Lecture 2: Marketing Automation with HITL

Structured outputs, evaluation loops, and human review

University of Chicago

December 29, 2025

What you will build today

- A reproducible workflow that turns lead data into outreach drafts
- A quality-control (QC) step that flags risky/low-quality outputs
- A simple human-in-the-loop (HITL) review queue for approvals/edits
- A tiny evaluation harness to compare prompt versions

Business problem

Scenario: You're on a growth team. You have many inbound leads and limited human time.

Goal: Generate compliant, personalized outreach at scale while minimizing:

- hallucinated claims
- policy / compliance violations
- off-brand tone
- missing personalization

Inputs (provided in `lecture_2/data`)

- `leads.csv`: lead attributes and notes
- `product_one_pager.md`: facts you are allowed to use
- `brand_guidelines.md`: tone and style constraints
- `rubric.md`: what counts as a “good” draft

Workflow (high-level)

1. **Summarize** lead notes → JSON
2. **Draft** email + subject → JSON
3. **QC** pass: check claims, tone, constraints → risk score + reasons
4. **HITL** queue: approve / edit / reject
5. **Evaluate**: measure pass-rate across a small set

New workflow primitive introduced

- **Human-in-the-loop review**: you do not ship raw model output to customers
- **Evaluation harness**: treat prompt edits like code changes (measure impact)

Structured output schemas (examples)

Lead summary schema

- lead_summary (string)
- pain_points (list[string])
- suggested_angle (string)
- missing_info (list[string])

Draft schema

- subject (string)
- email_body (string)
- personalization_tokens (list[string])

Exercises

- Baseline prompt → get working JSON output
- Improve personalization while keeping compliance constraints
- Add QC checks: hallucination/claims, tone, forbidden phrases
- Implement a HITL review loop in the notebook
- Run the mini-eval and compare prompt versions

Deliverable

- `notebooks/lecture_2_marketing_hitl.ipynb`
- Output files in `data/outputs/`:
  - `drafts.csv` (final approved drafts)
  - `qc_report.csv` (scores + reasons)

Extensions / Optional challenges

- **Rubric-based grader**: have the model score drafts using `rubric.md`; compare to human scores
- **Batching + cost controls**: cache summaries; estimate tokens/cost; compare one-pass vs two-pass QC
- **Policy-driven compliance**: map `compliance_tags` to explicit deny/allow rules and required disclaimers
- **Prompt/version tracking**: log prompts + model + params alongside outputs for reproducibility
- **Multi-variant testing**: A/B prompt variants; select winners by eval metrics

Discussion

- Where did the model fail? (missing facts vs wrong facts vs style)
- What did structured output enable?
- What checks should be automated vs left to humans?
- How do cost and latency constrain the pipeline?

Next time

- Add external actions (APIs) and state management
- Ground responses in a knowledge base