Customer Churn - Problem Formulation

1) Business problem

- Reduce addressable customer churn by predicting which active customers are at high risk of leaving, so retention teams can intervene before renewal or irreversible usage drop.
- Formulate as a binary classification: churned vs. retained. Select operating thresholds based on outreach capacity (e.g., contact top-k highest risk each week).
- Constrain to customers present in the engagement stream; reconcile label from multiple sources if needed.

2) Key business objectives

- **Improve retention**: flag at-risk customers in time for offers, education, or service recovery.
- **Optimize spend**: focus on customers that grant maximum leverage for our time and money spent
- **Reliability**: automated ingestion → validation → preparation → transformation → feature store → model training → orchestration.
- **Deliverables**: clean EDA dataset, feature matrix, versioned model + monitoring hooks.

3) Key data sources and attributes (merged for EDA)

- Two Mockaroo datasets to be merged on **customer_id**:
    - **subscriber_engagement** — behavior, plan, economics, comms.
    - **customer_satisfaction** — profile, contract, support, NPS, billing.
- **Merge approach**:
    - Start with an **inner join** for modeling; keep a **left join** (from engagement) to assess coverage loss and bias.
    - **Label rule**: if both sources include `churned`, adopt engagement as canonical; log mismatches for review.
- **Feature themes** (examples):
    - **Tenure & recency**: days since signup, days since last login.
    - **Plan & spend**: plan one-hot, monthly_spend, avg_monthly_bill, discounts, ratios.
    - **Engagement**: session length, email opens, login recency buckets.
    - **Friction**: support tickets, support calls, payment delays.
    - **Profile & contract**: age bands, region, contract_type, auto_renew.

4) Expected outputs

- **Clean EDA dataset**: harmonized schema, standardized dates, validated categories, imputed/flagged nulls, deduped customers, single authoritative label.
- **Feature set for ML**: encoded/scaled features with saved preprocessing (pipeline object) ready for train/val/test splits and feature store registration.
- **Model artifact**: versioned binary (e.g., sklearn pipeline), inference contract, and promotion path.

5) Evaluation metrics

- **Classification**: Accuracy, Precision, Recall, F1 (report for positive class = churn).
- **Threshold-free**: ROC-AUC, PR-AUC.
- **Ranking**: Precision@k and lift at deciles for outreach scenarios.
- **Calibration**: Brier score / calibration curves if probabilities drive spend.

6) Data dictionaries

subscriber_engagement — Data Dictionary

Field	Type	Null %	Allowed / Format	Min	Max	Notes
customer_id	MongoDB ObjectID	0				
signup_date	Datetime	0	`%Y-%m-%d`	06/11/2020	08/17/2025	
last_login_date	Datetime	2	`%-m/%-d/%Y`	06/11/2020	08/17/2025	Different date format vs signup_date
subscription_plan	Custom List	1	Basic, Pro, Enterprise			
monthly_spend	Number	0		5	500	
support_tickets_last_90d	Number	0			25	
avg_session_length_minutes	Number	1		1	120	
email_opens_last_30d	Number	0			25	
auto_renew_enabled	Boolean	0				
churned	Boolean	0				

customer_satisfaction — Data Dictionary

Field	Type	Null %	Allowed / Format	Min	Max	Notes
customer_id	MongoDB ObjectID	1				Has ~1% nulls; affects join
age	Number	0		18	75	
region	Custom List	0	Urban, Semi-Urban, Rural			
contract_type	Custom List	1	Monthly, Quarterly, Annual, Annul			Contains typo ‘Annul’ → normalize to ‘Annual’
avg_monthly_bill	Number	0		10	200	
payment_delay_days	Number	0			30	
customer_support_calls_last_6m	Number	0			15	
net_promoter_score	Number	0		-100	100	
discounts_received_last_6m	Number	0			5	
churned	Boolean	0				

7) First-pass validation checklist (tailored to the two schemas)

A. Cross-dataset

- **Join key**: `customer_id` must be present in engagement (100%) and typically present in satisfaction; quarantine rows with missing IDs in satisfaction before joining.
- **Duplicate customers**: assert single row per `customer_id` per source; if duplicates exist, deterministically select most recent by `last_login_date`/`signup_date`.

B. subscriber_engagement

- **Dates**: normalize formats (`signup_date` uses `%Y-%m-%d`, `last_login_date` uses `%-m/%-d/%Y`); coerce to UTC-naive dates, then derive features.
- **Categoricals**: `subscription_plan` ∈ {Basic, Pro, Enterprise}; clean up unknowns
- **Ranges**:
  - `monthly_spend` ∈ [5, 500] with 2 decimals.
  - `support_tickets_last_90d` ∈ [0, 25].
  - `avg_session_length_minutes` ∈ [1, 120] (treat 0 only if business rule explicitly allows).
  - `email_opens_last_30d` ∈ [0, 25].
- **Booleans**: `auto_renew_enabled`, `churned` strictly boolean; map any string variants.
- **Nulls**: handle ~1–2% nulls in `last_login_date` and `subscription_plan`/`avg_session_length_minutes` via imputation + missing-indicator flags.

C. customer_satisfaction

- **ID presence**: `customer_id` has ~1% nulls → exclude from join or impute only if a trusted mapping exists.
- **Categoricals**:
  - `region` ∈ {Urban, Semi-Urban, Rural}.
  - `contract_type` ∈ {Monthly, Quarterly, Annual}; map **"Annul"** → **"Annual"** before encoding.
- **Ranges**:
  - `age` ∈ [18, 75].
  - `avg_monthly_bill` ∈ [10, 200] with 1 decimal.
  - `payment_delay_days` ∈ [0, 30].
  - `customer_support_calls_last_6m` ∈ [0, 15].
  - `net_promoter_score` ∈ [−100, 100].
  - `discounts_received_last_6m` ∈ [0, 5].
- **Booleans**: `churned` strictly boolean.
- **Outliers**: flag boundary values (e.g., `age`=18 or 75, `NPS`=±100) for distribution checks; ensure business plausibility.

D. Derived features

- **Derived**: tenure (days since `signup_date`), recency (days since `last_login_date`), numeric ratios (spend/bill), interaction terms (auto_renew × tenure).