Analysis Summary

Zlatko Golubović

November 2025

Introduction

This short report summarizes the comparison of "auto" vs "manual" open episodes. Results shown are produced by the simple python project. Numeric values and figures are supplied from the run-specific folder under `results/`.

Executive summary

Auto-open episodes show substantially longer typical durations than manual opens. The effect is robust to log-transformations and bootstrap percentile confidence intervals; however a small fraction of extreme sessions (top 1%) materially increases totals and attenuates some point estimates when included.

Approach

Episode durations are compared between two open types: `auto` and `manual`. Primary check is the per-user median duration (seconds) for each open type, with inference performed on paired per-user log-transformed medians using the Wilcoxon signed-rank test. Complementary checks include Welch t-test on log-transformed episodes and Mann–Whitney U on raw seconds. Bootstrap percentile confidence intervals (default $n_{\text{boot}} = 4000$, seed $= 42$) are computed for the median ratio (auto / manual) and for the multiplicative factor on the log scale.

Assumptions

- Event timestamps are monotonic per user and in UTC milliseconds.
- Each meaningful session has a matching open and a close; unmatched or overlapping events are possible and handled explicitly.
- Per-user medians are stable summaries that mitigate influence of heavy tails better than means.
- Log-transform uses $\log_{10}(\text{duration\_sec} + 1)$ to accommodate zeros and compress heavy tails.

Scope and robustness

This report focuses on results from the run named `regular_only_all` because it represents the primary, conservative analysis: it uses only regular episodes and retains all durations (no tail trimming). Wherever possible the numeric values shown (counts, medians, inferential test statistics, and bootstrap CIs) are drawn from that run.

To evaluate sensitivity the same pipeline was reran with alternate settings: (1) trimming the extreme tail (`regular_only_exclude_top1`), (2) including `double_open` episodes by folding them into regular episodes (`include_double_open_all`) and (3) `include_double_open_exclude_top1` combining (1) and (2). The direction, magnitude (order of magnitude difference in medians), and statistical conclusions (very small p-values and bootstrap CIs that stay well above 1) remain consistent across those runs. Differences that do appear are predictable and small in the context of decision making: trimming barely reduces CI width and lowers point estimates moderately, while including folded double opens increases sample size and slightly raises variance. In short, conclusions reported here - that auto opens are substantially longer than manual opens and that a small tail drives a large share of total open-time - hold under the alternative pre-processing choices.

Data cleaning and messy data handling

1. Sort by `user_id` then timestamp.
2. Flag and handle orphaned events:
   - **Orphan closes**: flagged and ignored.
   - **Orphan opens (no subsequent close)**: flagged and ignored.
   - **Double opens**: if an open occurs while a prior open is still active, either close the earlier at the new open timestamp (fold) or ignore depending on analysis mode.
3. For extreme durations, allow trimming via `--trim`:
   - `all`: keep all episodes.
   - `exclude_top1`: drop episodes with duration $\geq$ 99th percentile (computed within the selected episode set).

Matching open/close strategy and duration calculation

Algorithm (deterministic, one-pass per user):

1. Sort events for each user by timestamp.
2. Maintain a simple state machine: last_open_event = None.
3. On an `open` event:
   - If last_open_event is None, set last_open_event to this event.
   - If last_open_event is not None, close last_open_event at new open timestamp and start new open. Mark as `DOUBLE_OPEN`.
4. On a `close` event:
   - If last_open_event is None, mark close as `ORPHAN_CLOSE` and save close timestamp.
   - If last_open_event exists, compute duration = (close.ts - last_open_event.ts) / 1000 (seconds), record episode with metadata, and set last_open_event = None. Mark as `REGULAR`.

5. After all events, if last_open_event remains, mark it as `ORPHAN_OPEN` and save open type and timestamp.

Recorded episode columns (canonical): `user_id`, `open_type` (auto/manual), `open_timestamp`, `close_timestamp`, `episode_type`, `duration_sec`, `duration_log10_sec`.

Results and summary statistics

Sample sizes

Metric	Value
Total episodes (all types)	1622
Auto episodes	1000
Manual episodes	622

Descriptive statistics (seconds)

Open type	Median	Mean	25%	75%
Auto	184.821	6323.446	35.195	1195.737
Manual	12.059	1470.260	2.151	135.664

Log-transformed (log10 seconds) per-episode summary

Open type	$\text{Mean}_{\log 10}$	$\text{SD}_{\log 10}$
Auto	2.379	1.088
Manual	1.416	1.070

Inferential results

- Wilcoxon signed-rank (paired per-user medians on log10): $n = 113$; test statistic = 742.0; p-value = 1.24e-12.
- Welch t-test on per-episode log10: t = 17.506; df = 1333.14; p = 6.36e-62.
- Mann–Whitney U on raw seconds: U = 465199.5; p = 1.995e-63.
- Bootstrap: Point estimates and bootstrap 95% percentile confidence intervals (paired bootstrap, n_{boot} =4000, seed=42).

Effect	Point estimate	95% CI (lower)	95% CI (upper)
Median ratio (auto / manual)	16.107	7.178	30.021
Mean log10 factor ($10^{mean_{log10}}$)	8.586	4.911	14.817

Tail diagnostics

Top 1% threshold (seconds): 89841.173.
Top 1% episodes account for 44.34% of total auto open-time and 45.30% of total manual open-time.
Number of users contributing any top-1% episode: 16, which is 8.12% of all users.

Visualizations

Figures below (include the PNGs from the run folder).

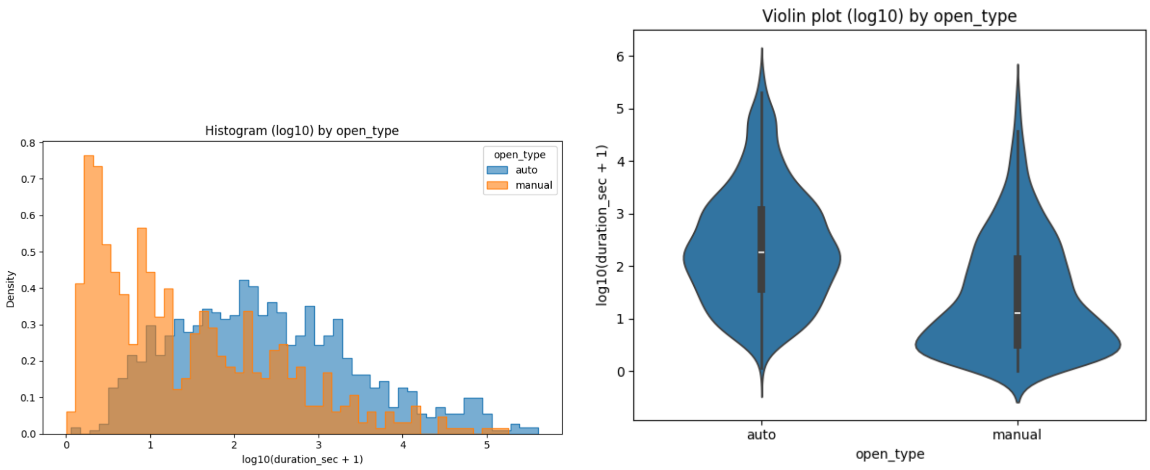

Figure 1: Left: histogram of $\log_{10}(\text{duration\_sec} + 1)$ by open type. Right: violin plot of $\log_{10}(\text{duration\_sec} + 1)$ by open type.

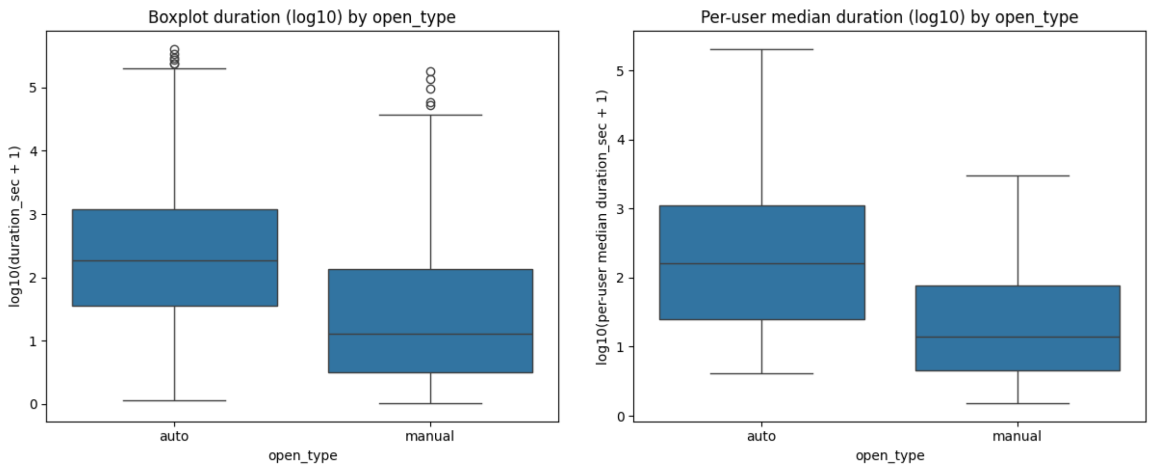

Figure 2: Left: per-episode boxplots on log10 scale. Right: per-user paired medians (log10).

Interpretation and practical implications

- Magnitude and practical meaning
  - The median auto-open duration is roughly 15× the median manual duration. In absolute terms, given a median manual of 12.06 s, the median auto is ≈ 184.8 s. This means a typical auto-open session lasts on the order of minutes rather than seconds and will materially affect aggregate open-time and resource usage.
- Statistical confidence and robustness
  - All three inferential checks (paired Wilcoxon on per-user medians, Welch t-test on log-transformed episodes, and Mann–Whitney on raw seconds) give extremely small p-values, indicating the effect is very unlikely to be noise in this dataset.

  - The paired bootstrap CIs (median ratio CI: [7.18, 30.02]) show the multiplicative effect is large and well separated from 1. The CI width reflects heavy tails and between-user variability. The lower bound still implies a large practical difference.

- Sensitivity to the long-tail
  - The top 1% of episodes accounts for 44–45% of total open-time. Including those extreme sessions inflates means and total-time metrics. Medians and log-transform analyses are more robust.
  - Recomputing key estimates with `exclude_top1` doesn't change conclusions. Auto open time is significantly longer than manual open time.

- Limitations and caveats
  - Causality: these are observational comparisons between open types. Differences may reflect different user tasks, client versions, or contexts that correlate with open type rather than the open mechanism itself.
  - Measurement: durations depend on event fidelity. The cleaning rules used help but do not eliminate the risk of bad data.
  - Estimates reflect the provided dataset and the chosen epoch. Repeat analysis on new data slices (time windows, client versions, regions) would help coming with conclusions.

Reproducibility note

All analysis code lives at the repository: Zerato7/analytics-for-data-products-ides. To reproduce the exact numbers and figures, run:

```
python main.py --mode regular_only --trim all --plots True
```

This creates a run folder under `results/` containing the JSON, text summary, and PNGs used in this report.