

Numpy, Pandas 튜토리얼

안녕하세요. 컴퓨터공학과 안영후입니다.

이번에는 Python에서 데이터 분석할 때, 꼭 필요한 라이브러리인 Numpy, Pandas에 대해서 알아보고 설치 및 사용하는 방법까지 알아보도록 하겠습니다.

목차

1. Numpy란?
2. Pandas란?
3. Jupyter notebook으로 설치없이 편하게 오픈소스 라이브러리 이용하기
4. 데이터 시각화를 위한 Matplotlib 라이브러리 사용해보기

1. Numpy란?

Numpy는 numerical_python의 약자로 Python에서 대규모 다차원 배열을 다룰 수 있게 도와주는 오픈소스 라이브러리입니다. 파이썬에서의 배열을 좀 더 쉽게 다룰 수 있으며, 특히 파이썬 자체에서의 연산 속도보다 Numpy에서의 연산 속도가 훨씬 빠르다는 이점이 있어, 데이터 분석 분야에서 많이 활용됩니다. 연산 속도가 빠른 이유는 파이썬은 파이썬 그 자체로 속도가 느리지만, Numpy는 핵심 기능들의 소스코드가 모두 C++로 작성되어 있기 때문에 연산 속도와 메모리 효율 측면에서 압도적으로 효율적일 수 밖에 없습니다.

2. Pandas란?

Pandas는 구조화된 데이터를 효과적으로 처리하고 저장할 수 있으며, 연산에 특화된 Numpy를 기반으로 설계된 오픈소스 라이브러리입니다. 또한, DataFrame이라는 데이터 구조를 지원함으로써, 데이터들을 보기 쉽게 시각화 해주는 데 도움을 주고, index 및 column으로 이루어진 사이에

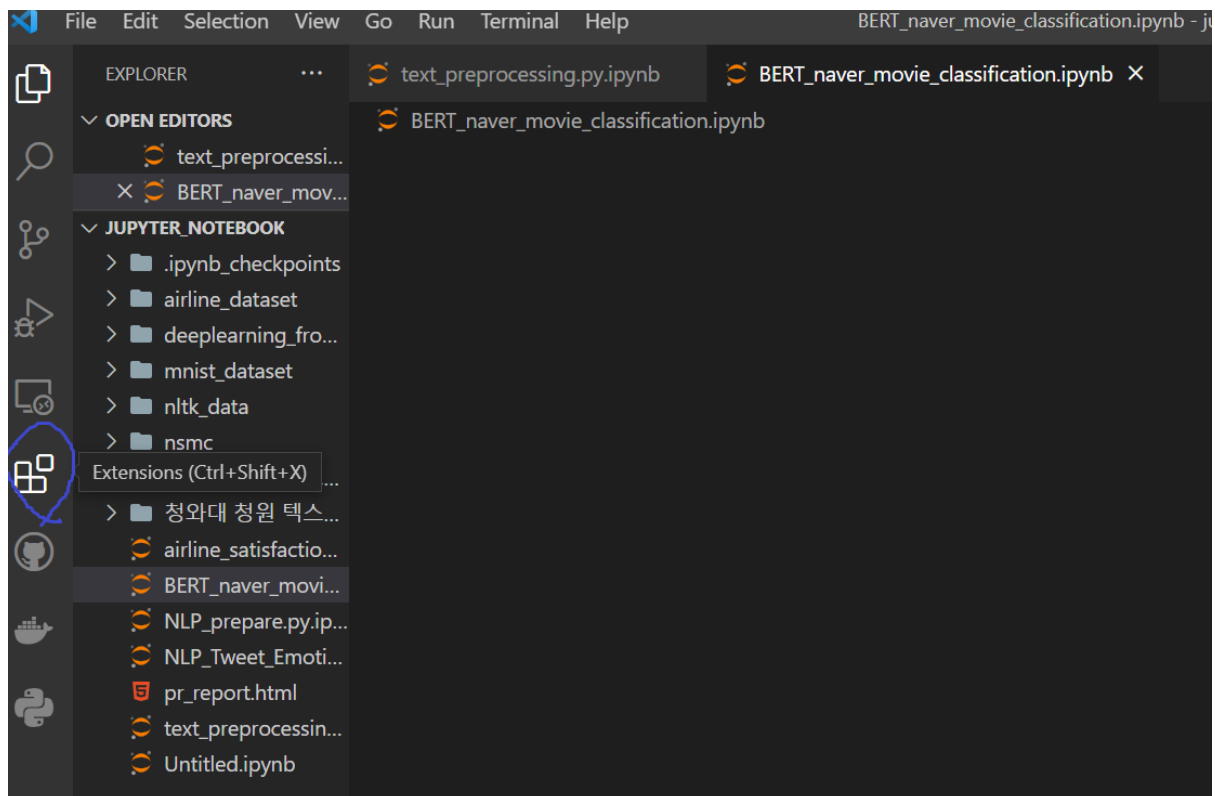
value들을 담고 있기 때문에 효율적으로 데이터 분석이 가능토록 합니다.

자, 그럼 본격적으로 Numpy, Pandas를 이용하도록 해보겠습니다.

3. Jupyter notebook으로 설치없이 오픈소스 라이브러리 이용하기

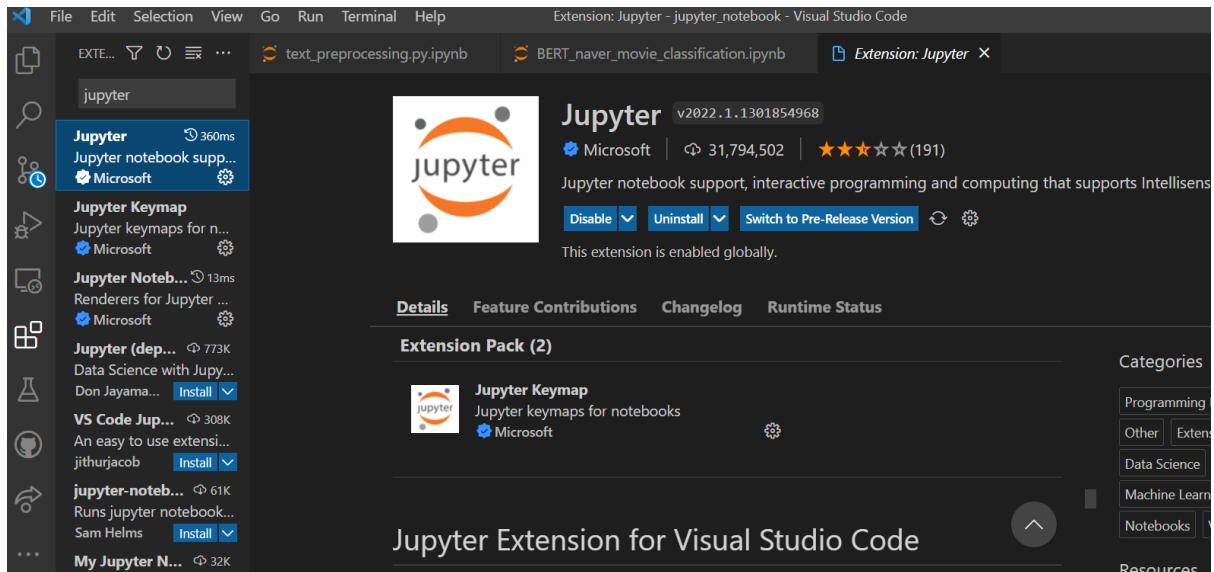
Jupyter notebook을 이용하는 방법은 여러가지가 있습니다. 구글의 colab을 이용할 수도 있고, 웹 브라우저에서 이용할 수도 있고, visual studio code라는 ide에서 이용할 수도 있습니다.

여기서는 visual studio code를 사용하여 설치해보도록 하겠습니다.

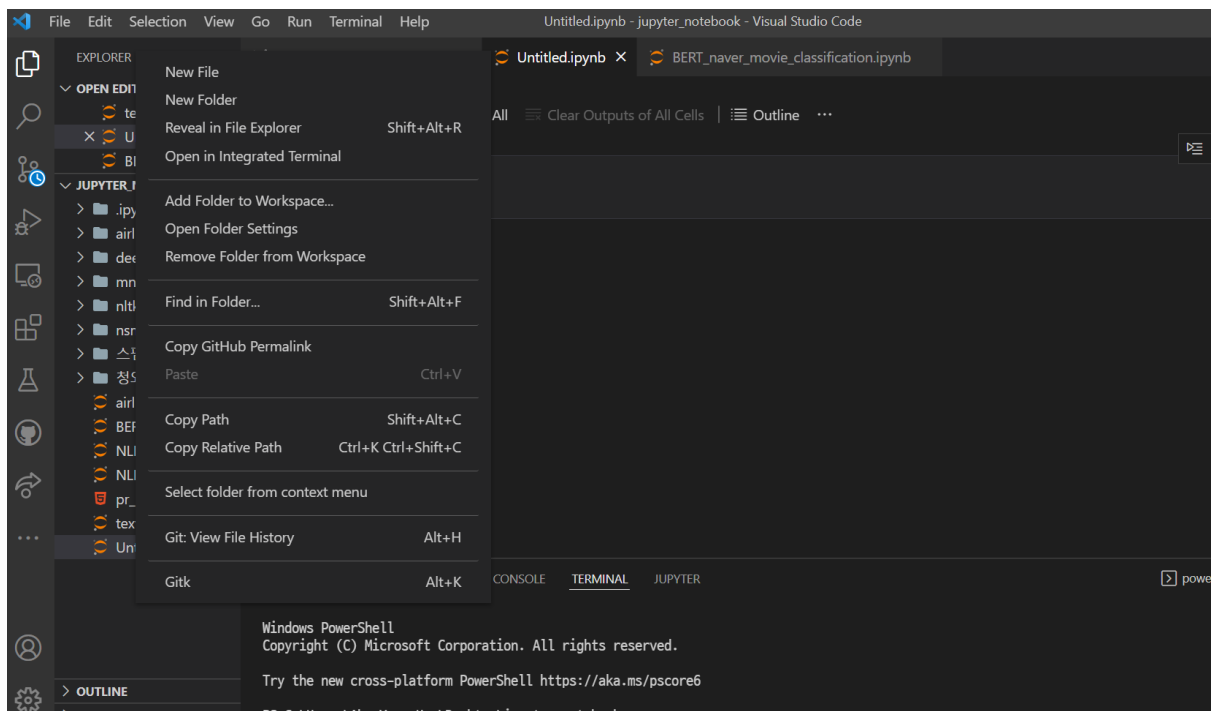


우선 vscode를 켜신 다음 extension 탭에 들어갑니다.

들어가시면 검색창이 나오는데, 검색창에 jupyter라고 입력 후 검색합니다.

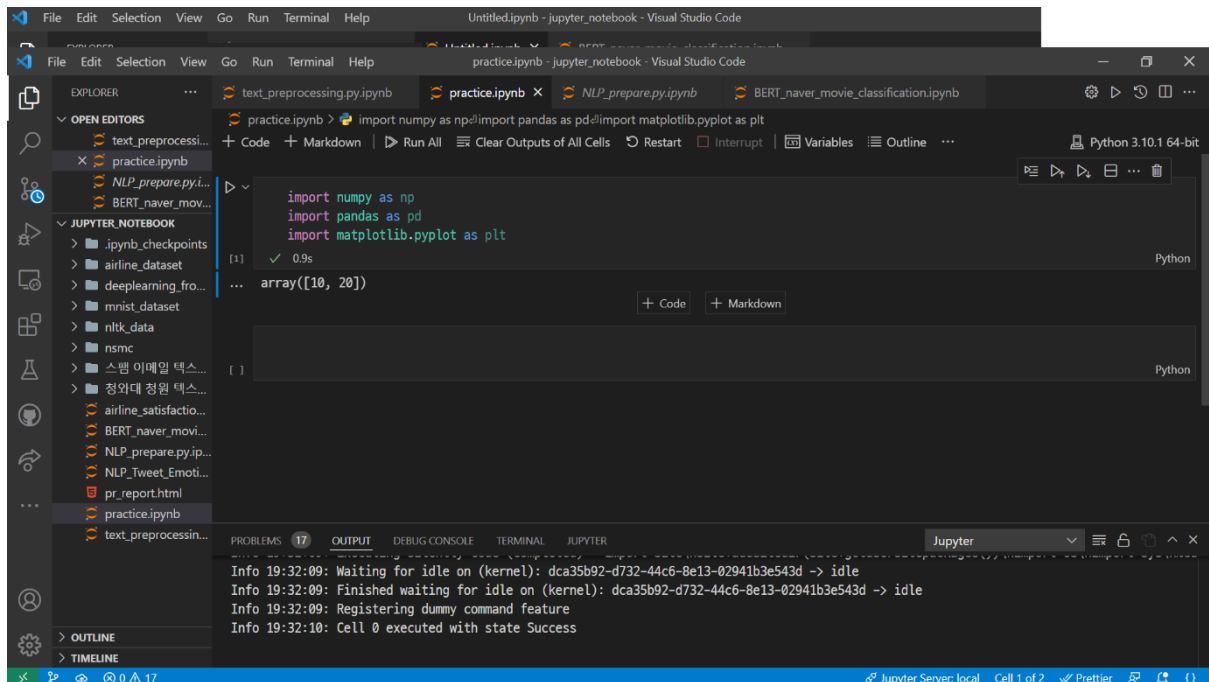


최상단에 나오는 jupyter라는 익스텐션을 설치합니다.



좌측 폴더와 파일들이 위치해있는 부분에서 비어있는 부분에 마우스 커서를 올리고, 좌클릭을 해서 new file을 눌러 새파일을 생성해주고, 파일명은 원하는 대로 만드시는데, 확장자를 .ipynb로 만드셔야 쥬피터 파일이 정상 작동합니다.

저는 practice.ipynb로 만들겠습니다!

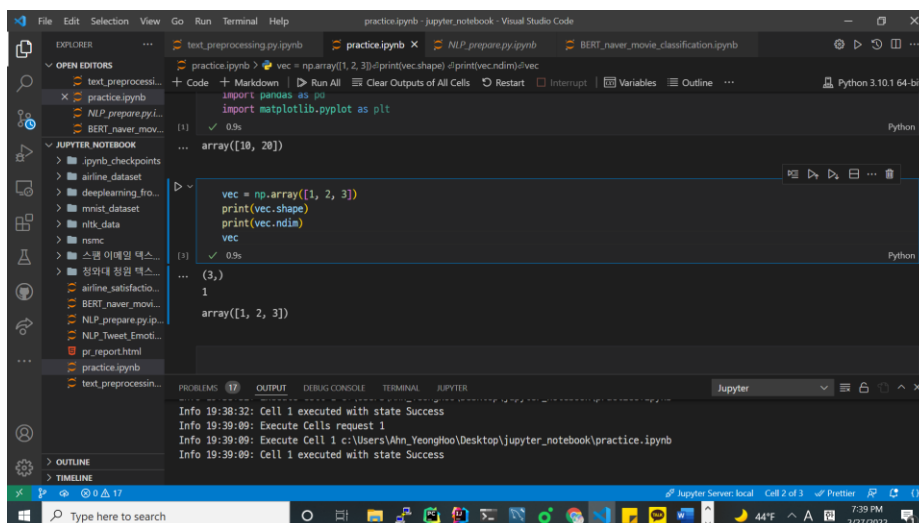


그 후, 세 개의 module을 import 해줄건데, module이란 미리 만들어놓은 기능들의 집합이라고 생각하시면 되고, import는 그 기능들의 집합을 불러오는 파이썬의 문법입니다!

따라서, 우리가 사용하고자 하는 numpy, pandas, matplotlib를 import 문법을 통해 불러오도록 합니다. 그 후 관례에 따라서 numpy는 np, pandas는 pd, matplotlib는 plt이라고 별칭을 붙여줍니다.

이렇게 불러오게 되면 앞으로 method를 사용할 때마다 numpy와 같이 풀네임을 사용하지 않고, np와 같이 별명을 통해 method를 사용할 수 있게 됩니다!

그럼, 본격적으로 numpy와 pandas를 이용해보도록 하겠습니다!



numpy에서는 array라는 메서드를 통해 다차원 배열을 생성할 수 있습니다.

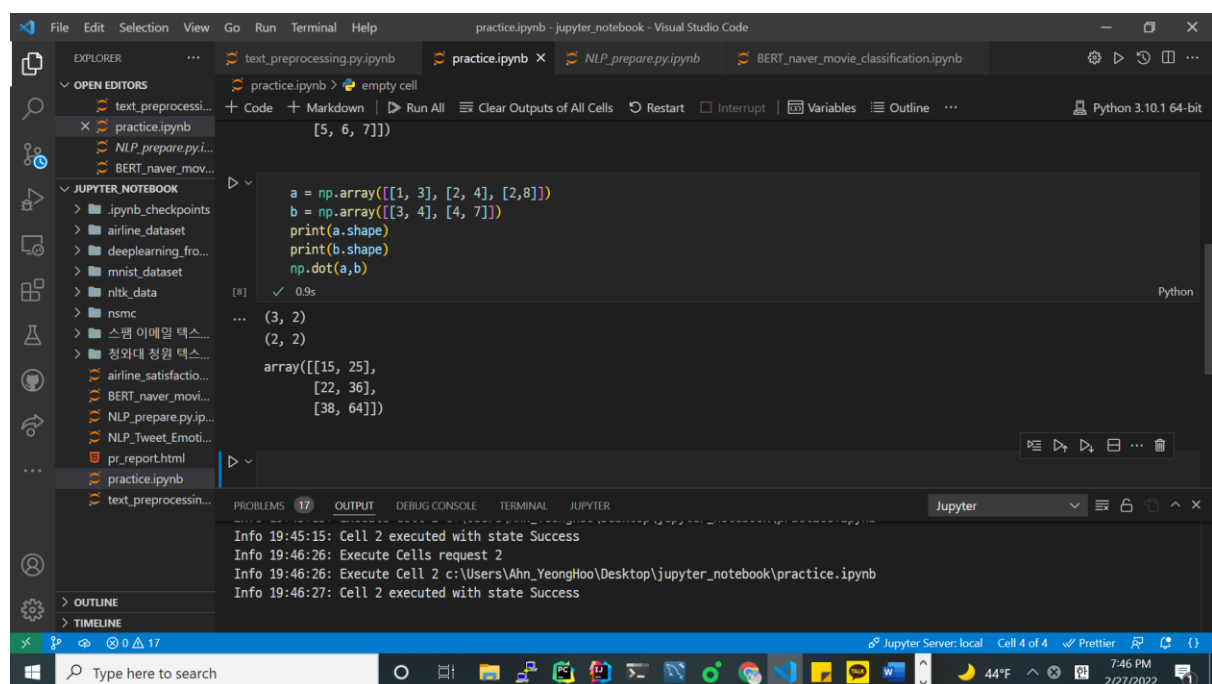
또, 생성한 다차원 배열을 출력하게 되면, 파이썬의 리스트와 다르게 보기 좋게 행과 열로 잘 정렬되어 출력해줍니다. 또한, shape라는 메서드를 이용하면 배열의 형상(몇 행, 몇 열인지)을 알려주고, ndim 메서드는 배열이 몇차원 배열인지 알려줍니다. 참고로 쥬피터 노트북에서는 print 함수없이 바로 결과를 작성해 실행하면, 마지막으로 작성된 결과만 출력하게 됩니다.

따라서 그 이전에 출력하고자 했던 shape와 ndim은 print를 통해 출력해줍니다.

또한, shift + enter 단축키를 이용하면 곧바로 실행결과를 보실 수 있습니다.

Numpy가 강력한 이유는 바로 다차원 배열의 계산을 빠르게 해주는 데 있다고 했죠.

행렬의 내적을 계산하는 것도 numpy는 아주 간편하게 해줍니다.



The screenshot shows a Jupyter Notebook interface within Visual Studio Code. The notebook is titled 'practice.ipynb' and contains a code cell with the following Python code:

```
a = np.array([[1, 3], [2, 4], [2, 8]])
b = np.array([[3, 4], [4, 7]])
print(a.shape)
print(b.shape)
np.dot(a,b)
```

The output of the code cell is displayed below the code:

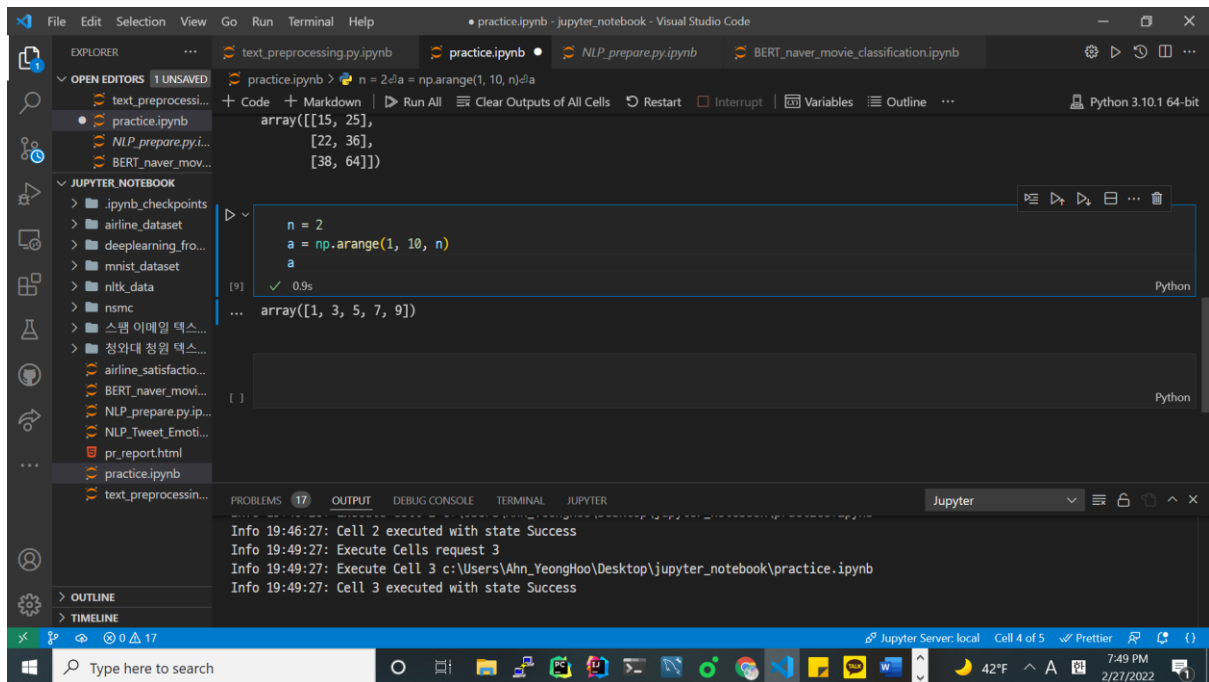
```
(3, 2)
(2, 2)
array([[15, 25],
       [22, 36],
       [38, 64]])
```

The bottom status bar indicates 'Jupyter Server: local' and 'Cell 4 of 4'. The Windows taskbar at the bottom shows the time as 7:46 PM on 2/21/2022.

바로 dot 메서드를 통해, 행렬의 내적(행렬곱) 연산을 아주 빠르게 할 수 있습니다.

하지만 주의할 점은, 연산할 두 배열의 행렬을 잘 맞추어야 하는데요.

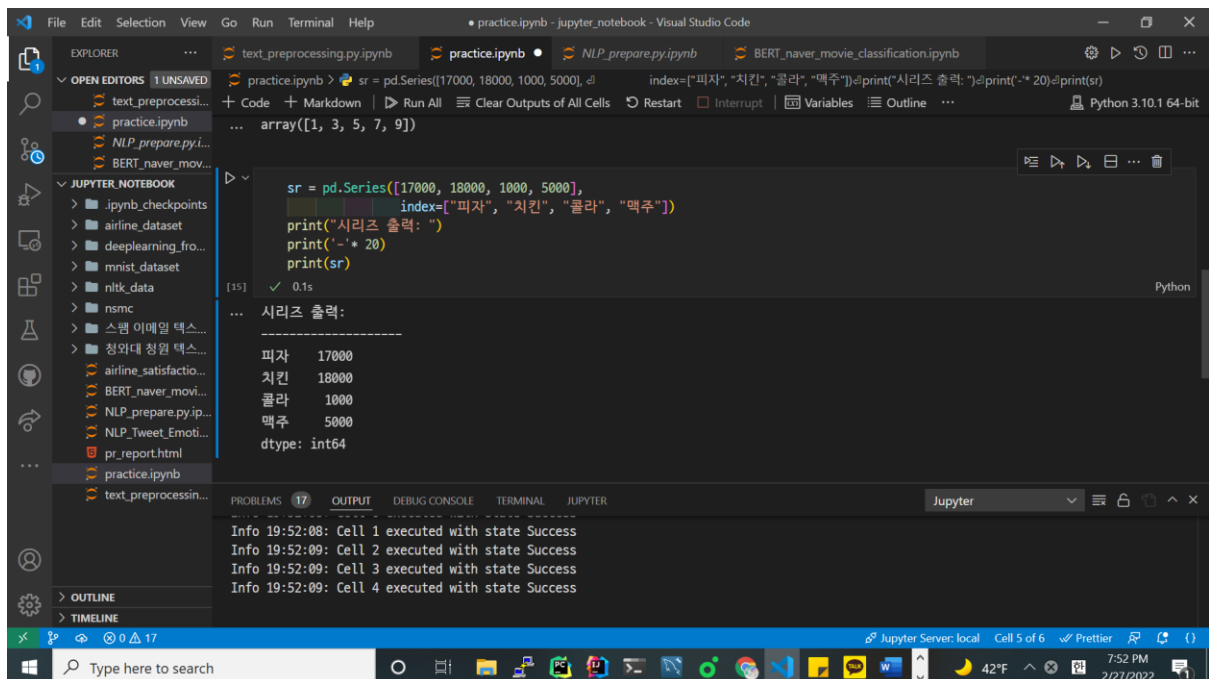
연산하고자 하는 첫 번째 배열의 1번 인덱스와 두 번째 배열의 0번 인덱스의 수가 같아야만 연산이 가능합니다! 그렇지 않으면 에러를 발생시키니 주의해야 합니다.(참고로 파이썬의 인덱스는 0번부터 시작합니다)



또한 `arange` 메서드를 이용하면 1부터 10까지 `n` 간격으로 배열을 생성 해줍니다.

저기선 `n`이 2므로 2 간격으로 생성해주었네요.

`pandas`로도 비슷한 배열을 생성할 수 있는데, `pandas`에서는 1차원 배열을 `Series` 클래스라고 부르고, 2차원 배열을 `DataFrame` 클래스라고 부릅니다.



우선 `Series` 클래스를 이용하여 1차원 배열을 생성해주는데, 사진과 같이 `index`명을 지정 해 줄 수 있습니다.

```

values = [[1,2,3], [4,5,6], [7,8,9]]
index = ['one', 'two', 'three']
columns = ['A', 'B', 'C']
df = pd.DataFrame(values, index=index, columns=columns)

print("데이터프레임 출력: ")
print("-"*20)
print(df)

```

데이터프레임 출력:

	A	B	C
one	1	2	3
two	4	5	6
three	7	8	9

이제 DataFrame을 통해 2차원 배열을 생성해봅시다. Values는 말 그대로 값을 의미하고, index는 행렬의 행을 의미하며, column은 행렬의 열을 의미합니다. Values는 행과 열 사이의 값으로써 자리하게 됩니다. DataFrame 메서드를 통해 생성할 수 있습니다.

```

data = {
    '학번': ['1000', '1001', '1002', '1003', '1004', '1005'],
    '이름': ['안영후1', '안영후2', '안영후3', '안영후4', '안영후5', '안영후6'],
    '점수': [90.72, 78.09, 98.43, 64.19, 81.30, 99.14],
}
df = pd.DataFrame(data)
print(df)

```

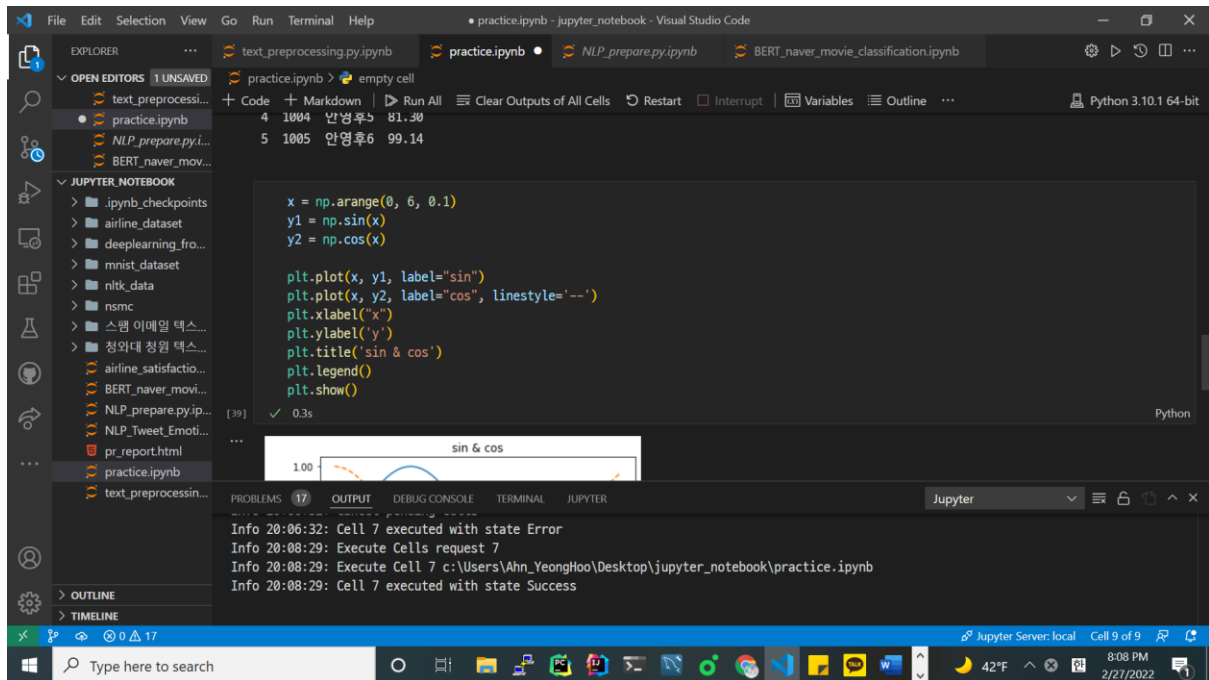
학번 이름 점수

0	1000	안영후1	90.72
1	1001	안영후2	78.09
2	1002	안영후3	98.43
3	1003	안영후4	64.19
4	1004	안영후5	81.30

또한, 파이썬의 자료형인 딕셔너리를 통해서 DataFrame을 생성할 수 있습니다. 여기서 딕셔너리의 key는 column이 되고, values는 values로 대입됩니다. Index는 따로 지정하지 않아서 0번부터

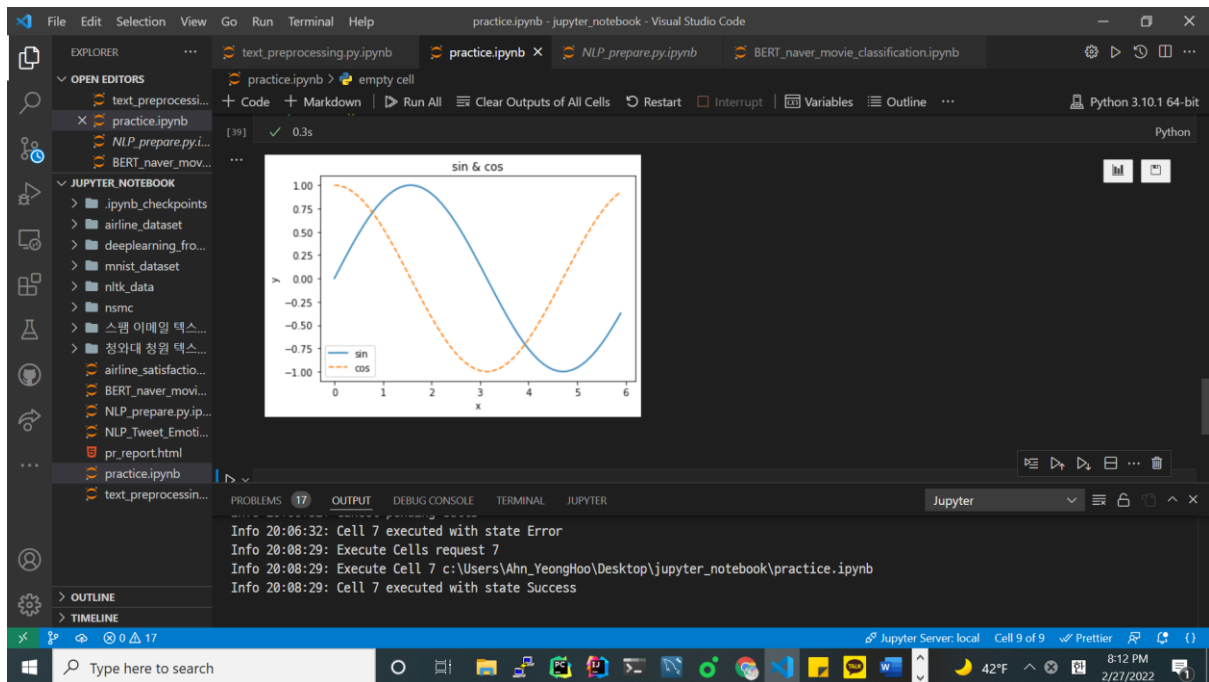
시작해 순서대로 행(row)이 생성됩니다.

이제 마지막으로 matplotlib라는 데이터시각화 라이브러리를 이용해 sin 그래프와 cos 그래프를 그려보도록 하겠습니다.



x에는 0부터 6까지 0.1 간격으로 배열을 생성해주고, sin과 cos 메서드를 통해 sin, cos 데이터를 각각 y1, y2 변수에 대입해줍니다. Matplotlib의 plot 메서드를 이용하여 각각 sin 그래프, cos 그래프를 생성 해줍니다. Label를 통해 각 그래프의 이름을 지정해줄 수 있고, linestyle 옵션을 --로 주면, 점선으로 나타납니다.

xlabel은 x축의 이름을, ylabel은 y축의 이름을 지정해줄 수 있고, title 메서드를 통해 그래프 전체의 제목을 정해줍니다. Legend 메서드는 범례를 설정해줄 수 있는 메서드로, 아무 값도 지정해주지 않으면 plot에서 만들어졌던 label값이 들어가게 됩니다. 마지막으로 show 함수를 이용해 그래프를 출력하면 그래프가 출력됩니다!



지정해준 것 처럼 cos 함수 그래프는 점선으로 표시되는 것을 알 수 있습니다.