

텍스트 감성 분류 체인 만들기

```
In [1]: !pip install -Uq langchain_core langchain_openai langchain
```

```
_____ 50.4/50.4 kB 1.6 MB/s eta 0:00:00
_____ 405.1/405.1 kB 10.1 MB/s eta 0:00:00
_____ 51.5/51.5 kB 2.7 MB/s eta 0:00:00
_____ 1.0/1.0 MB 28.2 MB/s eta 0:00:00
_____ 290.0/290.0 kB 14.9 MB/s eta 0:00:00
_____ 375.0/375.0 kB 19.1 MB/s eta 0:00:00
_____ 1.1/1.1 MB 33.5 MB/s eta 0:00:00
_____ 76.4/76.4 kB 4.4 MB/s eta 0:00:00
_____ 77.9/77.9 kB 5.0 MB/s eta 0:00:00
_____ 318.9/318.9 kB 16.2 MB/s eta 0:00:00
_____ 141.9/141.9 kB 6.6 MB/s eta 0:00:00
_____ 58.3/58.3 kB 2.8 MB/s eta 0:00:00
```

```
In [8]: import os
from openai import OpenAI

def init_api():
    with open("chatgpt_kict2409.env") as env:
        for line in env:
            key, value = line.strip().split("=")
            os.environ[key] = value

init_api()
# client = OpenAI(api_key = os.environ.get("API_KEY"))
os.environ["OPENAI_API_KEY"] = os.environ.get("API_KEY")
```

<openai.OpenAI object at 0x78234b7fbd00>

```
In [ ]: from langchain_core.prompts import ChatPromptTemplate
from langchain.output_parsers.enum import EnumOutputParser # 모델의 출력을 지정
from langchain_openai import ChatOpenAI # OpenAI의 채팅 모델
from enum import Enum # 열거형(enum) 타입
```

```
In [27]: # 감성을 나타내는 Enum 클래스 정의
class Sentiment(Enum):
    POSITIVE = 'positive'
    NEGATIVE = 'negative'

# 감성 분석을 위한 프롬프트 템플릿 생성
prompt_template = ChatPromptTemplate.from_template("Classify '{input_text}' as ")

# LangChain 체인 생성 :
# 1. 프롬프트 템플릿 적용 : 생성한 프롬프트 템플릿 적용
# 2. ChatOpenAI 모델 사용 : 모델을 사용하여 입력 텍스트 분석
# 3. EnumOutputParser를 사용하여 Sentiment Enum 값 반환 : Sentiment Enum 값을 받
chain = prompt_template | ChatOpenAI() | EnumOutputParser(enum=Sentiment)

# 문장 입력 받기
my_txt = input("오늘은 어떤 컨디션인가요? 한문장으로 표현해 주세요?")

result = chain.invoke({'input_text':my_txt})
print(result) # 출력: Sentiment.POSITIVE: '긍정'
```

오늘은 어떤 컨디션인가요? 한문장으로 표현해 주세요?나는 LLM을 좋아해.
Sentiment.POSITIVE

[실습 1] 출력을 긍정, 부정이 출력되도록 변경해 보자.

```
In [28]: # 감성을 나타내는 Enum 클래스 정의
class Sentiment(Enum):
    POSITIVE = '긍정'
    NEGATIVE = '부정'

# 감성 분석을 위한 프롬프트 템플릿 생성
prompt_template = ChatPromptTemplate.from_template(
    "다음 문장을 '긍정' 또는 '부정' 중 하나로만 정확하게 분류하여 응답하세요. 다른
)

# LangChain 체인 생성 :
# 1. 프롬프트 템플릿 적용 : 생성한 프롬프트 템플릿 적용
# 2. ChatOpenAI 모델 사용 : 모델을 사용하여 입력 텍스트 분석
# 3. EnumOutputParser를 사용하여 Sentiment Enum 값 반환 : Sentiment Enum 값을 받
chain = prompt_template | ChatOpenAI() | EnumOutputParser(enum=Sentiment)

# 문장 입력 받기
my_txt = input("오늘은 어떤 컨디션인가요? 한문장으로 표현해 주세요?")

result = chain.invoke({'input_text':my_txt})
print(result.value) # 출력: Sentiment.POSITIVE: '긍정'
```

오늘은 어떤 컨디션인가요? 한문장으로 표현해 주세요?오늘은 날씨가 참 좋다.
긍정

- [실습 2] 입력을 받아, 긍정, 부정이 출력되도록 해보자.
- [실습 3] 모델을 'gpt-4o-mini'을 사용하도록 해 보자.

In []: