Tutorial

Saemi Moon

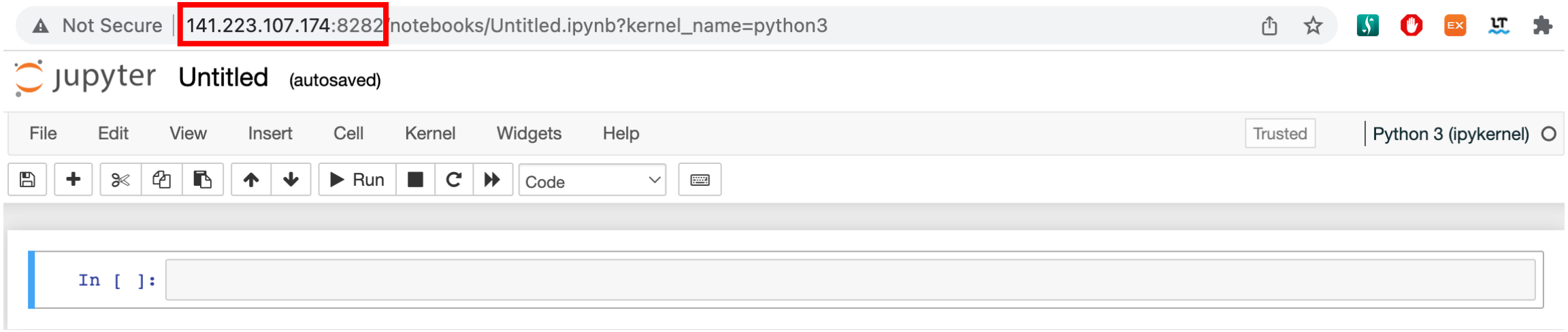
Machine Learning Laboratory
POSTECH

Table of Contents

Table of Contents

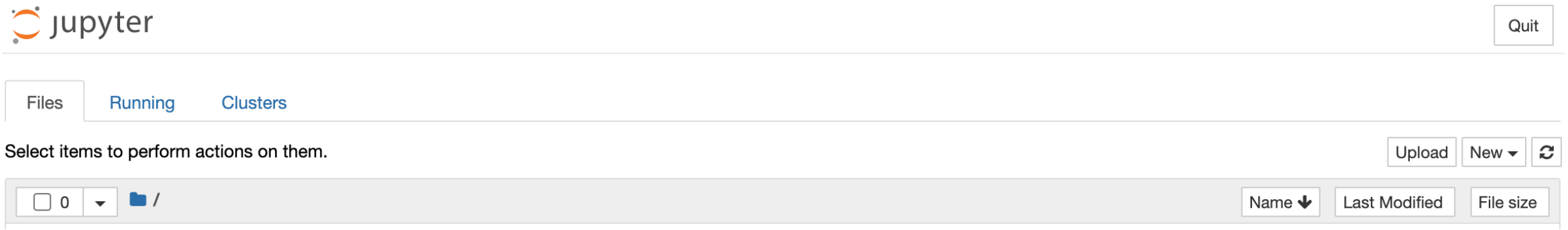
Jupyter notebook이란?

- 대화형 인터프리터(interpreter)
- 웹 어플리케이션
- 특정 url 또는 local로 접속



Jupyter notebook 접속

- 웹 브라우저를 열고 주소창에 아래 주소 입력
  - [해당 명령어를 입력한 ip]:8282
  - Ex: 141.223.000.000:8282
- 접속 화면



Tensorflow 2.x 설치 및 gpu 인식 확인

- Tensorflow 2.x 설치 확인 코드
  - import tensorflow as tf
  - print(tf.__version__)
- gpu 인식 확인 코드
  - !nvidia-smi
- 오른쪽과 같이 출력되면 제대로 설치된 것 :)

```
In [1]: import tensorflow as tf
        print(tf.__version__)

        2022-05-10 08:44:14.221202: I tensorflow/stream_executor/platform/default/dso_loa
        ic library libcudart.so.10.1

        2.3.0
```

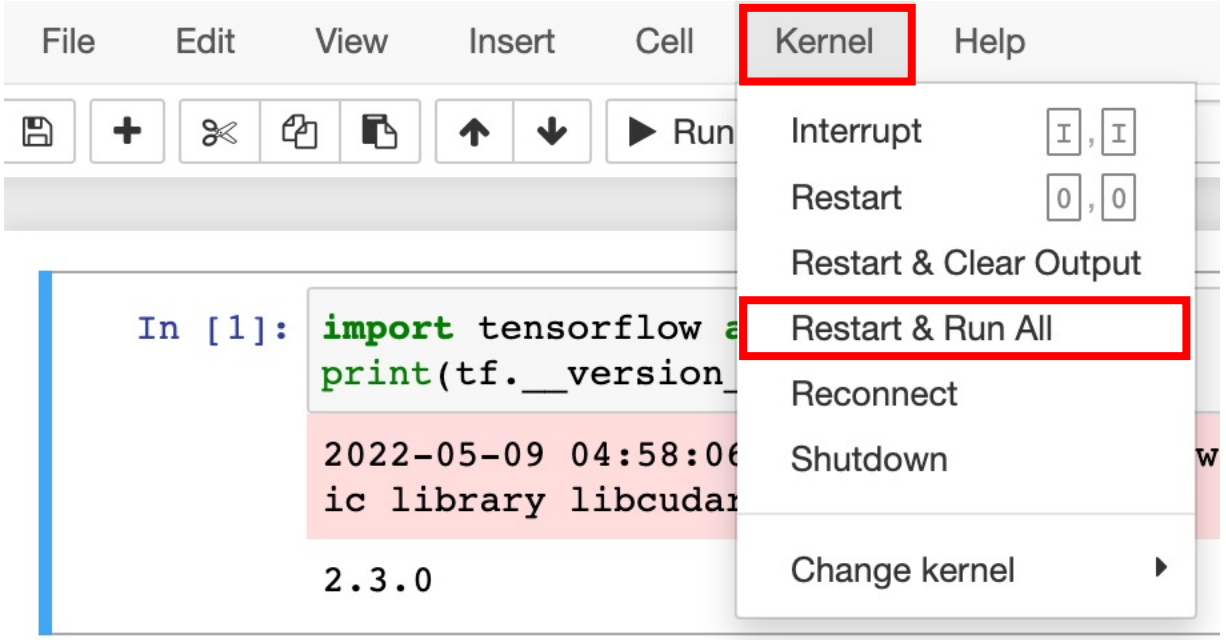
```
In [2]: !nvidia-smi

        Tue May 10 08:44:18 2022
        +-----------------------------------------------------------------------------+
        | NVIDIA-SMI 440.82       Driver Version: 440.82       CUDA Version: 10.2     |
        |-------------------------------+----------------------+----------------------+
        | GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
        | Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
        |===============================+======================+======================|
        |   0  GeForce GTX 108...  Off  | 00000000:03:00.0 Off |                  N/A |
        | 26%   43C    P0    54W / 250W |      0MiB / 11176MiB |      0%      Default |
        +-------------------------------+----------------------+----------------------+
        |   1  GeForce GTX 108...  Off  | 00000000:04:00.0 Off |                  N/A |
        | 27%   41C    P5    18W / 250W |      0MiB / 11178MiB |      3%      Default |
        +-------------------------------+----------------------+----------------------+

        +-----------------------------------------------------------------------------+
        | Processes:                                                       GPU Memory |
        |  GPU       PID   Type   Process name                             Usage      |
        |=============================================================================|
        |  No running processes found                                                 |
        +-----------------------------------------------------------------------------+
```

종종 사용되는 오류 해결법 - 1

- Kernel >> Restart & Run All
  - 대화형 인터프리터는 중간중간 출력값을 확인할 수 있어서 편리하지만, 셀 실행 순서가 꼬여 의도와 다른 결과가 출력될 수 있음
  - 커널을 재시작해 저장된 변수값을 모두 지우고 셀을 순서대로 실행하여 문제 해결 가능



종종 사용되는 오류 해결법 - 2

- Shutdown
  - 오류 메세지에 "~ cuda ~ memory ~" 키워드가 있는 경우, 이전에 돌렸던 코드가 여전히 gpu 메모리를 차지하고 있을 가능성 있음
  - 현재 사용하지 않는 커널을 "Shutdown"하면 해결되기도 함

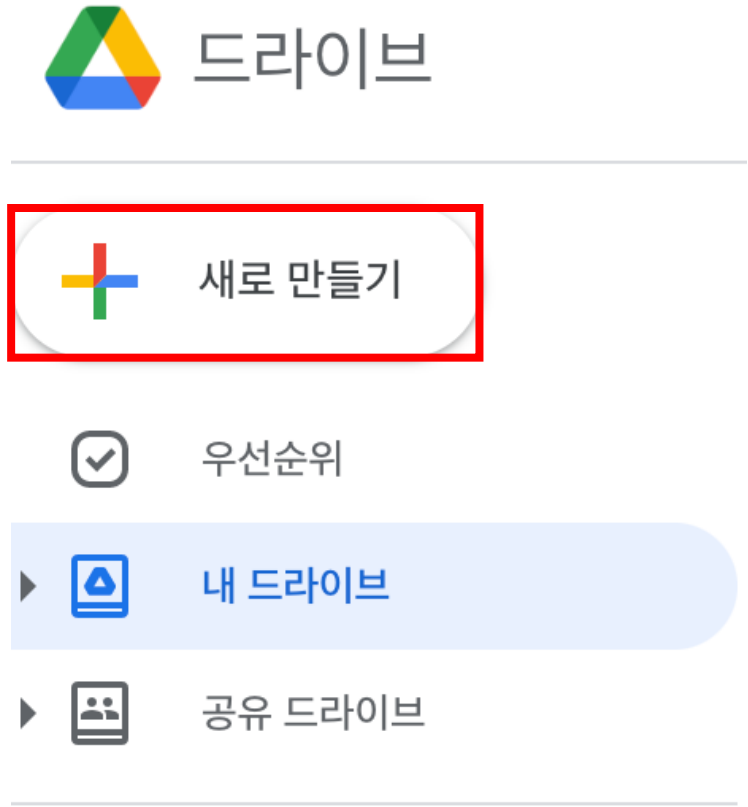
Table of Contents

Colab (Colaboratory) 이란?

- 구글에서 제공하는, python을 작성하고 실행할 수 있는 대화형 환경의 메모장 (jupyter와 유사)
- 장점
  - 많이 쓰이는 머신러닝 관련 python library(pytorch, tensorflow, keras 등)가 이미 설치되어 있음
  - 무료로 GPU 사용 가능
  - 공동 작업 용이

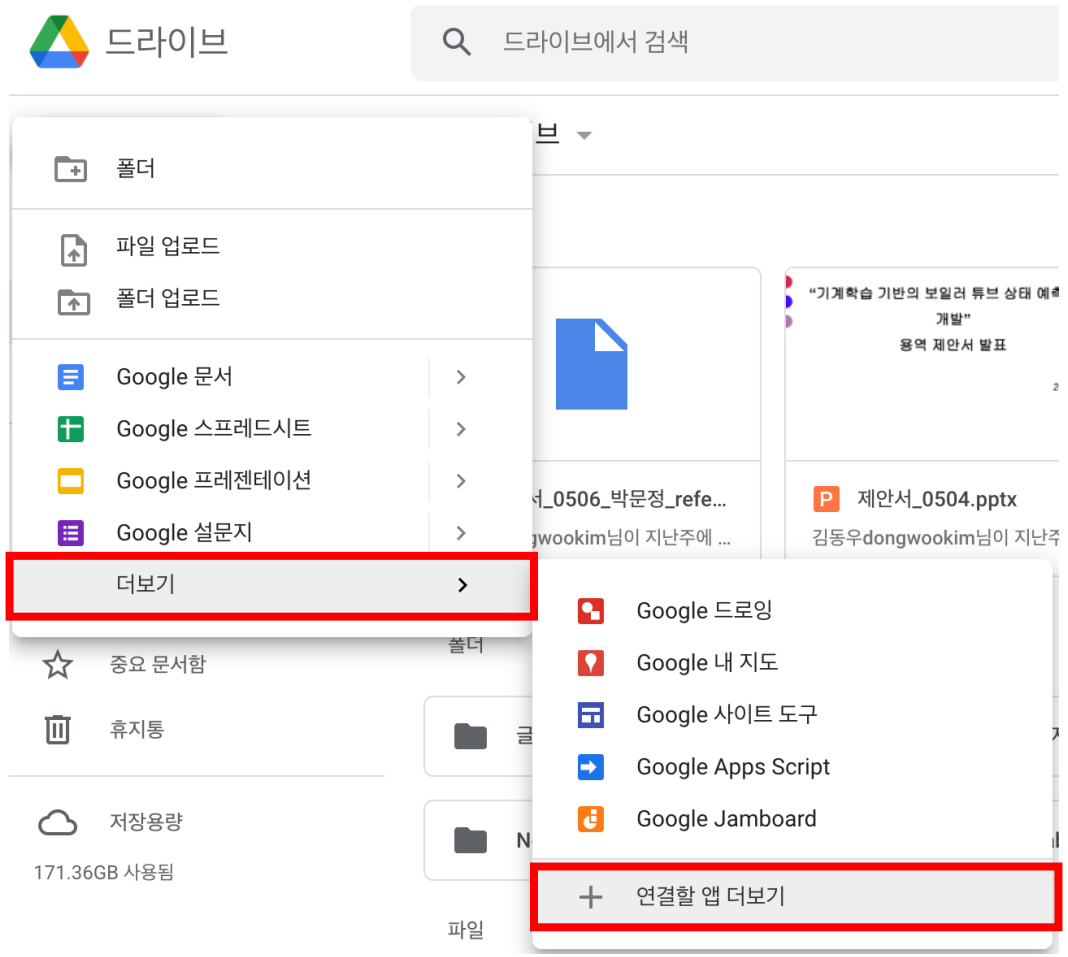
Colab 사용법

Google drive(drive.google.com)에 접속해서 "새로 만들기" 선택



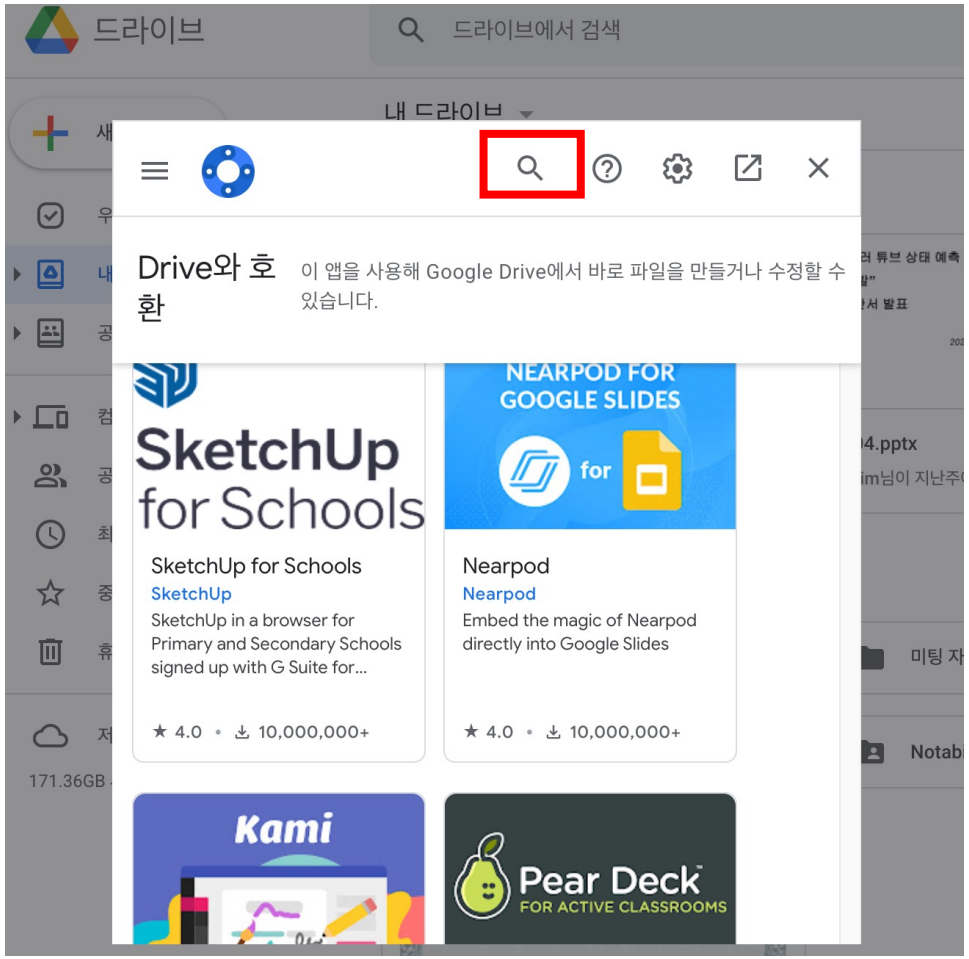
Colab 사용법

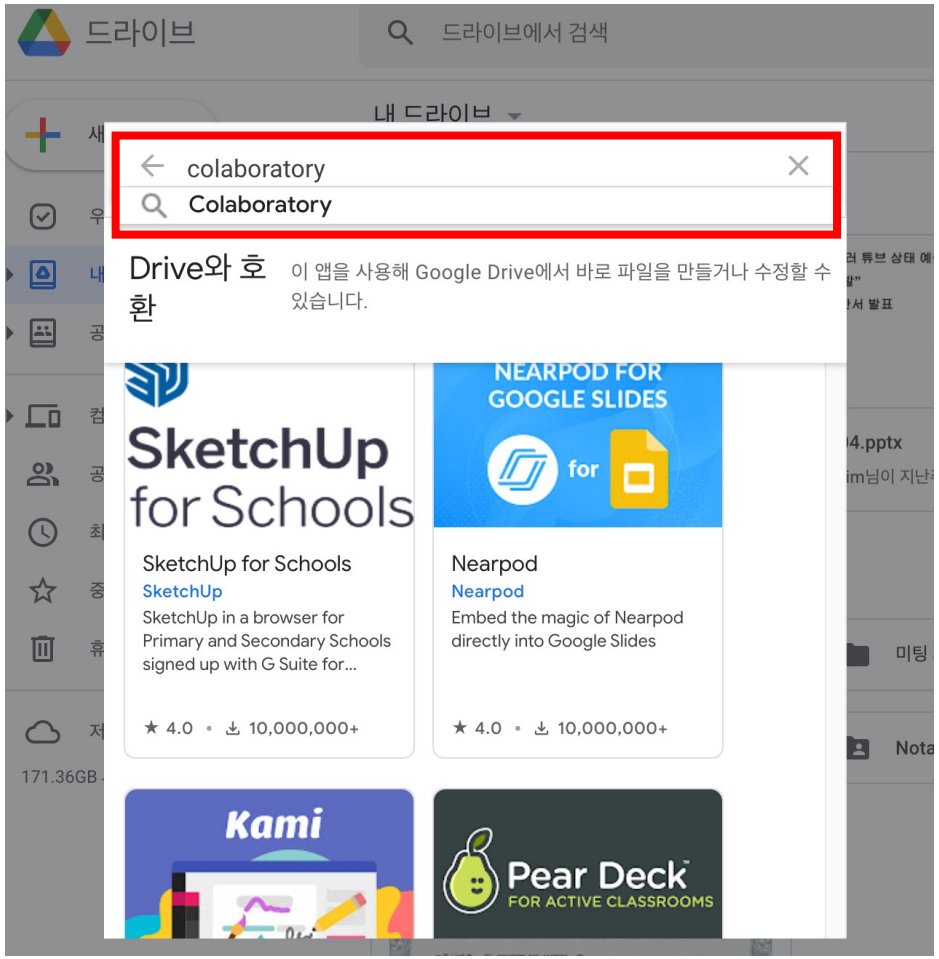
"연결할 앱 더보기" 선택



Colab 사용법

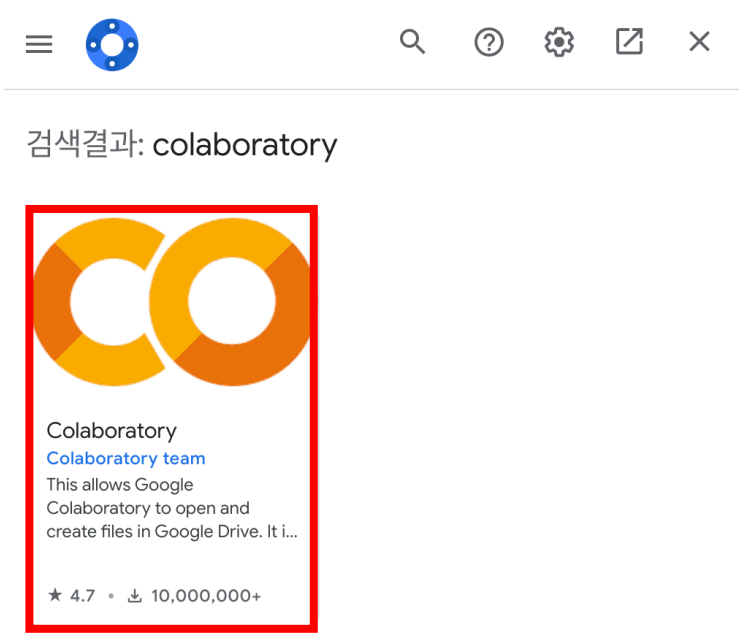
"Colaboratory" 검색

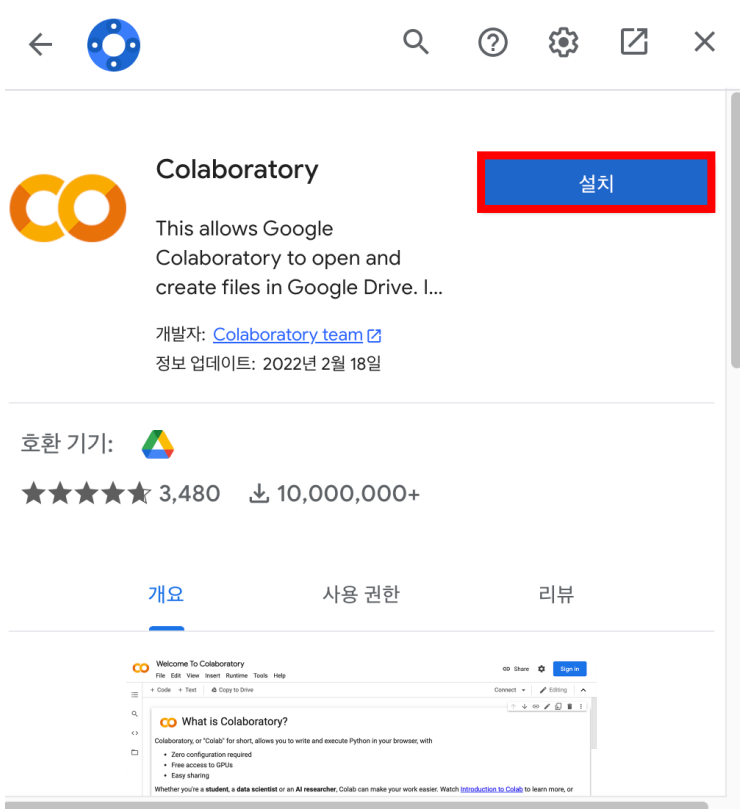


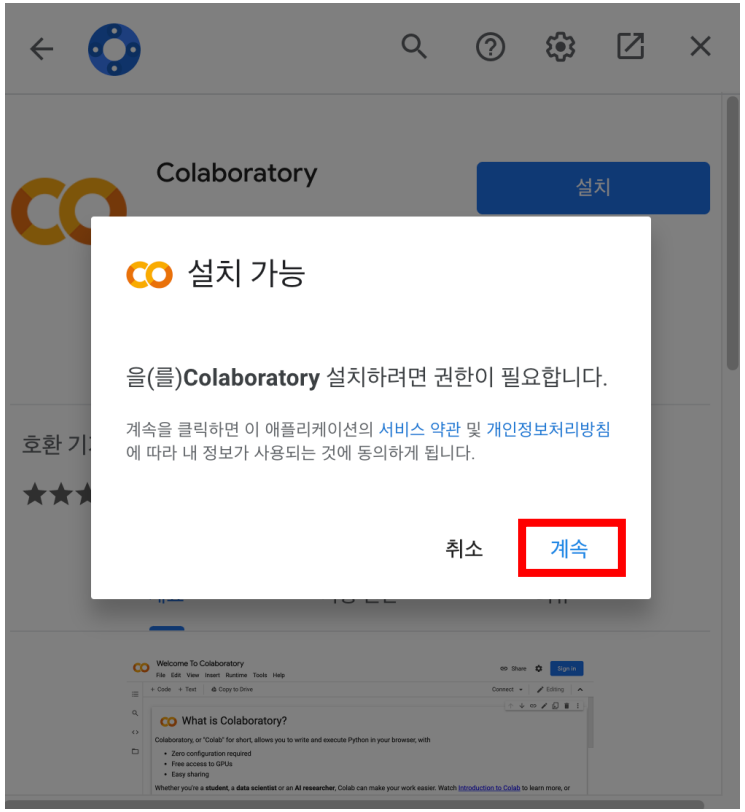


Colab 사용법

Colab 페이지에서 설치 버튼 클릭

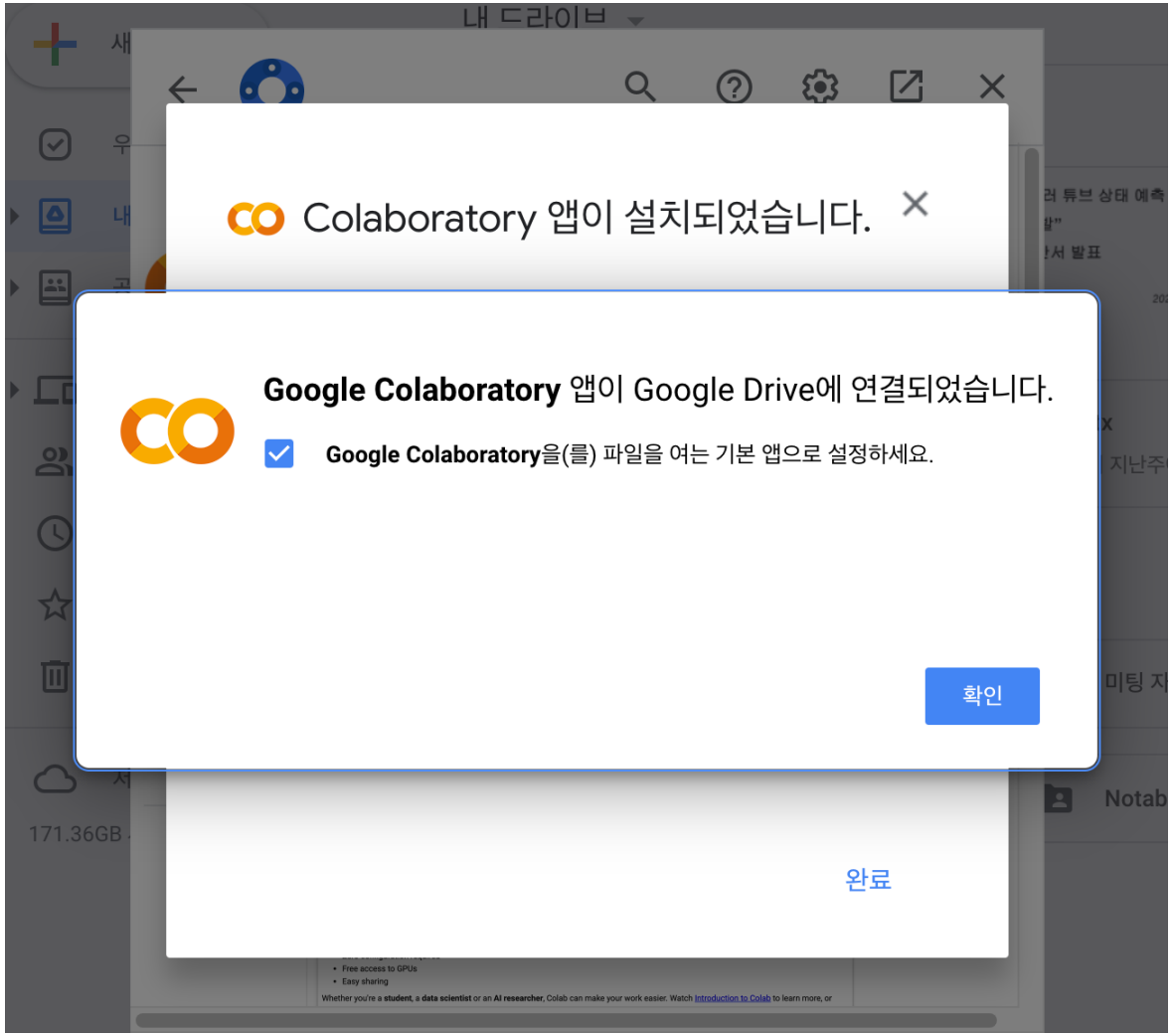






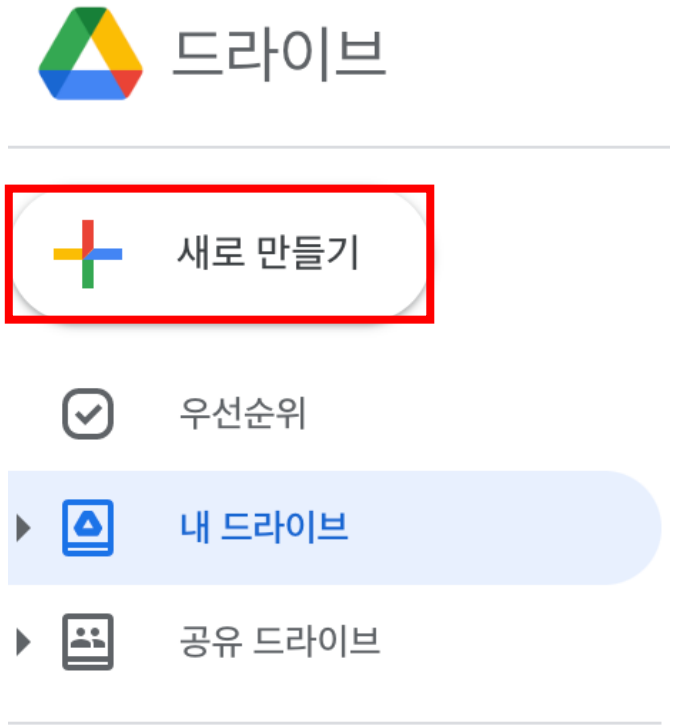
Colab 사용법

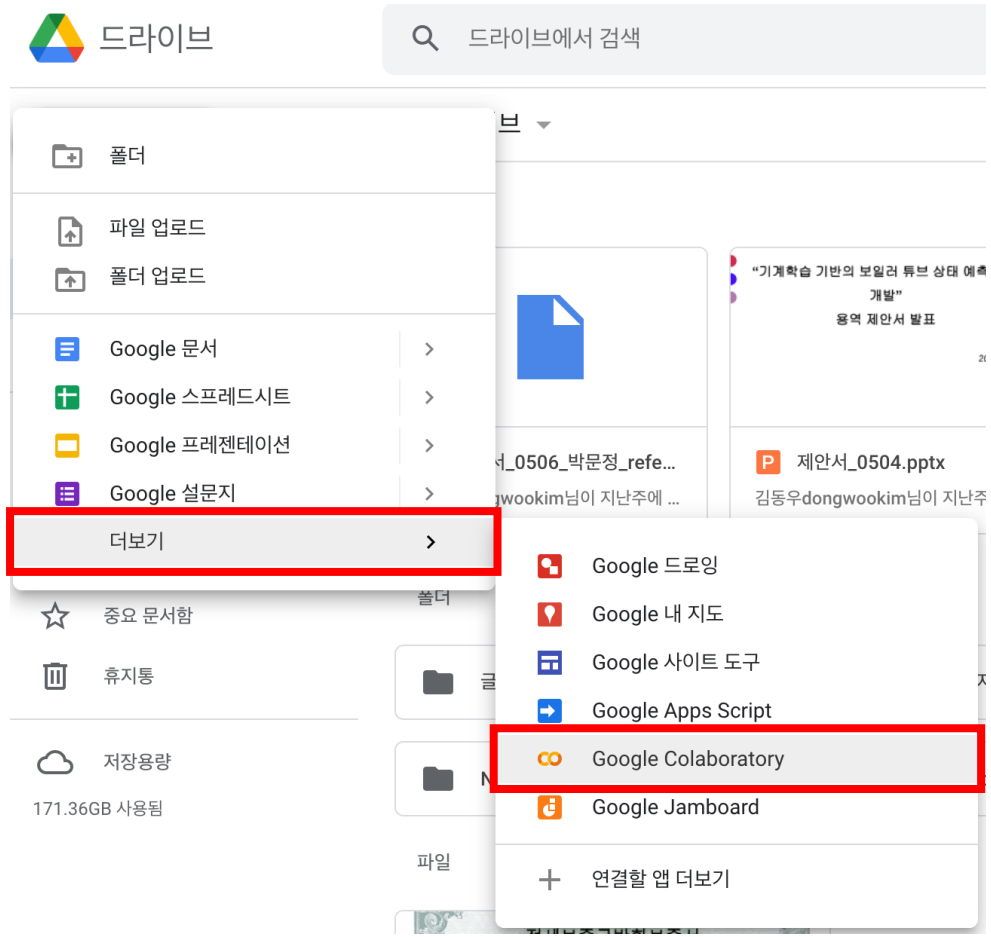
설치 완료!



Colab 사용법

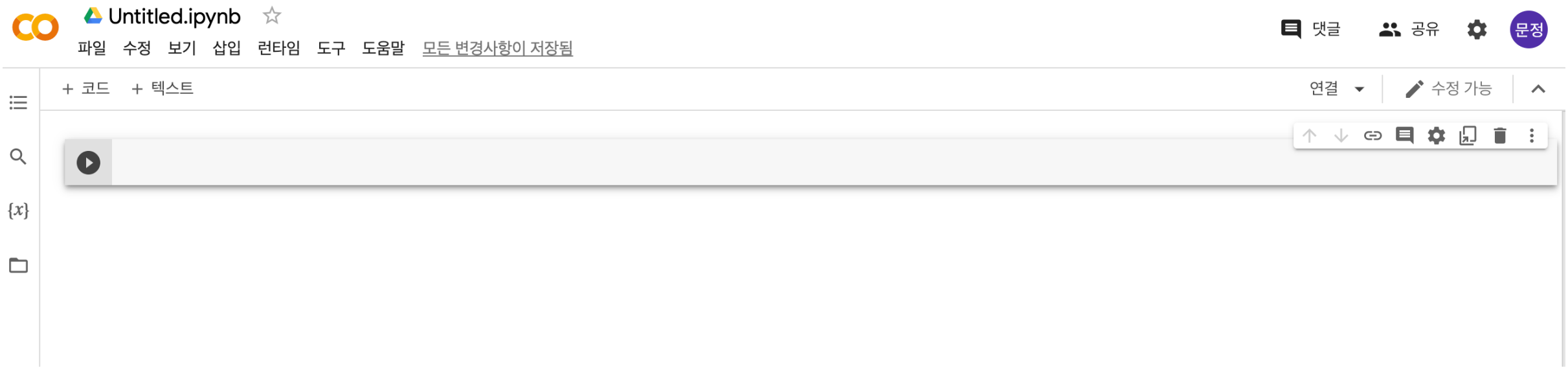
설치 후에는 “새로 만들기” >> “더보기” 선택 후 “Google Colaboratory” 선택





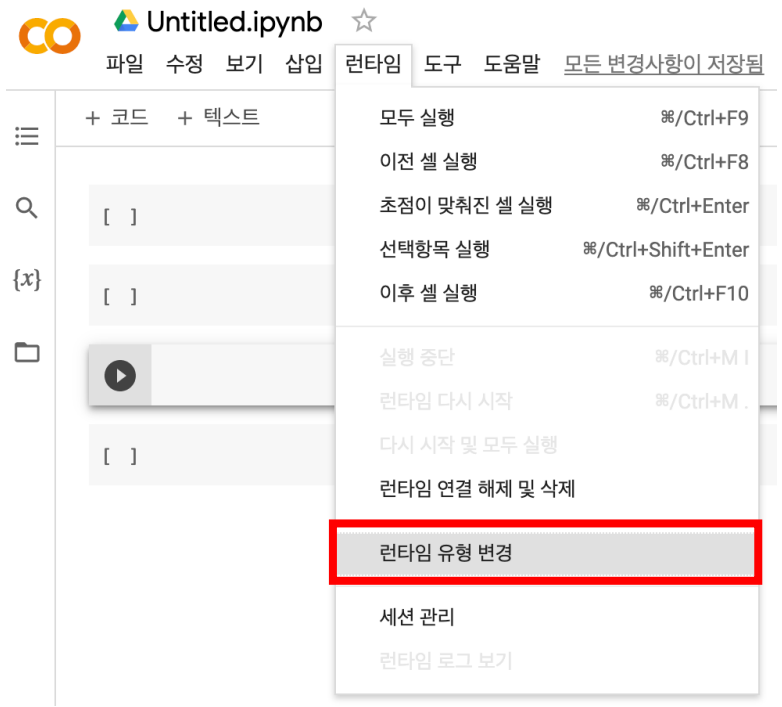
Colab 실행 화면

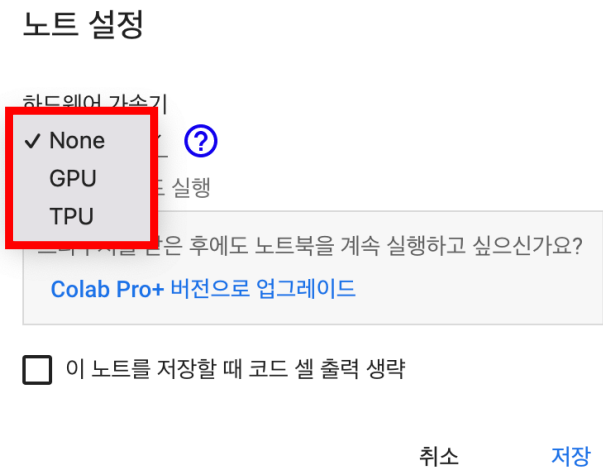
Jupyter notebook과 유사하게 사용 가능



Colab에서 gpu 사용하기

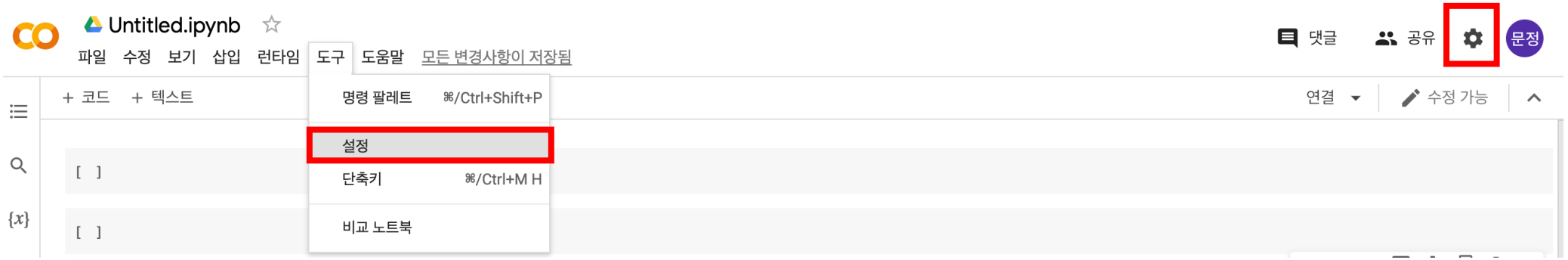
상단 바에서 “런타임” >> “런타임 유형 변경” 선택하면 gpu 사용하도록 설정 변경 가능





Colab 설정

- 상단 바에서 “도구” >> “설정” 혹은 상단바 오른쪽에 톱니 버튼 클릭하면 환경 설정 창 열림
  - 테마 등 변경 가능
  - “기타” 선택하면 재미있는 설정 가능 :D



Colab에서 라이브러리 불러오기

- Tensorflow 불러오기

```
[1]  import tensorflow as tf
```

```
[2]  print(tf.__version__)

     2.8.0
```

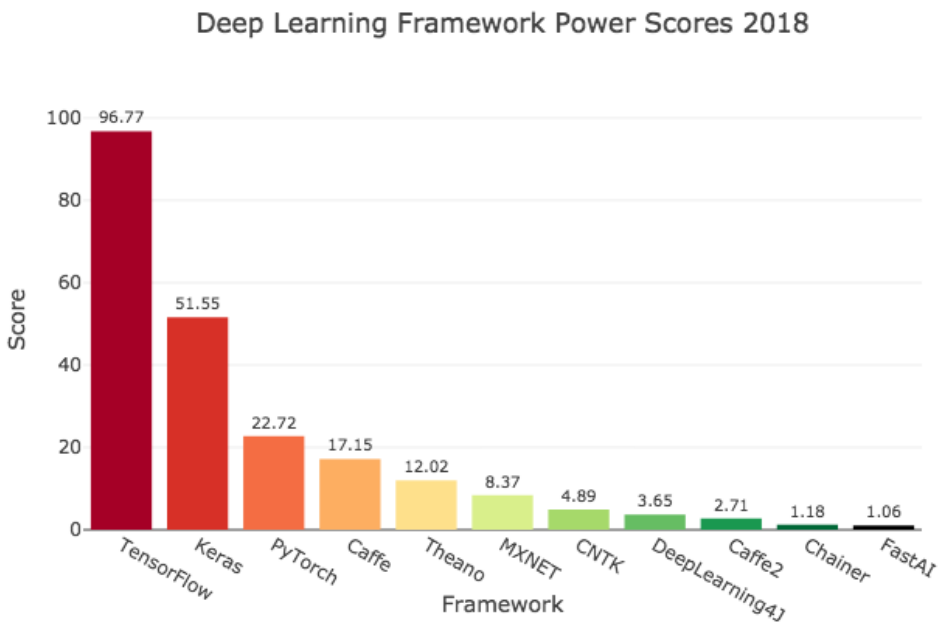
실습 중 Colab 활용하기

- 실습 중 서버에 문제가 생겨 서버 접속이 안되는 경우, colab 활용해 실습 참여
- 실습 중 코드 문제인지 서버 문제인지 정확히 모르는 오류가 생겼을 때 colab에서 똑같은 코드 동작 여부 확인
  - Colab에서도 문제가 발생하면 코드 문제이니 코드 확인
  - Colab에서는 코드가 이상 없이 동작한다면 서버 ip 및 오류 메세지 조교에게 전달
    - -> 이후 실습은 colab으로 진행

Table of Contents

TensorFlow란?

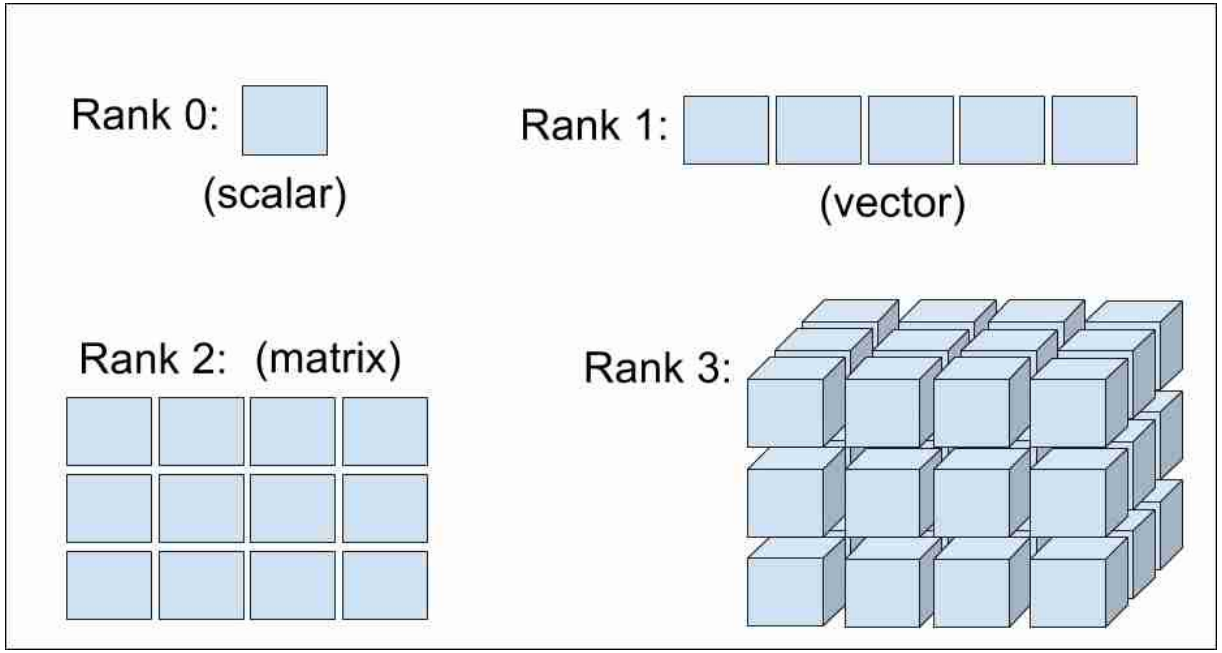
- 세계에서 가장 널리 쓰이는 machine learning framework
- 2015년에 Google Brain에서 오픈소스로 공개
- 공개 당시 버전 0.5
- 2019년 10월 1일에 2.0 버전 릴리즈



이미지 출처 : https://images.app.goo.gl/fgJffqaxEsvEfECK8

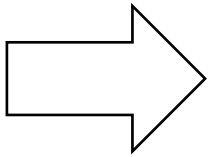
Tensor + Flow

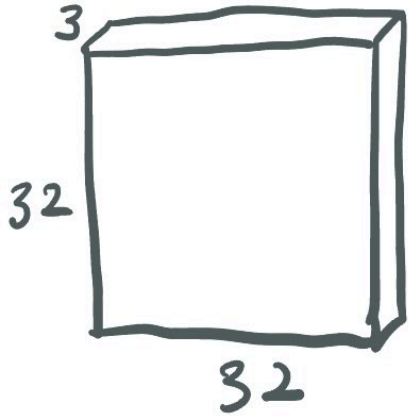
- 벡터와 행렬을 일반화한 것
- 고차원으로 확장 가능
- 다양한 형태의 데이터를 표현하기에 용이



Example of Tensor – 1

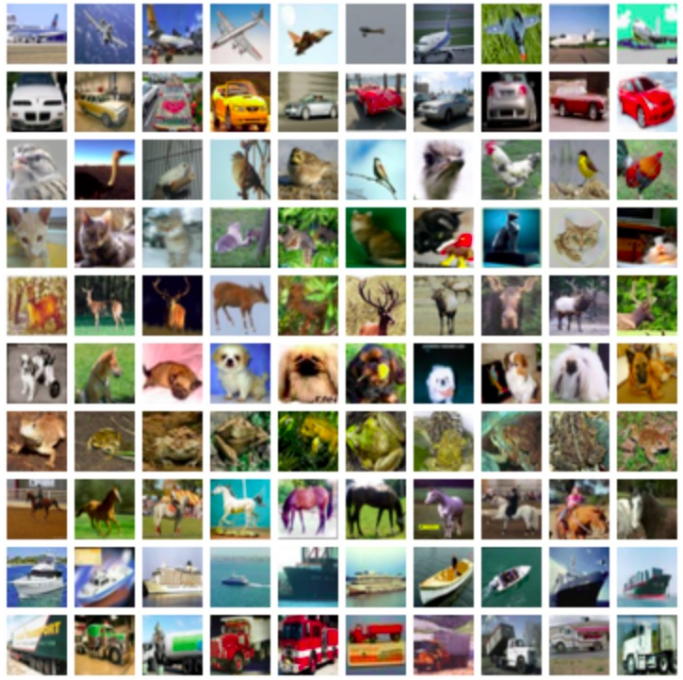
- cifar-10의 이미지 한장
  - 컬러 이미지는 3개의 channel (R, G, B)이 겹쳐져서 한 장의 이미지를 이룸
  - (3, 32, 32)의 크기를 가지는 3차원 텐서

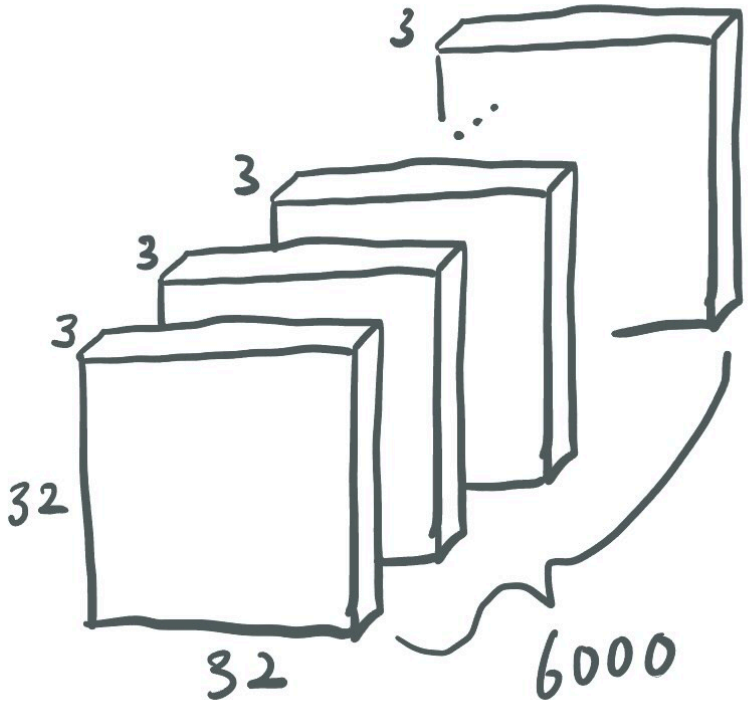




Example of Tensor – 2

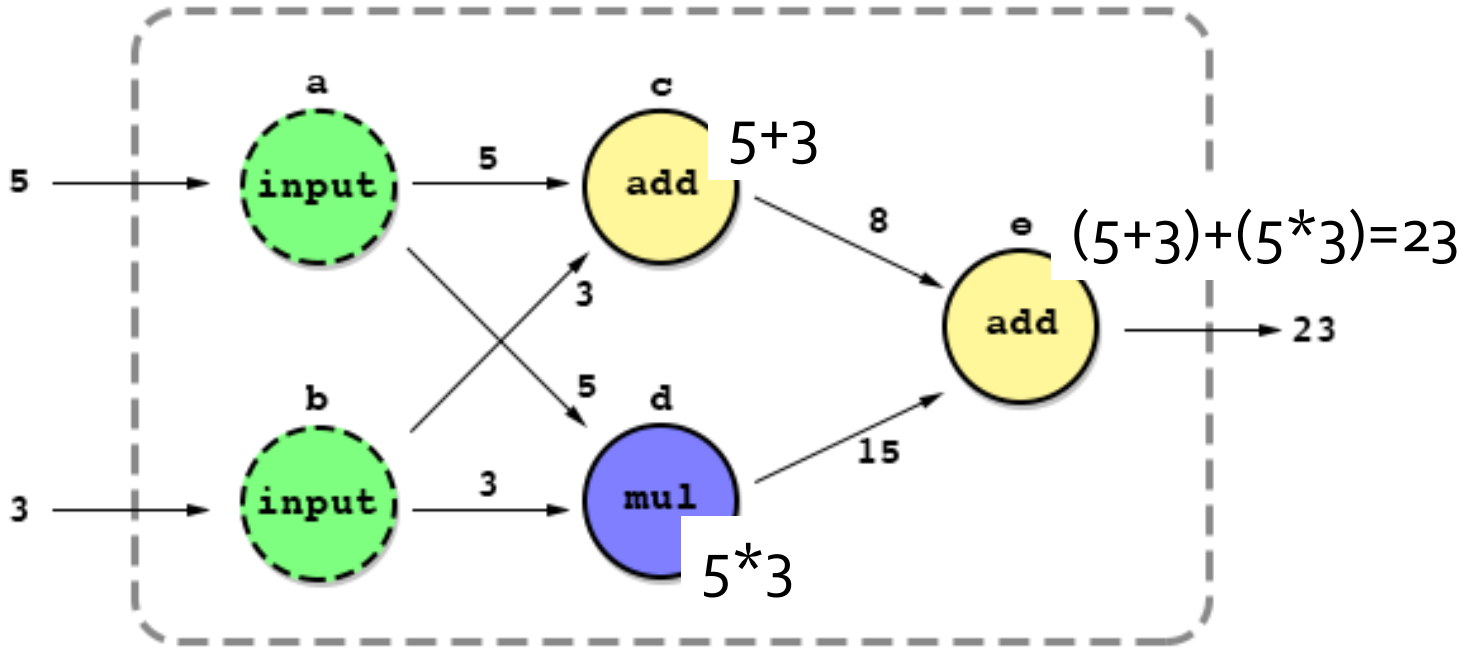
- cifar-10 데이터셋
  - Cifar-10 데이터셋은 6000장의 이미지로 구성되어 있음
  - (6000, 3, 32, 32)의 크기를 갖는 4차원 텐서





Tensor + **Flow**

- Data Flow Graph: node는 연산, edge는 데이터를 의미하며, edge의 데이터가 그래프를 따라 node로 흘러가며 node에 지정된 연산을 하도록 하는



이미지 출처 : https://images.app.goo.gl/tkUQwNKkxFE2f5EY8

Tensor + Flow

- TensorFlow는
  - Data Flow Graph의 방식을 활용해 수학 계산과 데이터 흐름을 정의하고 실행하는 framework이며,
  - edge에 흐르는 데이터 형식이 Tensor임.

Practice

- Tensor 생성하기
  - tf.constant
  - tf.Variable
  - tf.zeros, tf.ones
  - tf.zeros_like, tf.ones_like
  - tf.random.uniform, tf.random.normal

Practice

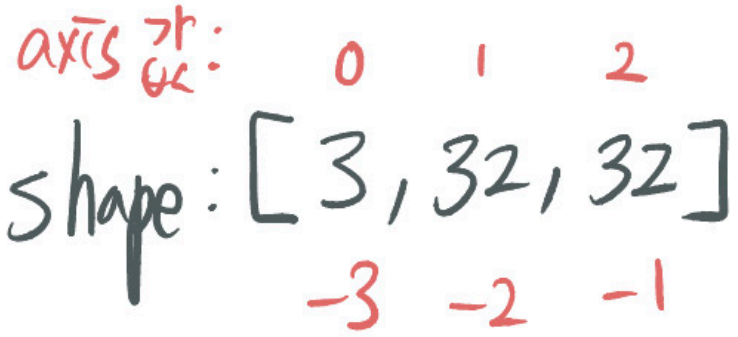
- "dtype" argument
  - tf.constant, tf.zeros, tf.ones 등 다양한 함수가 "dtype" argument를 가짐
  - dtype은 tensor가 가지는 값의 데이터 타입을 의미함
  - dtype 값 예시: tf.int32, tf.float32 등
  - 더 많은 예시는 링크 https://www.tensorflow.org/api_docs/python/tf/dtypes 참고
- 관련 함수: tf.cast
  - Tensor들의 dtype이 맞지 않는 경우 연산이 되지 않음
  - 이 경우 tf.cast 함수를 통해 타입을 변환해 줌

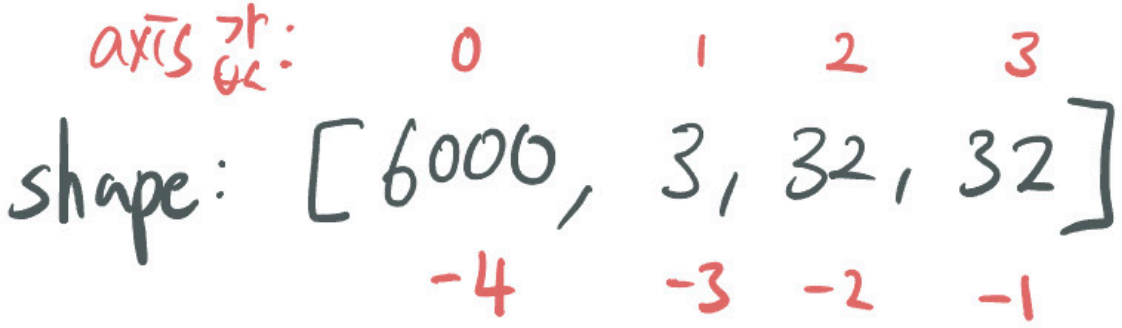
Practice

- 자주 사용하는 연산
  - +, - , *, /, **, …
  - tf.matmul (or @)
  - tf.reduce_mean, tf.reduce_sum, tf.reduce_max, tf.reduce_min
  - tf.argmax, tf.argmin

Practice

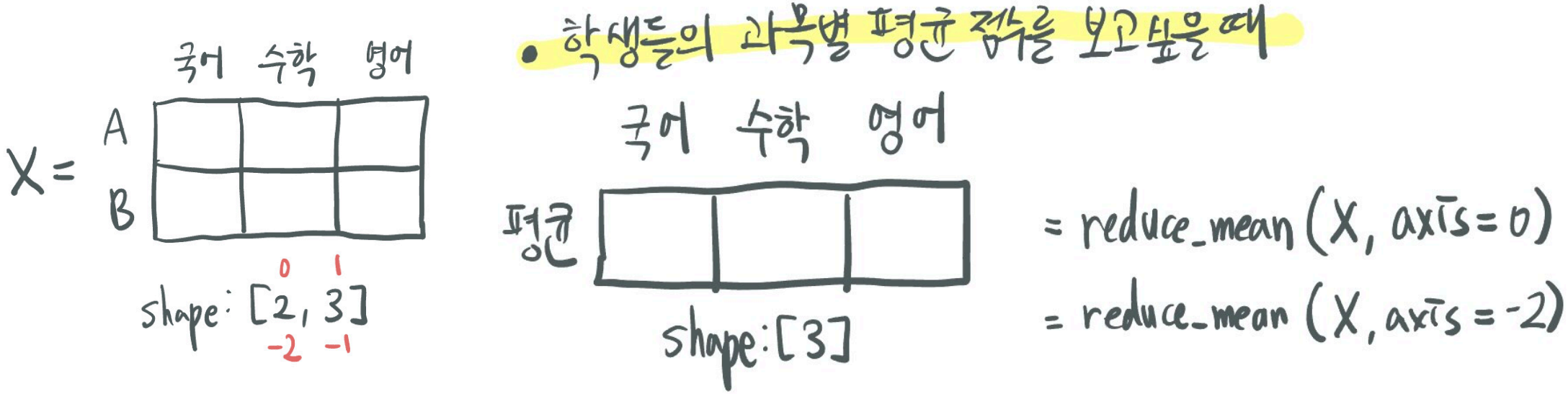
- "axis" argument
  - tf.reduce_mean, tf.reduce_sum, tf.max, tf.min, tf.argmax, tf.argmin 등 다양한 함수가 "axis" argument를 가짐
  - "axis" argument를 이용하면 특정 axis에 대해서만 연산 적용 가능
  - axis 값?





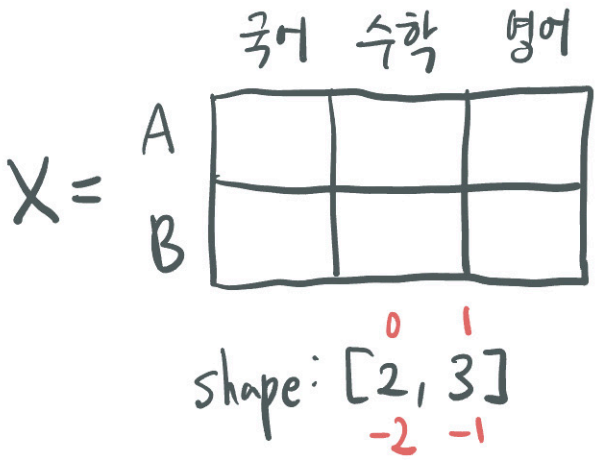
Practice

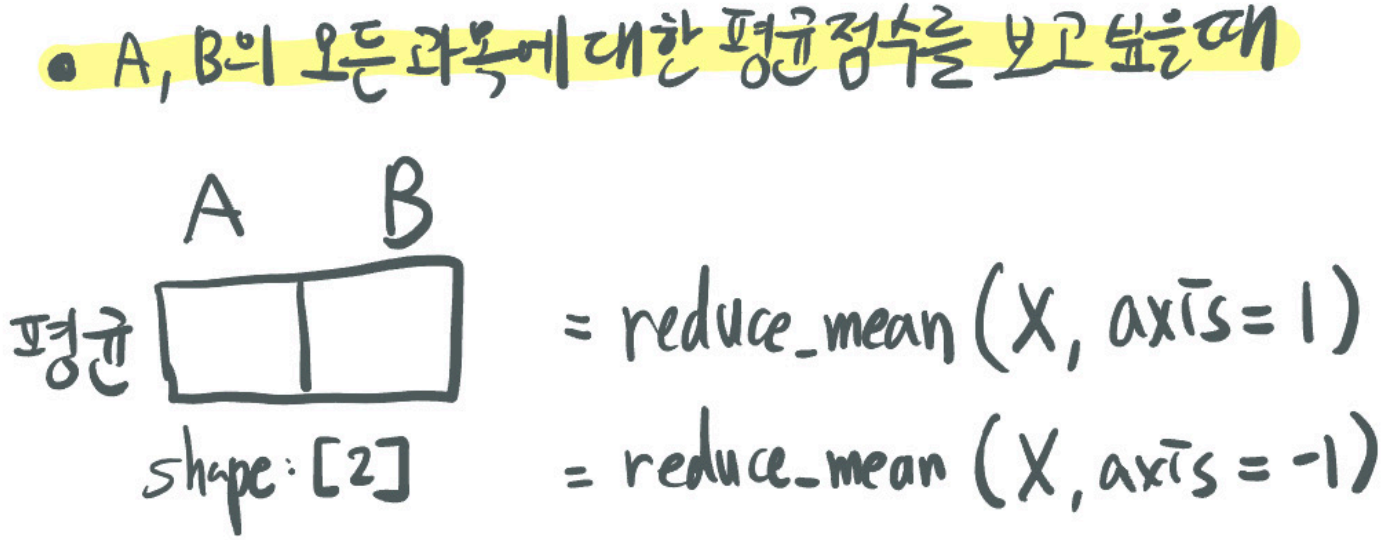
- "axis" argument 사용 tip
  - 함수 적용 후 원하는 shape을 먼저 생각한 후에 차원 값 지정해주면 쉬움
  - 적용 후 차원이 줄어드는 함수의 경우, 지우고 싶은 차원 값을 넣어 줌
  - 예시:



Practice

- "axis" argument 사용 tip
  - 함수 적용 후 원하는 shape을 먼저 생각한 후에 차원 값 지정해주면 쉬움
  - 적용 후 차원이 줄어드는 함수의 경우, 지우고 싶은 차원 값을 넣어 줌
  - 예시:





Practice

- GPU 사용 여부 확인
  - tf.debugging.set_log_device_placement(True)

- Tip
  - 위에서 확인할 수 있듯 TensorFlow 함수는 기본적으로 GPU를 사용해서 연산을 수행함
  - GPU는 "highly parallel"한 구조를 가지고 있어, 간단한 연산 처리에 있어 CPU보다 훨씬 빠른 연산 속도를 보임
  - 따라서, 효율적인 코드를 짜기 위해서는 for문 사용을 최소화하고 TensorFlow 함수 활용을 최대화하는 것이 좋음

Practice

- Tensor의 shape 확인 및 변형
  - tf.shape (tf.Tensor.shape)
  - tf.reshape
  - tf.squeeze, tf.expand_dims

Practice

- "axis" argument 사용 tip
  - 함수 적용 후 원하는 shape을 먼저 생각한 후에 차원 값 지정해주면 쉬움
  - tf.expand_dims의 경우, 적용 후 추가하고 싶은 차원 값을 넣어 줌
  - 예시:
    - shape [3,4] 에서 shape [1, 3, 4]를 만들고 싶은 경우: axis = 0
    - shape [3,4] 에서 shape [3, 1, 4]를 만들고 싶은 경우: axis = 1
    - shape [3,4] 에서 shape [3, 4, 1]를 만들고 싶은 경우: axis = 2

Quiz-1

- 아래와 같은 값을 가지는 shape=[3,2]인 tensor를 tf.constant 사용하여 생성하기 (필요시 tf.reshape 사용)

0.3	0.4
0.3	0.35
0.4	0.25

- 0~100 사이의 랜덤한 정수 값을 갖는 shape=[4,3]인 tensor를 tf.random.uniform 사용하여 생성하기

Quiz-2

- Quiz-1에서 생성한 tensor가 각각 학교 A, B의 수능/내신/면접 점수 반영 비율과 학생 1~4의 수능/내신/면접 점수를 나타낸다고 했을 때,

	A	B
수능	0.3	0.4
내신	0.3	0.35
면접	0.4	0.25

	수능	내신	면접
1			
2			
3			
4			

  - 각 학교 기준으로 환산 시 각 학생 별 점수 tf.cast, tf.matmul (or @) 사용하여 구하기
  - 각 학생 별 환산 점수에 대한 평균값을 tf.reduce_mean 사용하여 구하기
  - 각 학교 별 최고점을 받는 학생 tf.argmax 사용하여 구하기

Quiz-3

- Quiz-2에 이어, 추가적으로 학교 C의 수능/내신/면접 점수 반영 비율이 각각 0.2, 0.4, 0.4임을 알게 되었다고 했을 때
  - 학교 C의 점수 반영 비율을 나타내는 shape=[3,]인 tensor를 tf.constant 사용하여 생성하고,
  - tf.expand_dims, tf.matmul (or @) 사용하여 학교 C 기준으로 환산했을 때 각 학생이 받는 점수 구하기

Thank you :)

- *saemi@postech.ac.kr*