Become a Better Data Scientist with These Prompt Engineering Tips and Tricks

Why Prompt Engineering is a superpower for DSs

Sara Nobrega • 10 min read • 2025-06-30

https://towardsdatascience.com/become-a-better-data-scientist-with-these-prompt-engineering-hacks/


10101
10111010

Part 1: prompt engineering for planning, cleaning, and EDA

Image generated with DALL-E.

In this article, I am sharing with you my favorite prompts and prompt engineering tipsthat help me tackle Data Science and AI tasks.

As **Prompt Engineering** is emerging as a required skillin mostjobdescriptions, I thought it would be useful to share with you some tips and tricks to improve your Data Science workflows.

We are talking here about specific promptsfor cleaning data, exploratory data analysis, and feature engineering.

This is the **first** of a series of **3 articles** I am going to write about Prompt Engineering for Data Science:

• **Part 1: Prompt Engineering for Planning, Cleaning, and EDA** (this article)

• **Part 2:** Prompt Engineering for Features, Modeling, and Evaluation

• **Part 3:** Prompt Engineering for Docs, DevOps, and Learning

👉 All the promptsin this article are available at the end of this article as a cheat sheet 😉

In this article:

1. Why Prompt Engineering Is a Superpower for DSs

2. The DS Lifecycle, Reimagined with LLMs

3. Prompt Engineering for Planning, Cleaning, and EDA

I know, *Prompt Engineering* sounds just like a trending buzzword these days. I used to think that when I started hearing the term.

I would see it everywhere and think: it is just writing a prompt. Why are people so overhyped about it? What could be so difficult about it?

After testing several prompts and watching several tutorials, I now understand that it is one of the most useful (and also underestimated) skills a data scientist can acquire right now.

It is already common to see in the job descriptionsthat prompt engineering is one of the required skills for the job.

Reflect with me: how often do you ask ChatGPT/Claude/your fav chatbot to help you re-write code, clean data, or just brainstorm a project or some ideas you have? And how often do you get useful and meaningful, non-generical answers?

> **Prompt Engineering** is the art (and science) of getting large language models (LLMs) like GPT-4 or Claude to actually do what you want, when you want it, in a way that makes sense for your workflow.

Because here's the thing: **LLMs are everywhere now.**
In your notebooks.
In your IDE.
In your BI dashboards.
In your code review tools.

And they're only getting better**.**

As data science work gets more complex-more tools, more expectations, more pipelines-being able to *talk* to AI in a precise, structured way becomes a serious advantage.

I see prompt engineering as a superpower. Not just for junior folks trying to speed things up, but for experienced data scientists who want to work smarter.

In this series, I'll show you how prompt engineering can support you *at every stage* of the data science lifecycle-from brainstorming and cleaning, to modeling, evaluation, documentation, and beyond.

The DS lifecycle, reimagined with LLMs

When you are building a Data Science or Machine Learning project, it really feels like a whole journey.

From figuring out what problem you're solving, all the way to making a stakeholder understand *why* your model matters (without showing them a single line of code).

Here's a **typical DS lifecycle:**

- You **plan & brainstorm**, to figuring out the right questions to ask and what problems need to be solved
- You **gather** data, or data is gathered for you.
- You **clean data and preprocess it** - this where you spend 80% of your time (and patience!).
- The fun begins: you start making exploratory data analysis **(EDA)** - getting a feel for the data, finding stories in numbers.
- You start building: **feature engineering** and **modeling** begins.
- Then, you **evaluate** and **validate** if things actually *do* work.
- Finally, you **document** and **report** your findings, so others can understand it too.

Now... imagine having a helpful assistant that:

- Writes solid **starter code** in seconds,
- Suggests better ways to **clean** or **visualize** data,
- Helps you explain **model performance** to non-tech people,
- Reminds you to check for things you might miss (like data leakage or class imbalance),
- And is available 24/7.

That's what LLMs can be, **if you prompt them the right way!**

They won't replace you, do not worry. They are not able to do it!

But they can and will definitely amplify you. You still need to know **what** you're building and **how** (and why!), but now you have an assistant that allows you to do all of this in a smarter way.

Now I'll show you how prompt engineering can amplify you as a data scientist.

1. Planning & brainstorming: No more blank pages

You've got a dataset. You've got a goal. Now what?

You can prompt GPT-4 or Claude to list steps for an end-to-end project given a dataset description and goal.

This phase is where LLMs can already give you a boost.

Example: Planning an energy consumption prediction project

Here's an actual prompt I've used (with ChatGPT):

> "You are a senior data scientist. I have an energy consumption dataset (12,000 rows, hourly data over 18 months) including features like temperature, usage_kwh, region, and weekday. Task: Propose a step-by-step project plan to forecast future energy consumption. Include preprocessing steps, seasonality handling, feature engineering ideas, and model options. We'll be deploying a dashboard for internal stakeholders."

This kind of structured prompt gives:

• **Context** (dataset size, variables, goal)

• **Constraints** (class imbalance)

• Hints at **deployment**

Note: if you are using ChatGPT's newest model, **o3-pro**, make sure to give it a **lot** of context. This new model thrives when you feed it with full transcripts, docs, data, etc.

A similar **Claude** prompt would work, as Claude also favors explicit instructions. Claude's larger context window even allows including more dataset schema details or examples if needed, which can yield a more tailored plan

I re-tested this prompt with o3-pro as I was curious to see the results

The response from o3-pro was nothing less than a full data science project plan, from cleaning and feature engineering to model selection and deployment, but more importantly: with critical decision points, realistic timelines, and questions that challenge our assumptions upfront.

Here is a snapshot of the response:

Image by author.

Bonus strategy: Clarify - Confirm - Complete

If you need a more complex planning, there is a trick called "Clarify, Confirm, Complete" that you can use before the AI gives the final plan.

You can ask the model to:

1. Clarify what it needs to know first

2. Confirm the right approach

3. Then **complete** a full plan

For example:

“I want to analyze late deliveries for our logistics network.
Before giving an analysis plan:

1. **Clarify** what data or operational metrics might be relevant to delivery delays
2. **Confirm** the best analysis approach for identifying delay drivers
3. Then **complete** a detailed project plan (data cleaning, feature engineering, model or analysis techniques, and reporting steps).”

This approach forces the LLM to first ask questions or state assumptions (e.g., about available data or metrics). This forces the model to slow down and think, just like we humans do!

Data cleaning & preprocessing: Bye bye boilerplate

Now that the plan’s ready, it’s time to roll up your sleeves. Cleaning data is 80% of the job, and for sure not a fun task.

GPT-4 and Claude can both generate code snippets for common tasks like handling missing values or transforming variables, given a good prompt.

Example: Write me some pandas code

Prompt:

“I have a DataFrame `df` with columns `age`, `income`, `city`.
Some values are missing, and there are income outliers.
Task:

1. Drop rows where `city` is missing
2. Fill missing `age` with the median
3. Cap income outliers using IQR method
   Include comments in the code.”

Within seconds, you get a code block with `dropna()`, `fillna()`, and the IQR logic, all with explanations.

Example: Guidance on cleaning strategies

You can query conceptual advice as well.

Prompt:

“What are different approaches to handle outliers in a financial transactions dataset? Explain when to use each and the pros/cons.”

A prompt’s answer like this will output the multiple methodsspecific to your niche, instead of a one-size-fits-all solution.

This helps avoid the **simplistic** or even misleading advice one might get from a too-general question (for example, asking “best way to handle outliers” will probably output an oversimplified “remove all outliers” recommendation.

Try few-shot prompting for Consistency

Need variable descriptions in a consistent format?

Just show the LLM how:

Prompt:

“Original: “Customer age” !’ Standardized: “Age of customer at time of transaction.”
Original: “purchase_amt” !’ Standardized: “Transaction amount in USD.”

Now standardize:

• Original: “cust_tenure”

• Original: “item_ct” “

It follows the style perfectly. You can use this trick to standardize labels, define features, or even describe model steps later.

Exploratory data analysis (EDA): Ask better questions

EDA is where we start asking, “What’s interesting here?” and this is where vague prompts can really hurt.

A generic “analyze this dataset” will often return... generic suggestions.

Examples: EDA tasks

“I have an e-commerce dataset with `customer_id`, `product`, `date`, and `amount`.
I want to understand:

1. Purchase behavior patterns

2. Products often bought together

3. Changes in purchasing over time
For each, suggest columns to analyze and Python methods.”

The answer will probably include grouped stats, time trends, and even code snippets using `groupby()`, `seaborn`, and `market basket analysis`.

If you already have **synopsis statistics**, you could even paste them and ask:

Prompt:

“Based on these summary stats, what stands out or what potential issues should I look into?”.

GPT-4/Claude might point out a highvariance in one feature or a suspicious number of missing entries in another. (**Be cautious:** the model can only infer from what you show; it may hallucinate patterns if asked to speculate without data.)

Example prompt: Guided EDA

“I have a dataset with 50 columns (mix of numeric and categorical). Suggest an exploratory data analysis plan: list 5 key analyses to perform (e.g., distribution checks, correlations, etc.). For each, specify which specific columns or pairs of columns to look at, given I want to understand sales performance drivers.”

This promptis specific about the goal (sales drivers) so the AI might recommend, say, analyzing sales vs marketing_spend scatter plot, a time series plot if date is present, etc., customized to

"performance drivers." Besides, the structured output (list of 5 analyses) will be easier to follow than a long paragraph.

Example: Let the LLM explain your plots

You can even ask:

"What can a box plot of income by occupation tell me?"

It will explain quartiles, IQR, and what outliers might mean. This is more helpful when mentoring juniors or preparing slides for reports, presentations, etc.

Pitfalls to be careful about

This early stage is where most people misuse LLMs. Here's what to watch for:

Broad or vague prompts

If you say: "What should I do with this dataset?"
You'll get something like: "Clean the data, analyze it, build a model."

Instead, *always include*:

• **Context** (data type, size, variables)

• **Goals** (predict churn, analyze sales, etc.)

• **Constraints** (imbalanced data, missing values, domain rules)

Blind trust in the output

Yes, LLMs write code fast. But test everything.

I once asked for code to impute missing values. It used `fillna()` for *all columns*, including the categorical ones. It didn't check data types, and neither did I... the first time. 😬

Privacy and leakage

If you're working with real company data, **don't paste raw rows** into the prompt unless you're using a private/enterprise model. Describe the data abstractly instead. Or even better, consult your manager about this matter.

Thank you for reading!

👉 Grab the **Prompt Engineering Cheat Sheet** with all prompts of this article organized. I will send it to you when you subscribe to **Sara's AI Automation Digest**. You'll also get access to an AI tool library and my free AI automation newsletter every week!

Thank you for reading! 😉

I offer **mentorship** on career growth and transition here.

If you want to support my work, you can buy me my **favorite coffee**: a cappuccino. 😊

References

A Guide to Using ChatGPT For Data Science Projects | DataCamp

(29) Prompt Engineering for Document Analysis: What I Learned Moving from GPT-4 to Claude 4 🧠 | LinkedIn

Prompt Engineering for Data Professionals - Dataquest

Geeks for Geeks

Written By

Share This Article

- Share on Facebook
- Share on LinkedIn
- Share on X

Towards Data Science is a community publication. Submit your insights to reach our global audience and earn through the TDS Author Payment Program.

Other mentions by Author

- towardsdatascience.com | Related Articles
- towardsdatascience.com | Making Sense of the Promise (and Risks) of Large Language Models
- towardsdatascience.com | The Creative, Occasionally Messy World of Textual Data
- towardsdatascience.com | Deep Dive into LlaMA 3 by Hand ✍️
- towardsdatascience.com | Deep Dive into Transformers by Hand ✍️
- towardsdatascience.com | Deep Dive into Sora's Diffusion Transformer (DiT) by Hand ✍️
- towardsdatascience.com | UniFliXsg: AI-Powered Undergraduate Program Recommendations for Singapore Universities