

빅데이터시스템

BIG DATA ANALYTICS
TECHNOLOGY

통계분석

목 차

01 통계분석

목표설정, 데이터수집, 데이터준비, 데이터탐색, 데이터모델링,
결과시각화

02 AI 비서학습

AI 비서 시스템 구성, 디렉토리구성, 학습데이터셋생성, 학습프로그램,
colab에서 학습

*. 전수업리뷰

⌚ 라이브러리 크롤링 : Selenium

■ 의의

- Selenium(셀레니움) 은 사용자가 아닌 프로그램이 웹 브라우저를 제어할 수 있게 하는 해 주는 라이브러리
- Selenium 은 서버와 클라이언트로 나누는데, 웹 브라우저 종류 마다 클라이언트 프로그램이 별도로 필요(ChromeDriver, Microsoft WebDriver)
- Browser Driver 는 웹 브라우저와 Selenium 서버간 통신을 위한 인터페이스 도구

*. 전수업리뷰

⌚ 라이브러리 크롤링 : Selenium

■ 설치

CMD 창을 띄워서

```

명령 프롬프트
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

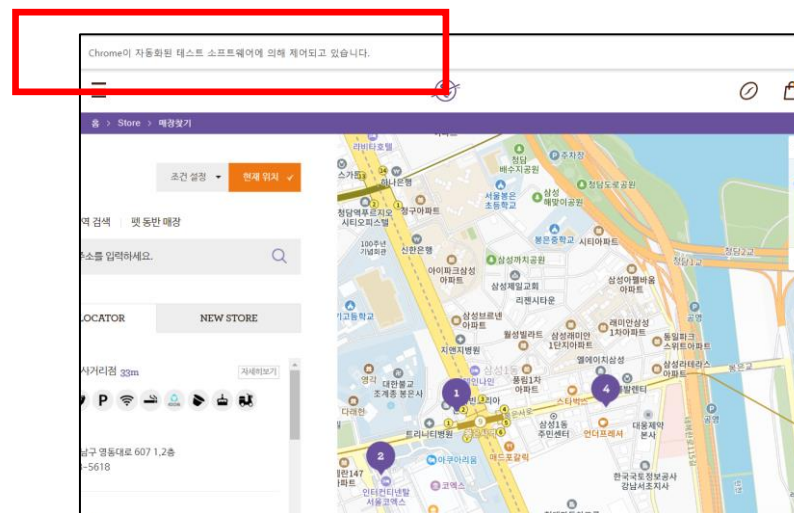
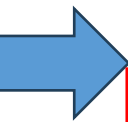
C:\Users\dossa>python -m pip install selenium
Collecting selenium
  Downloading selenium-4.14.0-py3-none-any.whl (9.9 MB)
----- 9.9/9.9 MB 9.6 MB/s
Requirement already satisfied: certifi>=2021.10.8 in c:\applicat
7)
Collecting trio-websocket~0.9
  Downloading trio_websocket-0.11.1-py3-none-any.whl (17 kB)

```

python -m pip install selenium

⌚ 라이브러리 크롤링 : Selenium

■ 실행



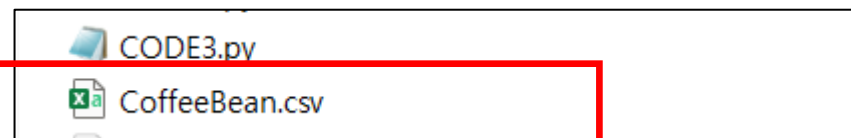
```
cd W_seok
```

cd_crawling

```
python clibmain.py
```

파일이 생성 됨

A	B	C	D
	store	address	phone
0	차병원점	서울시 강남구 논현로 566 강남차병원1층	02-538-7615
1	강남대로점	서울시 서초구 강남대로 369 1층	02-588-5778



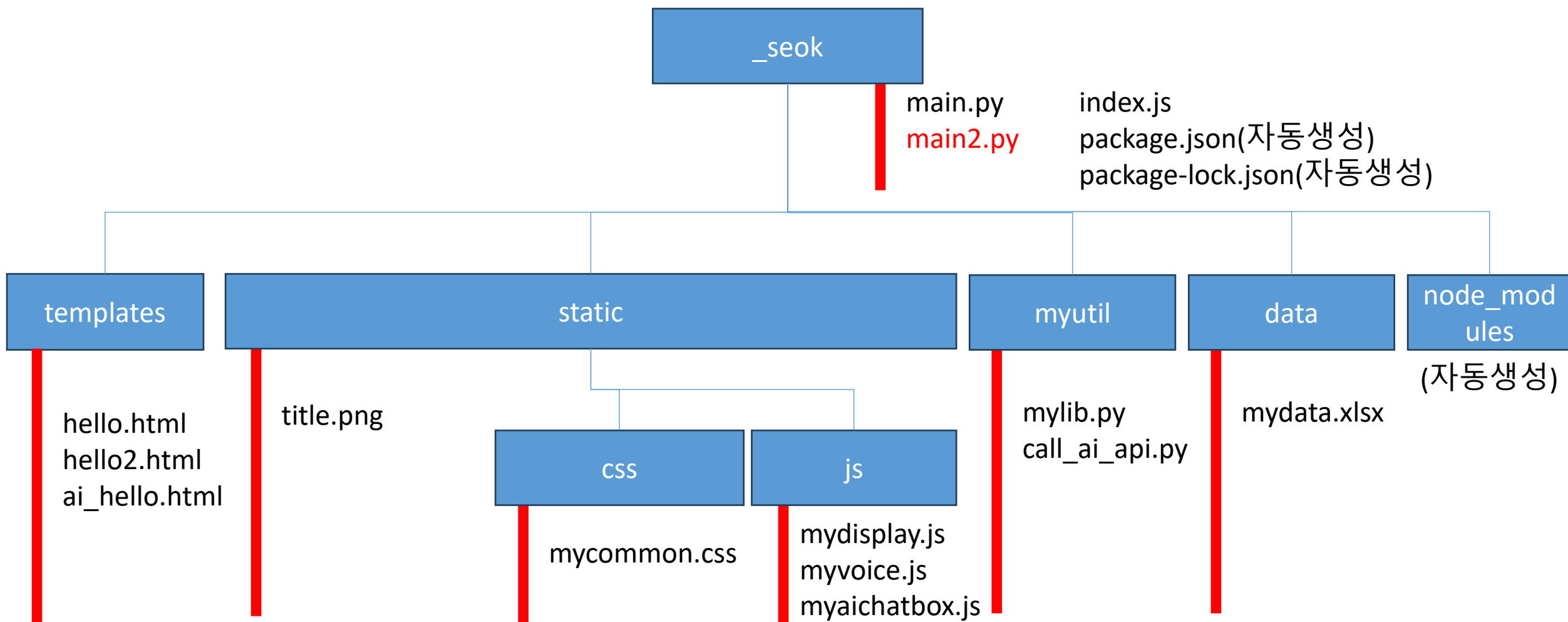
*. 전수업리뷰

⌚ KoGPT

- GPT(Generative Pre-trained Transformer)
 - 글로벌 AI 연구기관인 OpenAI 사에서 개발한 AI모델
 - 생성형 AI 로 딥러닝을 사용하여 인간과 유사한 텍스트를 생성하는 대규모 자연어 기술
 - 이전 단어나 문자를 기반으로 다음 단어나 문자를 예측하고 텍스트요약, 질문에 대한 답변, 챗봇생성 등 다양한 작업을 수행
 - KoGPT는 카카오브레인에서 만든 GPT-3 모델의 한국어 특화 버전
 - 60억개의 매개변수와 2000억개 토큰의 한국어 데이터를 바탕으로 구축
 - KoGPT는 오픈소스 커뮤니티 깃허브(github) 에 2021년 공개

*. 전수업리뷰

⌚ 디렉토리 구성



*. 전수업리뷰

⌚ 패키지 설치 : torch

<https://pytorch.org/get-started/locally/>

PyTorch

nightly. Please ensure that you have **met the prerequisites below (e.g., numpy)**, depending on your package manager. Anaconda is our recommended package manager since it installs all dependencies. You can also **install previous versions of PyTorch**. Note that LibTorch is only available for C++.

PyTorch Build	Stable (2.1.0)	Preview (Nightly)		
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python	C++ / Java		
Compute Platform	CUDA 11.8	CUDA 12.1	ROCm 5.6	CPU
Run this Command:	pip3 install torch torchvision torchaudio			

CMD 창에서 설치

python -m pip install torch torchvision torchaudio

*. 전수업리뷰

⌚ 실행

■ nodejs 실행

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dossa>cd \_seok

C:\_seok>npm start

> myai@1.0.0 start
> nodemon index.js

[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Server listening on port 5555
```

cd _seok

npm start

■ Flask 실행

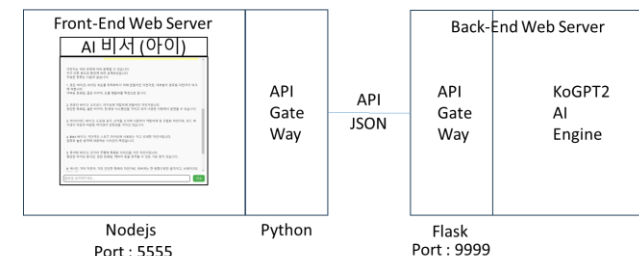
```
C:\_seok>명령 프롬프트 - python main2.py

Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

C:\_seok>python main2.py
* Serving Flask app 'main2'
* Debug mode: off
WARNING: This is a development server. Do not use without proper
security measures.
* Running on http://172.16.11.220:9999
Press CTRL+C to quit
```

cd _seok

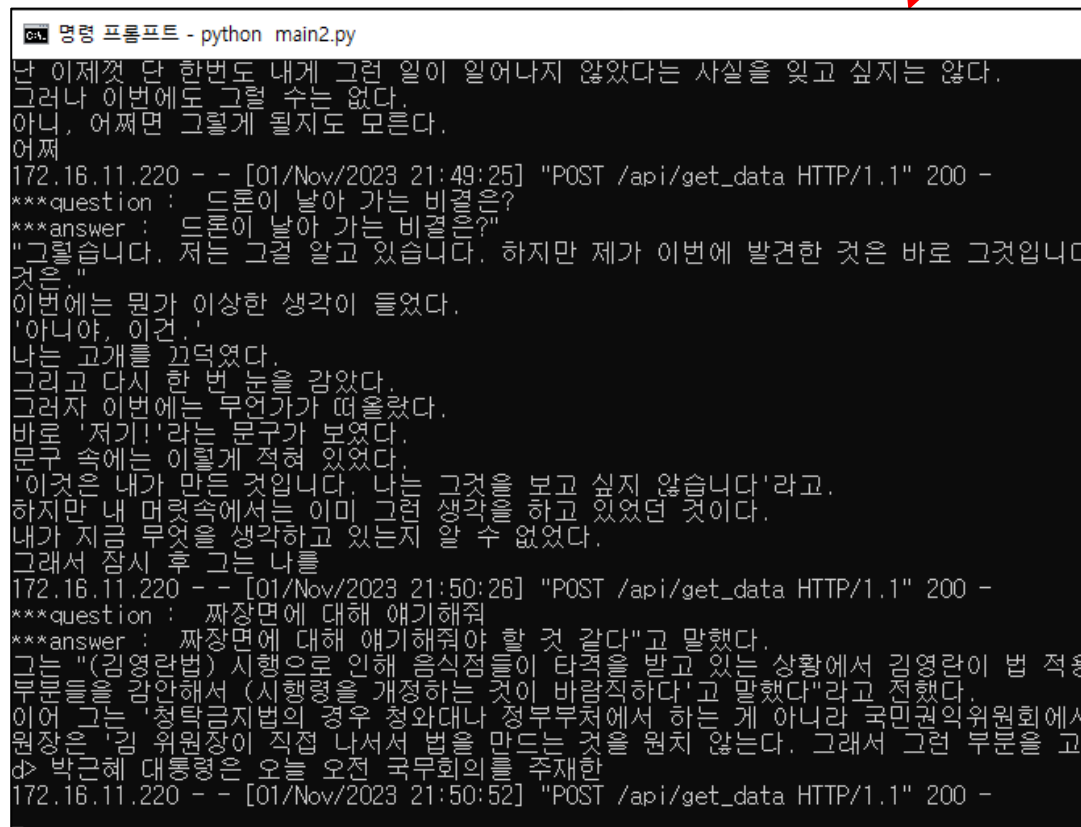
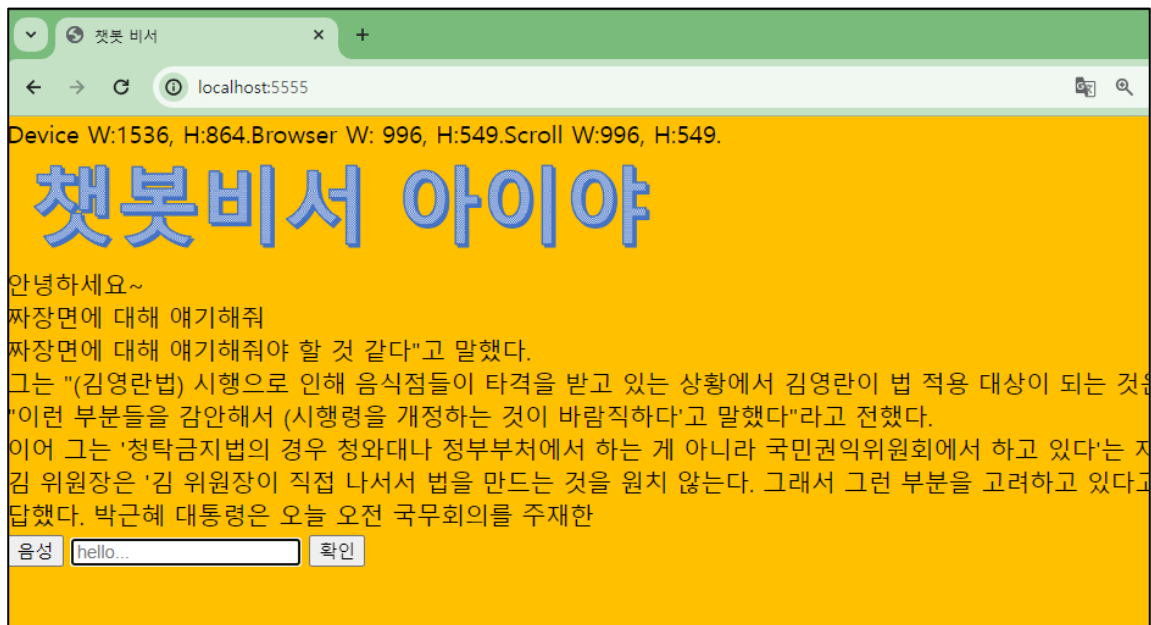
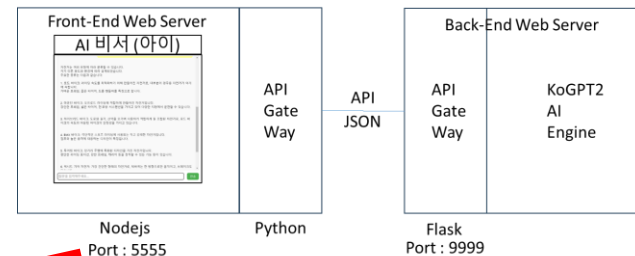
python main2.py



*. 전수업리뷰

⌚ 접속

http://localhost:5555/



*. 전달 사항



교재

주교재

- PowerPoint 로 만든 pdf 자료
- 데이터 과학 기반의 파이썬 빅데이터 분석 (이지영 지음, 한빛아카데미)

부교재

- 필요 시, 영상 공유



*. 전달 사항

RoadMap

Hadoop설치

- ✓ VM 셋업
- ✓ JDK
- ✓ Python
- ✓ Hadoop Engine
- ✓ Spark Engine
- ✓ Zeppelin

빅데이터분석

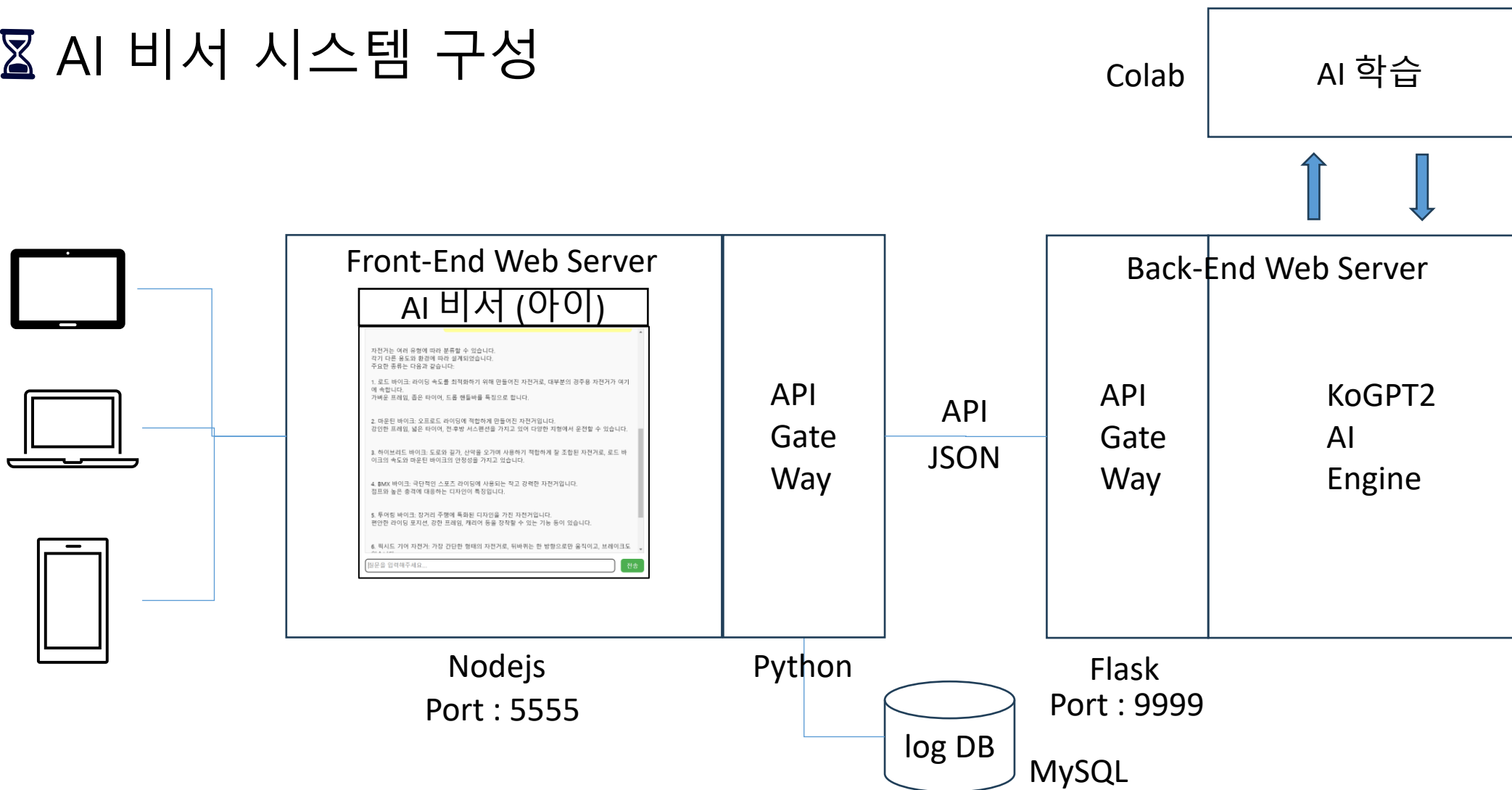
- ✓ 빅데이터 산업의 이해
- ✓ 파이썬 프로그래밍
- ✓ 크롤링
- ✓ 통계분석
- ✓ 텍스트빈도분석
- ✓ 지리정보분석
- ✓ 회귀분석/분류분석
- ✓ 텍스트마이닝

AI 비서학습

- ✓ 챗봇 데이터 수집
- ✓ Flask 웹서버
- ✓ Nodejs API 연동
- ✓ KoGPT2 환경구성
- ✓ Colab을 이용한 학습
- ✓ 말풍선생성기 활용
- ✓ MySQL
- ✓ 챗봇 비서 만들기

*. 전달 사항

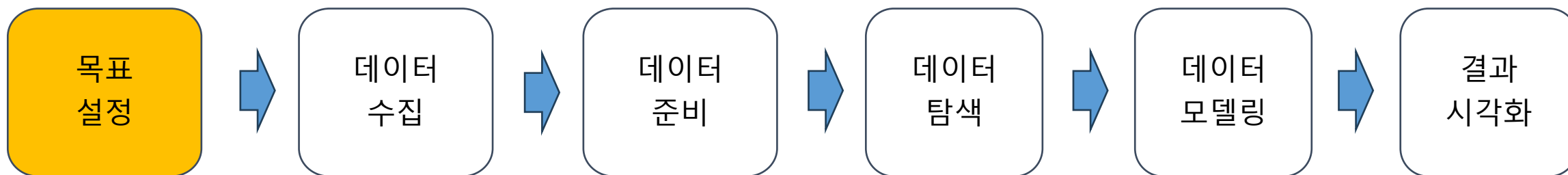
⌚ AI 비서 시스템 구성



1. 통계분석

⌚ 데이터과학

■ 절차



- 와인의 속성을 분석한 뒤 품질 등급을 예측하는 것
- 데이터의 기술 통계를 구함
- 레드 와인과 화이트 와인 그룹의 품질에 대한 t-검정을 수행
- 와인 속성을 독립 변수로, 품질 등급을 종속 변수로 선형 회귀 분석을 수행

1. 통계분석



⌚ 목표설정

■ 핵심 개념 이해

[기술통계]

- 데이터의 특성을 나타내는 수치를 이용하여 분석하는 기본적인 통계 방법
- 평균, 중앙값, 최빈값 등을 구할 수 있음

[회귀 분석]

- 독립변수 x 와 종속변수 y 간의 상호 연관성 정도를 파악하기 위한 분석 기법
- 하나의 변수가 변함에 따라 대응되는 변수가 어떻게 변하는지 측정하는 것
- 변수 간의 인과관계를 분석할 때 많이 사용

1. 통계분석



⌚ 목표설정

■ 핵심 개념 이해

[t-검정]

- 데이터에서 찾은 평균으로 두 그룹에 차이가 있는지 확인하는 방법

예) A 와인의 품질이 1등급인지 2등급인지에 따라 가격에 차이가 있는지 확인할 때 사용

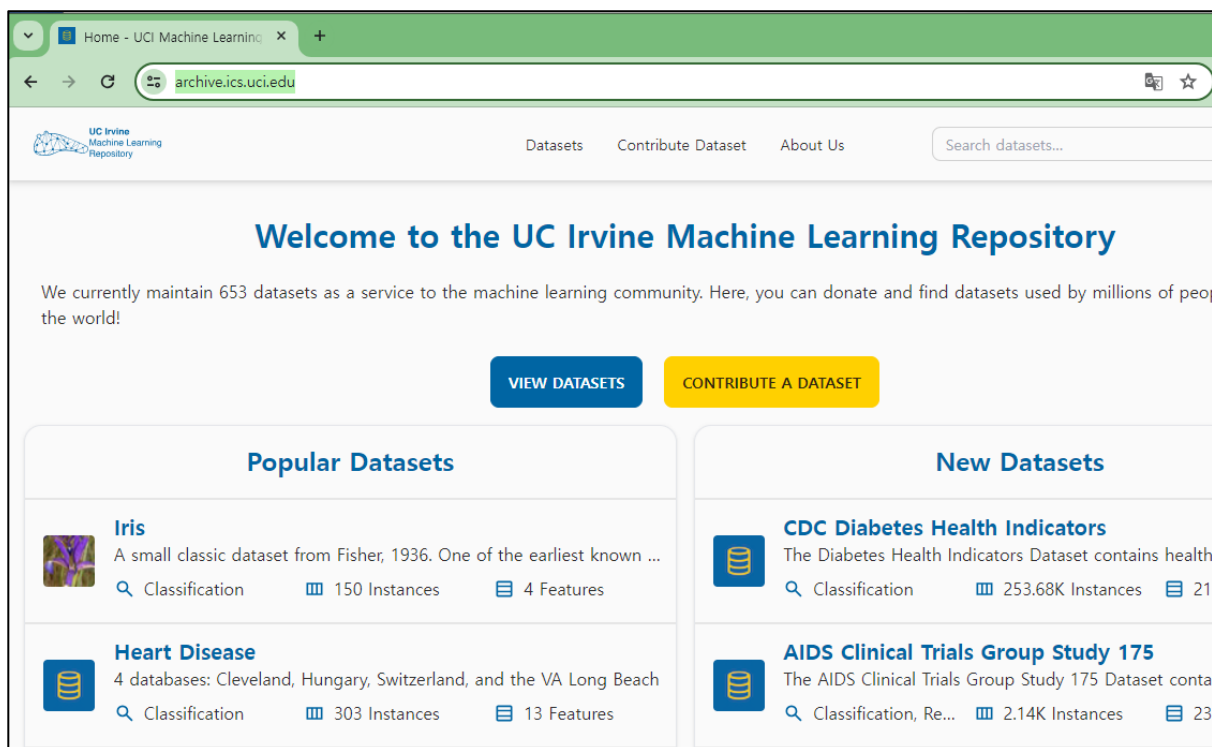
[히스토그램]

- 데이터 값의 범위를 몇 개 구간으로 나누고 각 구간에 해당하는 값의 숫자나 상대적 빈도 크기를 차트로 나타낸 것

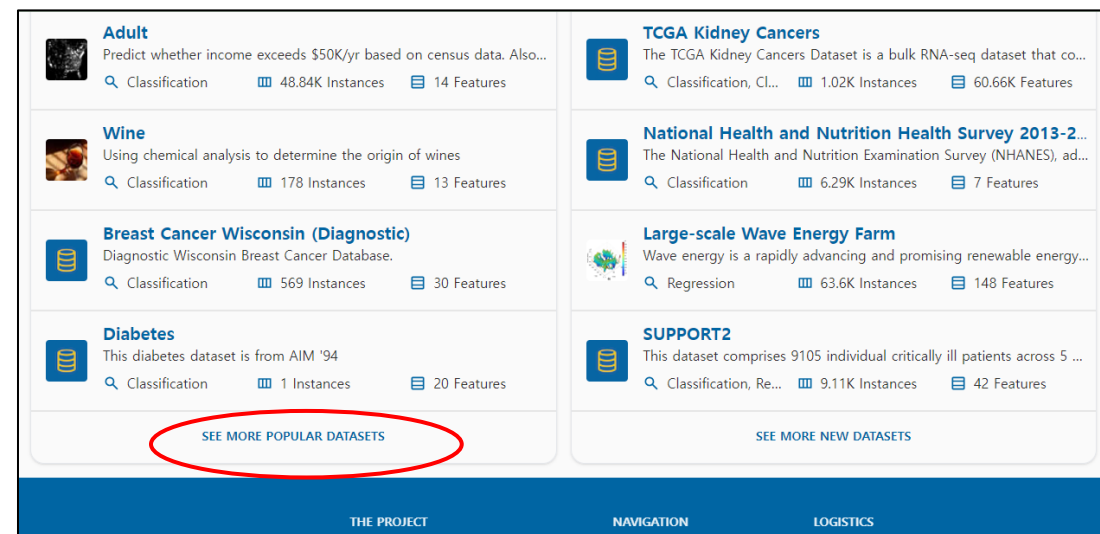
1. 통계분석

⌚ 데이터 수집

- 데이터셋 : 캘리포니아 어바인 대학의 머신러닝 저장소에서 제공



<https://archive.ics.uci.edu/>



1. 통계분석

⌚ 데이터 수집

- 데이터셋 : 캘리포니아 어바인 대학의 머신러닝 저장소에서 제공



Dry Bean Dataset
Images of 13,611 grains of 7 different registered dry beans were taken with a high-resolution camera. A tot...
Classification Multivariate 13.61K Instances 16 Features

Car Evaluation
Derived from simple hierarchical decision model, this database may be useful for testing constructive inducti...
Classification Multivariate 1.73K Instances 6 Features

Wine Quality
Two datasets are included, related to red and white vinho verde wine samples, from the north of Portugal. T...
Classification, Regre... Multivariate 4.9K Instances 12 Features

Rice (Cammeo and Osmancik)
A total of 3810 rice grain's images were taken for the two species, processed and feature inferences were m...
Classification Multivariate 3.81K Instances 7 Features

Rows per page 10 0 to 10 of 653

UC Irvine Machine Learning Repository

Datasets Contribute Dataset About Us Search datasets... Login

Wine Quality
Donated on 10/6/2009

Two datasets are included, related to red and white vinho verde wine samples, from the north of Portugal. The goal is to model wine quality based on physicochemical tests (see [Cortez et al., 2009], http://www3.dsi.uminho.pt/pcortez/wine/).

Dataset Characteristics	Subject Area	Associated Tasks
Multivariate	Business	Classification, Regression

Feature Type	# Instances	# Features
Real	4898	11

Dataset Information

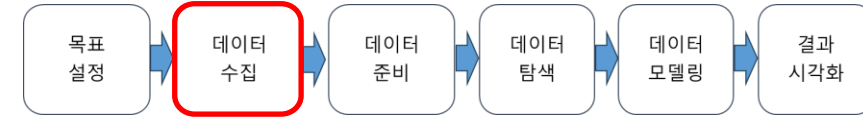
Additional Information
The two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. For more details, consult: http://www.vinhoverde.pt/en/ or the reference [Cortez et al., 2009]. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine ...

Download
IMPORT IN PYTHON
CITE

1 citations
129405 views

Keywords
Chemistry

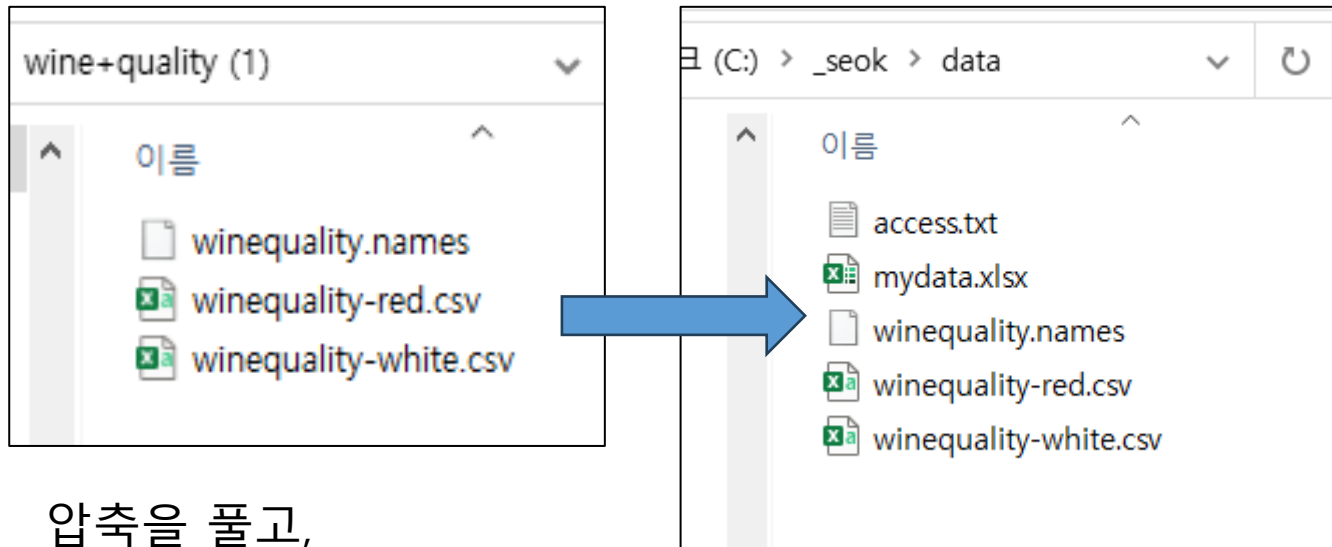
Creators
Paulo Cortez
A. Cerdeira
F. Almeida



1. 통계분석

⌚ 데이터 수집

- 데이터셋 : 캘리포니아 어바인 대학의 머신러닝 저장소에서 제공



압축을 풀고,

자신의 ₩_seok₩data 디렉토리로 이동



1. 통계분석

⌚ 데이터 준비

- 전처리-1 : 세미콜론(;) 으로 된 csv 파일을 읽어, xlsx 로 저장

```

#=====
# 전처리-1 : 세미콜론(;) 으로 된 csv 파일을 읽어, xlsx 로 저장
#=====
import pandas as pd

r_df = pd.read_csv('./winequality-red.csv', sep = ';', header = 0)
w_df = pd.read_csv('./winequality-white.csv', sep = ';', header = 0)

r_df.to_excel('./winequality-red2.xlsx', index=False)
w_df.to_excel('./winequality-white2.xlsx', index=False)
  
```

_seok\data\wine-01-preprocess.py 로 저장

1. 통계분석



⌚ 데이터 준비

- 전처리-1 : 세미콜론(;) 으로 된 csv 파일을 읽어, xlsx 로 저장

```

C:\_seok>명령 프롬프트
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

C:\_seok>cd _seok\data

C:\_seok\data>python wine-01-preprocess.py

C:\_seok\data>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: F6D9-BD25

C:\_seok\data 디렉터리

2023-11-06 오후 03:20 <DIR> .
2023-11-06 오후 03:20 <DIR> ..
2023-10-03 오전 02:07 10 access.txt
2023-10-03 오전 02:05 9,265 mydata.xlsx
2023-11-06 오후 03:19 407 wine-01-preprocess.py
2023-05-22 오후 10:24 84,199 winequality-red.csv
2023-11-06 오후 03:20 93,779 winequality-red2.xlsx
2023-05-22 오후 10:24 264,426 winequality-white.csv
2023-11-06 오후 03:20 278,171 winequality-white2.xlsx
2023-05-22 오후 10:24 3,305 winequality.names
                8개 파일              733,562 바이트
                2개 디렉터리 130,907,881,472 바이트 남음

C:\_seok\data>
  
```

cd _seok\data

python wine-01-preprocess.py

	A	B	C	D	E	F	G	H	I	J	K	L
1	winequality-red	winequality-white	winequality-red2	winequality-white2	winequality.names	access.txt	mydata.xlsx	wine-01-preprocess.py	winequality-red.csv	winequality-red2.xlsx	winequality-white.csv	winequality-white2.xlsx
2	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
3	7.8	0.88	0	2.6	0.098	25	67	0.9968	3.2	0.68	9.8	5
4	7.8	0.76	0.04	2.3	0.092	15	54	0.997	3.26	0.65	9.8	5
5	11.2	0.28	0.56	1.9	0.075	17	60	0.998	3.16	0.58	9.8	6
6	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
7	7.4	0.66	0	1.8	0.075	13	40	0.9978	3.51	0.56	9.4	5
8	7.9	0.6	0.06	1.6	0.069	15	59	0.9964	3.3	0.46	9.4	5
9	7.3	0.65	0	1.2	0.065	15	21	0.9946	3.39	0.47	10	7

	A	B	C	D	E	F	G	H	I	J	K	L
1	winequality-red	winequality-white	winequality-red2	winequality-white2	winequality.names	access.txt	mydata.xlsx	wine-01-preprocess.py	winequality-red.csv	winequality-red2.xlsx	winequality-white.csv	winequality-white2.xlsx
2	7	0.27	0.36	20.7	0.045	45	170	1.001	3	0.45	8.8	6
3	6.3	0.3	0.34	1.6	0.049	14	132	0.994	3.3	0.49	9.5	6
4	8.1	0.28	0.4	6.9	0.05	30	97	0.9951	3.26	0.44	10.1	6
5	7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.4	9.9	6
6	7.2	0.23	0.32	8.5	0.058	47	186	0.9956	3.19	0.4	9.9	6
7	8.1	0.28	0.4	6.9	0.05	30	97	0.9951	3.26	0.44	10.1	6
8	6.2	0.32	0.16	7	0.045	30	136	0.9949	3.18	0.47	9.6	6
9	7	0.27	0.36	20.7	0.045	45	170	1.001	3	0.45	8.8	6

winequality-red2.xlsx
winequality-white2.xlsx
파일이 생성됨

1. 통계분석

⌚ 데이터 준비

■ 전처리-2 : 파일 병합

```
#=====
# 전처리-2 : 파일 병합
#=====
import pandas as pd

r_df = pd.read_excel('./winequality-red2.xlsx')
w_df = pd.read_excel('./winequality-white2.xlsx')

r_df.insert(0, column = 'type', value = 'red')
w_df.insert(0, column = 'type', value = 'white')

wine = pd.concat([r_df, w_df])
wine.to_excel('./wine.xlsx', index=False)
```



`_seok\data\wine-02-pre-concat.py` 로 저장

`python wine-02-pre-concat.py` 실행

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	type	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	sulfur dioxide	sulfur dioxide	density	pH	sulphates	alcohol	quality
2	red	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
3	red	7.8	0.88	0	2.6	0.098	25	67	0.9968	3.2	0.68	9.8	5
4	red	7.8	0.76	0.04	2.3	0.092	15	54	0.997	3.26	0.65	9.8	5
5	red	11.2	0.28	0.56	1.9	0.075	17	60	0.998	3.16	0.58	9.8	6
6	red	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
7	red	7.4	0.66	0	1.8	0.075	13	40	0.9978	3.51	0.56	9.4	5
8	red	7.9	0.6	0.06	1.6	0.069	15	59	0.9964	3.3	0.46	9.4	5
9	red	7.3	0.65	0	1.2	0.065	15	21	0.9946	3.39	0.47	10	7
10	red	7.8	0.58	0.02	2	0.073	9	18	0.9968	3.36	0.57	9.5	7
11	red	7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5
12	red	6.7	0.58	0.08	1.8	0.097	15	65	0.9959	3.28	0.54	9.2	5
13	red	7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5

하나로 합쳐진,
`wine.xlsx`
파일이 생성됨

1. 통계분석

⌚ 데이터 탐색

■ 기본정보 확인

`_seok\data\wine-03-navi-data.py` 로 저장

```
#=====
# 데이터탐색 : 종속변수 값 확인
#=====
import pandas as pd

wine = pd.read_excel('./wine.xlsx')

print(wine.info())
print('=' * 40)
```

- 전체 샘플은 6,497개, 속성을 나타내는 열은 13개, 각 속성의 이름은 type부터 quality까지
- 실수 타입(float64)은 11개, 정수 타입(int64)은 1개(quality), 객체 타입(object)이 1개(type)
- 독립 변수(x)는 type부터 alcohol 까지 12개, 종속 변수(y)는 1개(quality)



<class 'pandas.core.frame.DataFrame'>

RangeIndex: 6497 entries, 0 to 6496

Data columns (total 13 columns):

#	Column	Non-Null Count	Dtype
0	type	6497 non-null	object
1	fixed acidity	6497 non-null	float64
2	volatile acidity	6497 non-null	float64
3	citric acid	6497 non-null	float64
4	residual sugar	6497 non-null	float64
5	chlorides	6497 non-null	float64
6	free sulfur dioxide	6497 non-null	float64
7	total sulfur dioxide	6497 non-null	float64
8	density	6497 non-null	float64
9	pH	6497 non-null	float64
10	sulphates	6497 non-null	float64
11	alcohol	6497 non-null	float64
12	quality	6497 non-null	int64

dtypes: float64(11), int64(1), object(1)

memory usage: 660.0+ KB

None

1. 통계분석

⌚ 데이터 탐색

■ 기술 통계 구하기

<계속>

```
wine.columns = wine.columns.str.replace(' ', '_')
print(wine.head())
print('=' * 40)

print(wine.describe())
print('=' * 40)
```

열이름에 공백이 있으면, 밑줄로 연결

type fixed_acidity volatile_acidity

=====

	fixed_acidity	volatile_acidity	
count	6497.000000	6497.000000	Null 이 아닌 값의 개수
mean	7.215307	0.339164	열의 평균 값
std	1.296434	0.164633	표준편차
min	3.800000	0.080000	열의 최소값
25%	6.400000	0.230000	하위 25%
50%	7.000000	0.290000	데이터정렬 중간 값
75%	7.700000	0.400000	상위 25%
max	15.900000	1.580000	열의 최대 값

[8 rows x 12 columns]

1. 통계분석

⌚ 데이터 탐색

■ 기술 통계 구하기

<계속>

```
print(sorted(wine.quality.unique()))
print('=' * 40)

wine.quality.value_counts()
print('=' * 40)
```

- wine.quality.unique() 함수를 사용하여 quality 속성값 중에서 유일한 값을 출력
이를 통해 와인 품질 등급quality은 3, 4, 5, 6, 7, 8, 9의 7개 등급이 있다는 것을 알 수 있음
- quality.value_counts() 함수는 quality 속성값에 대한 빈도수를 보여줌
6등급인 샘플이 2,836개로 가장 많고, 9등급인 샘플이 5개로 가장 적은 것을 알 수 있음



_seok\data\wine-03-navi-data.py 로 저장

[3, 4, 5, 6, 7, 8, 9]

```
=====
quality
6    2836
5    2138
7    1079
4     216
8     193
3      30
9       5
Name: count, dtype: int64
=====
```



1. 통계분석

⌚ 데이터 모델링

- 그룹 비교 `python -m pip install statsmodels`
`_seok\data\wine-04-modeling.py` 로 저장

```

#=====
# 데이터모델링 :
#=====
import pandas as pd

wine = pd.read_excel('./wine.xlsx')
wine.columns = wine.columns.str.replace(' ', '_')

print(wine.groupby('type')['quality'].describe())
print('=' * 40)

print(wine.groupby('type')['quality'].agg(['mean', 'std']))
print('=' * 40)

```

```

count    mean    std min 25% 50% 75% max
type
red    1599.0  5.636023  0.807569  3.0  5.0  6.0  6.0  8.0
white  4898.0  5.877909  0.885639  3.0  5.0  6.0  6.0  9.0
=====

```

```

mean    std
type
red    5.636023  0.807569
white  5.877909  0.885639
=====

```

- 레드 와인과 화이트 와인을 구분하는 속성인 type을 기준으로 그룹을 나눈 뒤 그룹 안에서 quality 속성을 기준으로 기술 통계를 구함
- mean() 함수로 평균만 구하거나 std() 함수로 표준편차만 따로 구할 수도 있으며, agg() 함수를 사용하여 두개를 동시에 나타낼 수 있음

1. 통계분석

⌚ 데이터 모델링

■ 그룹 비교

<계속>

```
#-----
# t-검정
#-----
from scipy import stats
from statsmodels.formula.api import ols, glm
r_wine_quality = wine.loc[wine['type'] == 'red', 'quality']
w_wine_quality = wine.loc[wine['type'] == 'white', 'quality']

print(stats.ttest_ind(r_wine_quality, w_wine_quality, equal_var = False))
print('=' * 40)
```

```
Ttest_indResult(
statistic=-10.149363059143164,
pvalue=8.168348870049682e-24)
=====
```

- t-검정에 필요한 scipy 패키지의 stats 함수와 회귀 분석에 필요한 statsmodels.formula.api 패키지의 ols, glm 함수를 로드
- 레드 와인 샘플의 quality 값만 찾아서 r_wine_quality에 저장, 화이트 와인 샘플의 quality 값만 찾아서 w_wine_quality에 저장
- scipy 패키지의 stats.ttest_ind() 함수를 사용하여 t-검정을 하고 두 그룹 간 차이를 확인



1. 통계분석

⌚ 데이터 모델링

■ 그룹 비교

<계속>

```
#-----
# 선형회귀분석
#-----

Rformula = 'quality ~ fixed_acidity + volatile_acidity + citric_acid +
residual_sugar + chlorides + free_sulfur_dioxide + total_sulfur_dioxide
+ density + pH + sulphates + alcohol'

regression_result = ols(Rformula, data = wine).fit()
print(regression_result.summary())
print('=' * 40)
```

- 선형 회귀 분석식의 종속 변수(y)와 독립 변수(x1~x10)를 구성 ➔ 종속변수(quality) ~ 독립변수(나머지 +)
- 선형 회귀 모델 중에서 OLS(Ordinary Least Squares) 모델을 사용
- 선형 회귀 분석과 관련된 통계값을 출력
- 하단에 나타난 오류는 현재 수업 과정에서는 무시하기로 함



OLS Regression Results

```
=====
Dep. Variable:    quality  R-squared:    0.292
Model:            OLS  Adj. R-squared:    0.291
Method:           Least Squares  F-statistic:    243.3
Date:            Mon, 06 Nov 2023  Prob (F-statistic):    0.00
Time:            16:55:38  Log-Likelihood:    -7215.5
No. Observations:    6497  AIC:            1.445e+04
Df Residuals:        6485  BIC:            1.454e+04
Df Model:            11
Covariance Type:    nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	55.7627	11.894	4.688	0.000	32.447	79.079
fixed_acidity	0.0677	0.016	4.346	0.000	0.037	0.098
volatile_acidity	-1.3279	0.077	-17.162	0.000	-1.480	-1.176
citric_acid	-0.1097	0.080	-1.377	0.168	-0.266	0.046
residual_sugar	0.0436	0.005	8.449	0.000	0.033	0.054
chlorides	-0.4837	0.333	-1.454	0.146	-1.136	0.168
free_sulfur_dioxide	0.0060	0.001	7.948	0.000	0.004	0.007
total_sulfur_dioxide	-0.0025	0.000	-8.969	0.000	-0.003	-0.002
density	-54.9669	12.137	-4.529	0.000	-78.760	-31.173
pH	0.4393	0.090	4.861	0.000	0.262	0.616
sulphates	0.7683	0.076	10.092	0.000	0.619	0.917
alcohol	0.2670	0.017	15.963	0.000	0.234	0.300

```
=====
Omnibus:            144.075  Durbin-Watson:            1.646
Prob(Omnibus):      0.000  Jarque-Bera (JB):      324.712
Skew:               -0.006  Prob(JB):              3.09e-71
Kurtosis:           4.095  Cond. No.              2.49e+05
=====
```

Notes:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
 [2] The condition number is large, 2.49e+05. This might indicate that there are strong multicollinearity or other numerical problems.

1. 통계분석

⌚ 데이터 모델링

■ 품질등급 예측

<계속> _seok\data\wine-04-modeling.py 로 저장

```
#-----
# 품질등급 예측
#-----
sample1 = wine[wine.columns.difference(['quality', 'type'])]
sample1 = sample1[0:5][:]
print(sample1)
print('=' * 40)

sample1_predict = regression_result.predict(sample1)
print(sample1_predict)
print('=' * 40)

print(wine[0:5]['quality'])
print('=' * 40)
```



```
alcohol chlorides citric_acid density fixed_acid
volatile_acidity
0 9.4 0.076 0.00 0.9978 7.4 ...
1 9.8 0.098 0.00 0.9968 7.8 ...
2 9.8 0.092 0.04 0.9970 7.8 ...
3 9.8 0.075 0.56 0.9980 11.2 ...
4 9.4 0.076 0.00 0.9978 7.4 ...
```

[5 rows x 11 columns]

```
0 4.997607
1 4.924993
2 5.034663
3 5.680333
4 4.997607
```

dtype: float64

```
0 5
1 5
2 5
3 6
4 5
```

Name: quality, dtype: int64

- 예측에 사용할 첫 번째 샘플 데이터 만들기
 - wine에서 quality와 type 열은 제외하고, 회귀 분석 모델에 사용할 독립 변수만 추출하여 sample1에 저장
 - sample1에 있는 샘플 중에서 0번~4번 5개 샘플만 추출하고, sample1에 다시 저장하여 예측에 사용할 샘플을 제작
- 첫 번째 샘플의 quality 예측하기
 - 샘플 데이터를 회귀 분석 모델 regression_result의 예측 함수 predict()에 적용하여, 결과 예측값을 sample1_predict에 저장
 - sample1_predict를 출력하여 예측한 quality를 확인
- wine에서 0번부터 4번까지 샘플의 quality 값을 출력하여 sample1_predict이 맞게 예측되었는지 확인

1. 통계분석



⌚ 결과 시각화

`python -m pip install seaborn`

- 히스토그램 : 와인유형에 따른 품질등급

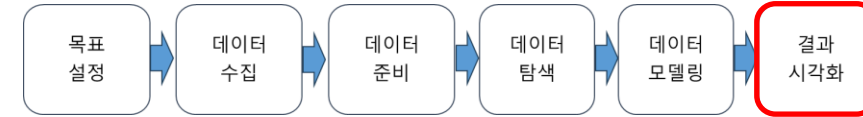
`_seok\data\wine-05-view-1.py` 로 저장

```

#=====
# 결과시각화 : 히스토그램
#=====
import pandas as pd
from statsmodels.formula.api import ols, glm
import matplotlib.pyplot as plt
import seaborn as sns

wine = pd.read_excel('./wine.xlsx')
wine.columns = wine.columns.str.replace(' ', '_')
r_wine_quality = wine.loc[wine['type'] == 'red', 'quality']
w_wine_quality = wine.loc[wine['type'] == 'white', 'quality']
  
```

- seaborn 라이브러리 패키지를 설치하고 import



1. 통계분석

⌚ 결과 시각화

- 히스토그램 : 와인유형에 따른 품질등급

<계속>

```

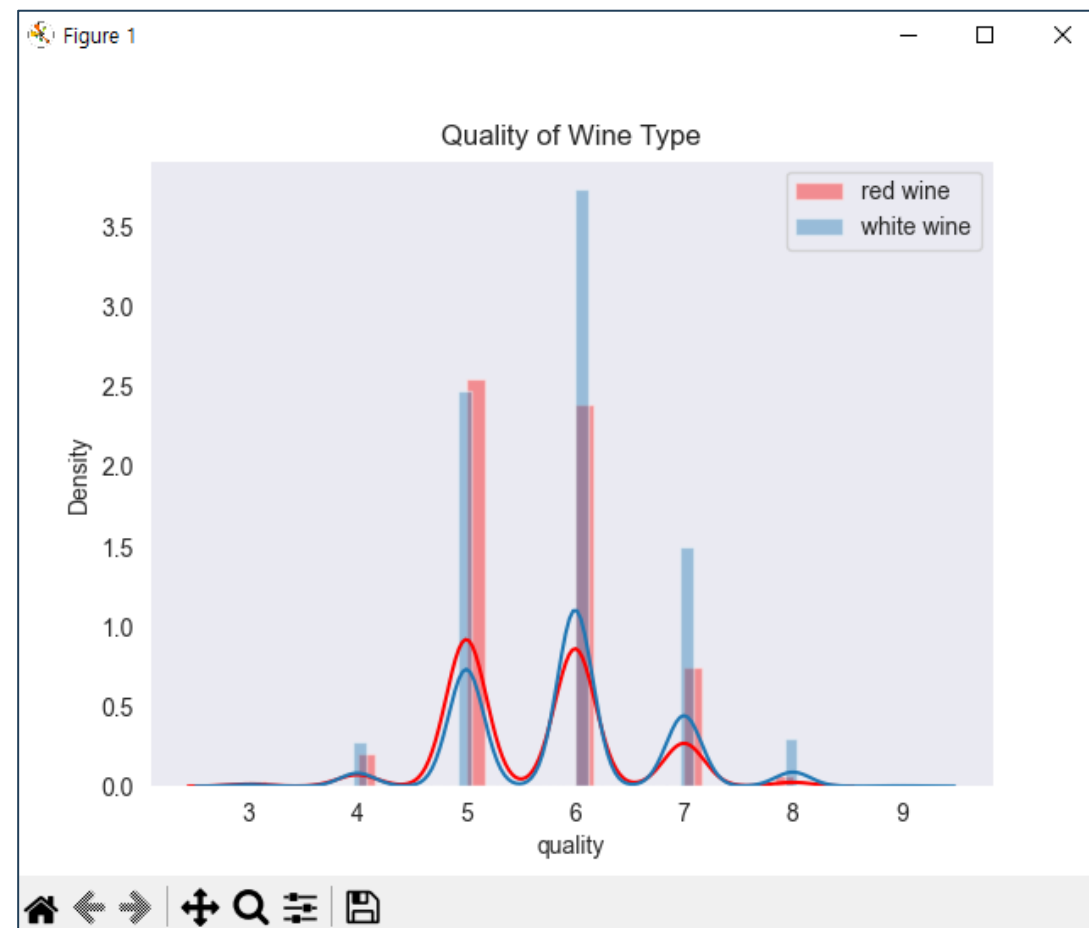
#-----
# 히스토그램 그리기
#-----
sns.set_style('dark')
sns.distplot(r_wine_quality, kde = True, color = "red", label = 'red wine')
sns.distplot(w_wine_quality, kde = True, label = 'white wine')
plt.title("Quality of Wine Type")
plt.legend()
plt.show()
  
```

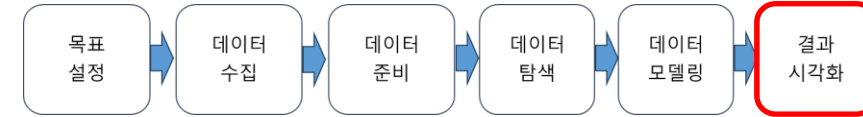
- 히스토그램 차트의 배경색 스타일을 설정
- 레드 와인에 대한 distplot 객체를 생성
- 화이트 와인에 대한 distplot 객체를 생성
- 차트 제목을 설정
- 차트 범례를 설정
- 설정한 내용대로 차트를 표시

1. 통계분석

⌚ 결과 시각화

- 히스토그램 : 와인유형에 따른 품질등급





1. 통계분석

⌚ 결과 시각화

- PLOT : 하나의 독립변수가 종속변수에 미치는 영향

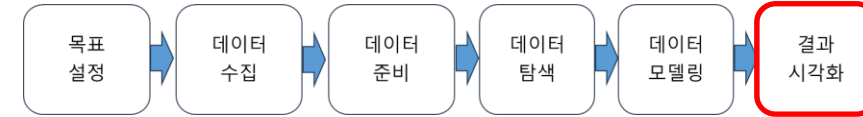
[_seok\data\wine-05-view-2.py](#) 로 저장

```

#=====
# 결과시각화 : 부분 회귀 플롯
#=====
import pandas as pd
from statsmodels.formula.api import ols, glm
import matplotlib.pyplot as plt
import statsmodels.api as sm

wine = pd.read_excel('./wine.xlsx')
wine.columns = wine.columns.str.replace(' ', '_')

# 선형회귀분석
Rformula = 'quality ~ fixed_acidity + volatile_acidity + citric_acid + residual_sugar + chlorides + free_sulfur_dioxide +
total_sulfur_dioxide + density + pH + sulphates + alcohol'
regression_result = ols(Rformula, data = wine).fit()
  
```



1. 통계분석

⌚ 결과 시각화

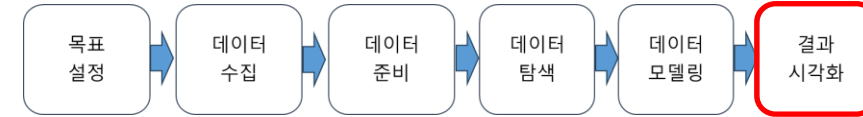
- PLOT : 하나의 독립변수가 종속변수에 미치는 영향

<계속>

```
#-----
# 부분 회귀 플롯 그리기
#-----
#others = list(set(wine.columns).difference(set(["quality", "fixed_acidity"])))
#p, resid = sm.graphics.plot_partregress("quality", "fixed_acidity", others, data = wine, ret_coords = True)
#plt.show()
```

```
fig = plt.figure(figsize = (10, 10))
sm.graphics.plot_partregress_grid(regression_result, fig = fig)
plt.show()
```

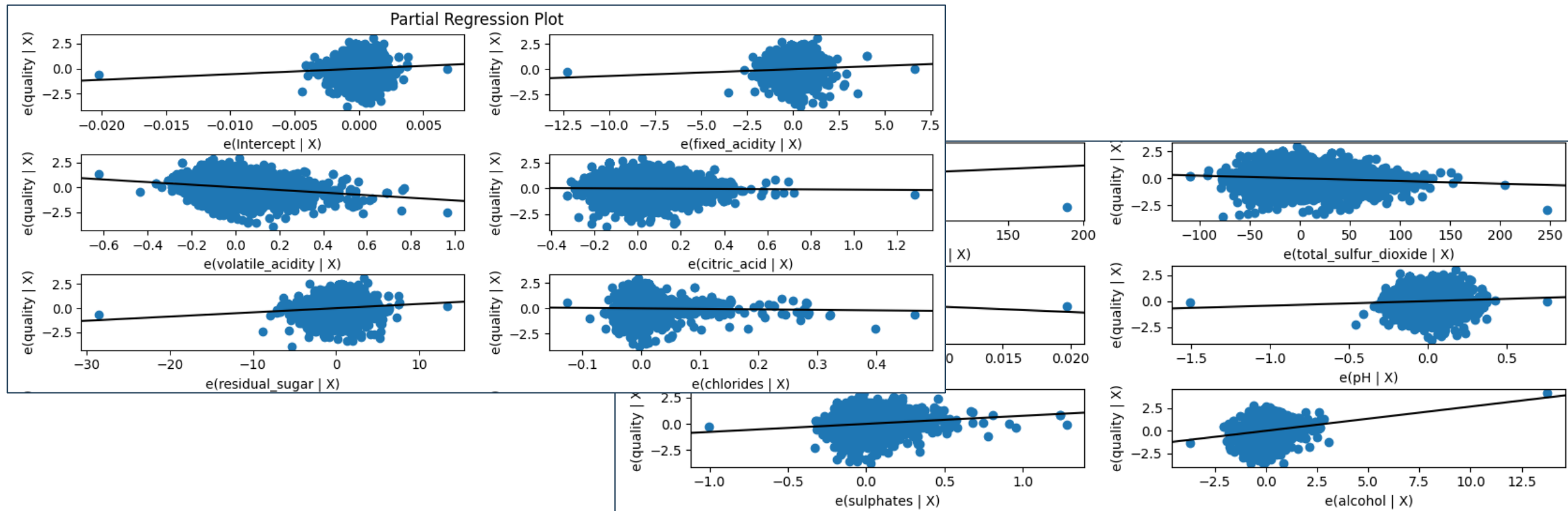
- fixed_acidity가 종속 변수 quality에 미치는 영향력을 시각화하기
 - 독립 변수와 종속 변수를 제외한 나머지 변수 이름을 리스트 others로 추출
 - 나머지 변수는 고정하고 fixed_acidity가 종속 변수 quality에 미치는 영향에 부분회귀를 수행
- 각 독립 변수가 종속 변수 quality에 미치는 영향력을 시각화하기
 - 차트의 크기를 지정하고, 다중 선형 회귀 분석 결과를 가지고 있는 regression_result를 이용해 각 독립 변수의 부분 회귀 플롯을 구함



1. 통계분석

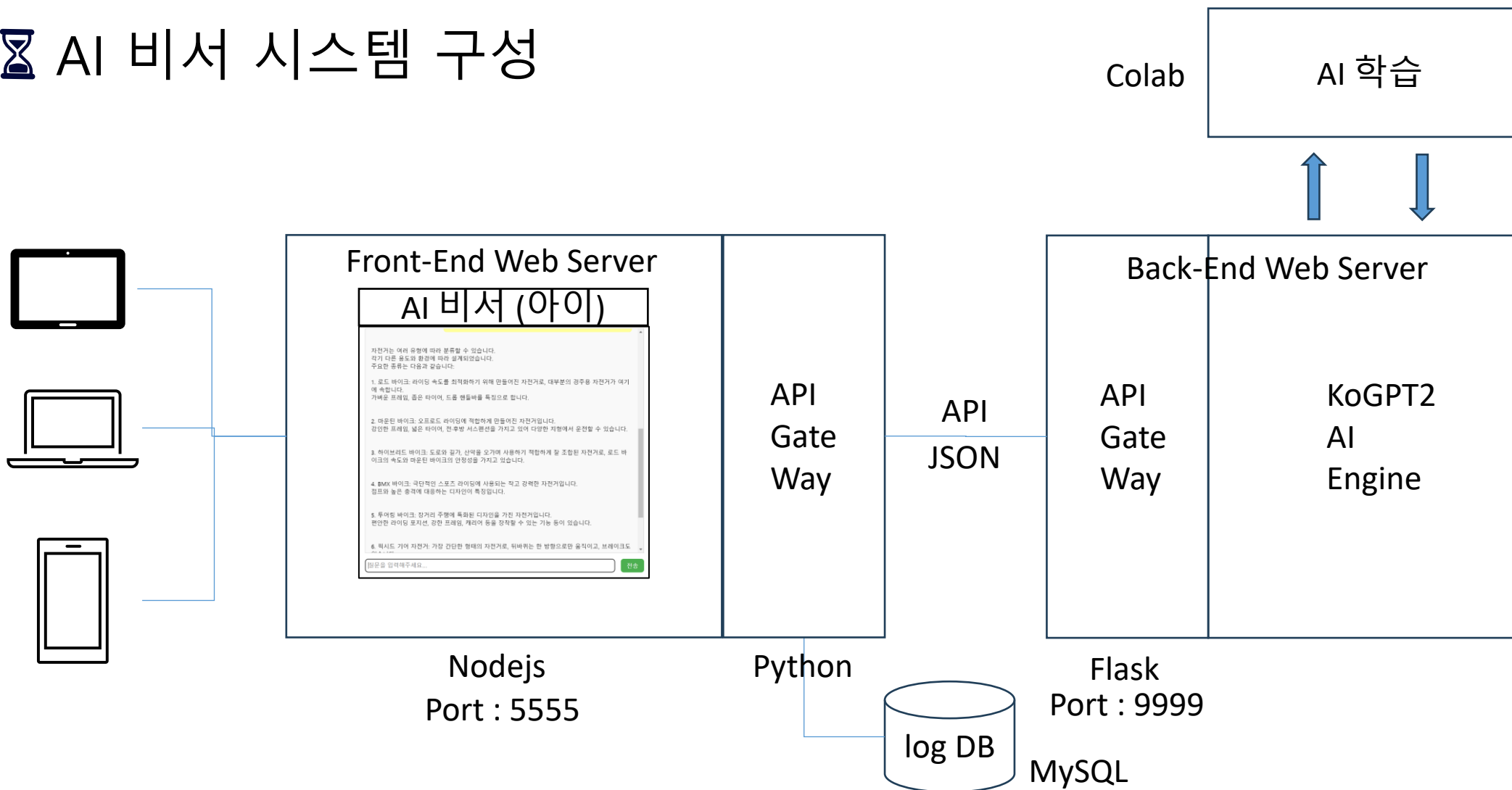
⌚ 결과 시각화

- PLOT : 하나의 독립변수가 종속변수에 미치는 영향



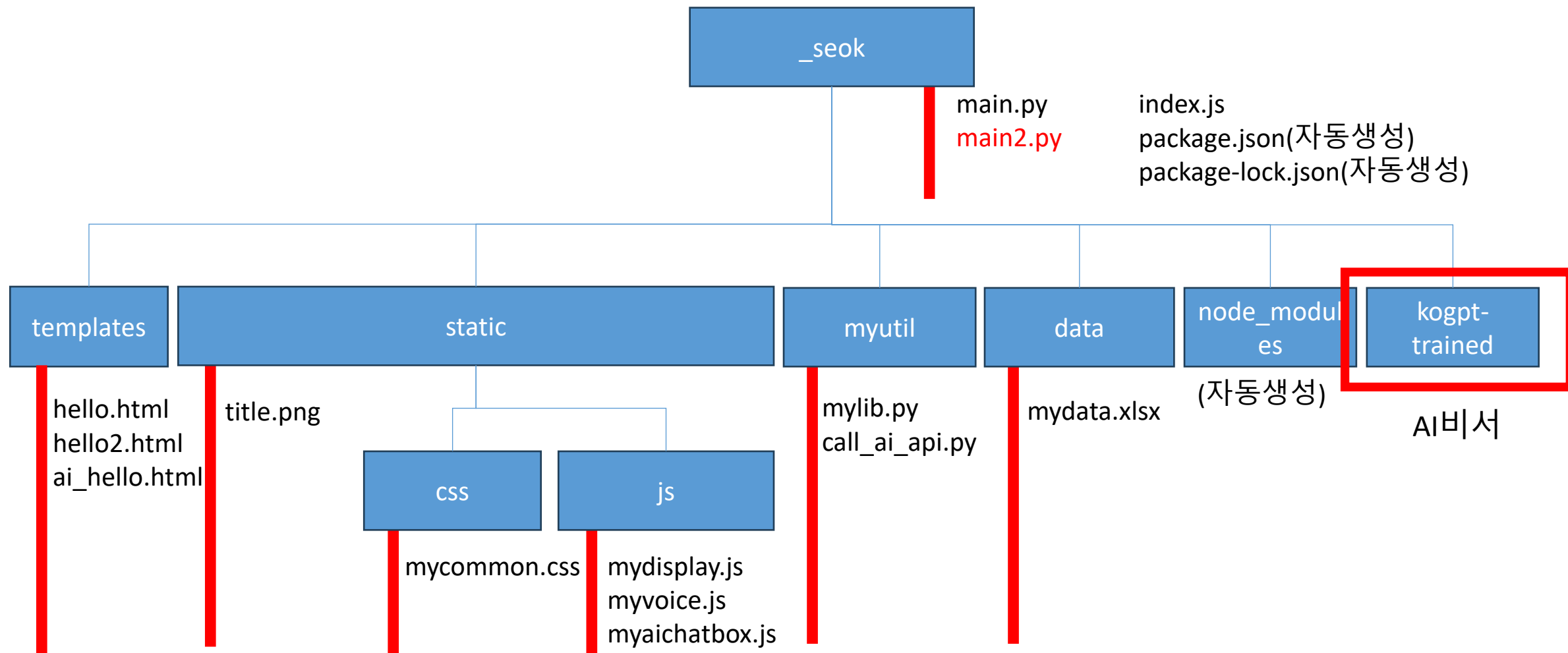
2. AI 비서 학습

⌚ AI 비서 시스템 구성



2. AI 비서 학습

⌚ 디렉토리 구성



2. AI 비서 학습

⌚ 학습 데이터 셋 생성

■ 질문/답변으로 구성

- ✓ 엑셀로 작성
- ✓ 질문은 prompt
- ✓ 답변은 completion
- ✓ 유사질문 10개당
답변 1개로 구성

	A	B
1	prompt	completion
2	치안의 주체는?	경찰청입니다
3	민생치안의 주체는?	경찰청입니다
4	나는 누구입니까?	홍길동입니다
5	나의 이름은 무엇입니까?	홍길동입니다
6	나를 무엇이라고 부릅니까?	홍길동입니다
7	나는 무엇을 하는 사람입니까?	민생치안을 돕는 사람입니다
8	제 이름은 홍길동입니다. 저를 불러 봐 주세요	안녕하세요, 홍길동 님! 저는 여러분의 도움이 되는 민생치안 비서입니다. 무엇을 도와드릴까요?
9	제 이름은 홍길동입니다. 민생치안을 돕는 일꾼입니다. 저를 불러 주세요	안녕하세요, 홍길동님! 홍길동 님을 위해 어떻게 도와드릴까요? 무엇이 필요하시거나 어떤 도움이 필요 한지 알려주시면 기꺼이 도와드리겠습니다.
10	어떤 방법으로 집 안전을 강화할 수 있을까요?	집 안전을 강화하기 위해 방범용품(카메라, 경보기 등)을 설치하고 현관문과 창문 잠금장치를 강화하는 것이 좋습니다. 또한, 이웃과의 소통을 통해 주변 상황을 주시하고 방법에 도움을 요청할 수도 있습니다
11	방범용품(카메라, 경보기 등)을 설치하는 것이 집 안전을 강화하는 좋은 방법일까요?	집 안전을 강화하기 위해 방범용품(카메라, 경보기 등)을 설치하고 현관문과 창문 잠금장치를 강화하는 것이 좋습니다. 또한, 이웃과의 소통을 통해 주변 상황을 주시하고 방법에 도움을 요청할 수도 있습니다
12	집 안전을 강화하기 위해 어떤 방범용품을 사용하는 것이 가장 효과적일까요?	집 안전을 강화하기 위해 방범용품(카메라, 경보기 등)을 설치하고 현관문과 창문 잠금장치를 강화하는 것이 좋습니다. 또한, 이웃과의 소통을 통해 주변 상황을 주시하고 방법에 도움을 요청할 수도 있습니다
13	현관문과 창문 잠금장치를 강화하는 것이 집 안전을 증가시키는데 도움이 될까요?	집 안전을 강화하기 위해 방범용품(카메라, 경보기 등)을 설치하고 현관문과 창문 잠금장치를 강화하는 것이 좋습니다. 또한, 이웃과의 소통을 통해 주변 상황을 주시하고 방법에 도움을 요청할 수도 있습니다
14	이웃과 소통을 하는 것이 집 안전을 강화하는 데 도움이 될까요?	집 안전을 강화하기 위해 방범용품(카메라, 경보기 등)을 설치하고 현관문과 창문 잠금장치를 강화하는 것이 좋습니다. 또한, 이웃과의 소통을 통해 주변 상황을 주시하고 방법에 도움을 요청할 수도 있습니다
15	주변 상황을 주시하는 것이 집 안전을 강화하는 데 어떤 영향을 주나요?	집 안전을 강화하기 위해 방범용품(카메라, 경보기 등)을 설치하고 현관문과 창문 잠금장치를 강화하는 것이 좋습니다. 또한, 이웃과의 소통을 통해 주변 상황을 주시하고 방법에 도움을 요청할 수도 있습니다
	방범용품을 설치하고 현관문과 창문 잠금장치를 강화하는 것 외에 집 안전을 강화	집 안전을 강화하기 위해 방범용품(카메라, 경보기 등)을 설치하고 현관문과 창문 잠금장치를 강화하는 것이 좋습니다. 또한, 이웃과의 소통을 통해 주변 상황을 주시하고 방법에 도움을 요청할 수도 있습니다

2. AI 비서 학습

⌚ 학습 데이터 셋 생성

■ 유사질문 만들기 : chatgpt 활용

아래의 질문과 응답을 읽어 보고, 질문에 대한 응답을 참고하여, 질문과 동일한 의미를 가지는 질문형식으로 10개정도 만들어 주세요

질문: ""

교통 안전을 지키기 위한 팁이 있을까요?

""

응답: ""

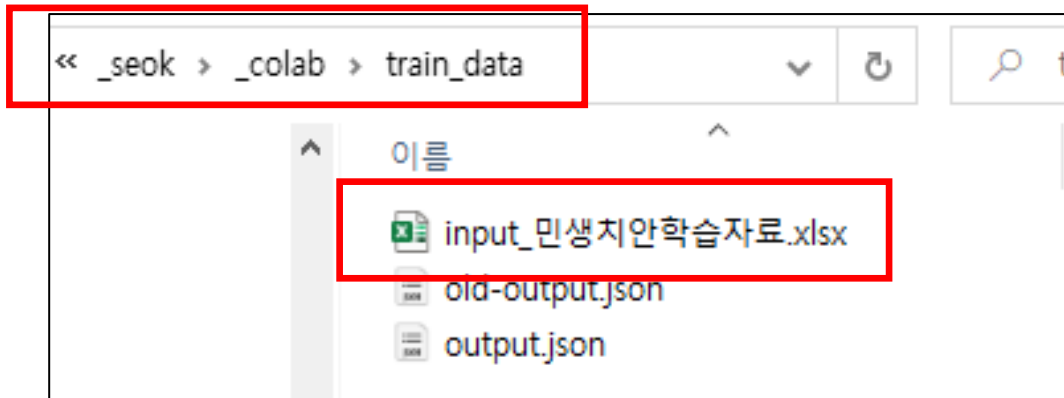
교통 안전을 지키기 위해 교통 규칙을 준수하는 것이 가장 중요합니다. 또한, 안전 운전을 위해 휴대전화 사용을 자제하고, 안전벨트를 착용하고, 근본적인 원인인 술이나 졸음 운전을 하지 않는 것도 필요합니다.

""

2. AI 비서 학습

⌚ 학습 데이터 셋 생성

■ 파일 저장



_seok_colab\train_data 생성한 후, 엑셀 파일 저장

2. AI 비서 학습

⌚ 학습 데이터 셋 생성

■ 학습용 json 생성

```

4칸 8칸
#=====
# 학습을 위한 json 데이터셋 생성
#=====
import pandas as pd
import re
import json
#-----
# 전처리 함수 정의
def preprocess(text):
    if type(text) != str:
        text = '비어있는 데이터'

    # 한글, 영문, 숫자 및 공백을 제외한 모든 문자 제거
    text = re.sub(r'^ㄱ-ㅎㅏ-ㅣ가-힣0-9a-zA-Z\s]', '', text)
    # 연속된 공백을 하나로 만들고 문자열 양 끝의 공백 제거
    text = re.sub(r'\s+', ' ', text).strip()

    return text

```

2. AI 비서 학습

⌚ 학습 데이터 셋 생성

■ 학습용 json 생성

```

#-----
# JSON 파일 작성
def write_json(data, file_path):
    with open(file_path, 'w', encoding='utf-8') as file:
        json.dump(data, file, ensure_ascii=False, indent=4)
#-----
# 메인 함수 정의
def main(input_xlsx, output_json):
    df = pd.read_excel (input_xlsx)
    print('=====')
    print (df)
    print('=====')

    # 전처리
    df['prompt'] = df['prompt'].apply(preprocess)
    df['completion'] = df['completion'].apply(preprocess)

    _sentences = []
  
```

2. AI 비서 학습

⌚ 학습 데이터 셋 생성

■ 학습용 json 생성

_seok_colab\1.make-json.py 파일 저장

```

4칸 8칸
for index, row in df.iterrows():
    p, c = row['prompt'], row['completion']
    _sen = []
    _sen.append((p, c))          # 쌍으로 묶어 준다
    _sentences.extend(_sen)

# 결과 저장
_data = [{'question': q, 'answer': a} for q, a in _sentences]
write_json(_data, output_json)

#-----
# 메인 함수 실행
if __name__ == '__main__':
    input_xlsx = './train_data/input_민생치안학습자료.xlsx'
    output_json = './train_data/output.json'

    main(input_xlsx, output_json)

```

2. AI 비서 학습

⌚ 학습 데이터 셋 생성

■ 학습용 json 생성

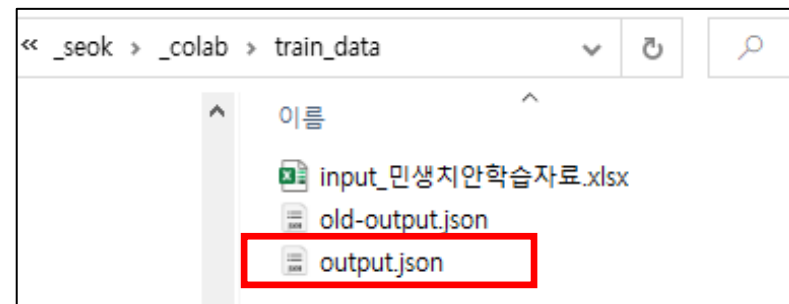
```

CA: 명령 프롬프트
(c) Microsoft Corporation. All rights reserved.
C:\Users\wdossa>cd \_seok\_colab
C:\_seok\_colab>python 1.make-json.py
=====
prompt
치안의 주체는?
민생치안의 주체는?
나는 누구입니까?
나의 이름은 무엇입니까?
다
나를 무엇이라고 부릅니까?
다
...
110 교통 안전을 위해 안전벨트 착용의 중요성은 무엇인가요? 교통 안전을
하는 것이 가장 중요합니다. 또한, 안전 ...
111 교통 안전을 위해 술이나 졸음 운전을 하지 않는 것이 왜 중요한가요? 교통 인
준수하는 것이 가장 중요합니다. 또한, 안전 ...
112 어떻게 하면 교통 안전을 위해 다른 운전자들과의 커뮤니케이션을 원활하게 할 수 있을까요?
통 규칙을 준수하는 것이 가장 중요합니다. 또한, 안전 ...
113 교통 안전을 위해 보행자로서 우리가 지켜야 할 규칙은 무엇인가요? 교통 인
준수하는 것이 가장 중요합니다. 또한, 안전 ...
114 교통 안전을 위해 우리가 주행 시에 고려해야 할 요소는 어떤 것들이 있을까요? 교
을 준수하는 것이 가장 중요합니다. 또한, 안전 ...
[115 rows x 2 columns]
=====
C:\_seok\_colab>

```

cd _seok_colab

python 1.make-json.py



output.json 이 생성됨

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

```

=====
# kogpt 학습
=====
import os
import json
import pandas as pd
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
from torch.utils.data import DataLoader, TensorDataset
from sklearn.model_selection import train_test_split
from transformers import PreTrainedTokenizerFast

# GPU가 사용 가능한 경우 사용하고, 그렇지 않으면 CPU를 사용하도록 설정
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print('device = ', device)

# epoch 크기 조정 : 학습량
EPOCH = 1 # 1 ~ 1000 값
    
```

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

```
# 읽어 올 데이터셋
DATA_SET_JSON = './train_data/output.json'
# 학습한 모델 저장하는 폴더명
SAVED_MODEL_NAME = './kogpt-trained'

# KoGPT-2 모델 및 토크나이저 로드
model = GPT2LMHeadModel.from_pretrained('skt/kogpt2-base-v2')
tokenizer = PreTrainedTokenizerFast.from_pretrained("skt/kogpt2-base-v2",
    bos_token='</s>', eos_token='</s>', unk_token='<unk>',
    pad_token='<pad>', mask_token='<mask>')

# 모델을 디바이스(GPU가 사용 가능한 경우 GPU)로 이동
model.to(device)

# output.json 파일에서 학습 데이터 로드
df = pd.read_json(DATA_SET_JSON, encoding='utf-8')
```

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

```

4칸 8칸 12칸
# 데이터를 학습 및 검증 세트로 분할
train_df, val_df = train_test_split(df, test_size=0.1, random_state=42)

def preprocess(df):
    input_ids = []
    attention_masks = []
    for index, row in df.iterrows():
        # 질문과 응답을 합쳐 하나의 텍스트로 생성
        text = row['question'] + tokenizer.eos_token + row['answer']
        # 텍스트를 토큰화하고 텐서로 변환
        tokens = tokenizer.encode_plus(
            text,
            return_tensors='pt',
            max_length=512,
            padding='max_length',
            truncation=True
        )

```

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

```

4칸   8칸   12칸
input_ids.append(tokens['input_ids'])
attention_masks.append(tokens['attention_mask'])

input_ids = torch.cat(input_ids, dim=0)
attention_masks = torch.cat(attention_masks, dim=0)

return input_ids, attention_masks

# 전처리된 데이터 생성
train_input_ids, train_attention_masks = preprocess(train_df)
val_input_ids, val_attention_masks = preprocess(val_df)

# DataLoader 생성
train_dataset = TensorDataset(train_input_ids, train_attention_masks)
train_dataloader = DataLoader(train_dataset, batch_size=8)

val_dataset = TensorDataset(val_input_ids, val_attention_masks)
val_dataloader = DataLoader(val_dataset, batch_size=8)

```

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

```

# 모델 학습
model.train()

optimizer = torch.optim.Adam(
    model.parameters(),
    lr=1e-5,           # 학습률 : 각 파라미터 업데이트의 크기
    betas=(0.9, 0.999), # 0.9 : 과거의 기울기 반영비율, 0.999 : 과거의 기울기 제곱의 평균 반영비율
    eps=1e-8,         # 0으로 나누는 것 방지 위에 분모에 추가되는 작은 상수
    weight_decay=0,    # L2 정규화를 적용하는 데 사용되는 가중치 감소계수, 0이 아니면 모델의 복잡성 감소, 과적합 방지
    amsgrad=False      # Adam 알고리즘의 문제점을 개선한 amsgrad 모델 반영여부
)

for epoch in range(EPOCH):
    for i, batch in enumerate(train_dataloader):
        batch_input_ids, batch_attention_masks = batch

        # 텐서를 디바이스(GPU가 사용 가능한 경우 GPU)로 이동
        batch_input_ids = batch_input_ids.to(device)
        batch_attention_masks = batch_attention_masks.to(device)

```

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

4칸 8칸 12칸

```
# 경사 초기화 및 순방향 전파
optimizer.zero_grad()
outputs = model(batch_input_ids, attention_mask=batch_attention_masks, labels=batch_input_ids)

# 손실 계산 및 역방향 전파
loss = outputs.loss
loss.backward()

# 가중치 업데이트
optimizer.step()

# 손실 출력
if i % 100 == 0:
    print(f'Epoch {epoch + 1}, Batch {i + 1}/{len(train_dataloader)}, Loss: {loss.item():.4f}')
```

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

```

4칸 8칸 12칸
# 모델 평가
model.eval()
correct = 0
total = 0

with torch.no_grad():
    for i, batch in enumerate(val_dataloader):
        batch_input_ids, batch_attention_masks = batch
        batch_input_ids = batch_input_ids.to(device)
        batch_attention_masks = batch_attention_masks.to(device)

        # 모델을 사용하여 예측값 생성
        outputs = model(batch_input_ids, attention_mask=batch_attention_masks)

        # 예측값 중 가장 높은 확률의 토큰 인덱스를 선택
        _, predicted = torch.max(outputs.logits, 2)
    
```

2. AI 비서 학습

⌚ 학습 프로그램

■ 학습용 프로그램

_seok_colab\2.learning.py 파일 저장

```

4칸      8칸      12칸
# 전체 예측 개수 및 정확한 예측 개수를 계산
total += batch_input_ids.size(0)
correct += (predicted == batch_input_ids).sum().item()

# 정확도 계산
accuracy = correct / total

print(f'Accuracy: {accuracy * 100:.2f}%',)

# 학습된 모델 저장
model.save_pretrained(SAVED_MODEL_NAME)

```


2. AI 비서 학습

⌚ 학습 프로그램

■ 학습 실행

```

명령 프롬프트
Microsoft Windows [Version 10.0.19045.3570]
(c) Microsoft Corporation. All rights reserved.

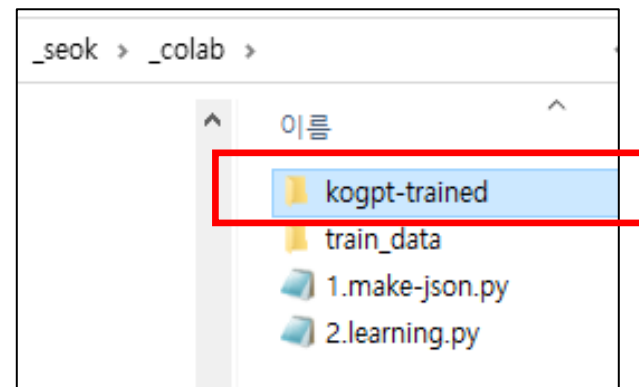
C:\Users\dossa>cd \_seok\_colab

C:\_seok\_colab>python 2.learning.py
device = cuda
The tokenizer class you load from this checkpoint is not the same type
result in unexpected tokenization.
The tokenizer class you load from this checkpoint is 'GPT2Tokenizer'.
The class this function is called from is 'PreTrainedTokenizerFast'.
Epoch 1, Batch 1/13, Loss: 15.0676
Accuracy: 46033.33%

C:\_seok\_colab>_
    
```

cd _seok_colab

python 2.learning.py

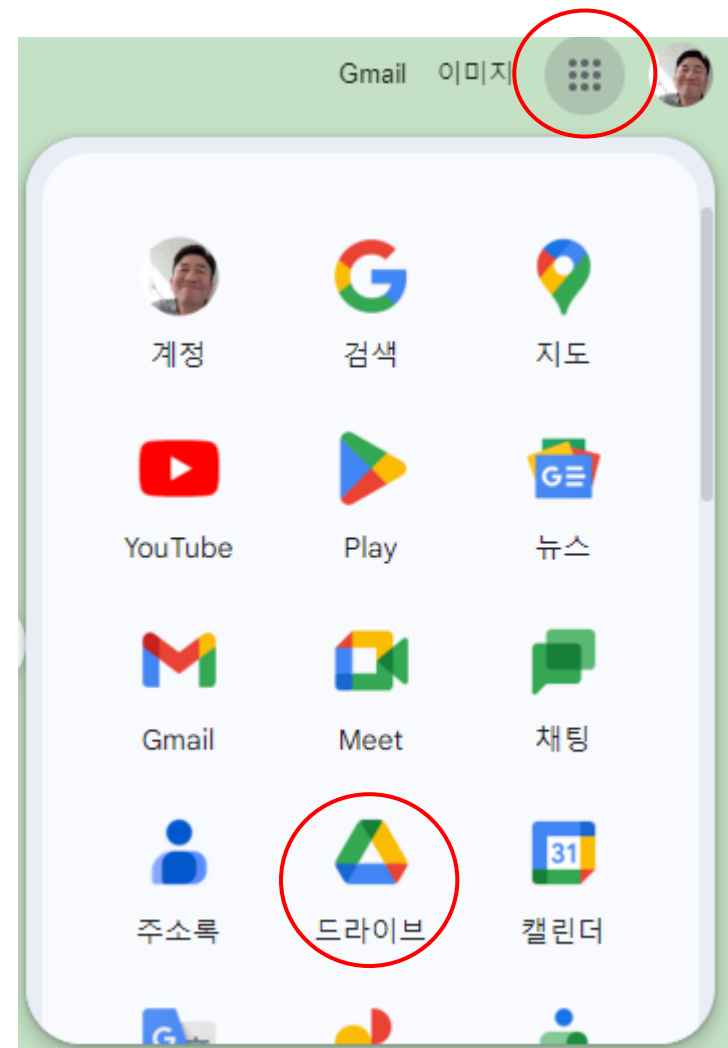
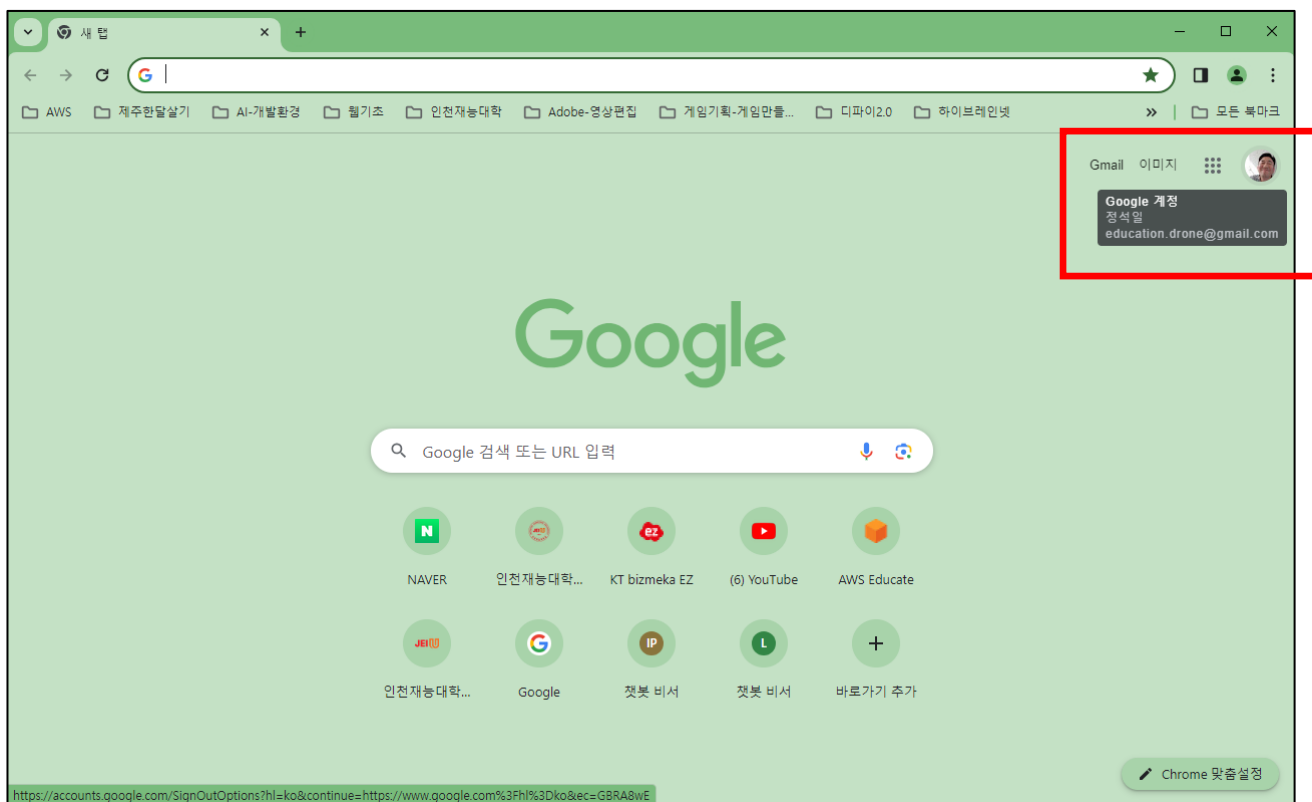


kogpt-trained 폴더 밑에 AI 가 생성됨

2. AI 비서 학습

⌚ colab 에서 학습

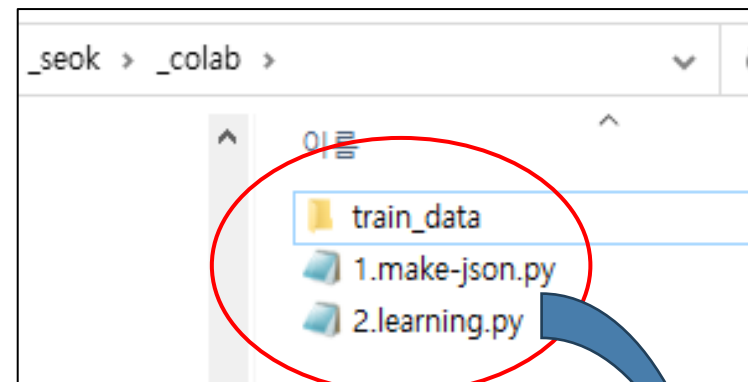
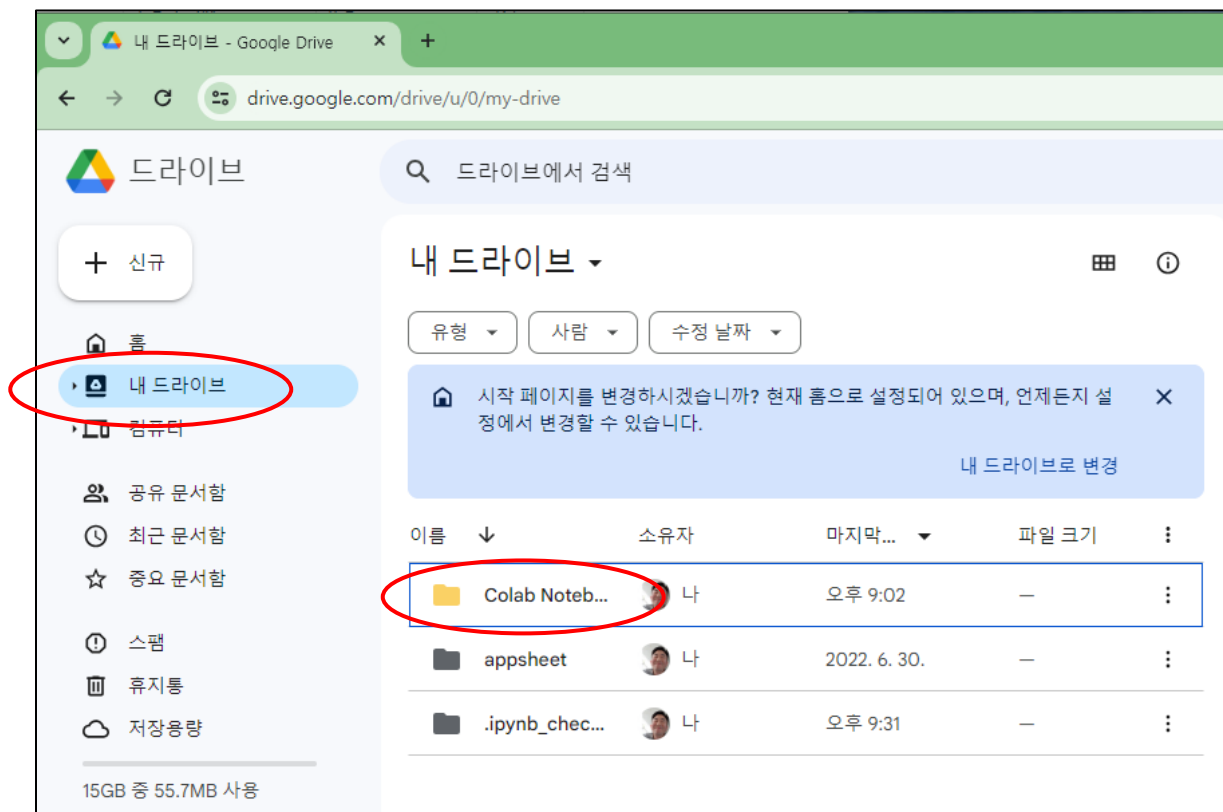
- 크롬에서 Google 로그인 → 구글드라이브



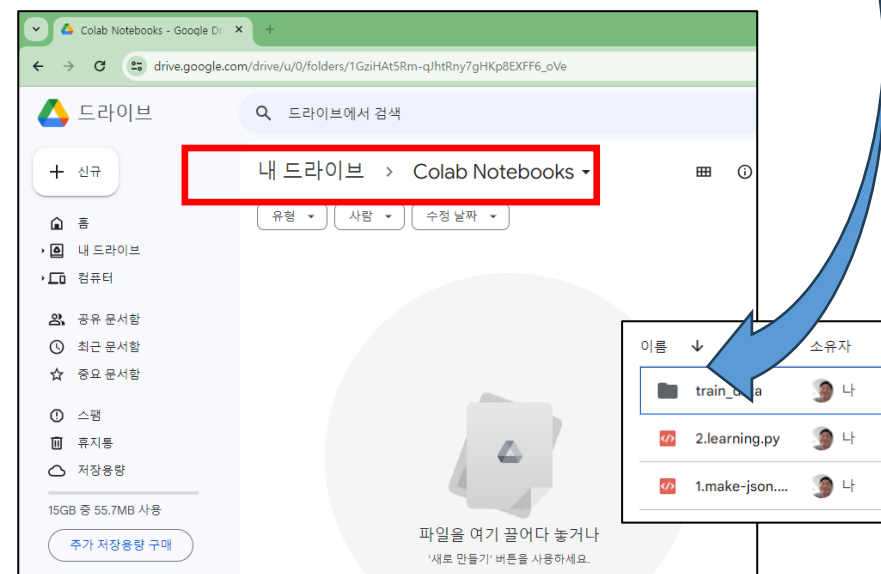
2. AI 비서 학습

⌚ colab 에서 학습

- 내 드라이브에 파일 업로드



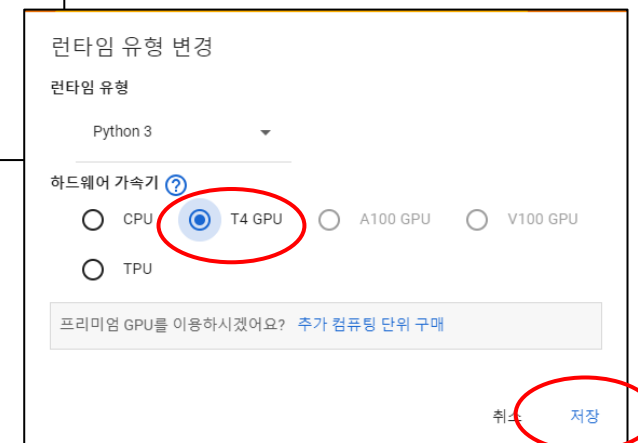
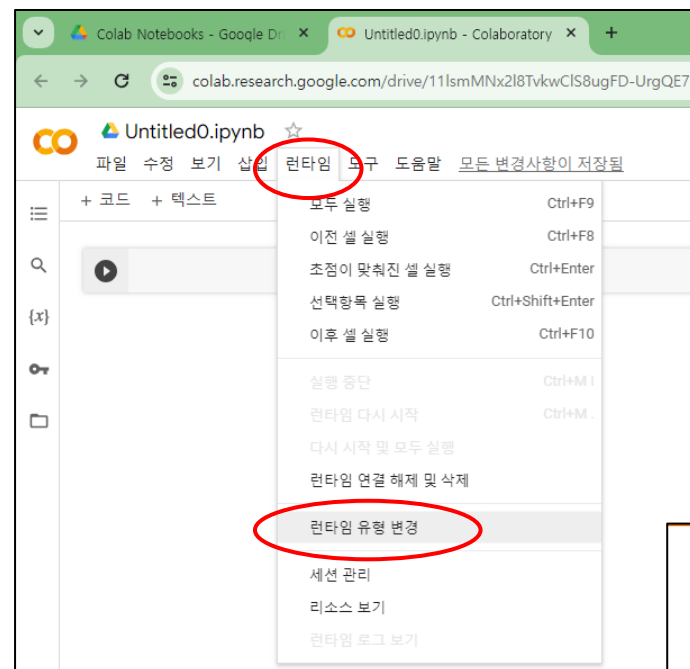
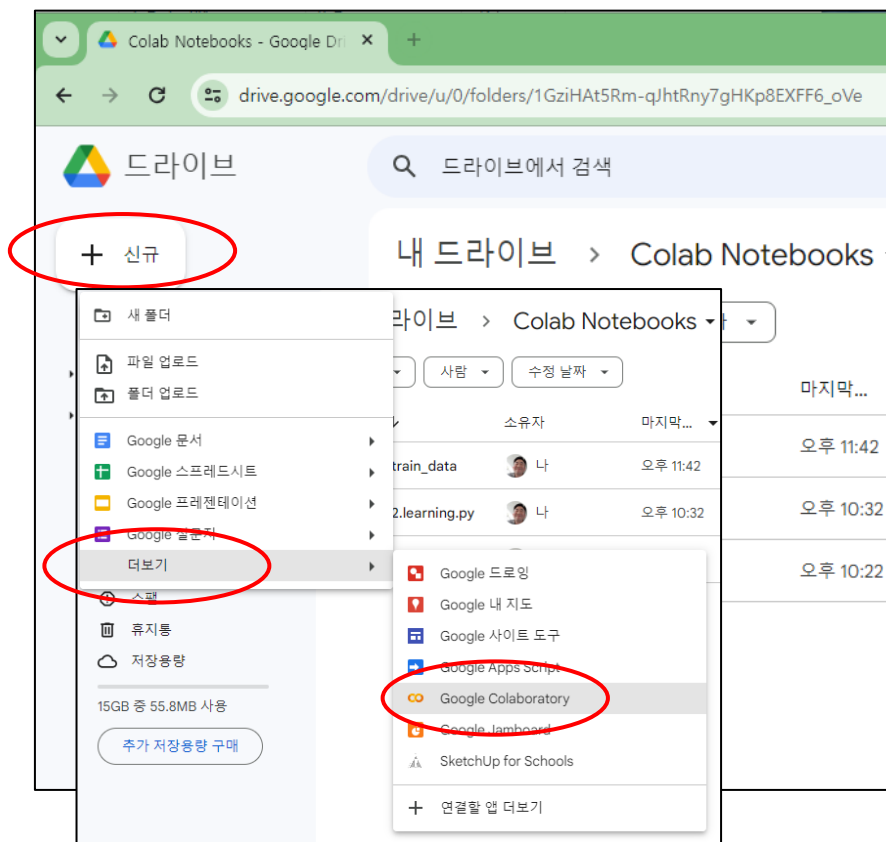
드래그인
드랍



2. AI 비서 학습

⌚ colab 에서 학습

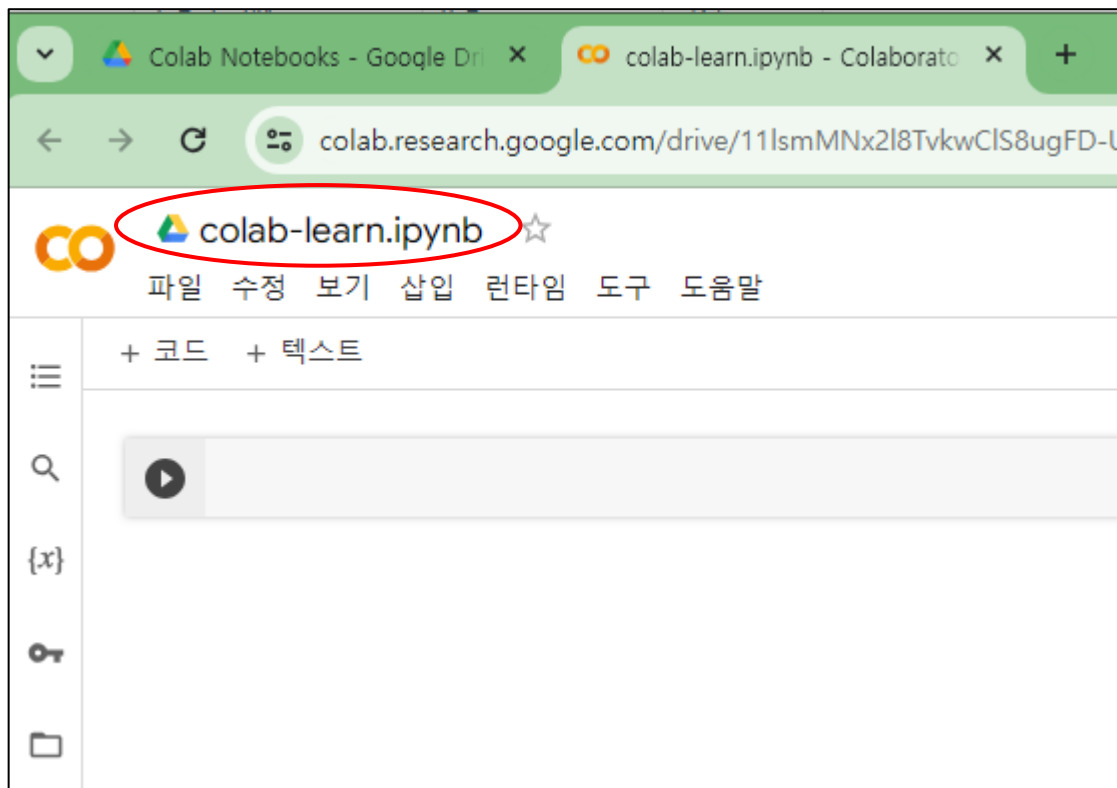
■ 시스템 런타임 설정 - GPU



2. AI 비서 학습

⌚ colab 에서 학습

- 학습 파일 생성



파일 이름 변경

colab-learn.ipynb

2. AI 비서 학습

⌚ colab 에서 학습

■ 학습 파일 생성

```
2.learning.py - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
#=====
# kogpt 학습
#=====
import os
import json
import pandas as pd
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
from torch.utils.data import DataLoader, TensorDataset
from sklearn.model_selection import train_test_split
from transformers import PreTrainedTokenizerFast

# GPU가 사용 가능한 경우 사용하고, 그렇지 않으면 CPU
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print('device = ', device)

# epoch 크기 조정 : 학습량
EPOCH = 1 # 1 ~ 1000 값

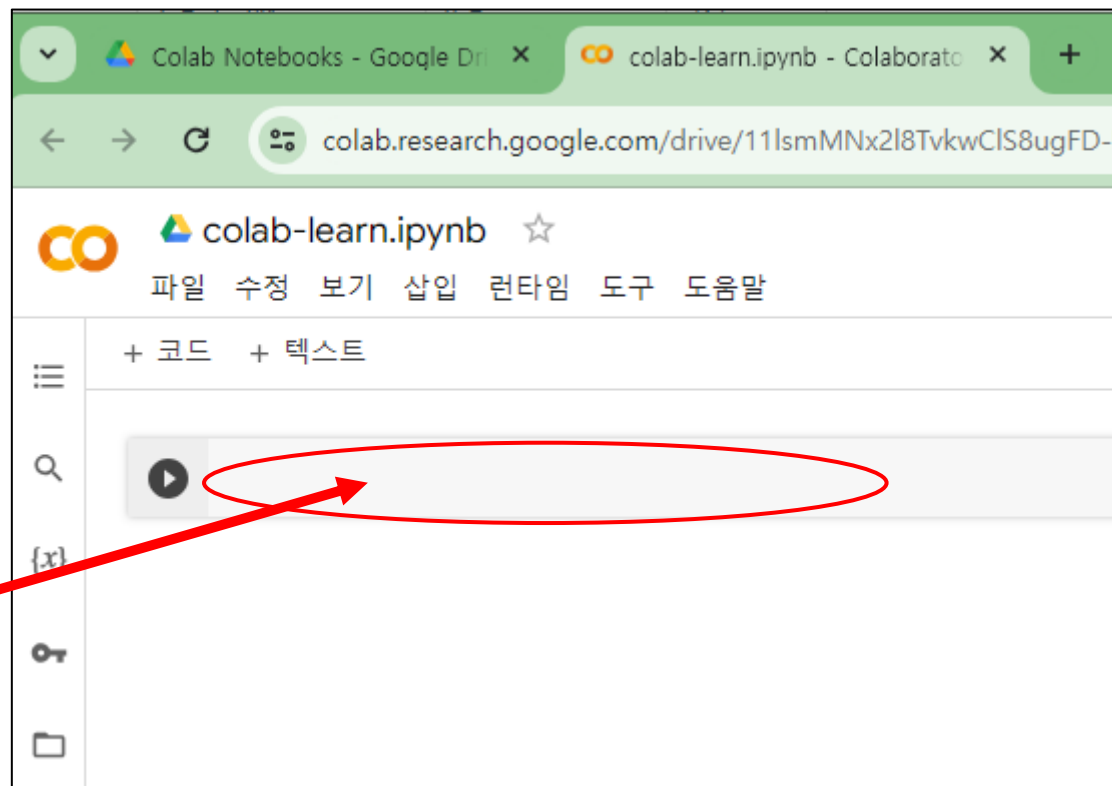
# 읽어 올 데이터셋
DATA_SET_JSON = './train_data/output.json'
# 학습한 모델 저장하는 폴더명
```

메모장을 열고,

_seok_colab\
2.learning.py

의 소스를 복사해서

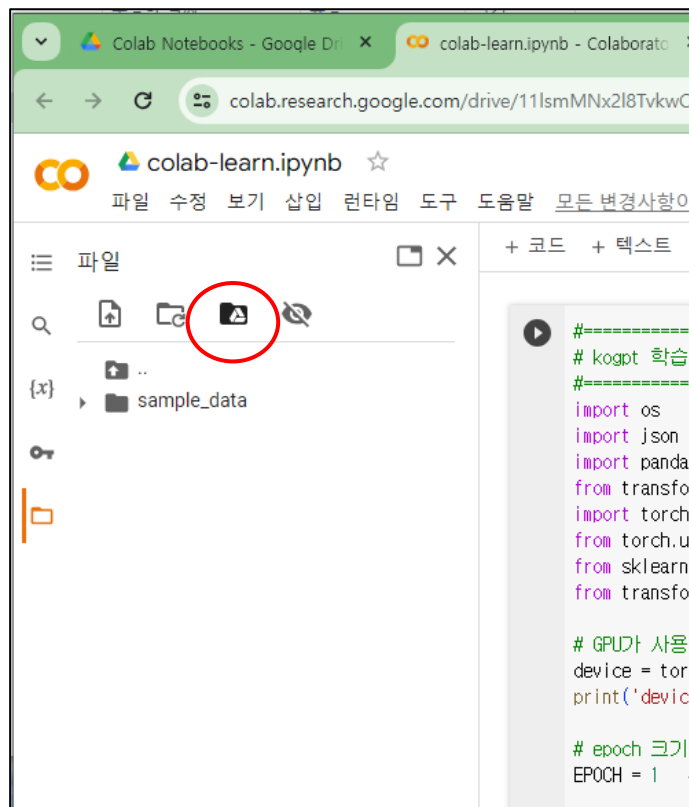
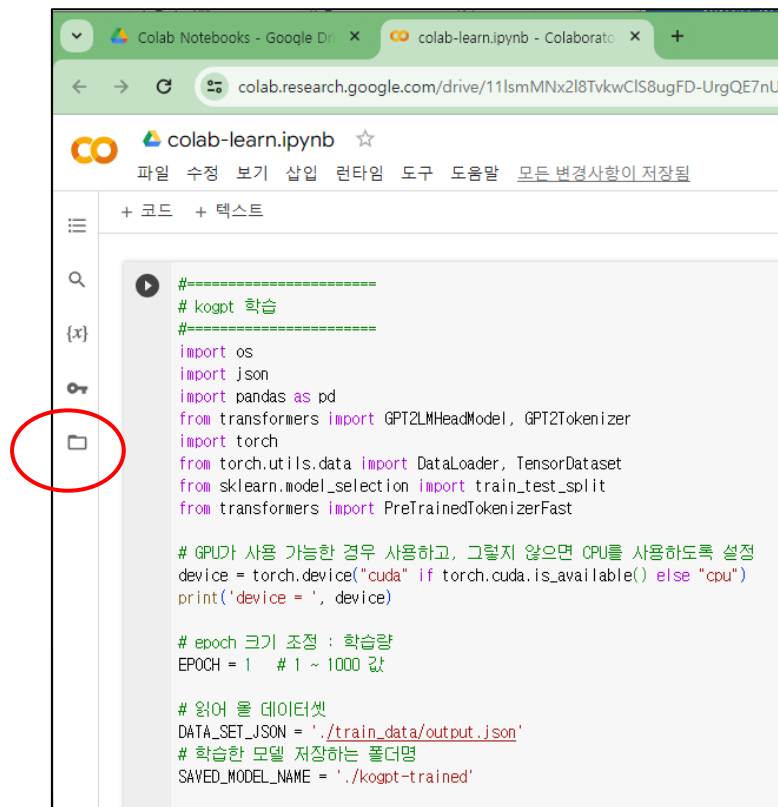
colab 에 붙여 넣기



2. AI 비서 학습

⌚ colab 에서 학습

■ 드라이브 마운트



노트북이 **Google Drive** 파일에 액세스하도록 허용하시겠습니까?

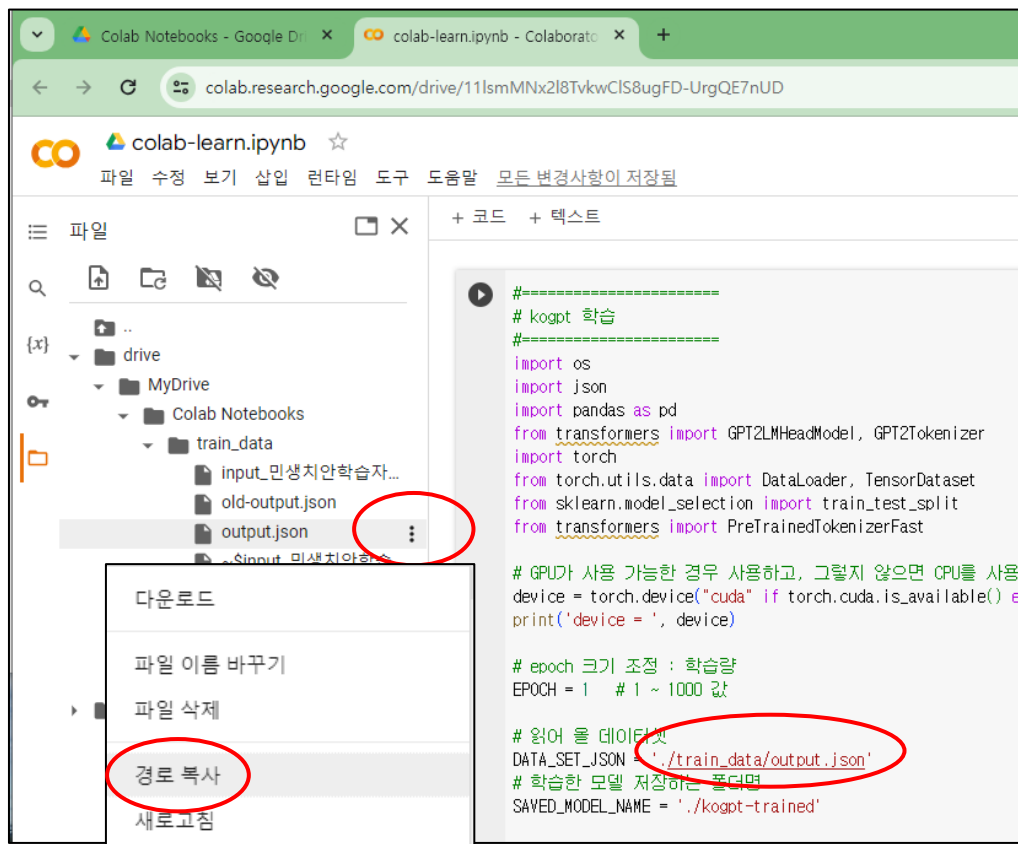
Google Drive에 연결하면 액세스 권한이 취소될 때까지 이 노트북에서 실행된 코드가 Google Drive의 파일을 수정할 수 있습니다.

아니요 [Google Drive에 연결](#)

2. AI 비서 학습

⌚ colab 에서 학습

- 드라이브 마운트 경로를 프로그램에 반영



drive → MyDrive → Colab Notebooks → train_data

에 있는 output.json 옆에 있는 메뉴 클릭

<경로 복사> 한 것을

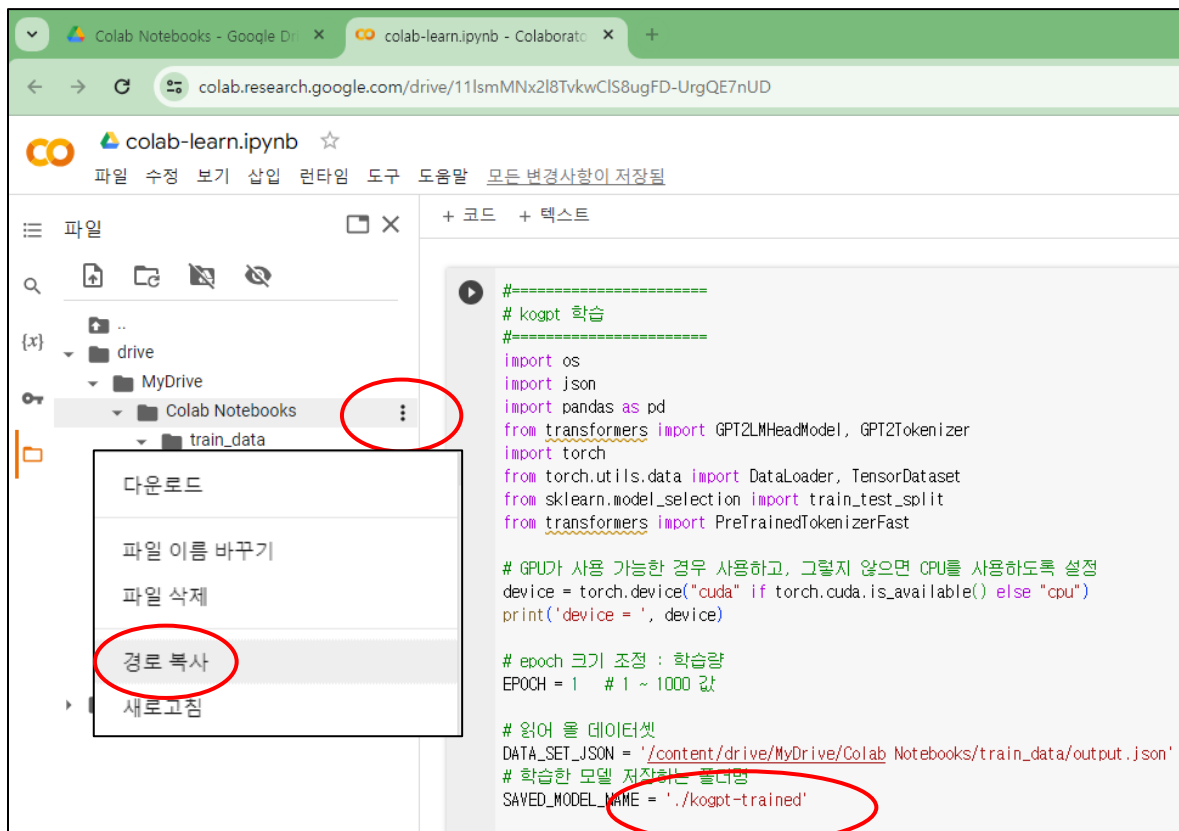
프로그램 소스의 DATA_SET_JSON 에 붙여 넣기 (Ctrl-V)

`/content/drive/MyDrive/Colab Notebooks/train_data/output.json`

2. AI 비서 학습

⌚ colab 에서 학습

- 드라이브 마운트 경로를 프로그램에 반영



drive → MyDrive → Colab Notebooks

옆에 있는 메뉴 클릭

<경로 복사> 한 것을

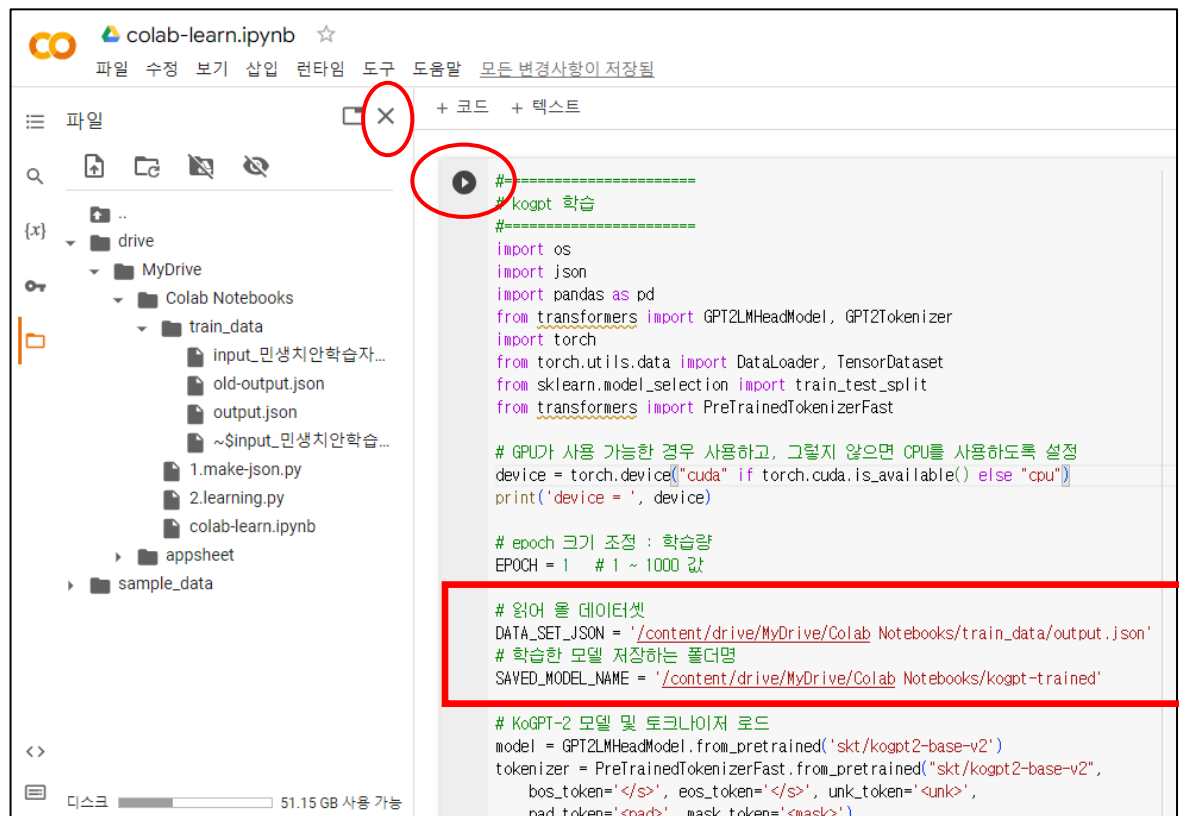
프로그램 소스의 SAVED_MODEL_NAME 에 붙여 넣기
(Ctrl +V)

/content/drive/MyDrive/Colab Notebooks/kogpt-trained

2. AI 비서 학습

⌚ colab 에서 학습

■ 학습 시작



```

colab-learn.ipynb
파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

{X}
+ 코드 + 텍스트

# =====
# kogpt 학습
# =====

import os
import json
import pandas as pd
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
from torch.utils.data import DataLoader, TensorDataset
from sklearn.model_selection import train_test_split
from transformers import PreTrainedTokenizerFast

# GPU가 사용 가능한 경우 사용하고, 그렇지 않으면 CPU를 사용하도록 설정
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print('device = ', device)

# epoch 크기 조정 : 학습량
EPOCH = 1 # 1 ~ 1000 값

# 읽어 올 데이터셋
DATA_SET_JSON = '/content/drive/MyDrive/Colab Notebooks/train_data/output.json'
# 학습한 모델 저장하는 폴더명
SAVED_MODEL_NAME = '/content/drive/MyDrive/Colab Notebooks/kogpt-trained'

# KoGPT-2 모델 및 토큰라이저 로드
model = GPT2LMHeadModel.from_pretrained('skt/kogpt2-base-v2')
tokenizer = PreTrainedTokenizerFast.from_pretrained('skt/kogpt2-base-v2',
    bos_token='</s>', eos_token='</s>', unk_token='<unk>',
    pad_token='<pad>', mask_token='<mask>')
  
```

[X] 눌러서 왼쪽 메뉴는 닫고,

[셀 실행] 클릭

경로 반드시 확인

2. AI 비서 학습

⌚ colab 에서 학습

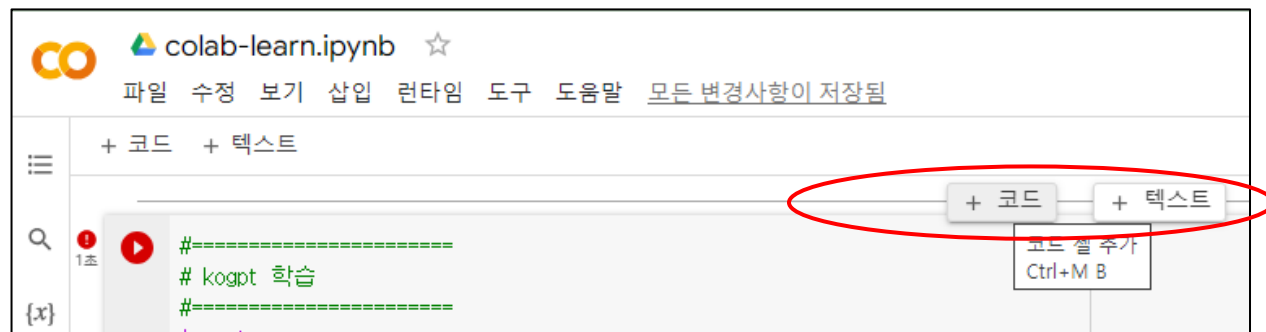
■ 학습 시작

```

#=====
# kogpt 학습
#=====
import os
import json
import pandas as pd
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
    
```

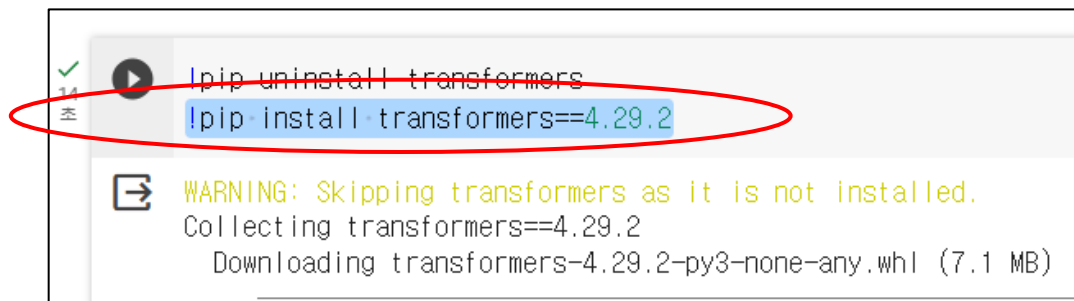
실행 시, 오류가 나면, 버튼이 빨간색으로 변하고,

패키지에 붉은 줄이 생김



마우스를 경계선에 가져다 대면, +코드, +텍스트 메뉴가 나타남

[+코드] 클릭

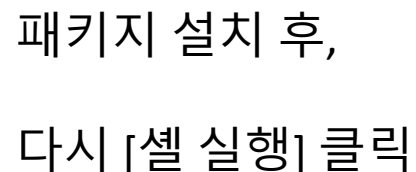


추가된 셀에서 패키지를 설치 함
(최신버전으로 하지 마시고)

`!pip install transformers==4.29.2`

⌚ colab 에서 학습

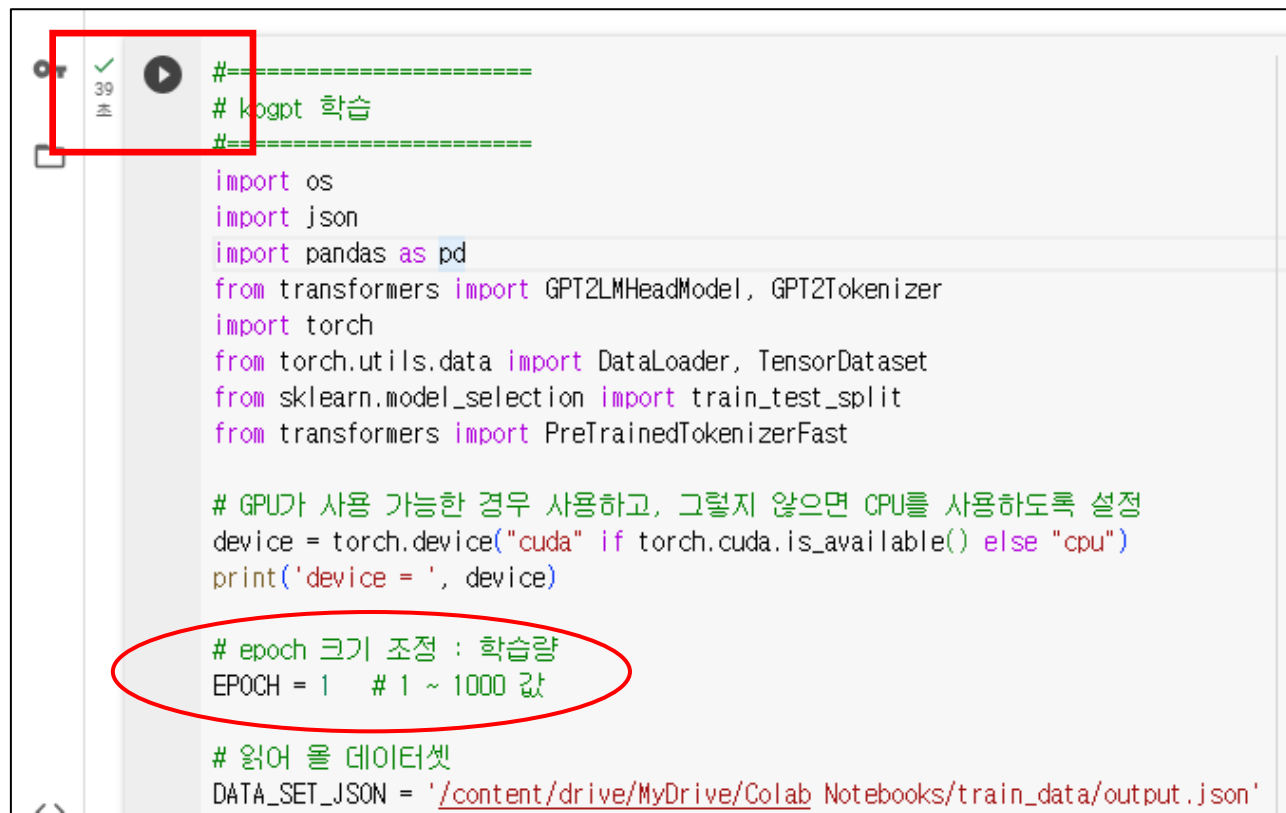
■ 학습 시작



2. AI 비서 학습

⌚ colab 에서 학습

- 학습 종료 후, 학습횟수 수정한 다음 재학습



```

#-----
# kogpt 학습
#-----

import os
import json
import pandas as pd
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
from torch.utils.data import DataLoader, TensorDataset
from sklearn.model_selection import train_test_split
from transformers import PreTrainedTokenizerFast

# GPU가 사용 가능한 경우 사용하고, 그렇지 않으면 CPU를 사용하도록 설정
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print('device = ', device)

# epoch 크기 조정 : 학습량
EPOCH = 1 # 1 ~ 1000 값

# 읽어 올 데이터셋
DATA_SET_JSON = '/content/drive/MyDrive/Colab Notebooks/train_data/output.json'
    
```

학습이 완료되면, 수행된 시간이 나타남

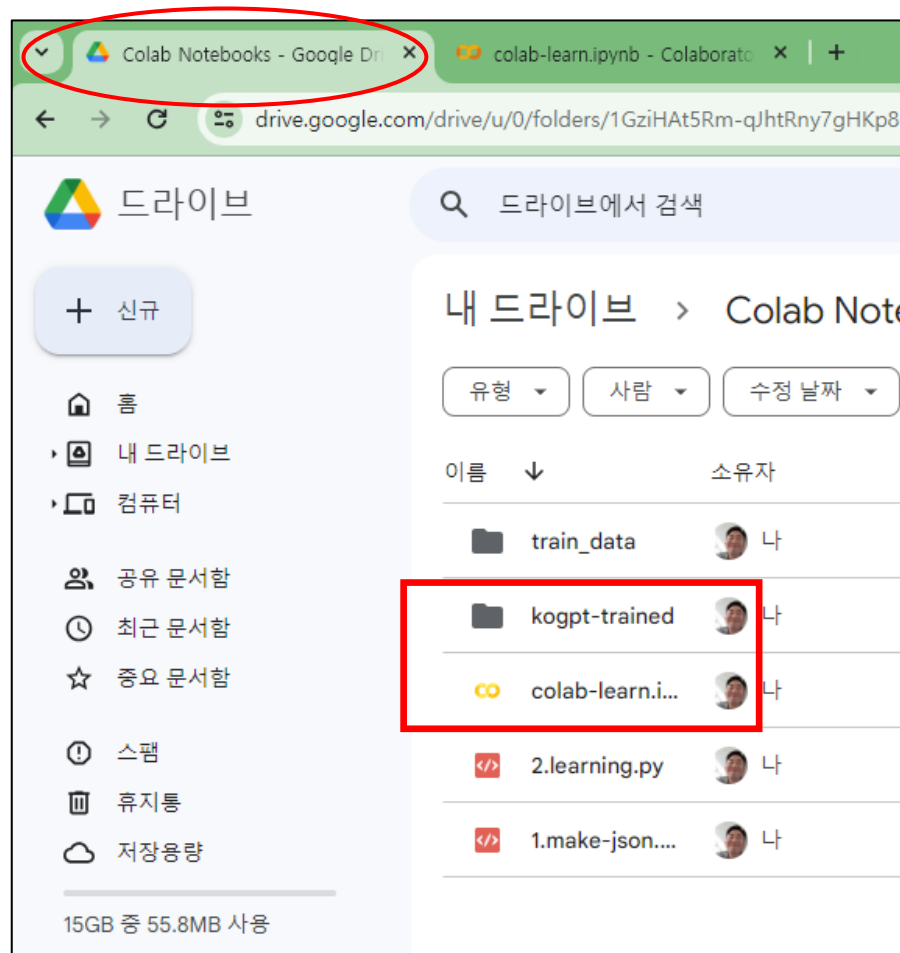
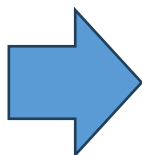
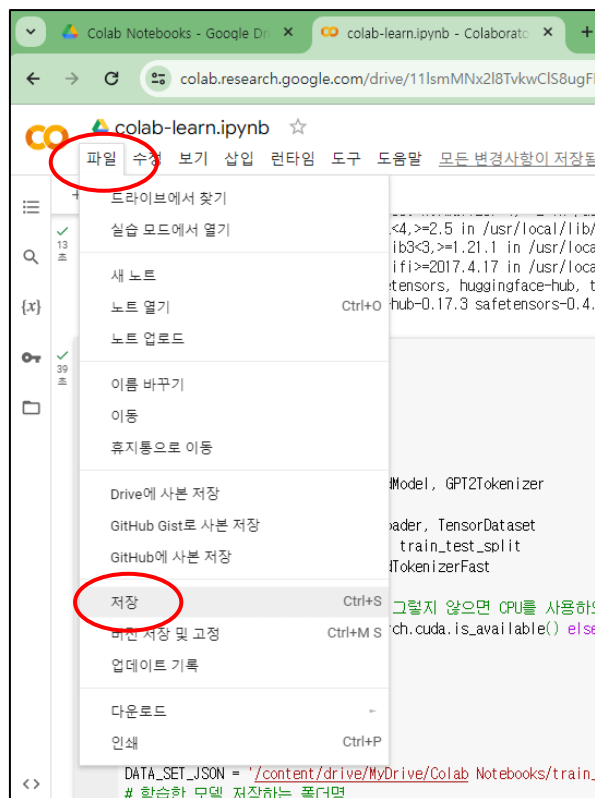
1번 학습하는 데, 약 39초 소요
100번 학습하면, 약 3,900초 소요

학습 횟수를 100으로 변경하여,
재 학습 수행

2. AI 비서 학습

⌚ colab 에서 학습

■ 프로그램 저장

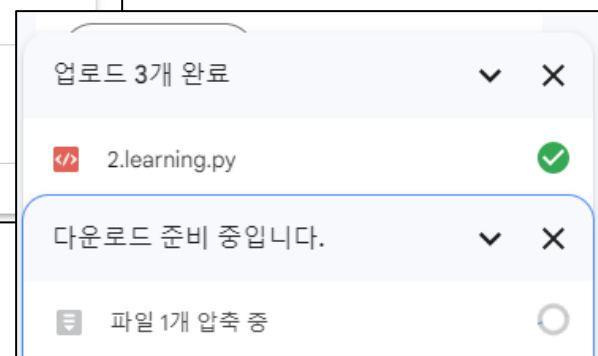
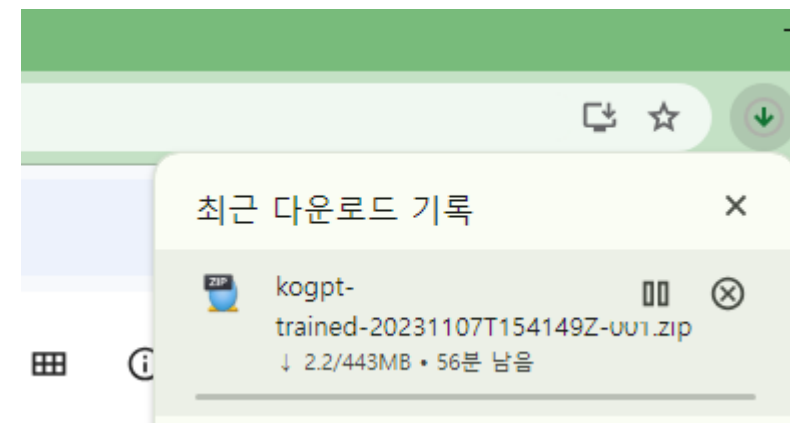
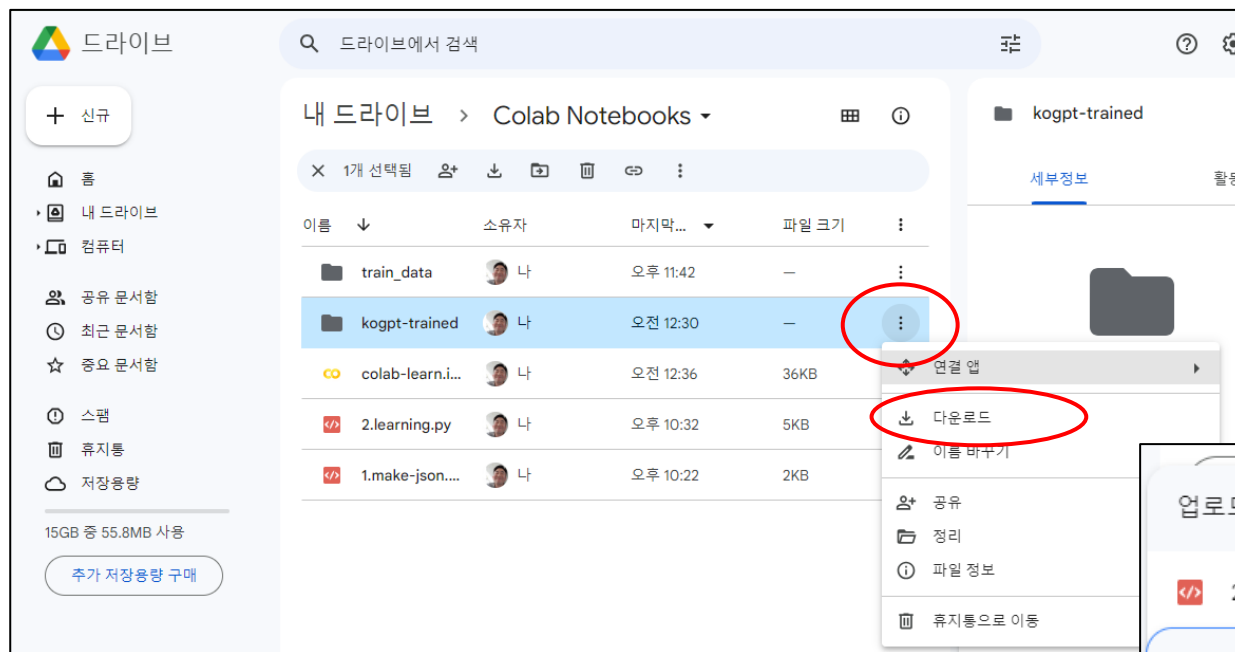


학습된 AI 폴더와
저장된 프로그램 소스

2. AI 비서 학습

⌚ colab 에서 학습

■ 학습된 AI 다운로드

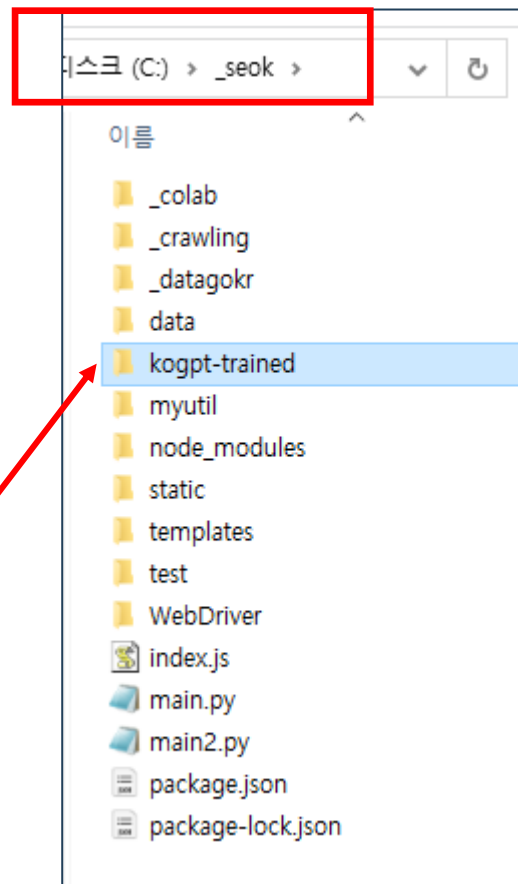
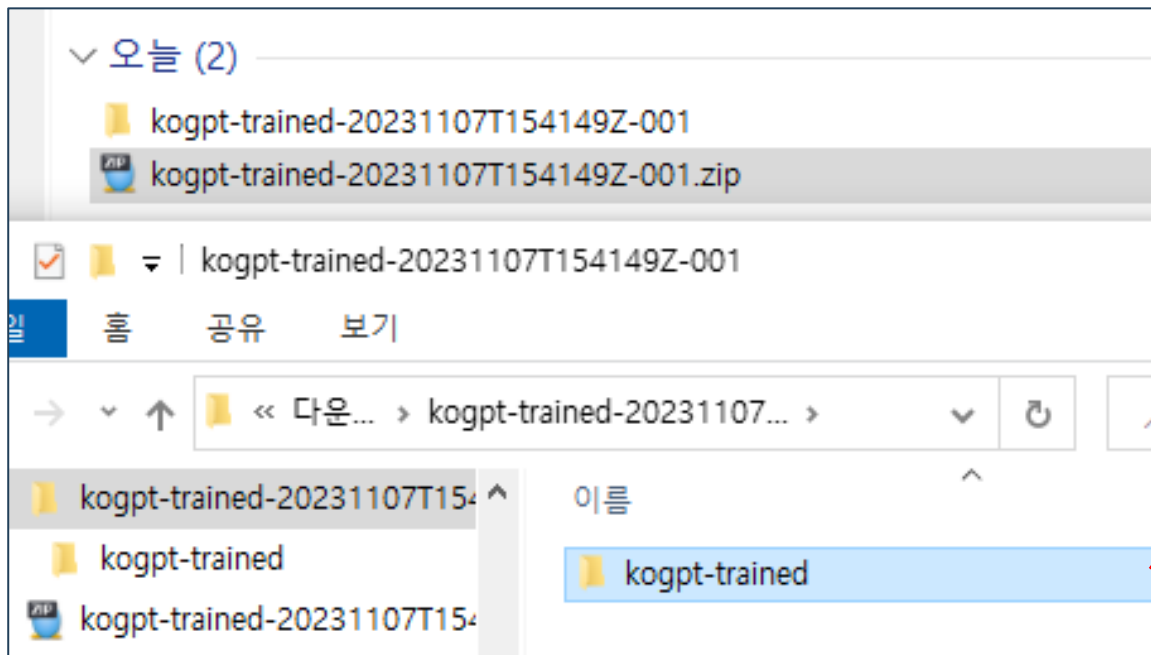


압축되어 다운로드 됨

2. AI 비서 학습

⌚ colab 에서 학습

- 압축 풀고, 본인의 디렉토리로 이동



참고 자료

- 자바와 파이썬으로 만드는 빅데이터시스템(제이펍, 황세규)
- 위키독스(<https://wikidocs.net/22654>)
- 네이버블로그(<https://blog.naver.com/classmethodkr/222822485338>)
- 데이터분석과 인공지능 활용 (NOSVOS, 데이터분석과인공지능활용편찬위원회 편)

참고 사이트

유튜버 : 빅공잼 : <https://www.youtube.com/watch?v=bnYxO2XRCQ0>

네이버 블로그 : 빅공잼

<https://biggongjam.notion.site/3-Hadoop-cd6944182da74edf8d2339b654e0bfb9>

<https://biggongjam.notion.site/4-Spark-2c341ddc8715411484cb2f0254b60126>

Q n A