Case Study: Customer Review Intelligence with n8n and Gemini

By Olusegun Marcus · Built and tested on public data (UCI Sentiment Labelled Sentences, Amazon reviews file) · Run date: 21 October 2026

Summary

I built a free, self-hosted n8n workflow that reads customer reviews from a Google Sheet, sends each one to Gemini, and writes a sentiment, a theme and an urgency flag back to a results sheet. I then scored the output against known answers, so every figure below comes from a comparison, not an impression.

Headline result: on a 100-review sample, the positive/negative call matched the human label on **97 of 100**. Themes and the urgent flag have no ground-truth labels, so I checked them by hand on 33 reviews against rules I wrote beforehand: themes were acceptable on **24 of 33** and the urgent flag on **30 of 33**. I report the theme result as a weakness, not a footnote.

Problem

Teams that receive reviews in volume cannot read them all. I wanted to test how far a no-cost workflow could sort them, and, more importantly, how to measure whether its output is accurate enough to trust.

Data

- Source: the public UCI Sentiment Labelled Sentences dataset, Amazon file. It holds 1,000 short sentences about phone accessories, 500 labelled positive (1) and 500 negative (0). There is no neutral label.
- Sample: every 10th review (IDs 10, 20 ... 1000), giving 100 reviews: 43 positive and 57 negative.
- Labels were used exactly as published and were not edited after seeing results.

Workflow

Built in n8n running on my own computer (localhost), with Google Sheets and the Gemini API free tier:

- Manual trigger, then Google Sheets reads the rows where in_sample = yes (100 rows).
- Loop Over Items, batch size 5, so requests stay under the free-tier rate limit shown in Google AI Studio on the run date (15 requests per minute for the model used).
- Gemini (model gemini-3.5-flash-lite, JSON output) returns sentiment, theme and urgent for each review, one request per review. Retry on failure: up to 3 tries, 5 seconds apart.
- Edit Fields attaches the review ID, text and true label, reads the JSON, and calculates a 1/0 correct flag.
- Google Sheets appends one row per review to a results tab, then a Wait node pauses 25 seconds before the next batch.

Prompt (one version, written once and not changed after seeing results):

```
You classify customer reviews of phone accessories. Return only JSON with exactly these keys: sentiment ("positive", "negative" or "neutral"), theme (one of: battery, sound, price, build_quality, connectivity, comfort, support, other), urgent (true or false). Set urgent to true only if the reviewer says the product stopped working, failed or is unusable, or asks for a refund or help. Review: {review text}
```

How I measured

- **Sentiment:** correct only if the AI said positive for label 1, or negative for label 0. A "neutral" answer counts as wrong, because the dataset has no neutral class. This is deliberately strict.

- **Theme and urgency:** no labels exist, so I judged 33 reviews (every review whose ID divides by 30) against rules written before judging. Theme is acceptable if it is the main thing the reviewer discusses, or if "other" was chosen and no listed feature is named. Urgency is acceptable if the flag matches the prompt's rule, whether the flag is true or false.
- **Integrity checks on the results sheet:** 100 rows scored, 100 unique review IDs, theme counts summing to 100, and 0 answers outside the three allowed sentiments.
- Scoring is done with sheet formulas on a summary tab, and my judgments and notes are kept in a spot-check tab.

Results

Measure	Result	How it was checked
Sentiment (positive/negative)	97 / 100	Human labels in the dataset
Positive reviews identified	41 / 43	Human labels
Negative reviews identified	56 / 57	Human labels
Theme acceptable	24 / 33	My manual check, my own written rule
Urgent flag acceptable	30 / 33	My manual check, my own written rule

The AI flagged 31 of the 100 reviews as urgent. Themes assigned across the 100 reviews: build_quality 34, other 21, connectivity 11, battery 10, sound 10, comfort 6, price 5, support 3.

The three sentiment misses

ID	Review	Label	AI said	What happened
70	This device is great in several situations:1.)	positive	neutral	Text is cut off mid-list, so there was little to judge.
360	The microphone also works well, but (according to people I have called) it applifies everything.	positive	neutral	Mixed review: one praise clause, one complaint. Neutral is defensible.
450	There's really nothing bad I can say about this headset.	negative	positive	In my reading the dataset label looks wrong and the AI read the sentence correctly.

What I found

- **Sentiment was strong on this dataset.** 97/100, with only one clear positive/negative reversal, and that one looks like a labelling error. I keep 97/100 as the headline. Counting ID 450 as correct would give 98/100.
- **Results varied between runs.** The same review gave different answers on different days with the same prompt and model. Review 20 came back negative in some runs and neutral in others, and review 70 was positive in an earlier 10-review test and neutral in the full run. I treat the single 100-review run as the official score. The earlier runs on 5 and 10 reviews were checks that the workflow was wired correctly.
- **Themes were the weak point.** 24/33 is about two in three. Part of the cause is my theme list, which has no category for setup, delivery or general praise, so reviews such as "Setup couldn't have been simpler" and "Over charge shipping" had no good fit. build_quality and other together hold 55 of the 100 labels.
- **The urgent flag matched my rule on 30/33,** but it is one person judging against my own rule. Borderline rows (for example "PRETTY USELESS!") are noted in the sheet, and a stricter reader could score it lower.

- **Checking row counts and unique IDs mattered.** They are how I caught wiring mistakes (a node pointed at the wrong sheet tab, and a duplicate run) before any result was scored.

Limitations

- Small samples: 100 reviews for sentiment and 33 for themes and urgency. The 33-review figures could easily move by several points with a different sample or a second reviewer.
- One scored run. Because answers vary between runs, another run could score a point or two differently.
- The dataset is made of short sentences that are labelled only positive or negative. It is easier than real reviews, which are longer, mixed, sarcastic or in several languages. 97/100 is not a general accuracy figure for customer reviews.
- The sample is systematic (every 10th review), not random.
- Theme and urgency were judged only by me, using rules I wrote. There is no second reviewer.
- Strict scoring counts a neutral answer as a miss.
- Model, free quotas and limits are as they stood on 21 October 2026 and may change.
- Not production-ready. Theme and urgency outputs would need human review.

Next steps

- Revise the theme list (for example add setup, delivery and general praise) and test it on fresh reviews not used here, such as IDs 5, 15 and 25 onwards, then report both versions. This is a hypothesis; I have not tested it.
- Run the same 100 reviews several times to measure run-to-run variation.
- Get a second person to judge the 33 spot-check rows.
- Add an alert for negative reviews flagged urgent (not built yet).
- Test on longer, messier reviews.

Method

I used an AI-assisted workflow design approach: I used Claude to plan the pipeline, debug errors and design the scoring checks, and I built, ran and verified every step myself against the dataset's labels. The theme and urgency judgments in the spot-check are my own.