Similarity ExampleSelector

The SemanticSimilarityExampleSelector selects examples based on which examples are most similar to the inputs. It does this by finding the examples with the embeddings that have the greatest cosine similarity with the inputs.

```
from langchain.prompts.example_selector import SemanticSimilarityExampleSelector
from langchain.vectorstores import Chroma
from langchain.embeddings import OpenAIEmbeddings
from langchain.prompts import FewShotPromptTemplate, PromptTemplate

example_prompt = PromptTemplate(
    input_variables=["input", "output"],
    template="Input: {input}\nOutput: {output}",
)

# These are a lot of examples of a pretend task of creating antonyms.
examples = [
    {"input": "happy", "output": "sad"},
    {"input": "tall", "output": "short"},
    {"input": "energetic", "output": "lethargic"},
    {"input": "sunny", "output": "gloomy"},
    {"input": "windy", "output": "calm"},
]
```

```
example_selector = SemanticSimilarityExampleSelector.from_examples(
    # This is the list of examples available to select from.
    examples,
    # This is the embedding class used to produce embeddings which are used to
measure semantic similarity.
    OpenAIEmbeddings(),
    # This is the VectorStore class that is used to store the embeddings and do a
similarity search over.
    Chroma,
    # This is the number of examples to produce.
    k=1
)
similar_prompt = FewShotPromptTemplate(
    # We provide an ExampleSelector instead of examples.
    example_selector=example_selector,
    example_prompt=example_prompt,
    prefix="Give the antonym of every input",
    suffix="Input: {adjective}\nOutput:",
    input_variables=["adjective"],
```

Skip to main content

```
Running Chroma using direct local API.
Using DuckDB in-memory for database. Data will be transient.
```

```
# Input is a feeling, so should select the happy/sad example
print(similar_prompt.format(adjective="worried"))
```

```
Give the antonym of every input

Input: happy
Output: sad

Input: worried
Output:
```

```
# Input is a measurement, so should select the tall/short example
print(similar_prompt.format(adjective="fat"))
```

```
Give the antonym of every input

Input: happy
Output: sad

Input: fat
Output:
```

```
# You can add new examples to the SemanticSimilarityExampleSelector as well
similar_prompt.example_selector.add_example({"input": "enthusiastic", "output":
"apathetic"})
print(similar_prompt.format(adjective="joyful"))
```

```
Give the antonym of every input

Input: happy
Output: sad

Input: joyful
Output:
```