CASE STUDY · PROCESS AUTOMATION · TWO BOTS, TWO DECISIONS

Customer Care Process Automation

Two bots replacing two manual back-office jobs, with a rule about when not to decide. One bot validates records against rules with known right answers, the other reads complaints and routes them to one of five back-end teams. Below 0.55 confidence, a ticket goes to a human instead of being forced through.

> What this evidences: a headline accuracy is not the result. What the agent does with the tickets it is unsure about is. Choosing not to use ML is as much a design decision as choosing to.

Taiwo Alabi · MSc Artificial Intelligence, National College of Ireland

Jul 2026 to Aug 2026 · IAPA Module 2 · Python · scikit-learn (TF-IDF, Naive Bayes) · pandas · BPMN 2.0 · openpyxl · python-docx

01 · The problem

Two manual jobs in the back office

Hundreds of customer complaints sorted by hand every morning, on a Nigerian mobile network. A person reads each one and decides which back-end team it belongs to: SIM activation, network, refunds, payment and billing, or general.

Two things go wrong. People misroute tickets, because after the sixtieth complaint they all start to read the same. And the customer waits while their ticket finds the right desk, then waits again once it gets there.

> **A misroute is not a small error.** It costs the customer days, and the business a second handling pass on work it already paid for once.

02 · The decision that matters most

When not to decide

Five tickets. Two of them fall under the 0.55 confidence line and go to a human. Three of them get auto-routed. The bar chart is the escalation decision in its rawest form.

A classifier will always return an answer. Asked to route a ticket it has never seen anything like, it will still pick the closest of the five categories and report a number. That is the failure mode worth designing against.

So the agent does not have to answer. Any ticket scoring below 0.55 confidence is flagged `needs_human_review` and escalated instead of routed.

> **The economics justify it.** A needless escalation costs an agent about five minutes. A confident misroute costs the customer days. When one error is two orders of magnitude more expensive than the other, the correct system is one that escalates freely, and I would rather report a lower automation rate than a higher misroute rate.

03 · Results

The headline number is the least interesting thing here

The accuracy number is the least interesting thing here. What matters is what the agent does with the tickets it is not confident about, which is section 02.

MEASURE	RESULT
Confidence threshold for escalation	0.55, below which a human decides
Routing accuracy on held-out tickets	95.2% (40 of 42)
Labelled corpus	210 tickets, 42 per category
Routing categories	5
Manual processes replaced	2

Reported per category rather than as a single figure, because a router that is strong on common categories and weak on rare ones is a different system from one that is evenly good, and an average hides that difference.

CATEGORY	PRECISION	RECALL	F1
SIM activation	1.00	1.00	1.00
Network	1.00	1.00	1.00
Refunds	1.00	1.00	1.00
Payment and billing	0.88	1.00	0.93
General	1.00	1.00	1.00

P_BILL is the weak class at 0.88 precision. The other four are at 1.00. The weakness is visible in the confusion matrix: P_BILL and REFUND swap with each other. Every other prediction is on the diagonal. This is the failure mode the escalation gate is meant to absorb.

> **Confusion between neighbouring categories** is a routing problem worth fixing. Errors scattered evenly would suggest the features themselves are too weak, and that is not the pattern.

04 · The process, before

Three lanes, one decision

Before the bot. Three lanes, one decision: a person reads the ticket and decides which of five back-end teams it belongs to. The bottleneck is the manual decision.

05 · The process, after

Two pools, two bots

After the bot. The classifier proposes a team. The escalation gate, the diamond at 0.55 confidence, is what the project is really about. Tickets scoring below the line go to a human instead of being forced through.

The redesign splits the work by the kind of decision each part requires. Field validation has known correct answers, so it goes to rules. Reading a complaint and inferring intent does not, so it goes to a classifier. The confidence gate sits between the classifier and the outcome.

06 · BPMN first, code second

Map the process before automating it

I mapped out the existing process as a flowchart, using BPMN 2.0, before writing any code. That order was deliberate: it meant the automation replaced a process I actually understood, rather than my assumption of one, and the model is what made the manual touchpoints explicit enough to decide which were worth automating.

07 · How the triage agent works

TF-IDF, Naive Bayes, by design

Ticket text is turned into numbers with TF-IDF (a way of scoring each word by how much it tells you about the topic), then sorted into one of the five routing categories by Naive

Bayes (a simple, fast classifier built on word probabilities).

Naive Bayes is a deliberate choice rather than a default. The corpus is 210 tickets, which is small, and Naive Bayes stays reliable at that size where a heavier, more complex model would start memorising the training tickets instead of learning the pattern. It is also inspectable: I can see which words drive a routing decision, which matters when somebody asks why a ticket went where it did.

`08 · The second bot`

Rule-based validation

The second automation produces the daily customer creation report. It reads the customer records, validates each field, writes the clean report to Excel and Word, and writes everything that failed validation to a separate exceptions file.

> **This half is deliberately not machine learning.** These are hard business rules with known correct answers, so a classifier would introduce uncertainty where none needs to exist. Choosing not to use ML is as much a design decision as choosing to.

The validator ran on 120 records and flagged 5. The exception types are named in the rule output: bad date, unknown segment, bad MSISDN prefix, bad MSISDN digits, missing field. The 4% exception rate is what the bot costs the team in manual review.

Outputs: `routed_tickets.csv`, `validation_exceptions.csv`, and the daily report in both `.xlsx` and `.docx`.

`09 · Honest limitations`

- **The corpus is small and synthetic.** 210 generated tickets across five categories. Real complaints are messier: codeswitching, abbreviations, several issues in one message. The generator was written to be realistic, but it is still a generator.
- **42 test tickets is a thin evaluation.** 95.2% means 40 of 42, so a single ticket moves the figure by more than two points. The pattern of errors is more informative than the headline number.
- **Five categories is a simplification.** A real network has more teams and less clean boundaries between them.

- **The confidence threshold was set by judgement, not swept.** 0.55 reflects the cost asymmetry described in section 02, but a proper sweep against real escalation costs would be the honest way to fix it.

---

10 · Running it

```
python gen_data.py          # build the labelled corpus
python triage_bot.py        # train, evaluate, route
python rpa_report_bot.py     # validate records, write the daily report
python plot_results.py       # regenerate every figure in this document
```

Seeded, so results reproduce on every run.

Stack: Python · scikit-learn (TF-IDF, Naive Bayes) · pandas · BPMN 2.0 · openpyxl · python-docx

---

11 · Terms used in this report

Precision
of the tickets the model sent to a category, how many actually belonged there. Low precision means it's wrong a lot when it says "this one".

Recall
of the tickets that truly belonged to a category, how many the model actually caught. Low recall means it misses real cases.

F1 score
one number that blends precision and recall, so a model can't look good by being great at only one of them.

TF-IDF
a way of scoring each word in a ticket by how much it tells you about the topic, turning text into numbers a model can use.

Naive Bayes
a simple, fast classifier that guesses a category based on which words showed up in similar tickets before.

BPMN 2.0
a standard way of drawing a business process as a flowchart, before automating any of it.

Escalation gate

a confidence threshold below which the classifier refuses to decide and the ticket is sent to a human instead.

Links

Project page: iamtamilore.github.io/p/care-agent

Source (BPMN diagrams): github.com/iamtamilore/customer-care-automation

As-Is BPMN: as-is diagram

To-Be BPMN: to-be diagram

Corpus: iamtamilore.github.io/corpus

Home: iamtamilore.github.io