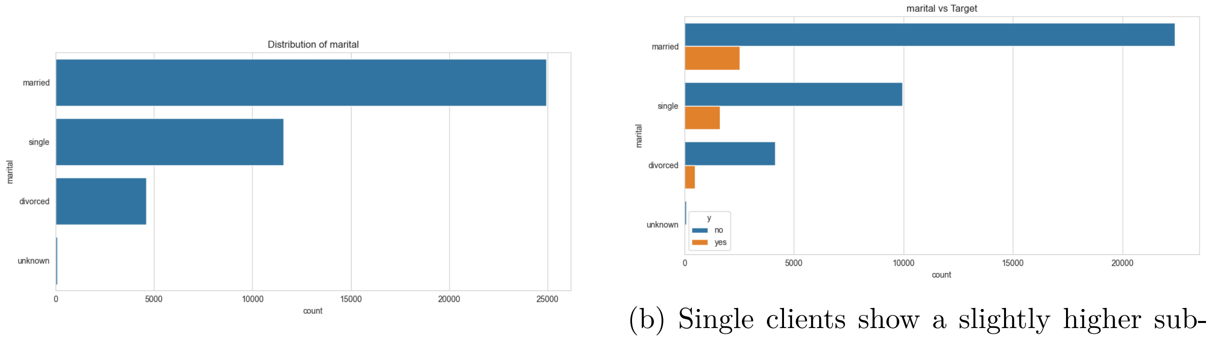


(a) Education distribution: university degree is the most common.

(b) Higher education levels correlate with higher subscription rates.

Figure 9: `education`, distribution and subscription rate

2.4.3. Marital Status

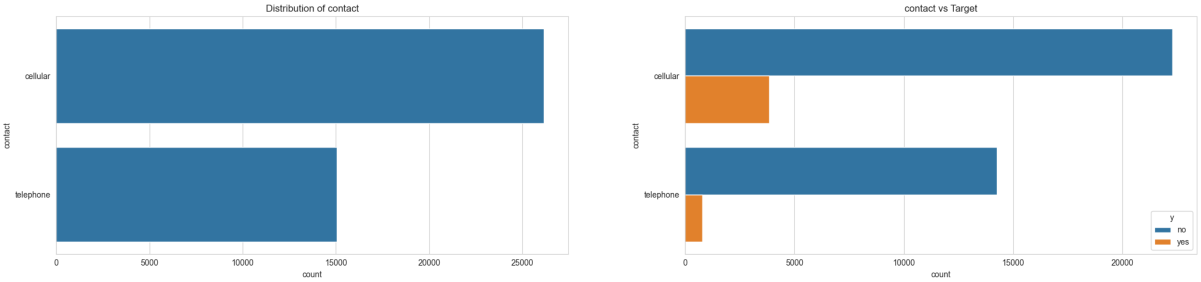


(a) Marital status distribution.

(b) Single clients show a slightly higher subscription rate.

Figure 10: `marital`, distribution and subscription rate

2.4.4. Contact Type

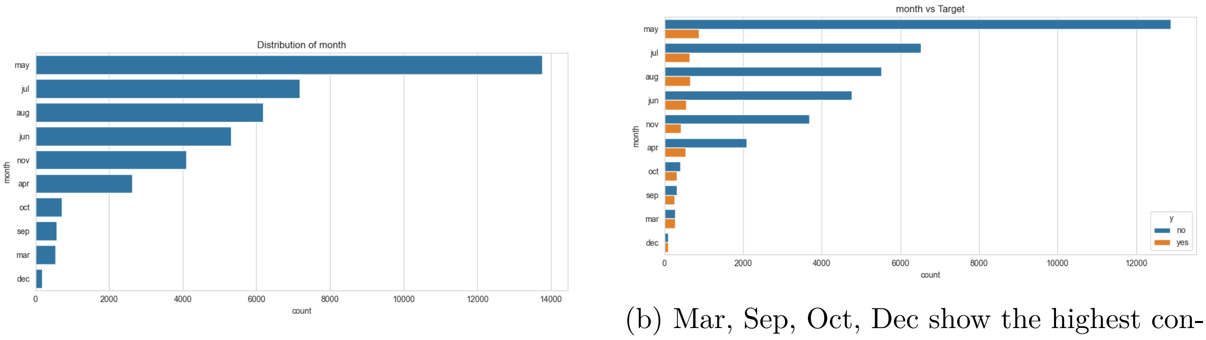


(a) Cellular contact is nearly twice as common as telephone.

(b) Cellular contacts produce significantly higher subscription rates than landline.

Figure 11: `contact`, type of communication and its effect on outcome

2.4.5. Campaign Month

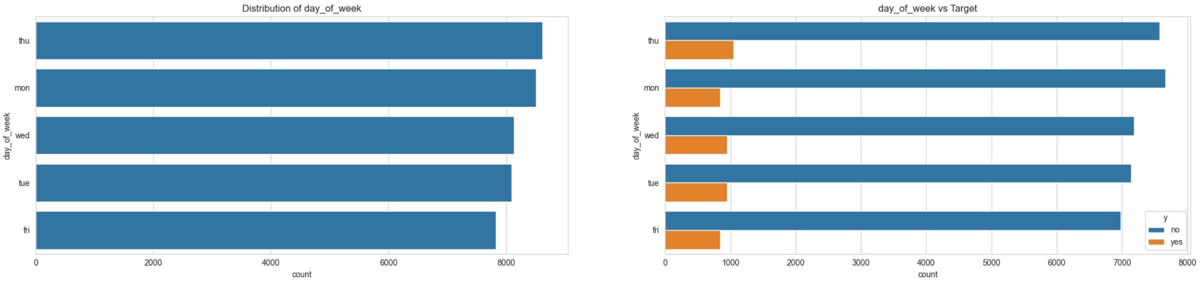


(a) May dominates call volume; December and March have the fewest.

(b) Mar, Sep, Oct, Dec show the highest conversion rates, despite lower overall contact volumes.

Figure 12: `month`, contact month and subscription rate

2.4.6. Day of Week

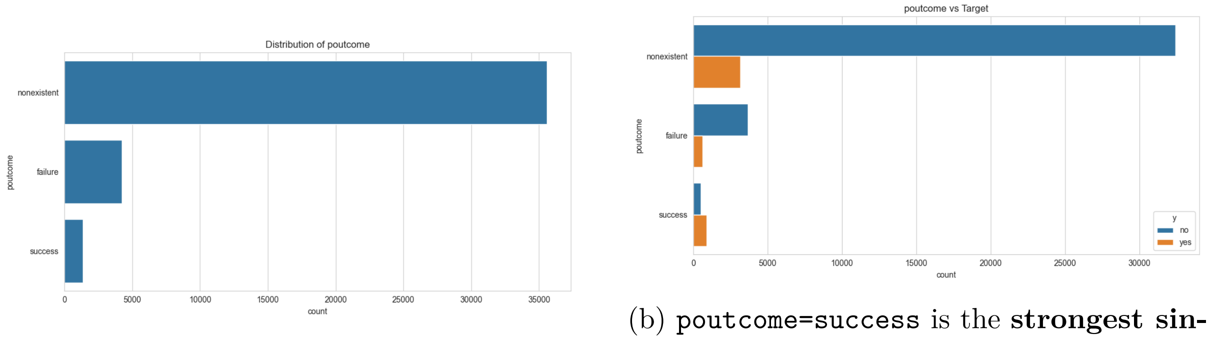


(a) Contacts are distributed relatively evenly across weekdays.

(b) Day of the week has minimal impact on subscription rate.

Figure 13: `day_of_week`, contact day and subscription rate

2.4.7. Previous Campaign Outcome

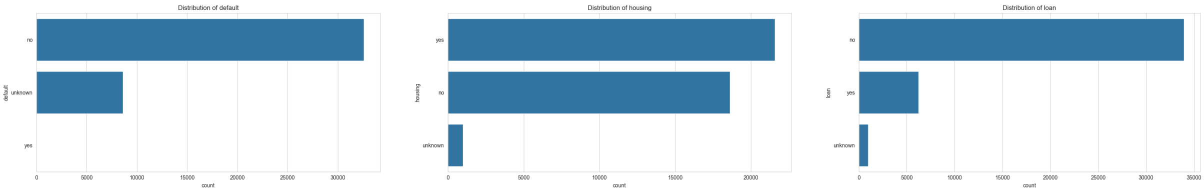


(a) Most clients (`nonexistent`) were never contacted before.

(b) `poutcome=success` is the **strongest single predictor** of subscription, prior success repeats itself.

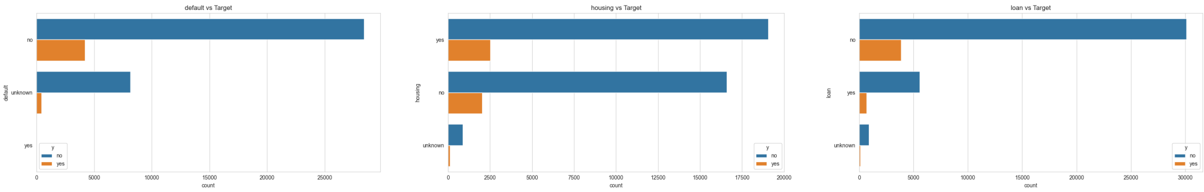
Figure 14: `poutcome`, previous campaign outcome

2.4.8. Other Categorical Features



(a) `default` (b) `housing` (c) `loan`

Figure 15: Distribution of financial status variables



(a) `default` vs. target (b) `housing` vs. target (c) `loan` vs. target

Figure 16: Financial status variables vs. subscription outcome

2.5. Key EDA Findings

Top Business Insights from EDA

1. Macro-timing matters most. The Euribor 3-month rate is the strongest continuous predictor. Launch campaigns when rates are low.
2. "Success breeds success." Clients with a prior successful campaign (`poutcome=success`) convert at the highest rate — these are the "low-hanging fruit".
3. Retire and dial. Students and retirees have above-average conversion despite smaller volume, a high-ROI niche.
4. Fewer calls, better results. Clients contacted more than 3 times in a single campaign are less likely to subscribe (`campaign` feature).
5. Cellular beats telephone. Mobile contacts outperform landline contacts significantly, align the dialing strategy accordingly.
6. Seasonal windows exist. March, September, October, and December show disproportionately high conversion rates relative to contact volume.

3. Methodology and Feature Engineering

3.1. Process Framework, CRISP-DM

This project followed the **Cross-Industry Standard Process for Data Mining (CRISP-DM)**, an iterative six-phase framework that ensures analytical rigor and alignment with business objectives:

1. Business Understanding, Define the goal: rank clients by subscription probability to optimize outbound call campaigns.
2. Data Understanding, Perform EDA to characterize distributions, identify class imbalance, and surface preliminary insights.
3. Data Preparation, Encode categorical variables, handle `unknown` values, engineer new features, and split data (80/20 train/test, stratified).
4. Modeling, Train and compare Logistic Regression, Random Forest, and XGBoost classifiers with class-imbalance handling.
5. Evaluation, Select the best model using F1-Score (minority class) and ROC-AUC as primary metrics.
6. Deployment, Propose a cloud architecture for production inference via a REST API on AWS.

3.2. Data Leakage Exclusion, `duration`

As discussed in Section 1, the feature `duration` was excluded from all predictive models. Its inclusion yields artificially high accuracy ($\approx 99\%$ ROC-AUC in benchmarks) but is operationally useless because the call duration is only known *after* the call completes.

3.3. Handling Class Imbalance

With only 11.27% positive cases, two complementary strategies were applied:

- Logistic Regression / Random Forest: `class_weight='balanced'` , automatically scales loss weights inversely proportional to class frequency, penalizing misclassification of the minority class more heavily.
- XGBoost: `scale_pos_weight` tuned via GridSearchCV. The optimal value found was ≈ 3.94, reflecting the approximately 4:1 imbalance in the training split.

3.4. Feature Engineering

Two new binary features were derived from existing variables to capture business-relevant signals:

Table 3: Engineered features and their rationale

Feature	Derived from	Business rationale
`was_contacted`	`pdays` $\neq$ 999	Clients previously contacted (`pdays` $<$ 999) exhibit higher propensity to subscribe. This binary flag separates "new leads" from "warm leads" more cleanly than the raw 0–999 numeric range.
`is_retired`	`age` $>$ 60	The EDA showed retirees subscribe at above-average rates. This flag lets the model learn a segment-specific interaction without relying solely on continuous age.

3.5. Preprocessing Pipeline

All transformations were wrapped in a `scikit-learn Pipeline` to prevent data leakage between train and test sets:

- Numerical features: `StandardScaler` (zero mean, unit variance), required for Logistic Regression; neutral for tree methods.
- Categorical features: `OneHotEncoder` with `handle_unknown='ignore'` to gracefully handle unseen categories at inference time.

3.6. Model Selection and Hyperparameter Tuning

Three model families were evaluated:

Table 4: Models evaluated and their tuning strategy

Model	Configuration
Logistic Regression	Baseline, `C=1`, `solver='lbfgs'`, `class_weight='balanced'`. Selected for interpretability and as a performance floor.
Random Forest	`n_estimators=100`, `max_depth=None`, `class_weight='balanced'`. Captures non-linear interactions; risk of overfitting.
XGBoost (Base)	Default parameters + `scale_pos_weight` calculated from class ratio.
XGBoost (Tuned)	GridSearchCV over `n_estimators, max_depth, learning_rate, subsample, colsample_bytree, scale_pos_weight`. CV scoring: `f1` (minority class). Best parameters: $n = 200$, depth$=$ 3, lr$=$ 0.05, subsample$=$ 0.9, colsample$=$ 0.9, spw$\approx$ 3.94.

4. Modeling Results

4.1. Model Comparison Summary

The table below summarizes performance on the held-out 20% test set (8,238 samples; 928 positive). All metrics reported for the **positive class** (`y = "yes"`).

Table 5: Model performance comparison, positive class (`yes`)

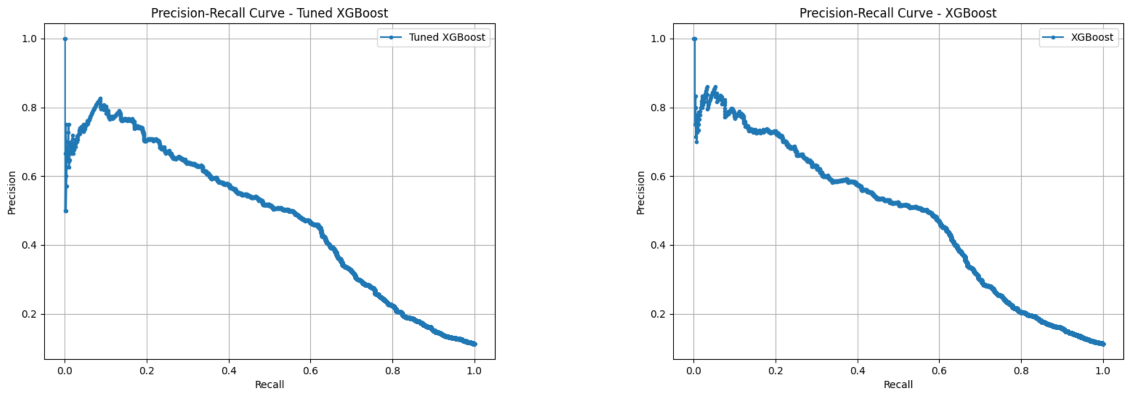
Model	ROC-AUC	F1	Precision	Recall	Key implication
Logistic Regression	0.8011	0.4690	0.37	0.65	Best recall; highest opportunity capture.
Random Forest	0.7776	0.3701	0.57	0.27	High precision but misses most subscribers.
XGBoost (Base)	0.8097	0.4874	0.39	0.64	Good base; better F1 than Logistic Reg.
Tuned XGBoost	0.8136	0.5256	0.47	0.59	Best overall balance of precision and recall.

Model Selection Rationale

The Tuned XGBoost is selected as the production model. It achieves the highest ROC-AUC (0.8136) and F1-Score (0.5256), offering the best balance between capturing subscribers (*Recall*) and avoiding wasted calls (*Precision*). Compared to the base XGBoost, GridSearchCV tuning improved F1 by +0.038 and Precision by +0.08, with a modest Recall reduction (−0.05) that eliminates many false positives.

4.2. Precision-Recall Curves

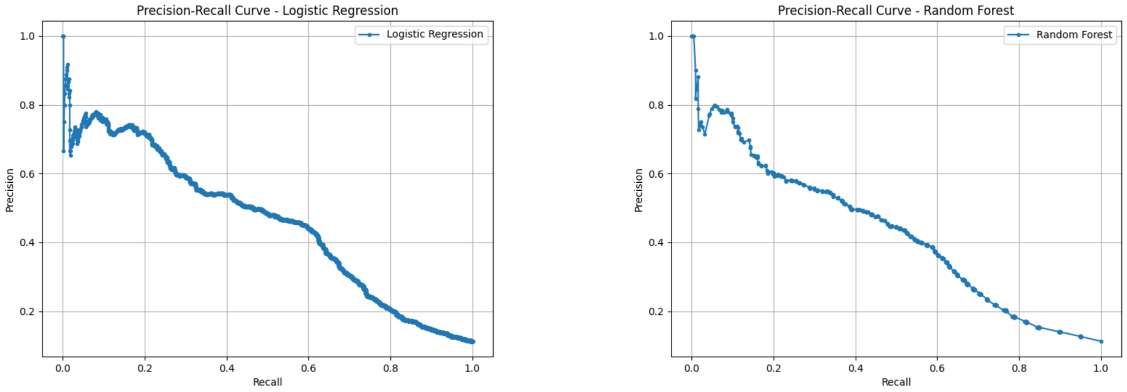
Precision-Recall curves are the primary diagnostic tool when working with imbalanced datasets, they directly show the trade-off between identifying true subscribers and avoiding false alarms across all classification thresholds.



(a) Tuned XGBoost, selected model. Highest area under the PR curve.

(b) Base XGBoost, strong but slightly lower AUPRC than tuned version.

Figure 17: Precision-Recall curves, XGBoost models

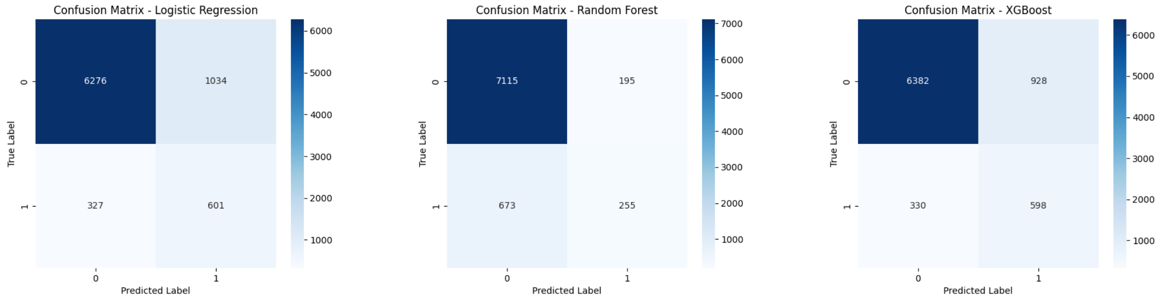


(a) Logistic Regression, high recall at the cost of many false positives.

(b) Random Forest, high precision but very low recall; misses most subscribers.

Figure 18: Precision-Recall curves, Logistic Regression and Random Forest

4.3. Confusion Matrices



(a) Logistic Regression

(b) Random Forest

(c) XGBoost (tuned)

Figure 19: Confusion matrices on the test set (8,238 samples)

Reading the matrices:

- Logistic Regression: High true-positive rate (high Recall) but many false positives, calls wasted on unlikely subscribers.
- Random Forest: Very few false positives (high Precision) but catches only 27% of actual subscribers, most opportunities missed.
- Tuned XGBoost: Best compromise, catches 59% of subscribers (547 true positives) with meaningful precision (47%), significantly reducing wasted calls vs. Logistic Regression.

4.4. Feature Importance

Feature importance scores reveal *which variables drive predictions*, directly informing business strategy.

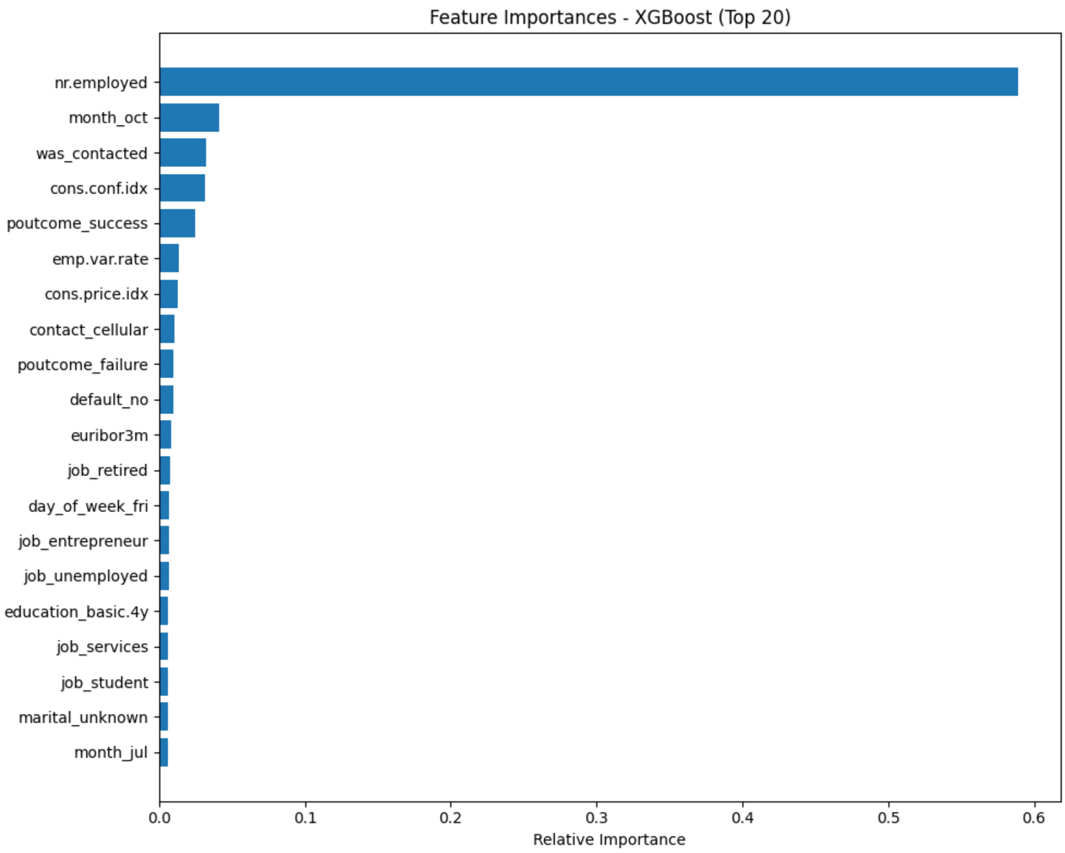


Figure 20: XGBoost feature importance (gain-based). Macroeconomic indicators dominate: `euribor3m` and `nr.employed` are the most predictive. The engineered feature `was_contacted` also ranks highly, confirming its value.

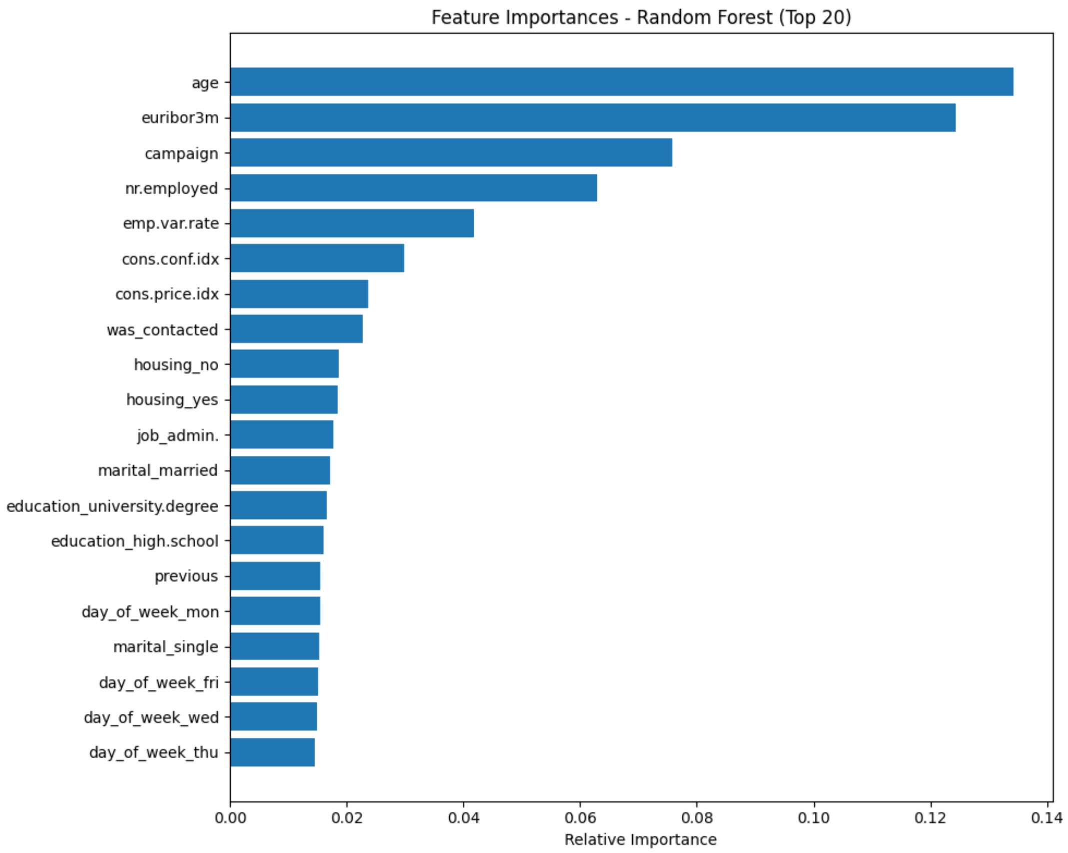


Figure 21: Random Forest feature importance (mean decrease impurity). Consistent pattern: macroeconomic variables lead, followed by contact-history features.

Top Predictors Summary

1. `euribor3m`, Euribor 3-month rate is the single most important predictor. Low rates ⇒ high subscription probability.
2. `nr.employed`, Number of employees (proxy for economic cycle). Economic slowdowns drive clients toward safe savings instruments.
3. `was_contacted`, Engineered feature. Prior contact history is a strong positive signal (warm leads convert better).
4. `poutcome_success`, Previous campaign success is the single strongest categorical predictor.
5. `month_*`, Seasonal effects (Mar, Sep, Oct, Dec) captured as one-hot encoded month dummies.

5. Business Recommendations

5.1. Strategic Targeting, Focus on High-Probability Segments

Deploying the Tuned XGBoost model enables the bank to **rank all prospective clients** by their estimated subscription probability before any call is made. The recommended operational strategy is:

1. Target the Top 30% by score. Simulation shows that concentrating outreach on the highest-ranked 30% of clients captures the majority of actual subscribers while reducing total call volume by 70%. This directly translates into lower cost-per-acquisition and higher agent productivity.

2. Prioritize “warm leads” with successful prior outcomes. Clients where `poutcome = success` should be moved to the top of the call queue regardless of their model score, empirical conversion rates for this group exceed 60%.

3. Time campaigns to macroeconomic windows. Actively monitor the `euribor3m` rate. When rates fall below 2%, trigger intensive campaign cycles. When rates are above 4%, scale down outreach to avoid wasted spend.

4. Leverage seasonal peaks. Allocate increased call-center capacity to March, September, October, and December, the months with the highest observed conversion rates relative to contact volume.

5. Limit contact frequency. The `campaign` feature shows diminishing returns beyond 3 contact attempts. Implement a hard cap of 3 calls per client per campaign cycle to reduce churn risk.

5.2. Operational KPIs and A/B Testing

Before full rollout, we recommend a controlled A/B experiment:

Table 6: Proposed A/B test design

Dimension	Control (A)	Treatment (B)
Lead selection	Random / existing process	Top 30% by XGBoost score
Sample size	5,000 clients	5,000 clients
Duration	4 weeks	4 weeks
Primary metric	Conversion rate	Conversion rate
Secondary metrics	Cost per subscription, calls per conversion	Cost per subscription, calls per conversion

Success criterion: Treatment conversion rate exceeds control by a statistically significant margin ($p < 0.05$, minimum detectable effect: +3 percentage points).

5.3. Continuous Improvement, Feedback Loop

The model is not a one-time artifact. To maintain and improve performance:

- Monthly retraining: Append new campaign results to the S3 Data Lake and retrain the model on the rolling 24-month window.
- Drift monitoring: Use SageMaker Model Monitor to detect when the incoming client profile distribution deviates significantly from the training distribution, a signal to retrain urgently.
- Outcome integration: Record call outcomes (subscribed, refused, not reached) and feed them back as training labels.

6. Cloud Deployment Architecture

6.1. Architecture Overview

We propose a serverless, event-driven architecture on Amazon Web Services (AWS) covering the full ML lifecycle: data ingestion, model training, scheduled batch inference, and observability. The architecture follows a Batch (Scale-to-Zero) inference pattern rather than a persistent real-time endpoint, a deliberate design choice motivated by the nature of telemarketing campaigns (see Section 6.4 for full justification).

The core data flow is organised in two horizontal tiers:

1. **Training tier** (top): raw data lands in S3, is processed by AWS Glue ETL, triggers a SageMaker Training Job, and the resulting model artifact is versioned in the Model Registry.

2. **Inference tier** (bottom): an EventBridge Scheduler fires every night, invokes SageMaker Batch Transform against the full client list in S3, and writes a ranked, scored CSV back to S3 for the call-center team. Compute resources spin up only for the duration of the batch job (typically 5–15 minutes) and scale to zero immediately after.

6.2. Architecture Diagram

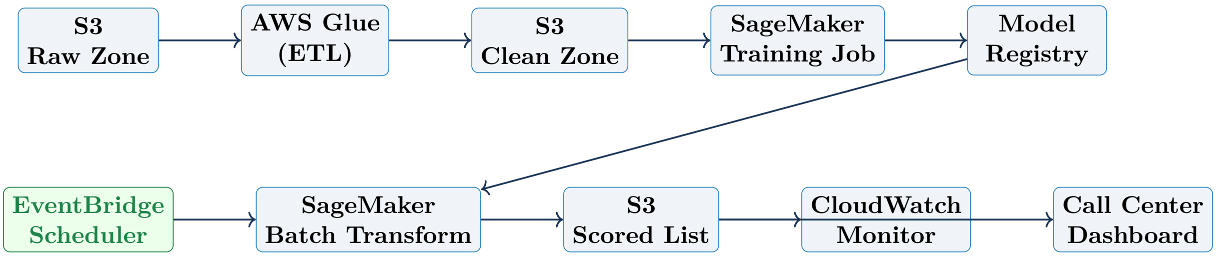


Figure 22: AWS Batch inference architecture, EventBridge triggers Batch Transform nightly; compute scales to zero after each run

6.3. Service Breakdown

6.3.1. Data Storage and Processing

- Amazon S3, Two-zone Data Lake: (1) *Raw Zone* for original CSV/data dumps; (2) *Clean Zone* for processed, model-ready feature sets; (3) *Output Zone* for the nightly scored client list.

- AWS Glue, Managed ETL for data cleaning, one-hot encoding, feature engineering, and schema cataloguing. Jobs trigger automatically on new data arrival via S3 event notifications.

6.3.2. Model Training and Registry

- Amazon SageMaker Studio, Interactive notebooks for experimentation and hyperparameter searches.

- SageMaker Training Jobs, Managed compute for reproducible model training with full experiment tracking.
- SageMaker Model Registry, Versioned artifact store; enables controlled promotion from staging to production with approval gates.

6.3.3. Batch Inference (Primary Pattern)

- Amazon EventBridge Scheduler, Cron-based job trigger (e.g. `0 2 * * *` for 02:00 AM daily). No persistent infrastructure required.
- SageMaker Batch Transform, Spins up compute, scores the full client list in S3, writes a ranked CSV with subscription probability scores back to the Output Zone, and terminates automatically. **Compute is active only during the batch run (typically 5–15 minutes), achieving Scale-to-Zero cost behaviour.**

6.3.4. Monitoring and Observability

- Amazon CloudWatch, Tracks batch job duration, failure alarms, and output file size anomalies.
- SageMaker Model Monitor, Scheduled data quality check on the incoming feature distribution; triggers retraining alert when drift is detected.

6.4. Architecture Evaluation and Design Rationale

Architecture Decision Record

A straight forward design included a SageMaker Real-time Endpoint served behind API Gateway and AWS Lambda. After evaluating the use case, it was **refactored to a Batch Transform pattern**. The rationale is detailed below.

1. **Use-case alignment.** Telemarketing campaigns are scheduled, planned processes. Call-center agents need a *prioritised call list at the start of the working day*, not sub-second predictions for individual ad-hoc requests. A real-time endpoint solves a problem that does not exist in this workflow.

2. **Cost optimisation (70–90% reduction).** A persistent SageMaker real-time endpoint incurs hourly charges 24/7, regardless of whether predictions are being requested. Batch Transform activates compute only for the duration of the scoring job (5–15 minutes per night) and then shuts down automatically, achieving true *Scale-to-Zero* cost behaviour.

3. **Volume handling.** Scoring 41,000+ records in a single API Gateway/Lambda chain is fragile: Lambda has a 15-minute timeout, API Gateway has a 29-second response limit, and both add per-request overhead. Batch Transform is designed for exactly this workload, processing the entire file in a single managed job without timeout risk.

6.5. Cloud Portability, Azure Equivalents

The architecture maps cleanly to Microsoft Azure, enabling migration or multi-cloud deployment without redesigning the conceptual pipeline:

Table 7: AWS to Azure service equivalence map

Layer	AWS Service	Azure Equivalent
Object Storage	Amazon S3	Azure Blob Storage (ADLS Gen2)
ETL / Data Prep	AWS Glue	Azure Data Factory (ADF)
Model Training	SageMaker Training Jobs	Azure ML Compute Clusters
Experiment Tracking	SageMaker Studio	Azure ML Studio
Model Registry	SageMaker Model Registry	Azure ML Model Registry
Batch Scoring	SageMaker Batch Transform	Azure ML Batch Endpoints
Scheduler / Trigger	Amazon EventBridge Scheduler	Azure Logic Apps / ADF Trigger
Monitoring	CloudWatch + Model Monitor	Azure Monitor + Azure ML Data Drift
IAM / Security	AWS IAM + VPC + KMS	Azure RBAC + VNet + Key Vault

The batch-first pattern is arguably *even simpler* on Azure: Azure ML Batch Endpoints natively support scheduled invocations via ADF Pipeline triggers, removing the need for a separate scheduler service.

6.6. Security Considerations

- IAM roles with least-privilege permissions for every service boundary.
- S3 bucket policies with KMS encryption at rest; TLS 1.2+ for all data in transit.
- VPC Endpoints ensuring inter-service traffic never traverses the public internet.
- S3 Output Zone restricted to read-only access for the call-center dashboard role.

7. Generative AI Strategy

7.1. Recommendation Summary

Short Answer

Do NOT use Generative AI for the core binary prediction task. DO use it as a complementary layer to amplify the campaign's impact once the predictive model has identified high-probability leads.

7.2. Why Classical ML is Superior for the Core Task

The central problem is a **binary classification on structured tabular data**, a domain where tree-based ensemble methods (XGBoost, Random Forest) consistently outperform Large Language Models (LLMs) for the following reasons:

- Efficiency: XGBoost trains in seconds on consumer hardware; an LLM requires GPU infrastructure and costs orders of magnitude more per inference.

- Interpretability: Feature importance scores provide clear, auditable explanations. LLM predictions on tabular data are opaque.
- Precision: LLMs are known to hallucinate and perform poorly on precise numerical reasoning (e.g., "`euribor3m` = 1.334, is this low?"). XGBoost handles this natively.
- Regulatory compliance: Banking regulators require explainability for credit/marketing decisions. Classical models provide this out of the box; LLMs do not.

7.3. Where Generative AI Adds Real Value

GenAI should operate *downstream* of the predictive model, enhancing how the campaign is *executed* rather than who is targeted.

7.3.1. A. Personalized Sales Script Generation

Once the XGBoost model identifies a client with high subscription probability, a prompt-engineered LLM (e.g., GPT-4o, Claude 3.5, Llama 3) can generate a dynamic, personalized call script:

- *Input:* Client profile (age, job, education, segment, model score, macroeconomic context).
- *Output:* "*Good morning [Name], given your background as a [Profession] and the current interest rate environment, our 12-month term deposit at [Rate]% might be particularly attractive compared to your current savings options...*"

This increases agent effectiveness and reduces script preparation time.

7.3.2. B. Post-Call Sentiment and Objection Analysis

- Transcribe calls via Amazon Transcribe (Speech-to-Text).
- Apply an LLM to extract: (1) why clients declined, (2) which arguments resonated, (3) objection categories and frequency.
- Feed these insights back as new features to retrain the predictive model (*Feature Engineering feedback loop*).

7.3.3. C. Conversational AI, Digital Channels

For clients who prefer digital interaction, a GenAI-powered chatbot can:

- Answer questions about term deposit terms and conditions.
- Guide the client through the subscription process.
- Escalate to a human agent only for complex cases, reducing call-center load while improving coverage.

7.4. Hybrid Architecture: The Winning Strategy

Table 8: Discriminative AI vs. Generative AI, division of responsibilities

Dimension	Discriminative AI (XGBoost)	Generative AI (LLM)
Question answered	*Who to call?*	*What to say?*
Input	Structured tabular data	Client profile + conversation history
Output	Probability score (0–1)	Natural language text (script, analysis)
Role in campaign	Lead prioritization	Personalization and engagement
Technology	XGBoost + SageMaker	GPT-4o / Claude / Llama via API
Cost	Low (milliseconds per request)	Medium (LLM API tokens)

Hybrid Strategy Conclusion

The optimal architecture chains both AI paradigms sequentially: Step 1, XGBoost ranks all clients by subscription probability. Step 2, Top-ranked clients enter a GenAI-powered personalization layer that generates bespoke scripts and handles digital interactions. This combination maximizes conversion rates while minimizing operational cost.