Contents

Insights

Modeling & Evaluation — User 0 Identification

Appendix

Insights

1) Executive summary

We explored User 0's sessions to understand **what they visit**, **when they browse**, **how long they stay**, and **how concentrated/diverse** their behavior is. We found a **strong site-driven pattern** (specialized, recurring site bundles), **distinct meta signals** (unique locale; constrained locations/OS), **mild temporal tendencies**, and **no major difference in browsing diversity** vs others. These observations directly inform our feature set (TF-IDF on sites, signature-site flags, meta one-hots, time features, and session-length stats).

2) Data & setup (what a "session" is)

Each row is **one session** with:

- **Meta**: `browser`, `os`, `locale`, `gender`, `country`, `city`, `date` (GMT), `time` (GMT).
- **Behavior**: up to 15 `(site_i, length_i)` pairs (seconds on site).

Why this matters for features: the prediction unit is the **session**, so all engineered features aggregate or summarize the 15 site slots + metadata into **one vector per session**.

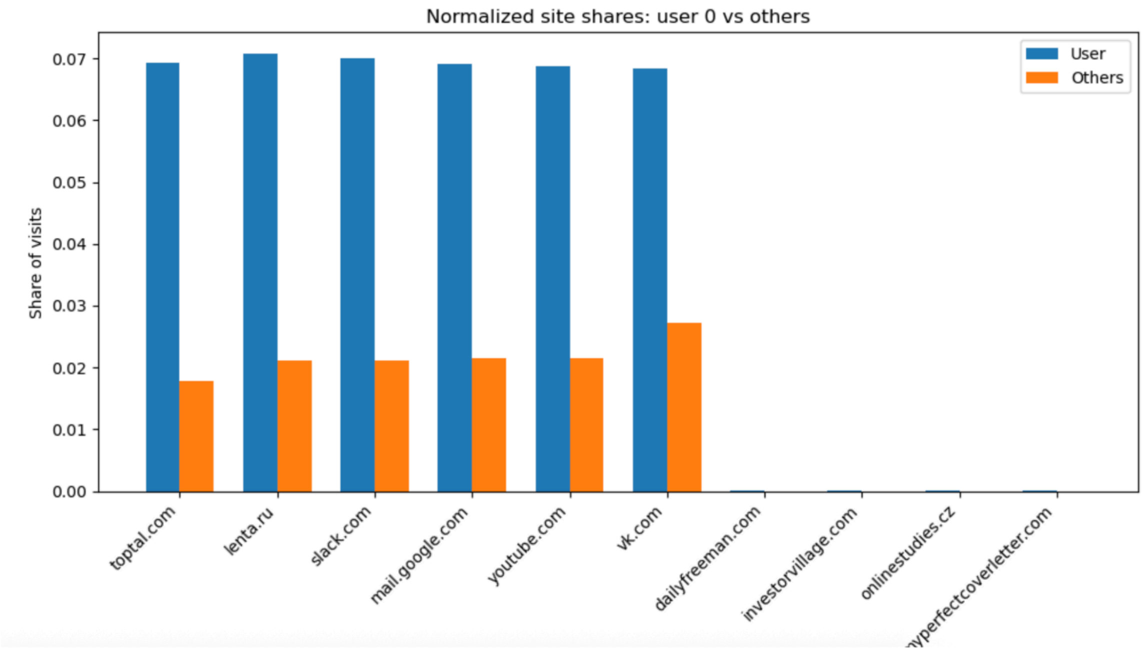
3) Who is User 0 (stable meta signals)

- **Locale**: *ru-RU* only (unique to this user).
- **Locations**: 4 out of 21 global cities (Paris, Toronto, Chicago, Singapore).
- **OS**: 2 out of 6 global OS (Ubuntu, Windows 10).
- **Browser/Gender**: Male

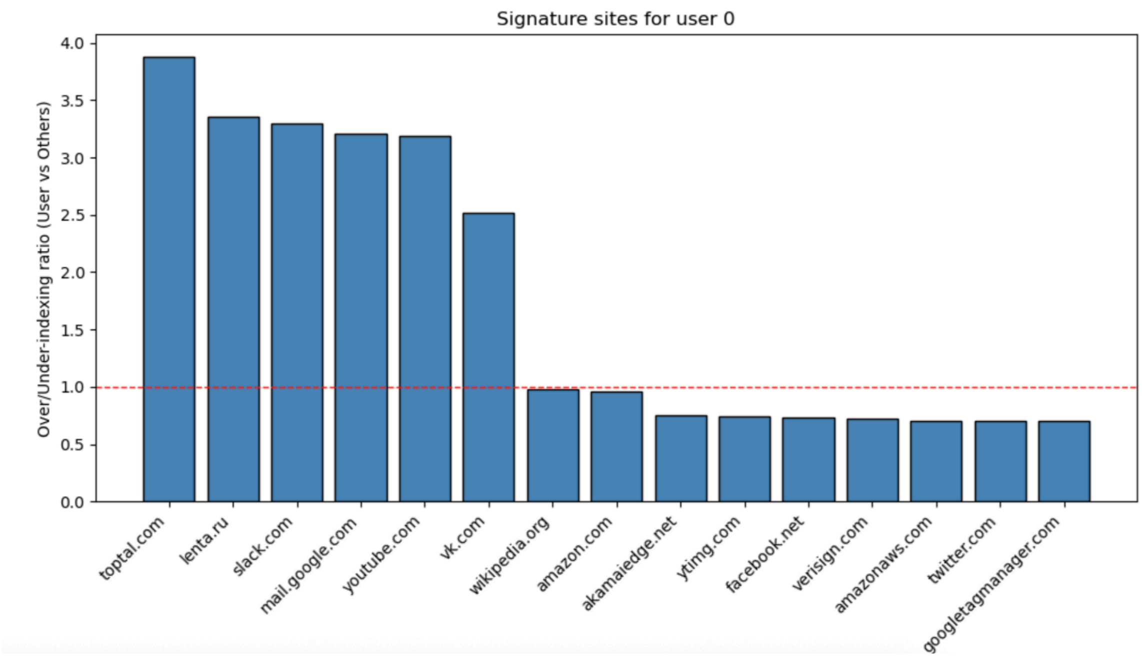
Why this matters for features:
We encode `locale`, `country`, `city`, `browser`, `os`, `gender` with **One-Hot Encoding** (OHE). Locale and the restricted city/OS set give **high-precision meta signals**; others provide weaker but still useful context.
Feature link: `meta::` one-hot vectors via the class's `OneHotEncoder`.

4) What they visit (site distribution & specialization)

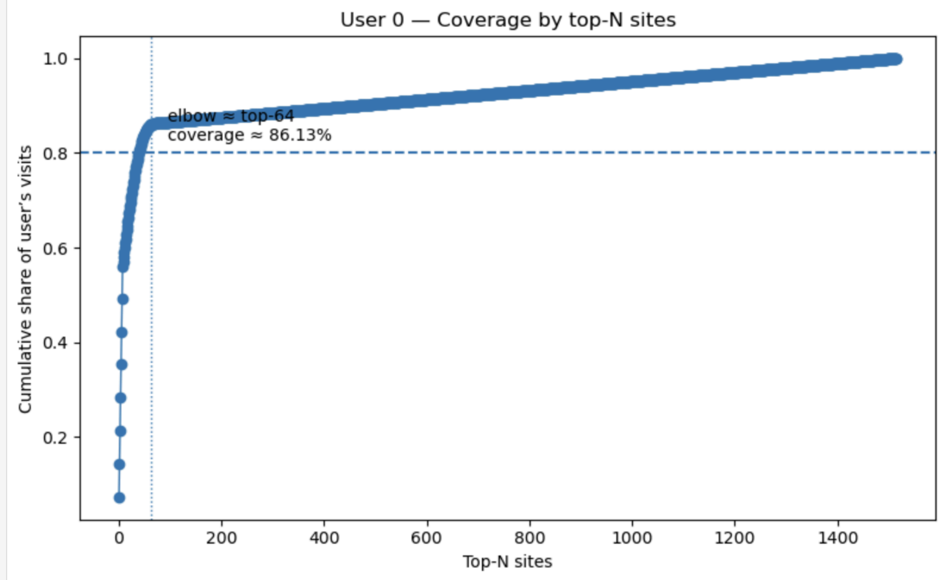
- There are **~158k unique sites** globally; User 0 has **~1.5k unique** lifetime.
- User 0 shows clear **specialization**: a set of sites recur frequently (e.g., toptal.com, vk.com, amazon.com, lenta.ru, slack.com, wikipedia.org).



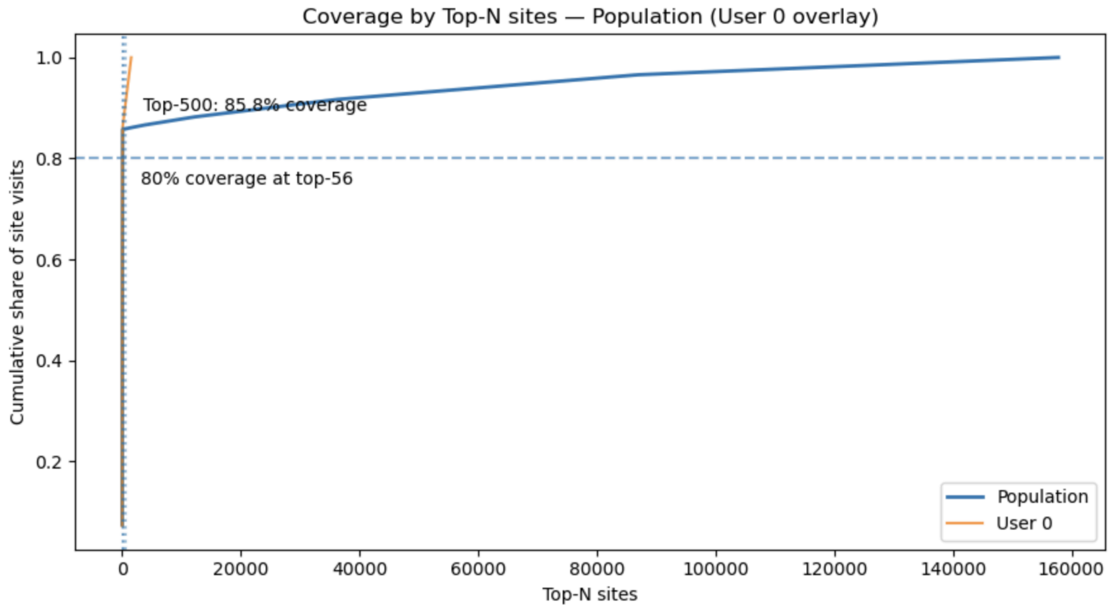
- **Over/under-indexing** vs population highlights “signature sites” where User 0’s **share of visits/time** is much higher than others.



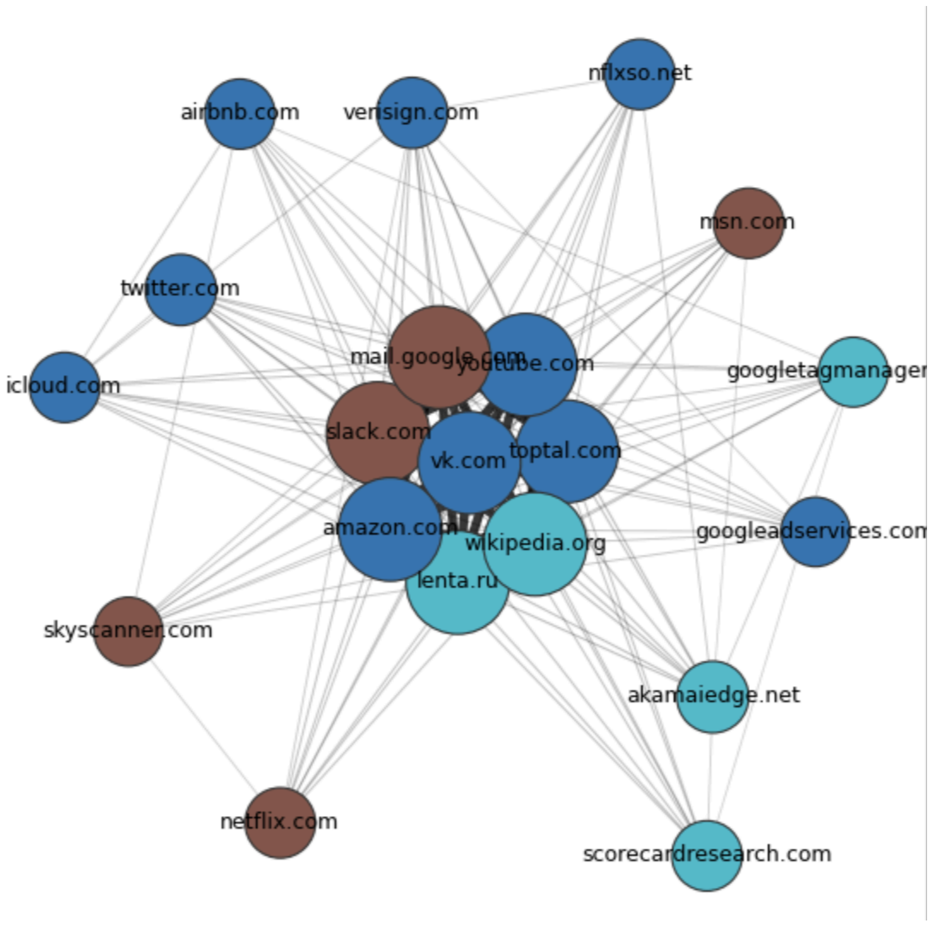
- **Site Coverage:** Although User 0 touched ~1.5k unique sites, their attention is highly concentrated. The top ~60–80 sites already account for ≈86% of all visits, after which the curve flattens into a long tail of rarely visited domains. In other words, most sessions are built from a compact “core set” of sites, with occasional forays into the tail.



The story is the same for other users, as per the coverage by top-N sites plot generalised:



- **Co-occurrence bundles**: within a session, certain sites tend to appear together (e.g., {lenta.ru, vk.com, …} with ad/infra domains like googleadservices.com, icloud.com). The co-occurrence network confirms **repeatable bundles**.



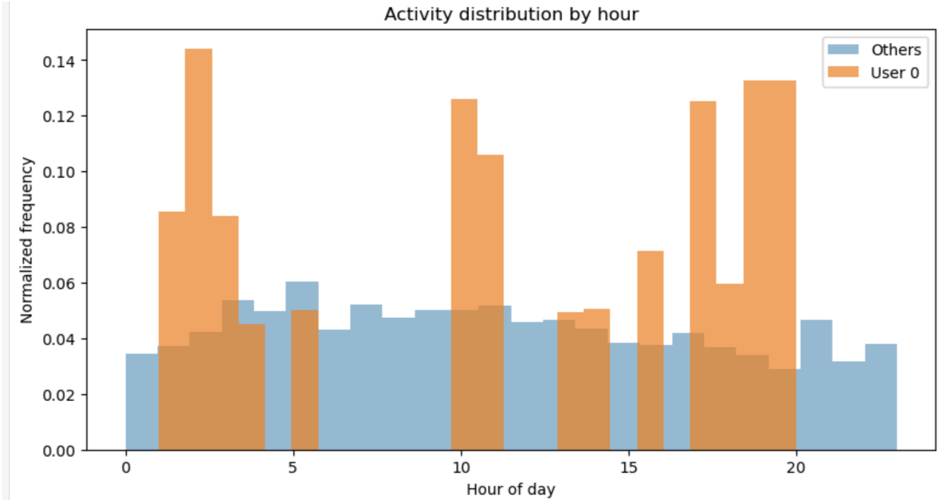
Why this matters for features:

- We treat the session's sites as a "document" and build **site TF-IDF** to emphasize **rare, distinctive** sites for identification.

- We add **User-0 signature-site flags** (top N sites most used by User 0) as binary features for crisp, high-precision triggers.
  Feature links: `tfidf::` (TfidfVectorizer over session sites) and `user_sig::` (signature site one-hots).
- See appendix for more details on the construction

5) When they browse (temporal patterns)

- **Hour-of-day**: mild preference for specific hours; slightly **longer sessions at night**, but not a large overall shift.



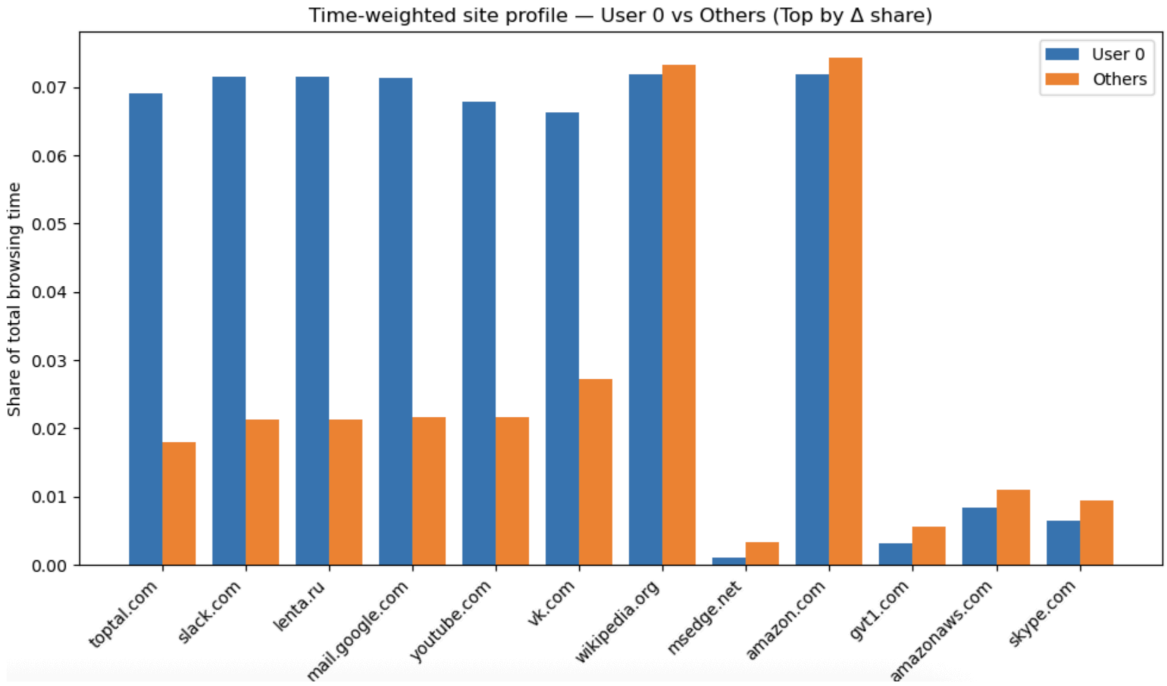
- **Day-of-week / Month**: routine patterns exist but are secondary to site choice.

Why this matters for features:
We encode time with both **categorical** and **cyclic** signals to capture daily periodicity and simple routines:

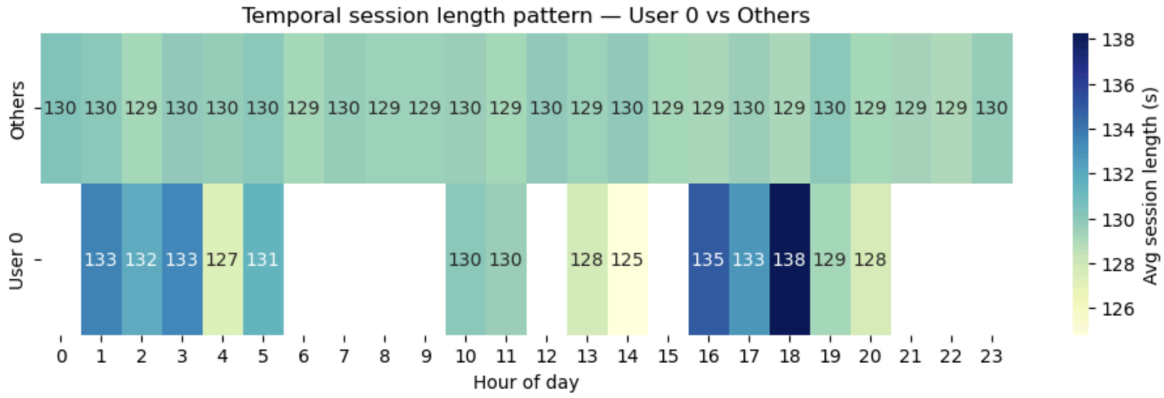
- `hour`, `dow`, `month`, `is_night`
- `hour_sin`, `hour_cos` (cyclic hour)
  Feature links: `time::hour`, `time::dow`, `time::month`, `time::is_night`, `time::hour_sin`, `time::hour_cos`.

6) How long they stay (session length profile)

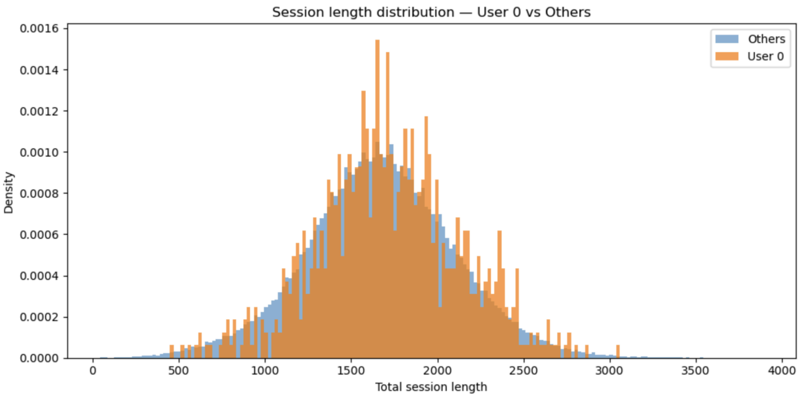
- Per-site, User 0 spends **more time** on a subset of sites; others are skimmed.

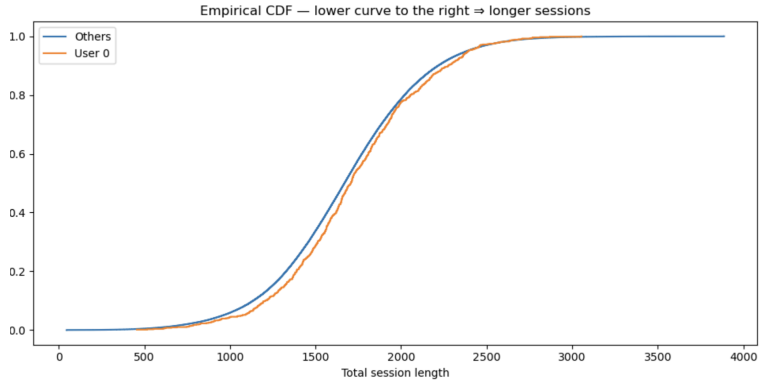


- **Per-session totals**: distribution indicates **slightly longer** night sessions, but **no dramatic global shift** vs others on average.



- **Total session time**: User0 tends to have longer total session time than others



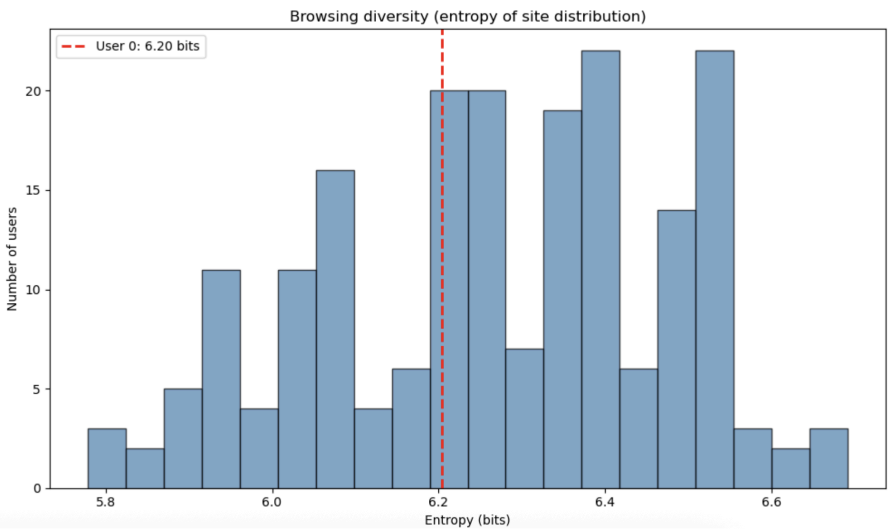


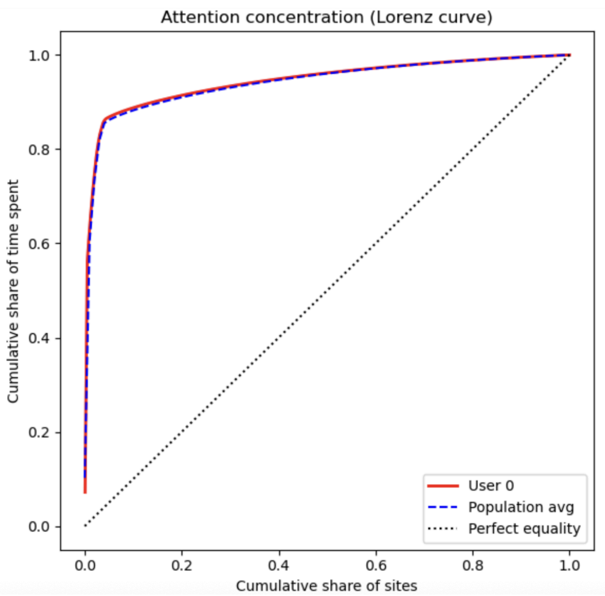
Why this matters for features:
We summarize the 15 length slots into robust session statistics to capture **depth** and **style**:

- `total_length`, `mean_length`, `median_length`, `std_length`, `max_length`, `min_length`
- `n_sites` (unique sites in the session), `n_visits` (non-null site slots)
  Feature links: `len::total_length`, `len::mean_length`, …, `len::n_sites`, `len::n_visits`.

7) How diverse the browsing is (entropy)

- **Entropy of site distribution** for User 0 ≈ **population mean**.





- Interpretation: User 0 is **not** unusually "generalist" or "specialist" overall vs others by entropy alone; the **identity signal is in *which* sites** (and bundles), not in aggregate diversity.

> **Why this matters for features:**
> Entropy itself is not used directly; instead the **TF-IDF + signature flags** capture the meaningful **which-sites** signal more effectively than a single diversity scalar.

8) From insights to features (recap mapping)

Insight	Feature choice	Rationale
Specialized, recurring sites/bundles	TF-IDF over session sites	Up-weights distinctive/rare sites; robust to long tails.
Specific "signature" sites for User 0	Binary flags for top User-0 sites	High-precision triggers when those personal sites appear.
Mild daily/weekly routines	Hour, DOW, Month, is_night; hour_sin/hour_cos	Captures periodic patterns without overfitting exact hours.
Longer on some sites; slight night depth	Session-length stats: total/mean/median/std/min/max; n_sites; n_visits	Encodes depth and variability compactly at session level.
Strong, stable meta (locale; city; OS)	One-hot: locale, country, city, browser, OS, gender	Crisp context; disambiguates similar sessions across time.

9) Modeling implications (brief)

- The **strongest signal is site-driven**; TF-IDF + signature flags should carry most of the lift.
- **Meta** (locale, city, OS) acts as a stabilizer and disambiguator across time.
- **Time** and **length stats** refine the decision boundary (especially in close calls).
- With severe class imbalance, use **PR-AUC**, class weighting, and thresholding by precision/recall targets.

10) What we did *not* include (on purpose)

- Raw per-site one-hots for the entire 160k vocabulary (too sparse/fragile). TF-IDF + signature subset is a better bias-variance trade-off.
- Entropy as a direct feature (it didn't differentiate the user; the **composition** of sites did).

11) Takeaway

User 0's identity is best captured by **which sites** they visit (and **which combinations** appear together), supported by **unique meta signals** and **light temporal/length cues**. The final feature set mirrors this: **TF-IDF + signature site flags** for the core signal, **OHE meta** for stable context, and **time/length summaries** for behavioral shape—exactly what `SessionFeatureEngineer` produces.

Modeling & Evaluation — User 0 Identification

1) Problem framing & class imbalance

Goal: predict whether a session belongs to **user_id = 0**. The data are highly imbalanced (~800 positives out of ~160k sessions). With so few positives, **accuracy** or **ROC-AUC** can look great even for models that miss the user entirely. We optimize for **recall** (catch user-0) while also maximizing **precision** (reduce false alerts).

Metric choice: we focus on **Precision–Recall** metrics—Average Precision (PR-AUC) and **precision at a recall target**—not accuracy/ROC-AUC. PR-AUC reflects positive-class performance under extreme imbalance.

2) Time-aware data split

We keep causality: **TRAIN** = sessions **before 2019-01-01**; **TEST** = sessions **on/after** that date. All tuning happens on TRAIN; TEST is held out for the final check.

3) Features (session-level vector)

Each session becomes one feature vector via `SessionFeatureEngineer`:

- **tfidf::** site TF-IDF (session as "document" of sites)
- **user_sig::** binary flags for top User-0 sites
- **len::** total/mean/median/std/min/max, `n_sites`, `n_visits`
- **time::** hour, DOW, month, `is_night` (+ cyclic `hour_sin`, `hour_cos`)
- **meta::** one-hot for `locale/country/city/browser/OS/gender`

4) Hyperparameter search with forward-chaining CV

On TRAIN we use **ForwardTimeSplit** (train on earlier, validate on later) and **RandomizedSearchCV** for each model family (LightGBM, Logistic Regression). We score every candidate with:

- **AP (PR-AUC)**, and
- **Precision@Recall ≥ target** (refit uses this), so the winner directly optimizes our operating goal.

Imbalance handling: LightGBM uses `scale_pos_weight ≈ (#neg / #pos)` from TRAIN.

5) Decision threshold tuning (recall-first)

Models output probabilities; we need a **threshold**. Using the same time-forward folds, we find the threshold per fold that **meets the recall target** (optionally choosing the **highest-precision** one among those). We **aggregate** the per-fold thresholds (median / p25 / p10) to get a **robust final threshold** that tolerates drift.

6) Final TEST results (2019 holdout)

Two operating points from your sweep:

Threshold	Accuracy	Precision	Recall	F1	TN	FP	FN	TP	AP / ROC-AUC
0.14	0.996	0.6	**1.0000**	0.75	15350	51	**0**	79	**AP=0.9818** / ROC=0.9999
0.26	0.99	**0.67**	0.987	0.80	15364	**37**	1	78	**AP=0.9818** / ROC=0.9999

Interpretation:

- **High-recall mode (thr = 0.14): 100% recall** (no misses), precision **60%**; 51 FPs among ~15.4k negatives.
- **Balanced mode (thr = 0.26):** recall ~**98.7%**, precision **67.6%**; FPs drop from 51 → **37**, only **1** FN.

Both points are excellent under severe imbalance; **PR-AUC ≈ 0.982** shows strong ranking quality. Choose the threshold by business tolerance for misses vs alerts.

7) Validation discipline & leakage avoidance

- Cutoff-based split prevents **training on the future**.
- Feature engineering is **fit on TRAIN only**; TEST reuses fitted transformers.
- Thresholds are **derived on TRAIN** via forward-chaining; TEST is evaluated once.
- Categorical unknowns map to **“Unknown”**; no user IDs leak into features.

8) What a session vector looks like (structure)

Mostly **sparse**: `tfidf::[K]` + `user_sig::[N]` + `len::[8]` + `time::[5–7]` + `meta::[~C]`.

9) Operating guidance (picking the threshold)

Start with **recall-first** (e.g., 0.06) if missing any user-0 session is unacceptable. If alert volume is high, raise to **0.13** (or the smallest threshold achieving a **target precision** like ≥60% on recent validation).

Appendix

Site Distribution & Specialization: Feature Rationale (Illustrated)

This section explains, step by step, how we build the site-based features so that even a junior data scientist can follow. We use a tiny numeric example to illustrate TF-IDF and the User-0 signature-site flags.

3.1 What features we build from sites

1) TF-IDF over session sites (prefix: tfidf::)

We treat each session as a "document" made of site tokens (e.g., vk.com, lenta.ru). TF-IDF emphasizes distinctive sites that help identify User 0, while down-weighting ubiquitous sites that everyone visits.

2) User-0 signature-site flags (prefix: user_sig::)

Binary 0/1 features that fire when a session contains one of User 0's top-N personal sites. These provide crisp, high-precision triggers when those personal sites appear.

3.2 Mini numeric example (5 sessions)

Five sessions (2 from User 0, 3 from other users):

Session	User	Sites (tokens)
s1	U0	lenta.ru, vk.com, lenta.ru, wikipedia.org
s2	U0	lenta.ru, amazon.com
s3	Other	google.com, amazon.com
s4	Other	google.com, wikipedia.org
s5	Other	google.com

Step A — Document frequency (df) per site

Count in how many sessions each site appears at least once (N = 5 sessions):

Site	df (sessions containing site)
lenta.ru	2
vk.com	1
wikipedia.org	2
amazon.com	2
google.com	3

Step B — IDF (scikit-learn)

Formula: idf(site) = log((1 + N) / (1 + df(site))) + 1, with N = 5. Rounded to 3 decimals:

Site	idf
lenta.ru	1.693
vk.com	1.916
wikipedia.org	1.693
amazon.com	1.693
google.com	1.405

Step C — TF and TF-IDF for session s1

Session s1 tokens: lenta.ru (×2), vk.com (×1), wikipedia.org (×1). Raw TF × IDF:

Token	TF (count in s1)	TF × IDF (pre-normalization)
lenta.ru	2	3.386
vk.com	1	1.916
wikipedia.org	1	1.693

scikit-learn then L2-normalizes the vector within each session. Intuition: rarer sites (higher IDF, like vk.com here) receive more weight than very common sites (e.g., google.com in other sessions).

3.3 User-0 signature-site flags (binary)

From User 0's training sessions only, we rank their sites by frequency and pick the top-N (e.g., N=300). For illustration, suppose the top-3 are: lenta.ru, vk.com, amazon.com. We create 3 binary features:

• **user_sig::lenta.ru** — 1 if the session contains lenta.ru, else 0
• **user_sig::vk.com** — 1 if the session contains vk.com, else 0
• **user_sig::amazon.com** — 1 if the session contains amazon.com, else 0

Examples:

Session	Contains lenta.ru?	Contains vk.com?	Contains amazon.com?	[user_sig::lenta, vk, amazon]
s1	Yes	Yes	No	[1, 1, 0]
s3	No	No	Yes	[0, 0, 1]

3.4 What goes into the model

For every session row, we concatenate: (1) tfidf:: weights for the session's sites (sparse), (2) user_sig:: 0/1 flags for User-0 top sites, (3) len:: session-length stats (total/mean/median/std/min/max, n_sites, n_visits), (4) time:: hour, dow, month, is_night, hour_sin, hour_cos, (5) meta:: one-hot locale, country, city, browser, os, gender. This single vector represents the entire session.