



LangChain Academy

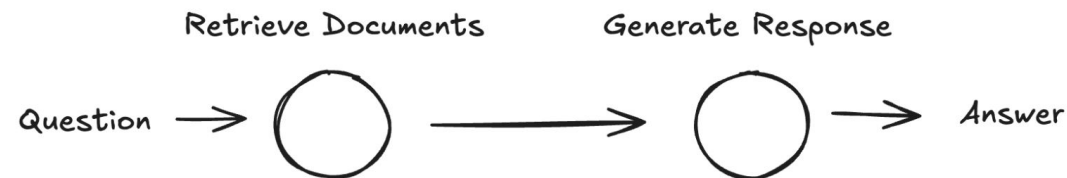
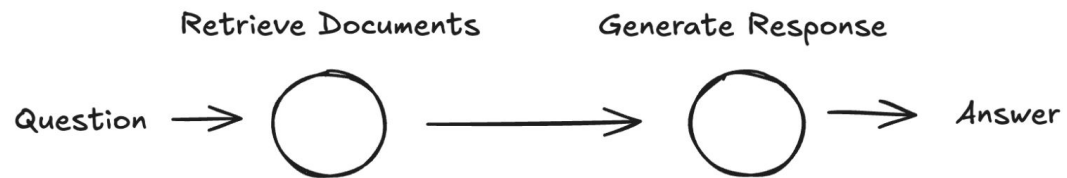
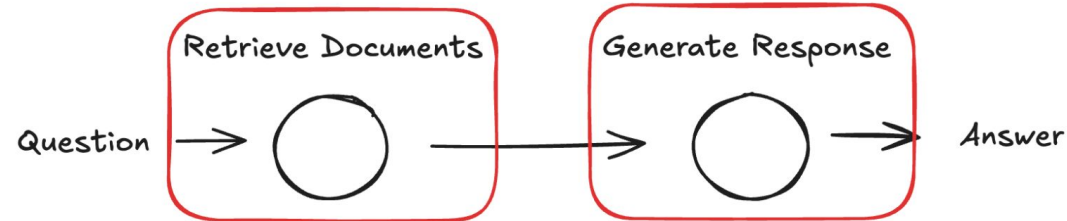
Tracing Basics

Tracing Concepts in LangSmith

Project

Traces

Runs



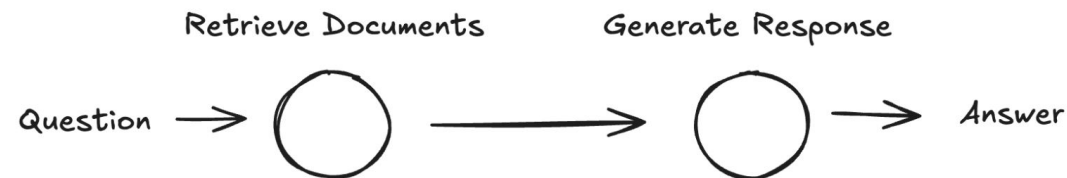
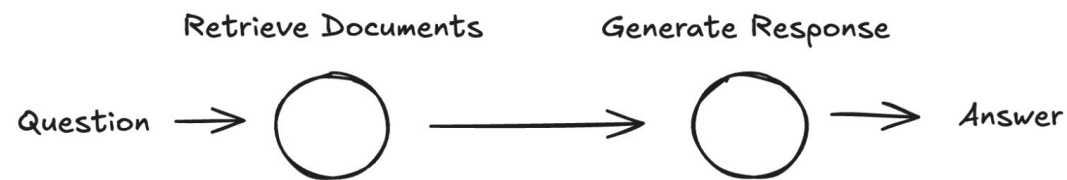
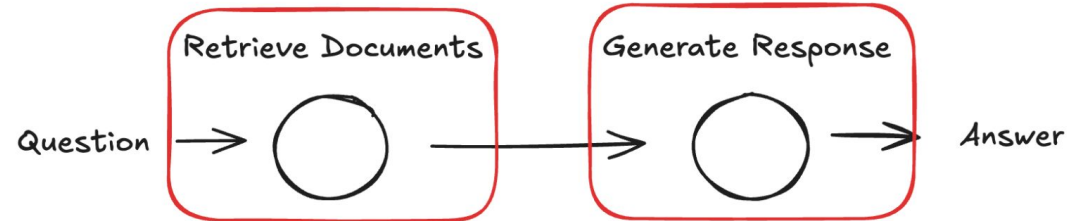
...

Tracing Concepts in LangSmith

Project

Traces

Runs

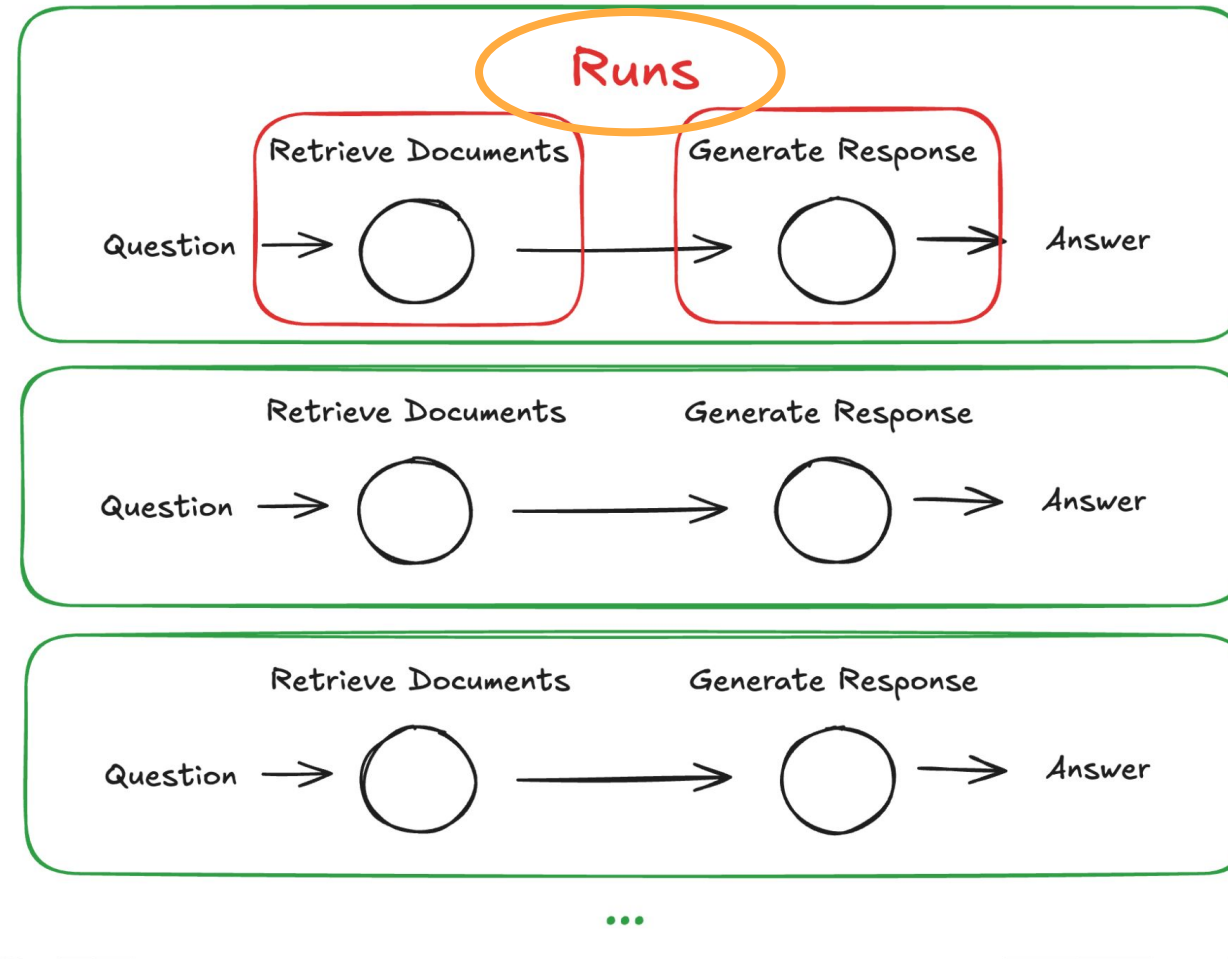


...

Tracing Concepts in LangSmith

Project

Traces



Tracing Concepts in LangSmith

Trace

Run Tree

RAG

- Retrieve Documents
 - SKLearn Vector DB
- Generate Response
 - OpenAI Chat Completion

Run

Name: Retrieve Documents

- Input: What can I do with LangSmith?
- Output: Document A, Document B ...
- Metadata: {"db_provider": "sklearn"}
- Run Type: "chain"
- Latency: 0.22s
- Feedback: {"docs_are_good": true}



LangChain Academy

Types of Runs

Logs vs. Tracing in LangSmith

```
(((), {'supervisor': {'next': 'researcher'}})
----
(('researcher:ee1ffa79-59fc-f3f5-5da0-f2c473080a21',), {'agent': {'me
----
(('researcher:ee1ffa79-59fc-f3f5-5da0-f2c473080a21',), {'tools': {'me
----
(('researcher:ee1ffa79-59fc-f3f5-5da0-f2c473080a21',), {'agent': {'me
----
(((), {'researcher': {'messages': [HumanMessage(content='The current p
----
(((), {'supervisor': {'next': 'coder'}})
----
(('coder:b4f947a0-983f-d52d-631e-b6c6fb40e38e',), {'agent': {'message
----
(('coder:b4f947a0-983f-d52d-631e-b6c6fb40e38e',), {'tools': {'message
----
(('coder:b4f947a0-983f-d52d-631e-b6c6fb40e38e',), {'agent': {'message
----
(((), {'coder': {'messages': [HumanMessage(content='The population of
----
(((), {'supervisor': {'next': '__end__'}})
```



The screenshot shows the LangSmith tracing interface for a run named 'chat-langchain'. The left pane displays a 'TRACE' tree with the following steps and durations:

- chat-langchain (5.13s, 5,846 tokens)
 - FindDocs (0.70s)
 - RetrievalChainWith... (0.68s)
 - Retriever (0.68s)
 - GenerateResponse (5.07s)
 - ChatOpenAI (4.33s)

The right pane shows the chat history:

- HUMAN 09:01:00**: How do I use a RecursiveUrlLoader to load content from a page?
- AI 09:06:13**: To use a RecursiveUrlLoader to load content from a page, follow these steps:
Initialize the RecursiveUrlLoader with the target URL and other optional parameters such as max depth, asynchronous loading, extractor function, and more.
Use the 'load()' method to load web pages, which returns a list of documents
Optionally, you can use the 'lazy_load()' method for lazy loading or the 'load_and_split()' method to load documents and split them into chunks.

At the bottom, there are three buttons: 'Add to Dataset', 'Annotate', and 'Playground'.

Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

-  **LLM**: Invokes an LLM

Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

-  **LLM:** Invokes an LLM
-  **Retriever:** Retrieves documents from databases or other sources





Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

-  **LLM**: Invokes an LLM
-  **Retriever**: Retrieves documents from databases or other sources
-  **Tool**: Executes actions with function calls





Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

-  **LLM:** Invokes an LLM
-  **Retriever:** Retrieves documents from databases or other sources
-  **Tool:** Executes actions with function calls
-  **Chain:** Default type; combines multiple Runs into a larger process







Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

-  **LLM:** Invokes an LLM
-  **Retriever:** Retrieves documents from databases or other sources
-  **Tool:** Executes actions with function calls
-  **Chain:** Default type; combines multiple Runs into a larger process
-  **Prompt:** Hydrates a prompt to be used with an LLM

Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

-  **LLM:** Invokes an LLM
-  **Retriever:** Retrieves documents from databases or other sources
-  **Tool:** Executes actions with function calls
-  **Chain:** Default type; combines multiple Runs into a larger process
-  **Prompt:** Hydrates a prompt to be used with an LLM
-  **Parser:** Extracts structured data



LangChain Academy

Alternative Ways to Trace

Method	When-to-use
@traceable	<ul style="list-style-type: none">- Default way to set up tracing- Manages RunTree, inputs, and outputs for you

Method	When-to-use
@traceable	<ul style="list-style-type: none">- Default way to set up tracing- Manages RunTree, inputs, and outputs for you
LangChain/LangGraph	<ul style="list-style-type: none">- Get tracing out of the box!

Method	When-to-use
@traceable	<ul style="list-style-type: none">- Default way to set up tracing- Manages RunTree, inputs, and outputs for you
LangChain/LangGraph	<ul style="list-style-type: none">- Get tracing out of the box!
with trace():	<ul style="list-style-type: none">- Want more control over what inputs and outputs get logged- Cannot use decorators or wrappers (not tracing a function)

Method	When-to-use
<code>@traceable</code>	<ul style="list-style-type: none">- Default way to set up tracing- Manages RunTree, inputs, and outputs for you
<code>LangChain/LangGraph</code>	<ul style="list-style-type: none">- Get tracing out of the box!
<code>with trace():</code>	<ul style="list-style-type: none">- Want more control over what inputs and outputs get logged- Cannot use decorators or wrappers (not tracing a function)
<code>wrap_openai()</code>	<ul style="list-style-type: none">- Want to use the OpenAI SDK directly

Method	When-to-use
@traceable	<ul style="list-style-type: none">- Default way to set up tracing- Manages RunTree, inputs, and outputs for you
LangChain/LangGraph	<ul style="list-style-type: none">- Get tracing out of the box!
with trace():	<ul style="list-style-type: none">- Want more control over what inputs and outputs get logged- Cannot use decorators or wrappers (not tracing a function)
wrap_openai()	<ul style="list-style-type: none">- Want to use the OpenAI SDK directly
RunTree	<ul style="list-style-type: none">- Want more low level control over tracing configuration- Need to know run_id (for adding user feedback or logging to another service)







LangChain Academy

Conversational Threads

Types of Runs in LangSmith

LangSmith supports many different types of Runs - you can specify what type your Run is in the **@traceable** decorator. The types of runs are:

-  **LLM:** Invokes an LLM
-  **Retriever:** Retrieves documents from databases or other sources
-  **Tool:** Executes actions with function calls
-  **Chain:** Default type; combines multiple Runs into a larger process

Thread in LangSmith

LangChain Inc > Tracing projects > chat-langchain-langgraph-cloud > 8bb5b426-08ec-402a-a68e-1...

Team gtm chat-langchain-langgraph-cloud

Thread 8bb5b426-08ec-402a-a68e-18d9e2896a65 LangGraph Studio Edit view

TRACE COUNT
4

TOTAL TOKENS
66,778 / \$0.0109356

LATENCY
P50: 22.24s P99: 25.42s

FEEDBACK
about_langchain 0.33 category FROM_S... 2 +1 contains_hello 0.50 helpfulness_few_shot 1.00

TRACE

Collapse Stats Filter Show All

RetrievalGraph Turn 1

13.21s 14,295

RetrievalGraph Turn 2

23.40s 14,465

RetrievalGraph Turn 3

25.49s 20,372

RetrievalGraph Turn 4

21.07s 17,646

Turn 1

about_langchain 1.00 contains_hello 1.00 helpfulness_few_shot 1.00

HUMAN

which is the best text splitter to use that can preserve headings and document formats

AI

To preserve headings and document formats while splitting text, the following text splitters in LangChain are recommended:

- **MarkdownHeaderTextSplitter**:
 - Specifically designed for Markdown documents, it splits text based on specified headers (e.g., '#', '##', '###').
 - It can group content under common headers, maintaining the structure of the document.
 - You can specify whether to strip headers from the output or keep them intact.
- **HTMLHeaderTextSplitter**:
 - This splitter is tailored for HTML documents and splits text at the HTML element level.
 - It adds metadata for each header relevant to the chunk, preserving the context-rich information encoded in the document structure.
- **HTMLSectionSplitter**:
 - Similar to the HTMLHeaderTextSplitter, it splits text at the element level and can return chunks element by element or combine elements with the same metadata.
 - It is designed to keep related text grouped semantically and preserve the document's context.

These splitters are effective for maintaining the hierarchical structure of documents, making them suitable for applications that require the retention of headings and formats during the splitting process.

Turn 2

category FROM_S... contains_hello 0.00 helpfulness_few_shot 1.00

Thread in LangSmith

LangChain Inc > Tracing projects > chat-langchain-langgraph-cloud > 8bb5b426-08ec-402a-a68e-1...

Team gtm chat-langchain-langgraph-cloud

Thread 8bb5b426-08ec-402a-a68e-18d9e2896a65

LangGraph Studio Edit view

TRACE COUNT: 4 TOTAL TOKENS: 66,778 / \$0.0109356 LATENCY: P50: 22.24s P99: 25.42s

FEEDBACK: about_langchain 0.33 category FROM_S... 2 +1 contains_hello 0.50 helpfulness_few_shot 1.00

TRACE

Collapse Stats Filter Show All

RetrievalGraph Turn 1
13.21s 14,295

RetrievalGraph Turn 2
23.40s 14,465

RetrievalGraph Turn 3
25.49s 20,372

RetrievalGraph Turn 4
21.07s 17,646

Turn 1

about_langchain 1.00 contains_hello 1.00 helpfulness_few_shot 1.00

HUMAN

which is the best text splitter to use that can preserve headings and document formats

AI

To preserve headings and document formats while splitting text, the following text splitters in LangChain are recommended:

- **MarkdownHeaderTextSplitter**:
 - Specifically designed for Markdown documents, it splits text based on specified headers (e.g., '#', '##', '###').
 - It can group content under common headers, maintaining the structure of the document.
 - You can specify whether to strip headers from the output or keep them intact.
- **HTMLHeaderTextSplitter**:
 - This splitter is tailored for HTML documents and splits text at the HTML element level.
 - It adds metadata for each header relevant to the chunk, preserving the context-rich information encoded in the document structure.
- **HTMLSectionSplitter**:
 - Similar to the HTMLHeaderTextSplitter, it splits text at the element level and can return chunks element by element or combine elements with the same metadata.
 - It is designed to keep related text grouped semantically and preserve the document's context.

These splitters are effective for maintaining the hierarchical structure of documents, making them suitable for applications that require the retention of headings and formats during the splitting process.

Turn 2

category FROM_S... contains_hello 0.00 helpfulness_few_shot 1.00

Thread in LangSmith

LangChain Inc > Tracing projects > chat-langchain-langgraph-cloud > 8bb5b426-08ec-402a-a68e-1...

Team gtm chat-langchain-langgraph-cloud

Thread 8bb5b426-08ec-402a-a68e-18d9e2896a65

LangGraph Studio Edit view

TRACE COUNT TOTAL TOKENS LATENCY

4 66,778 / \$0.0109356 P50: 22.24s P99: 25.42s

FEEDBACK

about_langchain 0.33 category FROM_S... 2 +1 contains_hello 0.50 helpfulness_few_shot 1.00

TRACE

Collapse Stats Filter Show All

RetrievalGraph Turn 1

13.21s 14,295

RetrievalGraph Turn 2

23.40s 14,465

RetrievalGraph Turn 3

25.49s 20,372

RetrievalGraph Turn 4

21.07s 17,646

Turn 1

about_langchain 1.00 contains_hello 1.00 helpfulness_few_shot 1.00

HUMAN

which is the best text splitter to use that can preserve headings and document formats

AI

To preserve headings and document formats while splitting text, the following text splitters in LangChain are recommended:

- **MarkdownHeaderTextSplitter**:
 - Specifically designed for Markdown documents, it splits text based on specified headers (e.g., '#', '##', '###').
 - It can group content under common headers, maintaining the structure of the document.
 - You can specify whether to strip headers from the output or keep them intact.
- **HTMLHeaderTextSplitter**:
 - This splitter is tailored for HTML documents and splits text at the HTML element level.
 - It adds metadata for each header relevant to the chunk, preserving the context-rich information encoded in the document structure.
- **HTMLSectionSplitter**:
 - Similar to the HTMLHeaderTextSplitter, it splits text at the element level and can return chunks element by element or combine elements with the same metadata.
 - It is designed to keep related text grouped semantically and preserve the document's context.

These splitters are effective for maintaining the hierarchical structure of documents, making them suitable for applications that require the retention of headings and formats during the splitting process.

Turn 2

category FROM_S... contains_hello 0.00 helpfulness_few_shot 1.00

Thread in LangSmith

LangChain Inc > Tracing projects > chat-langchain-langgraph-cloud > 8bb5b426-08ec-402a-a68e-1...

Team gtm chat-langchain-langgraph-cloud

Thread 8bb5b426-08ec-402a-a68e-18d9e2896a65 LangGraph Studio Edit view

TRACE COUNT: 4 TOTAL TOKENS: 66,778 / \$0.0109356 LATENCY: P50: 22.24s P99: 25.42s FEEDBACK: about_langchain 0.33 category FROM_S... 2 +1 contains_hello 0.50 helpfulness_few_shot 1.00

TRACE

Collapse Stats Filter Show All

RetrievalGraph Turn 1

13.21s 14,295

RetrievalGraph Turn 2

23.40s 14,465

RetrievalGraph Turn 3

25.49s 20,372

RetrievalGraph Turn 4

21.07s 17,646

Turn 1

about_langchain 1.00 contains_hello 1.00 helpfulness_few_shot 1.00

HUMAN

which is the best text splitter to use that can preserve headings and document formats

AI

To preserve headings and document formats while splitting text, the following text splitters in LangChain are recommended:

- **MarkdownHeaderTextSplitter**:
 - Specifically designed for Markdown documents, it splits text based on specified headers (e.g., '#', '##', '###').
 - It can group content under common headers, maintaining the structure of the document.
 - You can specify whether to strip headers from the output or keep them intact.
- **HTMLHeaderTextSplitter**:
 - This splitter is tailored for HTML documents and splits text at the HTML element level.
 - It adds metadata for each header relevant to the chunk, preserving the context-rich information encoded in the document structure.
- **HTMLSectionSplitter**:
 - Similar to the HTMLHeaderTextSplitter, it splits text at the element level and can return chunks element by element or combine elements with the same metadata.
 - It is designed to keep related text grouped semantically and preserve the document's context.

These splitters are effective for maintaining the hierarchical structure of documents, making them suitable for applications that require the retention of headings and formats during the splitting process.

Turn 2

category FROM_S... contains_hello 0.00 helpfulness_few_shot 1.00

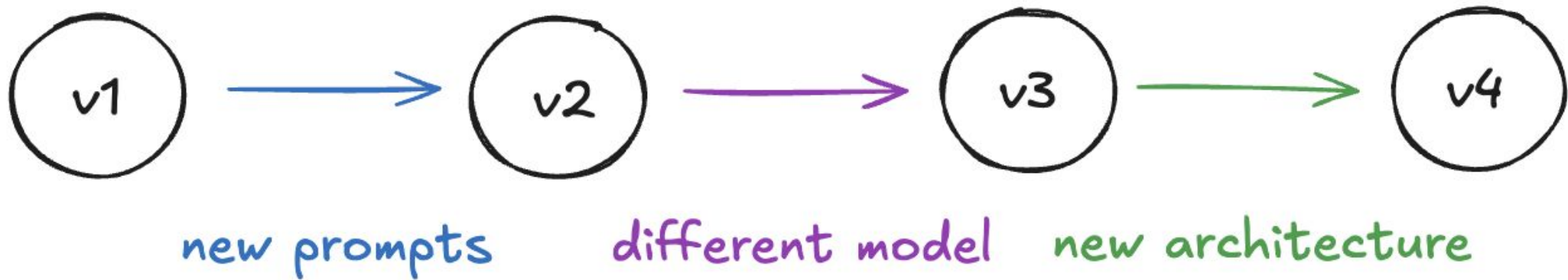


LangChain Academy

Datasets

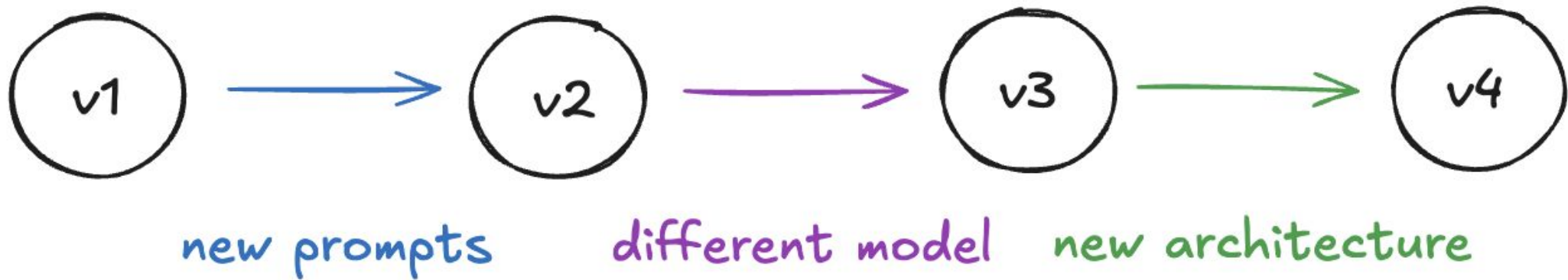
Datasets

How do we know that our application is getting better, not worse, over time?



Datasets

How do we know that our application is getting better, not worse, over time?



We need to do more than run a few “gut check” tests

Datasets

In order to test and evaluate our application, we need **datasets**

Dataset A

Example 1	Input=x1	Output=y1
-----------	----------	-----------

Example 2	Input=x2	Output=y2
-----------	----------	-----------

Example 3	Input=x3	Output=y3
-----------	----------	-----------

Example 4	Input=x4	Output=y4
-----------	----------	-----------

...

Datasets

In order to test and evaluate our application, we need **datasets**

Dataset A

Example 1 Input= x_1 Output= y_1

Example 2 Input= x_2 Output= y_2

Example 3 Input= x_3 Output= y_3

Example 4 Input= x_4 Output= y_4

...

Datasets are fundamentally a
list of examples

Datasets

In order to test and evaluate our application, we need **datasets**

Dataset A

Example 1	Input=x1	Output=y1
Example 2	Input=x2	Output=y2
Example 3	Input=x3	Output=y3
Example 4	Input=x4	Output=y4
...		

Datasets are fundamentally a **list of examples**

Examples contain an **input**, and an optional **output**

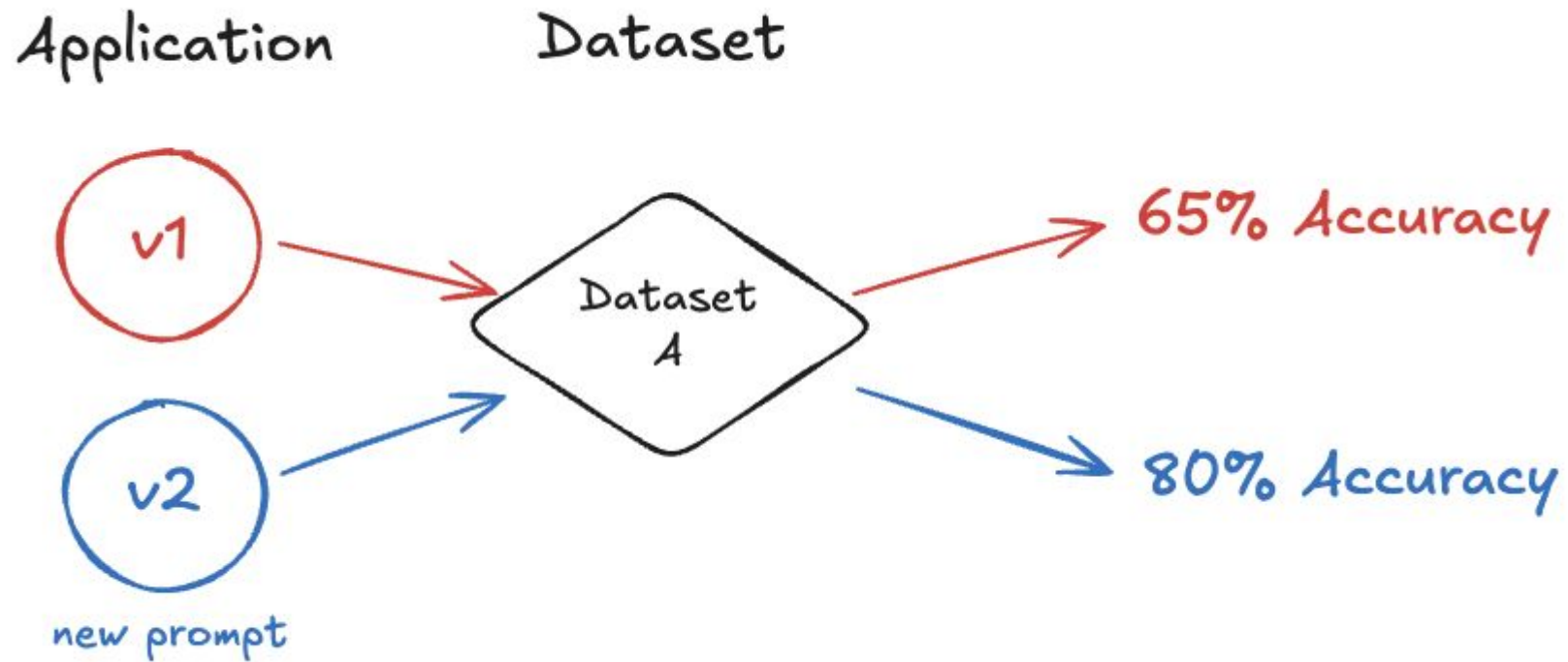


LangChain Academy

Evaluators

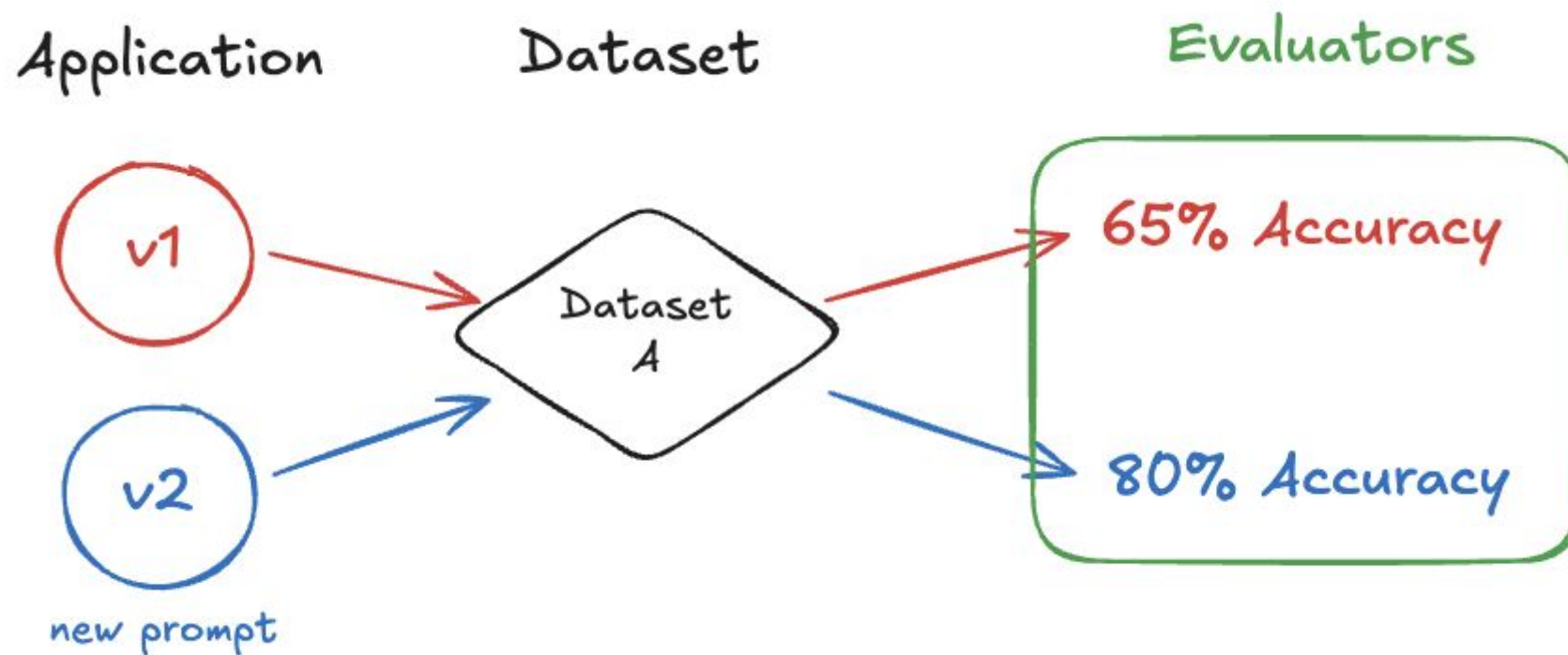
Evaluators

How do we measure performance against our datasets?



Evaluators

How do we measure performance against our datasets?



Evaluators

You can define evaluators for multiple metrics (ex. accuracy, hallucination)



Evaluators

Evaluators compare the dataset **example** against the output **run** from your app

Example

Input=x1

Reference Output=y1

Evaluators

Evaluators compare the dataset **example** against the output **run** from your app

Example

Input=x1

Reference Output=y1

Evaluators

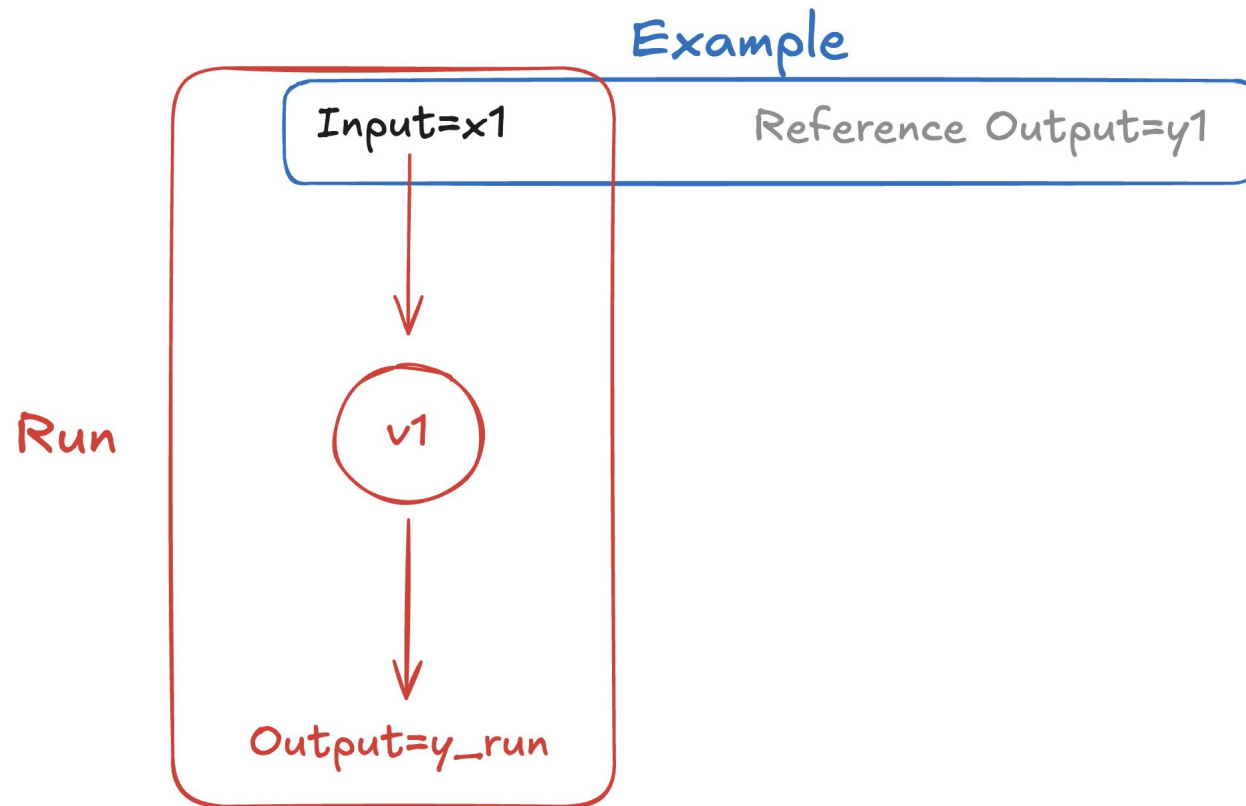
Evaluators compare the dataset **example** against the output **run** from your app

- From a Trace
- Manually Added
- AI Generated
- etc.



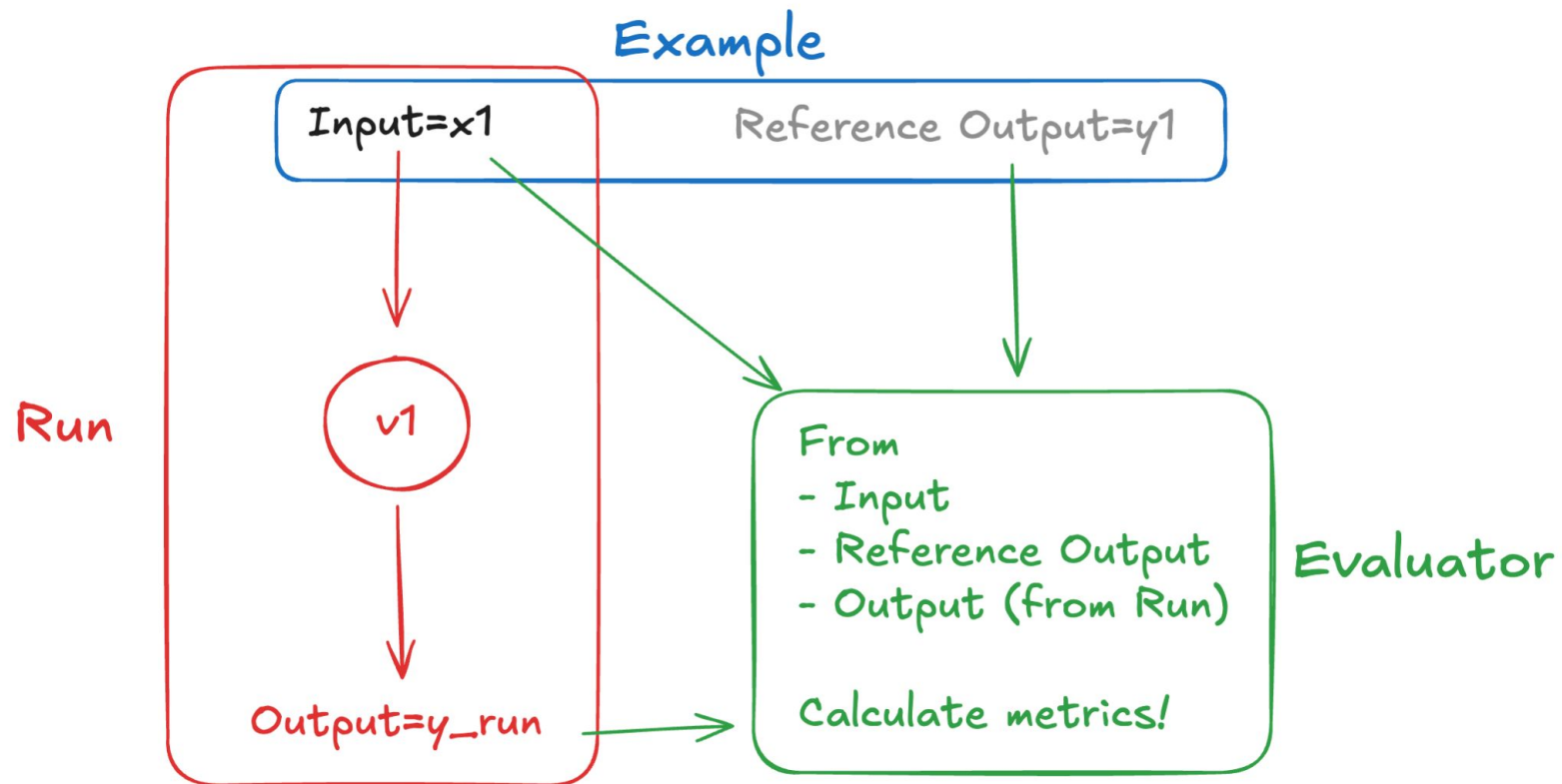
Evaluators

Evaluators compare the dataset **example** against the output **run** from your app



Evaluators

Evaluators compare the dataset **example** against the output **run** from your app



Takeaways

- Evaluators calculate metrics based on a Run and an Example

Takeaways

- Evaluators calculate metrics based on a Run and an Example
 - Specifically, from the Input, Reference output, and Run Output

Takeaways

- Evaluators calculate metrics based on a Run and an Example
 - Specifically, from the Input, Reference output, and Run Output
- You can define Evaluators directly in your local code

Takeaways

- Evaluators calculate metrics based on a Run and an Example
 - Specifically, from the Input, Reference output, and Run Output
- You can define Evaluators directly in your local code
- You can ALSO define Evaluators in the LangSmith UI
 - LLM-as-judge evaluators
 - Custom code evaluators



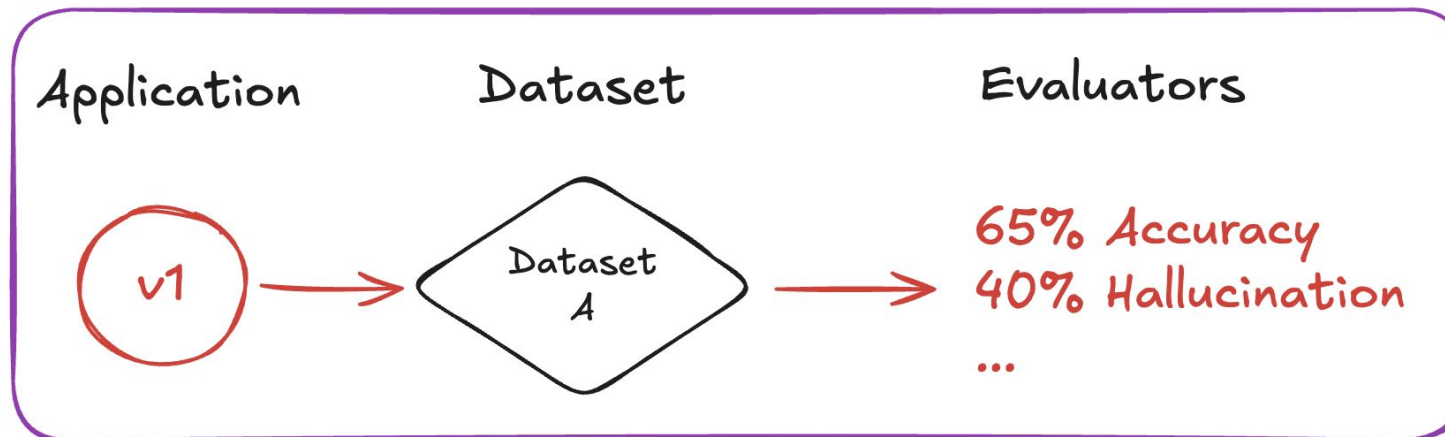
LangChain Academy

Experiments

Experiments

Experiment: Running your **application** over a **dataset**, and **evaluating** performance

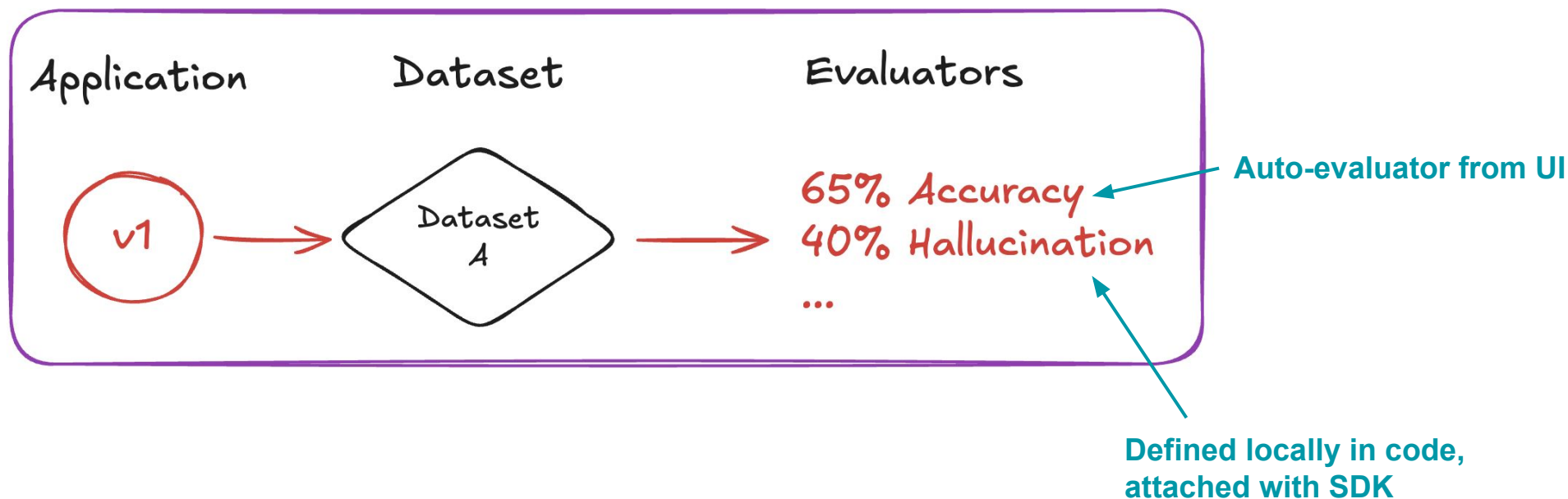
Experiment



Experiments

You can attach **evaluators** to your **experiment** in the UI, or locally with the SDK

Experiment



Takeaways

- Experiment: Running your **application** over a **dataset**, and **evaluating** its performance

Takeaways

- Experiment: Running your **application** over a **dataset**, and **evaluating** its performance
- Experiments can be run using the LangSmith SDK with **evaluate()**

Takeaways

- Experiment: Running your **application** over a **dataset**, and **evaluating** its performance
- Experiments can be run using the LangSmith SDK with **evaluate()**
- Experiments can be run over an entire dataset OR
 - A specific version
 - A split
 - Specific examples

Takeaways

- Experiment: Running your **application** over a **dataset**, and **evaluating** its performance
- Experiments can be run using the LangSmith SDK with **evaluate()**
- Experiments can be run over an entire dataset OR
 - A specific version
 - A split
 - Specific examples
- Experiments can be run with other parameters
 - Repetitions
 - Concurrent Threads
 - Metadata

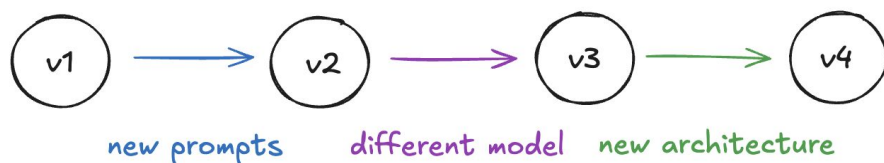


LangChain Academy

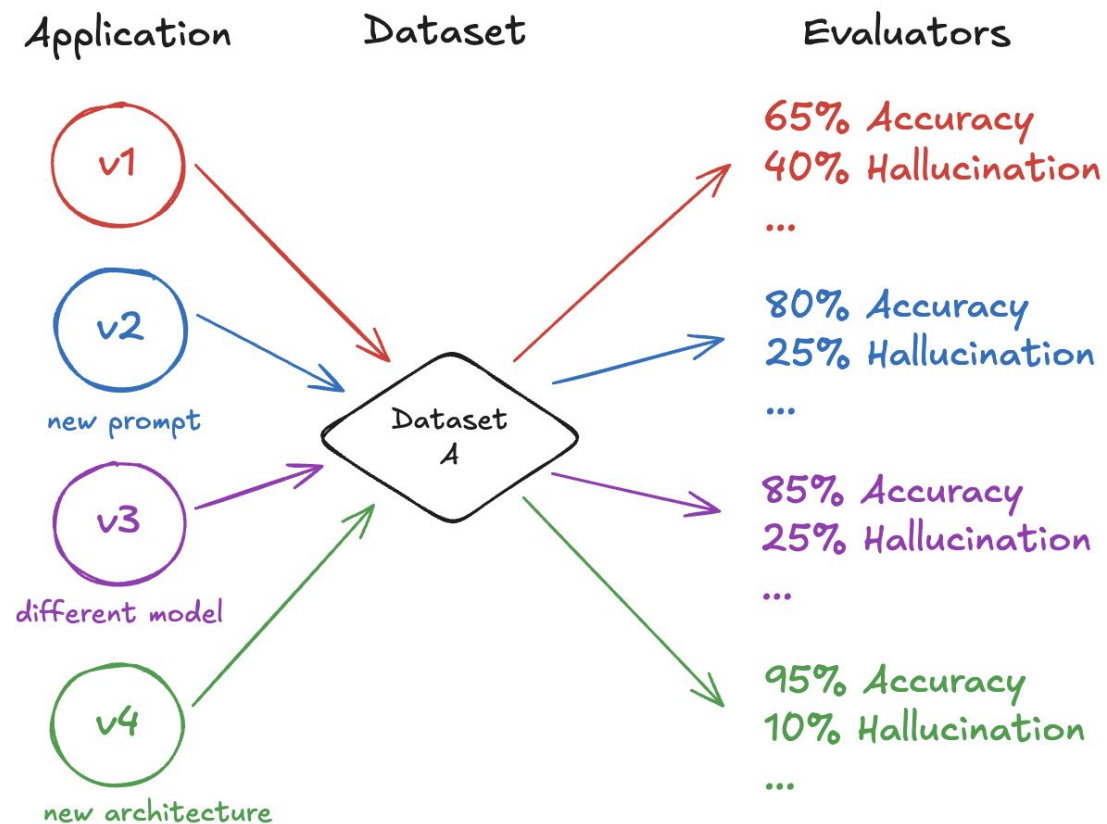
Analyzing Experiment Results

Analyzing Experiment Results

Without Testing



With Testing and Evaluation



Takeaways

- Experiments are useful for seeing trends in your application performance as you improve it over time

Takeaways

- Experiments are useful for seeing trends in your application performance as you improve it over time
- You can deep dive into a single experiment and look into how each individual run performed on the dataset example

Takeaways

- Experiments are useful for seeing trends in your application performance as you improve it over time
- You can deep dive into a single experiment and look into how each individual run performed on the dataset example
- You can compare multiple experiments side-by-side and see how they scored on your evaluator metrics

Takeaways

- Experiments are useful for seeing trends in your application performance as you improve it over time
- You can deep dive into a single experiment and look into how each individual run performed on the dataset example
- You can compare multiple experiments side-by-side and see how they scored on your evaluator metrics
- Experiments give you hard empirical data to push changes to production with confidence



LangChain Academy

Playground

Hard-coded Prompts → Prompt Templates

Hard-coded Prompt

You are a chat model who only speaks Spanish.

Prompt Template

You are a chat model who only speaks {language}

User
A

language = "English"

User
B

language = "Chinese"

User
C

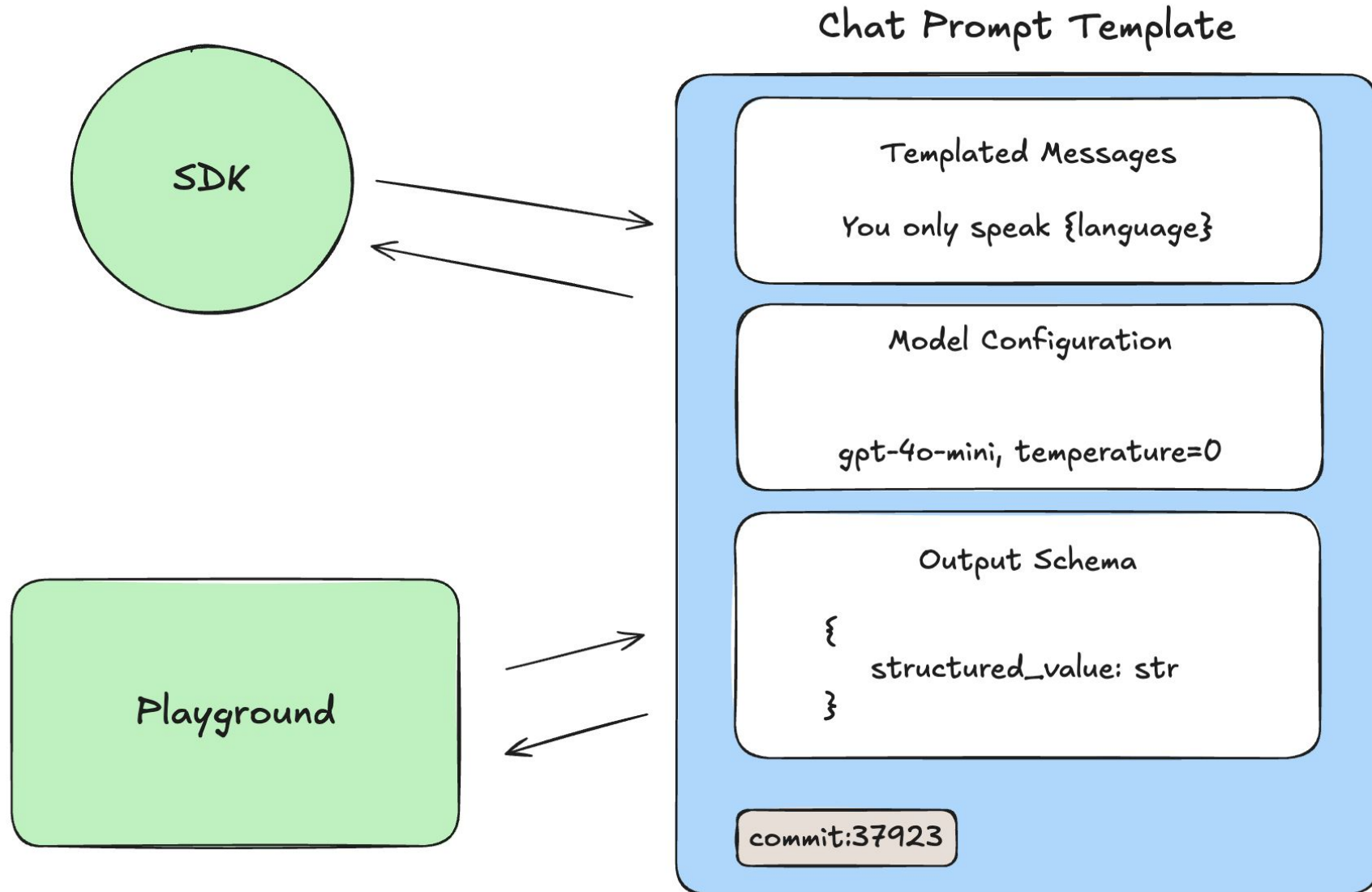
language = "Spanish"



LangChain Academy

Prompt Hub

Prompt Hub: Prompt Templates





LangChain Academy

Playground and Prompt Hub in Practice



LangChain Academy

User Feedback

User Feedback

It's very important to monitor how your users feel about your application's performance

User Feedback

Trace

Run Tree

RAG

- Retrieve Documents
 - SKLearn Vector DB
- Generate Response
 - OpenAI Chat Completion

Run

Name: Retrieve Documents

- Input: What can I do with LangSmith?
- Output: Document A, Document B ...
- Metadata: {"db_provider": "sklearn"}
- Run Type: "chain"
- Latency: 0.22s
- Feedback: {"docs_are_good": true}

Types of Feedback

Categorical

- is_correct (yes, no)
- type_of_input (integer, number, string)
- is_helpful (yes, no)
- did_succeed (succeeded, failed)

Continuous

- correctness 7 (1 - 10)
- similarity 3 (1 - 5)
- helpfulness 3 (1 - 10)

Types of Feedback

Categorical

- is_correct (yes, no)
- type_of_input (integer, number, string)
- is_helpful (yes, no)
- did_succeed (succeeded, failed)

Continuous

- correctness 7 (1 - 10)
- similarity 3 (1 - 5)
- helpfulness 3 (1 - 10)

ex.

```
{  
  "key": "is_correct"  
  "value": "yes"  
}
```

Types of Feedback

Categorical

- is_correct (yes, no)
- type_of_input (integer, number, string)
- is_helpful (yes, no)
- did_succeed (succeeded, failed)

ex.

```
{  
  "key": "is_correct"  
  "value": "yes"  
}
```

Continuous

- correctness 7 (1 - 10)
- similarity 3 (1 - 5)
- helpfulness 3 (1 - 10)

ex.

```
{  
  "key": "correctness"  
  "score": 7  
}
```


Feedback Tags

Feedback Tags

New tag

Create and organize feedback tags for this tenant

3184 feedback tags		
Feedback	Type	Values
<div><div></div> faithfulness_score</div>	Continuous	
<div><div></div> answer_relevancy_score</div>	Continuous	
<div><div></div> context_relevancy_score</div>	Continuous	
<div><div></div> test_entity_examples</div>	Continuous	
<div><div></div> test_employer_org_bias</div>	Continuous	
<div><div></div> test_person_profile_bias</div>	Continuous	

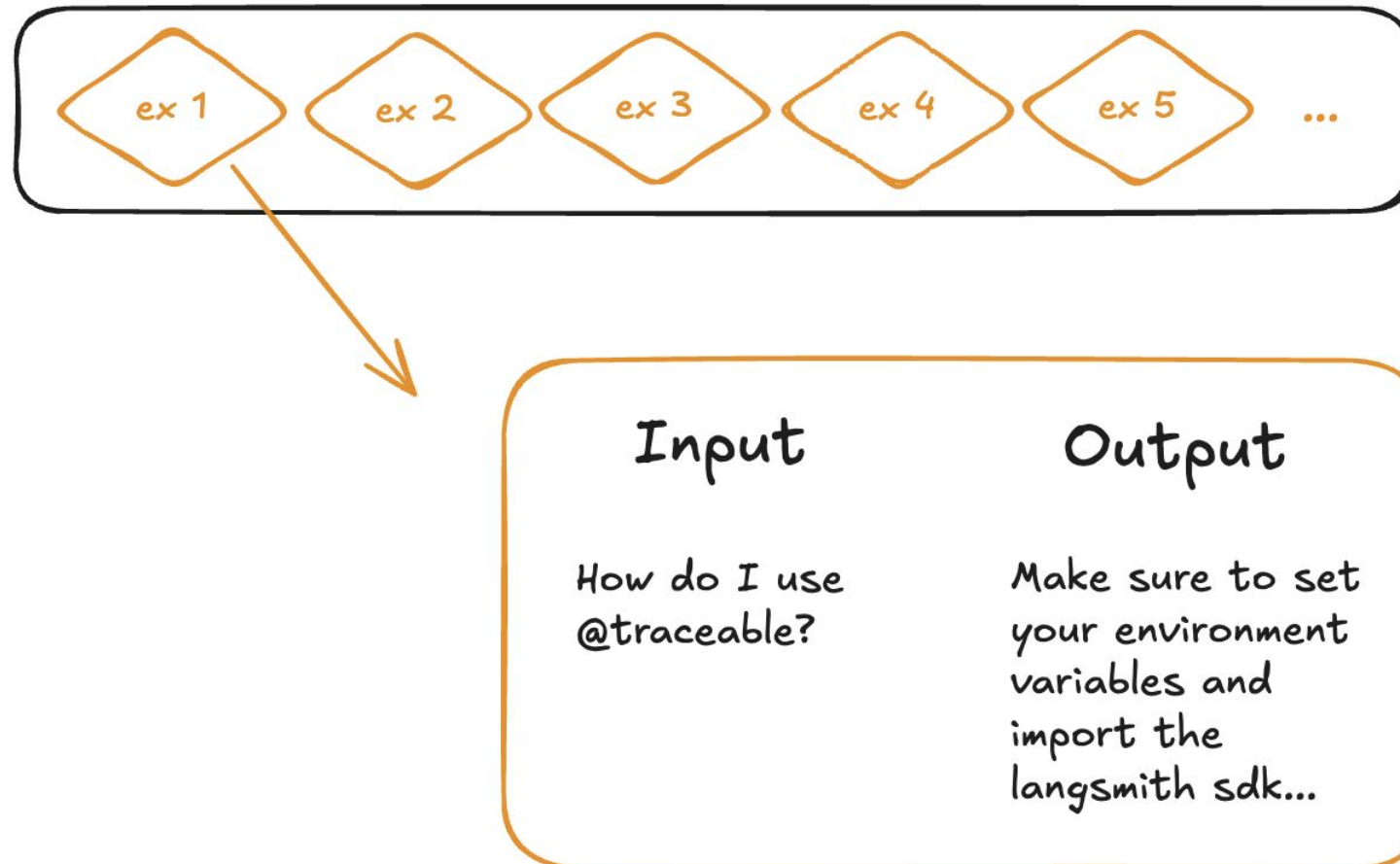


LangChain Academy

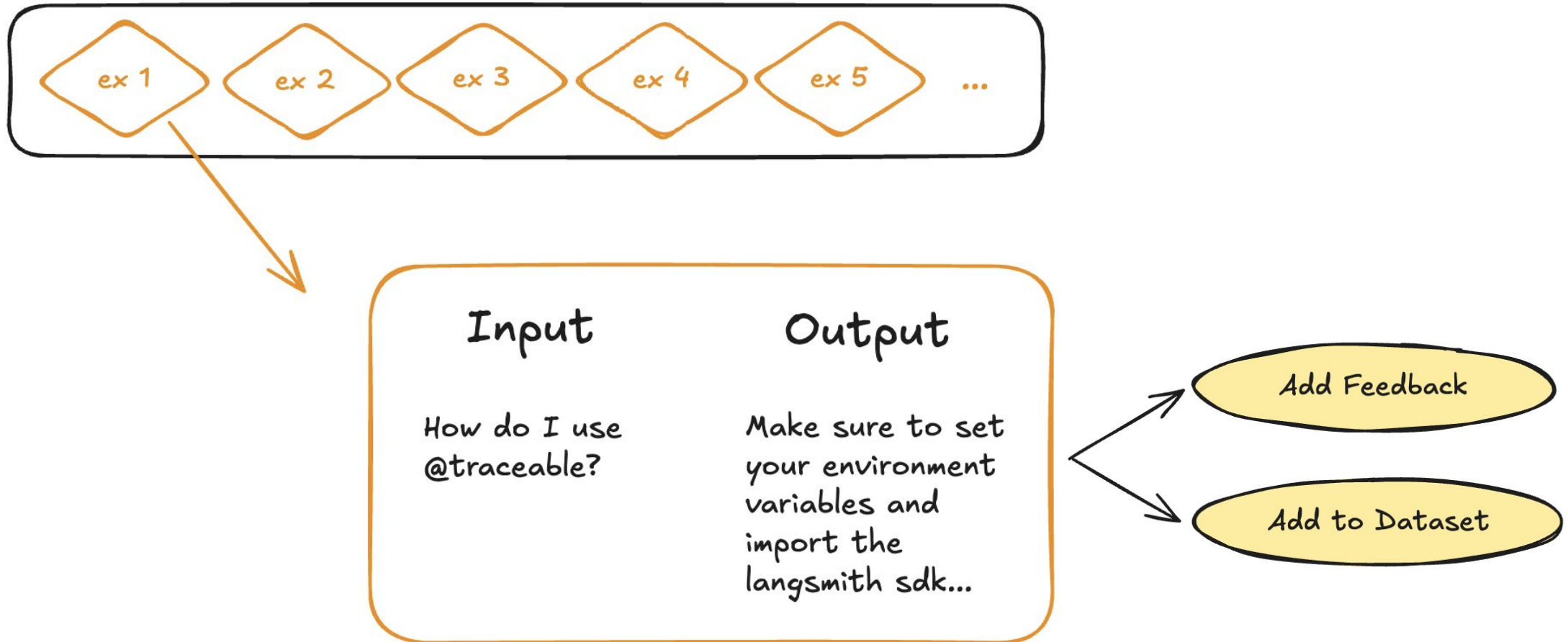
Annotation Queues

Annotation Queues

Annotation Queues are a user friendly way to quickly cycle through and annotate data



Annotation Queues





LangChain Academy

Filtering

Common Filters to Create

Here are some common filters that we see!

Common Filters to Create

Here are some common filters that we see!

- **Status = “error”** (all runs with errors)

Common Filters to Create

Here are some common filters that we see!

- **Status = “error”** (all runs with errors)
- **Feedback score < X** (received bad feedback)

Common Filters to Create

Here are some common filters that we see!

- **Status = “error”** (all runs with errors)
- **Feedback score < X** (received bad feedback)
- **run_type = “LLM” AND latency > X** (slow LLM runs)

Common Filters to Create

Here are some common filters that we see!

- **Status = “error”** (all runs with errors)
- **Feedback score < X** (received bad feedback)
- **run_type = “LLM” AND latency > X** (slow LLM runs)
- **Metadata[“Is_model_name”] = X** (create a filter for a metadata field)



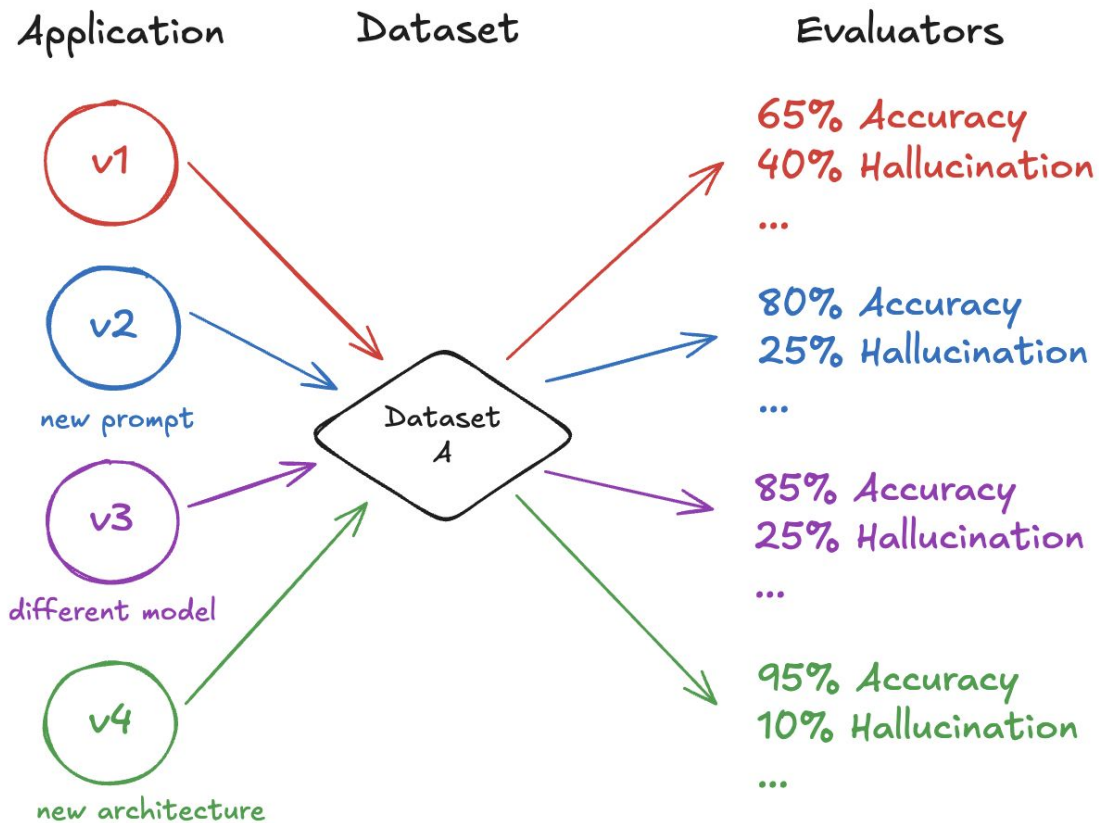
LangChain Academy

Online Evaluation

Offline and Online Evaluation

Recap: Offline Evaluation involves testing different versions of your Application against a dataset

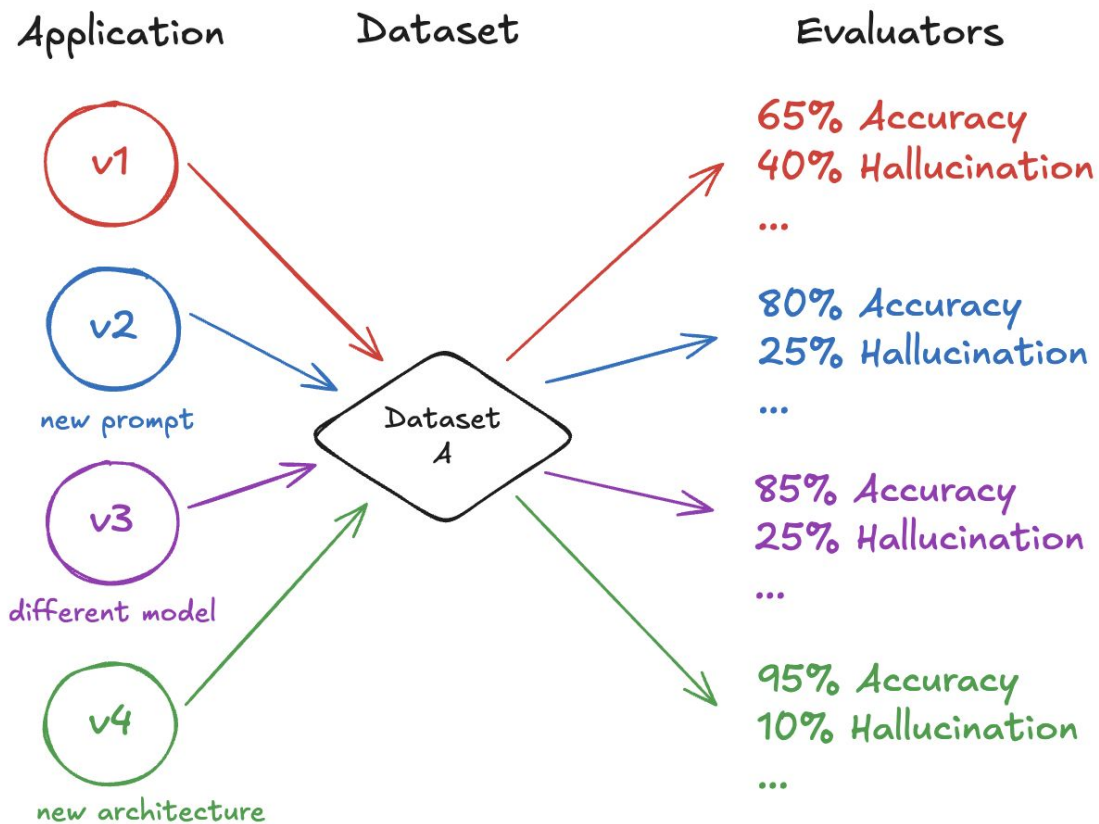
Offline Evaluation



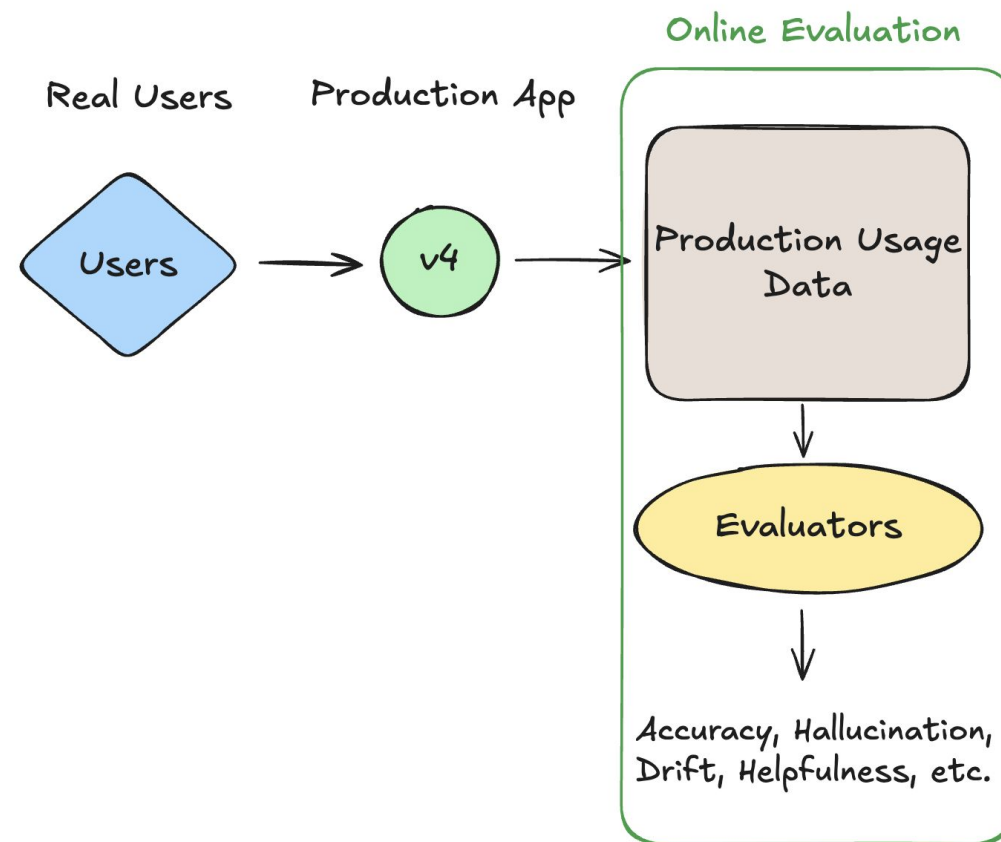
Offline and Online Evaluation

Online Evaluation involves evaluating real production performance with end users

Offline Evaluation



Online Evaluation

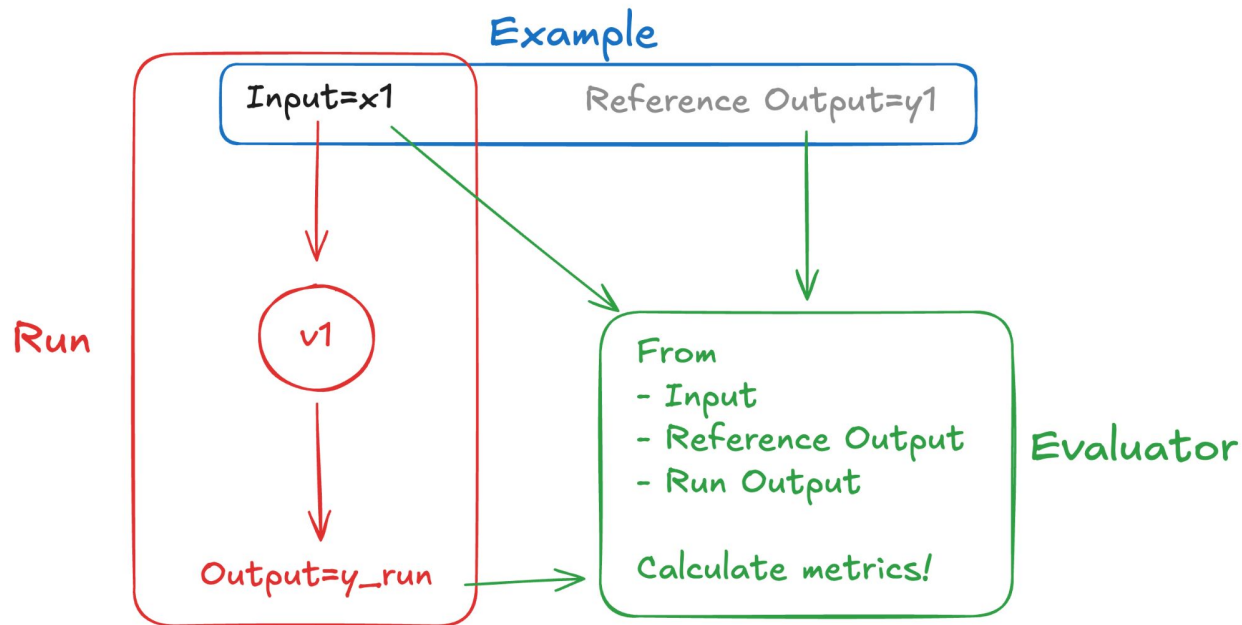


Offline and Online Evaluation

Offline Evaluators can make use Reference Output

Offline Evaluator

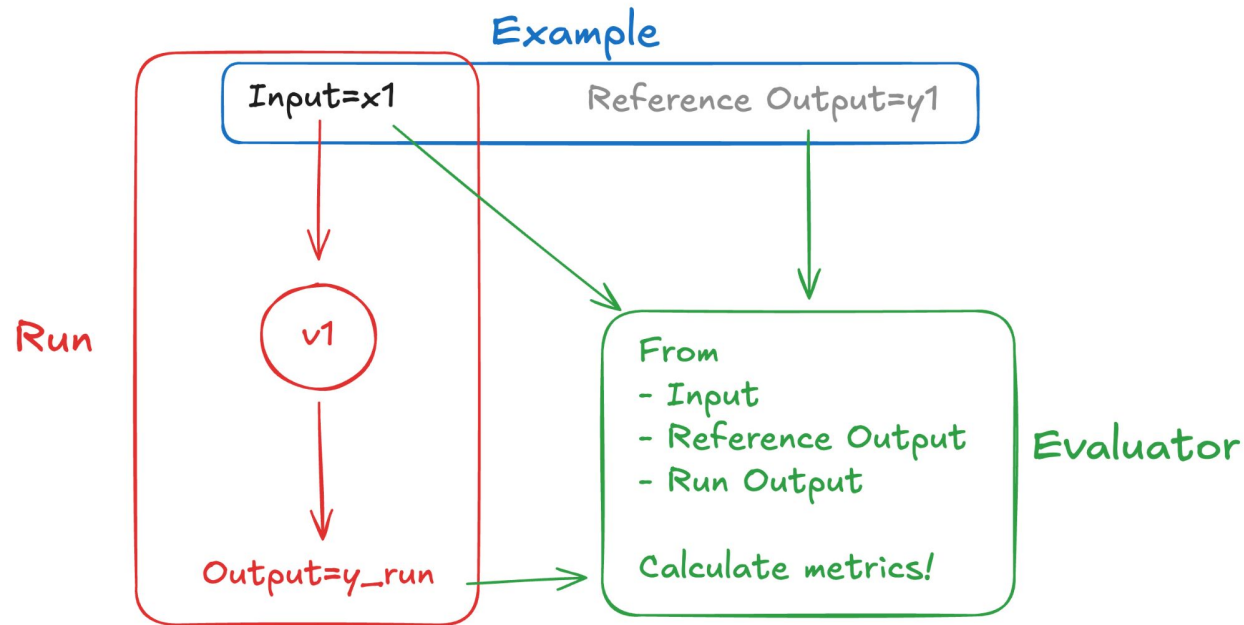
Online Evaluator



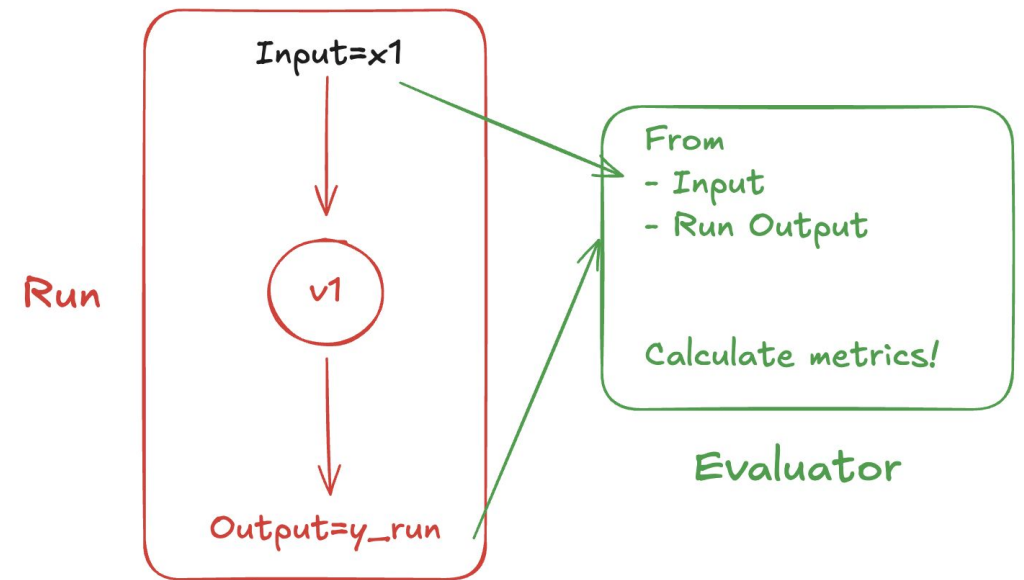
Offline and Online Evaluation

Online Evaluators only have access to the Input, and the output of the production Run

Offline Evaluator



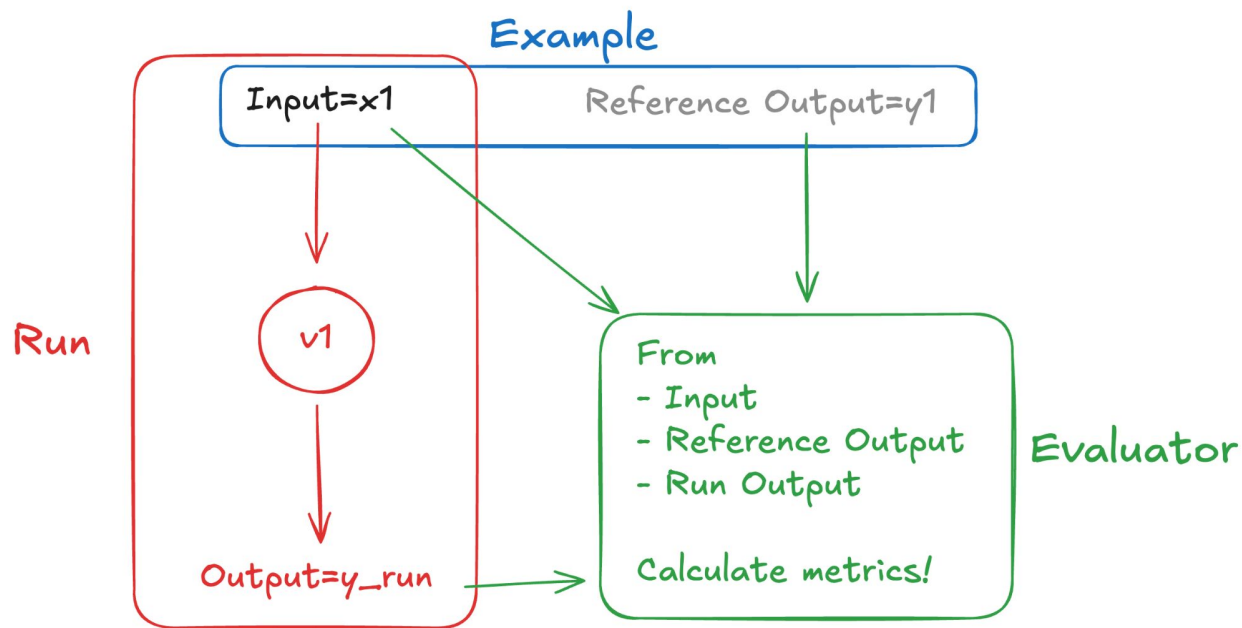
Online Evaluator



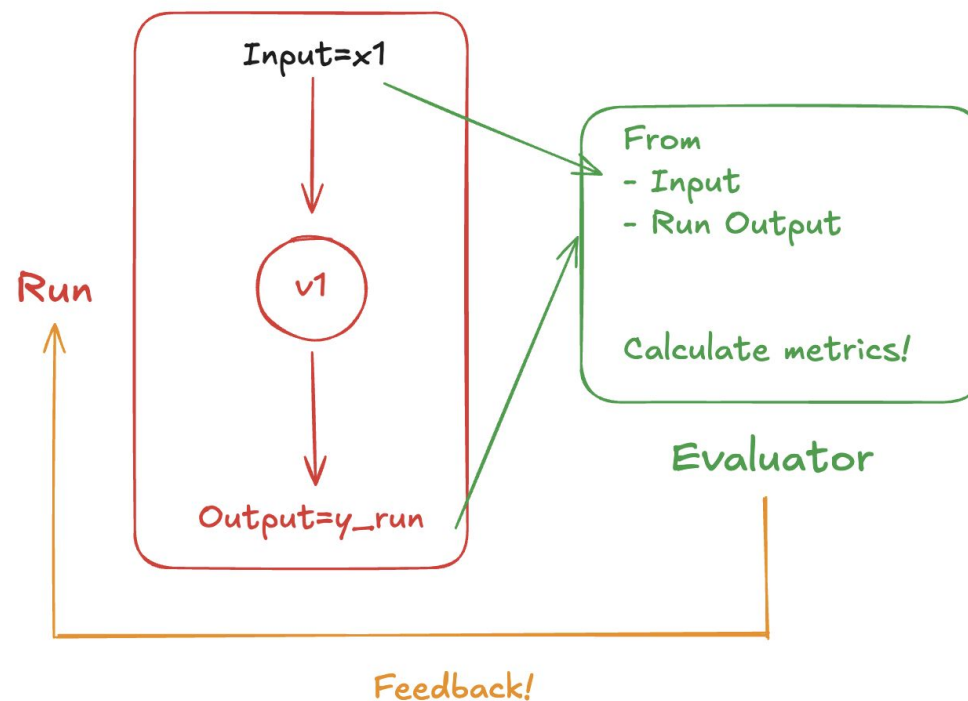
Offline and Online Evaluation

Online Evaluators only have access to the Input, and the output of the production Run

Offline Evaluator



Online Evaluator





LangChain Academy

Online Evaluation

Popular Examples of Online Evaluation

For document retrieval tasks, check for hallucinations in the output

LLM-as-Judge Online Evaluator

Check if this {answer} is grounded in these
{documents}

Popular Examples of Online Evaluation

For document retrieval tasks, check for hallucinations in the output

LLM-as-Judge Online Evaluator

Check if this {answer} is grounded in these
{documents}

Popular Examples of Online Evaluation

For document retrieval tasks, check for hallucinations in the output

LLM-as-Judge Online Evaluator

Check if this {answer} is grounded in these {documents}



Feedback Score

did_hallucinate = False

Popular Examples of Online Evaluation

Score the perceived helpfulness of an answer to a user

LLM-as-Judge Online Evaluator

Score the perceived helpfulness of the {answer}
to this {question} from 1-10

Popular Examples of Online Evaluation

Score the perceived helpfulness of an answer to a user

LLM-as-Judge Online Evaluator

Score the perceived helpfulness of the {answer}
to this {question} from 1-10



Feedback Score

helpfulness = 7

Popular Examples of Online Evaluation

For a Coding Assistant, check to see if the outputted code actually compiles and executes

Code Online Evaluator

Check if LLM output is a valid python code snippet that can execute with REPL

Popular Examples of Online Evaluation

For a Coding Assistant, check to see if the outputted code actually compiles and executes

Code Online Evaluator

Check if LLM output is a valid python code snippet that can execute with REPL



Feedback Score

`code_is_valid = True`

Popular Examples of Online Evaluation

Check that the output matches a certain structure
ex. Email Assistant should sign emails a certain way

Code Online Evaluator

For an Email Assistant, regex match that we correctly signed the email "Best wishes, Nick"

Popular Examples of Online Evaluation

Check that the output matches a certain structure
ex. Email Assistant should sign emails a certain way

Code Online Evaluator

For an Email Assistant, regex match that we correctly signed the email "Best wishes, Nick"



Feedback Score

`correctly_signed = True`

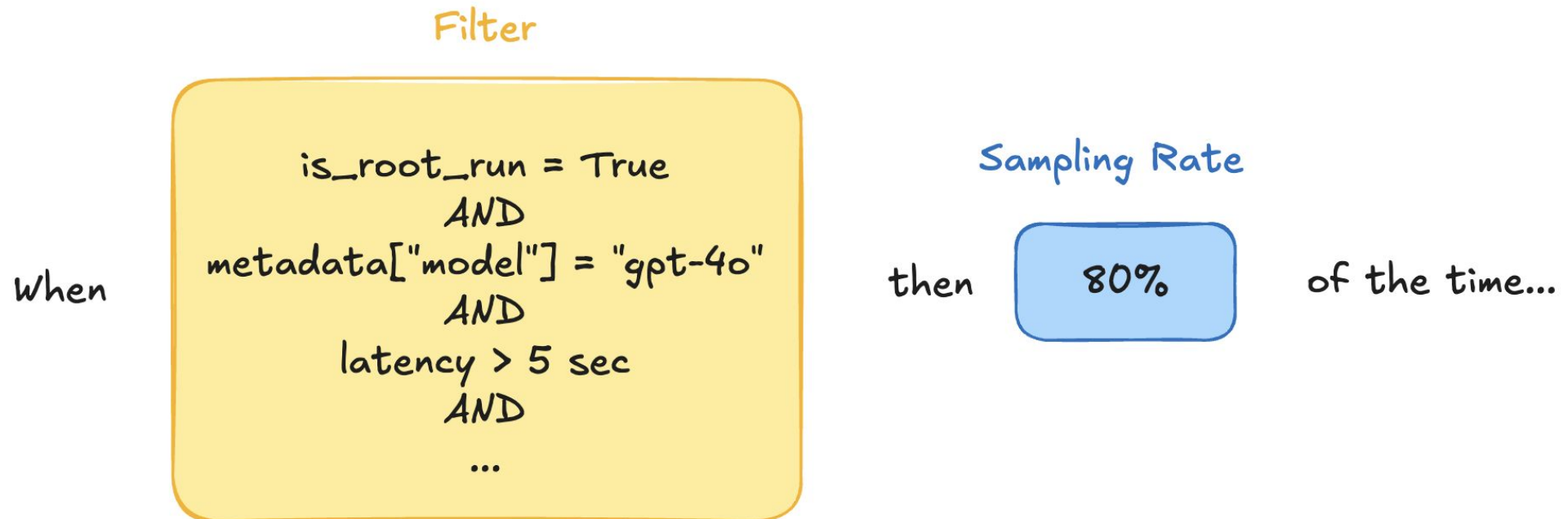


LangChain Academy

Automations

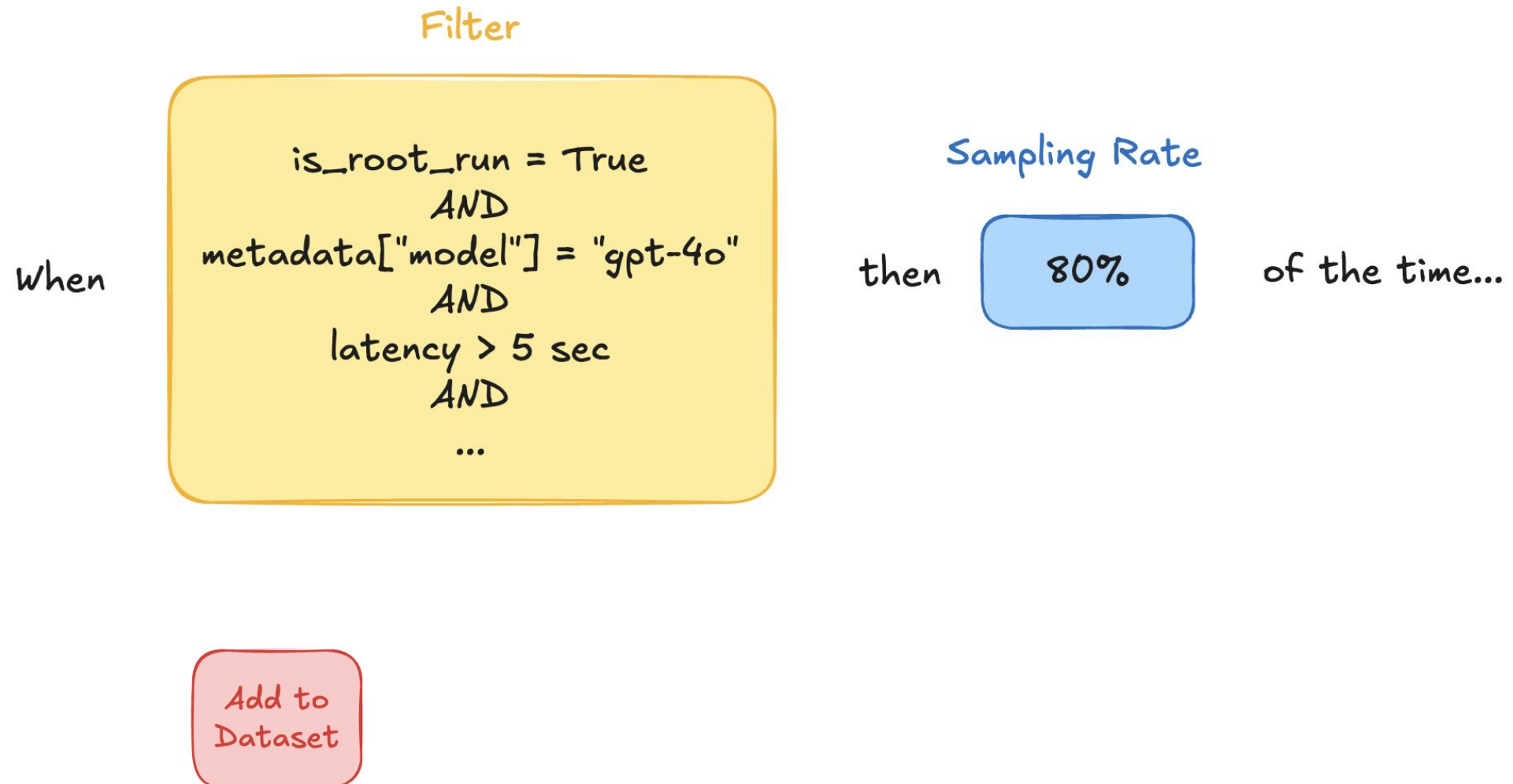
Automations

LangSmith allows you to create Rules that automatically trigger actions when in production



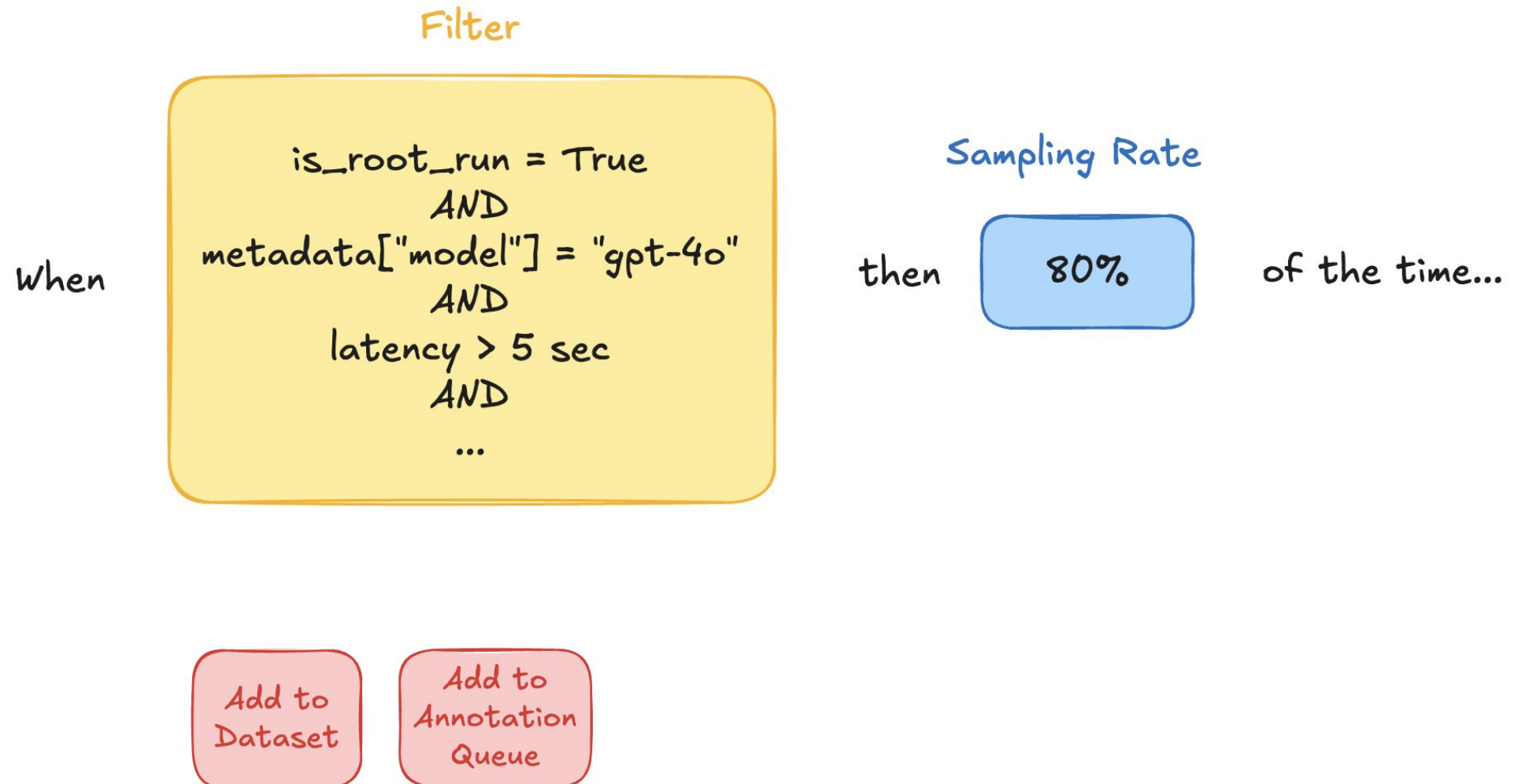
Automations

LangSmith allows you to create Rules that automatically trigger actions when in production



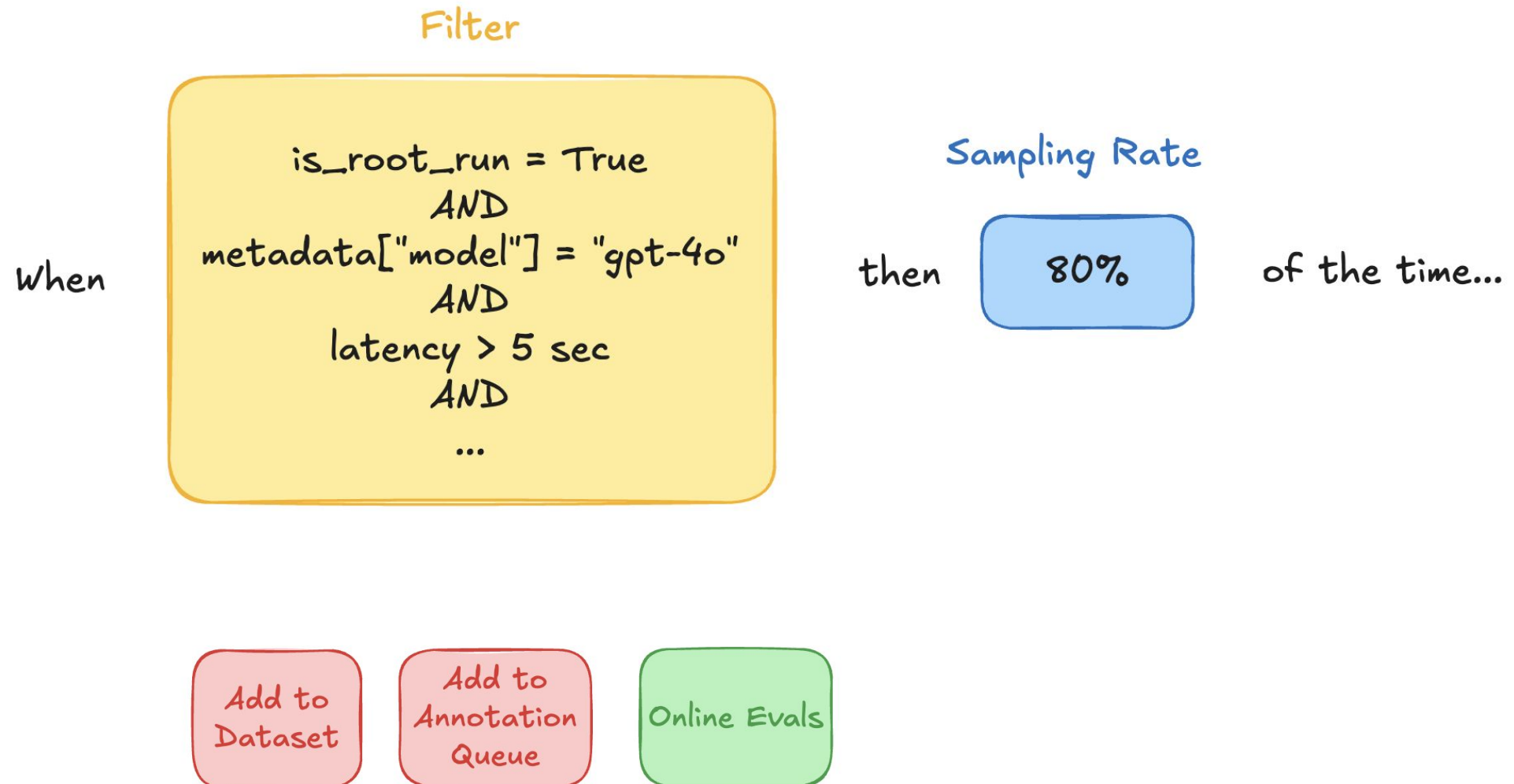
Automations

LangSmith allows you to create Rules that automatically trigger actions when in production



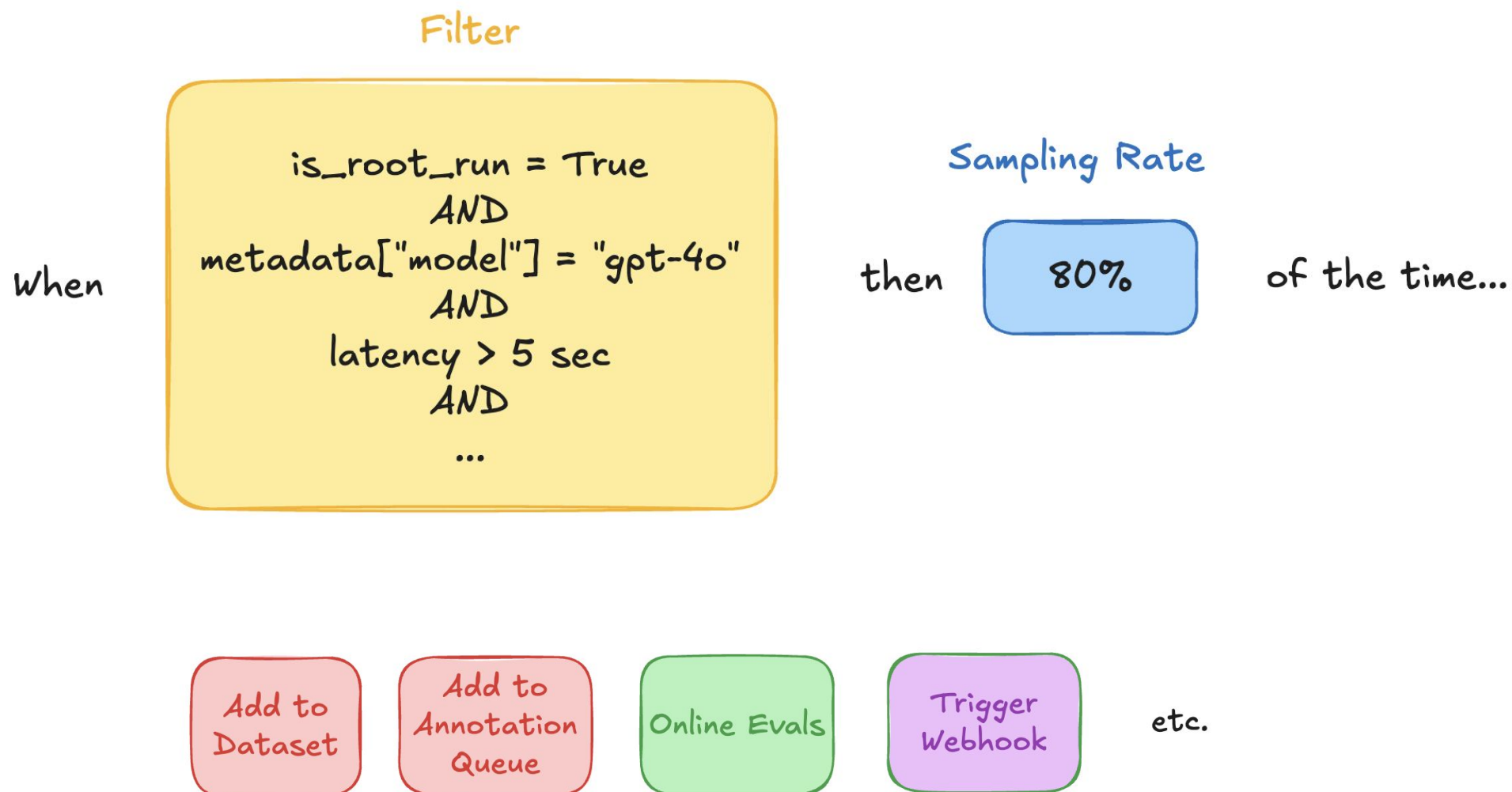
Automations

LangSmith allows you to create Rules that automatically trigger actions when in production



Automations

LangSmith allows you to create Rules that automatically trigger actions when in production





LangChain Academy

Automations

Popular Examples of Automations

Sample with low frequency from all traces, and add to an annotation queue

Filter

When

`is_root_run = True`

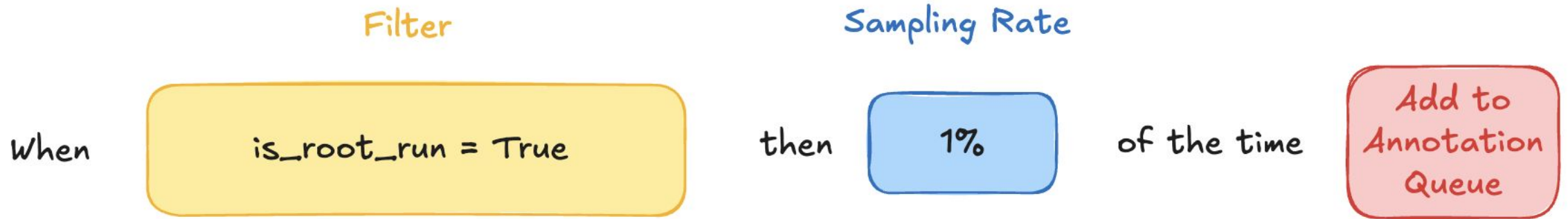
Popular Examples of Automations

Sample with low frequency from all traces, and add to an annotation queue



Popular Examples of Automations

Sample with low frequency from all traces, and add to an annotation queue



Popular Examples of Automations

Add all traces (or runs) with **negative** feedback to an annotation queue

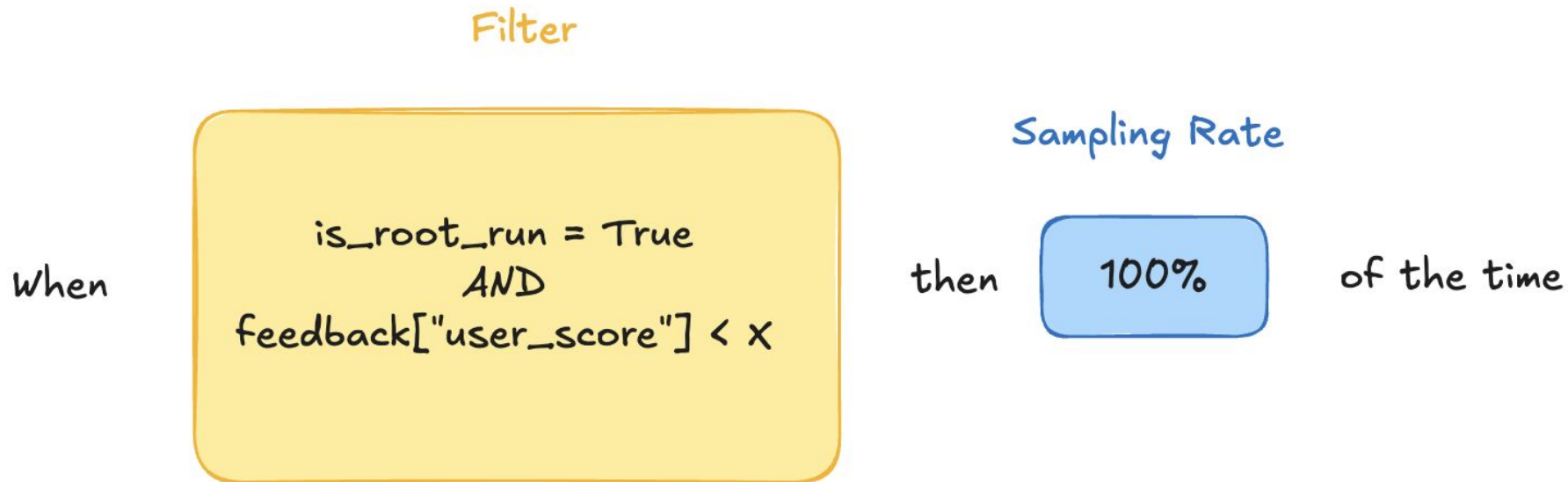
Filter

When

```
is_root_run = True  
AND  
feedback["user_score"] < x
```

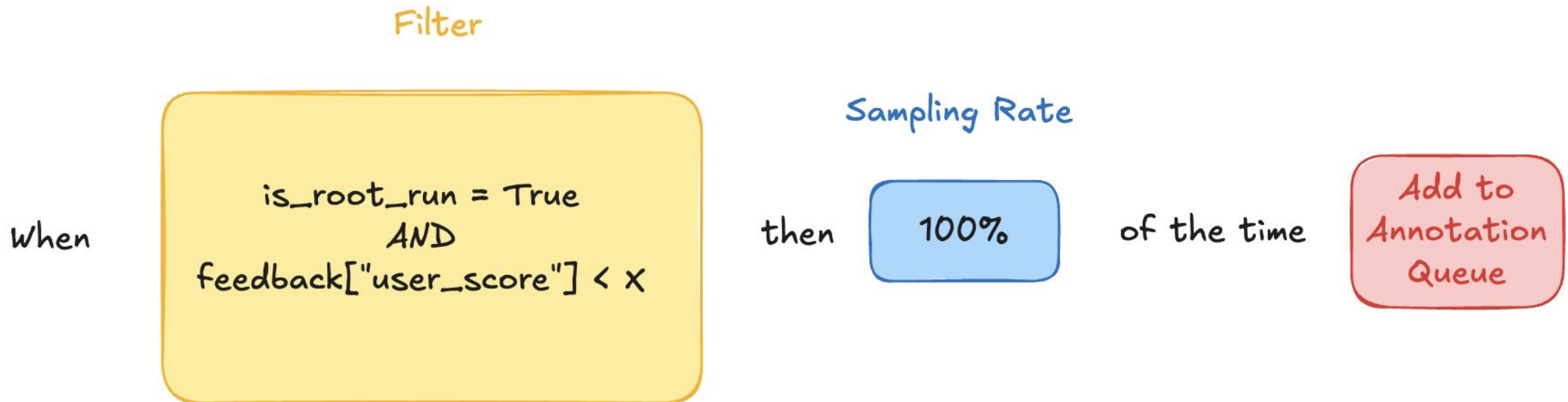
Popular Examples of Automations

Add all traces (or runs) with **negative** feedback to an annotation queue



Popular Examples of Automations

Add all traces (or runs) with **negative** feedback to an annotation queue



Popular Examples of Automations

Add all traces (or runs) with **positive** feedback to a golden dataset

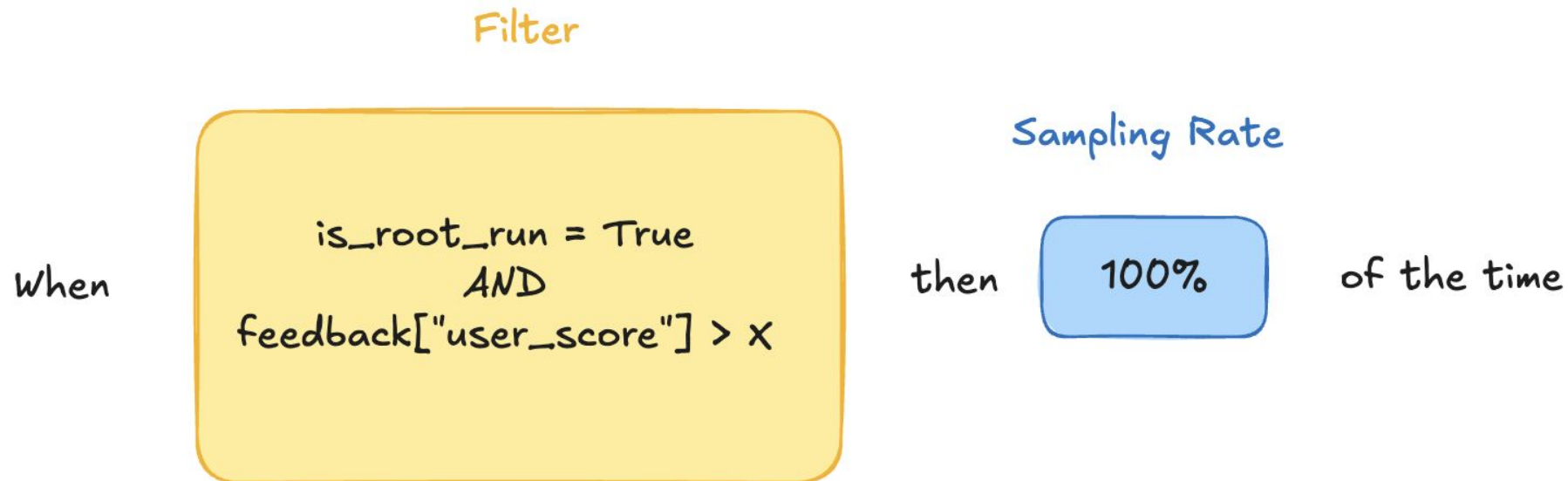
Filter

When

```
is_root_run = True  
AND  
feedback["user_score"] > x
```

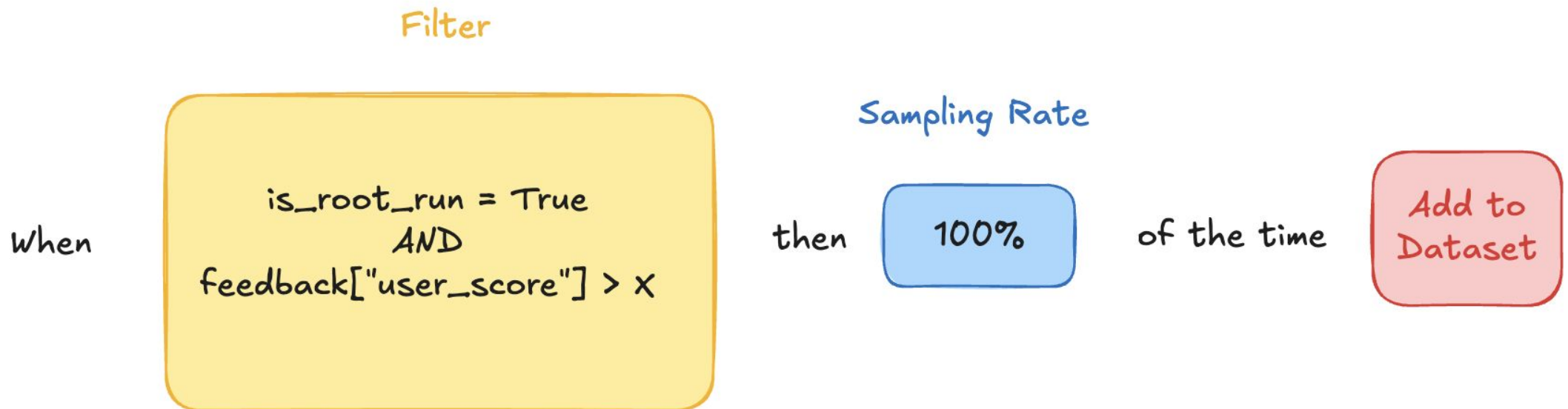

Popular Examples of Automations

Add all traces (or runs) with **positive** feedback to a golden dataset



Popular Examples of Automations

Add all traces (or runs) with **positive** feedback to a golden dataset



Popular Examples of Automations

For all traces (or runs) with an **error**, alert PagerDuty

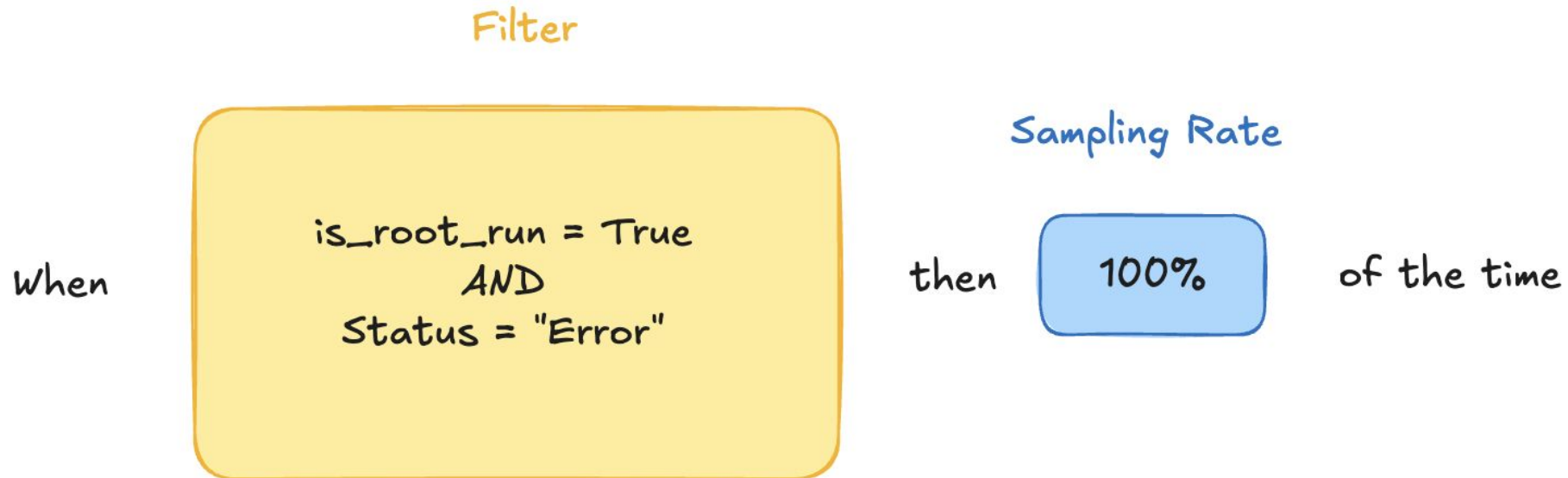
Filter

When

```
is_root_run = True  
AND  
Status = "Error"
```

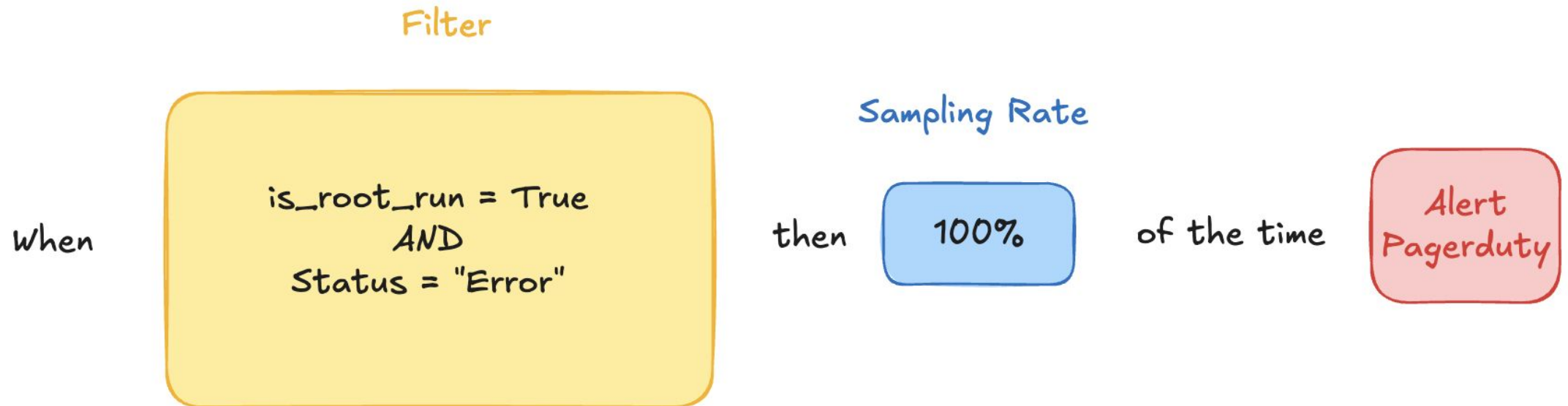
Popular Examples of Automations

For all traces (or runs) with an **error**, alert PagerDuty



Popular Examples of Automations

For all traces (or runs) with an **error**, alert PagerDuty



Popular Examples of Automations

Chain Online Evaluator Feedback, with Adding to Dataset

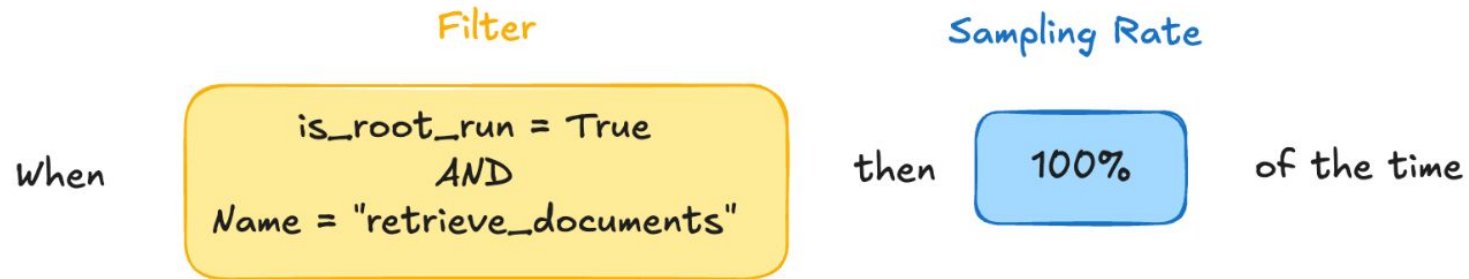
Filter

When

```
is_root_run = True  
AND  
Name = "retrieve_documents"
```

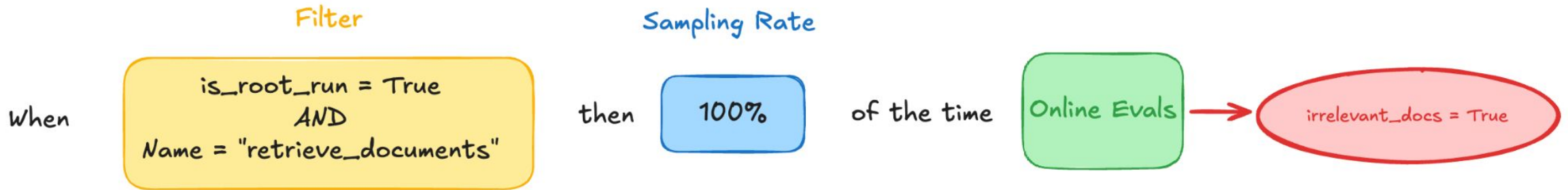
Popular Examples of Automations

Chain Online Evaluator Feedback, with Adding to Dataset



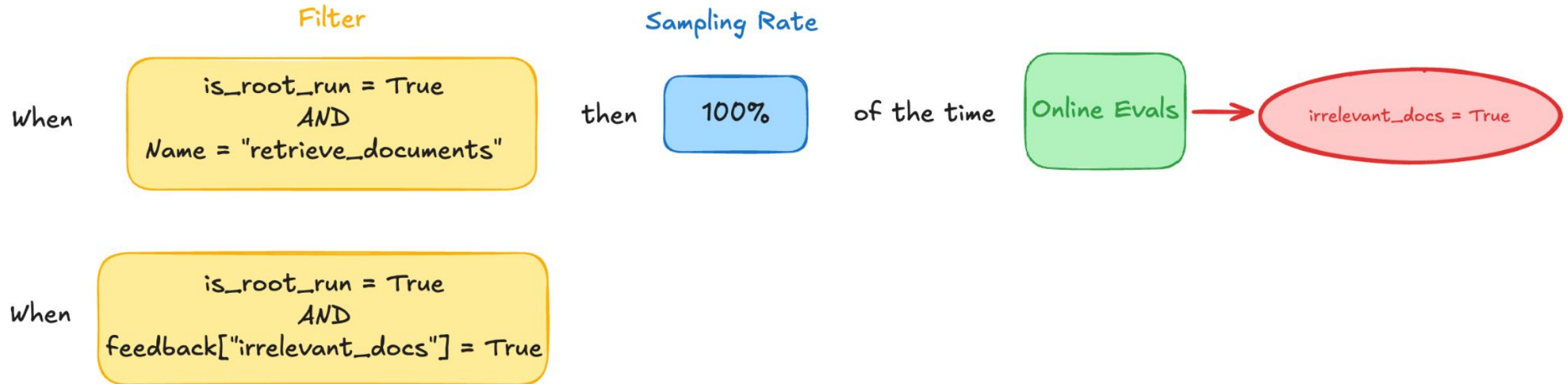
Popular Examples of Automations

Chain Online Evaluator Feedback, with Adding to Dataset



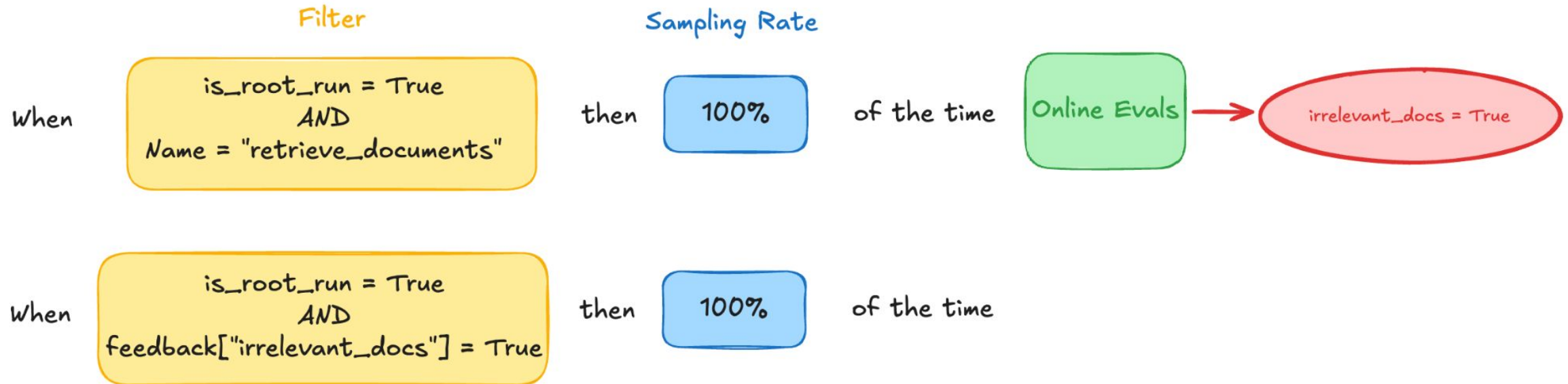
Popular Examples of Automations

Chain Online Evaluator Feedback, with Adding to Dataset



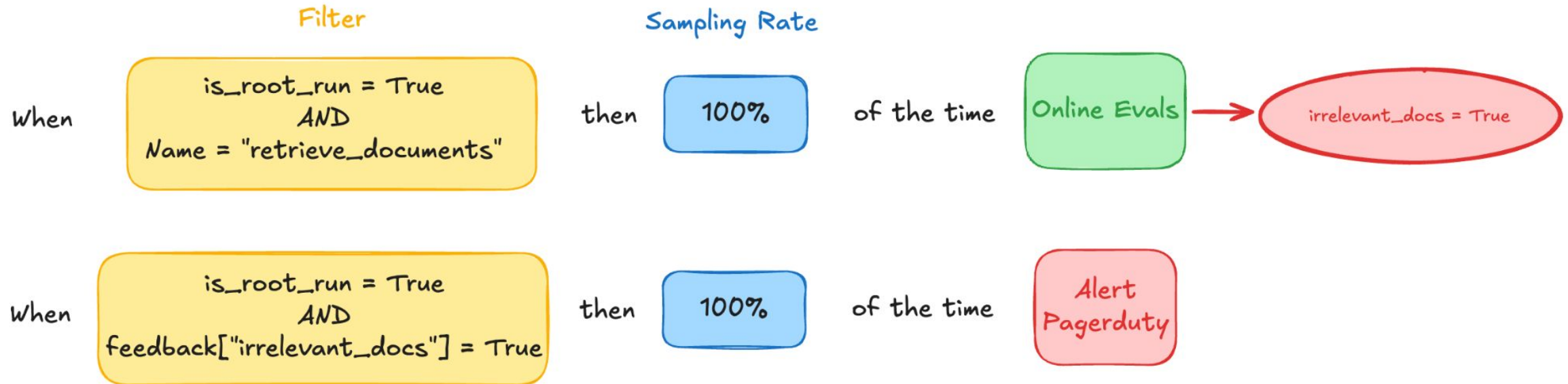
Popular Examples of Automations

Chain Online Evaluator Feedback, with Adding to Dataset



Popular Examples of Automations

Chain Online Evaluator Feedback, with Adding to Dataset





LangChain Academy

Monitoring View