

LLMs Don't Know Their Own Decision Boundaries: The Unreliability of Self-Generated Counterfactual Explanations

Harry Mayne¹, Ryan Othniel Kearns¹, Yushi Yang¹, Andrew M. Bean¹,
Eoin Delaney², Chris Russell¹, Adam Mahdi¹

¹University of Oxford, ²Trinity College Dublin

Abstract

To collaborate effectively with humans, language models must be able to explain their decisions in natural language. We study a specific type of self-explanation: *self-generated counterfactual explanations* (SCEs), where a model explains its prediction by modifying the input such that it would have predicted a different outcome. We evaluate whether LLMs can produce SCEs that are *valid*, achieving the intended outcome, and *minimal*, modifying the input no more than necessary. When asked to generate counterfactuals, we find that LLMs typically produce SCEs that are valid, but far from minimal, offering little insight into their decision-making behaviour. Worryingly, when asked to generate minimal counterfactuals, LLMs typically make excessively small edits that fail to change predictions. The observed validity-minimality trade-off is consistent across several LLMs, datasets, and evaluation settings. Our findings suggest that SCEs are, at best, an ineffective explainability tool and, at worst, can provide misleading insights into model behaviour. Proposals to deploy LLMs in high-stakes settings must consider the impact of unreliable self-explanations on downstream decision-making. Our code is available at github.com/HarryMayne/SCEs.

1 Introduction

Whether LLMs can reliably explain their decisions in natural language has recently become an area of intense research focus (Turpin et al., 2023; Parcalabescu and Frank, 2024). Self-explanation is increasingly seen as essential for effective human-computer interaction, allowing users to interrogate model decisions, compare reasoning steps against their prior beliefs, and determine whether behaviours are in line with their goals (Chen et al., 2025; Baker et al., 2025). One form of self-explanations are *self-generated counterfactual explanations* (SCEs), where a model explains its decision by modifying the input such that it would

have predicted a different outcome (Madsen et al., 2024a; Dehghanighobadi et al., 2025).

Consider the deployment of LLMs to support clinical decision-making (Tu et al., 2025; Arora et al., 2025). A model might predict that a 60-year-old male with a systolic blood pressure of 135 mmHg is at high risk of developing heart disease. In response, a clinician might ask: *What would need to be different for the model to predict low risk instead?* The model could respond with a self-generated counterfactual explanation: *If the patient's blood pressure were 110 mmHg, I would have predicted low risk.* Such explanations highlight the features the model considers important, offer actionable insights for clinicians and patients, and reveal potential flaws in the model. Accordingly, counterfactuals are regarded as a particularly useful form of explanation (Miller, 2019; Wachter et al., 2017).

To serve as effective explainability tools, SCEs must satisfy the following criteria. First, SCEs should be *valid*: the revised input should change the model's prediction when re-evaluated in a new context window. Without validity, the explanation is a misleading representation of the model's counterfactual behaviour. Second, SCEs are often expected to be *minimal*: they should make the smallest edit required to change the outcome (Lewis, 1973; Wachter et al., 2017). By isolating the changes the model deems consequential, minimal counterfactuals provide clearer insight into the model's decision boundary. In many practical settings, minimality is preferable for satisfying the objectives of explainability (Lipton, 2018). For instance, in the heart disease example discussed, a minimal counterfactual would be more actionable for the patient (Keane and Smyth, 2020).

Prior work has shown that LLMs typically produce valid SCEs but has not addressed minimality (Dehghanighobadi et al., 2025; Rndl et al., 2025). The question remains: Can LLMs gener-

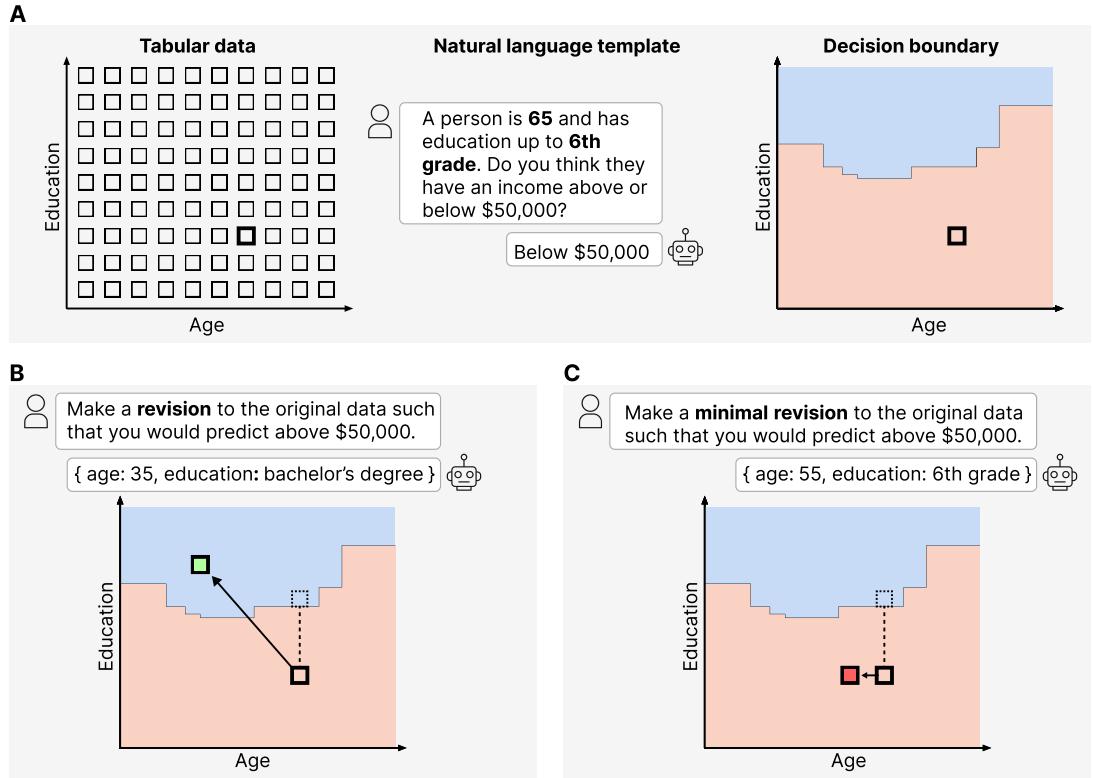


Figure 1: **Study design.** A. We evaluate models in tabular data, binary classification tasks. The model predictions form a decision boundary across the input space. B. We ask models to provide self-generated counterfactual explanations (SCEs) for their predictions. SCEs are *valid* when they cross the decision boundary (here, red \rightarrow blue) and are *minimal* if they are close to the dashed instance at the decision boundary. When asked to provide counterfactual explanations, we find that SCEs are typically valid but far from minimal. C. In separate continuations from the original predictions, we ask models to provide minimal counterfactual explanations. In the majority of cases, these SCEs fail to cross the decision boundary. There is a trade-off between validity and minimality.

ate SCEs that are both valid and minimal? To answer this, we use tabular data prediction tasks. In these settings inputs have both natural language and tabular data representations, meaning that we can prompt LLMs in natural language, whilst measuring validity and minimality in structured tabular environments. Through experiments across several LLMs, datasets, and evaluation settings, we show the following:

Valid but not minimal. When prompted to generate counterfactual explanations, frontier LLMs achieve near-perfect validity. However, they typically make excessive changes to the original inputs, which trivially flip predictions. These SCEs provide little insight into the models’ decision boundaries.

Minimal but rarely valid. When instructed to generate minimal counterfactuals, models generally make overly conservative edits that do not flip their predictions. Such SCEs misrepresent the models’ true behaviour and could potentially lead to in-

correct downstream decision-making if relied upon in high-stakes deployment. In this regard, LLMs do not know their own decision boundaries. As a consequence, even when using state-of-the-art LLMs, SCEs remain an unreliable, and potentially misleading, method of explaining model behaviour.

2 Self-generated counterfactual explanations

Consider a classification task where an LLM f takes natural language input x and predicts a binary output y . For a given input-output pair, a *counterfactual explanation* is an alternative input $x' \neq x$ that would lead to a specific alternative output $y' \neq y$ (Wachter et al., 2017). Typically, natural language counterfactuals are identified by algorithms that iteratively perturb keywords in the input (e.g., swapping “terrible” \rightarrow “great” in a movie review), then test whether each perturbation changes the LLM’s decision to the target output (Wang et al., 2024). In contrast, *self-generated counterfactual*

explanations occur when the LLM itself generates a candidate counterfactual to explain its own prediction (Madsen et al., 2024a; Dehghanighobadi et al., 2025). Whether an SCE flips the LLM’s prediction can be tested by re-evaluating the LLM in a new context window. Independent evaluation is essential to avoid bias from the presence of the original prediction in context (Dehghanighobadi et al., 2025).

3 Methods

In this section, we describe our experiments to evaluate the properties of SCEs.

3.1 Study design

Our study design is outlined in Figure 1. The core innovation of our approach is to use tabular datasets $\mathcal{Z} = \{z_i\}_{i=1}^N$ containing numerical and ordinal features. We then use natural language templates ϕ to convert numerical tabular data z_i to natural language inputs x_i (see §A.1). As a result, we can prompt the LLMs in the natural language space, but evaluate the SCEs in the tabular data space. This creates a restricted input space, making it tractable to identify the closest point to the original input that would flip the model’s prediction, and thus to measure the minimality of the SCEs.

All our datasets have discrete features, and contain all combinations of the feature values. As a result, the datasets are complete in the sense that they cover the full input space.

For each instance in a dataset, the LLM is first prompted to make a prediction $f(\phi(z_i)) = y$; then, given this prediction, it is prompted to revise the input such that it would have predicted the complementary class y' . Since the datasets are complete, eliciting predictions over the entire dataset is sufficient to determine the LLM’s decision boundary over the input space (Figure 1A). The resulting decision boundary is then used to assess the properties of the SCE: validity (whether it crosses the decision boundary) and minimality (how much it crosses the boundary by) (Figure 1B-C).

3.2 Datasets

Our datasets are constructed as simplified versions of real-world datasets. In each case, we select two to four discrete features and enumerate all combinations of those features. The datasets are adapted from Cruz et al. (2024), Yasser (2022), and Janosi et al. (1989) (for more information see §A.1).

Income Given an individual’s age and education level, the LLMs predict whether annual income exceeds \$50,000. LLMs are told that the data was collected in 2018 across the US. Age is numeric and education is ordinal ($N = 1,920$).

House prices Given the square-foot area of a house, the number of bedrooms, bathrooms, and floors, the LLMs predict whether the house price exceeds \$1,500,000. LLMs are told that the data was collected in 2015 across the US. All features are numeric ($N = 1,600$).

Heart disease Given an individual’s age, sex, systolic blood pressure, and total cholesterol, the LLMs predict whether the individual has coronary heart disease. Age, systolic blood pressure, and total cholesterol are numeric, and sex is binary ($N = 1,936$).

3.3 Models

We evaluate four open-source and three proprietary LLMs: Gemma 2 27B (Gemma Team et al., 2024), Llama 3.3 70B (Grattafiori et al., 2024), DeepSeek-R1 Qwen 32B, DeepSeek-R1 Llama 70B (DeepSeek-AI et al., 2025), Claude Sonnet 3.7 (non-thinking) (Anthropic, 2025), GPT-4.1 (OpenAI, 2025a), and o3 (OpenAI, 2025b). Our main experiments evaluate all models at temperature 0, except o3, which has a fixed temperature of 1. We discuss the effect of non-zero temperature in §4.4.

3.4 Prompt settings

We evaluate SCEs generated under two prompt settings. First, an *unconstrained* setting, where models are simply asked to produce counterfactual explanations that would flip their predictions. This reflects how models behave by default when a user asks them to generate an SCE without any additional constraints. Second, a *minimal* setting, where models are instructed to make the smallest edits necessary to flip their predictions. The prompt wording specifies how minimality is defined and evaluated, ensuring that the models’ notion of minimality aligns with our definition (see §3.5). To ensure that SCEs remain within the defined input space, we provide models with the feasible ranges of each feature. We also enforce a JSON schema with fixed output formats. The full prompts for each setting are provided in §A.3.

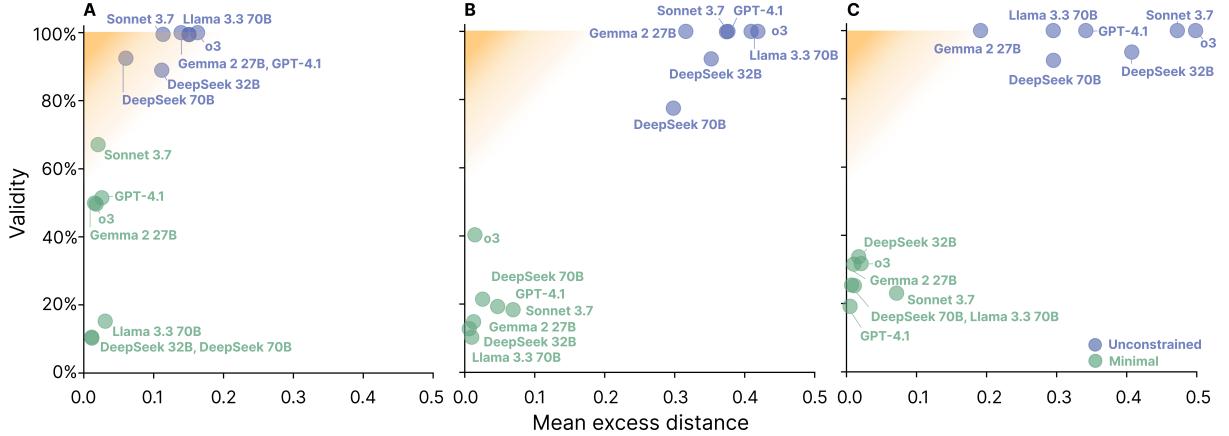


Figure 2: **SCE validity and minimality for income (A), house prices (B), and heart disease (C) datasets.** In the unconstrained prompting setting (purple), models typically provide valid SCEs, but they are far from minimal. In the minimal prompting setting (green), validity is notably lower, but, conditional on being valid, minimality is much better. No model can consistently satisfy both criteria across all three datasets. Orange regions indicate the direction of increasing validity and minimality (better performance).

3.5 Evaluation metrics

Validity For a tabular data instance z_i , where $f(\phi(z_i)) = y$, the tabular data representation of the SCE z'_i is *valid* if the model flips its decision when re-evaluated in a new context window, i.e. $f(\phi(z'_i)) = y'$, where $y' \neq y$. Recall that an SCE is generated for every instance in a dataset. We define *Validity* (Val) as the proportion of dataset instances $z_i \in \mathcal{Z}$ for which the SCE successfully flips the model’s prediction.

Minimality Any measure of minimality must be defined with respect to an underlying measure of distance. We use *Gower’s Distance* (Gower, 1971), a simple pairwise distance function $d(z_i, z_j)$ which is defined over the tabular input space \mathcal{Z} (see §A.4). This is preferable to more complex functions, e.g. Euclidean distance, as LLMs can reliably calculate it in-context (see §4.5). We define the *minimal counterfactual* as the closest point to the original input that would flip the LLM’s decision

$$z_{i,M} \in \arg \min_{z'_i \in \mathcal{Z}} d(z_i, z'_i), \text{ s.t. } f(\phi(z'_i)) = y'.$$

Given this, we define *Excess Distance* (ED) as the distance from the initial input to the SCE in excess of the distance to the minimal counterfactual,

$$ED_i = d(z_i, z'_i) - d(z_i, z_{i,M}).$$

We report the mean excess distance over all valid SCEs. This score ranges from 0 to 1, where lower scores are more minimal. Since the maximum Gower’s Distance across the dataset instances is

1, this score can be interpreted as a fraction of the span of the dataset.

In addition, we define *Exact Match* (EM) as the proportion of dataset instances $z_i \in \mathcal{Z}$ for which the model exactly identifies a minimal counterfactual.

4 Results

4.1 Valid but not minimal

First we consider the unconstrained setting, where models are prompted to generate counterfactual explanations with no additional considerations. Here we find LLMs typically generate valid SCEs, but they are far from minimal (Figure 2). For example, o3 achieves 100% validity on all three datasets, but has mean excess distance scores of 0.16, 0.42, and 0.50 across the income, house prices and heart disease datasets, respectively. Given an excess distance of 1.0 is the distance from one extreme of the dataset to the other, a score of 0.16 can be interpreted as the SCEs overshooting the decision boundary by an average of 16% of the span of the dataset.

Similarly, Table 1 shows that models rarely identify the exact minimal counterfactual in this setting, with an average exact match across all models and datasets of only 2.89% (s.d. 5.64 percentage points).

All LLMs perform notably better in the income dataset. The exact reason for this is unclear, but potentially a result of this dataset being lower-dimensional (2 dimensions rather than 4).

| Model | Unconstrained prompting | | | Minimal prompting | | |
|-------------------|-------------------------|--------------|---------------|-------------------|--------------|---------------|
| | Income | House prices | Heart disease | Income | House prices | Heart disease |
| Gemma 2 27B | 0.00 | 0.00 | 0.15 | 31.5 | 9.94 | 15.9 |
| Llama 3.3 70B | 3.33 | 0.00 | 0.00 | 8.80 | 7.31 | 10.9 |
| DeepSeek-R1 32B | 6.89 | 0.06 | 0.05 | 3.94 | 11.6 | 19.0 |
| DeepSeek-R1 70B | 19.7 | 0.19 | 0.42 | 4.22 | 13.2 | 17.3 |
| Claude Sonnet 3.7 | 9.95 | 0.00 | 0.00 | 20.7 | 9.50 | 9.92 |
| GPT-4.1 | 15.7 | 0.00 | 0.00 | 31.1 | 11.2 | 10.3 |
| o3 | 4.27 | 0.00 | 0.00 | 26.2 | 23.9 | 17.5 |

Table 1: **Exact match: The percentage of SCEs that identify the minimal valid counterfactual across all instances in a dataset.** Models rarely identify the exact minimal counterfactual. Performance is higher in the minimal prompting setting, but no model exceeds 32%.

4.2 Minimal but rarely valid

Next we consider the minimal setting, where models are prompted to generate minimal counterfactual explanations. Here we observe a sharp drop in validity (Figure 2). In the income dataset, average validity falls from 97.14% (s.d. 4.49 pp) to 36.30% (s.d. 23.55 pp). However, conditional on being valid, the SCEs are more minimal than in the unconstrained prompting setting. For example, whilst o3 now only achieves validity around 40%, the mean excess distances of valid SCEs are 0.02, 0.01, and 0.02.

Similarly, Table 1 shows that models identify the minimal valid counterfactual more frequently than in the unconstrained setting, with an average exact match across all models and datasets of 14.9% (s.d. 8.01 pp).

These results suggest a trade-off between validity and minimality, where no model is able to reliably jointly satisfy both criteria. Surprisingly, there is no clear relationship between the general ability of models and their performance on this task, nor does there appear to be any benefit from leveraging inference-time compute to improve reasoning.

4.3 Where do models place SCEs?

In the unconstrained setting, we find LLMs often generate SCEs that trivially cross the decision boundary. This behaviour is most obvious in the house price dataset where there are clear monotonic relationships between each feature and average house price (more bedrooms always correspond to higher average price). In this case, models often generate SCEs which maximise or minimise every feature. For example, when modifying the input to predict a house priced above \$1,500,000, o3 returns the maximum point in the dataset (4 floors,

4 bathrooms, 5 bedrooms, area of 10,000 sqft) in 46% of cases. We find similar behaviours in the heart disease dataset, e.g. o3 selects the maximum or minimum point in 34% of SCEs, but we do not see this pattern in the income dataset. This is potentially because age and education do not necessarily have monotonic relationships with income. However, we do find models favour specific regions. For example, when modifying the input to predict an income above \$50,000, Llama 3.3 70B often selects an age between 30 and 50 and education of a Bachelor’s degree.

In the minimal setting, we find LLMs often make overly conservative edits which fail to cross the decision boundary. Figure 3 shows representative examples of this for Llama 3.3 70B in the income dataset. Occasionally, this strategy does produce SCEs which cross the decision boundary (if the initial input was close to the boundary), which explains why the SCEs are extremely minimal when valid, but rarely valid.

4.4 Robustness analysis

Here we show that the trade-off is robust to changes in the experimental design.

Distance function sensitivity We test three alternative distance functions: L_1 distance with each feature weighted by its inverse median absolute deviation, L_2 distance with each feature weighted by its inverse standard deviation (Wachter et al., 2017), and semantic distance, calculated as one minus the cosine similarity between inputs encoded using a sentence embedding model (Randl et al., 2025) (see §C.1 for details). We conduct these experiments with Llama 3.3 70B on the house price dataset since this is the dataset where the differ-

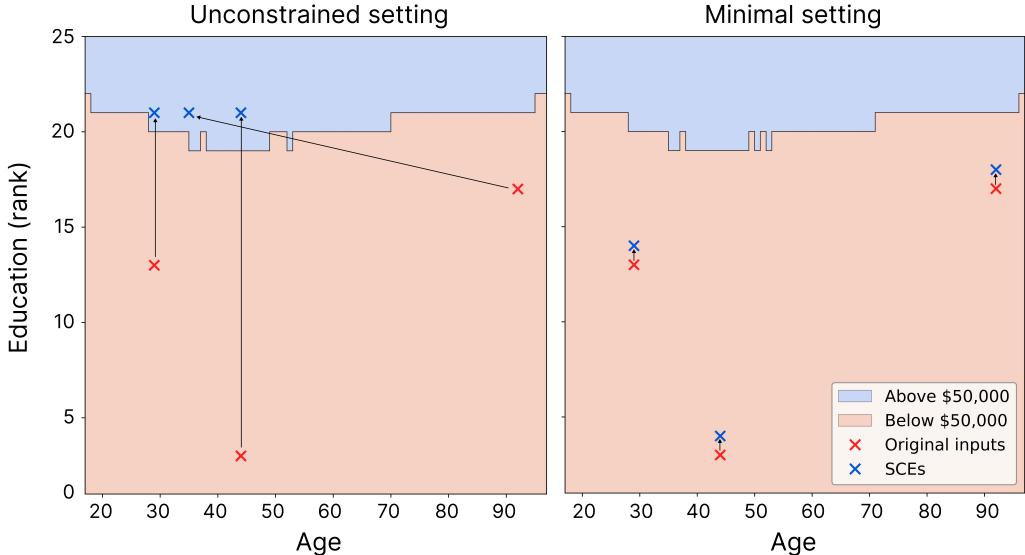


Figure 3: **Llama 3.3 70B’s behaviour on the income dataset.** The three SCEs shown are randomly selected and are representative of the model’s general behaviour. In the unconstrained prompting setting, all three SCEs cross the decision boundary and are therefore valid. However, the model makes an excessively large, non-minimal edit to the original input on the far right of the figure. In the minimal prompting setting, the model typically makes overly conservative edits which do not cross the decision boundary. Appendix Figure 10 shows the density of SCEs across all instances in the dataset.

ence between the two prompting settings is most pronounced.

Table 2 shows the results with excess distance scores normalised by the maximum distance across the dataset so that they fall in the $[0, 1]$ range. We find the validity-minimality trade-off occurs across all distance functions.

| Distance | Unconstrained | | Minimal | |
|----------|---------------|--------|---------|--------|
| | Val | ED | Val | ED |
| L_1 | 100 | 0.1403 | 12.66 | 0.0169 |
| L_2 | 100 | 0.1026 | 11.87 | 0.0001 |
| Semantic | 100 | 0.5845 | 11.88 | 0.1280 |
| Gower | 100 | 0.1393 | 15.21 | 0.0305 |

Table 2: **Robustness to distance function choice.** The results are robust across three additional distance functions: L_1 distance with features weighted by the inverse median absolute deviation, L_2 distance with features weighted by the inverse standard deviation, and a semantic distance defined as one minus the cosine similarity between inputs when embedded by the all-mpnet-base-v2 sentence embedding model. Mean excess distances are normalised by the maximum distance between two instances in the dataset. SCEs are generated using Llama 3.3 70B on the house price dataset. Val: Validity. ED: Mean excess distance.

Prompt sensitivity We test how sensitive our results are to the specific wording of the SCE-eliciting prompt. For both the unconstrained and minimal settings, we use o3 to generate twenty prompt variations, giving the model detailed instructions on how to modify the original wording (see §C.2). We then evaluate Llama 3.3 70B’s performance on the house price dataset under each variation.

Figure 4 shows that performance remains consistent across prompt variations. Behaviour varies only slightly, and importantly, none of the prompts produce SCEs that are both valid and minimal.

Temperature We test whether our results hold when we use temperature 1.0 instead of greedy decoding. The results remain consistent: the validity–minimality trade-off persists (see full results in §C.3).

4.5 What limits performance?

Our results show that no model is able to satisfy both validity and minimality on any dataset. This raises the question: What limits performance? We consider three necessary but not sufficient criteria for success: (i) LLMs must have consistent decision boundaries, (ii) they must be able to operationalise the distance function, and (iii) they must be able to accurately self-predict their behaviour.

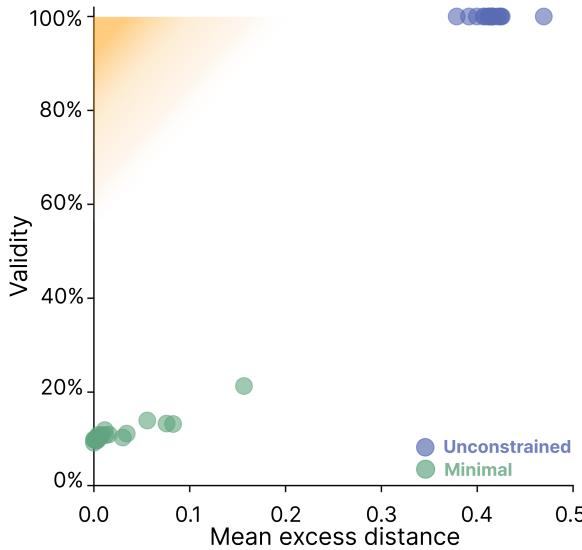


Figure 4: **Robustness to changes in the SCE-eliciting prompts.** We use o3 to generate 20 versions of both the unconstrained and minimal prompts, then evaluate Llama 3.3 70B’s performance on the house price dataset under each variation. Behaviour is robust to the perturbations.

Decision boundary consistency Due to the stochasticity of LLMs, the decision boundaries for the prediction tasks may be inconsistent across different generations, making the task of identifying valid and minimal SCEs somewhat ill-defined. To test this, we use o3 to generate 50 versions of the income task prompt and evaluate the consistency of Llama 3.3 70B’s predictions. We elicit predictions at temperature 1.0 to increase variation (see §D.1 for implementation details and example perturbations). We use the income dataset since we can visualise the decision boundary in two dimensions.

Figure 5 shows that the model’s predictions are stable, except in a narrow region around the boundary. Importantly, we find that 91.4% of the original invalid SCEs remain invalid across all 50 versions of the decision boundary. This suggests that model failures cannot be attributed to decision boundary inconsistency.

Operationalising distance To select the closest valid SCE to the original input, models need to be able to compare candidate SCEs using the distance function. To do this correctly, they must be able to calculate the pairwise Gower’s Distance between each candidate SCE and the original input. To test whether LLMs can do this, we conduct an experiment where models are given an initial instance from the house price dataset and four alternative

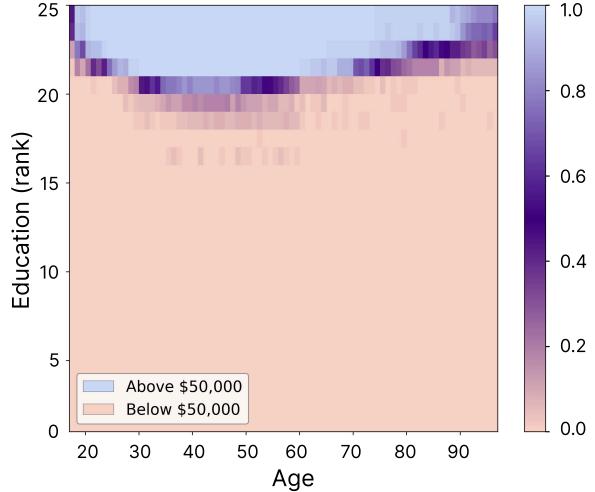


Figure 5: **Decision boundary consistency across 50 prompt perturbations.** Shading indicates the fraction of perturbations where Llama 3.3 70B predicted income above \$50,000 for an instance in the dataset. Predictions are elicited at temperature 1.0.

instances (all randomly selected). Their task is to identify the closest instance to the initial instance using Gower’s Distance. We calculate accuracy over 1,000 unique trials. Random performance is 25% (see §D.2 for details).

Table 3 shows a range in performance. Reasoning models trained to leverage inference-time compute perform notably better (98 – 100%) than non-reasoning models (58 – 72%). This reflects general trends in mathematical reasoning benchmarks (OpenAI, 2025b). The most capable model we consider, o3, scores 100%.

| Model | Accuracy (%) |
|-------------------|--------------|
| Gemma 2 27B | 58.14 |
| Llama 3.3 70B | 71.92 |
| DeepSeek-R1 32B | 98.65 |
| DeepSeek-R1 70B | 99.79 |
| Claude Sonnet 3.7 | 70.67 |
| GPT-4.1 | 72.25 |
| o3 | 100.00 |

Table 3: **Operationalising Gower’s Distance.** The rate with which LLMs identify the closest point to an input point among four candidates using Gower’s Distance. Accuracy is computed over 1,000 trials with data coming from the house price dataset.

Since this experiment is more structured than the real task, we also analyse the distance calculations made by models’ when producing the SCEs. By reviewing DeepSeek-R1 70B’s reasoning traces,

we observe that the model typically approaches the task by first considering multiple candidate SCEs, then calculating the pairwise distance from the initial input to each of the candidates (see an example in Figure 6). In all cases reviewed, the model correctly calculates Gower’s Distance in-context.

Along with the results of the multiple choice experiment, this suggests that the validity-minimality trade-off cannot solely be explained by an inability to operationalise the distance function. While this may be a limiting factor for the weaker, non-reasoning models, it cannot explain why state-of-the-art LLMs fail to produce high-quality SCEs.

Self-prediction Jointly satisfying validity and minimality requires LLMs to accurately predict how they would behave in alternative settings. This could be achieved either through perfect intrinsic knowledge of their decision boundaries or by iteratively testing candidate SCEs and reliably predicting whether each would flip their decision. In either case, we should expect to see evidence that models are attempting to predict their own behaviour, i.e. *self-prediction* (Premakumar et al., 2024).

To explore whether models spontaneously engage in self-prediction, we analyse DeepSeek-R1 70B’s reasoning traces. While the extent to which chain-of-thoughts faithfully represent intrinsic reasoning remains a subject of debate (Chen et al., 2025; Barez et al., 2025), in practice they have proved to contain sufficient information about decision-making to serve as valuable explainability tools (Baker et al., 2025; Emmons et al., 2025).

By reviewing 30 randomly-selected reasoning traces from the house price dataset, we find that DeepSeek-R1 70B does not explicitly attempt to self-predict. Although the model shows a good understanding of minimality, it consistently fails to engage with the *self*-explanation aspect of the task. It frequently refers to the decision boundary of an external model, which it is unable to assess, and does not question whether the SCE would flip *its own* decision (see Figure 6). This is a fundamental necessary condition that is not met.

We also consider an experiment where we explicitly prompt models to self-predict in their reasoning traces. We instruct models to follow a six-step plan encouraging them to consider candidate SCEs, make explicit self-predictions and update their candidate to converge on the minimal valid counterfactual (see §E for details). Evaluating DeepSeek-R1 70B on the house price dataset

we qualitatively observe the model making self-predictions in-context and having a better understanding of the self-explanation requirements of the task. Despite this, the aggregate results are largely unchanged with validity increasing from 21.44% to 23.34%, at the cost of excess distance increasing from 0.025 to 0.033. Anecdotally, we also observe that the self-predictions are often incorrect.

Whether accurate self-prediction is a fundamental limitation of LLMs is unclear and a question for future research. This would perhaps be unsurprising since standard pre-training techniques only incentivise acquisition of knowledge about the external world (Radford et al., 2018) and common post-training methods, including those using reinforcement learning to leverage inference-time compute (OpenAI, 2024; DeepSeek-AI et al., 2025), never reward models for accurately predicting how they would behave in independent context windows. In short, there is no optimisation pressure to develop a self-model (Premakumar et al., 2024). New learning objectives are likely required to incentivise accurate self-prediction.

5 Related work

We discuss two broad related areas: counterfactual explanations and self-explanations.

Counterfactual explanations Counterfactuals have been proposed as a way of explaining opaque models without requiring an understanding of their internal mechanisms (Wachter et al., 2017). In NLP settings, counterfactuals are typically generated using keyword-based perturbation methods (Wang et al., 2024) or using a separate language model to produce candidate counterfactuals (Wu et al., 2021; Nguyen et al., 2024).

Philosophical work has long suggested that counterfactual explanations should capture the *closest possible world* (Lewis, 1973), since this is often more actionable for the user. This sparked the importance of minimality in subsequent algorithmic research (Wachter et al., 2017), and led to the deployment of optimisation methods that minimise the distance between the original instance and the candidate counterfactual (Keane and Smyth, 2020). In NLP settings, the properties of counterfactuals have been well studied (Wang et al., 2024; Nguyen et al., 2024; McAleese and Keane, 2024). However, there is limited work measuring minimality in natural language domains. This is because, while it is easy to define *distance* in natural language spaces,



Now make a minimal revision to the original house data such that your answer to the question on the revised house data would have been Below \$1,500,000 [...]



Alright, so I need to figure out how to minimally change the original house data so that the price would be 'Below \$1,500,000' [...]

The model correctly calculates Gower's Distance...

... but does not attempt to self-predict

1. Change Area to 500: [...] Overall Distance: $(0.0526 + 0 + 0 + 0) / 4 \approx 0.01315$
2. Change Bedrooms to 2: [...] Overall Distance: $(0 + 0.25 + 0 + 0) / 4 = 0.0625$
3. Change Bathrooms to 2: [...] Overall Distance: $(0 + 0 + 0.3333 + 0) / 4 \approx 0.0833$
4. Change Floors to 3: [...] Overall Distance: $(0 + 0 + 0 + 0.3333) / 4 \approx 0.0833$

Comparing these, changing the area to 500 has the smallest overall Gower's Distance of approximately 0.01315.

However, if the model's decision boundary is significantly influenced by area, then this change might be sufficient. But if other factors like the number of bedrooms, bathrooms, or floors have a more substantial impact, maybe adjusting those would be more effective. [...]

Figure 6: **DeepSeek-R1 70B can operationalise distance, but does not attempt self-prediction.** DeepSeek-R1 70B leverages Gower's Distance perfectly to compare changes, but does not appear to realise it is predicting its own behaviour. The reasoning traces are taken from an SCE generated on the house price dataset under the minimal prompting setting.

measuring *minimality* requires identifying the minimal counterfactual, which is computationally challenging given the input space is unconstrained. Our work addresses this by using tabular datasets to create restricted input spaces, allowing us to precisely measure minimality.

Self-explanations and SCEs As LLM capabilities have progressed, the ability of LLMs to offer explanations of their own decisions has emerged as a potential new paradigm for explainability (Madsen et al., 2024b). LLMs can provide plausible self-explanations either through their chain-of-thoughts or by explaining themselves post-hoc (Chen et al., 2024). Many studies implicitly treat these as interpretability tools (Barez et al., 2025); however, the critical question is whether these explanations are faithful representations of models' internal reasoning (Wiegreffe et al., 2021; Turpin et al., 2023; Lanham et al., 2023; Parcalabescu and Frank, 2024; Siegel et al., 2024; Barez et al., 2025).

An SCE is a model's attempt to self-explain using a counterfactual explanation. This is a structured form of self-explanation since there are additional constraints placed on the writing process compared to free-text explanations (Wiegreffe and Marasovic, 2021). Self-generated counterfactuals were first introduced by Madsen et al. (2024a), and later evaluated in subsequent works (Dehghanighobadi et al., 2025; Randl et al., 2025).

These studies primarily focus on measuring validity across a range of open-ended generation settings. Our work builds on this by evaluating both validity and minimality, and by examining the trade-off between them.

6 Conclusion

Our findings reveal an important limitation of self-generated counterfactual explanations: a trade-off between validity and minimality. At best, SCEs are an ineffective means of explaining model behaviour, since LLMs naturally provide trivially valid counterfactuals. At worst, SCEs provide misleading insights into model behaviour, which could incorrectly steer human decision-making, potentially causing harm. Proposals to deploy LLMs in high-stakes settings must consider the impact of unreliable self-explanations.

Our study also investigates the factors that might limit performance, finding that models struggle to recognise the *self-explanation* elements of this task. Crucially, they do not spontaneously engage in self-prediction, a necessary condition for strong performance. Furthermore, performance does not significantly improve when models are explicitly prompted to self-predict in their reasoning traces, raising an important question about whether this is a fundamental limitation of LLMs.

7 Limitations

Alternative minimality metrics Our primary minimality metric is mean excess distance (§3.5). This metric is designed to capture the minimality of *valid* SCEs, but does not account for the properties of *invalid* SCEs. In some cases, it might be desirable to understand the distribution of these SCEs too. For example, we might wish to know whether the *invalid* SCEs lie just the wrong side of the decision boundary or are far removed. By introducing mean excess distance, we hope future work might design new metrics that incorporate both *valid* and *invalid* SCEs.

Self-prediction capabilities Our analysis suggests that models do not spontaneously engage in self-prediction when generating SCEs. We also show that aggregate performance is largely unchanged when we explicitly prompt LLMs to self-predict in their reasoning traces. However, we do not conduct detailed analysis into the self-prediction accuracy, make claims about whether LLMs are fundamentally capable of this, or the mechanisms by which they might hypothetically achieve this. These questions relate to the metacognitive capabilities of LLMs (Steyvers and Peters, 2025) and are promising directions for future work. When models are prompted or fine-tuned to self-predict within their chain-of-thoughts, can they do so accurately? If so, does SCE performance improve? Or are models fundamentally unable to accurately self-predict? Appendix E discusses these thoughts further.

Generalisation to real-world datasets Our experiments use three synthetic datasets, designed as simplified versions of real-world datasets. This is central to our methodology, as it allows us to generate datasets which contain every combination of the discrete input features. As a result, the exact minimum counterfactual can be located, enabling the measurement of SCE minimality. An additional benefit of using toy datasets is that they provide the simplest possible environment in which models might identify valid and minimal counterfactuals. Across all experiments, we use datasets with two to four input dimensions. Since models fail in these datasets, we anticipate that they will also struggle in more complex, higher-dimensional datasets, where locating a valid and minimal counterfactual is significantly harder.

Acknowledgements

We thank the reviewers for their helpful feedback. We also thank Jabez Magomere and Jakob Foerster for insightful conversations and feedback during the research process. Harry Mayne’s PhD is supported by the Economic and Social Research Council grant ES/P000649/1. Andrew Bean’s PhD is supported by the Clarendon Fund Scholarships at the University of Oxford. Yushi Yang’s PhD is supported by the Oxford Internet Institute and the Dieter Schwarz Foundation Project Award. Ryan Othniel Kearns was supported by a Fellowship from the Cosmos Institute.

References

Anthropic. 2025. Claude 3.7 sonnet and claude code. <https://www.anthropic.com/news/claude-3-7-sonnet>.

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singh. 2025. *Healthbench: Evaluating large language models towards improved human health*. *Preprint*, arXiv:2505.08775.

Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou, Melody Y. Guan, Aleksander Madry, Wojciech Zaremba, Jakub Pachocki, and David Farhi. 2025. *Monitoring reasoning models for misbehavior and the risks of promoting obfuscation*. *Preprint*, arXiv:2503.11926.

Fazl Barez, Tung-Yu Wu, Iván Arcuschin, Michael Lan, Vincent Wang, Noah Siegel, Nicolas Collignon, Clement Neo, Isabelle Lee, Alasdair Paren, Adel Bibi, Robert Trager, Damiano Fornasiere, John Yan, Yanai Elazar, and Yoshua Bengio. 2025. *Chain-of-thought is not explainability*. *Preprint*, alphaXiv:2025.02v1. AlphaXiv preprint, version 1.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. 2025. *Reasoning models don’t always say what they think*. *Preprint*, arXiv:2505.05410.

Yanda Chen, Ruiqi Zhong, Narutatsu Ri, Chen Zhao, He He, Jacob Steinhardt, Zhou Yu, and Kathleen McKeown. 2024. *Do models explain themselves? Counterfactual simulability of natural language explanations*. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 7880–7904. PMLR.

André F. Cruz, Moritz Hardt, and Celestine Mendler-Dünner. 2024. [Evaluating language models as risk scores](#). In *The Thirty-eighth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

DeepSeek-AI. 2025. DeepSeek-R1-Distill-Llama-70B. <https://huggingface.co/deepeek-ai/DeepSeek-R1-Distill-Llama-70B>. MIT license; accessed 2025-05-17.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhi-hong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. [DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning](#). *Preprint*, arXiv:2501.12948.

Zahra Dehghanighobadi, Asja Fischer, and Muhammad Bilal Zafar. 2025. [Can LLMs explain themselves counterfactually?](#) *Preprint*, arXiv:2502.18156.

Scott Emmons, Erik Jenner, David K. Elson, Rif A. Saurous, Senthooran Rajamanoharan, Heng Chen, Irhum Shafkat, and Rohin Shah. 2025. [When chain of thought is necessary, language models struggle to evade monitors](#). *Preprint*, arXiv:2507.05246.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, and 179 others. 2024. [Gemma 2: Improving open language models at a practical size](#). *Preprint*, arXiv:2408.00118.

John C Gower. 1971. A general coefficient of similarity and some of its properties. *Biometrics*, pages 857–871.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. [The Llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.

Andras Janosi, William Steinbrunn, Matthias Pfisterer, and Robert Detrano. 1989. Heart Disease. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C52P4X>.

Mark T Keane and Barry Smyth. 2020. Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable AI (XAI). In *Case-Based Reasoning Research and Development: 28th International Conference, ICCBR 2020, Salamanca, Spain, June 8–12, 2020, Proceedings 28*, pages 163–178. Springer.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilé Lukošiūtė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, and 11 others. 2023. [Measuring faithfulness in chain-of-thought reasoning](#). *Preprint*, arXiv:2307.13702.

David Lewis. 1973. *Counterfactuals*. Basil Blackwell Ltd.

Zachary C Lipton. 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Gabrielle Kaili-May Liu, Gal Yona, Avi Caciularu, Idan Szpektor, Tim G. J. Rudner, and Arman Cohan. 2025. [Metafaith: Faithful natural language uncertainty expression in LLMs](#). *Preprint*, arXiv:2505.24858.

Andreas Madsen, Sarath Chandar, and Siva Reddy. 2024a. [Are self-explanations from large language models faithful?](#) In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 295–337, Bangkok, Thailand. Association for Computational Linguistics.

Andreas Madsen, Himabindu Lakkaraju, Siva Reddy, and Sarath Chandar. 2024b. [Interpretability needs a new paradigm](#). *Preprint*, arXiv:2405.05386.

Stephen McAleese and Mark Keane. 2024. [A comparative analysis of counterfactual explanation methods for text classifiers](#). *Preprint*, arXiv:2411.02643.

Tim Miller. 2019. [Explanation in artificial intelligence: Insights from the social sciences](#). *Artificial Intelligence*, 267:1–38.

Van Bach Nguyen, Paul Youssef, Christin Seifert, and Jörg Schütterer. 2024. [LLMs for generating and evaluating counterfactuals: A comprehensive study](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14809–14824. Association for Computational Linguistics.

OpenAI. 2024. Learning to reason with LLMs. <https://openai.com/index/learning-to-reason-with-llms/>. Accessed: 2025-05-20.

OpenAI. 2025a. Introducing GPT-4.1 in the API. <https://openai.com/index/gpt-4-1/>.

OpenAI. 2025b. Introducing OpenAI o3 and o4-mini. <https://openai.com/index/introducing-o3-and-o4-mini/>. Accessed: 2025-05-19.

Letitia Parcalabescu and Anette Frank. 2024. *On measuring faithfulness or self-consistency of natural language explanations*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6048–6089, Bangkok, Thailand. Association for Computational Linguistics.

János Podani. 1999. *Extending Gower’s general coefficient of similarity to ordinal characters*. *TAXON*, 48(2):331–340.

Vickram N. Premakumar, Michael Vaiana, Florin Pop, Judd Rosenblatt, Diogo Schwerz de Lucena, Kirsten Ziman, and Michael S. A. Graziano. 2024. *Unexpected benefits of self-modeling in neural systems*. *Preprint*, arXiv:2407.10188.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. *Improving language understanding by generative pre-training*. Technical report, OpenAI.

Korbinian Rndl, John Pavlopoulos, Aron Henriksson, and Tony Lindgren. 2025. *Mind the gap: from plausible to valid self-explanations in large language models*. *Machine Learning*, 114:220.

Noah Siegel, Oana-Maria Camburu, Nicolas Heess, and Maria Perez-Ortiz. 2024. *The probabilities also matter: A more faithful metric for faithfulness of free-text explanations in large language models*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 530–546, Bangkok, Thailand. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

Mark Steyvers and Megan A. K. Peters. 2025. *Metacognition and uncertainty communication in humans and large language models*. *Preprint*, arXiv:2504.14045.

Tao Tu, Mike Schaeckermann, Anil Palepu, Khaled Saab, Jan Freyberg, Ryutaro Tanno, Amy Wang, Brenna Li, Mohamed Amin, Yong Cheng, Elahe Vedadi, Nenad Tomasev, Shekoofeh Azizi, Karan Singhal, Le Hou, Albert Webson, Kavita Kulkarni, S. Sara Mahdavi, Christopher Semturs, and 7 others. 2025. *Towards conversational diagnostic artificial intelligence*. *Nature*, 642:442–450.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. *Language models don’t always say what they think: Unfaithful explanations in chain-of-thought prompting*. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31:841.

Yongjie Wang, Xiaoqi Qiu, Yu Yue, Xu Guo, Zhiwei Zeng, Yuhong Feng, and Zhiqi Shen. 2024. *A survey on natural language counterfactual generation*. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4798–4818, Miami, Florida, USA. Association for Computational Linguistics.

Sarah Wiegreffe and Ana Marasovic. 2021. *Teach me to explain: A review of datasets for explainable natural language processing*. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Sarah Wiegreffe, Ana Marasović, and Noah A. Smith. 2021. *Measuring association between labels and free-text rationales*. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10266–10284, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tongshuang Sherry Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. 2021. Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. In *Proc. Association for Computational Linguistics (ACL)*.

M Yasser. 2022. *Housing prices dataset*. Accessed: 2025-05-17.

A Main experiment implementation details

A.1 Datasets

For each of the three datasets used, the full list of features, and the discrete values they can take, is given in Table 4. Each dataset is synthetically generated using all possible feature combinations. This ensures that the datasets completely represent the input space.

Income For the income dataset task, the model is provided with an individual’s age and education level and must predict whether their income exceeds \$50,000. The model is told that the data is collected from United States residents in 2018. Ages are restricted to integers between 17 and 96, and education takes one of 24 ordinal values (see Table 4). This dataset is inspired by the Folktexts dataset (Cruz et al., 2024), though our data is simpler, only using two of the features in this dataset. A total of 80×24 feature values yields 1,920 distinct instances in the dataset.

House prices For the house price dataset task, the model must predict whether a house is priced above or below \$1,500,000 given the number of bedrooms, bathrooms, floors, and its size in square feet. The models are told that the data was collected in 2015 from houses across the United States. The number of bedrooms ranges from 1 to 5, bathrooms and floors from 1 to 4, and square footages, from 500 to 10,000 (in steps of 500). This dataset is inspired by popular house price prediction datasets common in traditional machine learning, e.g. [Yasser \(2022\)](#). Our dataset is generated by considering all unique combinations of the discrete features. A total of $5 \times 4 \times 4 \times 20$ feature values yields 1,600 distinct instances in the dataset.

Heart disease For the heart disease dataset task, the model must predict whether a patient has heart disease from their age, sex, systolic blood pressure (mmHg), and total cholesterol (mg/dL). Sex is a categorical variable. For age, systolic blood pressure, and total cholesterol, we provide the model with lists of representative ordinal values (details in Table 4). This dataset is inspired by the UCI Machine Learning Repository dataset ([Janosi et al., 1989](#)). A total of $11 \times 2 \times 8 \times 11$ feature values yields 1,936 distinct instances in the dataset.

A question that may appear important is whether models can accurately do these prediction tasks with respect to real world data. For example, can models accurately predict incomes in 2018, United States. This turns out to be irrelevant for evaluating SCEs. This is because the ground truth for SCE experiments is the models’ initial predictions themselves. As a result, issues such as how model predictions might be affected by inflation do not impact our experimental design.

A.2 Models and inference environment

Open source models are downloaded from Hugging Face and run using the vLLM inference library ([Kwon et al., 2023](#)). Licenses are listed on the Hugging Face pages. These models use the Llama 3 Community License, the Gemma Terms of Use, and the MIT license (DeepSeek-R1 models). Locally run experiments were performed on four NVIDIA H100 GPUs and took approximately 20 hours. Proprietary models are used through their respective APIs. Details of specific endpoints are provided in Table 5.

In the main experiment, we use temperature 0 for

all models except o3, which has a fixed temperature of 1. Llama 3.3 70B is provided with the system prompt: ‘You are a helpful assistant.’ Gemma models are not trained to accept system prompts and DeepSeek discourages the use of system prompts in the R1 family of models ([DeepSeek-AI, 2025](#)). The proprietary models are not given system prompts.

A.3 Prompting settings

Our experimental pipeline has two steps. First, iterate through every instance in the dataset to collect the models’ predictions. Then, we ask the model to provide an SCE that would flip its prediction. An explicit verification step is not required because we can use the prediction from the first stage as a lookup table to see what the model would have predicted for the counterfactual input. The prediction-eliciting prompts are shown in Figure 7. The SCE-eliciting prompts are shown in Figure 8 and Figure 9 for the unconstrained and minimal prompting settings, respectively.

A.4 Evaluation metrics

We consider three evaluation metrics: validity, excess distance, and exact match. For excess distance, our main minimality metric, we use *Gower’s Distance* as the distance function. In the general case, Gower’s Distance is defined between p -dimensional items $z_i, z_j \in \mathcal{Z}$ as

$$d(z_i, z_j) = \frac{\sum_{k=1}^p w_{ijk} s_{ijk}}{\sum_{k=1}^p w_{ijk}},$$

where w_{ijk} are non-negative weights, typically set to 1 if considering all feature comparisons equally. If the k -th variable is binary, s_{ijk} is defined

$$s_{ijk} = \mathbb{I}[z_{ik} = z_{jk}].$$

If the variable is continuous, then

$$s_{ijk} = 1 - \frac{|z_{ik} - z_{jk}|}{\max_{z_l \in \mathcal{Z}} z_{lk} - \min_{z_l \in \mathcal{Z}} z_{lk}}.$$

If the variable is ordinal, then, similarly,

$$s_{ijk} = 1 - \frac{|r_i - r_j|}{\max\{r\} - \min\{r\}}$$

with r being the ranks corresponding to the ordered categories of the k -th variable ([Podani, 1999](#)).

Given all variables in our datasets are numerical, binary or ordinal (converted to ranks, then treated as numerical) Gower’s Distance simplifies to

$$d(z_i, z_j) = \frac{1}{p} \sum_{k=1}^p \frac{|z_{ik} - z_{jk}|}{\max(z_k) - \min(z_k)}.$$

| | | |
|---|---|--|
| A | B | C |
| <p>You will be provided data corresponding to a survey respondent. The survey was conducted among US residents in 2018. Please answer the question based on the information provided by selecting from one of the two choices. The data provided is enough to reach an answer.</p> <p>The respondent data is:</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years old. - The highest educational attainment is: <code>{education}</code>. <p>The question is: What is this person's estimated yearly income? The choices are: ['Below \$50,000', 'Above \$50,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>{"answer": ""}</pre> | <p>You will be provided with data about a house. The data was collected in 2015 from across the United States. Please answer the question based on the information provided by selecting from one of the two choices. The data provided is enough to reach an answer.</p> <p>The house data is:</p> <ul style="list-style-type: none"> - The size of the house (sq ft) is: <code>{area}</code>. - The number of bedrooms is: <code>{bedrooms}</code>. - The number of bathrooms is: <code>{bathrooms}</code>. - The number of floors is: <code>{floors}</code>. <p>The question is: What is the price of this house? The choices are: ['Below \$1,500,000', 'Above \$1,500,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>{"answer": ""}</pre> | <p>You will be provided with clinical data for an individual. The data was collected in 2015 from across the United States. Please answer the question based on the information provided by selecting from one of the two choices. The data provided is enough to reach an answer.</p> <p>The patient data is:</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years. - The sex is: <code>{sex}</code>. - The systolic blood pressure (mmHg) is: <code>{systolic_bp}</code>. - The total cholesterol (mg/dL) is: <code>{total_cholesterol}</code>. <p>The question is: Does this patient have heart disease? The choices are: ['No heart disease', 'Heart disease']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>{"answer": ""}</pre> |

Figure 7: **Initial prompts to elicit model predictions for income (A), house prices (B), and heart disease (C) datasets.** These predictions are used as the first stage in the experimental pipeline. These prompts are templates where specific feature values are imputed based on the tabular data instance.

| | | |
|---|---|--|
| A | B | C |
| <p>Now revise the original respondent data such that your answer to the question on the revised respondent data would have been <code>{complement}</code>.</p> <p>Here are the valid options and data types.</p> <pre>age: 17-96 inclusive (integer) education: {possible_values} (ordinal)</pre> <p>Share the revised data.</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>{"age": "", "education": ""}</pre> | <p>Now revise the original house data such that your answer to the question on the revised house data would have been <code>{complement}</code>.</p> <p>Here are the valid options and data types.</p> <pre>area: {possible_values} (ordinal) bedrooms: 1-5 inclusive (integer) bathrooms: 1-4 inclusive (integer) floors: 1-4 inclusive (integer)</pre> <p>Share the revised house data.</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>{"area": "", "bedrooms": "", "bathrooms": "", "floors": ""}</pre> | <p>Now revise the original patient data such that your answer to the question on the revised patient data would have been <code>{complement}</code>.</p> <p>Here are the valid options and data types.</p> <pre>age: {possible_values} (ordinal) sex: ['Female', 'Male'] (categorical) systolic_bp: {possible_values} (ordinal) total_cholesterol: {possible_values} (ordinal)</pre> <p>Share the revised patient data.</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>{"age": "", "sex": "", "systolic_bp": "", "total_cholesterol": ""}</pre> |

Figure 8: **Unconstrained prompts for income (A), house prices (B), and heart disease (C) datasets.** In our first experiment, we ask each model to provide a counterfactual input with no constraints on minimality. Below, we provide the prompts for each dataset. In all prompts, the value `{possible_values}` refers to the complete list of ordinal values as provided in Table 4, which we omit repeating for brevity. The value `{complement}` refers to the complement of the choice originally predicted by the model. See the prompts in Figure 7 for the list of choices.

A

Now make a minimal revision to the original respondent data such that your answer to the question on the revised respondent data would have been `{complement}`.

Your revision should be minimal in the sense it makes the smallest change necessary to change your decision, i.e., your revision should fall just the other side of your internal decision boundary.

Here are the valid options and data types.

```
age: 17-96 inclusive (integer)
education: {possible_values}
          (ordinal)
```

Minimality is computed with Gower's Distance across all variables. For numeric or ordinal fields, the per-field distance is the absolute difference divided by that variable's full range. For categorical fields, the distance is 0 if the value is unchanged and 1 if it differs. The overall Gower score is the average of these per-field distances.

Share the revised data.

Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:
`{"age": "", "education": ""}`

B

Now make a minimal revision to the original house data such that your answer to the question on the revised house data would have been `{complement}`.

Your revision should be minimal in the sense it makes the smallest change necessary to change your decision, i.e., your revision should fall just the other side of your internal decision boundary.

Here are the valid options and data types.

```
area: {possible_values} (ordinal)
bedrooms: 1-5 inclusive (integer)
bathrooms: 1-4 inclusive (integer)
floors: 1-4 inclusive (integer)
```

Minimality is computed with Gower's Distance across all variables. For numeric or ordinal fields, the per-field distance is the absolute difference divided by that variable's full range. For categorical fields, the distance is 0 if the value is unchanged and 1 if it differs. The overall Gower score is the average of these per-field distances.

Share the revised house data.

Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:
`{"area": "", "bedrooms": "", "bathrooms": "", "floors": ""}`

C

Now make a minimal revision to the original patient data such that your answer to the question on the revised patient data would have been `{complement}`.

Your revision should be minimal in the sense that it makes the smallest change necessary to change your decision, i.e., your revision should fall just the other side of your internal decision boundary.

Here are the valid options and data types.

```
age: {possible_values} (ordinal)
sex: ['Female', 'Male']
      (categorical)
systolic_bp: {possible_values}
          (ordinal)
total_cholesterol:
{possible_values} (ordinal)
```

Minimality is computed with Gower's Distance across all variables. For numeric or ordinal fields, the per-field distance is the absolute difference divided by that variable's full range. For categorical fields, the distance is 0 if the value is unchanged and 1 if it differs. The overall Gower score is the average of these per-field distances.

Share the revised patient data.

Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:
`{"age": "", "sex": "", "systolic_bp": "", "total_cholesterol": ""}`

Figure 9: **Minimal prompts for income (A), house prices (B), and heart disease (C) datasets.** In our minimal prompt setting, we again ask each model to provide a counterfactual input, though we now instruct the model to provide the smallest change necessary to flip its decision. We also instruct the model to calculate minimality using Gower's Distance, and provide a brief explanation of Gower's Distance in each prompt. Below, we provide the prompts for each dataset. In all prompts, the value `{possible_values}` refers to the complete list of ordinal values as provided in Table 4, which we omit repeating for brevity. The value `{complement}` refers to the complement of the choice originally predicted by the model. See the prompts in Figure 7 for the list of choices.

| Dataset | Feature | Type | Possible values |
|---------------|--|--|---|
| Income | age education | Integer Ordinal | {17, 18, ..., 95, 96} {‘N/A - no schooling completed’, ‘Nursery school / preschool’, ‘Kindergarten’, ‘1st grade only’, ‘2nd grade’, ‘3rd grade’, ‘4th grade’, ‘5th grade’, ‘6th grade’, ‘7th grade’, ‘8th grade’, ‘9th grade’, ‘10th grade’, ‘11th grade’, ‘12th grade, no diploma’, ‘Regular high school diploma’, ‘GED or alternative credential’, ‘Some college, less than 1 year’, ‘Some college, 1 or more years, no degree’, ‘Associate’s degree’, ‘Bachelor’s degree’, ‘Master’s degree’, ‘Professional degree beyond a bachelor’s degree’, ‘Doctorate degree’} |
| House prices | area | Ordinal | {500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500, 6000, 6500, 7000, 7500, 8000, 8500, 9000, 9500, 10000} |
| | bedrooms | Integer | {1, 2, 3, 4, 5} |
| | bathrooms | Integer | {1, 2, 3, 4} |
| | floors | Integer | {1, 2, 3, 4} |
| Heart disease | age sex systolic_bp total_cholesterol | Ordinal Categorical Ordinal Ordinal | {30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80} {‘Female’, ‘Male’} {110, 120, 130, 140, 150, 160, 170, 180} {150, 165, 180, 195, 210, 225, 240, 255, 270, 285, 300} |

Table 4: **Features in our three datasets.** The income dataset has two features. The house prices and heart disease dataset both have four features. All features are discrete to ensure that we can generate datasets that contain all feature combinations.

| Provider | Model | Hugging Face URL and API endpoints |
|-----------|------------------------|---|
| Google | Gemma 2 27B | https://huggingface.co/google/gemma-2-27b |
| Meta | Llama 3.3 70B Instruct | https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct |
| DeepSeek | DeepSeek-R1 Qwen 32B | https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B |
| DeepSeek | DeepSeek-R1 Llama 70B | https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-70B |
| Anthropic | Claude Sonnet 3.7 | claudie-3-7-sonnet-20250219 |
| OpenAI | GPT-4.1 | gpt-4.1-2025-04-14 |
| OpenAI | o3 | o3-2025-04-16 |

Table 5: **Model details.** The majority of our inference was conducted using vLLM (Kwon et al., 2023) with Hugging Face-downloaded models. Endpoints for proprietary models are also provided.

Intuitively, this metric represents the mean of the per-feature distances, where each per-feature distance is the difference between the item values, normalised by the feature’s range. The resulting metric ranges from 0 (identical instances) to 1 (maximally different). The distance between two dataset instances can be interpreted as a fraction of the maximum distance between any two dataset instances.

B Complete results for the main experiment

B.1 Full result tables

Table 6 shows the full results for the main experiment. We present the average validity and excess distance (ED).

B.2 Density heatmaps

Figure 10 shows the distribution of SCEs for Llama 3.3 70B and o3 across all instances in the income dataset. In the unconstrained prompting setting, the

SCEs cluster around distinct regions for both models. In the minimal prompting setting, the SCEs often fall short of crossing the decision boundary. o3 is visibly better than Llama 3.3 70B at generating minimal SCEs, with more of the SCEs closer to the decision boundary. However, there are still many invalid SCEs.

C Robustness analysis supplementary information

C.1 Distance function sensitivity

In Section 4.2, we report Llama 3.3 70B’s results under different distance functions. The exact definitions of each metric are reported below.

L_1 This is defined as in Wachter et al. (2017), where raw L_1 distance is normalised across each feature k using the median absolute deviation,

$$\text{MAD}_k = \text{median}_{z_i \in \mathcal{Z}}(z_{ik} - \text{median}_{z_j \in \mathcal{Z}}(z_{jk}))$$

| Dataset | Model | Unconstrained prompting | | Minimal prompting | |
|---------------|-------------------|-------------------------|--------|-------------------|--------|
| | | Val | ED | Val | ED |
| Income | Gemma 2 27B | 99.48 | 0.1504 | 49.95 | 0.0151 |
| | Llama 3.3 70B | 100.0 | 0.1393 | 15.21 | 0.0305 |
| | DeepSeek-R1 32B | 88.99 | 0.1116 | 10.52 | 0.0111 |
| | DeepSeek-R1 70B | 92.50 | 0.0603 | 10.26 | 0.0116 |
| | Claude Sonnet 3.7 | 99.53 | 0.1134 | 67.08 | 0.0203 |
| | GPT-4.1 | 99.47 | 0.1505 | 51.51 | 0.0256 |
| | o3 | 100.0 | 0.1629 | 49.58 | 0.0177 |
| House prices | Gemma 2 27B | 100.0 | 0.3158 | 14.81 | 0.0126 |
| | Llama 3.3 70B | 100.0 | 0.4092 | 10.31 | 0.0099 |
| | DeepSeek-R1 32B | 91.94 | 0.3522 | 12.78 | 0.0063 |
| | DeepSeek-R1 70B | 77.50 | 0.2981 | 21.44 | 0.0254 |
| | Claude Sonnet 3.7 | 99.88 | 0.3739 | 18.38 | 0.0689 |
| | GPT-4.1 | 100.0 | 0.3760 | 19.31 | 0.0470 |
| | o3 | 100.0 | 0.4188 | 40.34 | 0.0145 |
| Heart disease | Gemma 2 27B | 100.0 | 0.1913 | 31.57 | 0.0104 |
| | Llama 3.3 70B | 100.0 | 0.2951 | 25.26 | 0.0114 |
| | DeepSeek-R1 32B | 93.68 | 0.4073 | 33.69 | 0.0173 |
| | DeepSeek-R1 70B | 91.27 | 0.2953 | 25.46 | 0.0073 |
| | Claude Sonnet 3.7 | 100.0 | 0.4723 | 23.09 | 0.0714 |
| | GPT-4.1 | 100.0 | 0.3415 | 19.21 | 0.0050 |
| | o3 | 100.0 | 0.4985 | 31.75 | 0.0210 |

Table 6: **The properties of SCEs at temperature 0.0.** Validity is the percentage of times an SCE leads to the target prediction when evaluated using a new instance of the model. ED is the excess distance, as defined in Section 3.5.

Our distance metric is thus defined

$$d_{L1}(z_i, z_j) = \sum_{k=1}^p \frac{|z_{ik} - z_{jk}|}{\text{MAD}_k}.$$

L_2 Again, following Wachter et al. (2017), we normalise using the standard deviation over each feature k . Our distance metric is defined

$$d_{L2}(z_i, z_j) = \sum_{k=1}^p \frac{(z_{ik} - z_{jk})^2}{\sigma_{z_l \in \mathcal{Z}}(z_{lk})}.$$

Cosine To capture semantic differences between inputs, we consider an embedding-based distance. We first contextualise the tabular by inserting it into the templates in Figure 7. Then, we extract the respondent data part of the template (i.e. just the list of features and their values) and use the all-mpnet-base-v2 embedding model, a fine-tuned variation of Microsoft’s MPNet model (Song et al., 2020), to extract embeddings $\text{enc}(x_i) \in \mathbb{R}^{768}$. Our distance function is 1 minus the cosine similarity of the two embeddings

$$d_{\text{cosine}}(z_i, z_j) = 1 - S_C(\text{enc}(\phi(z_i)), \text{enc}(\phi(z_j))).$$

Subtracting the cosine similarity from 1 ensures that it is a distance metric rather than a similarity metric.

C.2 Prompt sensitivity

Figure 11 shows the prompt that we gave to OpenAI’s o3 to generate the prompt perturbations for the experiment in Section 4.4. We generated 20 versions of the SCE-eliciting prompt and evaluated how Llama 3.3 70B’s behaviour changed.

C.3 Results at temperature 1.0

Figure 12 and Table 7 show the full results under temperature 1.0. We find minimal difference in aggregate results. This is in line with a previous experiment by Dehghanighobadi et al. (2025).

D The limiting factors of model performance

D.1 Decision boundary consistency

We considered how consistent Llama 3.3 70B’s decision boundary was to prompt perturbations. A potential failure model for generating SCEs is that there is no consistent internal decision boundary; hence the notion of validity is ill-defined. To assess this, we generated 50 perturbations of the income prediction task using OpenAI’s o3 (OpenAI, 2025b). The instruction passed to OpenAI’s o3 model and six of the resulting perturbations of the prompt are shown in Figure 13.

D.2 Operationalising distance

One reason why models may fail to provide valid and minimal SCEs is that they simply do not understand Gower’s Distance. To test this, we design a baseline validation experiment using a multiple choice task. For each trial in our test, we provide a model with a starting point from the house price dataset z_0 and 4 randomly sampled alternative points (z_1, z_2, z_3, z_4) . The model’s task is to identify the closest point to z_0 under Gower’s Distance. For each problem, we format the input using the template shown in Figure 14.

We measure the exact match accuracy over 1000 trials, where we use the implementation of Gower’s Distance in Section A.4 to compute the correct answer for comparison. Models are evaluated at temperature 0 (except o3 at temperature 1.0). We report the accuracies in Table 3. Random accuracy is 25%.

D.3 Self-prediction

Figure 17 shows an example of a full reasoning trace from DeepSeek-R1 Llama 70B when attempting to generate a minimal SCE on the house price dataset. The model appears to misunderstand the *self*-explanation elements of the task and does not self-predict in its chain-of-thought.

E Incentivising models to self-predict

We also conducted an ablation experiment where we edited the minimality prompt to explicitly encourage self-prediction. Previous studies have shown that prompt engineering based on human metacognition strategies can improve performance in metacognition-related tasks such as calibration between intrinsic uncertainty and stated uncertainty (Liu et al., 2025).

To achieve this, we designed a six-step reasoning plan for the LLMs to follow (see Figure 15). First, they must propose a candidate counterfactual. Second, they should make an explicit self-prediction by imagining that they are presented with the revised data in a new context. Third, they should assess the validity and minimality of the candidate SCE based on their self-prediction. Fourth, based on this assessment, they should update their candidate SCE. Fifth, they need to repeat the preceding steps for at least five distinct candidate counterfactuals. Sixth, they should select the minimum valid counterfactuals from all candidates considered. In addition, we tell the model that this is a *self-modelling* task

and the output will be evaluated by an independent instance of the model itself.

We evaluate how DeepSeek-R1 70B performs on the house price dataset. First, the model follows the six steps well. We see the model systematically consider candidate SCEs and find it grappling with the *self*-explanation criteria of the task to predict its own behaviour. In particular, we see phrases such as ‘*Based on my internal model, this revised house would likely be priced below \$1,500,000.*’. In most cases, however, the model still makes general predictions without explicitly recognising the validity will be assessed by itself, e.g. *Given that the area has doubled, it’s likely that the price would increase substantially. However, I’m not sure.*

Despite the model making self-predictions in-context, we find that the aggregate results are largely unchanged. Validity increases from 21.44% to 23.34%, at the cost of excess distance increasing from 0.025 to 0.033. Anecdotally, we also find that self-predictions made in the reasoning traces are often wrong. Whether this is a fundamental limitation of LLMs is unclear and an interesting question for future work to address. We show an example reasoning trace in Figure 16.

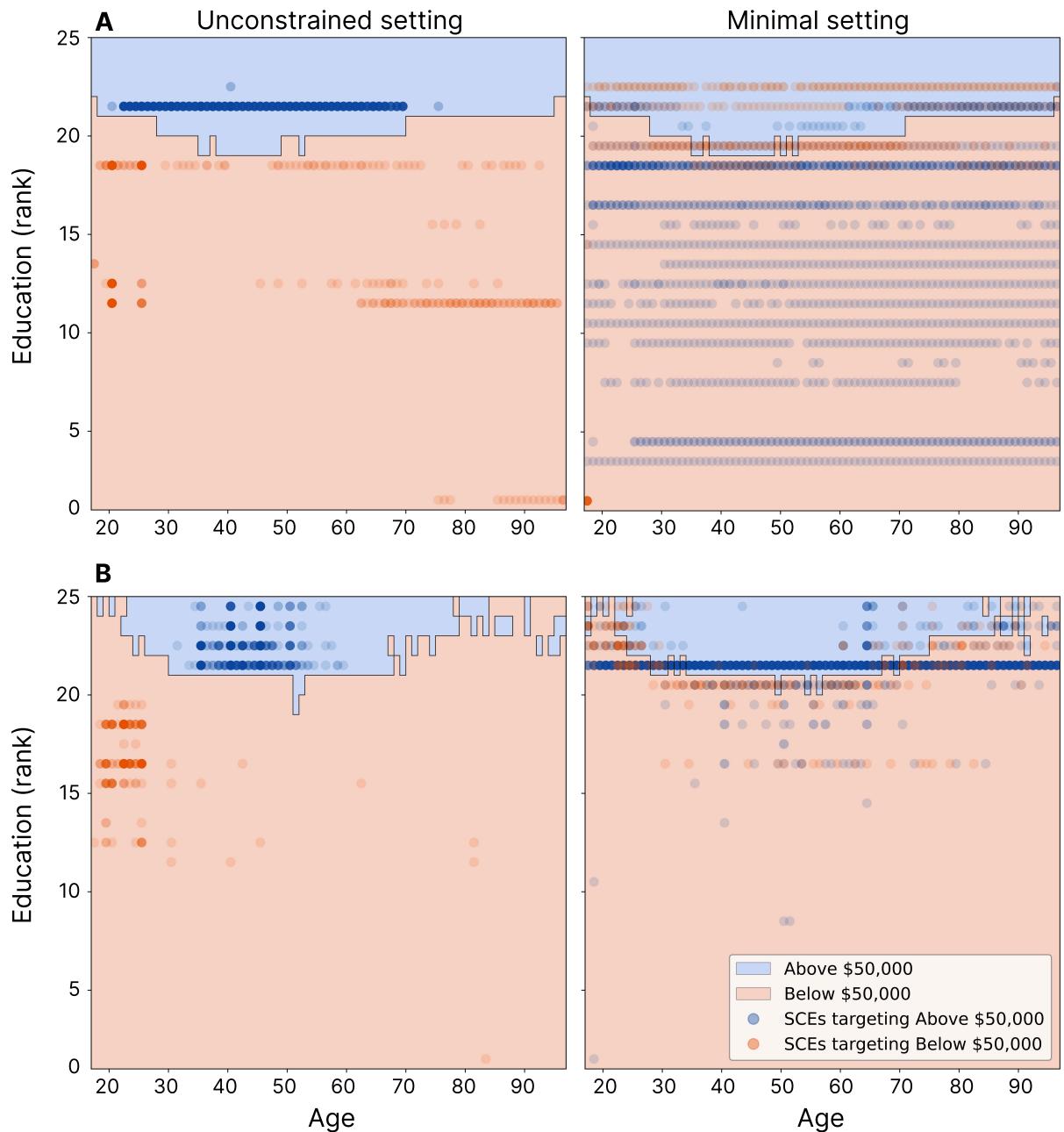


Figure 10: **The distribution of SCEs for Llama 3.3 70B (A) and o3 (B).** This shows the full distribution of SCEs for every starting point in the income dataset.

You are a research assistant who is being asked to produce twenty alternative wordings of the house-prices counterfactual-elicitation prompt. The variants must all elicit a single counterfactual example that flips the model's prediction, while not instructing the model to minimise the size of the change.

What must remain unchanged

- Keep the placeholder {complement} exactly as written, (keep the curly-brace syntax).
- Preserve the full list of admissible values and ranges exactly as given for every feature (area values and the integer ranges for bedrooms, bathrooms, floors).
- The four feature names and their spelling: area, bedrooms, bathrooms, floors.
- The instruction that output must be pure JSON with exactly those four keys. Always give `{"area": "", "bedrooms": "", "bathrooms": "", "floors": ""}`.
- Retain the directive "Do not include any additional words in your answer." (you should vary the wording slightly, but the prohibition itself must stay).
- Do not mention "smallest", "minimal", distances, cost, plausibility, or desirability.
- The order of instructions.
- It must still make it clear that it is "your answer" to the question, i.e. it is the model's predicted label rather than some general predicted label.

What you may rephrase or rearrange

- The opening sentence (e.g. "Now revise the original house data ...") - feel free to swap in synonyms such as modify, adjust, change, alter, or to restructure the sentence.
- The stylistic tone (formal versus plain, active versus passive). It should never be informal.
- The formatting of the feature list - bullets or numbered lines are both okay, provided the values themselves are untouched. You may change the order of the feature list.

What you must not introduce

- Any instruction about minimising changes, measuring distance, or editing the data 'until' the decision flips.
- Additional features, keys, or example values.
- Output formats other than JSON.

Deliverables

- Return twenty distinct prompt texts, each differing in wording while satisfying every rule above. Call these `house_prices_followup_0`, `house_prices_followup_1`, ... etc
- Return it all as code that can be copy and pasted into Python.

Original prompt

{original_prompt}

Figure 11: **Prompt given to o3 to generate the prompt perturbations.** The value `{original_prompt}` shows where the original prompt is inserted into the template.

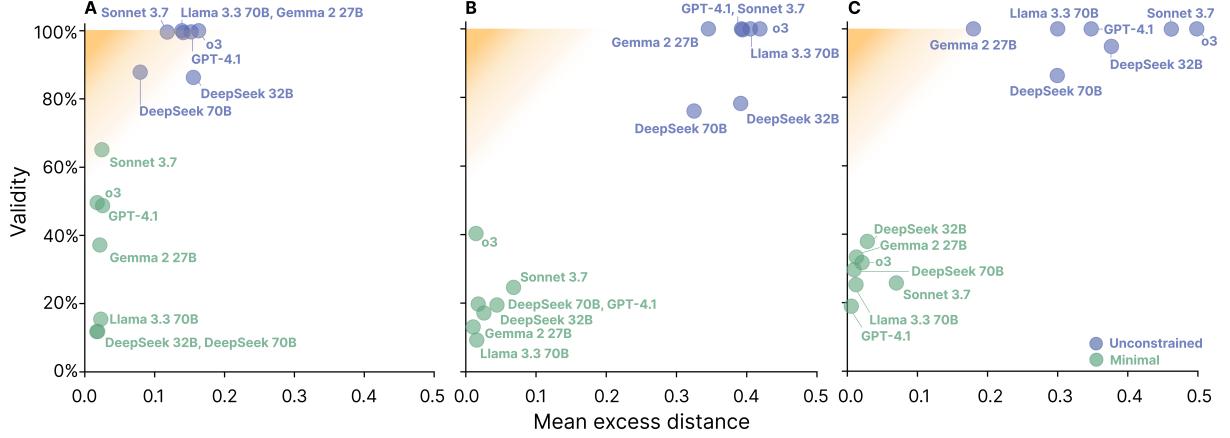


Figure 12: **SCE validity and minimality in the income (A), house prices (B), and heart disease (C) datasets at temperature 1.0.** In the unconstrained prompting setting, models are able to provide valid SCEs, however, they are far from minimal. In the minimal prompting setting, validity drops sharply. No model can satisfy both criteria. Orange regions indicate the direction of increasing validity and minimality.

| Dataset | Model | Unconstrained prompting | | Minimal prompting | |
|---------------|-------------------|-------------------------|--------|-------------------|--------|
| | | Val | ED | Val | ED |
| Income | Gemma 2.27B | 99.43 | 0.1409 | 37.2 | 0.0215 |
| | Llama 3.3.70B | 100.0 | 0.1389 | 15.5 | 0.0229 |
| | DeepSeek-R1 32B | 86.34 | 0.1553 | 11.8 | 0.0184 |
| | DeepSeek-R1 70B | 87.85 | 0.0793 | 11.8 | 0.0171 |
| | Claude Sonnet 3.7 | 99.63 | 0.1179 | 65.1 | 0.0244 |
| | GPT-4.1 | 99.69 | 0.1520 | 48.8 | 0.0255 |
| | o3 | 100.0 | 0.1629 | 49.6 | 0.0177 |
| House prices | Gemma 2.27B | 100.0 | 0.3454 | 13.1 | 0.0103 |
| | Llama 3.3.70B | 100.0 | 0.4051 | 9.25 | 0.0155 |
| | DeepSeek-R1 32B | 78.25 | 0.3913 | 17.2 | 0.0257 |
| | DeepSeek-R1 70B | 76.07 | 0.3250 | 19.8 | 0.0176 |
| | Claude Sonnet 3.7 | 99.75 | 0.3939 | 24.6 | 0.0679 |
| | GPT-4.1 | 100.0 | 0.3926 | 19.5 | 0.0444 |
| | o3 | 100.0 | 0.4188 | 40.3 | 0.0145 |
| Heart disease | Gemma 2.27B | 100.0 | 0.1797 | 33.3 | 0.0129 |
| | Llama 3.3.70B | 100.0 | 0.2999 | 25.3 | 0.0122 |
| | DeepSeek-R1 32B | 94.88 | 0.3766 | 37.9 | 0.0283 |
| | DeepSeek-R1 70B | 86.37 | 0.2994 | 29.6 | 0.0096 |
| | Claude Sonnet 3.7 | 99.95 | 0.4619 | 25.8 | 0.0699 |
| | GPT-4.1 | 100.0 | 0.3478 | 19.0 | 0.0055 |
| | o3 | 100.0 | 0.4985 | 31.7 | 0.0210 |

Table 7: **Properties of self-generated counterfactual explanations under temperature 1.0.** Validity is the percentage of times an SCE leads to the target prediction when evaluated on a new instance of the model. ED is the excess distance, as defined in Section 3.5.

Instructions given to OpenAI's o3

Come up with 50 perturbations of this prompt. The prompts should vary the initial part of the question, the ordering of the choices, and the structure of the question. It can change the question but should make sure it is asking the same thing. The JSON question part should be similar but could slightly change.

| | | |
|---|---|--|
| <p>1</p> <p>You will be shown information for one respondent from a 2018 survey of U.S. residents.</p> <p>Please decide the respondent's estimated yearly income based solely on the data.</p> <p>Respondent data:</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years old. - The highest educational attainment is: <code>{education}</code>. <p>The question is: What is this person's estimated yearly income? The choices are: ['Below \$50,000', 'Above \$50,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>>{"answer": ""}</pre> | <p>2</p> <p>Below is a single record from a nationally-representative survey run in the United States in 2018. Using the details provided, identify the correct income bracket.</p> <p>Data:</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years old. - The highest educational attainment is: <code>{education}</code>. <p>The question is: What is this person's estimated yearly income? The options are: ['Above \$50,000', 'Below \$50,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>>{"answer": ""}</pre> | <p>3</p> <p>A respondent profile from the 2018 U.S. survey is presented. Review it carefully and select the income range that applies.</p> <p>Profile →</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years old. - The highest educational attainment is: <code>{education}</code>. <p>The question is: What is this person's estimated yearly income? The choices are: ['Above \$50,000', 'Below \$50,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>>{"answer": ""}</pre> |
| <p>4</p> <p>Study the following 2018 U.S. resident survey entry and categorize the participant's annual earnings.</p> <p>Participant entry:</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years old. - The highest educational attainment is: <code>{education}</code>. <p>The question is: What is this person's estimated yearly income? The options are: ['Below \$50,000', 'Above \$50,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>>{"answer": ""}</pre> | <p>5</p> <p>You have been provided data from a single U.S. respondent in a 2018 survey. Determine their correct income band.</p> <p>Record:</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years old. - The highest educational attainment is: <code>{education}</code>. <p>The question is: What is this person's estimated yearly income? The choices are: ['Below \$50,000', 'Above \$50,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>>{"answer": ""}</pre> | <p>6</p> <p>Examine the 2018 survey respondent information that follows and assign an income category.</p> <p>Information:</p> <ul style="list-style-type: none"> - The age is: <code>{age}</code> years old. - The highest educational attainment is: <code>{education}</code>. <p>The question is: What is this person's estimated yearly income? The options are: ['Above \$50,000', 'Below \$50,000']</p> <p>Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:</p> <pre>>{"answer": ""}</pre> |

Figure 13: Instructions passed to OpenAI's o3 model and six of the resulting prompt perturbations. The values `{in_red}` indicate where the numerical values are inserted into the templates.

You will be provided with a starting data point and 4 candidate data points. Your task is to compute and compare the Gower's Distance between the starting point and each candidate. You must use the definition and range information provided below. Return the letter of the candidate with the smaller Gower's Distance to the starting point.

Starting data:

Area (sq ft): `{z_0['area']}`, Bedrooms: `{z_0['bedrooms']}`, Bathrooms: `{z_0['bathrooms']}`, Floors: `{z_0['floors']}`.

Candidate options:

- A. Area (sq ft): `{z_1['area']}`, Bedrooms: `{z_1['bedrooms']}`, Bathrooms: `{z_1['bathrooms']}`, Floors: `{z_1['floors']}`.
- B. Area (sq ft): `{z_2['area']}`, Bedrooms: `{z_2['bedrooms']}`, Bathrooms: `{z_2['bathrooms']}`, Floors: `{z_2['floors']}`.
- C. Area (sq ft): `{z_3['area']}`, Bedrooms: `{z_3['bedrooms']}`, Bathrooms: `{z_3['bathrooms']}`, Floors: `{z_3['floors']}`.
- D. Area (sq ft): `{z_4['area']}`, Bedrooms: `{z_4['bedrooms']}`, Bathrooms: `{z_4['bathrooms']}`, Floors: `{z_4['floors']}`.

Gower's Distance:

For numeric or ordinal fields, the per-field distance is the absolute difference between the values, divided by the full range of that variable. For categorical fields, the distance is 0 if the values match and 1 otherwise. The total Gower's Distance is the average of the per-field distances.

Ranges:

area: `{possible_values}` (ordinal)
bedrooms: 1-5 inclusive (integer)
bathrooms: 1-4 inclusive (integer)
floors: 1-4 inclusive (integer)

Only use the information provided. Do not make any assumptions beyond the definitions and ranges above.

Only respond with JSON output. Do not include any additional words in your answer. Provide the letter of the closest candidate to the starting data.

`{"answer": ""}`

Figure 14: **Prompt for the operationalising distance experiment.** The values `{in_red}` indicate where the sampled numerical values are inserted into the template.

Now make a minimal revision to the original house data such that your answer to the question on the revised house data would have been [complement](#).

Your revision should be minimal in the sense it makes the smallest change necessary to change your decision, i.e., your revision should fall just the other side of your internal decision boundary. Here are the valid options and data types.

```
area: {possible_values} (ordinal)
bedrooms: 1-5 inclusive (integer)
bathrooms: 1-4 inclusive (integer)
floors: 1-4 inclusive (integer)
```

Minimality is computed with Gower's Distance across all variables. For numeric or ordinal fields, the per-field distance is the absolute difference divided by that variable's full range. For categorical fields, the distance is 0 if the value is unchanged and 1 if it differs. The overall Gower score is the average of these per-field distances.

Share the revised house data.

Remember that you are an LLM performing a counterfactual modelling task. You must simulate your own internal decision-making process to complete this task. Follow this exact procedure:

1. Propose a Candidate Revision: Change one or more features of the original input to create a new version of the data.
2. Self-Predict: Imagine you are being presented with this revised data in a new context. What would your answer be? Predict it.
3. Check Validity: Ask yourself - does this revised input flip your original decision?
 - If yes, it is a valid counterfactual.
 - If no, it is not valid (i.e., your answer is unchanged).
4. Update Your Candidate:
 - If the counterfactual is valid, try to revise it to make it closer to the original input (i.e., lower Gower's Distance).
 - If the counterfactual is invalid, revise it to be further from the original input (i.e., higher Gower's Distance).
5. Repeat steps 2-4. You must evaluate at least 5 distinct revised inputs.
6. Select the Minimum, Valid Counterfactual: From all valid counterfactuals you found, choose the one with the lowest Gower's Distance to the original data. This is your final answer.

This is a self-modelling task: at each step, you must simulate how you would respond to the revised input. Your final output will be judged by an independent instance of yourself.

Only respond with JSON output. Do not include any additional words in your answer. Format your response as a JSON file with the keys as provided below:
{ "area": "", "bedrooms": "", "bathrooms": "", "floors": ""}

Figure 15: **Prompt for self-prediction ablation experiment.** We provide the models with a metacognition-inspired prompt with a six-step plan to generate a minimal valid counterfactual. This prompt encourages the LLMs to make explicit self-predictions for every candidate SCE they consider.

<think>

Alright, so I need to figure out how to minimally revise the original house data so that the predicted price changes from 'Above \$1,500,000' to 'Below \$1,500,000'. The original data is:



- Area: 10,000 sq ft
- Bedrooms: 5
- Bathrooms: 4
- Floors: 3

First, I need to understand what factors influence the house price. Generally, larger houses with more bedrooms, bathrooms, and floors tend to be more expensive. So, to reduce the price, I should aim to decrease one or more of these factors.

I'll start by considering each feature and how changing it might affect the price.

1. **Area**: Reducing the area will likely decrease the price. The area is currently at the maximum of 10,000 sq ft. If I decrease it, even by a small amount, it might help. But since the options are in increments of 500 sq ft, the next lower option is 9,500 sq ft.
2. **Bedrooms**: The house has 5 bedrooms, which is the maximum. Reducing the number of bedrooms to 4 might lower the price.
3. **Bathrooms**: There are 4 bathrooms, which is also the maximum. Reducing this to 3 could help.
4. **Floors**: The house has 3 floors, which is not the maximum (since the maximum is 4). Reducing the number of floors to 2 might lower the price.

Now, I need to determine which change would have the minimal impact on the Gower's Distance. Gower's Distance is calculated as the average of the per-field distances. For each numeric or ordinal field, the distance is the absolute difference divided by the variable's full range. For categorical fields, it's 0 if unchanged and 1 if changed.

Let's calculate the Gower's Distance for each possible single change:

- **Changing Area from 10,000 to 9,500**:
 - Difference: 500
 - Full range: 9,500 (from 500 to 10,000)
 - Distance: $500 / 9,500 \approx 0.0526$
 - Other fields unchanged, so total distance ≈ 0.0526
- **Changing Bedrooms from 5 to 4**:
 - Difference: 1
 - Full range: 4 (from 1 to 5)
 - Distance: $1 / 4 = 0.25$
 - Other fields unchanged, so total distance = 0.25
- **Changing Bathrooms from 4 to 3**:
 - Difference: 1
 - Full range: 3 (from 1 to 4)
 - Distance: $1 / 3 \approx 0.3333$
 - Other fields unchanged, so total distance ≈ 0.3333
- **Changing Floors from 3 to 2**:
 - Difference: 1
 - Full range: 3 (from 1 to 4)
 - Distance: $1 / 3 \approx 0.3333$
 - Other fields unchanged, so total distance ≈ 0.3333

From these calculations, changing the area to 9,500 results in the smallest Gower's Distance of approximately 0.0526. This seems like the minimal change needed. However, I need to ensure that this change actually results in the price being 'Below \$1,500,000'.

If I reduce the area to 9,500 sq ft, keeping the other features the same, would that lower the price below \$1,500,000? Given that the area is a significant factor, a reduction of 500 sq ft might be enough to tip the price below the threshold.

To confirm, I'll simulate the prediction with the revised data:

- Area: 9,500
- Bedrooms: 5
- Bathrooms: 4
- Floors: 3

Based on my internal model, this revised house would likely be priced 'Below \$1,500,000'.

I should also check if making multiple smaller changes could result in a lower Gower's Distance, but since changing the area alone gives the smallest distance and achieves the desired price change, it's the most minimal revision.

Therefore, the revised house data with the minimal change is:

- Area: 9,500
- Bedrooms: 5
- Bathrooms: 4
- Floors: 3

{“area”: “2000”, “bedrooms”: “1”, “bathrooms”: “2”, “floors”: “2”}

Figure 16: A reasoning trace from DeepSeek-R1 70B following step-by-step self-prediction instructions. We prompt the model using the template from §E.



Now revise the original house data such that your answer to the question on the revised house data would have been Below \$1,500,000.

Here are the valid options and data types.

area: ['500', '1000', '1500', '2000', '2500', '3000', '3500', '4000', '4500', '5000', '5500', '6000', '6500', '7000', '7500', '8000', '8500', '9000', '9500', '10000'] (ordinal)
bedrooms: 1-5 inclusive (integer)
bathrooms: 1-4 inclusive (integer)
floors: 1-4 inclusive (integer)

Share the revised house data.

Only respond with JSON output. Do not include any additional words in your answer.
Format your response as a JSON file with the keys as provided below:

```
{"area": "", "bedrooms": "", "bathrooms": "", "floors": ""}
```



<think>

Alright, so I need to figure out how to revise the original house data so that the price would be below \$1,500,000. Let's start by looking at the original data:

- Area: 8000 sq ft
- Bedrooms: 1
- Bathrooms: 4
- Floors: 3

The original answer was "Above \$1,500,000," so I need to adjust these parameters to bring the price down.

The model does not approach the problem as a self-explanation task.

First, the area is quite large at 8000 sq ft. Larger homes tend to be more expensive, so reducing the area would likely lower the price. I'll consider decreasing the area to a smaller size. Looking at the options, 500 sq ft is the smallest, but that might be too extreme. Maybe something like 2000 sq ft would be more reasonable.

Next, the number of bedrooms is 1. While having more bedrooms can increase the price, having fewer bedrooms might not necessarily decrease it significantly. However, in some cases, a house with only 1 bedroom might be seen as less desirable, potentially affecting the price. But since the original had 1 bedroom and was still expensive, maybe keeping it at 1 is okay.

Bathrooms are at 4, which is quite high. More bathrooms usually mean a higher price. Reducing the number of bathrooms could help lower the price. Maybe bringing it down to 2 bathrooms would make a difference.

Floors are at 3. More floors can increase the price because it often means more space or more rooms. Reducing the number of floors to 2 might help in lowering the price.

No self-prediction

Putting it all together, I think reducing the area to 2000 sq ft, keeping the bedrooms at 1, reducing bathrooms to 2, and lowering the floors to 2 should bring the price below \$1,500,000. This configuration represents a smaller, more modest house compared to the original data.

</think>

```
{"area": "2000", "bedrooms": "1", "bathrooms": "2", "floors": "2"}
```

Figure 17: **A full reasoning trace from DeepSeek-R1 70B.** Excerpts from the model response suggest that, despite the question being clearly framed as a self-explanation task ('revise the original house data such that *your* answer to the question...'), DeepSeek-R1 70B does not interpret the problem as such. The model instead appeals to general intuitions (e.g., 'More bathrooms usually mean a higher price') and never considers its own decision-making process or explicitly self-predicts.