Quick Intro to LangChain

What are

- Prompts
- Chains
- Indexes

Other important ideas I won't cover now:

- Memory
- Agents

In [30]:
```python
from langchain import PromptTemplate
from langchain.llms import OpenAI
from langchain.chains import LLMChain

template = """
I want you to act as a naming consultant for new companies.
What is a good name for a company that makes {product}?
"""

prompt = PromptTemplate(
    input_variables=["product"],
    template=template,
)

llm = OpenAI(temperature=0)

chain = LLMChain(llm=llm, prompt=prompt)

result = chain.run("colorful socks")

print(result)
```

```
Rainbow Socks Co.
```

In [31]:
```python
from demo.chains.oncall_agent.runbook_documents import DOCS

for doc in DOCS:
    print(doc)
    print("")
```

```
page_content='How to Handle a SegFault:\n\nWhen the service encounters a se
gmentation fault, the proper solution is to restart the service. \nThis can
be done by providing the following command to the VM: \n\n```\nsudo service
restart\n```\n' metadata={'error': 'segmentation fault'}

page_content='How to Handle a Timeout:\n\nWhen the service hits timeouts, w
e need to investigate whether or not there is enough space left on the devi
ce.\nTo do this, we need to run the following command:\n\n```\nsudo df\n```
\n' metadata={'error': 'timeout'}

page_content='How to Handle a Disk Error:\n\nDisk errors mean the entire VM
has reached a bad state. We need to reboot the entire VM by running this co
mmand: \n\n```\nsudo reboot\n```\n' metadata={'error': 'disk error'}
```

In [37]:

```python
from demo.chains.oncall_agent.oncall_agent_chain import OncallChain

index = create_index()
error = "segfault"

docs = index.similarity_search(error, k=1)
inputs = [{"runbook": doc.page_content, "error": error} for doc in docs]
chain = OncallChain.from_llm(llm)
result = chain.apply(inputs)

print(result)
```

```
[{'text': ' sudo service restart'}]
```

Data Generation and Auto-Evaluation

Two important insights:

- Chains can be prompted to produce synthetic data
- Chains can evaluate output from other LLMs

See Auto-evaluator for an example of evaluating question answering on documents.

Example 1: Sales Email Generation

In [3]:

```python
from demo.synthesizers.json_synthesizer.synthesize_data import synthesize

text_input = """
The goal of this language model task is to generate cold sales emails.
The input is a JSON containing a few details about a customer.

An example of the input would be:
```
{{
    "Name": "Artur Moczulski",
    "Title": "CTO",
    "Company": "Scout AI",
```
```

```
}}
```
Your prompts should follow this JSON format.
Everything between the ``` must be valid json.
Generate one such JSON input.
"""

synthesized_data = synthesize(text_input, 1)
```

```
2023-05-27 08:17:14.401 WARNING streamlit.runtime.caching.cache_data_api: N
o runtime found, using MemoryCacheStorageManager
2023-05-27 08:17:14.587
  Warning: to view this Streamlit app on a browser, run it with the followi
ng
  command:

    streamlit run /usr/local/lib/python3.11/site-packages/ipykernel_launche
r.py [ARGUMENTS]
2023-05-27 08:17:14.588 No runtime found, using MemoryCacheStorageManager
```

In [4]:
```
print(synthesized_data[0]['inputs'])
```

```
[{'Name': 'John Smith', 'Title': 'CEO', 'Company': 'ABC Inc.'}]
```

In [5]:
```
from demo.chains.sdr_agent.generate_email import generate_email

email = generate_email(synthesized_data)
```

```
2023-05-27 08:17:16.981 No runtime found, using MemoryCacheStorageManager
2023-05-27 08:17:16.983 No runtime found, using MemoryCacheStorageManager
```

In [6]:
```
print(email[0])
```
```

```
Subject: AI-powered LLM evaluation for ABC Inc.

Dear John,

I hope this email finds you well. My name is [Your Name] and I am a Sales D
evelopment Representative at [Your Company], a startup that specializes in
helping software development teams evaluate LLMs for use in their products.

I came across ABC Inc. and was impressed by your company's innovative appro
ach to [mention something specific about their business]. As a CEO, I under
stand that you are always looking for ways to improve your products and sta
y ahead of the competition.

That's where our AI-powered LLM evaluation comes in. Our technology can hel
p your development team evaluate LLMs quickly and accurately, saving you ti
me and resources. Our platform is user-friendly and can be easily integrate
d into your existing workflow.

I would love to schedule a brief call with you to discuss how our technolog
y can benefit ABC Inc. and answer any questions you may have. Please let me
know if this is something you would be interested in.

Thank you for your time and consideration. I look forward to hearing from y
ou soon.

Best regards,

[Your Name]
```

In [7]:
```python
from demo.evaluators.cold_email_binary_evaluator.evaluate_email import evalu

result = evaluate(synthesized_data[0]['inputs'], email)
```

```
2023-05-27 08:17:32.672 No runtime found, using MemoryCacheStorageManager
2023-05-27 08:17:32.674 No runtime found, using MemoryCacheStorageManager
```

In [8]:
```python
print(result)
```

```
[{'text': 'SATISFACTORY. The SDR addressed the customer by name and title,
and mentioned something specific about their business. The email is clear a
nd convincing, highlighting the benefits of the technology and offering to
schedule a call to discuss further.'}]
```

In [9]:
```python
from langchain.llms import Replicate
DOLLY_MODEL = "replicate/dolly-v2-12b:ef0e1aefc61f8e096ebe4db6b2bacc297daf2e

email2 = generate_email(synthesized_data, Replicate(model=DOLLY_MODEL))
```

In [10]:
```python
print(email2[0])
```

```
Hi Customer!,

I'm happy to talk about what NLP or Text Modeling could help [your company]
achieve. Could what NLP or Text Modeling could help [your company] achieve.
Could what NLP or Text Modeling could help [your company] achieve. Could yo
u let me know if there's someone who can assist with this topic today? I've
about what NLP or Text Modeling could help [your company] achieve. Could yo
u let me know if there's someone who can assist with this topic today? I've
included some information that might be of interest.
```

In [11]:
```python
result = evaluate(synthesized_data[0]['inputs'], email2)
```

In [12]:
```python
print(result[0]['text'])
```

```
UNSATISFACTORY

The email does not address the customer by name or title, and the content i
s repetitive and unclear. The SDR should take more time to personalize the
email and clearly explain the benefits of NLP or Text Modeling for the cust
omer's specific company.
```

In [13]:
```python
from demo.evaluators.cold_email_comparator_evaluator.evaluate_email import e

result = evaluate(synthesized_data[0]['inputs'], email, email2)
```

```
2023-05-27 08:17:47.284 No runtime found, using MemoryCacheStorageManager
2023-05-27 08:17:47.286 No runtime found, using MemoryCacheStorageManager
```

In [14]:
```python
print(result[0]['text'])
```

```
EMAIL A
EXPLANATION:
Email A is the better choice because it is clear, concise, and personalize
d. The SDR addresses the customer by name and company, and mentions somethi
ng specific about their business to show that they have done their researc
h. The email also clearly explains how the AI-powered LLM evaluation can be
nefit the customer's development team, and offers to schedule a call to dis
cuss further. In contrast, Email B is poorly written, with grammatical erro
rs and unclear language. It also lacks personalization and does not clearly
explain how the product can benefit the customer.
```

Example 2: Oncall Bot

In [16]:
```python
from demo.chains.oncall_agent.runbook_documents import create_index
from demo.chains.oncall_agent.generate_command import generate_command

error = "Error: segfault"
command = generate_command(error, index)
```

In [17]:
```python
print(command)
```

```
[{'text': ' sudo service restart'}]
```

```
In [20]: from demo.evaluators.oncall_action_evaluator.evaluate_action import evalaute

         result = evalaute(error, command, index)
```

```
[{'text': ' SUCCESS\n\nThe Site Reliability Engineer correctly followed the
runbook and provided the correct command to restart the service. Therefore,
they should be given a grade of SUCCESS.'}]
```

Using HuggingFace

```
In [22]: from langchain.llms import HuggingFaceEndpoint
         endpoint_url = (
             "endpoint_url_here"
         )
         hf = HuggingFaceEndpoint(
             endpoint_url=endpoint_url,
             huggingfacehub_api_token="api_key_here"
         )

         hf.task = "text2text-generation"
```