I have sessionized clickstream data for user interaction on website and mobile app with target variable satisfaction (satisfied , dissatisfied). There is a page sequence list with all page names in the order in each session with information like search_start, chat_start ect. hen I did a summery statistics , it turned out that people who are starting a chat are more dissatisfied than satisfied. Also sequences with more error pages in the page sequence list are more dissatisfied

I also have an activity sequence list, event sequence list, time gap list for page interactions, hit timestamp list, previous hit timestamps etc. How can I find user pain points (friction) from these sequences. I need to find and present specific pain points to the client. Please do your research and advice how can I do this. Consider industry best practices and other use cases.

There are systems like Fractal, Fullstory, content square, glassbox, quantum metric do friction detection. Also I know that session abandonments, error message, back and fourth navigation, repeated form submissions, high click frequence with low interaction depth, sudden exists, rage clicks etc are some reasons for user friction / frustration / dissatisfaction

do your research and give me specific instructions on how can I find friction / frustration / dissatisfaction or pain points (im using python for data analysis) and present it to my client the specific findings

Thanks for the detailed context. To give you the most relevant, actionable guidance, could you please clarify the following:

1. What format are your sessionized sequences in? (e.g., CSV files, JSON, pandas DataFrames)
2. How large is your dataset (approximate number of sessions)?
3. Are there any labeled examples of friction events, or are you aiming to detect these in an unsupervised way?
4. Do you already have satisfaction labeled for each session, or only a subset?
5. Do you want to use just descriptive statistics and rules, or are you also open to applying machine learning techniques (e.g., sequence modeling or anomaly detection)?

Once I have these details, I'll find best practices, tools, and specific methods (with Python code guidance) to help you detect user friction and present clear findings.

1. parquet file
2. 250,000
3. there is a satisfaction column with each session is labelled as satisfied or dissatisfied based on a survey answered by the users. 75% of sessions are satisfied and 25% are dissatisfied
4. satisfaction/ dissatisfaction is labelled for each session

5. I dont need machine learning at the moment. descriptive statistics, rule or anything that can specifically find the pain points will be great.

Great, I'll look into industry best practices and Python-based analytical methods to identify user friction and dissatisfaction patterns in your sessionized clickstream data. I'll focus on rule-based and statistical techniques, without using machine learning, and explore how leading systems like Fullstory, ContentSquare, and Glassbox approach this. I'll then outline specific steps you can take to find and present actionable pain points to your client.

I'll update you once I have the findings ready.

Identifying User Friction and Pain Points from Clickstream Data

Objective: Analyze session-level clickstream data (250k sessions from web & mobile) to uncover specific user **pain points** or friction signals that correlate with dissatisfaction. We have each session labeled as **satisfied (75%)** or **dissatisfied (25%)** based on user surveys, along with sequences of pages, events, activities, timestamps, etc. Using Python for data analysis (no machine learning needed), we will derive descriptive metrics and rules to pinpoint where users experience frustration. Below, we outline **industry best-practice** signals of user friction, how to **compute and analyze** them from your data, and how to **present** the findings to your client in a clear, insightful way.

Key Friction Signals in User Sessions

Digital experience analytics tools (like FullStory, ContentSquare, Glassbox, Quantum Metric, etc.) track certain **behavioral signals** as indicators of user frustration or friction. We can replicate these with our data. Here are the common pain point signals to look for, and why they matter:

- **Error Pages or Messages:** If a user encounters HTTP error pages (e.g. 404s) or in-app error messages, it's a strong friction signal. **Page load errors** are explicitly considered friction events by industry tools. Our summary already shows **sessions with more error pages skew heavily towards dissatisfied users**, which makes sense – broken links or errors prevent task completion and drive frustration.

- **Session Abandonment / Sudden Exit:** When users unexpectedly drop off before completing an intended task (e.g. leaving mid-funnel), it's often due to frustration. FullStory notes that **failing to address frustration signals can lead to site abandonment**. In our case, if many dissatisfied sessions end on a particular page (or abruptly after an error), that page or step is a likely pain point.

- **Back-and-Forth Navigation (Pogo-Sticking):** Repeatedly bouncing between pages (e.g. going back and forth between a search results page and a product page) indicates the user isn't finding what they need. UX research shows that **immediately tapping the back button after a page view means the user didn't find what they were looking for and is frustrated**. In our data, a page sequence like A → B → A (or frequent page revisits) is a sign of confusion or poor navigation flow.

- **Repeated Form Submissions or Aborted Processes:** If users submit a form multiple times or start a process and never finish, there may be usability issues or errors. For example, a user filling out checkout details but abandoning before purchase could signal frustration (due to slow loading, validation errors, or a hidden "Submit" button). Multiple submission attempts or form error events in a session are clear friction indicators – the user had to retry an action that should be smooth.

- **High Click Frequency with Low Progress (Rage Clicks):** "Rage clicks" occur when a user rapidly clicks/taps an element over and over in frustration. This often means the site/app isn't responding as expected (e.g. a button is unresponsive or slow). You might detect this as **very short time gaps between consecutive click events on the same page**. Such behavior is analogous to "punching the computer with your finger" and is a strong indicator of UI friction. Even if you can't perfectly identify individual rage clicks without UI coordinates, **a burst of clicks in a short time window** or a **high number of clicks on one page** is a red flag.

- **Dead Clicks (Clicks with No Effect):** These are clicks that **have no visible result** – e.g. clicking a UI element that looks clickable but nothing happens. They frustrate users because the site appears unresponsive or misleading. In your data, if an event sequence shows multiple click events without a new page load or state change, that could indicate dead clicks. (This might be harder to extract without detailed UI event data, but keep the concept in mind when examining sessions.)

- **Thrashed Cursor or Random Scrolling:** Users who are **extremely confused or waiting for something** might wiggle the mouse rapidly or scroll up and down repeatedly. While precise "cursor thrash" might not be directly measurable in your dataset, you can infer frustration if you see patterns like **very quick successive page scroll events** or unusually long pauses followed by sudden navigation. "Random scrolling" (fast scrolling through content without engaging) can mean the user is **struggling to find information**.

- **Use of Support Channels (Chat/Help):** Your initial finding is that sessions where users **start a live chat** are more often dissatisfied. This is intuitive – if users resort to chat or support, it's often because they hit a roadblock or confusion point they couldn't solve on their own. So, **initiating a chat or opening help/FAQ pages** during a session can be logged as a frustration signal. We should verify how frequently chat is used in dissatisfied vs. satisfied sessions and which pages trigger chat starts.

- **Search Queries or Repeated Searches:** If users invoke the search function (e.g. `search_start` in your page sequence) frequently, it may indicate they couldn't navigate to what they wanted easily via the menu or links. Particularly, **multiple searches in one session** or a search immediately followed by another search suggests the first attempt wasn't successful – a possible friction point (content findability issue). Analyze if dissatisfied users tend to search more (or sooner) than satisfied users.

- **Slow Page Loads or Long Pauses:** A page taking too long to load is frustrating and is considered a friction event by tools like Quantum Metric (they explicitly track **page load slowness** as a friction signal). In your data, you have timestamps and time gap lists – use these to spot **unusually long gaps between page hits or events**. For example, if a user spends an abnormally long time on a loading page or in between steps, they might be stuck waiting (possibly leading to drop-off). On the flip side, **very short time on a critical page** (quick exit) could mean they gave up immediately. Identifying pages with either **extremely high dwell time or extremely short time** among dissatisfied sessions can highlight performance or UX issues.

By focusing on the above signals, you align with known **best practices** in UX analytics for finding friction. Next, we'll detail how to extract these insights from your clickstream data using Python.

Analyzing Clickstream Sequences to Detect Friction (Step-by-Step)

To systematically find pain points, follow these steps in your Python analysis. We will create metrics from the session data, compare behaviors of satisfied vs. dissatisfied users, and drill down into specific patterns:

Load and Prepare the Data:
Use pandas (or PySpark if the dataset is too large for memory) to load the Parquet file containing session data. Each session should include an identifier, the satisfaction label, and the various sequences (pages, events, timestamps, etc.). For example:

```
 import pandas as pd
sessions = pd.read_parquet("sessions.parquet")
```

1. Ensure that the sequences are accessible (perhaps as lists or strings). You might need to convert JSON strings to Python lists if they are stored as text. Also, separate the web and mobile app sessions if needed, since friction signals might differ slightly (though

many are common).

2. **Feature Engineering – Derive Friction Metrics:**
   Create new columns in the DataFrame for each friction signal/metric of interest. Specifically:

   - **Error Page Count:** Count how many page names in the page sequence match known error pages (e.g., "404", "500" or any page with "error" status). Also create a binary flag for **any_error** (1 if ≥1 error occurred).

   - **Chat Start Flag:** A binary flag if `chat_start` appears in the page or event sequence. Similarly, maybe a count of how many times chat was used (if multiple chat interactions can occur).

   - **Search Usage:** Count of search events (e.g., occurrences of `search_start` or search result pages in the sequence). Also possibly a flag for multiple searches (>=2).

   - **Back-and-Forth Navigation:** This can be quantified in a few ways:

     - **Revisit Count:** Count of pages that appear more than once in the sequence (especially non-consecutive repeats). If a user visited the same page, went elsewhere, then returned, increment a counter.

     - **Immediate Backtrack:** You can specifically check if any page is followed immediately by a return to the previous page (e.g., sequence[i] == sequence[i+2] which implies a pattern A → B → A). A simple approach: slide a window of length 3 over the page list and check for patterns of form [X, Y, X]. Count those occurrences or flag a boolean.

   - **Repeated Form Submissions:** If you have an event sequence that logs form submissions (e.g., an event like "form_submit" or "payment_attempt"), count them. A high count (or simply more than 1) suggests the user tried multiple times. Also check for events like validation errors or resets. If such event types exist, include an **any_form_repeated** flag.

   - **Rage Click Tendency:** Without precise UI data, approximate this by looking at the **time gap list** between interactions and the number of clicks:

     - Calculate the minimum time gap in each session or count how many gaps were extremely short (e.g., < 1 second) – multiple sub-second gaps could indicate rapid clicks.

    - If possible, also count the number of click events in the event sequence on the same page without an intervening page navigation. For instance, if the event list shows `"click_button, click_button, click_button"` before a new page or outcome, that's a potential rage click cluster.

    - Create a feature like `rapid_clicks` = True if >N clicks occurred within M seconds on the same page (choose N, M thresholds that seem reasonable, e.g., 3 clicks in 2 seconds).

  - **Dead Click Proxy:** Check if there are consecutive click events with no page change or state change. For example, if the event sequence has two or more clicks in a row, and the page sequence doesn't move forward, it might indicate clicking an unresponsive element. This overlaps with the rage clicks analysis, but even a pair of clicks on the same page could be a mini "dead click" signal (user clicked once, nothing happened, clicked again).

  - **Thrashed Cursor/Erratic Behavior Proxy:** If you have granular events like mouse movements or scrolls, you could measure bursts of scroll events (many scrolls in a short time) as a sign of frustration. If not, you might use **very long time on page followed by sudden exit** as a proxy (user waited or hesitated, then gave up). Create features such as `max_idle_time` (max gap between events) and `total_session_duration`. Extremely high `max_idle_time` might mean the user was stuck waiting; extremely low `session_duration` with dissatisfaction might mean they rage-quit.

  - **Navigation Depth and Click Depth:** Compute how far users get in the session. For example, `unique_pages_count` and `total_pages_count` in the session. A **low unique/high total** (like visiting same few pages repeatedly) could indicate going in circles. Also, if `total_pages_count` is very low (user gave up after 1-2 pages) and the session is dissatisfied, it suggests the entry page or second page caused them to quit (could be a UX issue on those pages). On the other hand, a **very high number of page visits** in one session might mean the user is frantically navigating without finding what they want. Calculate averages for each segment.

  - **Outcome/Funnel Flags:** If your site has a known funnel (like add to cart → checkout → purchase), you can mark whether the user reached a successful outcome. A **dissatisfied session likely failed to reach the goal.** Identifying *where* they dropped can highlight the friction point. For example, many dissatisfied users might drop on the payment page – possibly due to a payment error or confusing form. These funnel drop-off points are pain points.

Using Python (pandas), you can derive these metrics with vectorized operations or by writing functions to apply to each row (be mindful of performance on 250k sessions – use efficient methods or sample if needed). For example:

```
 sessions['error_count'] = sessions['page_sequence'].apply(lambda pages: sum('error' in p.lower() for p in pages))
sessions['repeat_page_visits'] = sessions['page_sequence'].apply(lambda pages: len(pages) - len(set(pages)))
sessions['back_and_forth'] = sessions['page_sequence'].apply(lambda pages: any(i < len(pages)-2 and pages[i] == pages[i+2] for i in range(len(pages)-2)))
# ... etc for other metrics
```

3. *(Adjust the logic to your actual data schema and naming.)*

4. **Compare Satisfied vs. Dissatisfied Sessions:**
   With these new features in place, perform **descriptive statistics** segmented by the satisfaction label:

Calculate the **mean or proportion** of each friction metric for *dissatisfied sessions* vs *satisfied sessions*. For example, what % of dissatisfied sessions had ≥1 error page, compared to satisfied? How many average backtracks in each group? Use `groupby` on the satisfaction column:

```
 metrics = ['error_count','any_error','repeat_page_visits','back_and_forth', ...]
sessions.groupby('satisfaction')[metrics].mean().round(2)
```

    -
    - Identify the **biggest differences**. Likely, you'll see metrics like error presence, chat usage, back-and-forth, etc., significantly higher in the dissatisfied group. These differences are your prime candidates for pain points. For instance, you might find "40% of dissatisfied sessions had an error page, vs only 5% of satisfied sessions" – a huge red flag to investigate those errors. Or "Average number of distinct pages visited is 8 for dissatisfied vs 5 for satisfied – suggesting frustrated users wander more."

5. Also, check **distribution** of some metrics (e.g., histogram of page count or duration by satisfaction) to see patterns. Dissatisfied sessions might cluster at extremes (very short sessions that ended quickly or very long sessions that indicate struggle).

6. **Identify Specific Problematic Pages/Steps:**
   Drilling down further, examine *which* pages or actions are most often involved in friction for dissatisfied users:

   - **Error pages:** List the top error pages or messages encountered by dissatisfied users. If one particular error (e.g. an outage on "/checkout/error") is common, that's a known bug/issue to fix.

   - **Common last pages for dissatisfied sessions:** The last page in the session sequence for many dissatisfied users could reveal where they gave up. Perhaps many exit on the **search results page** (indicating they didn't find what they wanted), or on a **form page** (indicating they might have gotten stuck on a form). Identify the top exit pages for dissatisfied vs satisfied. If, say, **"Shipping Info Page"** is a top exit for dissatisfied, that page experience might be causing frustration (maybe a form or error there).

   - **Frequent backtrack patterns:** If possible, find which page pairs are involved in back-and-forth loops. For example, many dissatisfied users might go **Product Listing → Product Detail → Listing → Detail** repeatedly. That suggests the product details aren't helping them decide, a possible content or layout issue. Use sequence analysis or simple counting of transitions for dissatisfied sessions to spot these loops.

   - **Chat triggers:** Look at what pages or events immediately precede a chat start in dissatisfied sessions. If users often start chat on the checkout page or after seeing no search results, those contexts are problematic.

   - **Search analysis:** Check what queries (if available) dissatisfied users search for, or at least how many times they refine searches. It might reveal content gaps. Even without query text, the fact that they search multiple times per session is insightful.

   - **Form retries:** If you have data on form field errors or validation failures, see which forms cause the most repeat attempts (e.g., password reset attempts, payment form errors, etc.). Those are pain points to highlight (e.g., "Many users struggled with the payment form – multiple failed submissions were observed in 20% of dissatisfied sessions that reached checkout").

7. *Technical tip:* You might need to explode the sequences or iterate through them to count specific patterns or transitions. Python's itertools or collections (for counting pair frequencies) can help. For large data, consider sampling or using SQL-like queries if available.

8. **Validate with Examples (Optional but Insightful):**
   To ensure your metrics truly reflect frustration, you can **sample a few sessions** flagged with high friction metrics and manually inspect their sequences (just print out the page_sequence for a handful of top friction sessions). This is analogous to a mini "session replay". For example, you might see a sequence like:

```
["Home", "Search", "Product Listing", "Product Detail", "Product
Listing", "Product Detail", "Error Page", "ContactUs", "Exit"]
```

It’s clear this user searched, bounced between listing and detail pages (back-and-forth), hit an error, then went to contact (possibly chat or support) and quit. This anecdotal example confirms the quantitative signals and provides a narrative you can share with the client. (Be sure to anonymize any IDs – stick to page names or types.)

9. **Statistical Significance (if needed):**
   While not strictly required, you may perform simple statistical tests to strengthen your findings. For instance, a chi-square test for categorical differences (like % with errors in dissat vs sat) or a t-test/Mann-Whitney for numeric differences (like time on page). Given 250k sessions, even small differences will likely be significant, but the **practical significance (effect size)** matters more – focus on the largest gaps.

Presenting the Findings to Your Client

Now that you’ve identified where users experience friction, it’s crucial to **communicate these insights clearly and actionably**. Here’s how to present your findings:

- **Use Headline Metrics and Visuals:** Start by highlighting the most impactful pain points. For example: *“Sessions with an error page are 8x more likely to be dissatisfied (40% vs 5%).”* or *“Users who engaged with live chat had a 60% dissatisfaction rate, indicating they were struggling and needed help.”* Quantify each point. Visualize with a bar chart or percentage graphic if possible (e.g., a bar for each friction signal showing the incidence in satisfied vs dissatisfied sessions). Visuals make these comparisons instantly clear.

- **List Top Friction Patterns:** Use a bullet list or table for the top 3-5 specific friction scenarios you found. For each, describe the pattern and why it’s problematic. For instance:

    - *High occurrence of checkout errors:* “**Checkout Error – Payment Failed** page appeared in 15% of dissatisfied sessions, often leading users to abandon the purchase.”

    - *Back-and-forth navigation:* “**Product page loop** – 20% of dissatisfied users oscillated between product list and detail pages multiple times, indicating difficulty in finding suitable products or information.”

    - *Form resubmissions:* “**Account Signup Form repeated** – Many dissatisfied sessions show users submitting the registration form twice or more, suggesting validation issues or confusion in the signup process.”
      Each point should tie a **user frustration** to a **potential root cause** (navigation

issue, bug, slow performance, unclear content, etc.).

- **Incorporate Session Stories:** To make the data relatable, consider presenting one or two **user journey stories** (as reconstructed from the data). For example: *"One user's journey (dissatisfied): they searched for a product, clicked on 5 different items, added one to cart, got an error at checkout, tried to refresh, and then left without purchasing. This exemplifies the checkout error issue we see in many sessions."* Such narratives, even if simplified, help the client **empathize** with the user experience behind the numbers.

- **Highlight Business Impact:** Emphasize why addressing these pain points matters. For example, *"These frustration signals likely contribute to lost conversions and negative user sentiment. Customers encountering errors or dead-ends are more likely to abandon the session, and perhaps not return."* If you have any quantification (like users who had an error spent X less time or had Y% lower conversion), mention that.

- **Prioritize Recommendations:** Not all pain points are equal. Recommend focusing on the **most frequent or impactful issues first**. For instance, if error pages and checkout issues affect a large chunk of users, those are top priority to fix (e.g., bug fixes, clearer error messaging, etc.). If back-navigation suggests missing content, maybe improving site search or product info is next. Frame these as *actionable recommendations* driven by the data.

- **Use clear visuals for sequences if possible:** A Sankey diagram or flow chart highlighting where dissatisfied users diverge from the happy path can be powerful. For example, a simple flow: out of 100% of sessions that reach page X, only 30% of dissatisfied go to the next step vs 80% of satisfied – indicating a drop-off. Even a basic funnel chart comparing the two groups can pinpoint where friction happens (e.g., "50% of satisfied users who view Product page add an item to cart, but only 20% of dissatisfied do – implying an issue on the product page for those frustrated users").

- **Cite Industry Concepts (to add credibility):** As you explain each pain point, you can subtly mention that these patterns are recognized in UX analytics. For example, *"The rapid clicking behavior – often called 'rage clicks' – is a known sign of user frustration. We detected many such instances on the mobile app's login screen, suggesting users tapped the login button repeatedly when nothing happened."* This shows the client that the analysis aligns with best practices and research, not just arbitrary numbers.

- **Keep the Presentation Client-Friendly:** Avoid internal jargon or overly technical terms when presenting. Instead of saying "low interaction depth with high click frequency," you might say "**Users clicked around a lot on the same page without getting anywhere** – a sign they were frustrated and couldn't find what they needed." The goal is to clearly convey the *experience* behind the data.

- **Preserve Citations and Evidence:** If this is a written report, include the data evidence (charts, percentages, maybe a snippet of an example session sequence) for each finding. Since the question is about analysis, you might not literally include academic citations in a client presentation, but you can reference known benchmarks or tool definitions if needed (e.g., mention that *FullStory/Quantum Metric use similar signals to identify frustrated sessions*).

By following these steps, you'll produce a comprehensive analysis of user friction in your clickstream data, and present concrete, data-backed pain points to your client. This approach mirrors what leading product analytics tools do – surfacing **where users struggle** so that the business can take action to improve the experience. Addressing the identified issues (e.g. fixing errors, streamlining navigation, speeding up pages, improving forms) should ultimately increase the satisfaction rate beyond the current 75%. Good luck with your analysis and presentation!

Sources

- FullStory Team (2022). *What's a frustrated session?* – Definition of frustration signals like dead clicks, rage clicks, error clicks and their impact.

- FullStory Help Center – *Frustration Signals (Rage Clicks, Error Clicks, Dead Clicks, Thrashed Cursor)* – Definitions of rage clicks (rapid repeated clicks), dead clicks (clicks with no effect), etc., indicating user frustration.

- UXCam Blog (2019). *How to Identify User Frustration* – Discusses frustration indicators: **direct back-taps** (immediate back navigation), **aborted events** (users abandoning a flow, possibly due to frustration), and erratic scrolling behavior.

- Adobe/Quantum Metric Integration (2025). *Quantum Metric friction events* – Notes that **page load slowness** and **page load errors** are tracked as friction events alongside rage clicks, underlining the importance of performance and error-free experience.