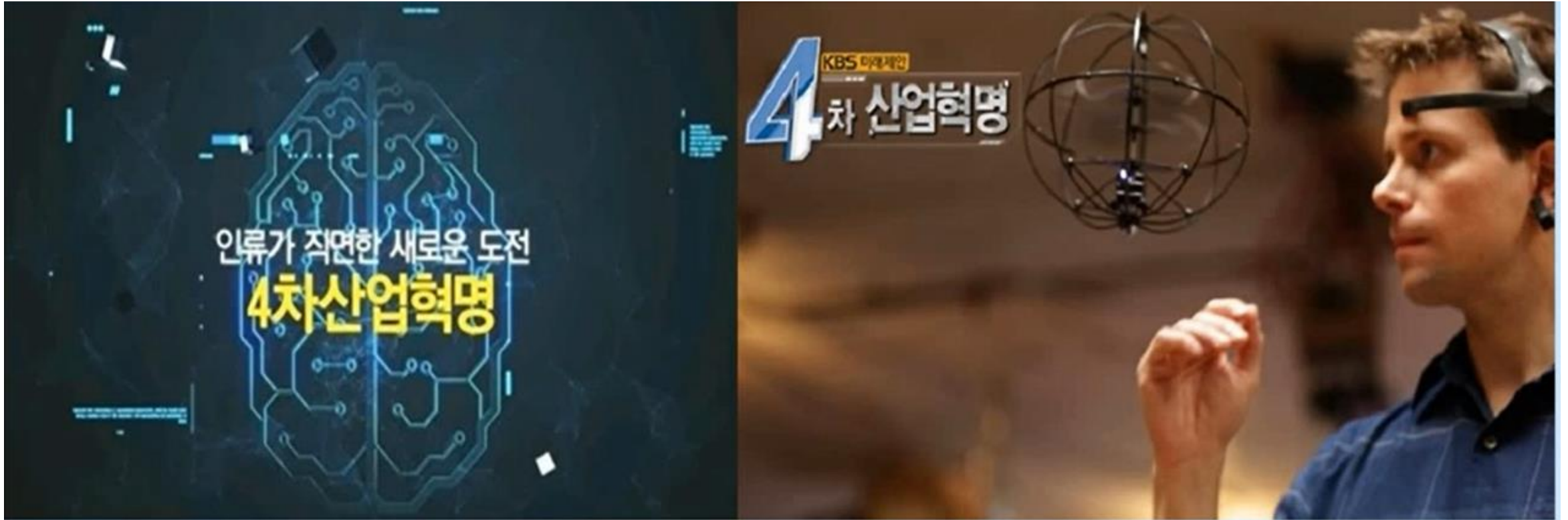


AI 구현을 위한 파이썬 기본 문법과 기초 프로그래밍

인공지능과 파이썬

■ 4차산업혁명과 인공지능



- 2016년 스위스 다보스에서 개최된 세계 경제포럼에서 처음 언급
- 학자에 따라 정의는 조금씩 다르나, 대체로 4차 산업혁명은 모든 것이 연결(Connectivity)되어 있는 환경에서 인공지능(Artificial Intelligence)에 의해 더욱 편리하고 지능적인 사회로의 혁신적 변화를 지칭

□ 인공지능/머신러닝/딥러닝

1 머신러닝 기존의 데이터 마이닝과 다른점임

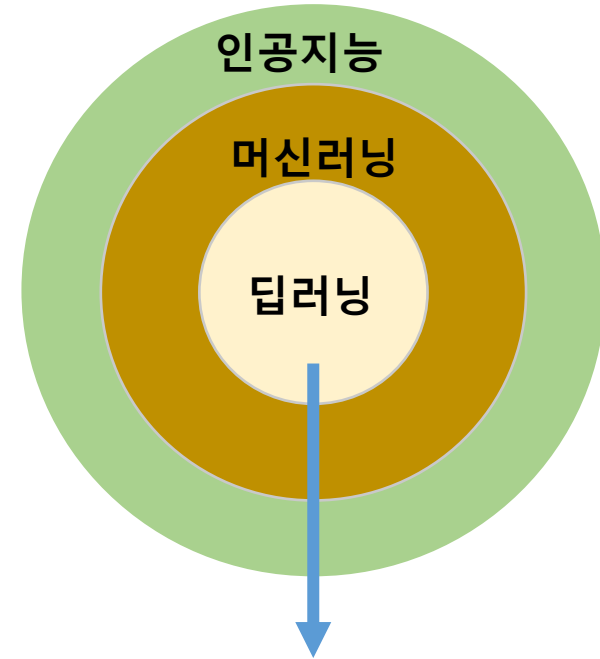
데이터를 분석, 스스로 학습하여 입력하지 않은 정보도 판단, 결정함

2 딥러닝

컴퓨터가 사람처럼 생각하고 배울 수 있도록 스스로 학습함

3 인공지능

인간의 학습능력을 프로그래밍하여 컴퓨터가 인간의 지능적 행동을 모방하게 함



심층신경망

■ AI 개발에 가장 적합한 프로그래밍 언어

머신러닝 / 딥러닝 구현 - python, not framework



머신러닝 프레임워크(Framework) 사용 단점

- 머신러닝의 알고리즘을 API로 추상화 함으로서 개발을 쉽고 빠르게 할 수 있지만, 동작원리와 내부 구현을 자세히 알 수 없는 **블랙박스(Black Box)** 로서 동작함



파이썬(Python) 직접 구현 시 장점

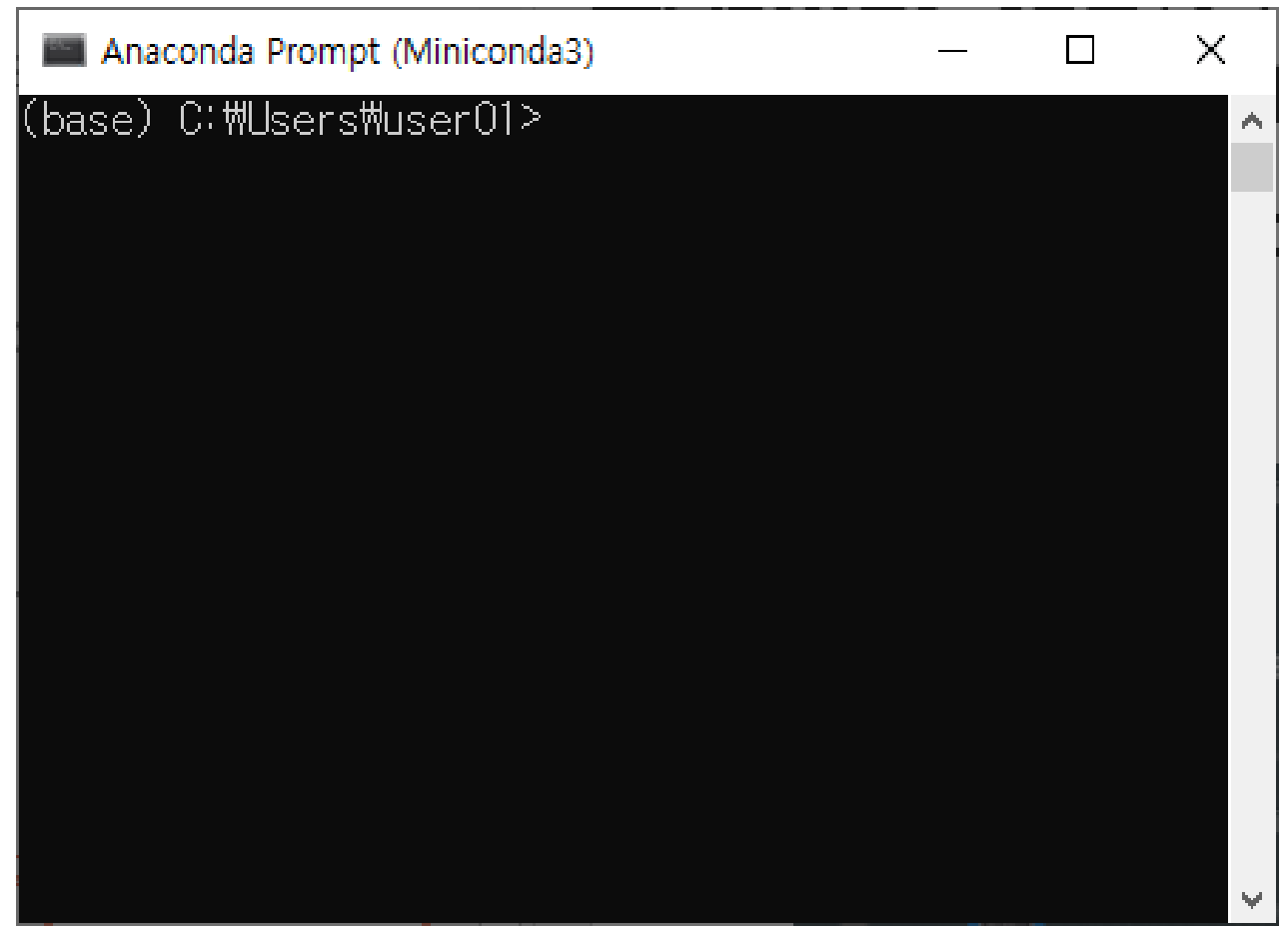
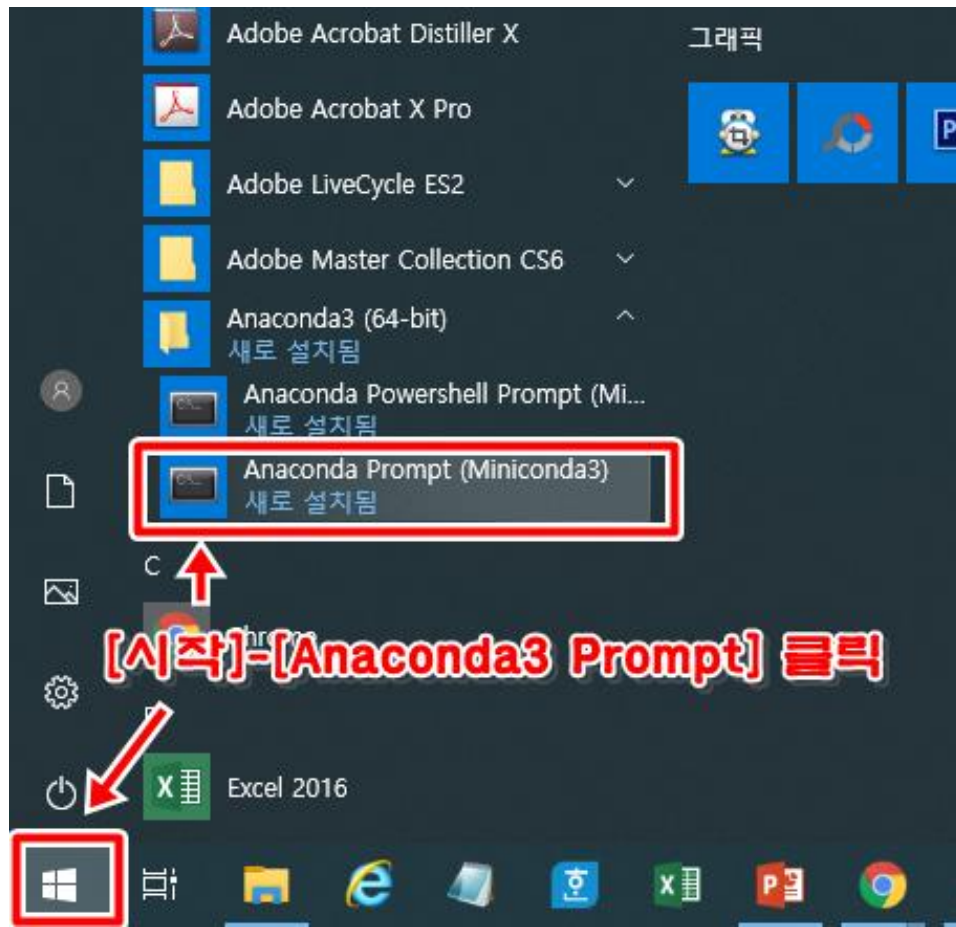
- 머신러닝 동작원리를 자세히 알 수 있어 재미있으며 동시에 알고리즘에 대한 깊은 이해 가능
- 새로운 머신러닝 알고리즘이 나왔을 때 빠르게 코드를 이해할 수 있는 **insight 획득**

이미지출처: <https://www.add-for.com/2016/11/02/blog-post-deep-learning-frameworks/>

파이썬 기본 환경 설정

■ 아나콘다 환경 설정

- [시작] -> [anaconda prompt] 실행



▣ 아나콘다 환경 설정

> > conda update conda

> > Proceed ([y]/n)? y

> > conda create -n nhi python=3.7

(가상환경 생성: conda create -n "가상환경 이름" python=3.7)

> > conda info -envs

(설치된 가상환경 확인)

> > activate nhi

(nhi 가상환경으로 이동)

▣ 아나콘다 환경 설정 및 실행

> > conda install jupyter

(주피터 노트북 설치)

> > conda install spyder

(스파이더 설치)

> > conda deactivate

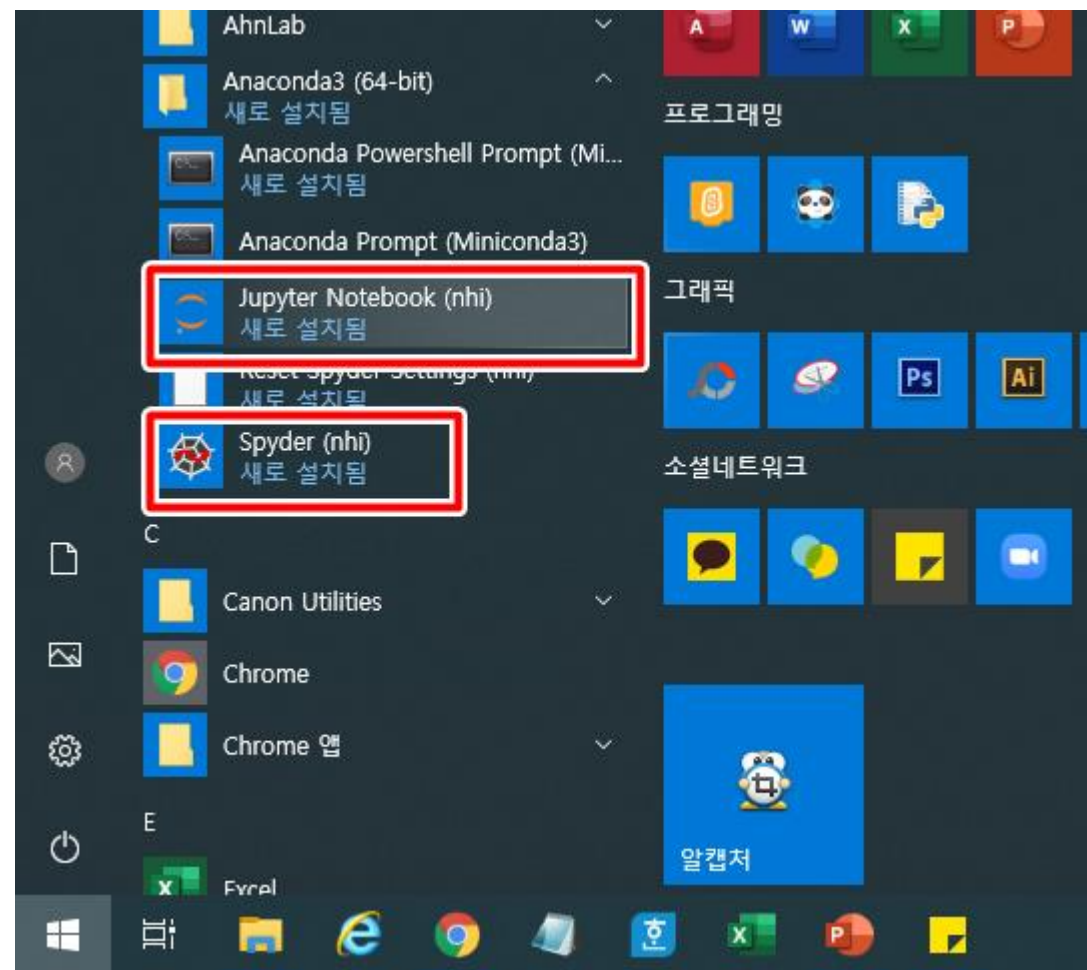
(가상환경 종료)

> > conda remove -name nhi --all

(가상환경 삭제: conda remove --name nhi --all)

■ 설치 확인

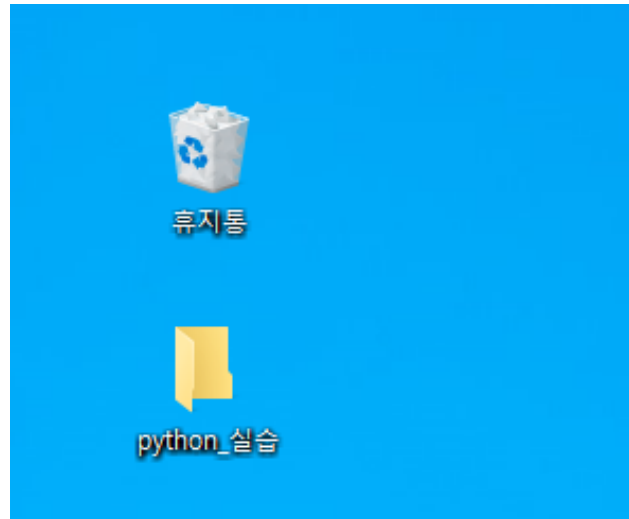
- Jupyter Notebook(nhi) 생성 확인
- Spyder(nhi) 생성 확인
- Jupyter Notebook(nhi) 실행
(잘 실행되는지 확인)



■ 작업폴더 생성

- 코드를 저장 폴더 생성

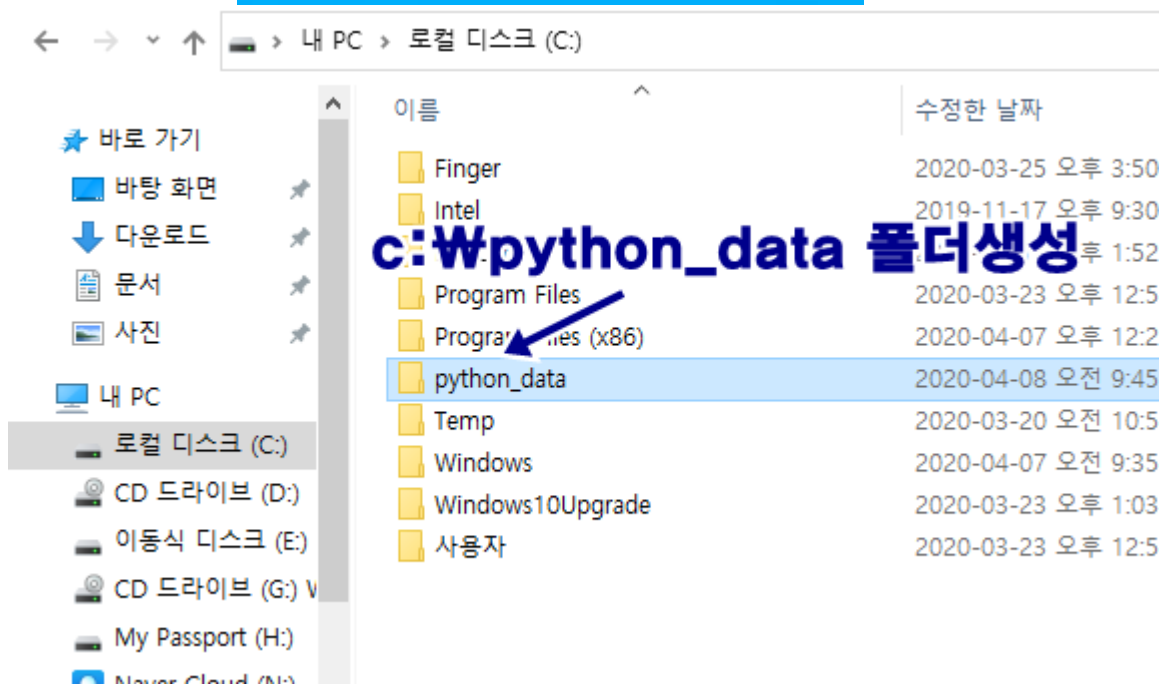
바탕화면에 'python_실습' 폴더 생성



- 데이터 저장 폴더 생성

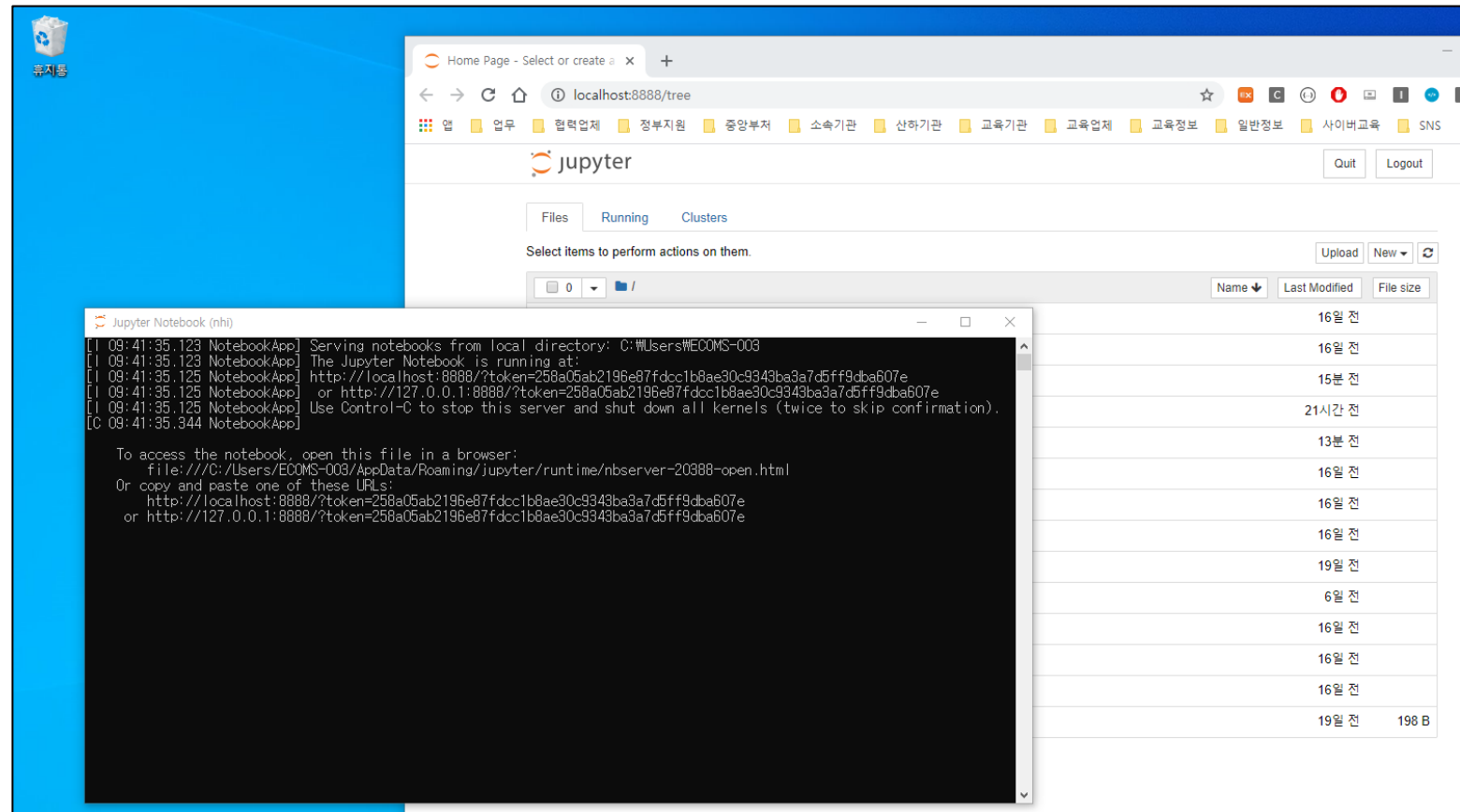
C드라이브를 선택하고

'python_data' 폴더 생성



■ Jupyter Notebook(nhi) 실행

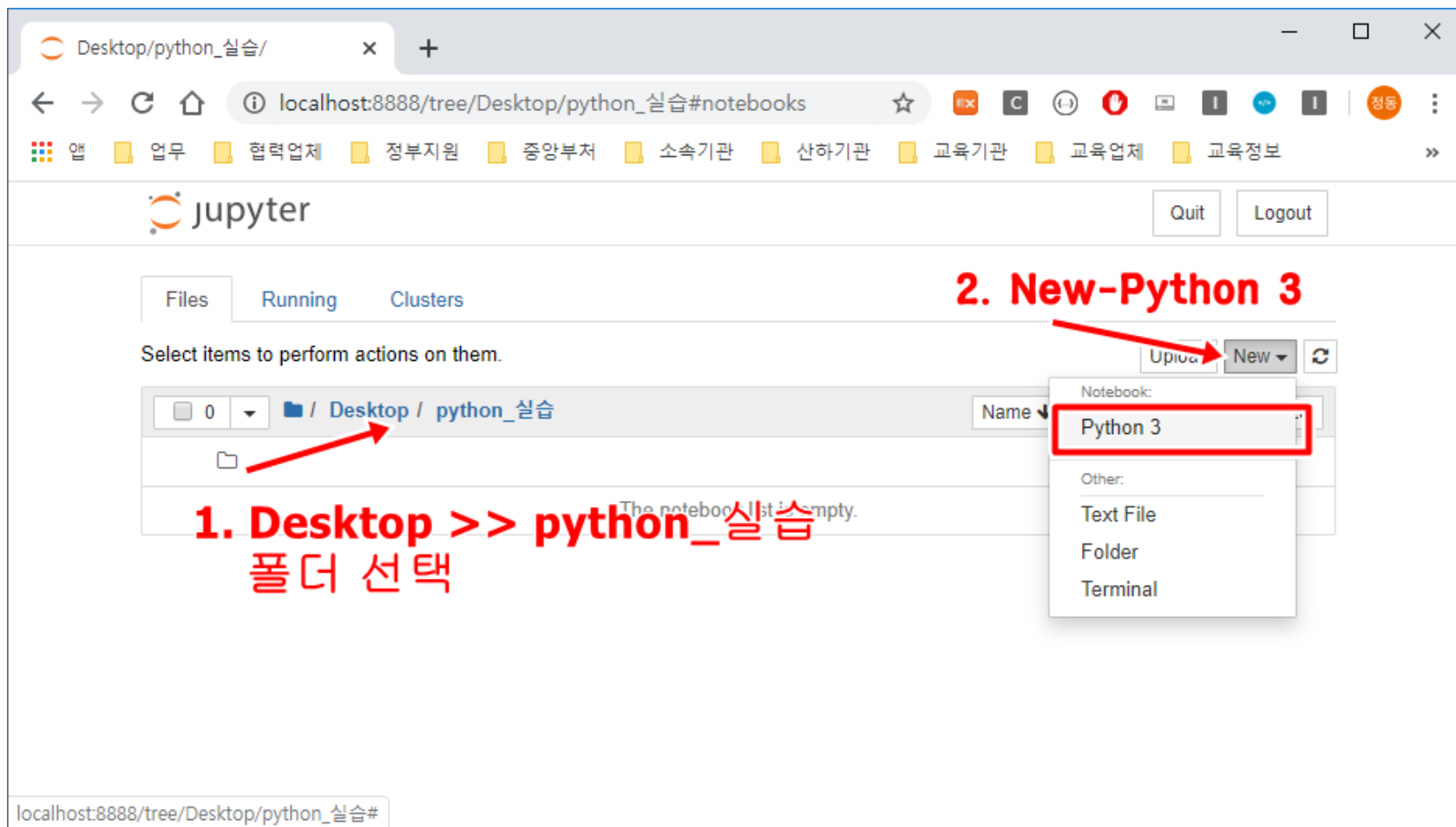
- Jupyter Notebook(nhi) 실행되면 프롬프트가 생성되고 크롬에서 Jupyter notebook가 실행됩니다.



Pandas를 이용한 데이터 관리

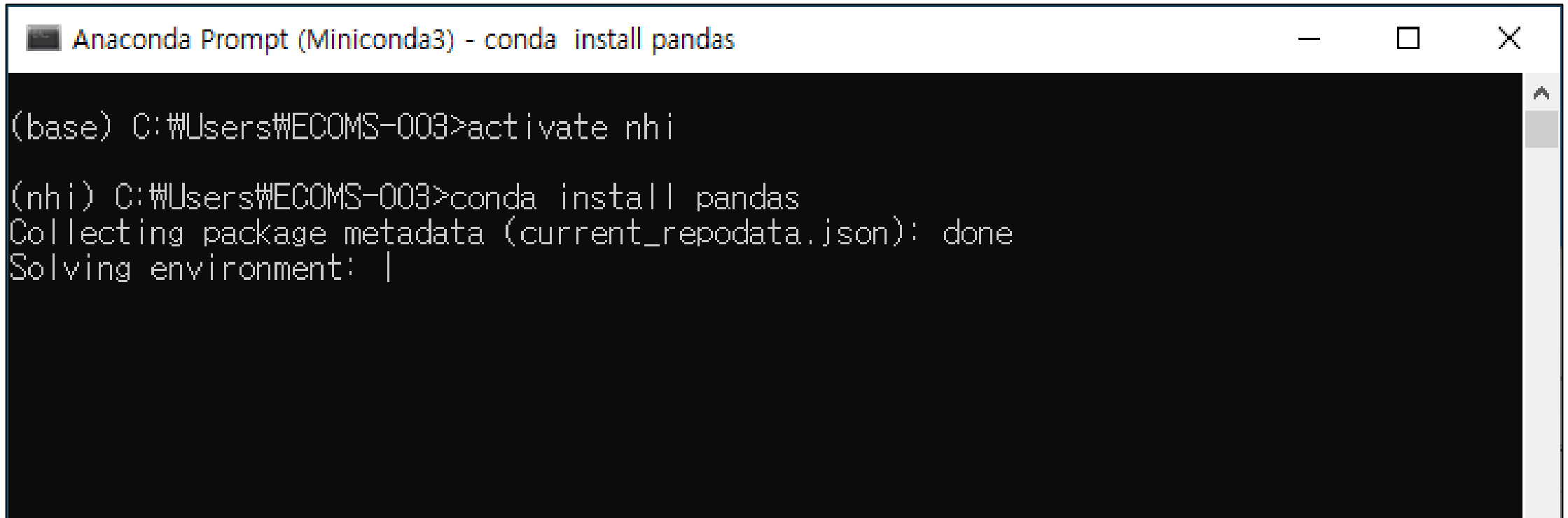
■ Jupyter Notebook(nhi) 실행

- 주피터 노트북에서 새로운 프로젝트 파일 생성([new]->[Python 3] 클릭)
- 코드 작성이 가능한 새창이 생성됩니다.



▣ pandas 모듈 설치

- Anaconda Prompt(Miniconda3)를 실행
- >> activate nhi : nhi 환경으로 이동
- >> conda install pandas : pandas 파일관리 모듈 설치
추가 설치까지 모두 진행



```
Anaconda Prompt (Miniconda3) - conda install pandas

(base) C:\Users\ECOMS-003>activate nhi

(nhi) C:\Users\ECOMS-003>conda install pandas
Collecting package metadata (current_repodata.json): done
Solving environment: |
```

■ csv 자료작성 및 저장

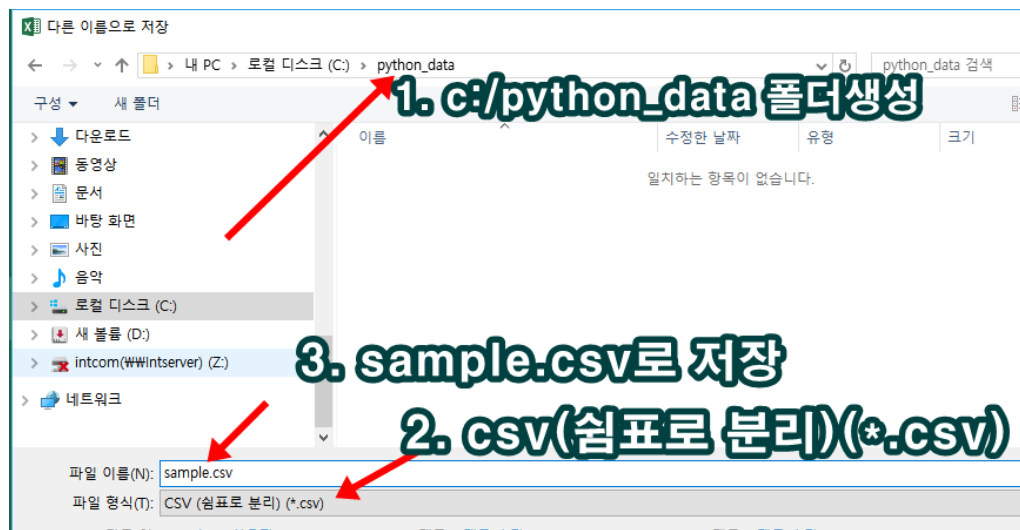
1. 엑셀에서 아래와 같이 자료를 작성함.

	A	B	C	D
1	bun	name	kor	eng
2	1	a	1	2
3	2	b	2	3
4	3	c	2	3
5	4	d	3	2
6	5	e	4	1

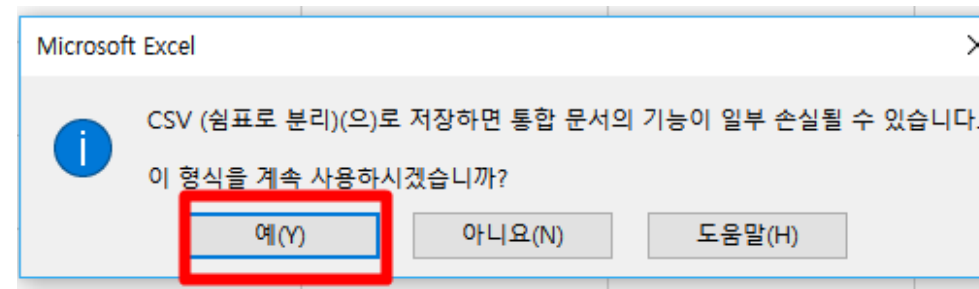
제목(header) : 한글X

data

2. 파일-다른이름으로저장-



3. [예] 선택



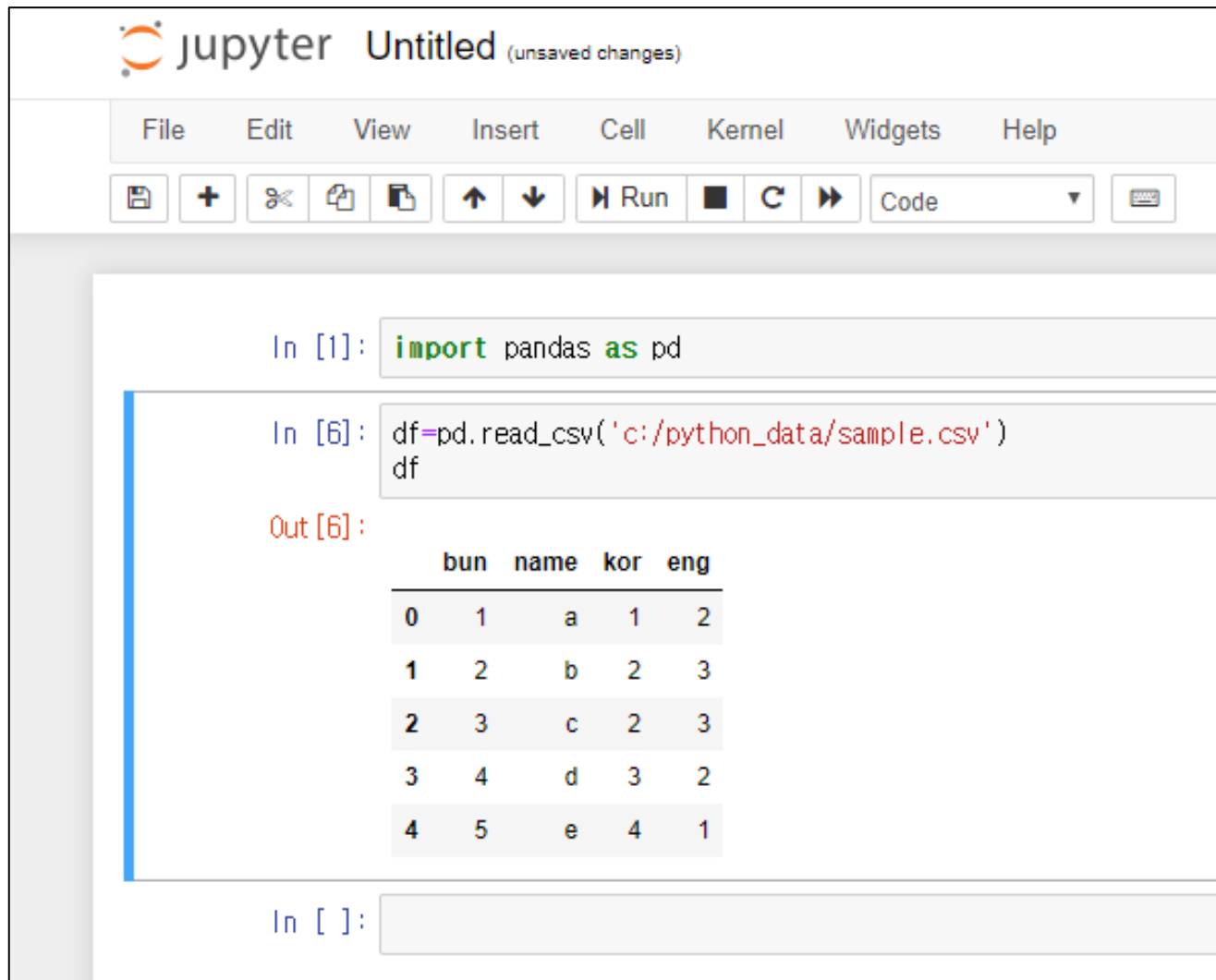
■ 파이썬에서 csv 파일 불러오기

1. Pandas로 csv 자료 읽기

```
import pandas as pd
    ==> pandas 모듈을 pd로 지정
df=pd.read_csv("c:/python_data/sample.csv")
pd
```

주피터 노트북 실행

1. Ctrl+Enter : 현재코드 실행
2. Shift+Enter : 현재코드 실행후
새로운 입력란 생성
3. Alt+Enter : 새로운 입력란 생성



The screenshot shows a Jupyter Notebook interface with the title "jupyter Untitled (unsaved changes)". The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu is a toolbar with icons for saving, adding cells, undo, redo, copy, paste, and running code. The code area contains two input cells. The first cell, labeled "In [1]:", contains the code "import pandas as pd". The second cell, labeled "In [6]:", contains the code "df=pd.read_csv('c:/python_data/sample.csv')" followed by "df" on a new line. Below the second cell, the output is displayed, labeled "Out [6]:". It shows a table with 5 rows and 5 columns: "bun", "name", "kor", and "eng". The data is as follows:

	bun	name	kor	eng
0	1	a	1	2
1	2	b	2	3
2	3	c	2	3
3	4	d	3	2
4	5	e	4	1

At the bottom, there is an empty input cell labeled "In []:".

▣ 파이썬에서 csv 파일 불러오기

```
In [1]: import pandas as pd
```

```
In [6]: df=pd.read_csv('c:/python_data/sample.csv')  
df
```

Out [6]:

	bun	name	kor	eng
0	1	a	1	2
1	2	b	2	3
2	3	c	2	3
3	4	d	3	2
4	5	e	4	1

bun(0번컬럼)
name(1번컬럼)
kor(2번컬럼)
eng(3번컬럼)

```
In [12]: df.columns[0]
```

df.columns[0] → 0번째 컬러명 출력

Out [12]: 'bun'

```
In [13]: df.columns[1]+df.columns[2]
```

df.columns[1]+df.columns[2]

→ 1번째 컬럼값과 2번째 컬럼값 연결

Out [13]: 'namekor'

▣ pandas 파일관리

```
In [14]: df.head(2)
```

```
Out [14]:
```

	bun	name	kor	eng
0	1	a	1	2
1	2	b	2	3

위로2줄 읽기
df.head() 는 5줄 읽기임

```
In [16]: df.values
```

```
Out [16]: array([[1, 'a', 1, 2],  
                [2, 'b', 2, 3],  
                [3, 'c', 2, 3],  
                [4, 'd', 3, 2],  
                [5, 'e', 4, 1]], dtype=object)
```

자료내용 확인

```
In [18]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 5 entries, 0 to 4  
Data columns (total 4 columns):  
#   Column  Non-Null Count  Dtype  
---  -  
0   bun     5 non-null      int64  
1   name    5 non-null      object  
2   kor     5 non-null      int64  
3   eng     5 non-null      int64  
dtypes: int64(3), object(1)  
memory usage: 288.0+ bytes
```

DataFrame 개요 확인

▣ pandas 파일관리

```
In [23]: df.sort_values(by='kor', ascending=False)
```

Out [23]:

	bun	name	kor	eng
4	5	e	4	1
3	4	d	3	2
1	2	b	2	3
2	3	c	2	3
0	1	a	1	2

kor값을 기준으로 ascending 하지 않기(False) => 큰값이 먼저 나오기(내림차순 정렬)

```
In [24]: df.sort_values(by='kor')
```

Out [24]:

	bun	name	kor	eng
0	1	a	1	2
1	2	b	2	3
2	3	c	2	3
3	4	d	3	2
4	5	e	4	1

kor값을 기준으로 ascending ==> 작은값이 먼저나오기(오름차순 정렬)

▣ pandas 파일관리

In [25]: `df.describe()`

자료 기술통계

Out [25]:

	bun	kor	eng
count	5.000000	5.000000	5.000000
mean	3.000000	2.400000	2.200000
std	1.581139	1.140175	0.836666
min	1.000000	1.000000	1.000000
25%	2.000000	2.000000	2.000000
50%	3.000000	2.000000	2.000000
75%	4.000000	3.000000	3.000000
max	5.000000	4.000000	3.000000

개수
평균
표준편차
최소
사분위수

최대값

원본데이터

	A	B	C	D
1	bun	name	kor	eng
2	1	a	1	2
3	2	b	2	3
4	3	c	2	3
5	4	d	3	2
6	5	e	4	1

각자 입력해보세요

- > `df.count()`
- > `df.mean()`
- > `df.min()`
- > `df.max()`

■ pandas 파일관리

표준편차가 크면 숫자값의 분포가 넓다.
kor보다 eng값의 분포도가 더 일정하다.

```
In [27]: df.std()
```

```
Out [27]: bun    1.581139  
kor    1.140175  
eng    0.836660  
dtype: float64
```

(편차) = (각 변량) - (전체 평균)

(분산) = $\frac{\text{편차의 제곱}(\text{편차}^2) \text{의 합}}{\text{총 변량의 개수}}$

(표준편차) = 분산의 제곱근 ($\sqrt{\text{분산}}$)

	A	B	C	D
1	bun	name	kor	eng
2	1	a	1	2
3	2	b	2	3
4	3	c	2	3
5	4	d	3	2
6	5	e	4	1

■ KOR의 평균: 2.4

■ 편차 = (1-2.4), (2-2.4), (2-2.4), (3-2.4), (4-2.4)

■ 분산 = $(1.4*1.4) + (0.4*0.4) + (0.4*0.4) + (-0.6*-0.6) + (-1.6*-1.6)$

5

====> 1.04

■ 표준편차 => 1.04의 제곱근 ==> 1.019804

▣ pandas 파일관리

엑셀에서 eng의 마지막 값을 20으로 변경하고,
bigo 열을 추가한뒤 글자 입력
후 저장(csv로 저장됨)

eng값 변경, 표준편차값 변화확인을 위하여

bigo추가, 한글자료의 읽기 에러 실습

```
In [29]: df=pd.read_csv('c:/python_data/sample.csv')
```

UnicodeDecodeError

Trace

pandas\libs\parsers.pyx in pandas._libs.parsers._box_utf8()

UnicodeDecodeError 'utf-8' codec can't decode byte 0xc0 in position 0: invalid start byte

	A	B	C	D	E	F
1	bun	name	kor	eng	bigo	
2	1	a	1	2		
3	2	b	2	3	재시험	
4	3	c	2	3		
5	4	d	3	2		
6	5	e	4	20		
7						

df=pd.read_csv("c:/python_data/sample.csv")
입력하고 실행하면 에러 나옴

▣ pandas 파일관리

encoding='EUC-KR' 또는 encoding='utf-8' 등 한글을 표현하는 인코딩방식지정

```
In [32]: df=pd.read_csv('c:/python_data/sample.csv', encoding='euc-kr')  
df
```

Out [32]:

	bun	name	kor	eng	bigo
0	1	a	1	2	NaN
1	2	b	2	3	재시험
2	3	c	2	3	NaN
3	4	d	3	2	NaN
4	5	e	4	20	NaN

▣ pandas 파일관리

기술통계 자료를 통해 변경 후 값 확인

```
In [34]: df.describe()
```

Out [34]:

	bun	kor	eng
count	5.000000	5.000000	5.000000
mean	3.000000	2.400000	6.000000
std	1.581139	1.140175	7.842194
min	1.000000	1.000000	2.000000
25%	2.000000	2.000000	2.000000
50%	3.000000	2.000000	3.000000
75%	4.000000	3.000000	3.000000
max	5.000000	4.000000	20.000000

▣ pandas - 열 지정하여 자료 보기

In [35]:

```
df
```



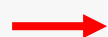
자료확인

Out [35]:

	bun	name	kor	eng	bigo
0	1	a	1	2	NaN
1	2	b	2	3	재시험
2	3	c	2	3	NaN
3	4	d	3	2	NaN
4	5	e	4	20	NaN

In [40]:

```
df[0:2]
```



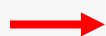
자료명[시작행:끝행]

Out [40]:

	bun	name	kor	eng	bigo
0	1	a	1	2	NaN
1	2	b	2	3	재시험

▣ pandas - 열 지정하여 자료 보기

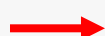
```
In [38]: df['eng']
```



eng 자료만 보기

```
Out [38]: 0      2  
          1      3  
          2      3  
          3      2  
          4     20  
          Name: eng, dtype: int64
```

```
In [39]: df['eng'].std()
```

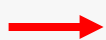


eng 자료의 편차만 보기

```
Out [39]: 7.842193570679061
```

▣ pandas - 두개의 열 지정하여 자료 보기

```
In [43]: df[['eng', 'kor']]
```

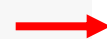


eng, kor 자료확인

```
Out [43]:
```

	eng	kor
0	2	1
1	3	2
2	3	2
3	2	3
4	20	4

```
In [44]: df[['eng', 'kor']].std()
```



eng, kor 표준편차확인

```
Out [44]: eng    7.842194  
kor    1.140175  
dtype: float64
```

■ 파이썬 시각화(차트) 모듈 설치 및 실행

- Anaconda Prompt(Miniconda3)로 이동
- activate nhi : nhi 환경으로 이동
- conda install matplotlib : 파이썬 시각화 모듈 설치(추가 모듈 모두 설치)

```
Anaconda Prompt (Miniconda3) - conda install pandas - conda install matplotlib...  
(nhi) C:\Users\ECOMS-003>conda install matplotlib  
Collecting package metadata (current_repodata.json): done  
Solving environment: done
```

```
==> WARNING: A newer version of conda exists.  
current version: 4.8.2  
latest version: 4.8.3
```

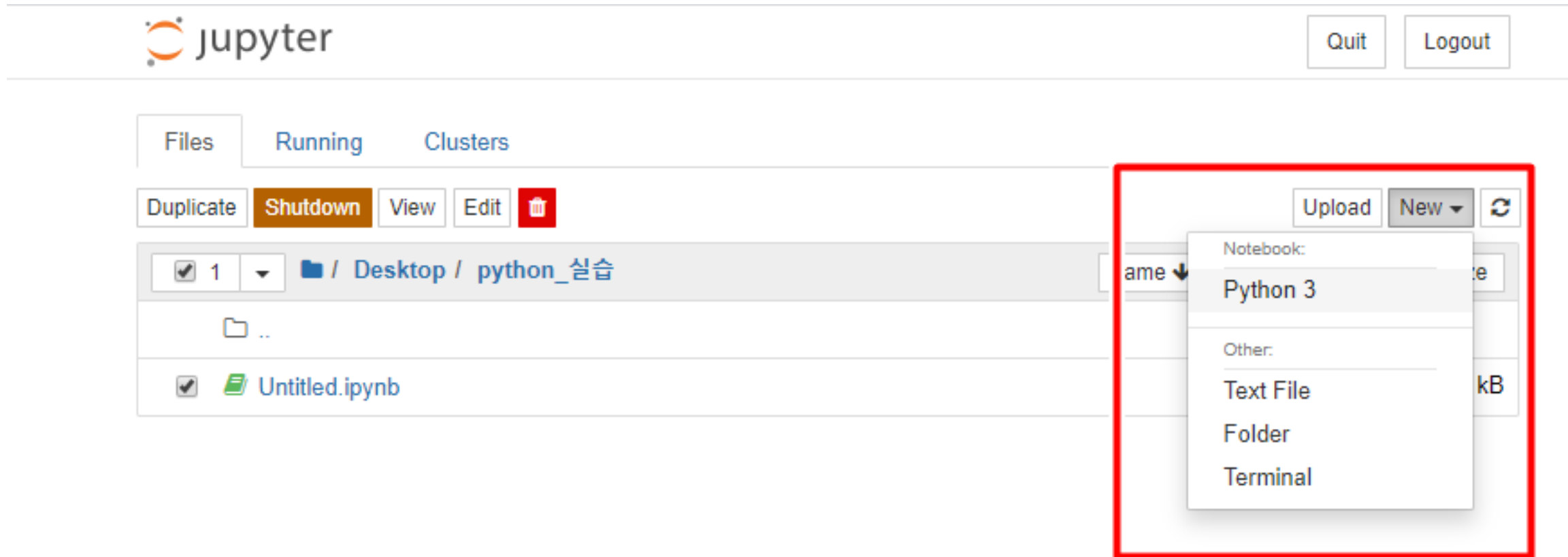
Please update conda by running

```
$ conda update -n base -c defaults conda
```

```
Anaconda Prompt (Miniconda3) - conda install pandas - conda install matplotlib...  
Proceed ([y]/n)? y  
  
Downloading and Extracting Packages  
cycler-0.10.0      | 13 KB      | ##### | 100%  
freetype-2.9.1     | 450 KB     | ##### | 100%  
matplotlib-3.1.3   | 22 KB      | ##### | 100%  
kiwisolver-1.1.0   | 53 KB      | ##### | 100%  
matplotlib-base-3.1. | 4.9 MB     | ##### | 100%  
Preparing transaction: done  
Verifying transaction: done  
Executing transaction: █
```

■ 파이썬 시각화(차트)

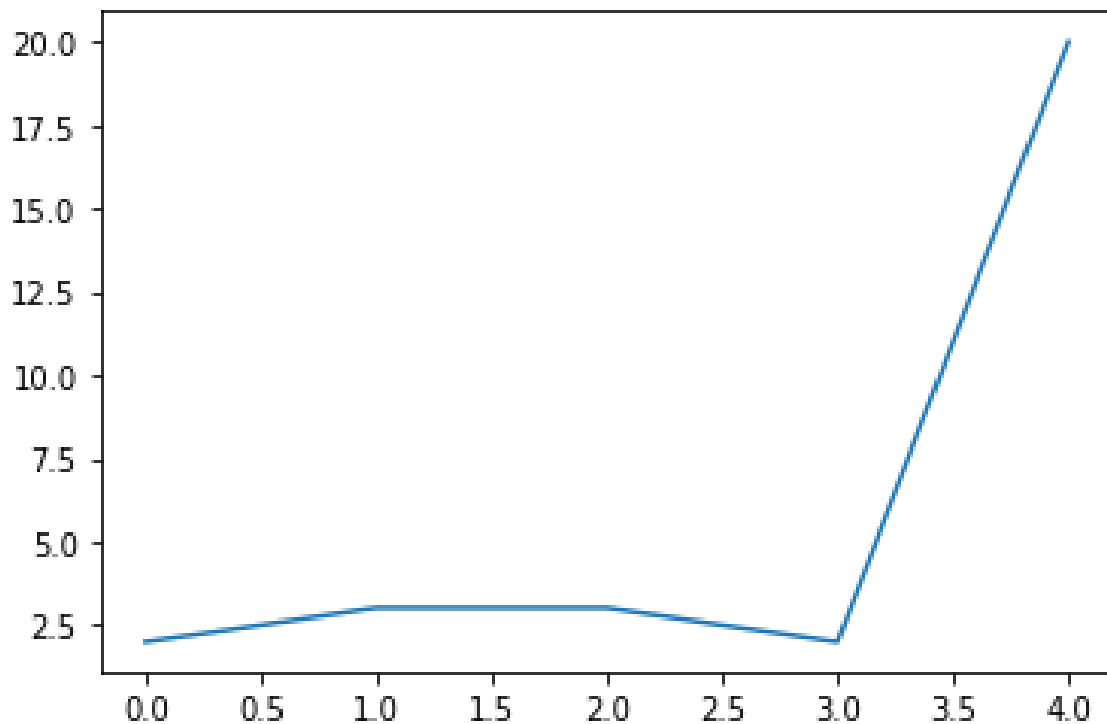
1. 새로운 파이썬 파일 생성



■ 파이썬 시각화(차트)

```
In [6]: import matplotlib.pyplot as plt  
  
plt.figure  
plt.plot(df['eng'])  
plt.show()
```

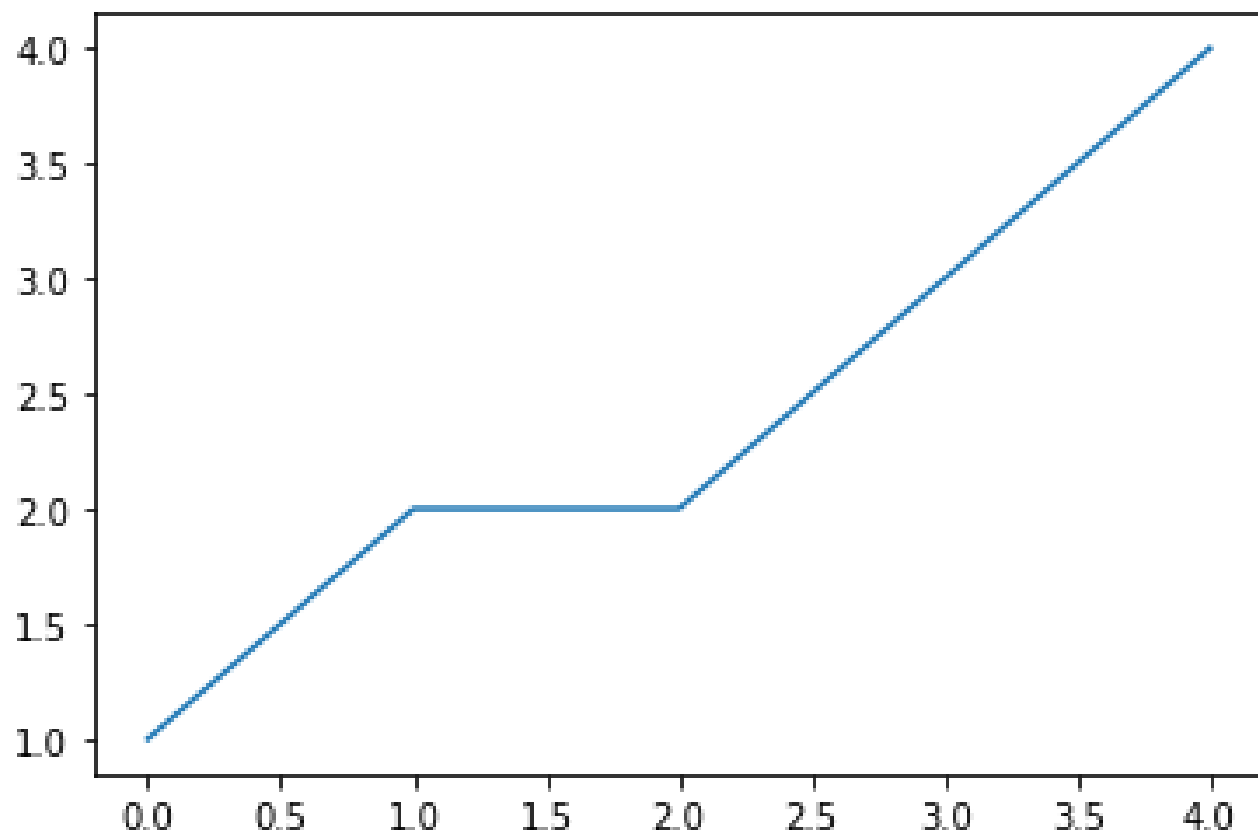
eng자료



■ 파이썬 시각화(차트)

```
In [7]: plt.plot(df['kor'])  
plt.show()
```

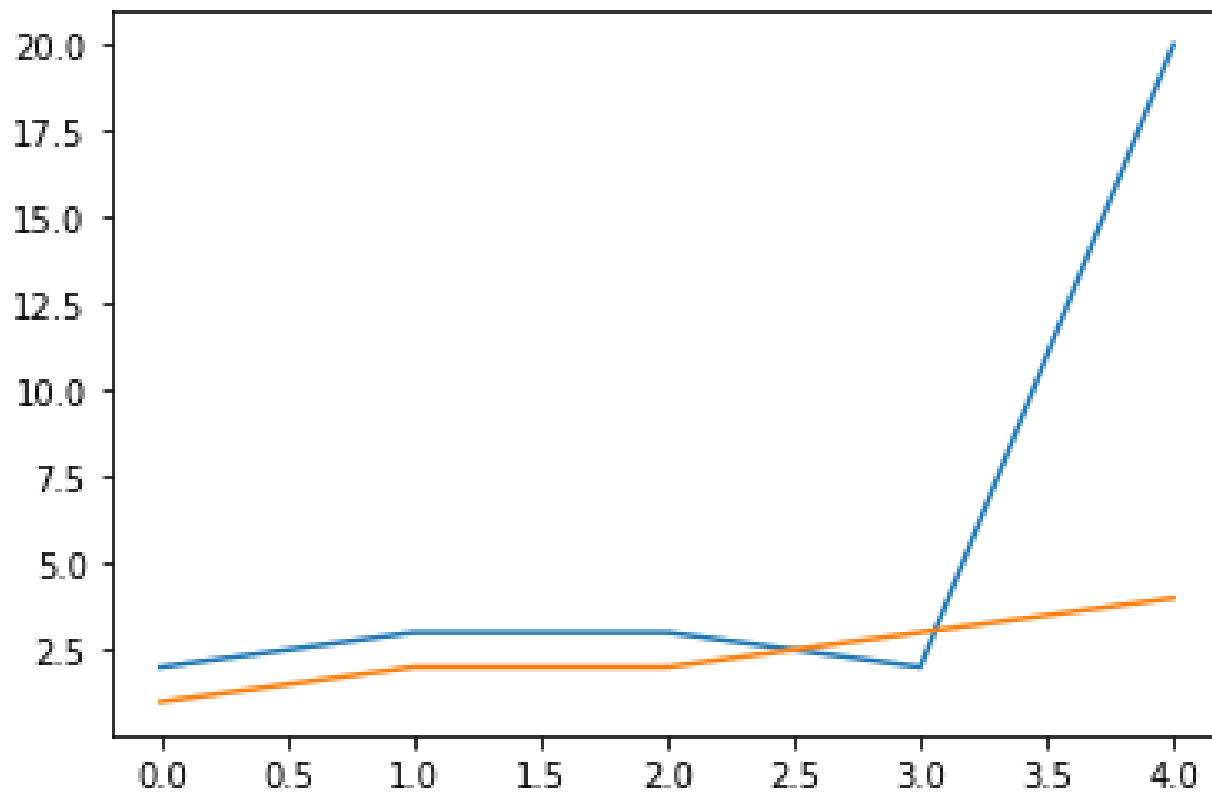
kor자료



■ 파이썬 시각화(차트)

```
In [12]: plt.plot(df[['eng', 'kor']])  
plt.show()
```

eng, kor 자료

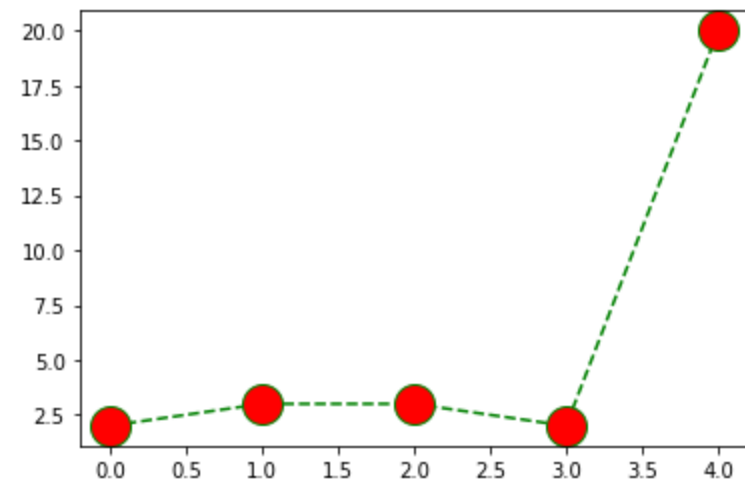


■ 파이썬 시각화(차트)

```
In [1]: import pandas as pd
df=pd.read_csv("c:/python_data/sample.csv",encoding="EUC-KR")
print(df)
```

```
In [2]: import matplotlib.pyplot as plt
plt.figure
plt.plot(df["eng"],color='green',linestyle='dashed',marker='o',
         markerfacecolor='red', markersize=20)
plt.show()
```

주의: 파이썬은 enter하여 두줄 이상의 명령은
들여쓰기가 잘 맞아야함



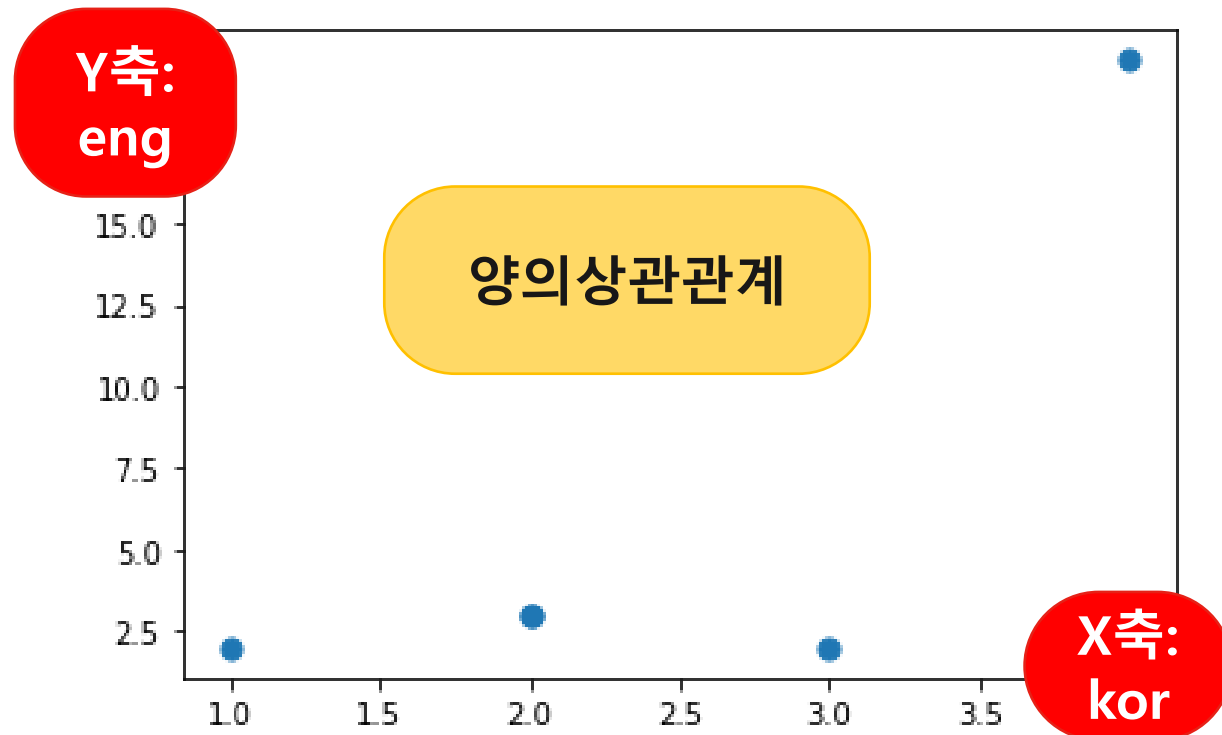
■ 파이썬 시각화(차트) - 상관계수 확인 및 산점도 차트

```
In [3]: print(df)
        print(df.corr())
```

	bun	name	kor	eng	bigo
0	1	a	1	2	NaN
1	2	b	2	3	재시험
2	3	c	2	3	NaN
3	4	d	3	2	NaN
4	5	e	4	20	NaN

	bun	kor	eng
bun	1.000000	0.970725	0.705668
kor	0.970725	1.000000	0.782868
eng	0.705668	0.782868	1.000000

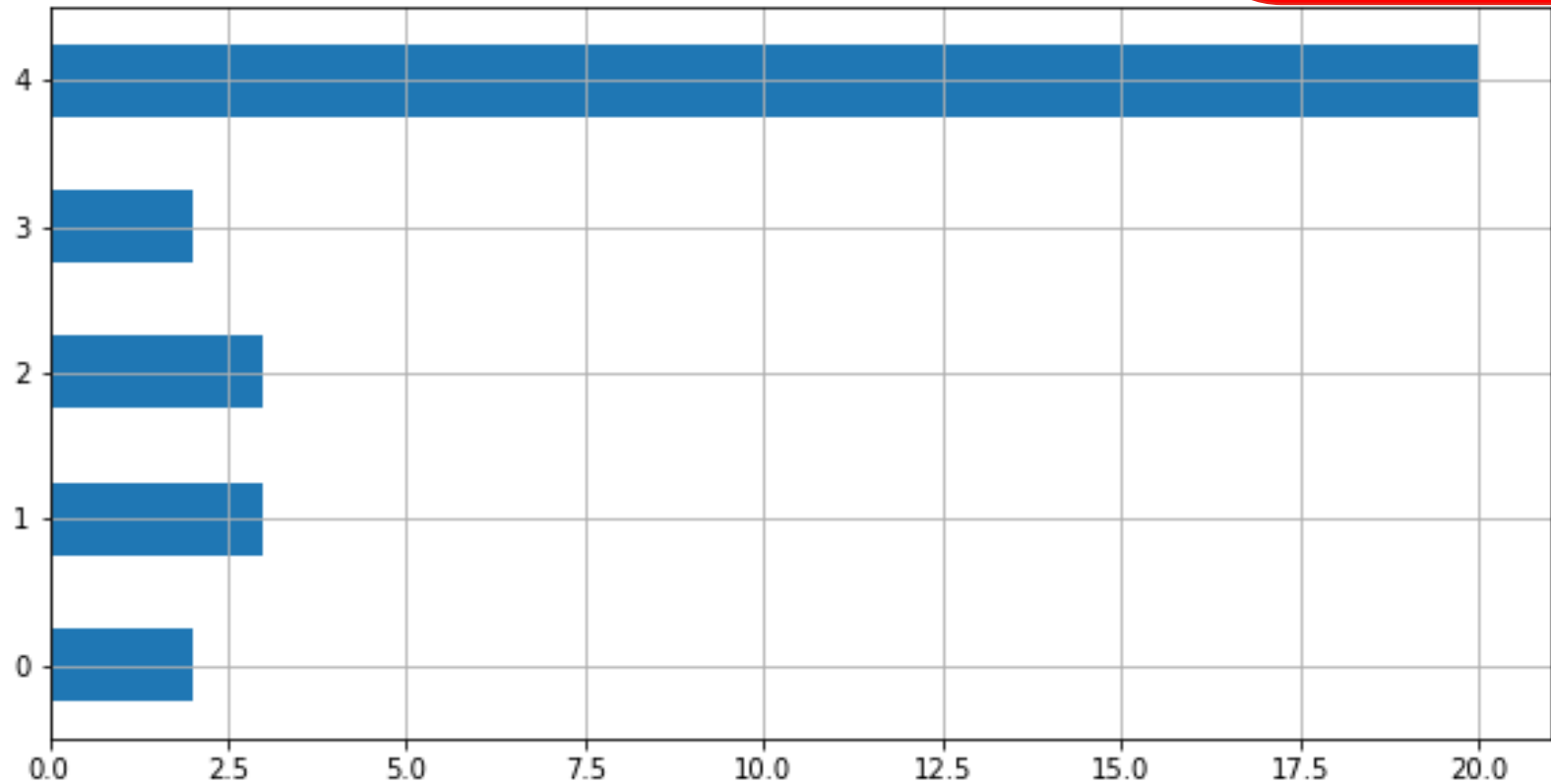
```
In [4]: plt.scatter(df["kor"],df["eng"],s=50)
        plt.show()
```



■ 파이썬 시각화(차트) - 가로막대

```
In [5]: df["eng"].plot(kind="barh", grid=True, figsize=(10,5))  
plt.show()
```

figsize=(10,5)
가로가 세로보다 2배 큼

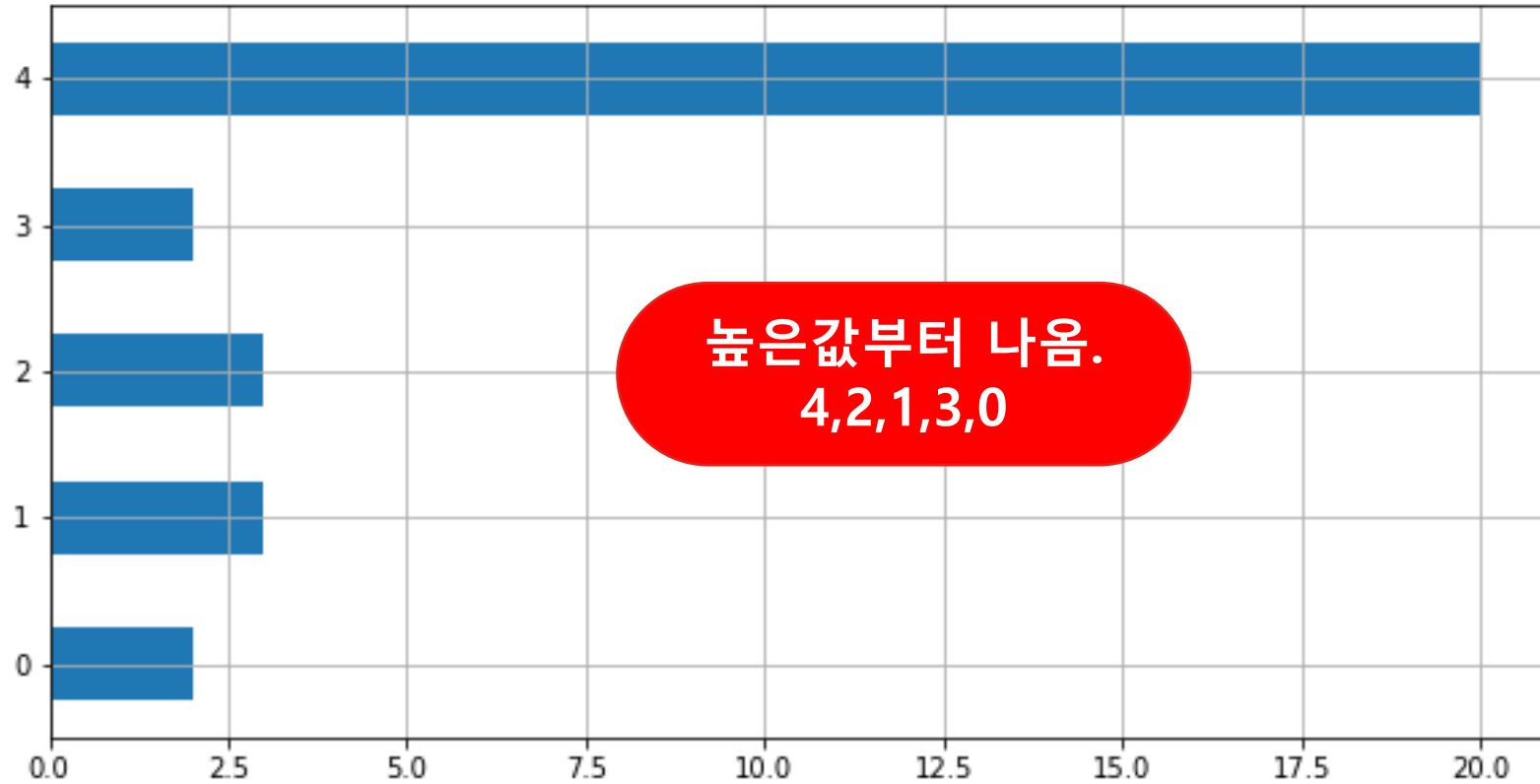


세로(5)

가로(10)

■ 파이썬 시각화(차트) - 가로막대

```
In [5]: df["eng"].plot(kind="barh", grid=True, figsize=(10,5))  
plt.show()
```



bun	name	kor		eng
1	a	1	0	2
2	b	2	1	3
3	c	2	2	3
4	d	3	3	2
5	e	4	4	20

▣ 공공데이터를 활용한 pandas 파일관리 실습

제공된 파일을 이용하여 실습 진행

1. seoul_in_cctv.csv 파일 'c:/python_data' 폴더에 복사
2. 'pandsa_파일 관리.html' 파일 내용을 이용해 실습 진행
복사 붙이기 후 실행