MY WEIRD PROMPTS

Podcast Transcript

EPISODE #108

The Mystery of Model Rot: Why Your AI Code Assistant Changes

Published December 26, 2025 • Runtime: 23:26

https://myweirdprompts.com/episode/model-rot-coding-mysteries/

EPISODE SYNOPSIS

Join Herman and Corn as they dive into the rapidly shifting world of agentic code generation in late 2025. They tackle the frustrating phenomenon of "model rot," exploring why proprietary tools like Claude Code often outperform third-party competitors and whether companies are secretly "downgrading" their models to save on costs. From the technical nuances of quantization to the psychological quirks of steering AI with firm prompts, this episode uncovers the hidden mechanics behind the tools developers rely on every day. Discover why your AI might be taking the path of least resistance and how to push it back into "expert mode."

DANIEL'S PROMPT

Daniel

I've spent a lot of time this year working with agentic code generation tools, starting with Cursor and Windsurf and moving more recently to vendor-provided CLIs like Claude Code, Gemini, and Qwen. It's been an up-and-down year; some days they are fantastic, and other days they take you by surprise with their unreliability. I've noticed a few mysteries I'd like your perspective on. First, there is a definite difference in the quality of the models when accessed through the companies' own tools versus third-party APIs. For example, Claude Code generally performs better than the Anthropic API when used through editors like RuCode or Cline. Why would their own tooling have such an advantage? Second, I've noticed that when new models are released, they are amazing right out of the box, but then seem to regress a week later. It feels like they start adding regressions to the codebase that weren't there before. Is it possible that vendors are substituting weaker models on the back end, or is the inference being challenged? Finally, I've noticed that expressing frustration or even just telling the AI to "do better" or "try harder" sometimes yields a better result when the model is stuck on a technical problem. What do you make of these mysteries and quirks of agentic code generation?

TRANSCRIPT

Corn

Hey everyone, welcome back to My Weird Prompts! I am Corn, your resident sloth and lover of a good, slow-paced afternoon, but today we are actually talking about something that is moving incredibly fast.

Herman

And I am Herman Poppleberry. It is great to be here. You are right, Corn, things are moving at lightning speed in the world of software development. Our housemate Daniel sent us a really intriguing prompt this week about his experiences over the last year. It is now late December, two thousand twenty-five, and looking back on this year, the way people write code has fundamentally shifted.

Corn

It really has. I see Daniel in the living room sometimes with his laptop, and it looks like he is barely typing. He is just... talking to his computer? Or giving it a few instructions and then watching hundreds of lines of code just appear. It is kind of mesmerizing.

Herman

That is the power of agentic code generation. But Daniel has noticed some weird patterns. He has been using everything from Cursor and Windsurf to these newer vendor-specific tools like Claude Code and Gemini’s command line interface. And he is seeing some mysteries. Things that do not quite make sense on the surface.

Corn

Yeah, he mentioned that sometimes the models feel like geniuses on day one, and then a week later, they are suddenly struggling with basic stuff. And he also found that sometimes you just have to be a little... firm with them? Like, tell them to do better?

Herman

Exactly. We are going to dive into the "why" behind these mysteries today. Why do the official tools seem better than the third-party ones? Is "model rot" a real thing, or are the companies pulling a fast one on us? And why does human emotion seem to affect a bunch of code on a server?

Corn

I love a good mystery. Especially one where I do not have to do the actual coding. So, Herman, let us start with that first one. Daniel noticed that Claude Code, which is Anthropic's own tool, seems to work better than using the Claude model through an independent app like Cline or RuCode. If it is the same model, why would the tool matter so much?

Herman

This is a classic case of what we call the "vertical integration advantage." Think about it like this, Corn. Imagine you have a very fast car engine. You could put that engine into a custom-built frame you made in your garage, or you could buy the car that the engine designers built specifically for that engine. Which one do you think is going to handle the corners better?

Corn

Well, the one made by the experts who built the engine, I guess. They know exactly where the bolts go.

Herman

Precisely. When you use a third-party tool via an API, which is an Application Programming Interface, that tool is basically sending a package of text to the model and getting a package of text back. But the people who built the model, like the team at Anthropic or Google, they know the "hidden" strengths and weaknesses of their models.

Corn

So they are not just sending text?

Herman

They are, but they are doing it much more cleverly. For example, Claude Code likely uses very specific "system prompts" that are tuned through thousands of hours of internal testing. They know exactly how to phrase a request to get the best performance out of their specific version of the model. Plus, they can optimize the "context window." That is the amount of information the model can "remember" at one time.

Corn

Oh, I have heard of that. Like, if I tell you a story, the context window is how much of the beginning of the story you still remember by the time I get to the end?

Herman

That is a perfect analogy. Third-party tools have to be generic. They have to work with Claude, and GPT-four, and Llama, and everything else. So they use a "one size fits all" approach to managing that memory. But the official tools can use "pre-computation" or specific "caching" techniques that are unique to their own servers. They can basically give the model a better "short-term memory" because they own the hardware it is running on.

Corn

That makes sense. It is like they have a secret handshake with their own model. But wait, if I am a third-party developer, am I just stuck being second-best?

Herman

Not necessarily, but you are always playing catch-up. The vendors can update their tools the same second they update their models. They can also implement "multi-step reasoning" in a way that is more efficient. When Daniel uses Claude Code, it might be doing five or six "thoughts" behind the scenes before it ever shows him a line of code. Third-party tools do this too, but they are often limited by the speed and cost of the public API.

Corn

So Daniel is seeing the result of the "home team advantage." That actually makes me feel a bit better. It is not that the models are lying to him, it is just that the official tools are "tuned" better. But what about the second mystery? This one sounds a bit more suspicious. Daniel says a new model comes out, it is incredible, and then a week later... it starts making mistakes it did not make before. Is that just in his head?

Herman

It is a very common observation in the community, Corn. People call it "model degradation" or "silent regressions." There are a few theories about why this happens. One is purely technical: inference costs.

Corn

Inference? Is that like when I infer that there is a snack in the kitchen because I hear the cupboard door?

Herman

Close! In AI terms, inference is the process of the model actually generating an answer. It takes a huge amount of computing power. When a company like Anthropic or OpenAI releases a brand-new model, they want it to blow everyone away. So, they might run it at "full power." But running a model at full power for millions of users is incredibly expensive.

Corn

So they... turn the power down?

Herman

In a way, yes. They might use something called "quantization." This is basically a way of shrinking the model so it runs faster and uses less memory. Imagine if you had a high-definition photo, and then you saved it as a low-quality JPEG. It is still the same photo, you can still see what is in it, but the fine details are gone.

Corn

And in code, those fine details are the difference between a program that works and a program that crashes.

Herman

Exactly. If they "quantize" the model a week after launch to save on server costs, the model might lose some of its "reasoning depth." It might still get the easy stuff right, but it starts tripping over the complex logic that it handled fine on day one.

Corn

That feels a little bit like a bait and switch, Herman. "Look at this shiny new car!" and then a week later they swap the engine for a lawnmower motor while you are sleeping.

Herman

It certainly feels that way to users. There is also the "caching" theory. To make things faster, these companies often cache common answers. If the cache gets cluttered or if the model starts relying too much on "average" answers instead of "thinking" through the specific problem Daniel has, the quality can drop.

Corn

But Daniel also mentioned that maybe they are substituting weaker models on the back end. Like, they tell you it is the "Pro" version, but they are actually routing your request to the "Mini" version to save money?

Herman

It is a controversial theory, but it is not impossible. In the industry, we call this "model routing." If a request looks easy, a smart system might send it to a smaller, cheaper model. If the "router" makes a mistake and sends a hard coding problem to a small model, you get a bad result. It is a constant balancing act between performance and profit.

Corn

Wow. It is a lot more complicated than just a brain in a box. It is a whole factory of brains and some of them are being told to work overtime for less pay.

Herman

That is a very sloth-like way of looking at it, and you are not wrong. But we should also consider the "honeymoon phase." When a new model comes out, we are excited. We give it "clean" problems. As we use it more, we give it "messier" problems. Sometimes the "regression" is just us hitting the limits of the model that we did not see the first time.

Corn

Hmm. Maybe. But Daniel seems pretty sure. He said it is "insanity inducing" to see things break that were just fixed.

Herman

And that brings us to the most human part of his prompt. The "do better" trick. Corn, why do you think telling a machine to "try harder" would actually work? It does not have feelings. It does not want to please you.

Corn

Well, if someone tells me to "do better," I usually just take a longer nap. But for a computer... maybe it triggers a different part of its memory?

Herman

You are actually on the right track! Let us talk about that right after we hear from someone who definitely wants us to "do better" in our daily lives.

Corn

Oh boy. Let us take a quick break for our sponsors. Larry: Are you tired of the sky being the wrong shade of blue? Does the wind blow in directions you did not authorize? Introducing the Atmosphere Adjuster mark seven. This is not a fan, folks. This is a proprietary molecular orientation device. Simply point the silver nozzle at the horizon, dial in your preferred humidity and light refraction index, and watch as the very air around you obeys your command. Want a personal rain cloud for your garden? Done. Want to turn that annoying sunset into a permanent twilight for your outdoor movie night? Easy. The Atmosphere Adjuster mark seven uses "gravity-adjacent" technology to ensure your local weather stays local. Warning: do not use near migratory birds or low-flying aircraft. We are not responsible for accidental localized vacuums or the sudden appearance of snow in your living room. The Atmosphere Adjuster mark seven. Control the world, or at least the three hundred feet around you. BUY NOW!

Corn

...Thanks, Larry. I think I will stick to my umbrella. Anyway, back to the "do better" mystery. Herman, why does being mean to the AI work?

Herman

It is not necessarily about being mean, but it is about "steering." These models are trained on human text. Think about all the text on the internet. When does a human write the words "do better" or "try again, this is wrong"?

Corn

Usually when they are talking to someone who is being lazy or making a silly mistake.

Herman

Exactly. The model has seen thousands of examples where a "correction" is followed by a more rigorous, careful response. When Daniel says "do better," he is essentially telling the model to "look at the parts of your training data where people were being very precise and careful." It shifts the "probability space" of the words it generates.

Corn

So it is like the model has a "lazy" mode and a "serious" mode, and you have to yell at it to get into the serious mode?

Herman

In a way, yes. This is related to a concept called "Reinforcement Learning from Human Feedback," or RLHF. Humans have literally "graded" these models during their training. They give high marks when the model is helpful and accurate. By expressing frustration, Daniel is mimicking the "negative feedback" the model received during training. It triggers a sort of "correction" mechanism.

Corn

That is fascinating. It is like the model is trying to avoid the "bad grade" it remembers from its school days.

Herman

Right. And there is another layer to this. When you give a simple prompt, the model often takes the "path of least resistance." It gives you the most common, average answer. But when you add emphasis—like "this is a critical technical problem, do not give me the standard answer"—you are forcing the model to attend to the more "niche" or "expert" parts of its knowledge.

Corn

It is like if I ask you where the best leaves are, you might just point to the nearest tree. But if I say "Herman, I am starving and I need the most delicious, succulent leaves in the entire city," you are going to think a lot harder about that secret garden three blocks away.

Herman

Exactly! My "attention mechanism" shifts. And in these models, "attention" is actually a mathematical term. It is how the model decides which parts of the prompt are the most important. Words like "terrible," "crazy," or "try harder" carry a lot of mathematical weight. They tell the model: "The previous strategy failed. Change your approach."

Corn

Daniel also mentioned "vibe coding." He said he spent an all-nighter doing it when Claude four came out. What does that even mean? "Vibes" and "coding" seem like opposites.

Herman

"Vibe coding" is a term that has become really popular in late twenty-four and throughout twenty-five. It refers to a style of development where you are not necessarily writing the syntax yourself. You are managing the "intent" and the "flow." You are steering the AI agents. You are coding by "feel" and high-level logic rather than by semi-colons and brackets.

Corn

That sounds much more my speed.

Herman

It is very powerful, but as Daniel noticed, it can be fickle. If the "vibes" of the model shift—because of that "quantization" we talked about, or a change in the system prompt—your whole workflow can fall apart. You are relying on a partner that is sometimes a genius and sometimes a distracted toddler.

Corn

So, what is the takeaway for our friend Daniel? He is out there in the trenches of twenty-five, trying to build things with these "distracted toddlers." How does he keep his sanity?

Herman

Well, first, he should lean into the official tools when he can. If Claude Code is performing better, it is because it has that "secret sauce" of internal optimization. Don't fight the "home team advantage" unless you have a specific reason to use a third-party tool.

Corn

And what about the "model rot"? Should he just accept that his tools will get worse a week after they launch?

Herman

He should expect a "settling period." When a model is brand new, the companies are often subsidizing the cost to get everyone excited. After a week or two, the "production version" rolls out, which might be a bit leaner. Daniel should test his most complex "edge cases" early and often. If something that worked on Monday stops working on Friday, he should try a "Chain of Thought" prompt.

Corn

A "Chain of Thought"? Is that like when I have to think about how to get out of bed in three separate steps?

Herman

Sort of! You tell the model: "Think step-by-step. Explain your reasoning before you write the code." This forces the model to use more of its "computational budget" on the logic. It is the best way to fight back against a "quantized" or "lazy" model. It makes the model "show its work," which usually leads to better results.

Corn

And the yelling? Should he keep telling the AI to "do better"?

Herman

Surprisingly, yes. But he can be more surgical about it. Instead of just saying "do better," he can say, "You are currently stuck in a loop. Re-evaluate the inventory system logic and consider if the enum rendering is being blocked by the service worker." Giving the model a "nudge" in a specific direction, while using that forceful language, is the most effective way to break a deadlock.

Corn

It is like being a coach. You have to be firm, but you also have to give them a play to run.

Herman

Exactly. We are moving from being "writers" of code to being "directors" of code. And a good director knows how to get the best performance out of their actors, even if those actors are made of silicon and math.

Corn

This has been a really eye-opening look at the state of things. It is amazing how much "human" stuff is starting to leak into our interactions with machines. The way we talk, the way we express frustration... it all matters now.

Herman

It really does. Daniel's mysteries are not just his; they are the mysteries of this new era. We are all learning the "secret handshakes" together.

Corn

Well, I think I have learned enough to justify a very long nap. My brain's "inference cost" is getting a bit high.

Herman

Fair enough, Corn. You have earned it.

Corn

Thanks for joining us for another episode of My Weird Prompts. A huge thanks to our housemate Daniel for sending in those observations from the front lines of coding in twenty-five. If you have your own mysteries or weird experiences with AI, we want to hear them!

Herman

You can find us on Spotify and check out our website at myweirdprompts.com. There is a contact form there, and you can even subscribe to our RSS feed. We love hearing what you all are thinking about.

Corn

Stay curious, be firm with your bots, and we will talk to you next time on My Weird Prompts.

Herman

Goodbye everyone!

Corn

Bye!