챗봇 프로젝트 06

복습

https://littlefoxdiary.tistory.com/42 블로그 접속

해당 블로그에 정리된 한국어 데이터셋(JSON) 중 하나를 선택하여

원하는 값을 pandas 데이터프레임으로 추출하는 전처리 과정 진행하기

JSON 데이터 전처리

aihub 접속
도서자료 기계독해 샘플데이터 다운로드

JSON 데이터 전처리

In [1]:
```
import json
import pandas as pd
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity
```

In [2]:
```
model = SentenceTransformer('jhgan/ko-sroberta-multitask')
```

In [6]:
```
df = pd.read_json('tqa_sample.json')
df.head()
```

Out [6]:

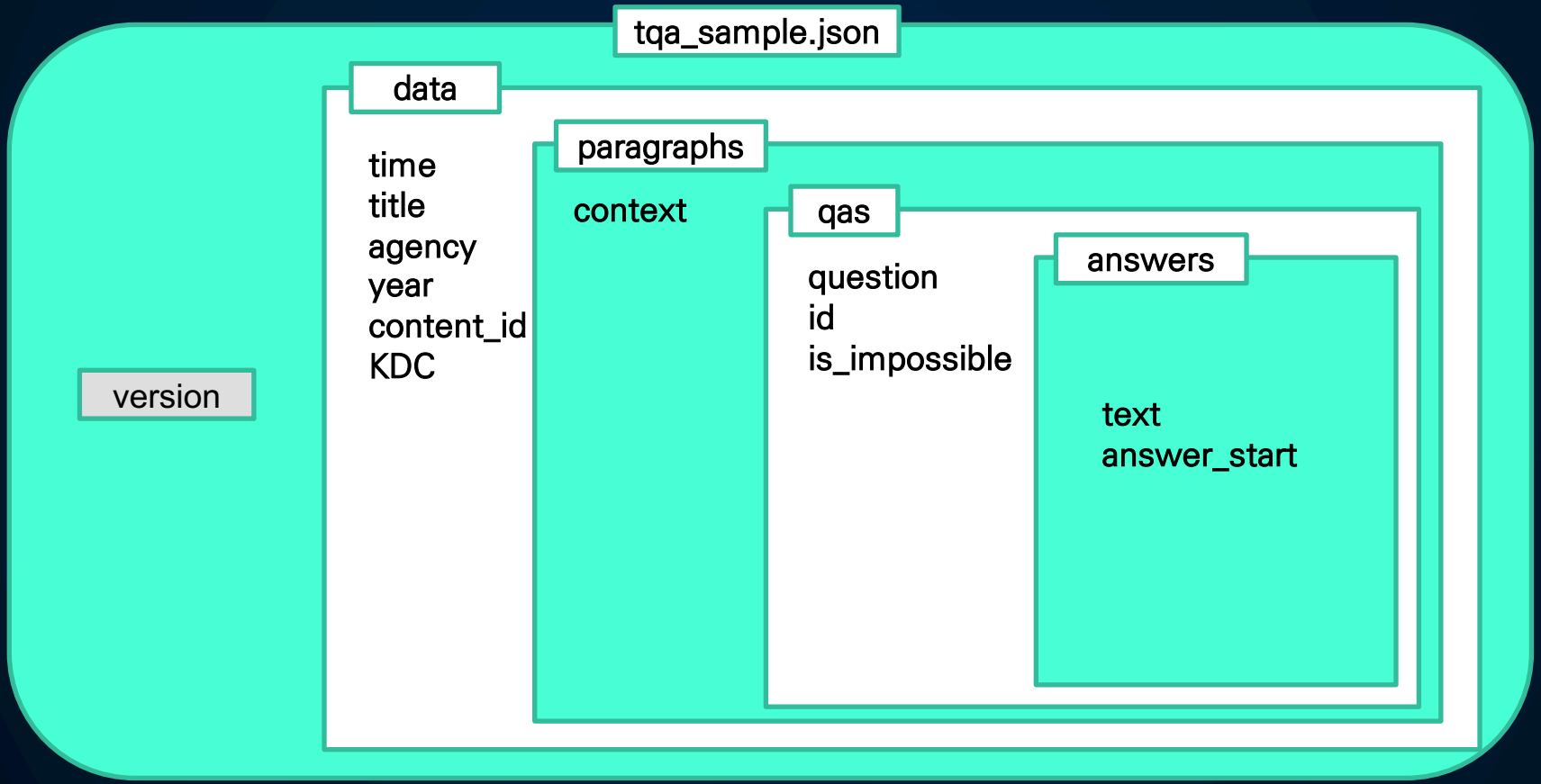
	version	data
0	v0.1	{'time': '20201029154920', 'title': '생물다양성과 합성...
1	v0.1	{'time': '20201029154602', 'title': '해외 환경정책 인...
2	v0.1	{'time': '20201029155823', 'title': '범죄 빅데이터를 ...
3	v0.1	{'time': '20201029154348', 'title': '한국관광정책Kor...
4	v0.1	{'time': '20201029155011', 'title': '미국 중학교 제도...

data 안의 하위 데이터들이
전부 합쳐진 형태로 생성됨

sample.json


```
{
    "version": "v0.1",
    "data": [
        {
            "time": "20201029154920",
            "title": "생물다양성과 합성생물학",
            "agency": "환경부 국립생물자원관,세종대학교",
            "year": "2019",
            "content_id": "PCY_202003100356096481",
            "KDC": "00",
            "paragraphs": [
                {
                    "context": "국제연합을 비롯하여 전문성을 갖춘 국제기구들이 설립되기 시작하면서 분야별 과학적 증거의 수집·검토 및 국가별 실질적 
                    "qas": [
                        {
                            "question": "국제사회는 무엇을 거쳐서 국제적 거버넌스를 구성하였지",
                            "id": 427521,
                            "is_impossible": false,
                            "answers": [
                                {
                                    "text": "구조적 변화",
                                    "answer_start": 257
                                }
                            ]
                        },
                        {
                            "question": "무엇을 거쳐서 국제사회는 국제적 거버넌스를 조직하였어",
                            "id": 427522,
                            "is_impossible": false,
                            "answers": [
                                {
                                    "text": "구조적 변화",
                                    "answer_start": 257
                                }
                            ]
                        }
                    ]
                }
            ]
        }
```

tqa_sample.json 구조



json-parsing.ipynb

데이터가 너무 크기 때문에 시작하기 전에 데이터 일부 삭제하는 과정 필요

```
import json
import pandas as pd
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity
```

```
model = SentenceTransformer('jhgan/ko-sroberta-multitask')
```

```
with open('tqa_sample.json', 'r', encoding='utf-8') as f:
    data = json.load(f)
```

json-parsing.ipynb

```
rows = []
for d in data['data']:
    for p in d['paragraphs']:
        for q in p['qas']:
            row = {
                'question': q['question'],
                'answer': q['answers'][0]['text'],
                'context': p['context']
            }
            rows.append(row)

df = pd.DataFrame(rows, columns=['question', 'answer', 'context'])
```

```
df.head()
```

pd.DataFrame()
Dataframe을 생성하는 함수

rows
각 행에 저장할 데이터
(question, answer, context)를
순서대로 저장한 리스트

columns
각 열의 이름을 지정

json-parsing.ipynb

	question	answer	context
0	국제사회는 무엇을 거쳐서 국제적 거버넌스를 구성하였지	구조적 변화	국제연합을 비롯하여 전문성을 갖춘 국제기구들이 설립되기 시작하면서 분야별 과학적 증...
1	무엇을 거쳐서 국제사회는 국제적 거버넌스를 조직하였어	구조적 변화	국제연합을 비롯하여 전문성을 갖춘 국제기구들이 설립되기 시작하면서 분야별 과학적 증...
2	국제사회가 국제적 거버넌스를 만들기 위해 거친 게 뭘까	구조적 변화	국제연합을 비롯하여 전문성을 갖춘 국제기구들이 설립되기 시작하면서 분야별 과학적 증...
3	대학교가 학생회를 만들기 위해 거친 게 뭘까	구조적 변화	국제연합을 비롯하여 전문성을 갖춘 국제기구들이 설립되기 시작하면서 분야별 과학적 증...
4	국제사회는 구조적 변화를 거쳐서 무엇을 구성했지	국제적 거버넌스	국제연합을 비롯하여 전문성을 갖춘 국제기구들이 설립되기 시작하면서 분야별 과학적 증...

해당 데이터 활용 예시

유저가 특정 단어에 대한 질문(question)을 말하면,
챗봇은 유저가 설명하는 단어의 정답(answer)을 말하고,
그 단어에 대한 의미(context)를 출력한다.

json-parsing.ipynb

앞서 진행했던 데이터셋과 동일하게
데이터프레임에 embedding열 추가한 후
json파일로 내보내기

```
df.to_json('tqa_dataset.json')
```

quiz-chatbot.py

```
import streamlit as st
from streamlit_chat import message
import pandas as pd
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity
import json

@st.cache(allow_output_mutation=True)
def cached_model():
    model = SentenceTransformer('jhgan/ko-sroberta-multitask')
    return model

@st.cache(allow_output_mutation=True)
def get_dataset():
    df = pd.read_json('tqa_dataset.json')
    return df
```

quiz-chatbot.py

```
model = cached_model()
df = get_dataset()

st.header('도서 퀴즈 챗봇')

if 'generated' not in st.session_state:
    st.session_state['generated'] = []

if 'past' not in st.session_state:
    st.session_state['past'] = []

if 'context' not in st.session_state:
    st.session_state['context'] = []
```

quiz-chatbot.py

```
with st.form('form', clear_on_submit=True):
    user_input = st.text_input('당신: ', '')
    submitted = st.form_submit_button('전송')

if submitted and user_input:
    embedding = model.encode(user_input)

    df['distance'] = df['embedding'].map(lambda x:
cosine_similarity([embedding], [x]).squeeze())
    answer = df.loc[df['distance'].idxmax()]

    st.session_state.past.append(user_input)
    st.session_state.generated.append(answer['answer'])
    st.session_state.context.append(answer['context'])
```

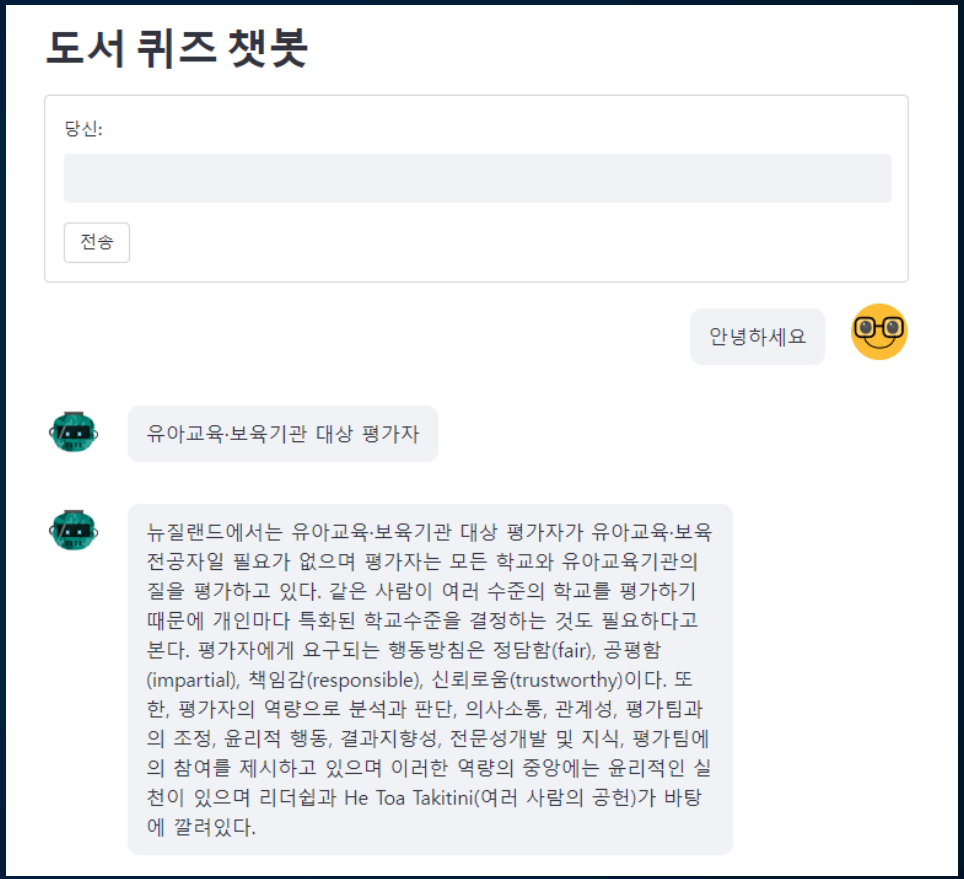
quiz-chatbot.py

```
for i in range(len(st.session_state['past'])):
    message(st.session_state['past'][i], is_user=True, key=str(i)
+ '_user')
    if len(st.session_state['generated']) > i:
        message(st.session_state['generated'][i], key=str(i) +
'_bot')
        message(st.session_state['context'][i], key=str(i) +
'_botContext')
```

작성 후, streamlit run quiz-chatbot.py 실행

데이터를 많이 잘라냈기 때문에

답변의 정확도가 높지 않음.



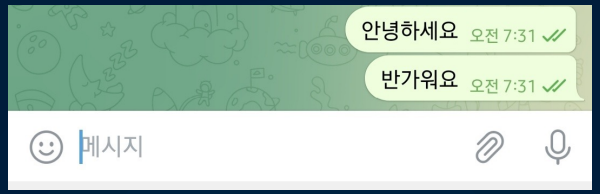
python으로 구현한 챗봇 텔레그램 연동

기존에 만들었던 텔레그램 봇 재사용(없을 경우 새로 생성)

bot 토큰 필요

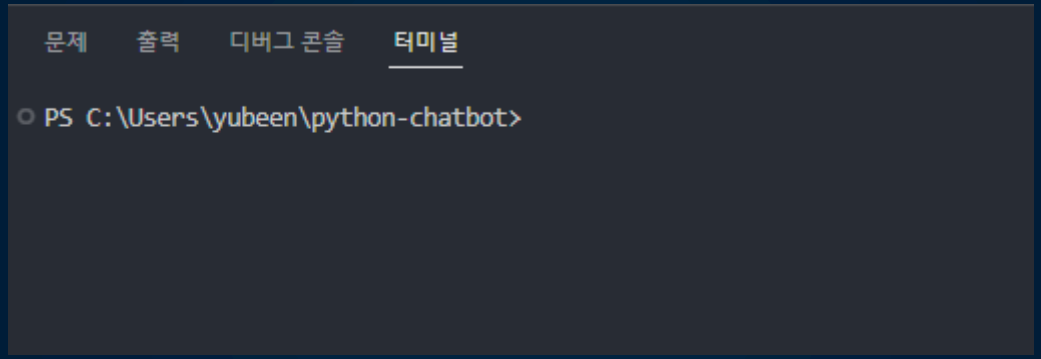
https://api.telegram.org/bot{봇토큰}/getUpdates
ex) https://api.telegram.org/bot6148368452:AAF-w4V91NLqst87DrBaJmMxK80M4p62HNI/getUpdates

{"ok":true,"result":[]}

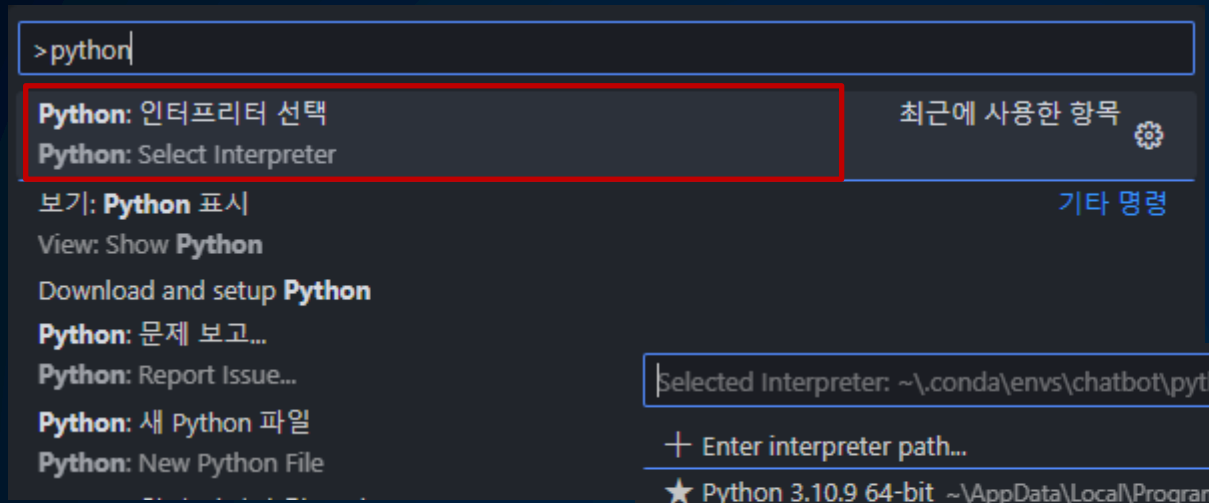


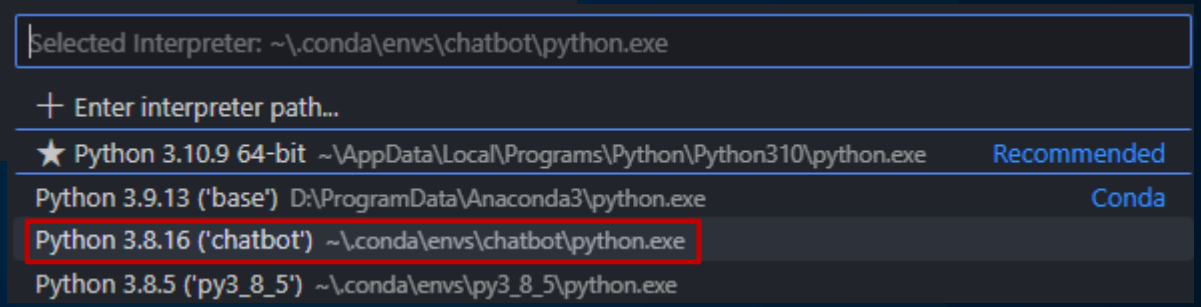
{"ok":true,"result":[{"update_id":28530052,
"message":{"message_id":103,"from":{"id":6095113280,"is_bot":false,"first_name":"#uc720#ube48","last_name":"#uc774","language_code":"ko"},"chat":
{"id":6095113280,"first_name":"#uc720#ube48","last_name":"#uc774","type":"private"},"date":1682202658,"text":"#uc57c#uc548#ub155"}},{"update_id":28530053,
"message":{"message_id":104,"from":{"id":6095113280,"is_bot":false,"first_name":"#uc720#ube48","last_name":"#uc774","language_code":"ko"},"chat":
{"id":6095113280,"first_name":"#uc720#ube48","last_name":"#uc774","type":"private"},"date":1682202659,"text":"#uc548#ub155"}},{"update_id":28530054,
"message":{"message_id":105,"from":{"id":6095113280,"is_bot":false,"first_name":"#uc720#ube48","last_name":"#uc774","language_code":"ko"},"chat":
{"id":6095113280,"first_name":"#uc720#ube48","last_name":"#uc774","type":"private"},"date":1682202660,"text":"#ud558#uc774"}},{"update_id":28530055,
"message":{"message_id":106,"from":{"id":6095113280,"is_bot":false,"first_name":"#uc720#ube48","last_name":"#uc774","language_code":"ko"},"chat":
{"id":6095113280,"first_name":"#uc720#ube48","last_name":"#uc774","type":"private"},"date":1682202661,"text":"#ubb50#ud568"}}]}

visual studio code anaconda연결



vscode 기본 터미널
아나콘다 가상환경과 연결이 되어있지 않음.

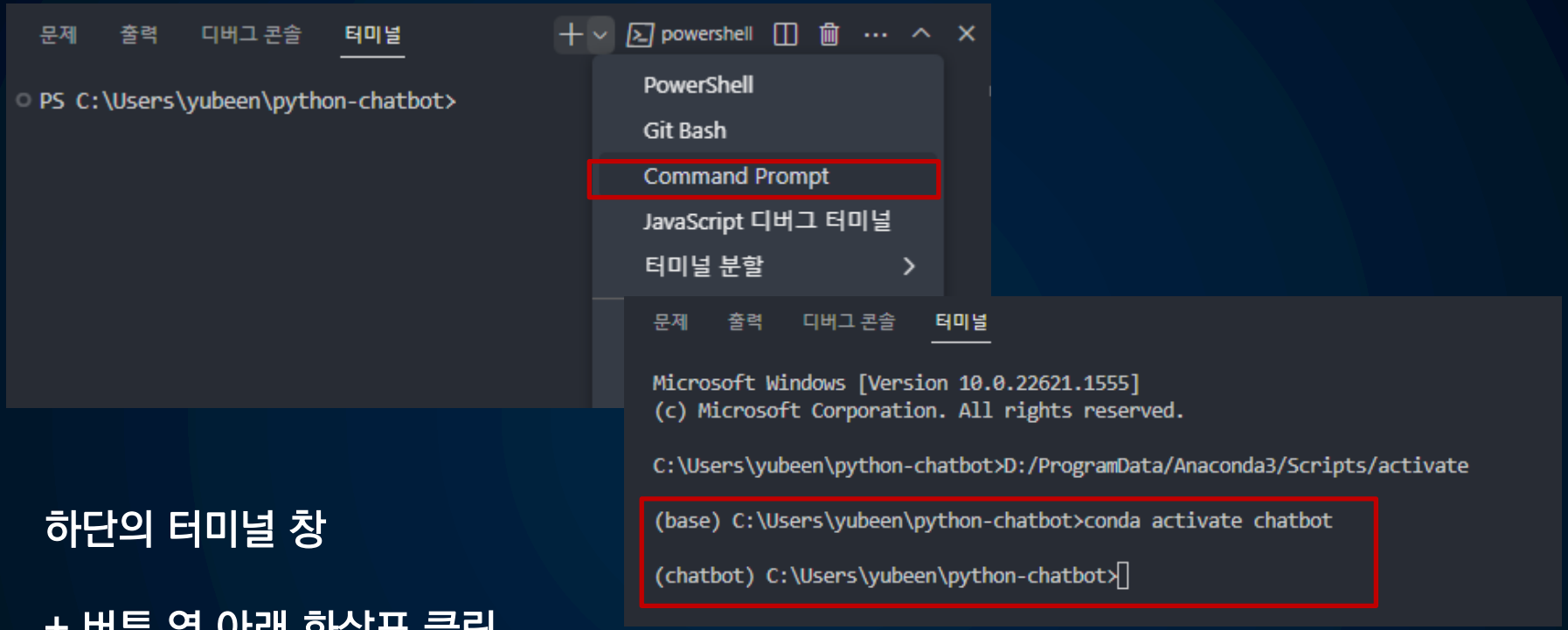




ctrl + shift + p 입력,

python: select interpreter 선택

그 후 chatbot 가상환경 선택



하단의 터미널 창

+ 버튼 옆 아래 화살표 클릭,

Command Prompt 선택

클릭 후, 터미널을

conda prompt처럼 이용가능

```
(chatbot) C:\Users\yubeen\python-chatbot>pip install python-telegram-bot
```

하단의 터미널 창

+ 버튼 옆 아래 화살표 클릭,

Command Prompt 선택

telegram.py

```
import requests

token = '6148368452:AAF-w4V91NLqst87DrBaJmMxK80M4p62HNI'
chat_id = 6095113280
bot_message = "hello"

url =
f'https://api.telegram.org/bot{token}/sendMessage?chat_id={chat_id
}&text={bot_message}'

requests.get(url)
```

```
(chatbot) C:\Users\yubeen\python-chatbot>python telegram.py
https://api.telegram.org/botbot6148368452:AAF-w4V91NLqst87DrBaJmMxK80M4p62HNI/sendMessage
?chat_id=6095113280&text=hello
```

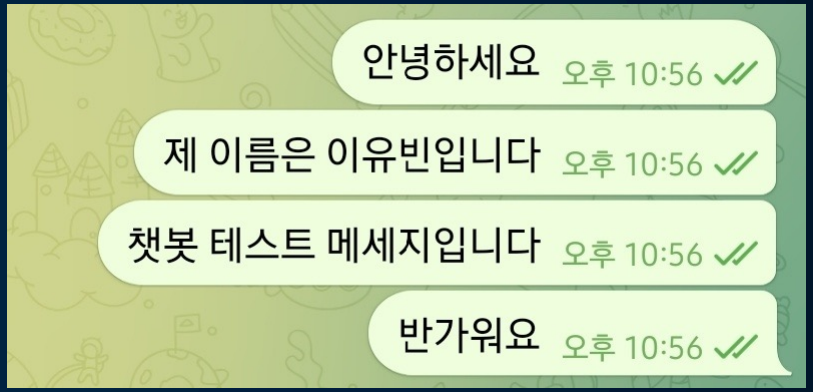
1. python telegram.py 입력 (코드 실행)
   실행 시 텔레그램 봇 채널에서 “hello” 메시지 전송

2. print(url)로 해당 api 주소 출력
   (ctrl + 클릭으로 해당 주소 이동)
   주소 접속 시 동일하게 봇 채널에서 “hello” 메시지 전송

텔레그램 api로 json 데이터 추출

getUpdates 요청을 통해
새로운 메시지 정보 받아오기

텔레그램 봇에게 메시지 전송



get-json.py

```
import requests
import json

token = '6148368452:AAF-w4V91NLqst87DrBaJmMxK80M4p62HNI'
chat_id = 6095113280
bot_message = "hello"

url = f"https://api.telegram.org/bot{token}/getUpdates"
data = requests.get(url).json()

print(url)

with open('chat.json', 'w', encoding='utf-8') as file:
    json.dump(data, file, indent="\t", ensure_ascii=False)
```

```
(chatbot) C:\Users\yubeen\python-chatbot>python get-json.py
https://api.telegram.org/bot6148368452:AAF-w4V91NLqst87DrBaJmMxK80M4p62HNI/getUpdates
```

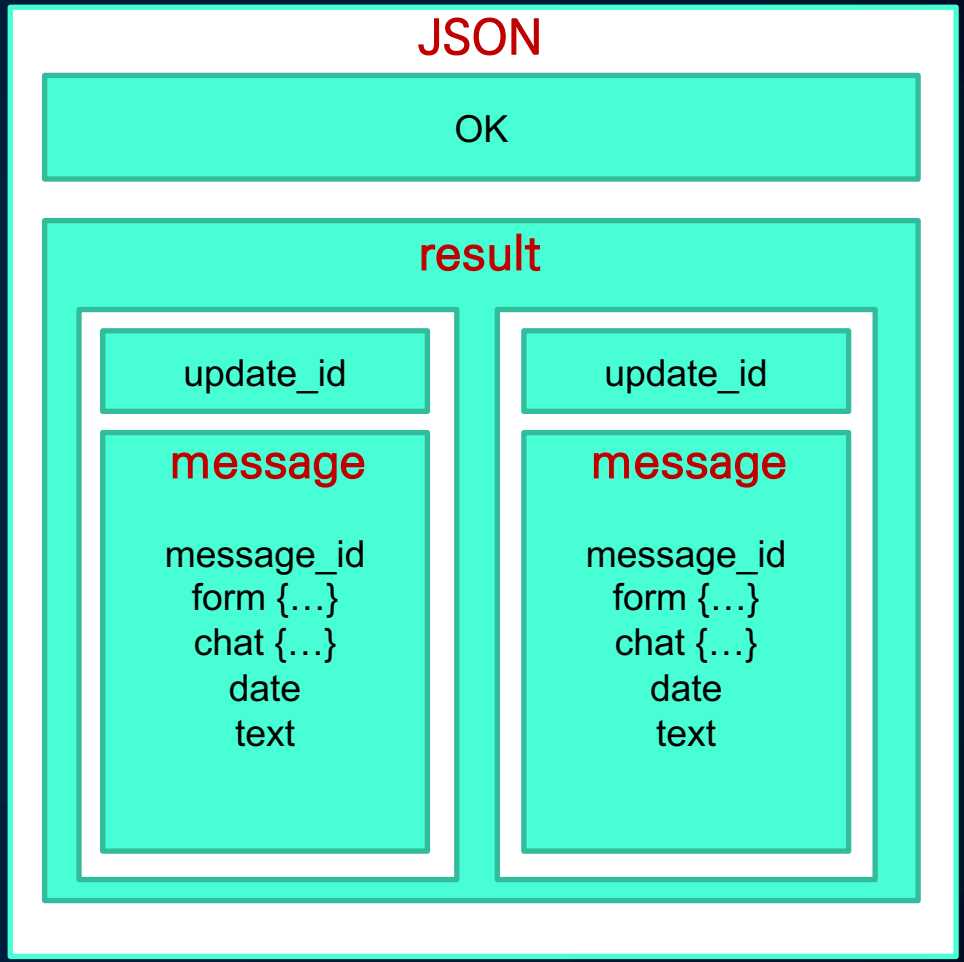
1. python get-json.py입력 (코드 실행)
   유저가 봇 채널에 입력한 메시지의 정보를
   해당 폴더에 json파일로 저장

2. print(url)로 해당 api 주소 출력
   (ctrl + 클릭으로 해당 주소 이동)
   유저가 입력한 메시지에 대한 정보를 json 형식으로
   보여주는 페이지로 이동

```
{
    "ok": true,
    "result": [
        {
            "update_id": 28530064,
            "message": {
                "message_id": 133,
                "from": {
                    "id": 6095113280,
                    "is_bot": false,
                    "first_name": "유빈",
                    "last_name": "이",
                    "language_code": "ko"
                },
                "chat": {
                    "id": 6095113280,
                    "first_name": "유빈",
                    "last_name": "이",
                    "type": "private"
                },
                "date": 1682776593,
                "text": "안녕하세요"
            }
        },
        {
            "update_id": 28530065,
            "message": {
```

{"ok":true,"result":[{"update_id":28530064,
"message":{"message_id":133,"from":
{"id":6095113280,"is_bot":false,"first_name":"₩uc720₩ube48","last_name":"₩uc774","lan
guage_code":"ko"},"chat":
{"id":6095113280,"first_name":"₩uc720₩ube48","last_name":"₩uc774","type":"private"},"
date":1682776593,"text":"₩uc548₩ub155₩ud558₩uc138₩uc694"}},{"update_id":28530065,
"message":{"message_id":134,"from":
{"id":6095113280,"is_bot":false,"first_name":"₩uc720₩ube48","last_name":"₩uc774","lan
guage_code":"ko"},"chat":
{"id":6095113280,"first_name":"₩uc720₩ube48","last_name":"₩uc774","type":"private"},"
date":1682776597,"text":"₩uc81c ₩uc774₩ub984₩uc740
₩uc774₩uc720₩ube48₩uc785₩ub2c8₩ub2e4"}},{"update_id":28530066,
"message":{"message_id":135,"from":
{"id":6095113280,"is_bot":false,"first_name":"₩uc720₩ube48","last_name":"₩uc774","lan
guage_code":"ko"},"chat":
{"id":6095113280,"first_name":"₩uc720₩ube48","last_name":"₩uc774","type":"private"},"
date":1682776602,"text":"₩ucc57₩ubd07 ₩ud14c₩uc2a4₩ud2b8
₩uba54₩uc138₩uc9c0₩uc785₩ub2c8₩ub2e4"}},{"update_id":28530067,
"message":{"message_id":136,"from":
{"id":6095113280,"is_bot":false,"first_name":"₩uc720₩ube48","last_name":"₩uc774","lan
guage_code":"ko"},"chat":
{"id":6095113280,"first_name":"₩uc720₩ube48","last_name":"₩uc774","type":"private"},"
date":1682776604,"text":"₩ubc18₩uac00₩uc6cc₩uc694"}}]}

```
{
    "ok": true,
    "result": [
        {
            "update_id": 28530064,
            "message": {
                "message_id": 133,
                "from": {
                    "id": 6095113280,
                    "is_bot": false,
                    "first_name": "유빈",
                    "last_name": "이",
                    "language_code": "ko"
                },
                "chat": {
                    "id": 6095113280,
                    "first_name": "유빈",
                    "last_name": "이",
                    "type": "private"
                },
                "date": 1682776593,
                "text": "안녕하세요"
            }
        },
        {
            "update_id": 28530065,
            "message": {
```

chatbot-telegram.py

```
import requests
import pandas as pd
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity
import json

token = "6148368452:AAF-w4V91NLqst87DrBaJmMxK80M4p62HNI"
chat_id = 6095113280
```

필요한 라이브러리 import

```
def cached_model():
    model = SentenceTransformer('jhgan/ko-sroberta-multitask')
    return model

def get_dataset():
    df = pd.read_csv('wellness_dataset.csv')
    df['embedding'] = df['embedding'].apply(json.loads)
    return df

model = cached_model()
df = get_dataset()
```

언어처리 모델과 챗봇 dataset 캐싱

```python
last_update_id = None

while True:
    url = f"https://api.telegram.org/bot{token}/getUpdates"
    if last_update_id:
        url += f"?offset={last_update_id + 1}"
    response = requests.get(url)
    if response.status_code != 200:
        continue

    data = response.json()
```

getUpdates로 유저가 입력한 대화 정보를 받아온다.
last_update_id = 마지막 메시지의 id를 저장

while문 안에 이어서 계속 작성.

```
for update in data["result"]:
        text = update["message"]["text"]
        chat_id = update["message"]["chat"]["id"]
        update_id = update["update_id"]

        last_update_id = update_id

        embedding = model.encode(text)

        df['distance'] = df['embedding'].map(lambda x:
cosine_similarity([embedding], [x]).squeeze())
        answer = df.loc[df['distance'].idxmax()]
        bot_message = answer['챗봇']
```

json data에서 text, chat_id, update_id 추출.

하단 코드는 streamlit챗봇 코드에서 사용했던 코드의 재사용

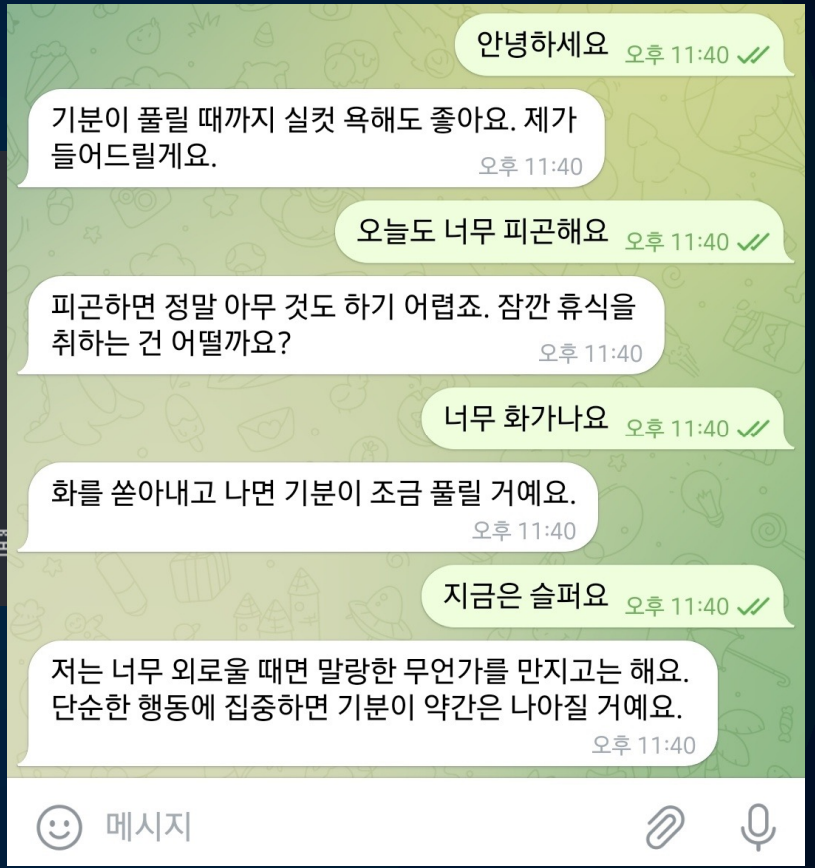
while문 안에 이어서 계속 작성.

```
        print(answer['distance'])
        print(answer['유저'])
        print(answer['챗봇'])

        url =
f"https://api.telegram.org/bot{token}/sendMessage?chat_id={chat_i
d}&text={bot_message}"

        requests.get(url)
```

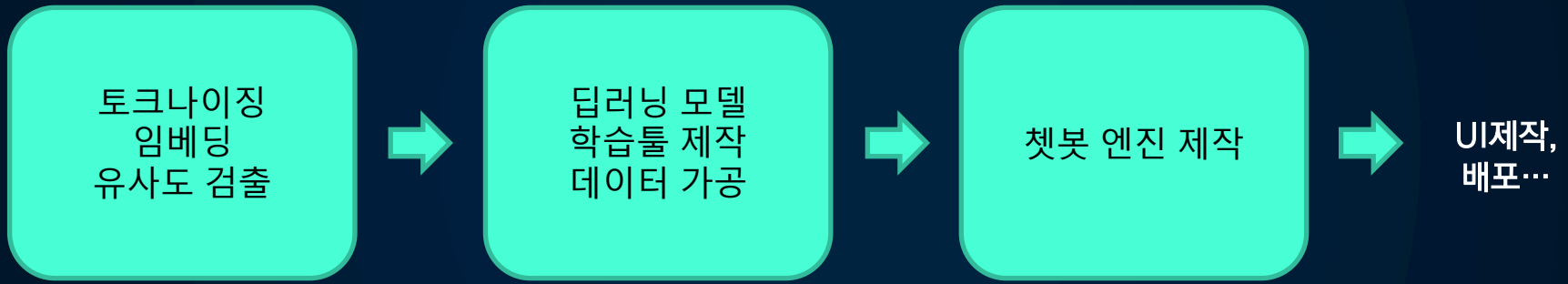
코드 작성 후
python chatbot-telegram.py 실행

```
(chatbot) C:\Users\yubeen\python-chatbot>python chatbot-telegram.py
```

```
0.4965766004177441
어휴, 그런 말은 나도 하겠네.
기분이 풀릴 때까지 실컷 욕해도 좋아요. 제가 들어드릴게요.
0.8991264861037545
요즘 너무 피곤해요.
피곤하면 정말 아무 것도 하기 어렵죠. 잠깐 휴식을 취하는 건 어떨까요?
0.851989148353177
진짜 어이없고 화가 나서 숨이 턱 막히는 거야.
화를 쏟아내고 나면 기분이 조금 풀릴 거예요.
0.6422633553449788
너무 외로워요.
저는 너무 외로울 때면 말랑한 무언가를 만지고는 해요. 단순한 행동에 집중
```



자연어 처리 & 챗봇 엔진 구현



KoNLPy

```
[('아버지', 'Noun'), ('가', 'Josa'), ('방', 'Noun'), ('에', 'Josa'), ('들어가신다',
'Verb'), ('.', 'Punctuation')]
```

한국어 정보를 처리를 위한 파이썬 라이브러리 패키지

자연어처리(NLP) 과정에서 문장을 명사, 조사, 동사 등 형태소 단위로 분리하는 역할
(토크나이징, Tokenizing)

Hannanum, Kkma, Komoran, Mecab, Okt 5가지 클래스로 제공된다.

개발환경 세팅

가상환경 추가 설치 (chatbot2)

anaconda prompt 실행 후

conda create -n chatbot2 python=3.8 // python3.8 버전의 가상환경 생성

conda activate chatbot2 // chatbot2 가상환경 활성화

pip install konlpy

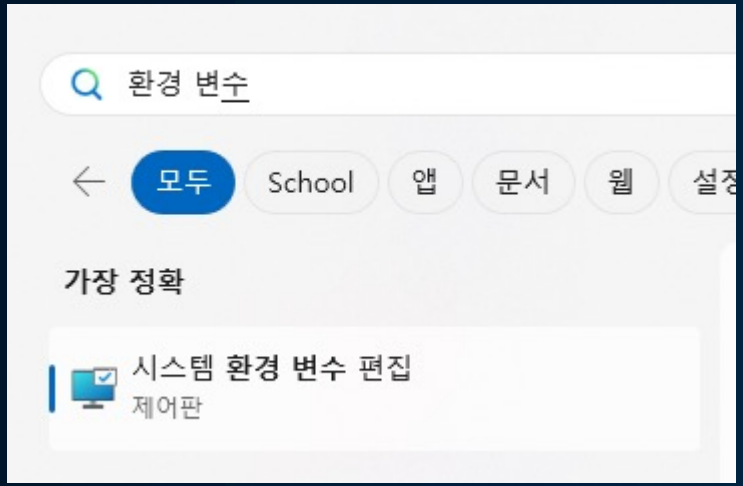
JDK 설치

C:\Program Files\Java 폴더가 없다면 JAVA 설치

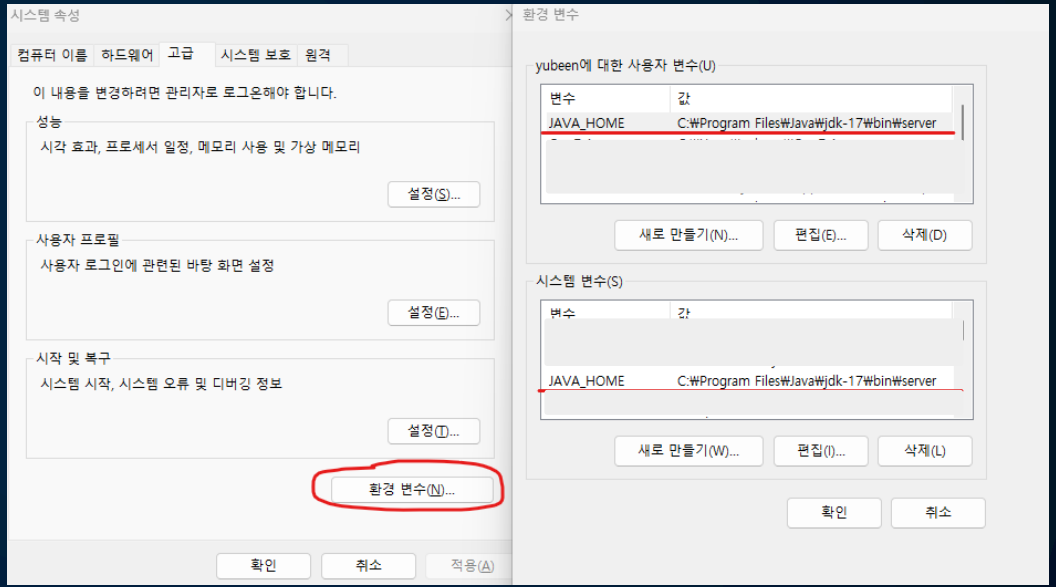
JAVA 설치 링크
https://www.java.com/ko/download/ie_manual.jsp?locale=ko

JDK 설치 (17.0.7)
https://www.oracle.com/kr/java/technologies/downloads/#java17

환경변수 설정



환경변수 시스템 변수 새로만들기
변수 이름 : JAVA_HOME
값 : C:\Program Files\Java\jdk-17



시스템 변수 〉 Path 〉 새로만들기
%JAVA_HOME%\bin 추가

JPYPE1 설치

설치 링크
https://www.lfd.uci.edu/~gohlke/pythonlibs/#jpype

JPype1-1.1.2-cp38-cp38-win_amd64.whl – 다운로드

Anaconda prompt, chatbot2 가상환경 활성화 상태에서
pip install JPype1-1.1.2-cp38-cp38-win_amd64.whl 입력
(pip install {파일 이름})

```
(chatbot) C:\Users\yubeen\python-chatbot>pip install JPype1-1.1.2-cp38-cp38-win_amd64.whl
Processing c:\users\yubeen\python-chatbot\jpype1-1.1.2-cp38-cp38-win_amd64.whl
Installing collected packages: JPype1
```

설치 확인

Anaconda Prompt 실행 〉 chatbot2 활성화

python 입력 // 파이썬 실행

파이썬 실행 후 다음 코드 입력.

```
〉〉〉 from konlpy.tag import kkma
〉〉〉 kkma = Kkma()
〉〉〉 print(kkma.nouns('안녕하세요 좋은 아침입니다'))
['안녕', '아침']
```

위와 같이 안녕, 아침 출력 시 정상 설치

kkma.py

```
from konlpy.tag import Kkma

kkma = Kkma()
text = "아버지가 방에 들어갑니다."

morphs = kkma.morphs(text)
print(morphs)

pos = kkma.pos(text)
print(pos)

nouns = kkma.nouns(text)
print(nouns)

sentences = "오늘 날씨는 어때요? 내일은 덥다던데."
s = kkma.sentences(sentences)
print(s)
```

anaconda prompt창에서 py파일이 있는 경로로 이동 후,
python kkma.py 입력

품사	설명
NNG	일반 명사
JKS	주격 조사
JKM	부사격 조사
VV	동사
EFN	평서형 종결 어미
SF	마침표, 물음표, 느낌표

```
(chatbot2) C:\Users\yubeen\python-chatbot>python Kkma.py
['아버지', '가', '방', '에', '들어가', 'ㅂ니다', '.']
[('아버지', 'NNG'), ('가', 'JKS'), ('방', 'NNG'), ('에', 'JKM'), ('들어가', 'VV'), ('ㅂ니다', 'EFN'), ('.', 'SF')]
['아버지', '방']
['오늘 날씨는 어 때요?', '내일은 덥다 던데.']
```

Komoran.py

```
from konlpy.tag import Komoran

komoran = Komoran()
text = "아버지가 방에 들어갑니다."

morphs = komoran.morphs(text)
print(morphs)

pos = komoran.pos(text)
print(pos)

nouns = komoran.nouns(text)
print(nouns)
```

anaconda prompt창에서 py파일이 있는 경로로 이동 후,
python komoran.py 입력

komoran과 kkma 클래스는 함수와 품사 태그 등의
인터페이스가 동일하다

품사	설명
NNG	일반 명사
JKS	주격 조사
JKM	부사격 조사
VV	동사
EFN	평서형 종결 어미
SF	마침표, 물음표, 느낌표

```
(chatbot2) C:\Users\yubeen\python-chatbot>python komoran.py
['아버지', '가', '방', '에', '들어가', 'ㅂ니다', '.']
[('아버지', 'NNG'), ('가', 'JKS'), ('방', 'NNG'), ('에', 'JKB'), ('들어가', 'VV'), ('ㅂ니다', 'EF'), ('.', 'SF')]
['아버지', '방']
```

Okt.py

```
from konlpy.tag import Okt

okt = Okt()
text = "아버지가 방에 들어갑니다."

morphs = okt.morphs(text)
print(morphs)

pos = okt.pos(text)
print(pos)

nouns = okt.nouns(text)
print(nouns)

text = "오늘 날씨가 좋아요ㅎㅎ~"
print(okt.normalize(text))
print(okt.phrases(text))
```

anaconda prompt창에서 py파일이 있는 경로로 이동 후,
python okt.py 입력

phrases로 어구를 추출하면 띄어쓰기를 포함한
여러 개의 단어를 추출하기도 한다.

표 3-7 Okt 품사 태그 표

품사	설명
Noun	명사
Verb	동사
Adjective	형용사
Josa	조사
Punctuation	구두점

```
(chatbot2) C:\Users\yubeen\python-chatbot>python okt.py
['아버지', '가', '방', '에', '들어갑니다', '.']
[('아버지', 'Noun'), ('가', 'Josa'), ('방', 'Noun'), ('에', 'Josa'), ('들어갑니다', 'Verb'), ('.', 'Punctuation')]
['아버지', '방']
오늘 날씨가 좋아요ㅎㅎ~
['오늘', '오늘 날씨', '날씨']
```

custom_word.py

```python
from konlpy.tag import Komoran

komoran = Komoran()
text = "자연어처리는 엔엘피라고 하지."
pos = komoran.pos(text)
print(pos)
```

실행

```
(chatbot2) C:\Users\yubeen\python-chatbot>python customword.py
[('자연어', 'NNP'), ('처리', 'NNP'), ('는', 'JX'), ('엔', 'NNB'), ('엘', 'NNP'), ('피', 'NNG'), ('이', 'VCP'), ('라
고', 'EC'), ('하', 'VX'), ('지', 'EF'), ('.', 'SF')]
```

custom_word.py 와 같은 경로에 user_dic.tsv 파일 생성

생성 후, 추가 하고 싶은 고유명사 등을 형식에 맞춰 작성
([단어] Tab [품사])

user_dic.tsv 파일 작성 후,
custom_word.py 코드도 아래와 같이 수정.

```
#user_dic.tsv
엔엘피	NNG
스즈메의 문단속	NNG
```

```python
from konlpy.tag import Komoran

komoran = Komoran(userdic='./user_dic.tsv')
text = "자연어처리는 엔엘피라고 한다."
text2 = "나 저번주에 스즈메의 문단속을 봤어."
pos = komoran.pos(text)
pos2 = komoran.pos(text2)
print(pos)
print(pos2)
```

단어의 경우 추가가 되었지만,

공백이 포함 된 여러 개의 단어는 지정이 되지 않음

```
(chatbot2) C:\Users\yubeen\python-chatbot>python customword.py
[('자연어', 'NNP'), ('처리', 'NNP'), ('는', 'JX'), ('엔엘피', 'NNG'), ('이', 'VCP'), ('라고', 'EC'), ('하', 'VX'), (
'ㄴ다', 'EF'), ('.', 'SF')]
[('나', 'NP'), ('저', 'XPN'), ('번주', 'NNG'), ('에', 'JKB'), ('스즈메의', 'NA'), ('문단속', 'NNG'), ('을', 'JKO'),
('보', 'VV'), ('았', 'EP'), ('어', 'EF'), ('.', 'SF')]
```

단어 임베딩

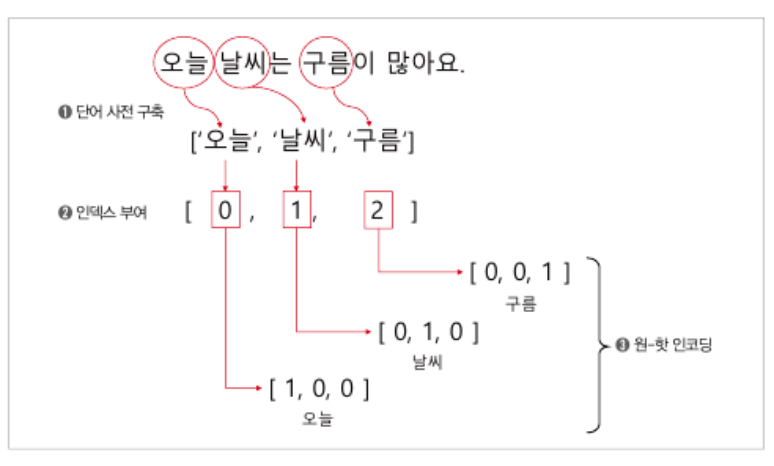
말뭉치에서 각각의 단어를 벡터로 변환하는 기법.

단어를 표현하고 변환하는 기법에는 다양한 종류가 있으며 이 방법에 따라

다양한 모델이 존재.

희소 표현, 분산 표현
원-핫 인코딩
Word2Vec
FastText
BERT(Bidirectional Encoder Representations from Transformers)
...

원-핫 인코딩



단어를 숫자 벡터로 변환하는 가장 기본적인 방법.

요소들 중 단 하나의 값만 1이고 나머지는 0인 인코딩 방식

말뭉치 전체에서 가져온 단어들의 집합이 필요

간단한 방식에 비해 성능이 빠르다.

단어의 수 만큼 벡터의 차원이 결정됨
(단어가 100개면 벡터의 차원도 100개)

차원이 많아지면 성능 저하.

단어	단어 인덱스	원-핫 벡터
you	0	[1, 0, 0, 0, 0, 0, 0]
say	1	[0, 1, 0, 0, 0, 0, 0]
goodbye	2	[0, 0, 1, 0, 0, 0, 0]
and	3	[0, 0, 0, 1, 0, 0, 0]
I	4	[0, 0, 0, 0, 1, 0, 0]
say	5	[0, 0, 0, 0, 0, 1, 0]
hello	6	[0, 0, 0, 0, 0, 0, 1]

oneandhot.py

```
from konlpy.tag import Komoran
import numpy as np

komoran = Komoran()
text = "오늘 날씨는 구름이 많네요"

nouns = komoran.nouns(text)
print(nouns)

dics = {}
for word in nouns:
    if word not in dics.keys():
        dics[word] = len(dics)
print(dics)

nb_classes = len(dics)
targets = list(dics.values())
one_hot_targets = np.eye(nb_classes)[targets]
print(one_hot_targets)
```

실행 결과

```
(chatbot2) C:\Users\yubeen\python-chatbot>python onehot.py
['오늘', '날씨', '구름']
{'오늘': 0, '날씨': 1, '구름': 2}
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```