langgraph

August 27, 2024

[1]:
```
!pip install langgraph langsmith
```

```
Collecting langgraph
  Downloading langgraph-0.2.14-py3-none-any.whl.metadata (13 kB)
Collecting langsmith
  Downloading langsmith-0.1.105-py3-none-any.whl.metadata (13 kB)
Collecting langchain-core<0.3,>=0.2.27 (from langgraph)
  Downloading langchain_core-0.2.35-py3-none-any.whl.metadata (6.2 kB)
Collecting langgraph-checkpoint<2.0.0,>=1.0.2 (from langgraph)
  Downloading langgraph_checkpoint-1.0.6-py3-none-any.whl.metadata (4.5 kB)
Collecting httpx<1,>=0.23.0 (from langsmith)
  Downloading httpx-0.27.2-py3-none-any.whl.metadata (7.1 kB)
Collecting orjson<4.0.0,>=3.9.14 (from langsmith)
  Downloading orjson-3.10.7-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_
64.whl.metadata (50 kB)
                           50.4/50.4 kB
2.2 MB/s eta 0:00:00
Requirement already satisfied: pydantic<3,>=1 in
/usr/local/lib/python3.10/dist-packages (from langsmith) (2.8.2)
Requirement already satisfied: requests<3,>=2 in /usr/local/lib/python3.10/dist-
packages (from langsmith) (2.32.3)
Requirement already satisfied: anyio in /usr/local/lib/python3.10/dist-packages
(from httpx<1,>=0.23.0->langsmith) (3.7.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-
packages (from httpx<1,>=0.23.0->langsmith) (2024.7.4)
Collecting httpcore==1.* (from httpx<1,>=0.23.0->langsmith)
  Downloading httpcore-1.0.5-py3-none-any.whl.metadata (20 kB)
Requirement already satisfied: idna in /usr/local/lib/python3.10/dist-packages
(from httpx<1,>=0.23.0->langsmith) (3.7)
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-
packages (from httpx<1,>=0.23.0->langsmith) (1.3.1)
Collecting h11<0.15,>=0.13 (from httpcore==1.*->httpx<1,>=0.23.0->langsmith)
  Downloading h11-0.14.0-py3-none-any.whl.metadata (8.2 kB)
Requirement already satisfied: PyYAML>=5.3 in /usr/local/lib/python3.10/dist-
packages (from langchain-core<0.3,>=0.2.27->langgraph) (6.0.2)
Collecting jsonpatch<2.0,>=1.33 (from langchain-core<0.3,>=0.2.27->langgraph)
  Downloading jsonpatch-1.33-py2.py3-none-any.whl.metadata (3.0 kB)
Requirement already satisfied: packaging<25,>=23.2 in
```

```
/usr/local/lib/python3.10/dist-packages (from langchain-
core<0.3,>=0.2.27->langgraph) (24.1)
Collecting tenacity!=8.4.0,<9.0.0,>=8.1.0 (from langchain-
core<0.3,>=0.2.27->langgraph)
  Downloading tenacity-8.5.0-py3-none-any.whl.metadata (1.2 kB)
Requirement already satisfied: typing-extensions>=4.7 in
/usr/local/lib/python3.10/dist-packages (from langchain-
core<0.3,>=0.2.27->langgraph) (4.12.2)
Requirement already satisfied: annotated-types>=0.4.0 in
/usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langsmith) (0.7.0)
Requirement already satisfied: pydantic-core==2.20.1 in
/usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langsmith)
(2.20.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langsmith) (3.3.2)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langsmith) (2.0.7)
Collecting jsonpointer>=1.9 (from jsonpatch<2.0,>=1.33->langchain-
core<0.3,>=0.2.27->langgraph)
  Downloading jsonpointer-3.0.0-py2.py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-
packages (from anyio->httpx<1,>=0.23.0->langsmith) (1.2.2)
Downloading langgraph-0.2.14-py3-none-any.whl (87 kB)
                         87.7/87.7 kB
5.6 MB/s eta 0:00:00
Downloading langsmith-0.1.105-py3-none-any.whl (150 kB)
                         150.5/150.5 kB
9.4 MB/s eta 0:00:00
Downloading httpx-0.27.2-py3-none-any.whl (76 kB)
                         76.4/76.4 kB
5.0 MB/s eta 0:00:00
Downloading httpcore-1.0.5-py3-none-any.whl (77 kB)
                         77.9/77.9 kB
4.7 MB/s eta 0:00:00
Downloading langchain_core-0.2.35-py3-none-any.whl (394 kB)
                         394.9/394.9 kB
12.8 MB/s eta 0:00:00
Downloading langgraph_checkpoint-1.0.6-py3-none-any.whl (15 kB)
Downloading
orjson-3.10.7-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (141
kB)
                         141.9/141.9 kB
8.6 MB/s eta 0:00:00
Downloading jsonpatch-1.33-py2.py3-none-any.whl (12 kB)
Downloading tenacity-8.5.0-py3-none-any.whl (28 kB)
Downloading h11-0.14.0-py3-none-any.whl (58 kB)
                         58.3/58.3 kB
3.8 MB/s eta 0:00:00
```

```
Downloading jsonpointer-3.0.0-py2.py3-none-any.whl (7.6 kB)
Installing collected packages: tenacity, orjson, jsonpointer, h11, jsonpatch,
httpcore, httpx, langsmith, langchain-core, langgraph-checkpoint, langgraph
  Attempting uninstall: tenacity
    Found existing installation: tenacity 9.0.0
    Uninstalling tenacity-9.0.0:
      Successfully uninstalled tenacity-9.0.0
Successfully installed h11-0.14.0 httpcore-1.0.5 httpx-0.27.2 jsonpatch-1.33
jsonpointer-3.0.0 langchain-core-0.2.35 langgraph-0.2.14 langgraph-
checkpoint-1.0.6 langsmith-0.1.105 orjson-3.10.7 tenacity-8.5.0
```

[2]:
```
!pip install langchain langchain_groq langchain_community
```

```
Collecting langchain
  Downloading langchain-0.2.14-py3-none-any.whl.metadata (7.1 kB)
Collecting langchain_groq
  Downloading langchain_groq-0.1.9-py3-none-any.whl.metadata (2.9 kB)
Collecting langchain_community
  Downloading langchain_community-0.2.12-py3-none-any.whl.metadata (2.7 kB)
Requirement already satisfied: PyYAML>=5.3 in /usr/local/lib/python3.10/dist-
packages (from langchain) (6.0.2)
Requirement already satisfied: SQLAlchemy<3,>=1.4 in
/usr/local/lib/python3.10/dist-packages (from langchain) (2.0.32)
Requirement already satisfied: aiohttp<4.0.0,>=3.8.3 in
/usr/local/lib/python3.10/dist-packages (from langchain) (3.10.5)
Requirement already satisfied: async-timeout<5.0.0,>=4.0.0 in
/usr/local/lib/python3.10/dist-packages (from langchain) (4.0.3)
Requirement already satisfied: langchain-core<0.3.0,>=0.2.32 in
/usr/local/lib/python3.10/dist-packages (from langchain) (0.2.35)
Collecting langchain-text-splitters<0.3.0,>=0.2.0 (from langchain)
  Downloading langchain_text_splitters-0.2.2-py3-none-any.whl.metadata (2.1 kB)
Requirement already satisfied: langsmith<0.2.0,>=0.1.17 in
/usr/local/lib/python3.10/dist-packages (from langchain) (0.1.105)
Requirement already satisfied: numpy<2,>=1 in /usr/local/lib/python3.10/dist-
packages (from langchain) (1.26.4)
Requirement already satisfied: pydantic<3,>=1 in /usr/local/lib/python3.10/dist-
packages (from langchain) (2.8.2)
Requirement already satisfied: requests<3,>=2 in /usr/local/lib/python3.10/dist-
packages (from langchain) (2.32.3)
Requirement already satisfied: tenacity!=8.4.0,<9.0.0,>=8.1.0 in
/usr/local/lib/python3.10/dist-packages (from langchain) (8.5.0)
Collecting groq<1,>=0.4.1 (from langchain_groq)
  Downloading groq-0.10.0-py3-none-any.whl.metadata (13 kB)
Collecting dataclasses-json<0.7,>=0.5.7 (from langchain_community)
  Downloading dataclasses_json-0.6.7-py3-none-any.whl.metadata (25 kB)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain)
(2.4.0)
```

```
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain)
(1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-
packages (from aiohttp<4.0.0,>=3.8.3->langchain) (24.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain)
(1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain)
(6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-
packages (from aiohttp<4.0.0,>=3.8.3->langchain) (1.9.4)
Collecting marshmallow<4.0.0,>=3.18.0 (from dataclasses-
json<0.7,>=0.5.7->langchain_community)
  Downloading marshmallow-3.22.0-py3-none-any.whl.metadata (7.2 kB)
Collecting typing-inspect<1,>=0.4.0 (from dataclasses-
json<0.7,>=0.5.7->langchain_community)
  Downloading typing_inspect-0.9.0-py3-none-any.whl.metadata (1.5 kB)
Requirement already satisfied: anyio<5,>=3.5.0 in
/usr/local/lib/python3.10/dist-packages (from groq<1,>=0.4.1->langchain_groq)
(3.7.1)
Requirement already satisfied: distro<2,>=1.7.0 in /usr/lib/python3/dist-
packages (from groq<1,>=0.4.1->langchain_groq) (1.7.0)
Requirement already satisfied: httpx<1,>=0.23.0 in
/usr/local/lib/python3.10/dist-packages (from groq<1,>=0.4.1->langchain_groq)
(0.27.2)
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-
packages (from groq<1,>=0.4.1->langchain_groq) (1.3.1)
Requirement already satisfied: typing-extensions<5,>=4.7 in
/usr/local/lib/python3.10/dist-packages (from groq<1,>=0.4.1->langchain_groq)
(4.12.2)
Requirement already satisfied: jsonpatch<2.0,>=1.33 in
/usr/local/lib/python3.10/dist-packages (from langchain-
core<0.3.0,>=0.2.32->langchain) (1.33)
Requirement already satisfied: packaging<25,>=23.2 in
/usr/local/lib/python3.10/dist-packages (from langchain-
core<0.3.0,>=0.2.32->langchain) (24.1)
Requirement already satisfied: orjson<4.0.0,>=3.9.14 in
/usr/local/lib/python3.10/dist-packages (from
langsmith<0.2.0,>=0.1.17->langchain) (3.10.7)
Requirement already satisfied: annotated-types>=0.4.0 in
/usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain) (0.7.0)
Requirement already satisfied: pydantic-core==2.20.1 in
/usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain)
(2.20.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain) (3.3.2)
```

```
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests<3,>=2->langchain) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain)
(2024.7.4)
Requirement already satisfied: greenlet!=0.4.17 in
/usr/local/lib/python3.10/dist-packages (from SQLAlchemy<3,>=1.4->langchain)
(3.0.3)
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-
packages (from anyio<5,>=3.5.0->groq<1,>=0.4.1->langchain_groq) (1.2.2)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.10/dist-
packages (from httpx<1,>=0.23.0->groq<1,>=0.4.1->langchain_groq) (1.0.5)
Requirement already satisfied: h11<0.15,>=0.13 in
/usr/local/lib/python3.10/dist-packages (from
httpcore==1.*->httpx<1,>=0.23.0->groq<1,>=0.4.1->langchain_groq) (0.14.0)
Requirement already satisfied: jsonpointer>=1.9 in
/usr/local/lib/python3.10/dist-packages (from jsonpatch<2.0,>=1.33->langchain-
core<0.3.0,>=0.2.32->langchain) (3.0.0)
Collecting mypy-extensions>=0.3.0 (from typing-inspect<1,>=0.4.0->dataclasses-
json<0.7,>=0.5.7->langchain_community)
  Downloading mypy_extensions-1.0.0-py3-none-any.whl.metadata (1.1 kB)
Downloading langchain-0.2.14-py3-none-any.whl (997 kB)
                         997.8/997.8 kB
13.7 MB/s eta 0:00:00
Downloading langchain_groq-0.1.9-py3-none-any.whl (14 kB)
Downloading langchain_community-0.2.12-py3-none-any.whl (2.3 MB)
                         2.3/2.3 MB
54.4 MB/s eta 0:00:00
Downloading dataclasses_json-0.6.7-py3-none-any.whl (28 kB)
Downloading groq-0.10.0-py3-none-any.whl (106 kB)
                         106.3/106.3 kB
7.0 MB/s eta 0:00:00
Downloading langchain_text_splitters-0.2.2-py3-none-any.whl (25 kB)
Downloading marshmallow-3.22.0-py3-none-any.whl (49 kB)
                         49.3/49.3 kB
3.3 MB/s eta 0:00:00
Downloading typing_inspect-0.9.0-py3-none-any.whl (8.8 kB)
Downloading mypy_extensions-1.0.0-py3-none-any.whl (4.7 kB)
Installing collected packages: mypy-extensions, marshmallow, typing-inspect,
groq, dataclasses-json, langchain-text-splitters, langchain_groq, langchain,
langchain_community
Successfully installed dataclasses-json-0.6.7 groq-0.10.0 langchain-0.2.14
langchain-text-splitters-0.2.2 langchain_community-0.2.12 langchain_groq-0.1.9
marshmallow-3.22.0 mypy-extensions-1.0.0 typing-inspect-0.9.0
```

[10]:
```
from google.colab import userdata

GROQ_API= userdata.get("GROQ_API")
LANGSMITH_API_KEY= userdata.get("LANGSMITH_API_KEY")
```

[11]:
```
import os
os.environ["LANGCHAIN_API_KEY"] = LANGSMITH_API_KEY
os.environ["LANGCHAIN_TRACING_V2"]="true"
os.environ["LANGCHAIN_PROJECT"]="simple_chatbot"
```

[12]:
```
from langchain_groq import ChatGroq

llm=ChatGroq(groq_api_key=GROQ_API,model_name="Gemma2-9b-It")
llm
```

[12]:
```
ChatGroq(client=<groq.resources.chat.completions.Completions object at
0x7e0cad4c1a50>, async_client=<groq.resources.chat.completions.AsyncCompletions
object at 0x7e0cad4ff340>, model_name='Gemma2-9b-It',
groq_api_key=SecretStr('**********'))
```

1 Chatbot using LangGraph

[26]:
```
from typing import Annotated
from typing_extensions import TypedDict
from langgraph.graph import StateGraph, START, END
from langgraph.graph.message import add_messages
```

[27]:
```
class State(TypedDict):
  # Messages have the type "list". The `add_messages` function
    # in the annotation defines how this state key should be updated
    # (in this case, it appends messages to the list, rather than overwriting␣
 ↪them)
  messages:Annotated[list,add_messages]

graph_builder=StateGraph(State)
```

[28]:
```
def chatbot(state:State):
  return {"messages":llm.invoke(state['messages'])}
```

[29]:
```
graph_builder.add_node("chatbot", chatbot)
```

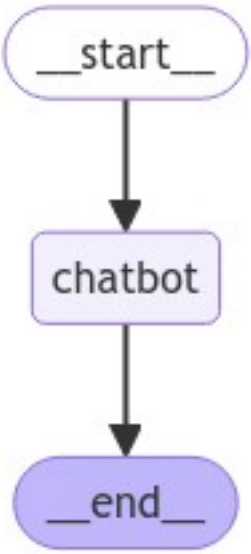
[18]:
```
graph_builder
```

[18]:
```
<langgraph.graph.state.StateGraph at 0x7e0cad28ca60>
```

[30]:
```
graph_builder.add_edge(START,"chatbot")
graph_builder.add_edge("chatbot",END)

graph = graph_builder.compile()
```

[31]:
```
from IPython.display import Image, display
try:
  display(Image(graph.get_graph().draw_mermaid_png()))
except Exception:
  pass
```



[32]:
```
while True:
  user_input=input("User: ")
  if user_input.lower() in ["quit","q"]:
    print("Good Bye")
    break
  for event in graph.stream({'messages':("user",user_input)}):
    print(event.values())
    for value in event.values():
      print(value['messages'])
      print("Assistant:",value["messages"].content)
```

```
User: hello
dict_values([{'messages': AIMessage(content='Hello!    How can I help you today?
', response_metadata={'token_usage': {'completion_tokens': 14, 'prompt_tokens':
10, 'total_tokens': 24, 'completion_time': 0.025454545, 'prompt_time': 3.7e-07,
'queue_time': 0.014253309, 'total_time': 0.025454915}, 'model_name':
'Gemma2-9b-It', 'system_fingerprint': 'fp_10c08bf97d', 'finish_reason': 'stop',
```

'logprobs': None}, id='run-58a69fbd-a344-4b5c-b0f7-8bdaadaf5da8-0',
usage_metadata={'input_tokens': 10, 'output_tokens': 14, 'total_tokens': 24})}])
content='Hello! How can I help you today? '
response_metadata={'token_usage': {'completion_tokens': 14, 'prompt_tokens': 10,
'total_tokens': 24, 'completion_time': 0.025454545, 'prompt_time': 3.7e-07,
'queue_time': 0.014253309, 'total_time': 0.025454915}, 'model_name':
'Gemma2-9b-It', 'system_fingerprint': 'fp_10c08bf97d', 'finish_reason': 'stop',
'logprobs': None} id='run-58a69fbd-a344-4b5c-b0f7-8bdaadaf5da8-0'
usage_metadata={'input_tokens': 10, 'output_tokens': 14, 'total_tokens': 24}
Assistant: Hello! How can I help you today?
User: Explain Machine Learning
dict_values([{'messages': AIMessage(content='Imagine you\'re teaching a dog a
new trick. You show them what to do, reward them when they get it right, and
correct them when they make mistakes. Over time, the dog learns the trick
through this process of example and feedback.\n\nMachine learning is similar,
but instead of a dog, we have a computer program. \n\n**In essence, machine
learning is a type of artificial intelligence (AI) that allows computers to
learn from data without being explicitly programmed.** \n\nHere\'s a
breakdown:\n\n**1. Data is Key:**\n\nMachine learning algorithms are "trained"
on large datasets. This data can be anything: images, text, numbers, sounds,
etc. The more data, the better the algorithm learns.\n\n**2. Finding
Patterns:**\n\nThe algorithm analyzes the data, looking for patterns and
relationships. It\'s like the dog figuring out the connection between your hand
gesture and the desired action.\n\n**3. Making Predictions:**\n\nOnce trained,
the algorithm can use these learned patterns to make predictions on new, unseen
data. For example, it might predict whether an email is spam or not, or identify
objects in an image.\n\n**Types of Machine Learning:**\n\n* **Supervised
Learning:** The algorithm is given labeled data (e.g., images labeled "cat" or
"dog"). It learns to map the input data to the correct output label.\n\n*
Unsupervised Learning: The algorithm is given unlabeled data and must find
patterns and structure on its own (e.g., clustering customers with similar
buying habits).\n\n* **Reinforcement Learning:** The algorithm learns through
trial and error, receiving rewards for correct actions and penalties for
incorrect ones (e.g., training a game-playing AI).\n\n**Examples of Machine
Learning in Action:**\n\n* **Recommendation systems:** Netflix suggesting movies
you might like\n* **Spam filters:** Identifying unwanted emails\n* **Fraud
detection:** flagging suspicious transactions\n* **Self-driving cars:**
Navigating roads and avoiding obstacles\n* **Medical diagnosis:** Assisting
doctors in identifying diseases\n\n**Machine learning is a powerful tool with
the potential to revolutionize many industries. It\'s a rapidly evolving field
with new algorithms and applications constantly being developed.**\n\n\nLet me
know if you have any other questions!\n', response_metadata={'token_usage':
{'completion_tokens': 471, 'prompt_tokens': 12, 'total_tokens': 483,
'completion_time': 0.856363636, 'prompt_time': 0.000173739, 'queue_time':
0.5764447260000001, 'total_time': 0.856537375}, 'model_name': 'Gemma2-9b-It',
'system_fingerprint': 'fp_10c08bf97d', 'finish_reason': 'stop', 'logprobs':
None}, id='run-bc92ce3d-c8ce-44be-baf8-8cbc0f046a87-0',
usage_metadata={'input_tokens': 12, 'output_tokens': 471, 'total_tokens':

483})}])
content='Imagine you\'re teaching a dog a new trick. You show them what to do, reward them when they get it right, and correct them when they make mistakes. Over time, the dog learns the trick through this process of example and feedback.\n\nMachine learning is similar, but instead of a dog, we have a computer program. \n\n**In essence, machine learning is a type of artificial intelligence (AI) that allows computers to learn from data without being explicitly programmed.** \n\nHere\'s a breakdown:\n\n**1. Data is Key:**\n\nMachine learning algorithms are "trained" on large datasets. This data can be anything: images, text, numbers, sounds, etc. The more data, the better the algorithm learns.\n\n**2. Finding Patterns:**\n\nThe algorithm analyzes the data, looking for patterns and relationships. It\'s like the dog figuring out the connection between your hand gesture and the desired action.\n\n**3. Making Predictions:**\n\nOnce trained, the algorithm can use these learned patterns to make predictions on new, unseen data. For example, it might predict whether an email is spam or not, or identify objects in an image.\n\n**Types of Machine Learning:**\n\n* **Supervised Learning:** The algorithm is given labeled data (e.g., images labeled "cat" or "dog"). It learns to map the input data to the correct output label.\n\n* **Unsupervised Learning:** The algorithm is given unlabeled data and must find patterns and structure on its own (e.g., clustering customers with similar buying habits).\n\n* **Reinforcement Learning:** The algorithm learns through trial and error, receiving rewards for correct actions and penalties for incorrect ones (e.g., training a game-playing AI).\n\n**Examples of Machine Learning in Action:**\n\n* **Recommendation systems:** Netflix suggesting movies you might like\n* **Spam filters:** Identifying unwanted emails\n* **Fraud detection:** flagging suspicious transactions\n* **Self-driving cars:** Navigating roads and avoiding obstacles\n* **Medical diagnosis:** Assisting doctors in identifying diseases\n\n**Machine learning is a powerful tool with the potential to revolutionize many industries. It\'s a rapidly evolving field with new algorithms and applications constantly being developed.**\n\n\nLet me know if you have any other questions!\n' response_metadata={'token_usage': {'completion_tokens': 471, 'prompt_tokens': 12, 'total_tokens': 483, 'completion_time': 0.856363636, 'prompt_time': 0.000173739, 'queue_time': 0.5764447260000001, 'total_time': 0.856537375}, 'model_name': 'Gemma2-9b-It', 'system_fingerprint': 'fp_10c08bf97d', 'finish_reason': 'stop', 'logprobs': None} id='run-bc92ce3d-c8ce-44be-baf8-8cbc0f046a87-0' usage_metadata={'input_tokens': 12, 'output_tokens': 471, 'total_tokens': 483}
Assistant: Imagine you're teaching a dog a new trick. You show them what to do, reward them when they get it right, and correct them when they make mistakes. Over time, the dog learns the trick through this process of example and feedback.

Machine learning is similar, but instead of a dog, we have a computer program.

In essence, machine learning is a type of artificial intelligence (AI) that allows computers to learn from data without being explicitly programmed.

Here's a breakdown:

1. Data is Key:

Machine learning algorithms are "trained" on large datasets. This data can be anything: images, text, numbers, sounds, etc. The more data, the better the algorithm learns.

2. Finding Patterns:

The algorithm analyzes the data, looking for patterns and relationships. It's like the dog figuring out the connection between your hand gesture and the desired action.

3. Making Predictions:

Once trained, the algorithm can use these learned patterns to make predictions on new, unseen data. For example, it might predict whether an email is spam or not, or identify objects in an image.

Types of Machine Learning:

* **Supervised Learning:** The algorithm is given labeled data (e.g., images labeled "cat" or "dog"). It learns to map the input data to the correct output label.

* **Unsupervised Learning:** The algorithm is given unlabeled data and must find patterns and structure on its own (e.g., clustering customers with similar buying habits).

* **Reinforcement Learning:** The algorithm learns through trial and error, receiving rewards for correct actions and penalties for incorrect ones (e.g., training a game-playing AI).

Examples of Machine Learning in Action:

* **Recommendation systems:** Netflix suggesting movies you might like
* **Spam filters:** Identifying unwanted emails
* **Fraud detection:** flagging suspicious transactions
* **Self-driving cars:** Navigating roads and avoiding obstacles
* **Medical diagnosis:** Assisting doctors in identifying diseases

Machine learning is a powerful tool with the potential to revolutionize many industries. It's a rapidly evolving field with new algorithms and applications constantly being developed.

Let me know if you have any other questions!

```
User: what is AI?
dict_values([{'messages': AIMessage(content="Artificial intelligence (AI) is a
broad field of computer science focused on creating machines capable of
performing tasks that typically require human intelligence. \n\nHere's a
breakdown:\n\n**What AI aims to do:**\n\n* **Learn from data:** AI systems can
analyze vast amounts of data and identify patterns, trends, and insights.\n*
**Reason and solve problems:** AI can use logic and rules to solve problems,
make decisions, and find solutions.\n* **Understand and respond to language:**
AI can process and understand natural language, allowing for conversations and
text generation.\n* **Recognize patterns and images:** AI can identify objects,
faces, and other patterns in images and videos.\n* **Adapt and improve:** AI
systems can learn from their experiences and improve their performance over
time.\n\n**Types of AI:**\n\n* **Narrow or Weak AI:** Designed to perform a
specific task, like playing chess or recommending products. Most AI today falls
into this category.\n* **General or Strong AI:** Hypothetical AI with human-
level intelligence, capable of learning and performing any intellectual task a
human can.\n* **Super AI:**  Hypothetical AI that surpasses human intelligence
in all aspects.\n\n**Examples of AI in action:**\n\n* **Virtual assistants:**
Siri, Alexa, and Google Assistant\n* **Self-driving cars:** Tesla Autopilot,
Waymo\n* **Recommendation systems:** Netflix, Spotify\n* **Medical diagnosis:**
AI-powered tools to assist doctors in detecting diseases\n* **Fraud detection:**
AI algorithms to identify suspicious transactions\n\n**It's important to
note:**\n\nAI is a rapidly evolving field with both exciting potential and
ethical considerations. It's crucial to have open discussions and responsible
development to ensure AI benefits society as a whole.\n",
response_metadata={'token_usage': {'completion_tokens': 362, 'prompt_tokens':
13, 'total_tokens': 375, 'completion_time': 0.658181818, 'prompt_time':
7.507e-05, 'queue_time': 0.013005699, 'total_time': 0.658256888}, 'model_name':
'Gemma2-9b-It', 'system_fingerprint': 'fp_10c08bf97d', 'finish_reason': 'stop',
'logprobs': None}, id='run-64e1a8e8-0dde-4b97-9919-1fd41218ddbf-0',
usage_metadata={'input_tokens': 13, 'output_tokens': 362, 'total_tokens':
375})}])
content="Artificial intelligence (AI) is a broad field of computer science
focused on creating machines capable of performing tasks that typically require
human intelligence. \n\nHere's a breakdown:\n\n**What AI aims to do:**\n\n*
**Learn from data:** AI systems can analyze vast amounts of data and identify
patterns, trends, and insights.\n* **Reason and solve problems:** AI can use
logic and rules to solve problems, make decisions, and find solutions.\n*
**Understand and respond to language:** AI can process and understand natural
language, allowing for conversations and text generation.\n* **Recognize
patterns and images:** AI can identify objects, faces, and other patterns in
images and videos.\n* **Adapt and improve:** AI systems can learn from their
experiences and improve their performance over time.\n\n**Types of AI:**\n\n*
**Narrow or Weak AI:** Designed to perform a specific task, like playing chess
or recommending products. Most AI today falls into this category.\n* **General
or Strong AI:** Hypothetical AI with human-level intelligence, capable of
learning and performing any intellectual task a human can.\n* **Super AI:**
```

Hypothetical AI that surpasses human intelligence in all aspects.\n\n**Examples of AI in action:**\n\n* **Virtual assistants:** Siri, Alexa, and Google Assistant\n* **Self-driving cars:** Tesla Autopilot, Waymo\n* **Recommendation systems:** Netflix, Spotify\n* **Medical diagnosis:** AI-powered tools to assist doctors in detecting diseases\n* **Fraud detection:** AI algorithms to identify suspicious transactions\n\n**It's important to note:**\n\nAI is a rapidly evolving field with both exciting potential and ethical considerations. It's crucial to have open discussions and responsible development to ensure AI benefits society as a whole.\n" response_metadata={'token_usage': {'completion_tokens': 362, 'prompt_tokens': 13, 'total_tokens': 375, 'completion_time': 0.658181818, 'prompt_time': 7.507e-05, 'queue_time': 0.013005699, 'total_time': 0.658256888}, 'model_name': 'Gemma2-9b-It', 'system_fingerprint': 'fp_10c08bf97d', 'finish_reason': 'stop', 'logprobs': None} id='run-64e1a8e8-0dde-4b97-9919-1fd41218ddbf-0' usage_metadata={'input_tokens': 13, 'output_tokens': 362, 'total_tokens': 375}
Assistant: Artificial intelligence (AI) is a broad field of computer science focused on creating machines capable of performing tasks that typically require human intelligence.

Here's a breakdown:

What AI aims to do:

* **Learn from data:** AI systems can analyze vast amounts of data and identify patterns, trends, and insights.
* **Reason and solve problems:** AI can use logic and rules to solve problems, make decisions, and find solutions.
* **Understand and respond to language:** AI can process and understand natural language, allowing for conversations and text generation.
* **Recognize patterns and images:** AI can identify objects, faces, and other patterns in images and videos.
* **Adapt and improve:** AI systems can learn from their experiences and improve their performance over time.

Types of AI:

* **Narrow or Weak AI:** Designed to perform a specific task, like playing chess or recommending products. Most AI today falls into this category.
* **General or Strong AI:** Hypothetical AI with human-level intelligence, capable of learning and performing any intellectual task a human can.
* **Super AI:** Hypothetical AI that surpasses human intelligence in all aspects.

Examples of AI in action:

* **Virtual assistants:** Siri, Alexa, and Google Assistant
* **Self-driving cars:** Tesla Autopilot, Waymo
* **Recommendation systems:** Netflix, Spotify

```
* **Medical diagnosis:** AI-powered tools to assist doctors in detecting
diseases
* **Fraud detection:** AI algorithms to identify suspicious transactions

**It's important to note:**

AI is a rapidly evolving field with both exciting potential and ethical
considerations. It's crucial to have open discussions and responsible
development to ensure AI benefits society as a whole.

User: q
Good Bye
```

[ ]: