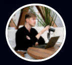
GenAI Accelerator

# Introduction to LLM Evaluations



by Dave Ebbelaar

# Every AI Engineer will ask at some point:

- What will happen if we adjust the system prompts?
- How do I know my RAG pipeline is working correctly?
- Are we performing the correct classification step?
- Besides factual accuracy, is the tone of voice on point?
- How will this behave with real user inputs we haven't seen yet?
- Are we catching unsafe or problematic outputs before users see them?
- How do we know when the system performance is degrading in production?
- Is our system robust enough to flag prompt injections?

# The Challenge

By now, we know that unlike traditional software:

- LLM outputs are **non-deterministic**
- Responses are often **subjective** and context-dependent
- A response may be factually correct but **"vibes are off"**
- Multiple valid responses can exist
- Failure modes are subtle and hard to predict

> **Core Question:** How can we effectively evaluate AI system performance and identify specific areas where improvements are needed?

# Three Core Challenges in LLM Development

Every team building LLM applications faces these fundamental challenges:

### 🔎 Understanding the Data

You build a workflow, but when users start interacting with it, you realize you don't understand what's actually happening at scale. What are users asking? How is your system responding? Where are the edge cases?

### 📝 Specification Gaps

There's always a gap between what you want your system to do and what you can actually specify in prompts and code. You know good output when you see it, but translating that into clear instructions is surprisingly hard.

### 🎯 Inconsistent Behavior

Your LLM works great on your test cases, but then behaves unpredictably on real user inputs. Small changes in wording can lead to completely different outputs, making your system feel unreliable.

# Success with AI = How fast you can iterate

### 🧪 Evaluate Quality

Systematic measurement of your workflow's performance through automated tests, human evaluation, and success metrics.

### 🔧 Debug Issues

Tools to understand and diagnose failures including trace logging, data inspection, and error analysis.

### ⚡ Change Behavior

Techniques to improve your system based on insights from evaluation and debugging.

> Most teams focus only on #3, which prevents them from improving beyond a demo. Doing all three creates a virtuous improvement cycle.

# What is Evaluation?

Evaluation = Systematic measurement of quality in an AI system

### 🎯 An "Eval" is a single metric that measures a specific aspect of performance

- Factual accuracy
- Tone appropriateness
- Following instructions
- Output format compliance

### 🛠️ Evals can be operationalized as:

- **Background Monitoring:** Track performance over time
- **Guardrails:** Block bad outputs in real-time
- **Improvement Tools:** Label data for fine-tuning

# The Analyze-Measure-Improve Lifecycle [1]

This iterative cycle addresses the three core challenges through systematic evaluation and creates a flywheel for continuous improvement.

**Analyze**

Collect examples and categorize failure modes

**Measure**

Translate insights into quantitative metrics

**Improve**

Refine prompts, models, and architecture

[1] Based on "Application-Centric AI Evals for Engineers and Technical PMs" by Shankar & Husain, 2025

# The Three Levels of Evaluation

Cost and effort increase with each level, dictating when and how often you run them

### 🛠️ Level 1: Unit Tests

Fast, cheap assertions that run on every code change

- Run: Every commit
- Cost: Very Low

### 🕵️ Level 2: Human & Model Evaluation

Systematic review and automated critique of quality

- Run: Weekly/Biweekly
- Cost: Medium

### 🧪 Level 3: A/B Testing

Real user experiments to measure business impact

Run: Major releases

Cost: High

# Level 1: Unit Tests for LLMs

## What are LLM Unit Tests?

- Fast, automated assertions
- Check specific, testable properties
- Organized for reuse beyond testing
- Run on every code change
- Don't require 100% pass rate

## When to Use

- Data cleaning and validation
- Automatic retries with feedback
- CI/CD workflows
- Catching obvious failures

## 📝 Example: Structured Output Test

python

```python
def test_ticket_categorization():
    # Test case
    ticket = "My credit card was charged twice this month"

    # Get the structured response
    result = categorize_ticket(ticket)

    # Simple assertions
    assert result.category in ["billing", "technical", "general"]
    assert isinstance(result.confidence, float)
    assert 0 <= result.confidence <= 1

    # Specific test for this input
    assert result.category == "billing"  # We expect billing for this input
```

> In Python, we can implement these tests using simple "assert" statements to verify LLM outputs meet expected criteria.

# Level 2: Human & Model Evaluation

## 🧑🏻‍💻 Human Evals

Humans look at system outputs and score them

- Start with binary: Good/Bad
- Remove ALL friction from data viewing
- Build custom tools for your domain
- Look at as much data as possible initially
- Save good examples for later reference

## ⚙️ Model Evals (LLM-as-Judge)

Use a LLM to critique your system's outputs

- Use the most powerful model you can afford
- Focus on specific quality dimensions
- Generate detailed critiques, not just scores
- Must be aligned with human judgment
- Track human-model agreement over time

> LLM-as-Judge systems need alignment with human evaluators. Track correlation and iterate on the judge's prompt until it agrees with humans.

# LLM-as-Judge: Creating Aligned Evaluators

The Goal: Build an automated evaluator that thinks like your human experts

## ⚖️ The Alignment Process

1. Collect model predictions
2. Generate model critiques
3. Get human evaluations on same data
4. Compare model vs human judgments
5. Iterate on evaluator prompt
6. Repeat until sufficient agreement

## 📄 Example Judge Prompt

Evaluate this customer service response on these criteria:

1. Accuracy: Are the facts correct?
2. Helpfulness: Does it solve the customer's problem?
3. Tone: Is it professional and empathetic?

Provide a detailed critique explaining your reasoning, then rate 1 (Good) or 0 (Bad).

> **Pro Tip:** Start with simple binary ratings and detailed critiques. Use the critiques to understand disagreements and refine your evaluator.

# The Process

LLM Alignment Example

| Input | Model Response | Model Critique | Model Outcome | Human Critique | Human Outcome | Alignment | | Agreement |
|---|---|---|---|---|---|---|---|---|
| My order is 3 days late, where is it? | I understand you're concerned about your order. Let me check that for you right away. Can you provide your order number so I can track its current status? | The response is polite and helpful. It acknowledges the customer's concern and asks for the necessary information to resolve the issue. Shows good empathy and provides clear next steps. | GOOD | Response is too casual for a 3-day delay. Customer said 'late' which implies frustration. Should acknowledge the delay is unacceptable and show more urgency. Also should proactively offer compensation discussion. | BAD | 0 | | 70.00% |
| How do I cancel my subscription? | To cancel your subscription, simply log into your account, go to Settings, click Billing, and select Cancel Subscription. Is there anything specific about our service that prompted this decision? | The response provides clear step-by-step instructions for cancellation, which directly answers the question. The follow-up question about reasons is appropriate for potential retention without being pushy. | GOOD | Good clear instructions and the retention question is appropriate. Strikes the right balance between being helpful and understanding why they want to cancel without being pushy. | GOOD | 1 | | |
| Your product broke after 2 days, this is ridiculous! | I apologize that you're experiencing issues with our product. Product quality is important to us. Please provide your order details and we'll process a replacement right away. | The response appropriately apologizes and acknowledges the customer's frustration. It offers a concrete solution (replacement) and asks for necessary information. However, it could show more empathy given the customer's strong emotional language. | GOOD | The apology feels generic and doesn't match the customer's anger level. 'Product quality is important to us' sounds corporate. Should acknowledge this is completely unacceptable and offer multiple options, not just replacement. | BAD | 0 | | |
| What are your business hours? | Our customer service team is available Monday through Friday from 9 AM to 6 PM EST. You can also reach us via email anytime at support@company.com and we'll respond within 24 hours. | The response directly answers the question with specific hours and time zone. It also provides an alternative contact method with response time expectations. This is comprehensive and helpful. | GOOD | Perfect response. Answers the question completely with specific details and provides alternative contact method. Very helpful. | GOOD | 1 | | |
| I was charged twice for the same order | I see you have a billing concern. Billing issues can be frustrating. You should contact our billing department directly at billing@company.com or call 1-800-555-0123 during business hours. | The response acknowledges the issue but deflects to another department without taking ownership or offering immediate help. For a serious issue like duplicate charges, the agent should take more direct action to investigate or escalate immediately rather than just providing contact information. | BAD | Terrible response. Double charging is a serious issue that needs immediate action. Just deflecting to another department is unacceptable. Should take ownership, investigate immediately, and offer refund processing. | BAD | 1 | | |
| Can you help me track my package? | Absolutely! I'd be happy to help you track your package. Could you please provide your tracking number or order number? Once I have that, I can give you the most up-to-date shipping information. | The response is enthusiastic and helpful. It clearly asks for the necessary information and sets expectations about what information will be provided. The tone is friendly and professional. | GOOD | Great response. Enthusiastic, helpful, asks for exactly what's needed, and sets clear expectations. Perfect tone. | GOOD | 1 | | |
| I'm trying to return an item but your website is confusing | I understand the return process can be frustrating when it's not clear. Let me walk you through it step by step. First, go to your account and find the order, then click 'Request Return' next to the item. Would you like me to stay on while you try these steps? | The response shows empathy for the customer's frustration and provides a helpful step-by-step solution. Offering to stay on the line while they try the steps shows excellent customer service. | GOOD | Good empathy and clear steps, but should also acknowledge the website UX issue and mention it will be passed to the web team. The offer to stay on the line is excellent though. | GOOD | 1 | | |
| Do you price match with competitors? | We don't currently offer price matching, but we do have a satisfaction guarantee. If you're not happy with your purchase for any reason, you can return it within 30 days for a full refund. We also frequently have sales and promotions that might help with pricing. | The response honestly answers the question and provides alternative value propositions. However, it doesn't explore the customer's specific needs or offer to check current promotions that might be available. | GOOD | Answer is honest but feels dismissive. Should ask what competitor price they found and show willingness to work with them. Maybe offer to check for current discounts or upcoming sales. Feels like they're not trying hard enough. | BAD | 0 | | |
| Your delivery driver was rude to me | I'm really sorry to hear about your negative experience with our delivery driver. That's not the level of service we strive for. Can you tell me more details about what happened? I'd like to document this and ensure it's addressed with our delivery team. | The response appropriately apologizes and acknowledges the seriousness of the complaint. It asks for more information to properly address the issue and commits to taking action. This shows good customer service protocol. | GOOD | Good apology and right approach to gather details. Shows they take it seriously and will follow up. This is how you handle service complaints properly. | GOOD | 1 | | |
| I love your product but it's too expensive for me right now | Thank you so much for the positive feedback about our product! I understand budget constraints can be challenging. While I can't offer discounts, I can let you know about our payment plan options or upcoming sales. Would either of those be helpful? | The response appropriately thanks the customer and acknowledges their budget concern. It offers practical alternatives without immediately offering discounts, which shows good business sense while still trying to help. | GOOD | Nice response that acknowledges both the compliment and the concern. Offers practical solutions without devaluing the product. Good balance of being helpful while maintaining pricing integrity. | GOOD | 1 | | |

# Level 3: A/B Testing

## 🧪 When to Use A/B Tests

- System is mature and stable
- You want to measure real user impact
- Testing significant changes
- Need business metric validation
- Comparing different approaches

## 📊 What to Measure

- User satisfaction scores
- Task completion rates
- Time to resolution
- User engagement metrics
- Business outcomes (sales, retention)

## 💡 Example: Prompt Comparison

**Hypothesis:** A more conversational prompt will improve user satisfaction

**Setup:**

- **Control (A):** Formal, direct responses
- **Treatment (B):** Conversational, empathetic tone

### Results & Metrics

**Metrics:**

- User satisfaction rating (1-5)
- Conversation completion rate
- Follow-up questions needed

**Result:** Treatment B increased satisfaction by 15% with no decrease in accuracy

# Types of Evaluation Metrics

## 📑 Reference-Based

Compare LLM output against known "golden" answers

- Exact string matches
- Semantic similarity
- Code execution results
- SQL query correctness
- Structured data validation

## 🎭 Reference-Free

Evaluate inherent properties without "golden" answers

- Tone appropriateness
- Length constraints
- No hallucinations
- Format compliance
- Safety and toxicity

# Common Mistakes to Avoid

### ❌ Tool-First Thinking

- RAG problems? → New vector database
- Accuracy issues? → Try different models
- Need metrics? → Buy an eval platform

**Fix:** Start with simple, custom solutions

### ❌ Generic Metrics Obsession

- Drowning in meaningless scores
- "Helpfulness: 4.2, Truthfulness: 4.5"
- What does this actually tell you?

**Fix:** Focus on specific, actionable metrics

### ❌ Avoiding Your Data

- "A tool will handle it"
- "Real users will tell us if broken"
- "Trust your gut"

**Fix:** Look at lots of real data constantly

### ❌ Unaligned LLM Judges

- Not measuring human-judge agreement
- Assuming the judge "just works"
- No validation process

**Fix:** Always validate judge alignment

# Key Principles for Success

### 🎯 Start Simple & Specific

- Focus on your biggest problems first
- Use existing tools before buying new ones
- Build domain-specific evaluations
- Start with unit tests and manual review

### 🔍 Look at Lots of Data

- Manual inspection has the highest value-to-prestige ratio
- You're doing it wrong if you're not looking at data
- Sample broadly, then focus on patterns
- Never stop examining real examples

### 📊 Remove All Friction

- Make data viewing effortless
- Build custom tools for your domain
- Automate repetitive evaluation tasks
- Track metrics over time

### 🚀 Use LLMs to Scale

- Generate test cases automatically
- Create synthetic data at scale
- Automate critique and labeling
- But always validate with humans

# Building the Evaluation-Driven Workflow

🎯 The Ultimate Goal: Create a system where improvement is systematic, not accidental

### ✅ You'll Know You're Succeeding When:

- You can deploy changes confidently
- Failures are caught before users see them
- You understand your system's behavior
- Improvements compound over time

### ⚠️ You're Struggling If:

- Every fix breaks something else
- You're surprised by user complaints
- Progress feels like trial and error
- You can't measure if changes help