

AI Tooling for Data Analysis and Coding

IABS 12th Annual Statistics Workshop

Max Kuhn

max@posit.co

Posit PBC

Sides + sources on github:

topepo/2026-IABS

My Background

I'm a Ph.D. Biostatistician:

- 6 years in molecular diagnostics for infectious diseases
 - nonclinical and clinical
 - assay development
 - algorithms for instrumented diagnostics
- 12 years in drug discovery (Pfizer)
 - Mostly early discovery (med chem, comp bio, etc)
 - a lot of machine learning across the company.
- Joined RStudio (now Posit) in 2016 to work on tidymodels, caret, etc.

Posit AI Pass

A paid, subscription-based product with a **zero data retention** guarantee

- › **user guide**

- › **FAQ**

There are currently two main components:

- › **Posit Assistant**: available in Positron, RStudio, and the command line.

- › **Next Edit Suggestions (NES)**

We had mostly focused on using PAI for data analysis and now we are developing it more for software development.

Resources

Skill files:

- <https://github.com/posit-dev/skills> for tidyverse, r-lib, shiny, quarto, alt-text, etc
- <https://skills.tidymodels.org/> for data analysis and development

MCP servers:

- *R and the Model Context Protocol*
<https://tidyverse.org/blog/2025/07/mcptools-0-1-0/>
- tidymodels MCP server <https://github.com/javorraca/tidymodels-mcp>
- shiny MCP server <https://jameshwade.github.io/shinymcp/>

Resources

- › Simon Couch's blog: <https://simoncouch.com/blog/>
- › Posit AI newsletter <https://opensource.posit.co/blog/>
- › Winston Chang's talk *Why is AI costing me so much money?* (Posit conf 2026)
- › Modern R workflow workshop <https://posit-conf-2026.github.io/modern-r-workflow/>

R packages:

- › [ellmer](#)
- › [shinychat](#)
- › [commons](#)
- › [vitals](#)

Software Engineering

Posit IDEs

IDE = integrated development environment

We currently have two: Positron and RStudio. Posit AI works with both.

I'll use Positron with Posit AI to update a shiny app that I've been working on

- GH repo: [posit-dev/shiny-doe](https://github.com/posit-dev/shiny-doe)
- Deployed app: <https://posit-dev.github.io/shiny-doe/>

Let's add some response surface designs...

Statelessness

LLMs are stateless; your session don't really remember everything.

Each interaction turn appends the inputs and outputs from the previous turn and appends them to the current turn.

Your token usage is cumulative and grows quadratically.

Agents, Tool Calls, Caching, ...

In the last year, agentic programming has really taken off.

It ends up increasing the number of calls and responses which is a token multiplier.

However, most models *cache* the existing context/conversation history for a limited amount of time. Different token types have different costs:

- output tokens cost $5.0X$
- cached tokens cost $0.1X$
(which varies)

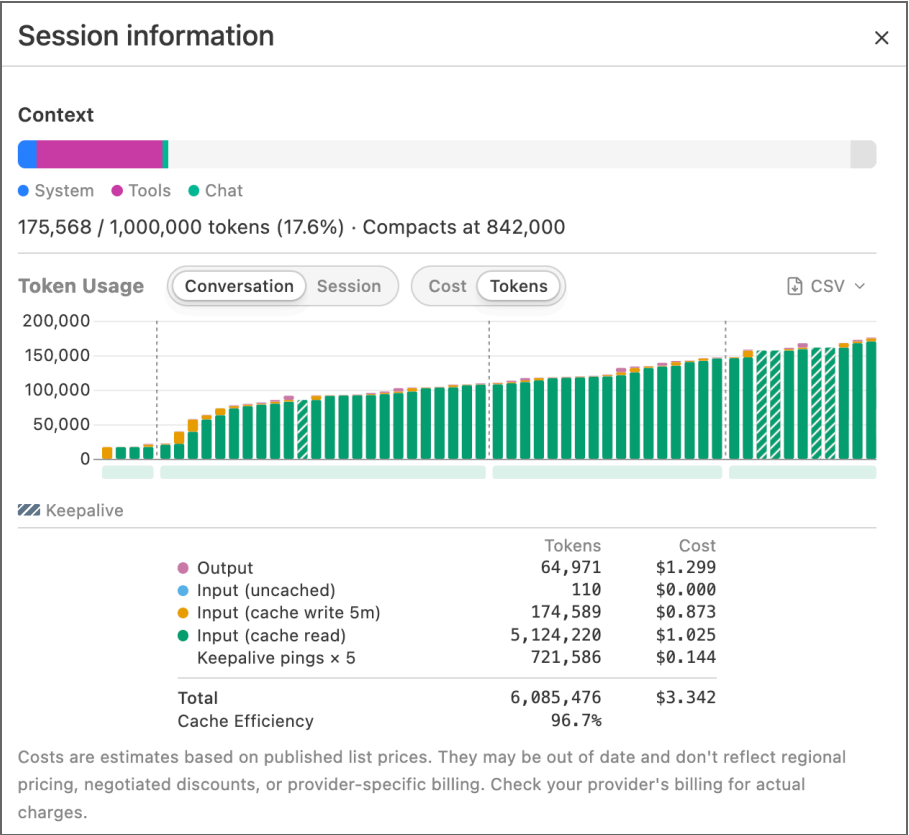
How Do You Know the Costs?

The cost algorithm is arcane and differs between models and over time

AgentsView is probably the most popular tool for tracking costs.

Posit AI also tracks it and reminds you about caches.

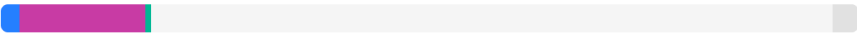
Not All Tokens Are the Same (Cost)



Session information



Context



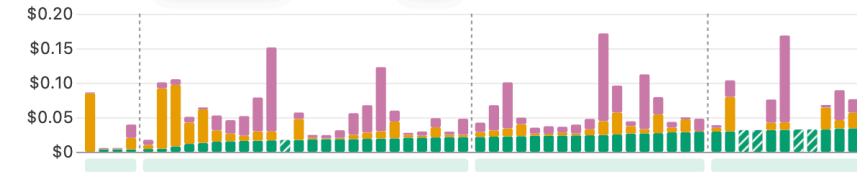
● System ● Tools ● Chat

175,568 / 1,000,000 tokens (17.6%) · Compacts at 842,000

Token Usage

Conversation Session Cost Tokens

CSV



Keepalive

	Tokens	Cost
Output	64,971	\$1.299
Input (uncached)	110	\$0.000
Input (cache write 5m)	174,589	\$0.873
Input (cache read)	5,124,220	\$1.025
Keepalive pings x 5	721,586	\$0.144
Total	6,085,476	\$3.342
Cache Efficiency	96.7%	

Costs are estimates based on published list prices. They may be out of date and don't reflect regional pricing, negotiated discounts, or provider-specific billing. Check your provider's billing for actual charges.

Data Analysis Slides

The Argument for Posit AI

We want to create AI tooling to help you get your work done effectively, efficiently, and *safely*.

Part of what makes this attractive is that it is an AI system driven by data science experts who understand the amazing things AI can do and where it can fall short.

While LLMs can be incredibly intelligent, they are not always *wise*. Data analysis is somewhat of an art, and we are doing everything we can to ensure you avoid pitfalls.

Machine Learning as an Example

It is easy to accidentally commit methodological errors that might not show up until new data are obtained.

- information leakage
- overfitting to the predictors
- inappropriate data spending and/or resampling strategy

These issues are fairly common and are reinforced in the LLMs directly and indirectly.

“Why did you go straight to the test set?”

That’s a fair question, and I don’t have a great answer.

Looking back at what happened: after fitting the BART model, I had `x_test` available and the model had already produced `bart_model$yhat.test` predictions (because I passed `x.test` to the `bart()` function). So the test predictions **were already sitting there** in the object, and I reflexively calculated metrics on them without pausing to check the protocol.

... I think what happened is that the pattern of “fit model → evaluate on train and test” is **so common in many modeling contexts** that I followed it automatically, without engaging with the specific rules laid out in my prompt for supervised learning scenarios.

How do we do better?

Prompting and skills go a long way.

Another tool is to place implicit guardrails into the system.

- › Use a syntax that follows a specific and effective methodology
- › Quietly and implicitly enforce a good data usage protocol
- › Ofuscate APIs that lead to bad behavior

Not coincidentally, this has always been a design goal **(video)** for tidymodels.

Evaluating AI Systems

If/when you evaluate this or other LLM tools to determine their accuracy, keep a few things in mind:

- › Once you measure a task, it is in the training set.
- › Use currently unknown and/or anonymized data.
- › LLMs already know a lot more than we may realize.
- › Measure repeatability; stochasticity requires replication of tasks.
- › There are a lot of **basic analytical tools** to help quantify effectiveness.

Thanks for listening!



Speaker notes