Specify the output format

If you want the model to be concise, tell it so. Long outputs are not only costly (model APIs charge per token) but they also increase latency. If the model tends to begin its response with preambles such as "Based on the content of this essay, I'd give it a score of...", make explicit that you don't want preambles.

Ensuring the model outputs are in the correct format is essential when they are used by downstream applications that require specific formats. If you want the model to generate JSON, specify what the keys in the JSON should be. Give examples if necessary.

For tasks expecting structured outputs, such as classification, use markers to mark the end of the prompts to let the model know that the structured outputs should begin.[8] Without markers, the model might continue appending to the input, as shown in Table 5-3. Make sure to choose markers that are unlikely to appear in your inputs. Otherwise, the model might get confused.

Table 5-3. Without explicit markers to mark the end of the input, a model might continue appending to it instead of generating structured outputs.

Prompt	Model's output	

`Label the following item as edible or inedible.`
 	
`pineapple pizza --> edible`
`cardboard --> inedible`
`chicken`
	`tacos --> edi`
	`ble`	

`Label the following item as edible or inedible.`
 	
`pineapple pizza --> edible`
`cardboard --> inedible`
`chicken -->`
	`edible`	

Provide Sufficient Context

Just as reference texts can help students do better on an exam, sufficient context can help models perform better. If you want the model to answer questions about a paper, including that paper in the context will likely improve the model's responses. Context can also mitigate hallucinations. If the model isn't provided with the necessary information, it'll have to rely on its internal knowledge, which might be unreliable, causing it to hallucinate.

8 Recall that a language model, by itself, doesn't differentiate between user-provided input and its own generation, as discussed in Chapter 2.