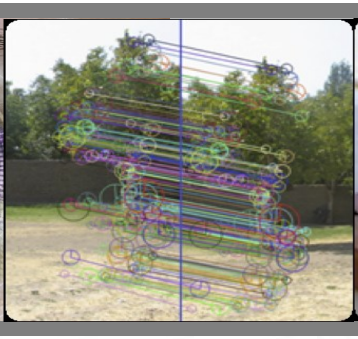
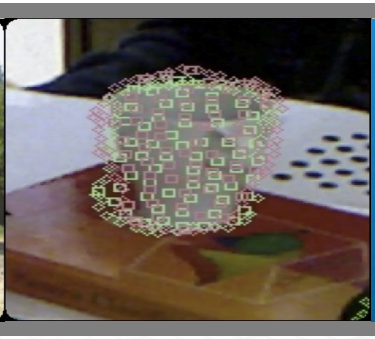
Computer Vision

Deep Learning Basics
# (#18: Google Colab-based deep learning environment setup)



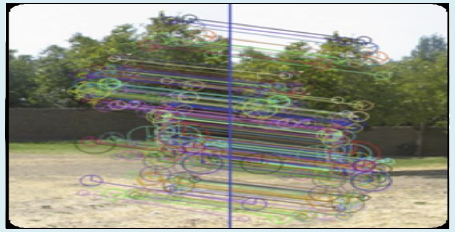


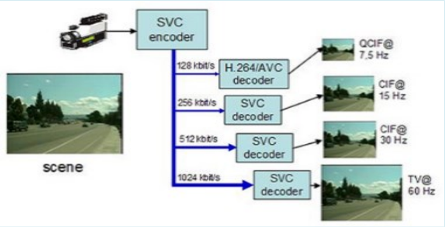
2019. Autumn

Prof. Byung-Gyu Kim
Intelligent Vision Processing Lab. (IVPL)
http://ivpl.sookmyung.ac.kr
Dept. of IT Engineering, Sookmyung Women's University
E-mail: bg.kim@ivpl.sookmyung.ac.kr

Gaol of this lecture

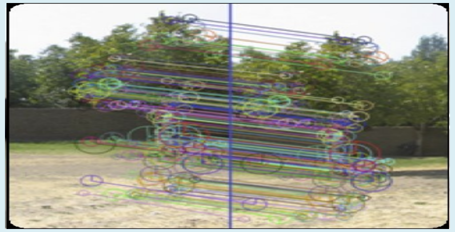
- How to set-up the deep learning development using **Google Colab**?
  - What is **Google Colab**?
  - How to use it and develop the deep learning system?
  - Basic configuration of **Google Colab**

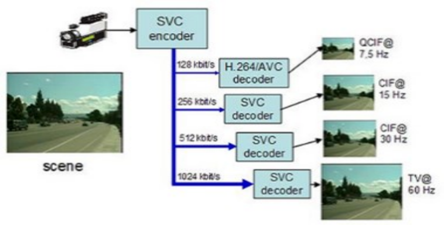




Contents

- What is the **Google Colab**?
- Basic configuration of **Google Colab**





Contents

- What is the **Google Colab**?
- Basic configuration of **Google Colab**

Google Colab: What is it? (1)

- Deep learning requirements
  - **Big data (images)**
  - **Huge time to compute and train the CNN**
  - **Parallel processing → GPU is necessary because of time**

GPU system is always needed to compute the desired algorithms efficiently.

Google Colab: What is it? (2)

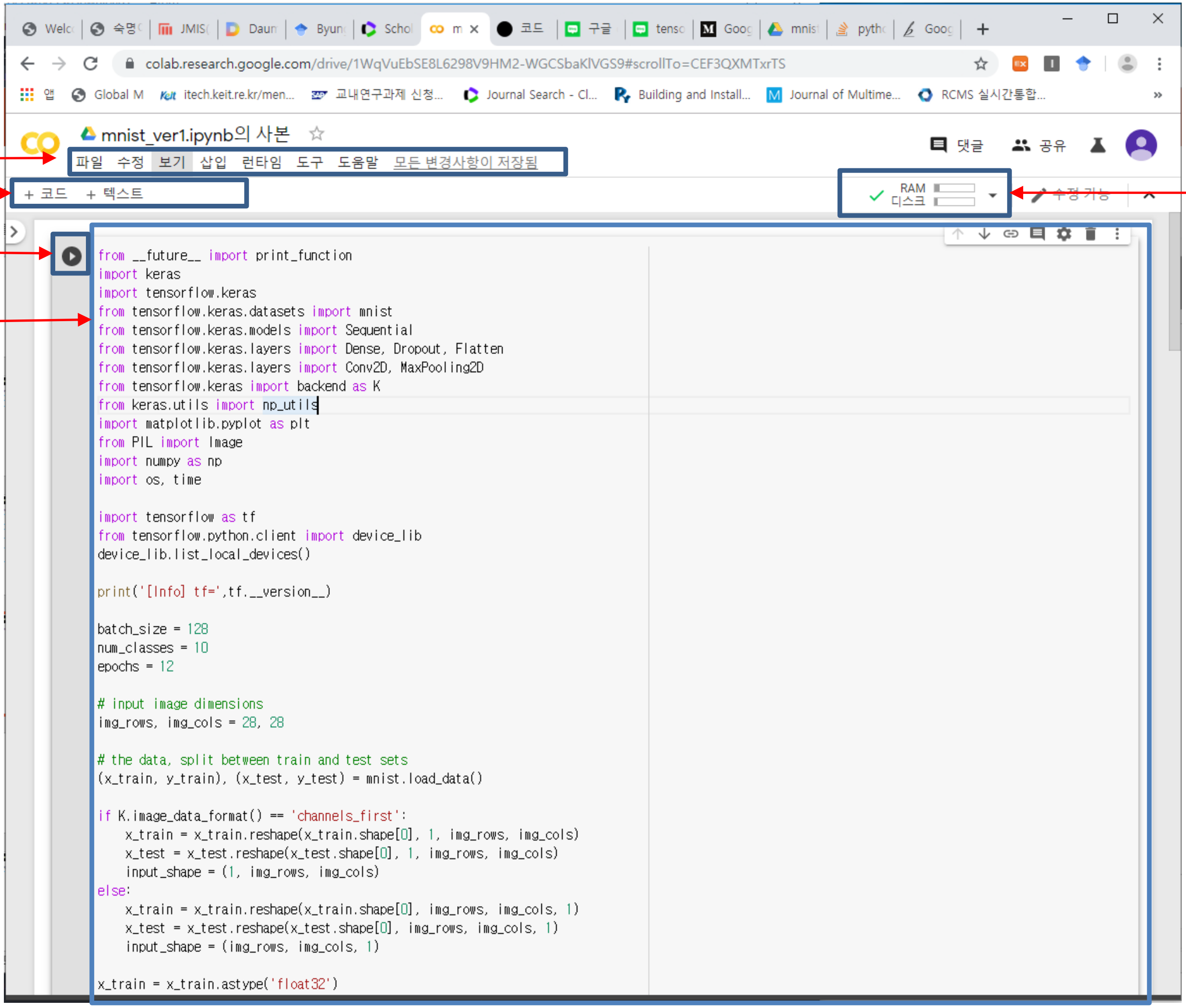
❖ Google Colaboratory

- Google 내부에서 사용하던 jupyter Notebook을 교육과 연구 목적으로 customize한 데이터 분석 도구
- 특히 machine learning 교육 및 연구용 도구로 open된 클라우드 기반의 서비스
- 이미 Python2.x와 3.x 버전이 설치되어 있고 GPU 클라우드 기반의 GPU 병렬 처리를 제공하여 Google 계정만 있으면 기본 GPU 서비스 기반 병렬 처리를 지원함
- 웹브라우저 기반으로 docker 환경에서 google GPU 서버에 접속하여 서비스가 지원되므로 기본적으로 Google chrom 브라우저를 권장함
- Colaboratory 실행 코드는 google 계정 전용의 가상 머신에서 동작하므로 세션이 끊으지거나 유휴 상태가 오래 지속되면 리소스가 자동 재할당 되므로 본인의 데이터가 메모리에서 제거됨
- 따라서 Google Drive (내 드라이브)를 연결하여 주로 본인의 개발 소스를 관리하는 체계를 지원함

Google Colaboratory = Google Drive + Juputer Notebook

Google Colab: What is it? (3)

❖ Google Colaboratory: **Basic UI**


colab.research.google.com/drive/1WqVuEbSE8L6298V9HM2-WGCSbaKlVGS9#scrollTo=CEF3QXMTxrTS
mnist_ver1.ipynb의 사본
파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨
+ 코드 + 텍스트
댓글
공유
RAM
디스크
수정 가능

메뉴

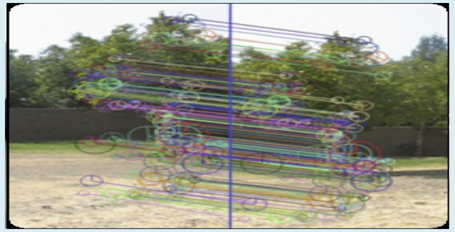
셀 메뉴

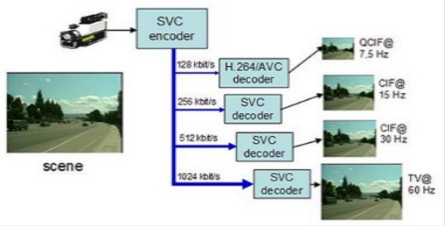
셀 실행버튼

셀 필드

리소스 연결상태

```python
from __future__ import print_function
import keras
import tensorflow.keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras import backend as K
from keras.utils import np_utils
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
import os, time

import tensorflow as tf
from tensorflow.python.client import device_lib
device_lib.list_local_devices()

print('[Info] tf=',tf.__version__)

batch_size = 128
num_classes = 10
epochs = 12

# input image dimensions
img_rows, img_cols = 28, 28

# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)

x_train = x_train.astype('float32')
```





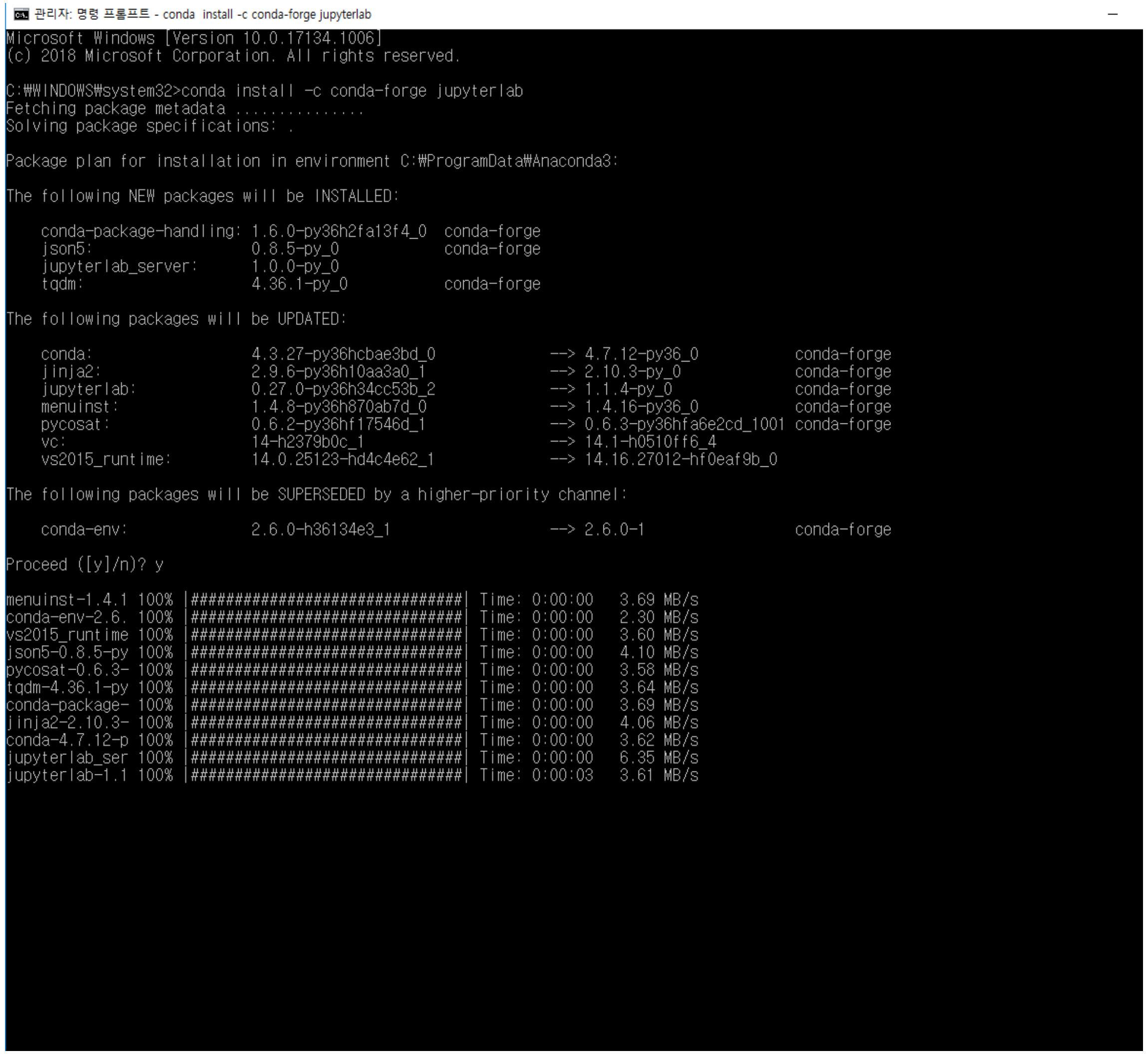
Contents

- What is the **Google Colab**?
- Basic configuration of **Google Colab**

Google Colab 활용하기: 기본 설정 및 동작성 확인

❖ Jupyter 노트 또는 Jupyterab 설치 (anaconda 설치된 상태)

- 1) conda install -c conda-forge jupyterlab (CMD 창에서)


관리자: 명령 프롬프트 - conda install -c conda-forge jupyterlab

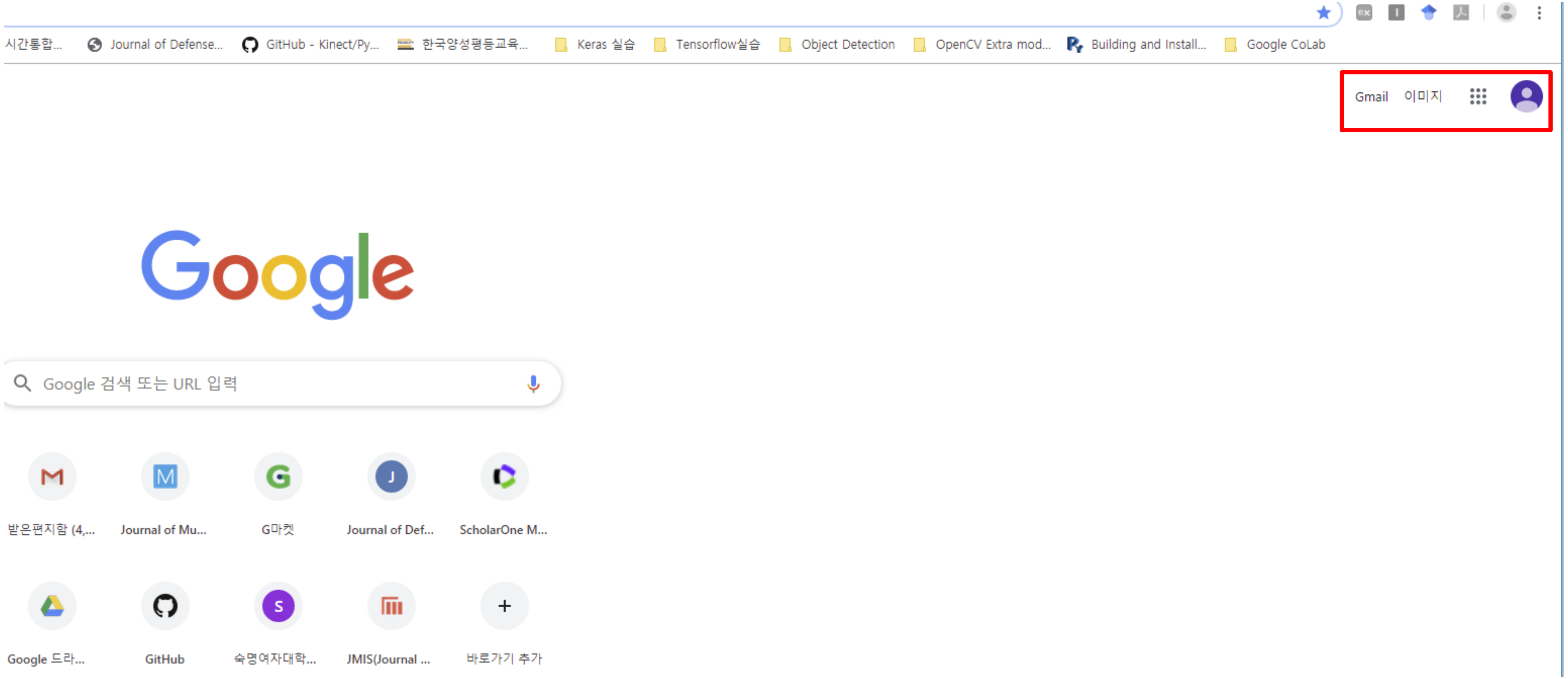
```
Microsoft Windows [Version 10.0.17134.1006]
(c) 2018 Microsoft Corporation. All rights reserved.

C:₩WINDOWS₩system32>conda install -c conda-forge jupyterlab
Fetching package metadata ...............
Solving package specifications: .

Package plan for installation in environment C:₩ProgramData₩Anaconda3:

The following NEW packages will be INSTALLED:

    conda-package-handling: 1.6.0-py36h2fa13f4_0  conda-forge
    json5:                  0.8.5-py_0            conda-forge
    jupyterlab_server:      1.0.0-py_0
    tqdm:                   4.36.1-py_0           conda-forge

The following packages will be UPDATED:

    conda:                  4.3.27-py36hcbae3bd_0            --> 4.7.12-py36_0          conda-forge
    jinja2:                 2.9.6-py36h10aa3a0_1             --> 2.10.3-py_0            conda-forge
    jupyterlab:             0.27.0-py36h34cc53b_2            --> 1.1.4-py_0             conda-forge
    menuinst:               1.4.8-py36h870ab7d_0             --> 1.4.16-py36_0          conda-forge
    pycosat:                0.6.2-py36hf17546d_1             --> 0.6.3-py36hfa6e2cd_1001 conda-forge
    vc:                     14-h2379b0c_1                    --> 14.1-h0510ff6_4
    vs2015_runtime:         14.0.25123-hd4c4e62_1            --> 14.16.27012-hf0eaf9b_0

The following packages will be SUPERSEDED by a higher-priority channel:

    conda-env:              2.6.0-h36134e3_1                 --> 2.6.0-1                conda-forge

Proceed ([y]/n)? y

menuinst-1.4.1 100% |###############################| Time: 0:00:00   3.69 MB/s
conda-env-2.6. 100% |###############################| Time: 0:00:00   2.30 MB/s
vs2015_runtime 100% |###############################| Time: 0:00:00   3.60 MB/s
json5-0.8.5-py 100% |###############################| Time: 0:00:00   4.10 MB/s
pycosat-0.6.3- 100% |###############################| Time: 0:00:00   3.58 MB/s
tqdm-4.36.1-py 100% |###############################| Time: 0:00:00   3.64 MB/s
conda-package- 100% |###############################| Time: 0:00:00   3.69 MB/s
jinja2-2.10.3- 100% |###############################| Time: 0:00:00   4.06 MB/s
conda-4.7.12-p 100% |###############################| Time: 0:00:00   3.62 MB/s
jupyterlab_ser 100% |###############################| Time: 0:00:00   6.35 MB/s
jupyterlab-1.1 100% |###############################| Time: 0:00:03   3.61 MB/s
```

Google Colab 활용하기 : 기본 설정 및 동작성 확인

- 2) Anaconda (최신 버전 설치): jupyter notebook 자동으로 설치됨

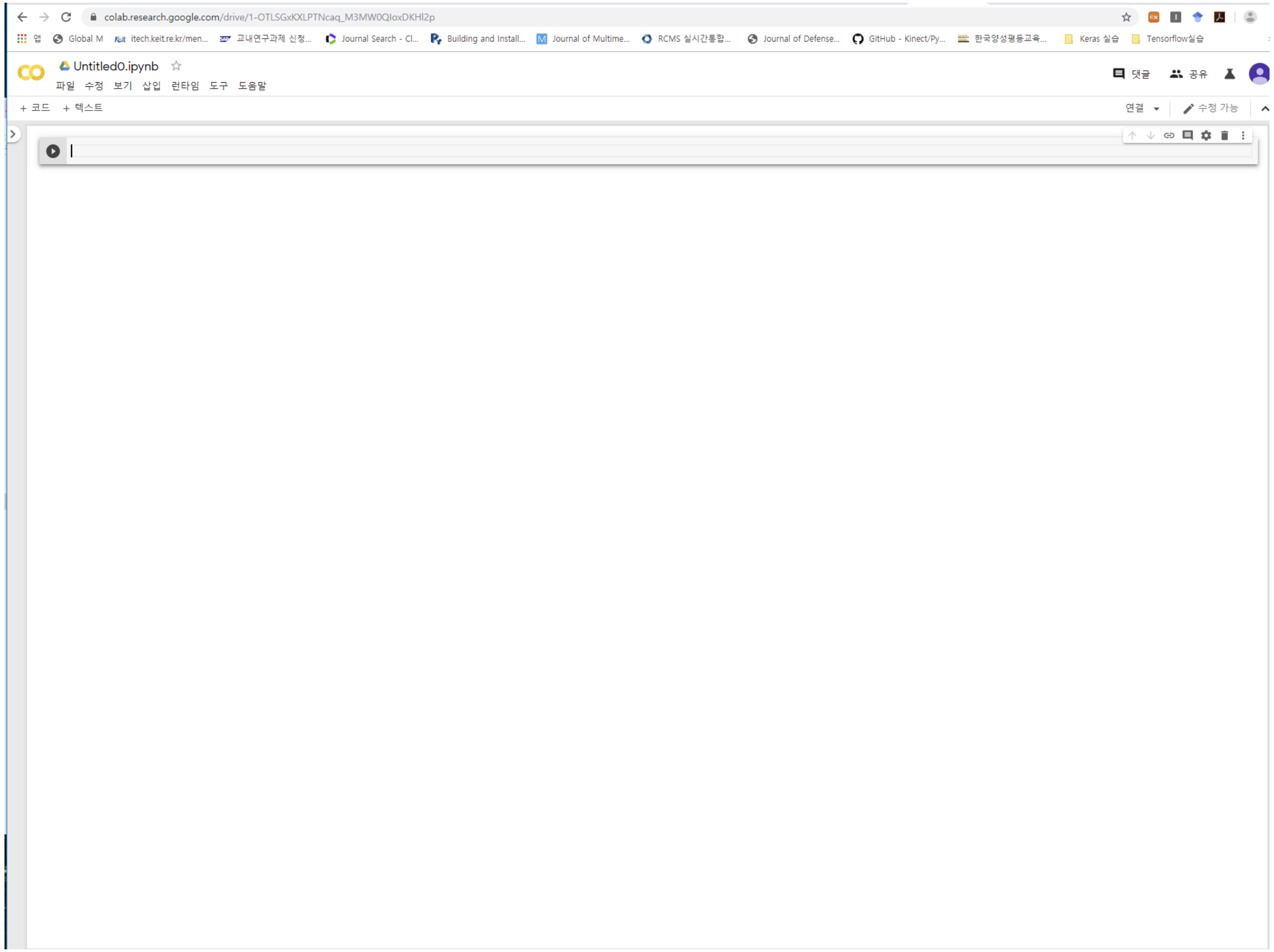
Google Colab 활용하기 : 기본 설정 및 동작성 확인

- Google에 로그인 하기 (상태 확인)



Google Colab 활용하기 : 기본 설정 및 동작성 확인

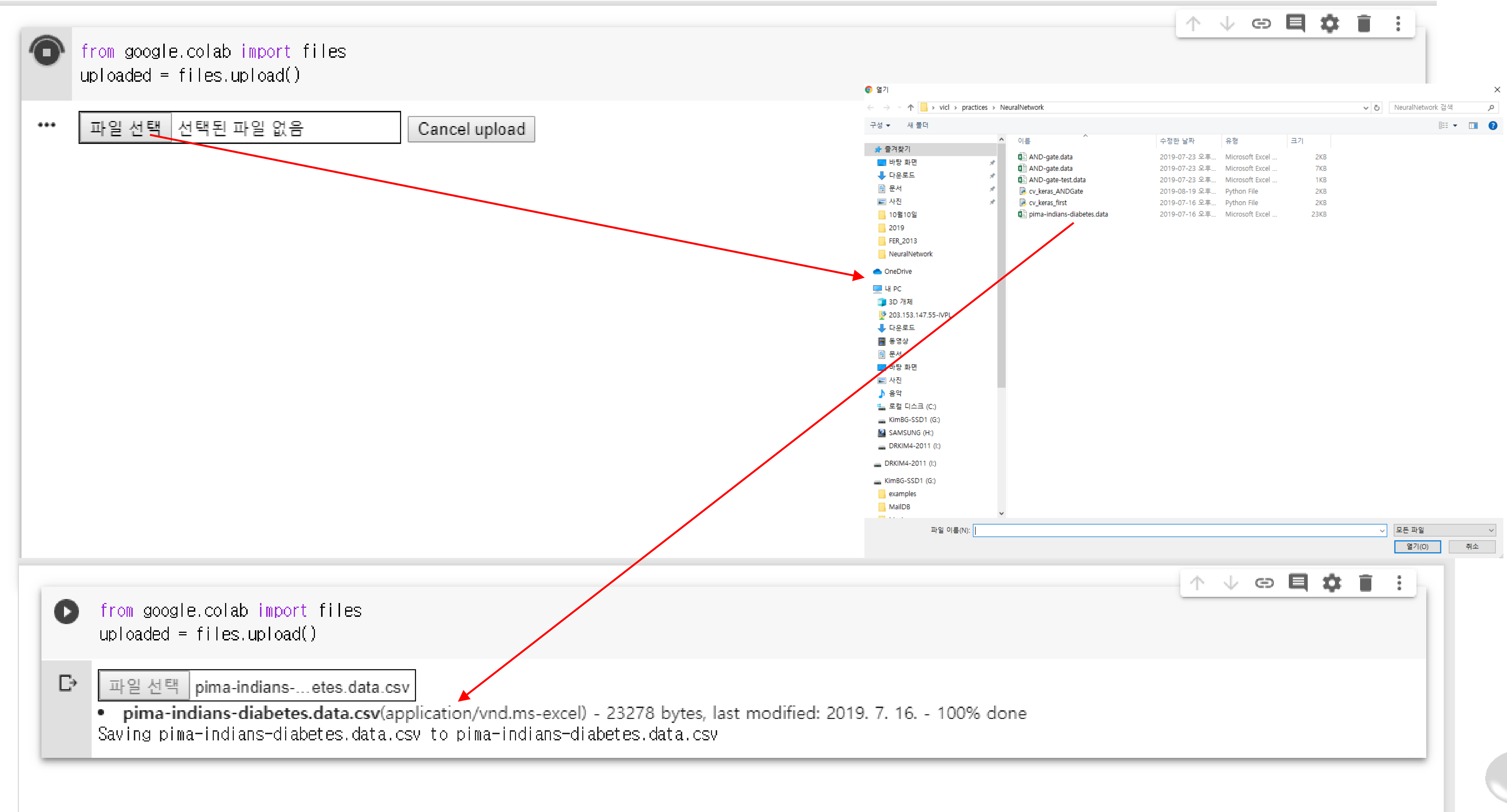
❖ https://colab.research.google.com/ (실제 colab 을 시작하기)



Google Colab 활용하기 : 기본 설정 및 동작성 확인

❖ Jupyter book에서 간단한 예제 실행 (python 기반)

- 오류 발생은 “pop-up” 창 disable되어 있는 것 “허용”으로 해 주면 아래와 같이 동작함



```
from google.colab import files
uploaded = files.upload()
```

파일 선택 | 선택된 파일 없음 | Cancel upload

열기

vicl › practices › NeuralNetwork

이름	수정한 날짜	유형	크기
AND-gate.data	2019-07-23 오후...	Microsoft Excel ...	2KB
AND-gate.data	2019-07-23 오후...	Microsoft Excel ...	7KB
AND-gate-test.data	2019-07-23 오후...	Microsoft Excel ...	1KB
cv_keras_ANDGate	2019-08-19 오후...	Python File	2KB
cv_keras_first	2019-07-16 오후...	Python File	2KB
pima-indians-diabetes.data	2019-07-16 오후...	Microsoft Excel ...	23KB

```
from google.colab import files
uploaded = files.upload()
```

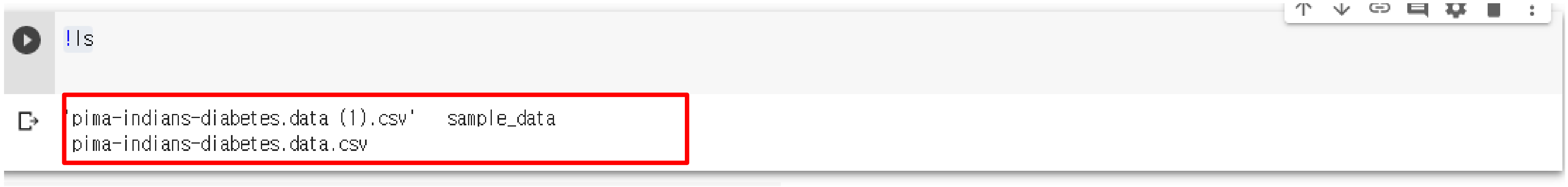
파일 선택 | pima-indians-...etes.data.csv

```
• pima-indians-diabetes.data.csv(application/vnd.ms-excel) - 23278 bytes, last modified: 2019. 7. 16. - 100% done
Saving pima-indians-diabetes.data.csv to pima-indians-diabetes.data.csv
```

IVPL
Intelligent Vision Processing Lab

Google Colab 활용하기 : 기본 설정 및 동작성 확인

- Upload된 데이터 로딩하여 실행확인해 보기
  - 명령어 필드에서 “!ls” 명령어 실행: 업로드한 파일 보이죠???



```
!ls
```

```
'pima-indians-diabetes.data (1).csv'   sample_data
pima-indians-diabetes.data.csv
```

- 실제 저장된 데이터 python 프로그램으로 출력해 보기

```
import numpy as np
dataset = np.loadtxt("pima-indians-diabetes.data.csv", delimiter=",")
print(dataset)
```

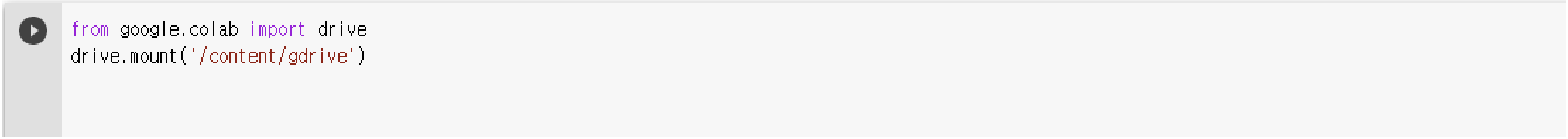
Google Colab 활용하기 : GoogleDrive 연결하기

❖ 프로젝트 코드 및 데이터 저장소 만들기

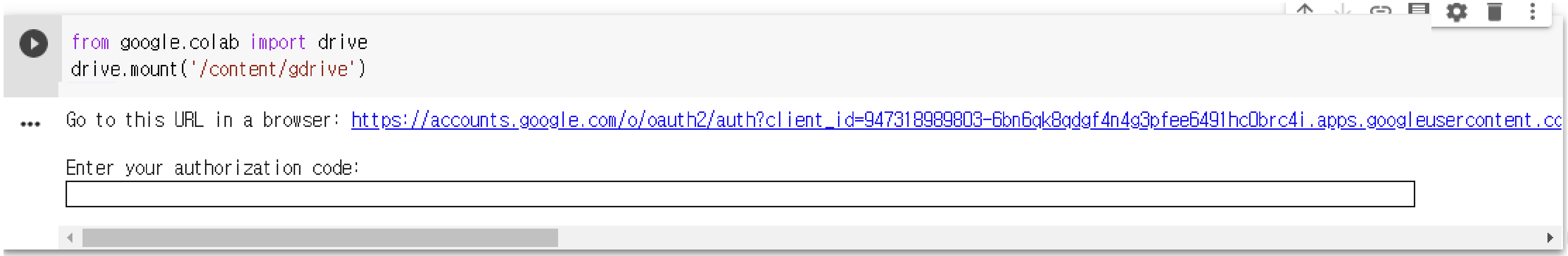
- Colab 클라우드에 파일을 업로드하는 방식: 파일이 삭제되면 다시 업로드를 해야 함
- GoogleDrive(개인): 특정 파일을 계속 사용할 경우 구글 드라이브에 파일을 업로드 한 후에 계속 사용하는 방식이 바람직함

- 1) 아래 명령어 수행

```
from google.colab import drive
drive.mount('/content/gdrive')
```



```
from google.colab import drive
drive.mount('/content/gdrive')
```

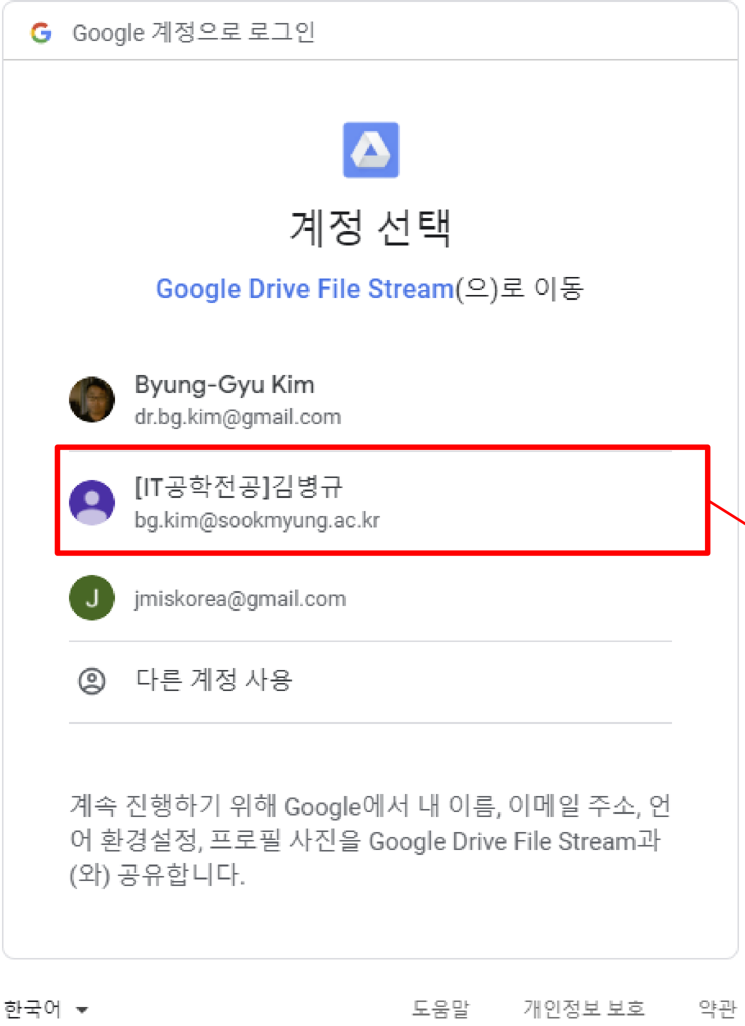


```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.cc

Enter your authorization code:
```

Google Colab 활용하기 : GoogleDrive 연결하기

- 2) 계정 선택화면으로



Google 계정으로 로그인

계정 선택

Google Drive File Stream(으)로 이동

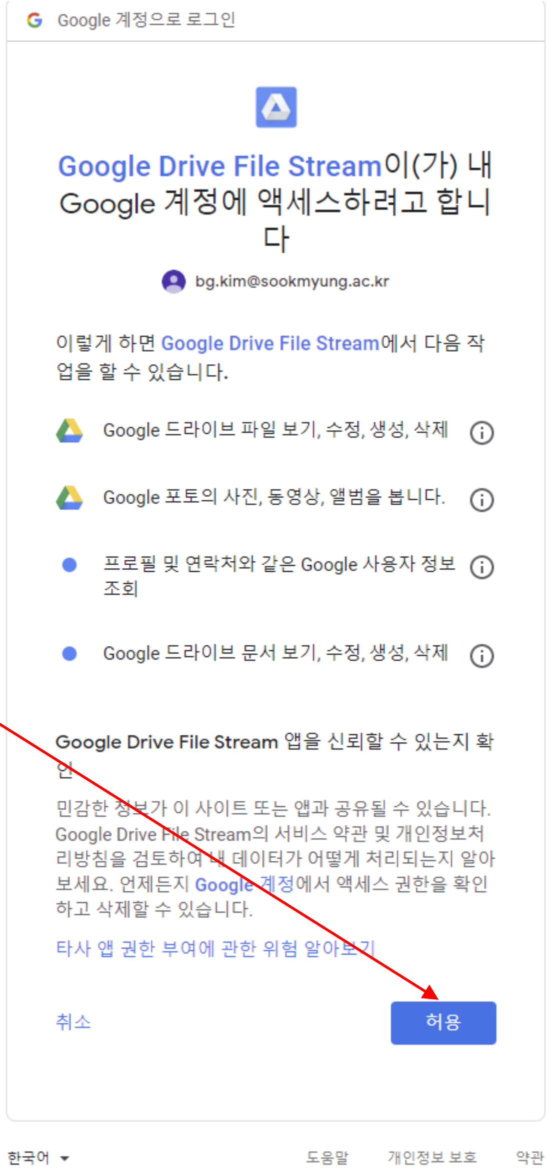
Byung-Gyu Kim
dr.bg.kim@gmail.com

[IT공학전공]김병규
bg.kim@sookmyung.ac.kr

jmiskorea@gmail.com

다른 계정 사용

계속 진행하기 위해 Google에서 내 이름, 이메일 주소, 언어 환경설정, 프로필 사진을 Google Drive File Stream과(와) 공유합니다.



Google 계정으로 로그인

Google Drive File Stream이(가) 내 Google 계정에 액세스하려고 합니다

bg.kim@sookmyung.ac.kr

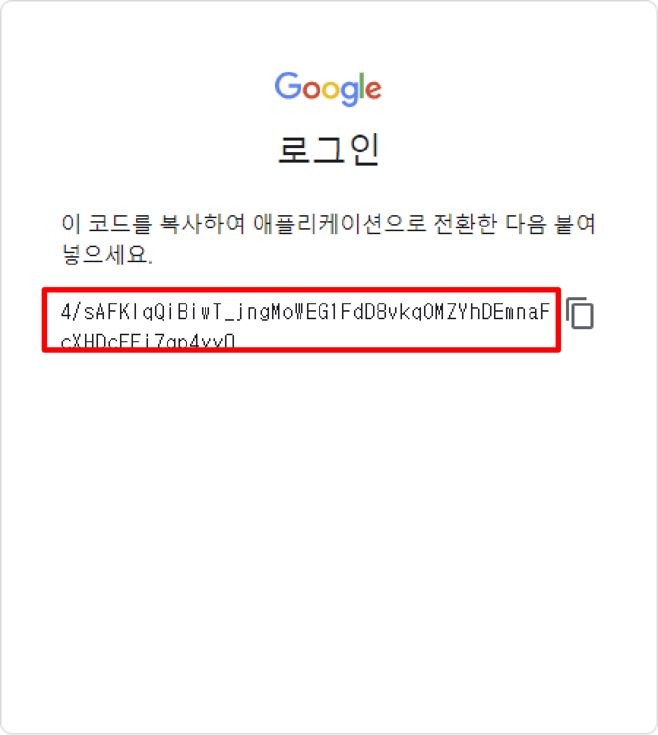
이렇게 하면 Google Drive File Stream에서 다음 작업을 할 수 있습니다.

- Google 드라이브 파일 보기, 수정, 생성, 삭제
- Google 포토의 사진, 동영상, 앨범을 봅니다.
- 프로필 및 연락처와 같은 Google 사용자 정보 조회
- Google 드라이브 문서 보기, 수정, 생성, 삭제

Google Drive File Stream 앱을 신뢰할 수 있는지 확인

민감한 정보가 이 사이트 또는 앱과 공유될 수 있습니다. Google Drive File Stream의 서비스 약관 및 개인정보처리방침을 검토하여 내 데이터가 어떻게 처리되는지 알아보세요. 언제든지 Google 계정에서 액세스 권한을 확인하고 삭제할 수 있습니다.

타사 앱 권한 부여에 관한 위험 알아보기



Google

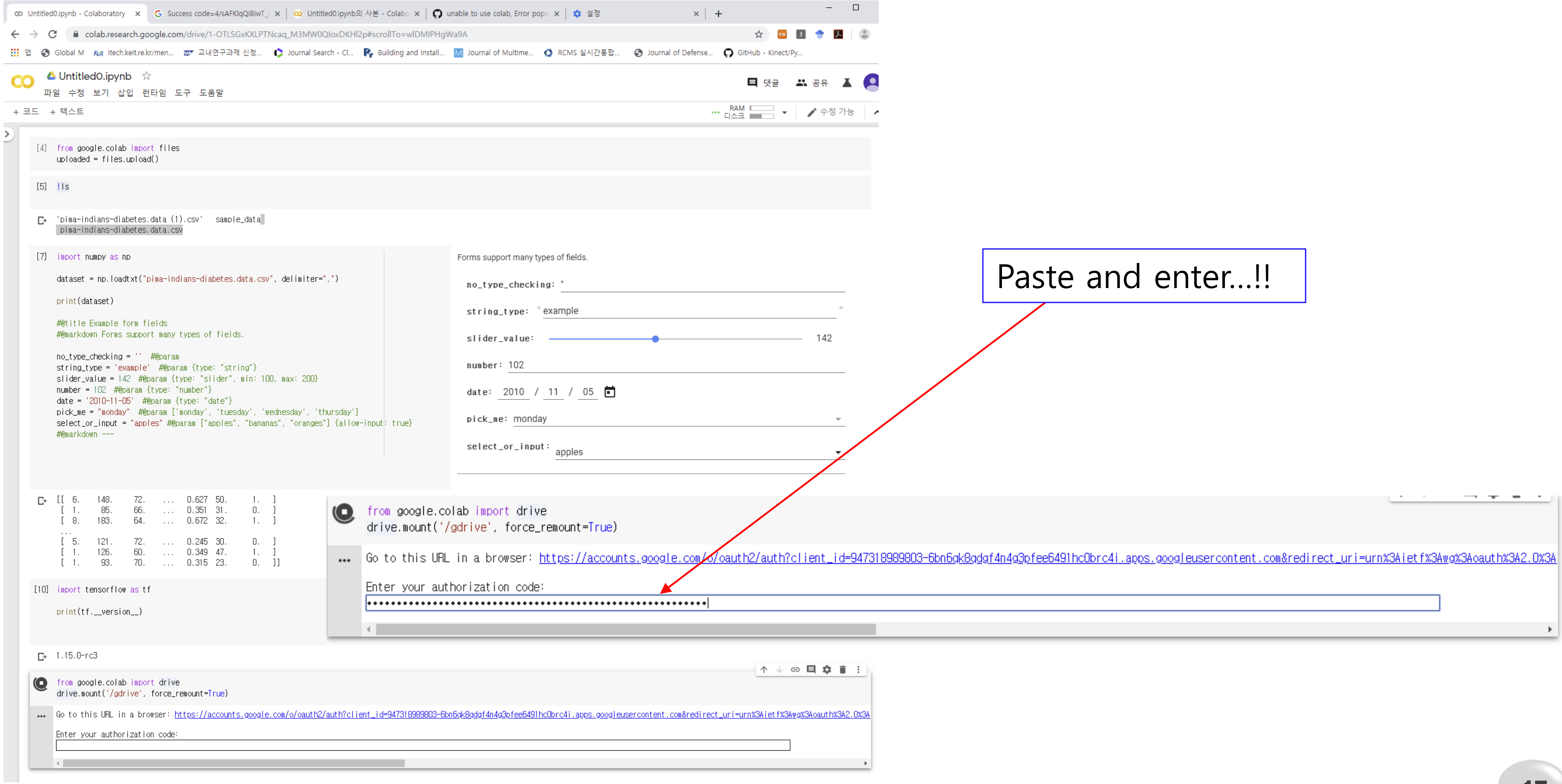
로그인

이 코드를 복사하여 애플리케이션으로 전환한 다음 붙여넣으세요.

```
4/sAFKlqQiBiwT_jngMoWEG1FdD8vkqOMZYhDEmnaF
cXHDcFFj7qp4vvQ
```

Google Colab 활용하기 : GoogleDrive 연결하기

- 3) 원래 jupyter notebook으로 돌아 온후 복사해 준다.


Untitled0.ipynb - Colaboratory
Success code=4/sAFKlqQiBiwT_
Untitled0.ipynb의 사본 - Colabo
unable to use colab, Error popu
설정
colab.research.google.com/drive/1-OTLSGxKXLPTNcaq_M3MW0QIoxDKHl2p#scrollTo=wIDMIPHgWa9A
앱
Global M
itech.keit.re.kr/men...
고내연구과제 신청...
Journal Search - Cl...
Building and Install...
Journal of Multime...
RCMS 실시간통합...
Journal of Defense...
GitHub - Kinect/Py...
Untitled0.ipynb
파일 수정 보기 삽입 런타임 도구 도움말
댓글
공유
+ 코드 + 텍스트
RAM
디스크
수정 가능

```
[4]  from google.colab import files
     uploaded = files.upload()
```

```
[5]  !ls
```

```
'pima-indians-diabetes.data (1).csv'   sample_data
 pima-indians-diabetes.data.csv
```

```
[7]  import numpy as np

     dataset = np.loadtxt("pima-indians-diabetes.data.csv", delimiter=",")

     print(dataset)

     #@title Example form fields
     #@markdown Forms support many types of fields.

     no_type_checking = ''  #@param
     string_type = 'example'  #@param {type: "string"}
     slider_value = 142  #@param {type: "slider", min: 100, max: 200}
     number = 102  #@param {type: "number"}
     date = '2010-11-05'  #@param {type: "date"}
     pick_me = "monday"  #@param ['monday', 'tuesday', 'wednesday', 'thursday']
     select_or_input = "apples" #@param ["apples", "bananas", "oranges"] {allow-input: true}
     #@markdown ---
```

Forms support many types of fields.

no_type_checking: ''

string_type: "example"

slider_value: 142

number: 102

date: 2010 / 11 / 05

pick_me: monday

select_or_input: apples

```
[[  6.    148.     72.    ...    0.627  50.      1.   ]
 [  1.     85.     66.    ...    0.351  31.      0.   ]
 [  8.    183.     64.    ...    0.672  32.      1.   ]
 ...
 [  5.    121.     72.    ...    0.245  30.      0.   ]
 [  1.    126.     60.    ...    0.349  47.      1.   ]
 [  1.     93.     70.    ...    0.315  23.      0.   ]]
```

```
[10] import tensorflow as tf

     print(tf.__version__)
```

```
1.15.0-rc3
```

```
from google.colab import drive
drive.mount('/gdrive', force_remount=True)
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3A

Enter your authorization code:
```

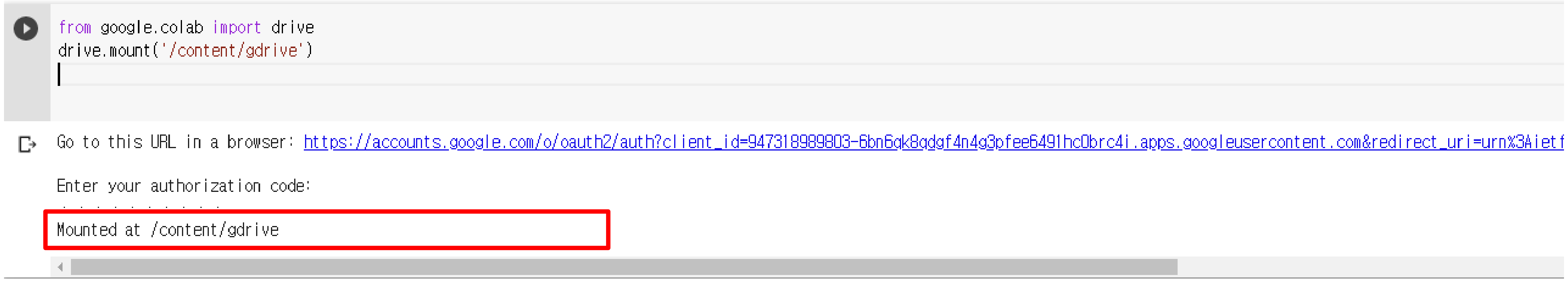
```
from google.colab import drive
drive.mount('/gdrive', force_remount=True)
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3A

Enter your authorization code:
•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
```

Paste and enter...!!

Google Colab 활용하기 : GoogleDrive 연결하기

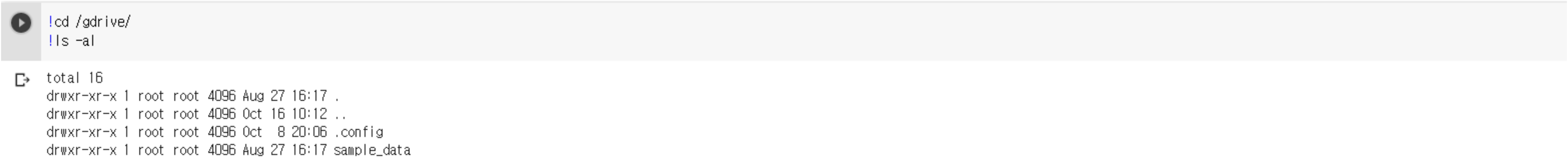
- 4) "gdrive"라는 폴더가 생성되어 mount 된 것을 볼 수 있다.



```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf

Enter your authorization code:
Mounted at /content/gdrive
```

```
!cd /gdrive/
!ls -al
```



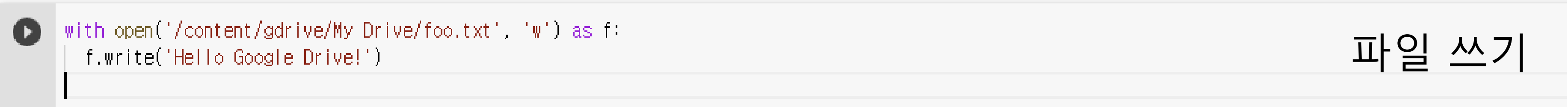
```
!cd /gdrive/
!ls -al
```

```
total 16
drwxr-xr-x 1 root root 4096 Aug 27 16:17 .
drwxr-xr-x 1 root root 4096 Oct 16 10:12 ..
drwxr-xr-x 1 root root 4096 Oct  8 20:06 .config
drwxr-xr-x 1 root root 4096 Aug 27 16:17 sample_data
```

Google Colab 활용하기 : GoogleDrive 연결하기

- 5) "gdrive"→"My drive"에 있는 파일 쓰기/접근하기

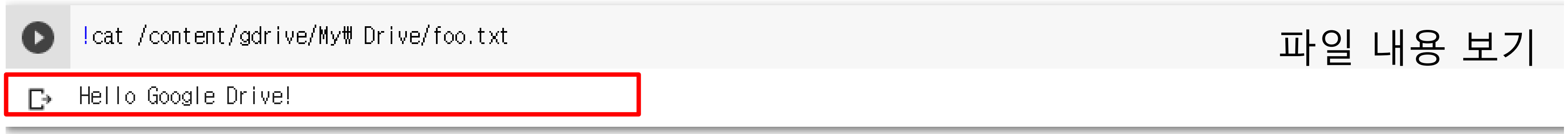
```
with open('/content/gdrive/My Drive/foo.txt', 'w') as f:
    f.write('Hello Google Drive!')
```



```
with open('/content/gdrive/My Drive/foo.txt', 'w') as f:
  f.write('Hello Google Drive!')
```

파일 쓰기

```
!cat /content/gdrive/My\ Drive/foo.txt
```



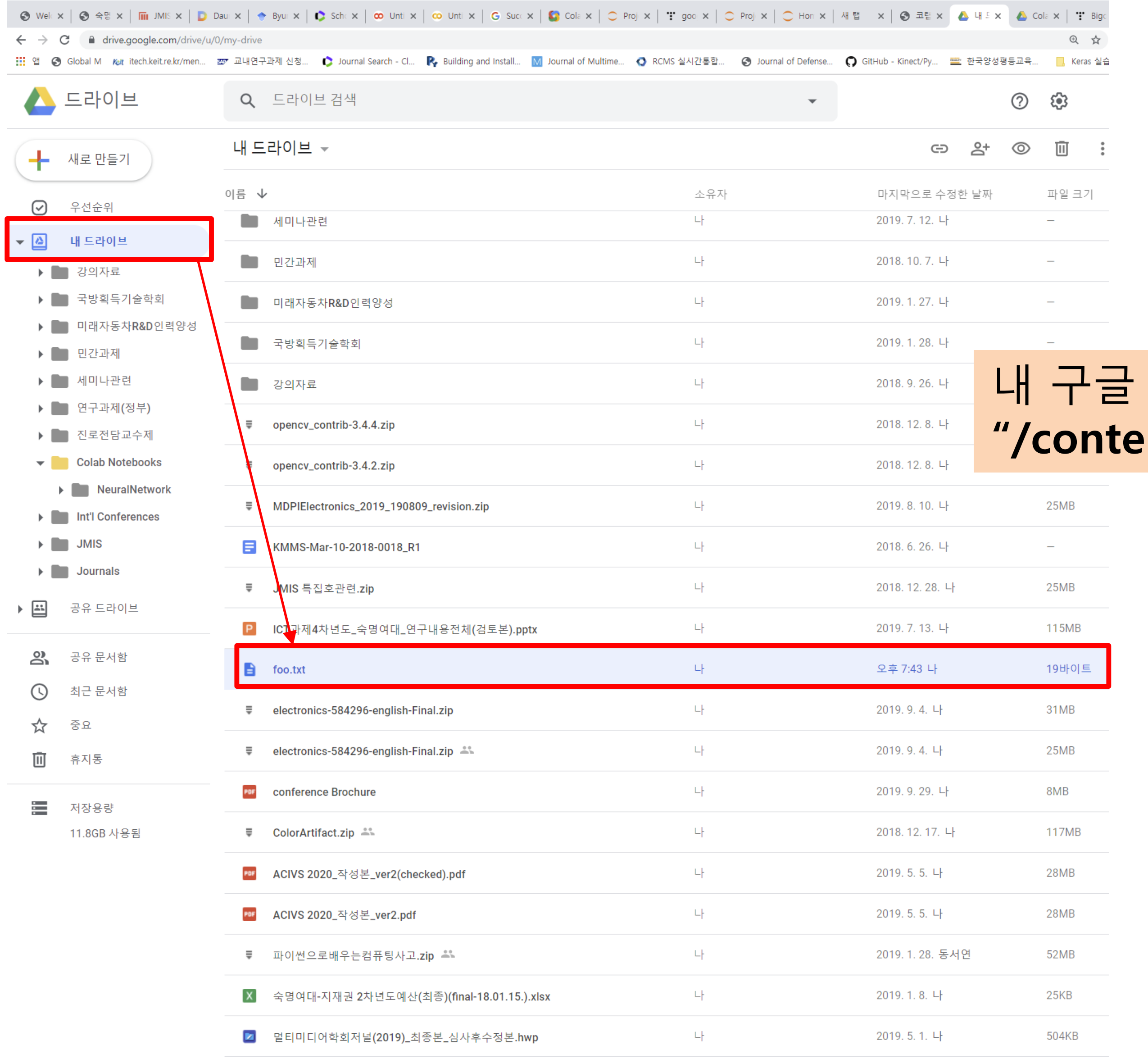
```
!cat /content/gdrive/My₩ Drive/foo.txt
```

```
Hello Google Drive!
```

파일 내용 보기

Google Colab 활용하기 : GoogleDrive 연결하기

❖ 실제 google drive 어디에 foo.txt가 만들어졌는지 가볼까요??

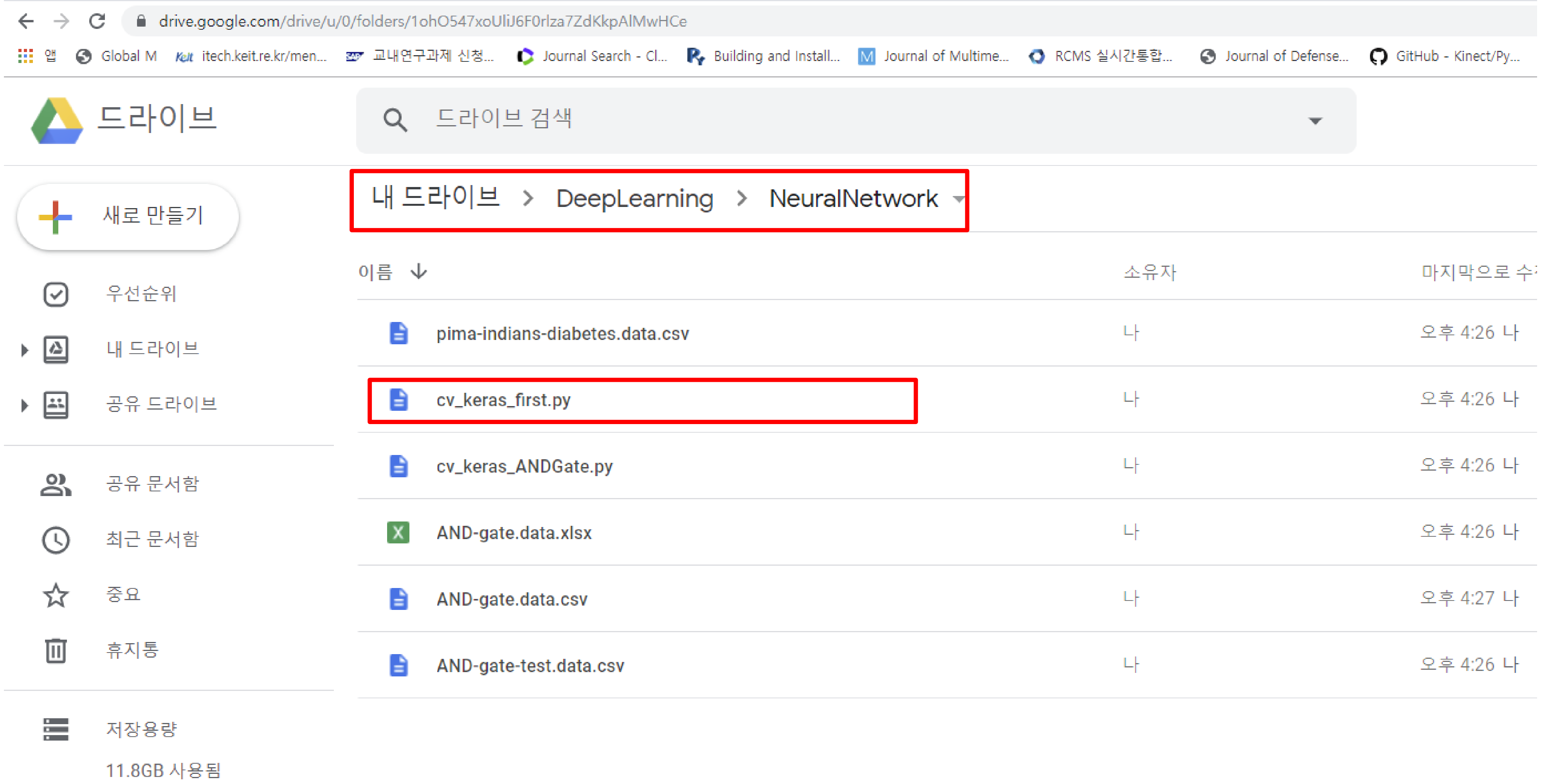


이름 ↓	소유자	마지막으로 수정한 날짜	파일 크기
세미나관련	나	2019. 7. 12. 나	–
민간과제	나	2018. 10. 7. 나	–
미래자동차R&D인력양성	나	2019. 1. 27. 나	–
국방획득기술학회	나	2019. 1. 28. 나	–
강의자료	나	2018. 9. 26. 나	
opencv_contrib-3.4.4.zip	나	2018. 12. 8. 나	
opencv_contrib-3.4.2.zip	나	2018. 12. 8. 나	
MDPIElectronics_2019_190809_revision.zip	나	2019. 8. 10. 나	25MB
KMMS-Mar-10-2018-0018_R1	나	2018. 6. 26. 나	–
JMIS 특집호관련.zip	나	2018. 12. 28. 나	25MB
ICT과제4차년도_숙명여대_연구내용전체(검토본).pptx	나	2019. 7. 13. 나	115MB
foo.txt	나	오후 7:43 나	19바이트
electronics-584296-english-Final.zip	나	2019. 9. 4. 나	31MB
electronics-584296-english-Final.zip	나	2019. 9. 4. 나	25MB
conference Brochure	나	2019. 9. 29. 나	8MB
ColorArtifact.zip	나	2018. 12. 17. 나	117MB
ACIVS 2020_작성본_ver2(checked).pdf	나	2019. 5. 5. 나	28MB
ACIVS 2020_작성본_ver2.pdf	나	2019. 5. 5. 나	28MB
파이썬으로배우는컴퓨팅사고.zip	나	2019. 1. 28. 동서연	52MB
숙명여대-지재권 2차년도예산(최종)(final-18.01.15.).xlsx	나	2019. 1. 8. 나	25KB
멀티미디어학회저널(2019)_최종본_심사후수정본.hwp	나	2019. 5. 1. 나	504KB

내 구글 드라이브까지 표준 경로:
"/content/gdrive/My Drive/"

Google Colab 활용하기 : GoogleDrive 연결하기

❖ Google drive→"내드라이브" → 임의의 폴더 내 파일 읽어 보기


drive.google.com/drive/u/0/folders/1ohO547xoUliJ6F0rlza7ZdKkpAlMwHCe
앱 Global M itech.keit.re.kr/men... 교내연구과제 신청... Journal Search - Cl... Building and Install... Journal of Multime... RCMS 실시간통합... Journal of Defense... GitHub - Kinect/Py...
드라이브
드라이브 검색
새로 만들기
우선순위
내 드라이브
공유 드라이브
공유 문서함
최근 문서함
중요
휴지통
저장용량
11.8GB 사용됨

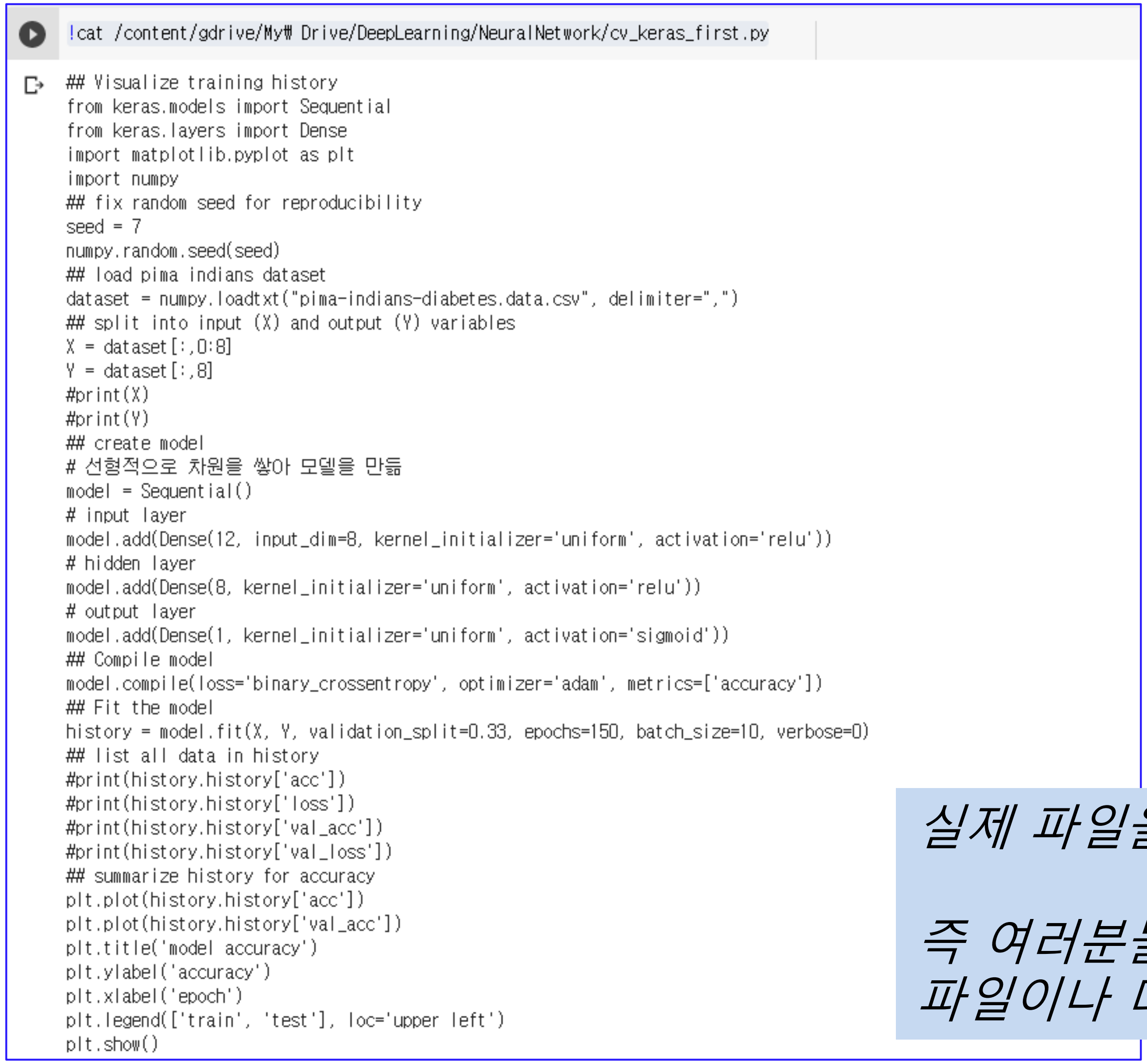
내 드라이브 > DeepLearning > NeuralNetwork

이름 ↓	소유자	마지막으로 수
pima-indians-diabetes.data.csv	나	오후 4:26 나
cv_keras_first.py	나	오후 4:26 나
cv_keras_ANDGate.py	나	오후 4:26 나
AND-gate.data.xlsx	나	오후 4:26 나
AND-gate.data.csv	나	오후 4:27 나
AND-gate-test.data.csv	나	오후 4:26 나

Google Colab 활용하기 : GoogleDrive 연결하기

- Jupyter Notebook 에서

```
!cat /content/gdrive/My\ Drive/DeepLearning/NeuralNetwork/cv_keras_first.py
```



```
!cat /content/gdrive/My₩ Drive/DeepLearning/NeuralNetwork/cv_keras_first.py
```

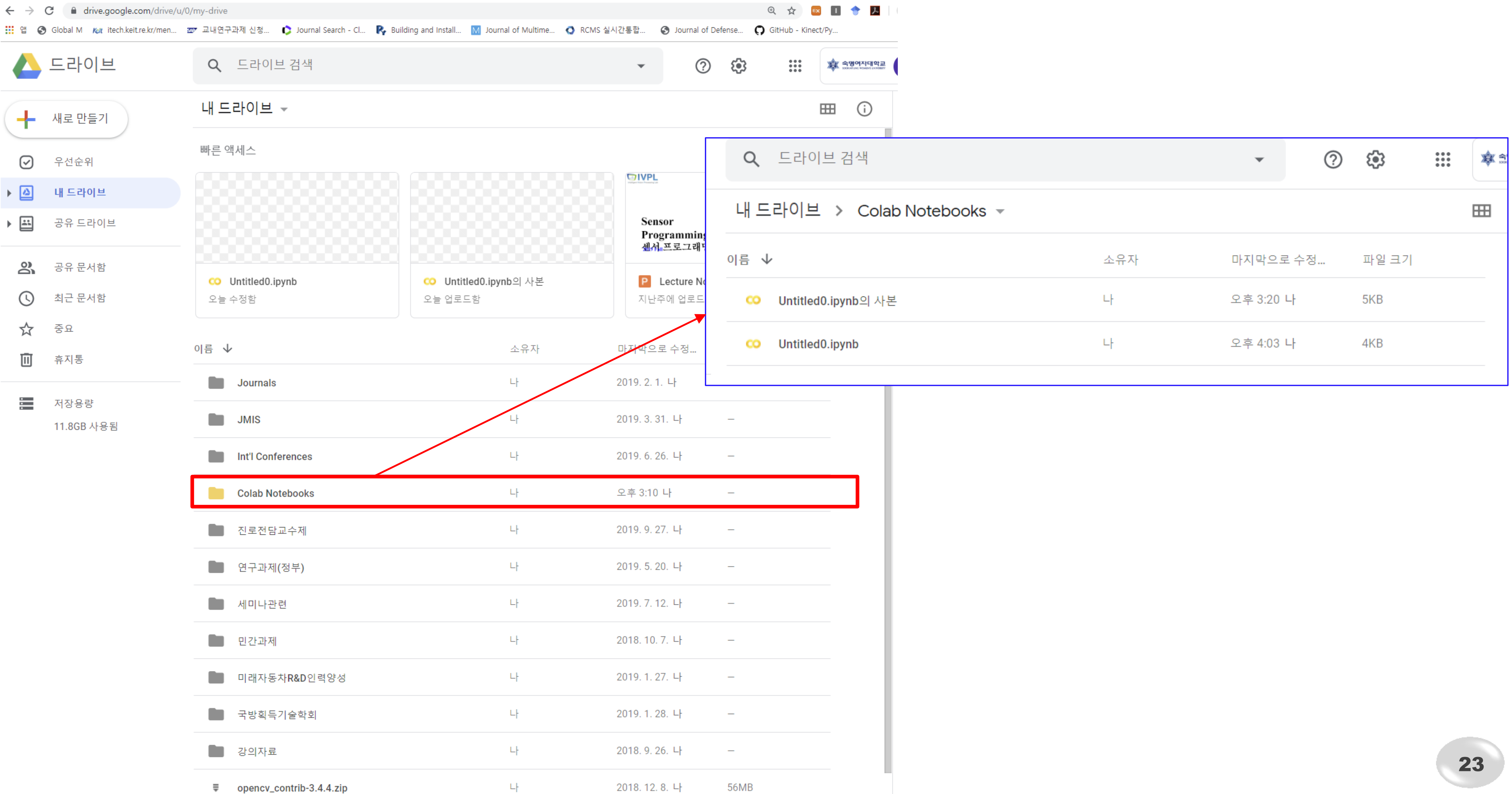
```
## Visualize training history
from keras.models import Sequential
from keras.layers import Dense
import matplotlib.pyplot as plt
import numpy
## fix random seed for reproducibility
seed = 7
numpy.random.seed(seed)
## load pima indians dataset
dataset = numpy.loadtxt("pima-indians-diabetes.data.csv", delimiter=",")
## split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
#print(X)
#print(Y)
## create model
# 선형적으로 차원을 쌓아 모델을 만듦
model = Sequential()
# input layer
model.add(Dense(12, input_dim=8, kernel_initializer='uniform', activation='relu'))
# hidden layer
model.add(Dense(8, kernel_initializer='uniform', activation='relu'))
# output layer
model.add(Dense(1, kernel_initializer='uniform', activation='sigmoid'))
## Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
## Fit the model
history = model.fit(X, Y, validation_split=0.33, epochs=150, batch_size=10, verbose=0)
## list all data in history
#print(history.history['acc'])
#print(history.history['loss'])
#print(history.history['val_acc'])
#print(history.history['val_loss'])
## summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

실제 파일을 접근하여 잘 읽어올 수 있음을 알 수 있다.

즉 여러분들이 소스나 실습 코드 구글 드라이브에 올리고 파일이나 데이터에 대한 접근이 가능하다는 것을 확인함

Google Colab 활용하기 : GoogleDrive 연결하기

- Google drive 확인: Colab Notebooks 폴더 확인(본인이 작업하는 작업의 임시 저장소)



drive.google.com/drive/u/0/my-drive

11.8GB 사용됨

내 드라이브

빠른 액세스

Untitled0.ipynb
오늘 수정함

Untitled0.ipynb의 사본
오늘 업로드함

Sensor Programming
센서 프로그래밍

Lecture No
지난주에 업로드

이름 ↓	소유자	마지막으로 수정...	
Journals	나	2019. 2. 1. 나	
JMIS	나	2019. 3. 31. 나	–
Int'l Conferences	나	2019. 6. 26. 나	–
Colab Notebooks	나	오후 3:10 나	–
진로전담교수제	나	2019. 9. 27. 나	–
연구과제(정부)	나	2019. 5. 20. 나	–
세미나관련	나	2019. 7. 12. 나	–
민간과제	나	2018. 10. 7. 나	–
미래자동차R&D인력양성	나	2019. 1. 27. 나	–
국방획득기술학회	나	2019. 1. 28. 나	–
강의자료	나	2018. 9. 26. 나	–
opencv_contrib-3.4.4.zip	나	2018. 12. 8. 나	56MB

드라이브 검색

내 드라이브 > Colab Notebooks

이름 ↓	소유자	마지막으로 수정...	파일 크기
Untitled0.ipynb의 사본	나	오후 3:20 나	5KB
Untitled0.ipynb	나	오후 4:03 나	4KB

IVPL
Intelligent Vision Processing Lab

Google Colab 활용하기 : Keras 기반 python code testing (1)

❖ Test code 수행(keras 기반)

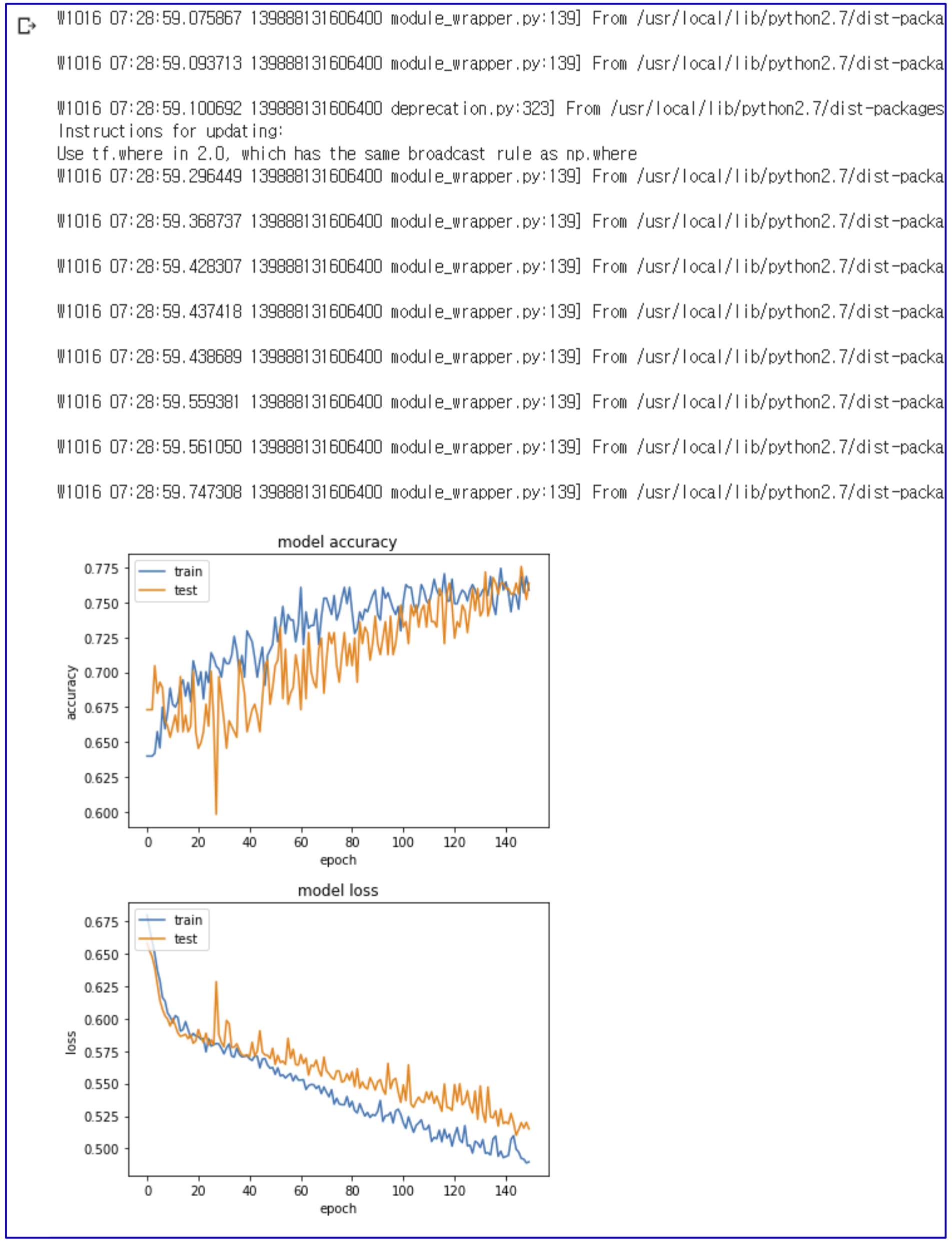
```
## Visualize training history
from keras.models import Sequential
from keras.layers import Dense
import matplotlib.pyplot as plt
import numpy
## fix random seed for reproducibility
seed = 7
numpy.random.seed(seed)
## load pima indians dataset
dataset = numpy.loadtxt("pima-indians-
diabetes.data.csv", delimiter=",")
## split into input (X) and output (Y)
variables
X = dataset[:,0:8]
Y = dataset[:,8]
#print(X)
#print(Y)
## create model
     (계 속)
```

```
## create model
# 선형적으로 차원을 쌓아 모델을 만듦
model = Sequential()
# input layer
model.add(Dense(12, input_dim=8,
kernel_initializer='uniform',
activation='relu'))
# hidden layer
model.add(Dense(8,
kernel_initializer='uniform',
activation='relu'))
# output layer
model.add(Dense(1,
kernel_initializer='uniform',
activation='sigmoid'))
## Compile model
model.compile(loss='binary_crossentropy',
optimizer='adam', metrics=['accuracy'])
## Fit the model
history = model.fit(X, Y,
validation_split=0.33, epochs=150,
batch_size=10, verbose=0)
```

Google Colab 활용하기 : Keras 기반 python code testing (2)

(계 속)

```
## list all data in history
## summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper
left')
plt.show()
## summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper
left')
plt.show()
```

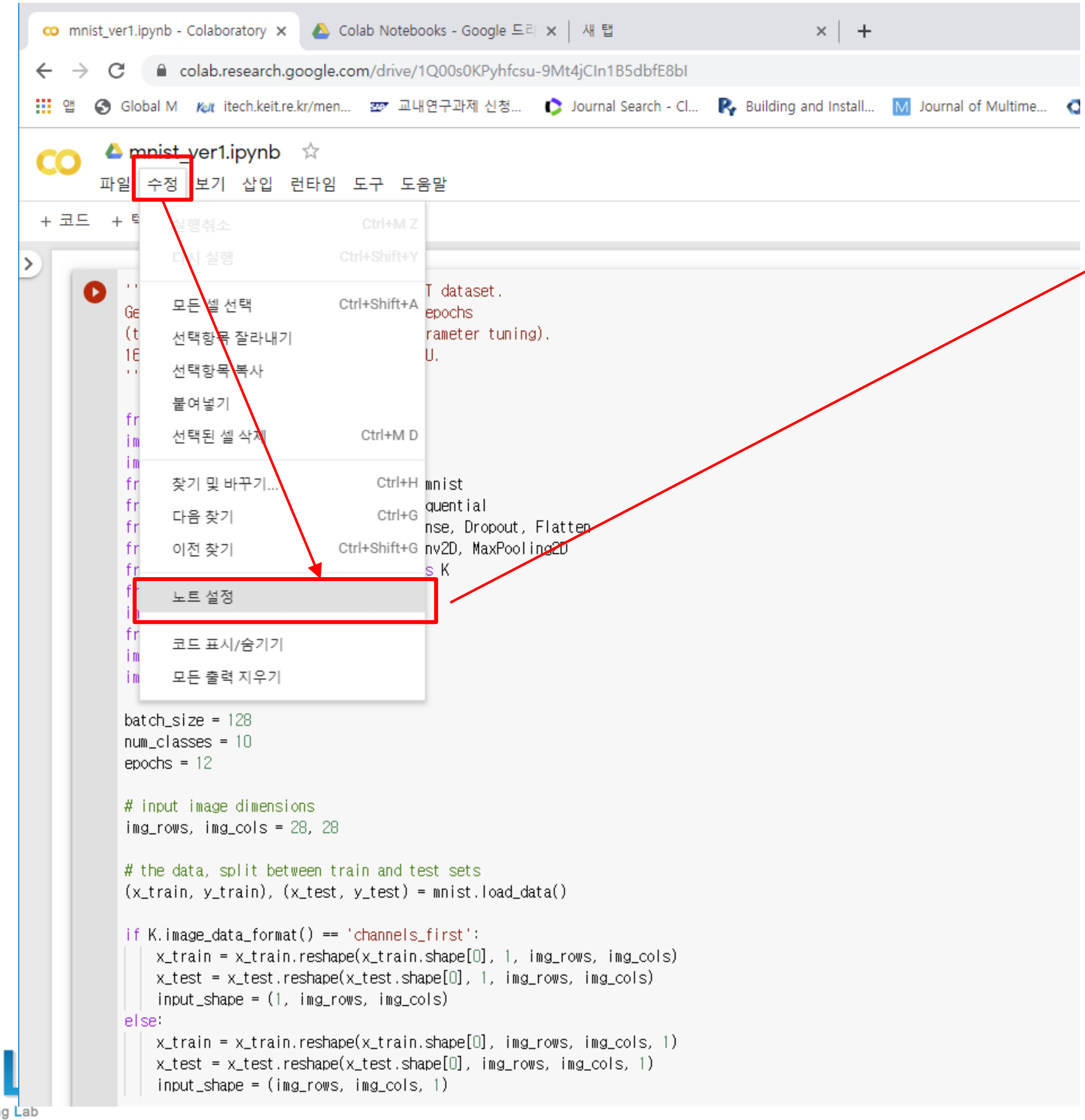


```
W1016 07:28:59.075867 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.093713 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.100692 139888131606400 deprecation.py:323] From /usr/local/lib/python2.7/dist-packages
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
W1016 07:28:59.296449 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.368737 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.428307 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.437418 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.438689 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.559381 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.561050 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa

W1016 07:28:59.747308 139888131606400 module_wrapper.py:139] From /usr/local/lib/python2.7/dist-packa
```

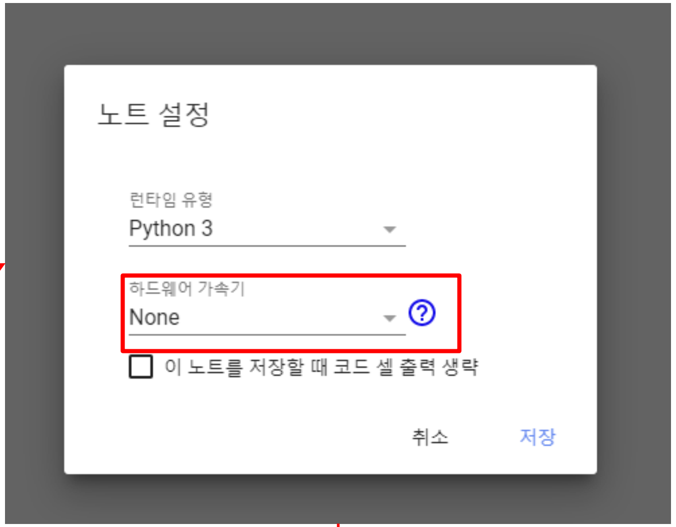
Jupyter notebook에서 수행 결과 확인

Google Colab 활용하기 : GPU 사용 설정하기 (1)

❖ Notebook 메뉴: "수정"→ "노트설정 "



```
batch_size = 128
num_classes = 10
epochs = 12

# input image dimensions
img_rows, img_cols = 28, 28

# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)
```

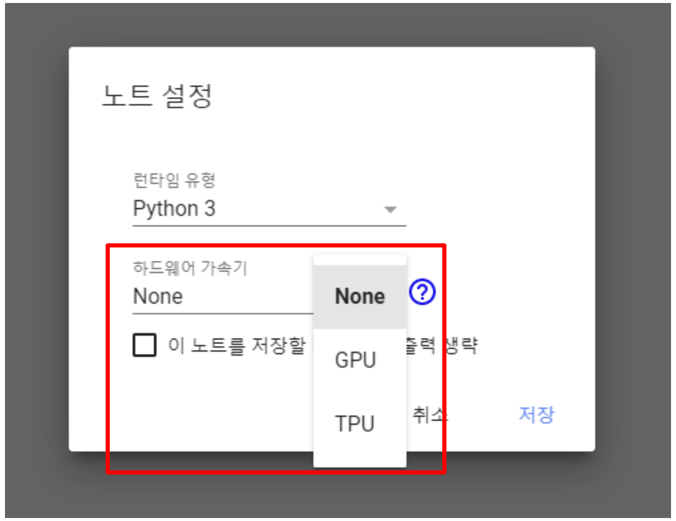


노트 설정

런타임 유형: Python 3

하드웨어 가속기: None

□ 이 노트를 저장할 때 코드 셀 출력 생략



노트 설정

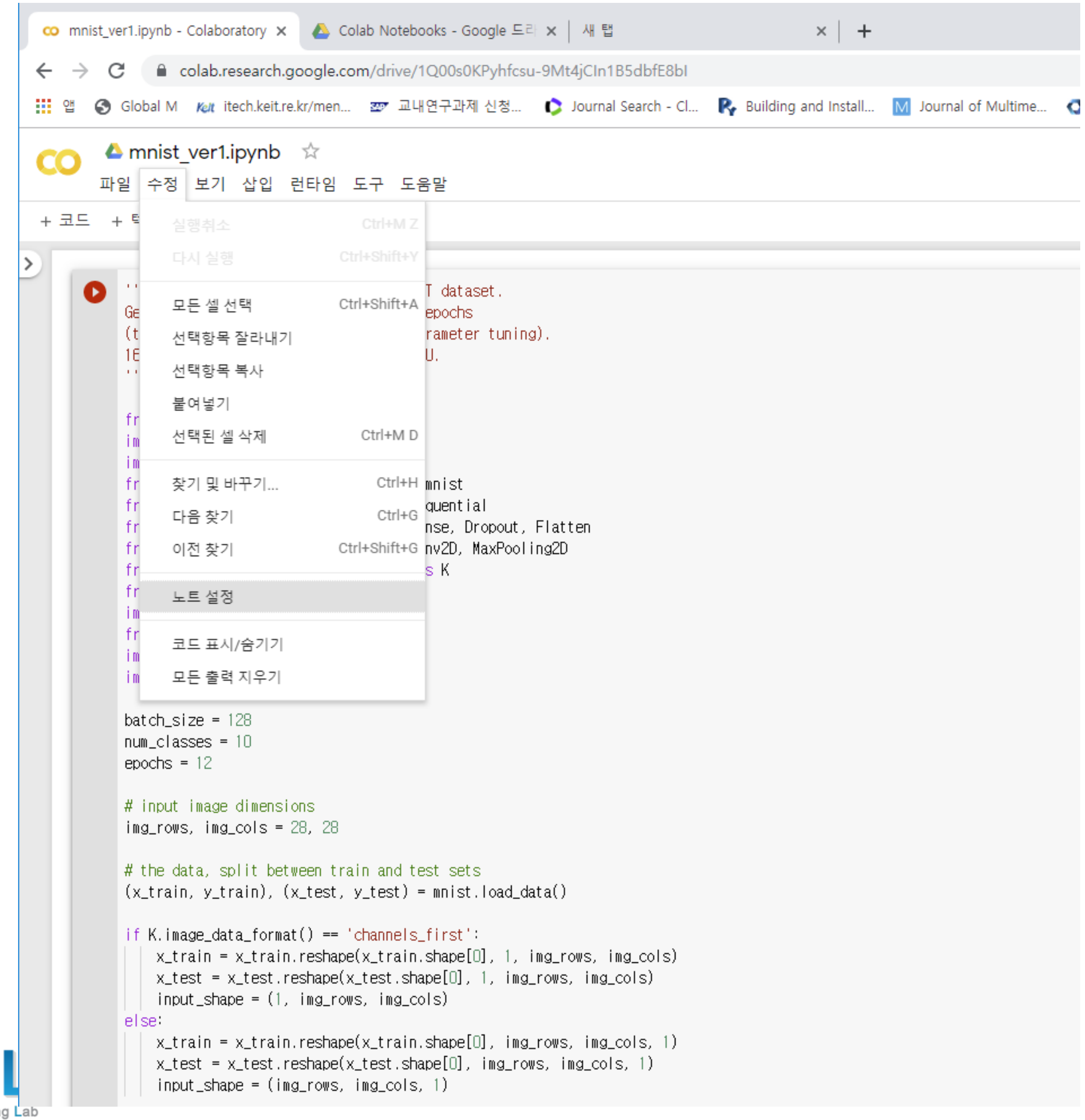
런타임 유형: Python 3

하드웨어 가속기: None

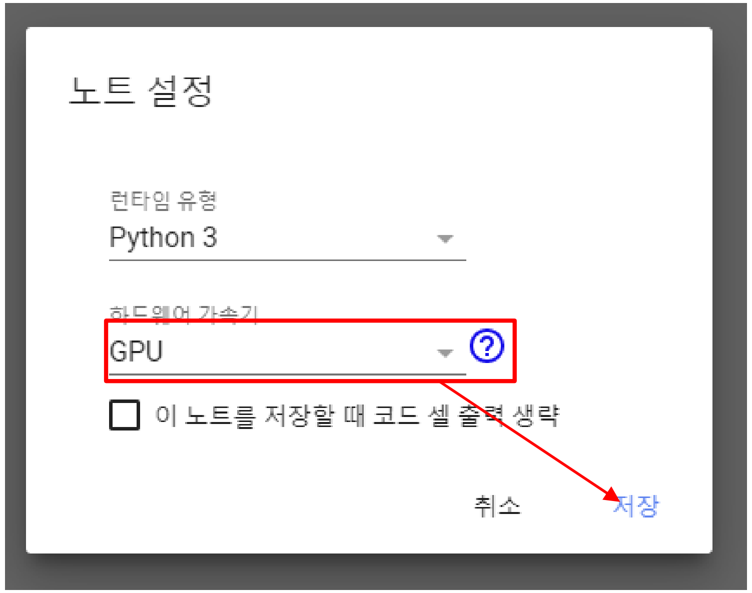
□ 이 노트를 저장할 출력 생략

Google Colab 활용하기 : GPU 사용 설정하기 (2)

❖ Notebook 메뉴: “수정”→ “노트설정”→ 하드웨어 가속기: GPU 선택 후 저장



```
'''                T dataset.
Ge                 epochs
(t                 rameter tuning).
16                 U.
'''

fr
im
im
fr                 mnist
fr                 quential
fr                 nse, Dropout, Flatten
fr                 nv2D, MaxPooling2D
fr                 s K
fr
im
fr
im
im

batch_size = 128
num_classes = 10
epochs = 12

# input image dimensions
img_rows, img_cols = 28, 28

# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)
```



노트 설정

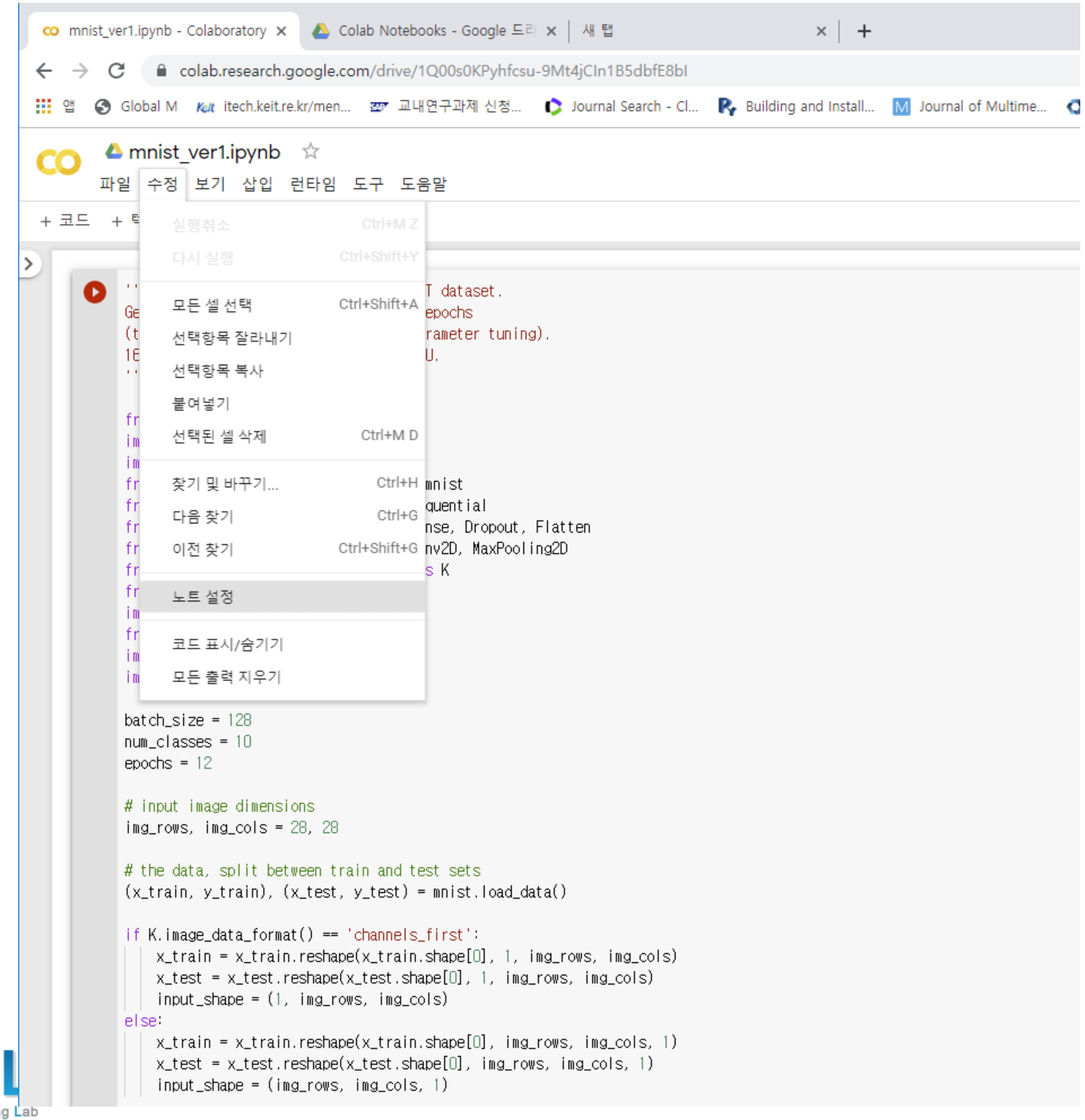
런타임 유형: Python 3

하드웨어 가속기: GPU

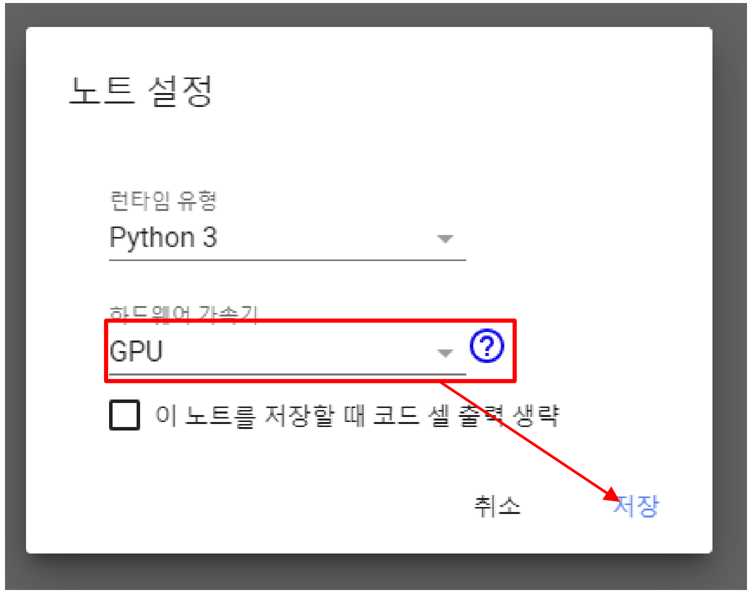
□ 이 노트를 저장할 때 코드 셀 출력 생략

Google Colab 활용하기 : GPU 사용 설정하기 (3)

❖ Notebook 메뉴: "수정"→ "노트설정"→ 하드웨어 가속기: GPU 선택 후 저장



```
'''                              T dataset.
Ge                               epochs
(t                               rameter tuning).
16                               U.
'''

fr
im
im
fr                               mnist
fr                               quential
fr                               nse, Dropout, Flatten
fr                               nv2D, MaxPooling2D
fr                               s K
fr
im
fr
im
im

batch_size = 128
num_classes = 10
epochs = 12

# input image dimensions
img_rows, img_cols = 28, 28

# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)
```



노트 설정

런타임 유형: Python 3

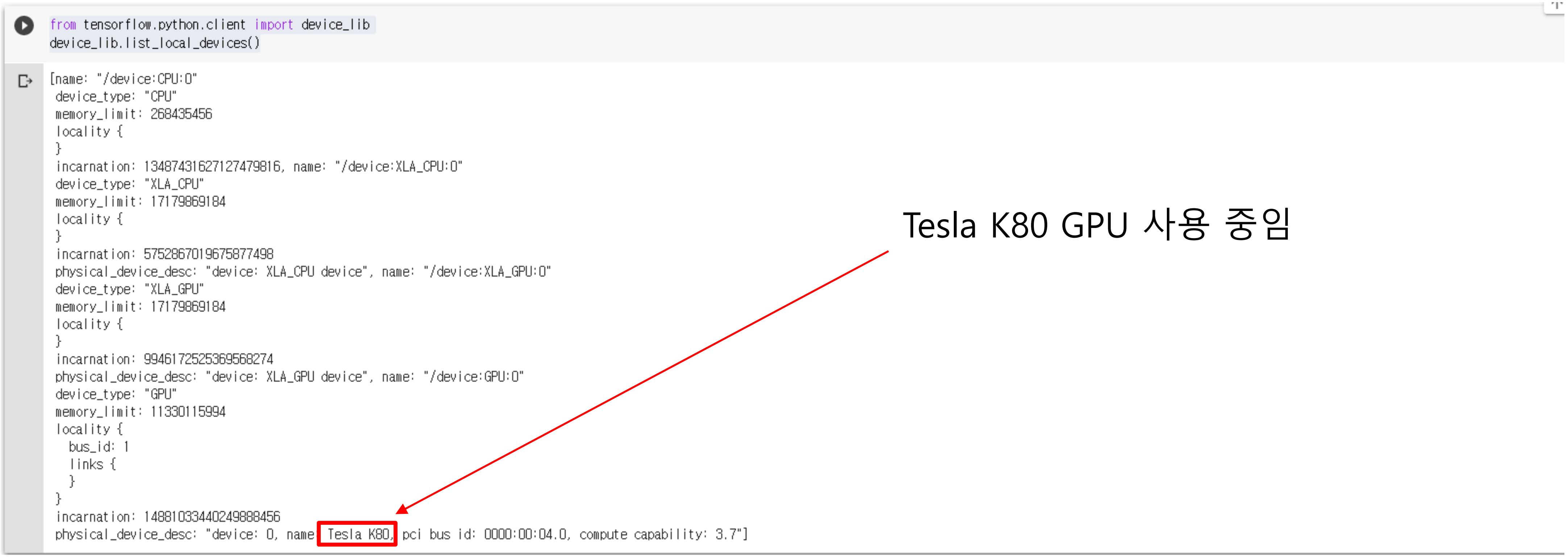
하드웨어 가속기: GPU

□ 이 노트를 저장할 때 코드 셀 출력 생략

Google Colab 활용하기 : GPU 사용 설정하기 (4)

- GPU 종류 확인하기

```
from tensorflow.python.client import device_lib
device_lib.list_local_devices()
```



```
from tensorflow.python.client import device_lib
device_lib.list_local_devices()
```

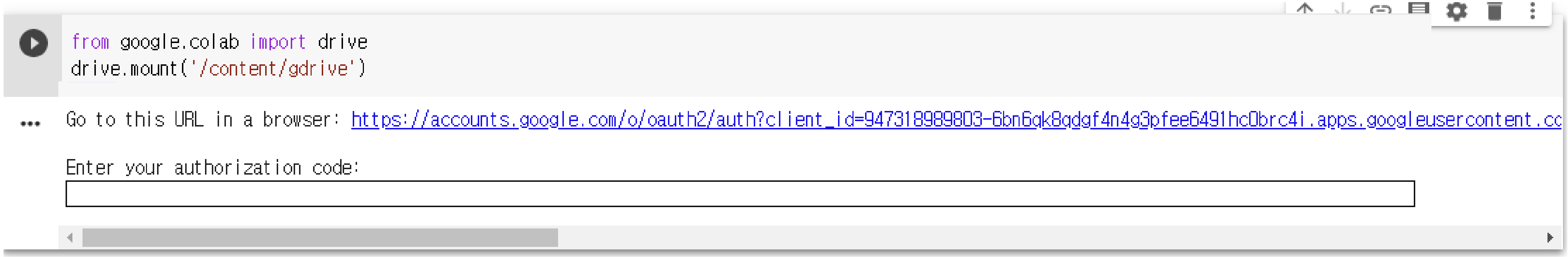
```
[name: "/device:CPU:0"
 device_type: "CPU"
 memory_limit: 268435456
 locality {
 }
 incarnation: 13487431627127479816, name: "/device:XLA_CPU:0"
 device_type: "XLA_CPU"
 memory_limit: 17179869184
 locality {
 }
 incarnation: 5752867019675877498
 physical_device_desc: "device: XLA_CPU device", name: "/device:XLA_GPU:0"
 device_type: "XLA_GPU"
 memory_limit: 17179869184
 locality {
 }
 incarnation: 9946172525369568274
 physical_device_desc: "device: XLA_GPU device", name: "/device:GPU:0"
 device_type: "GPU"
 memory_limit: 11330115994
 locality {
   bus_id: 1
   links {
   }
 }
 incarnation: 14881033440249888456
 physical_device_desc: "device: 0, name: Tesla K80, pci bus id: 0000:00:04.0, compute capability: 3.7"]
```

Tesla K80 GPU 사용 중임

Mnist : Digit recognition project (1)

❖ 1) google drive를 먼저 연결한다 (강의 자료 10페이지 이후 참고).

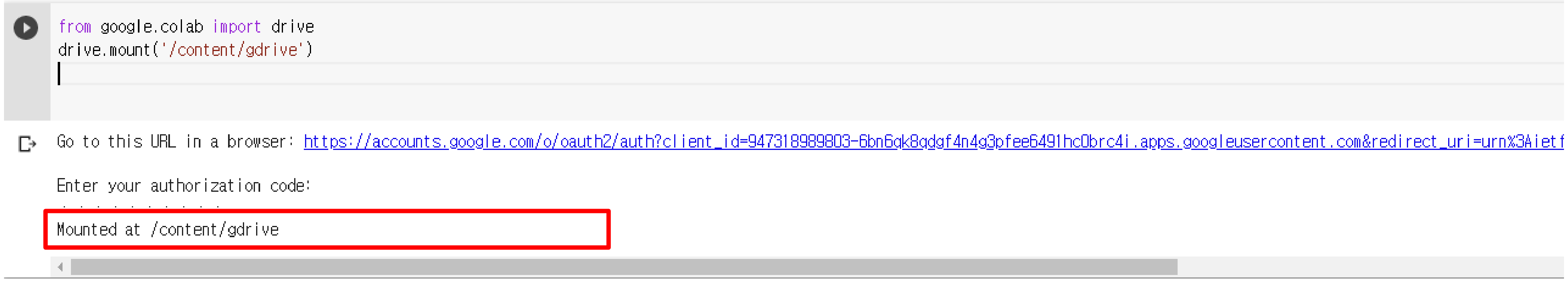
```python
from google.colab import drive
drive.mount('/content/gdrive')
```



```python
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.cc

Enter your authorization code:
```

- “Mounted at /content/gdrive” : Mount is successful….!!!!



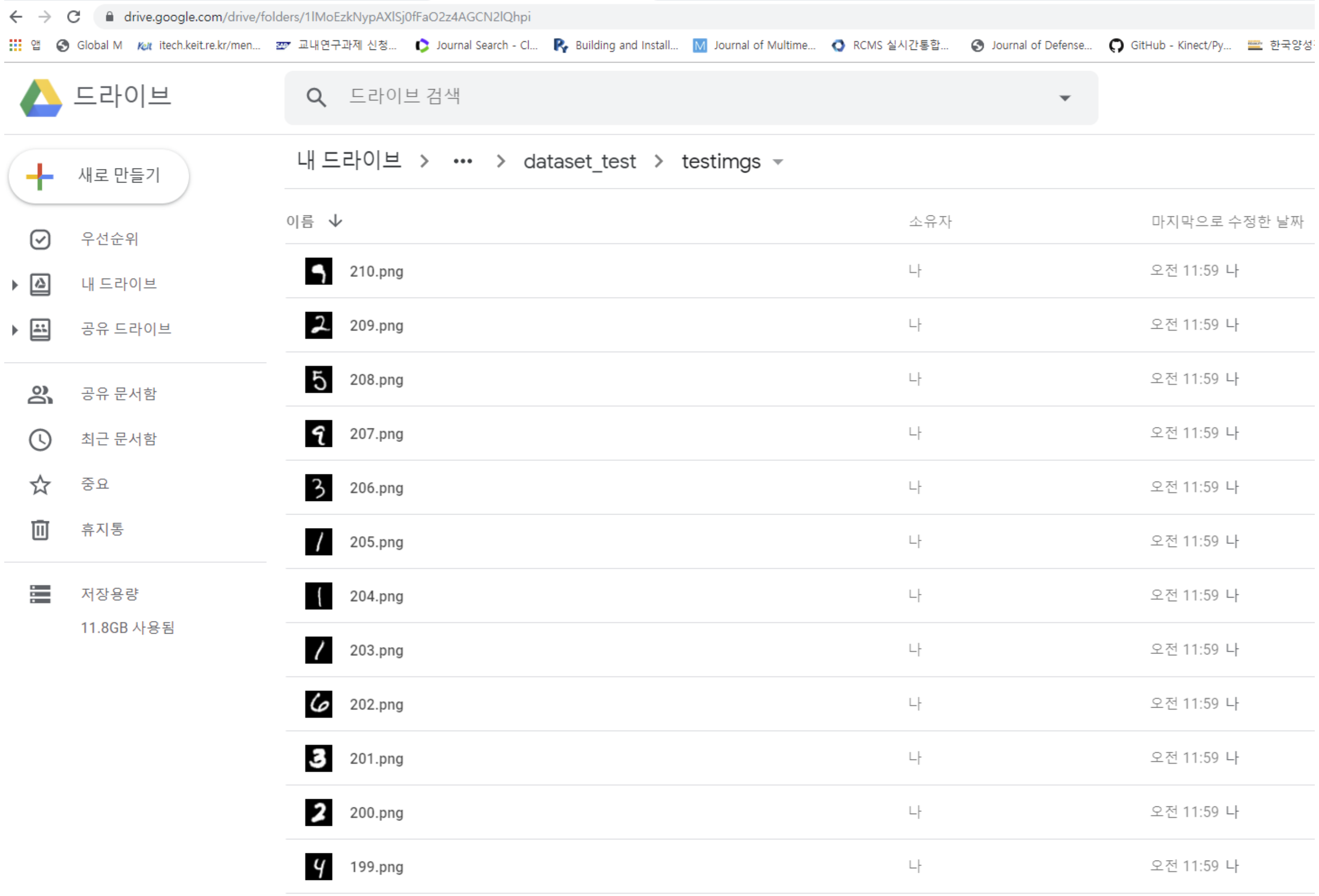
```python
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf

Enter your authorization code:
··········
Mounted at /content/gdrive
```

Mnist : Digit recognition project (2)

❖ 2) mnist 프로젝트에 필요한 데이트를 원하는 google drive (내 드라이브)에 미리 복사해 놓는다.

"My Drive/DeepLearning/cnn/mnist/dataset_test/testimgs/"



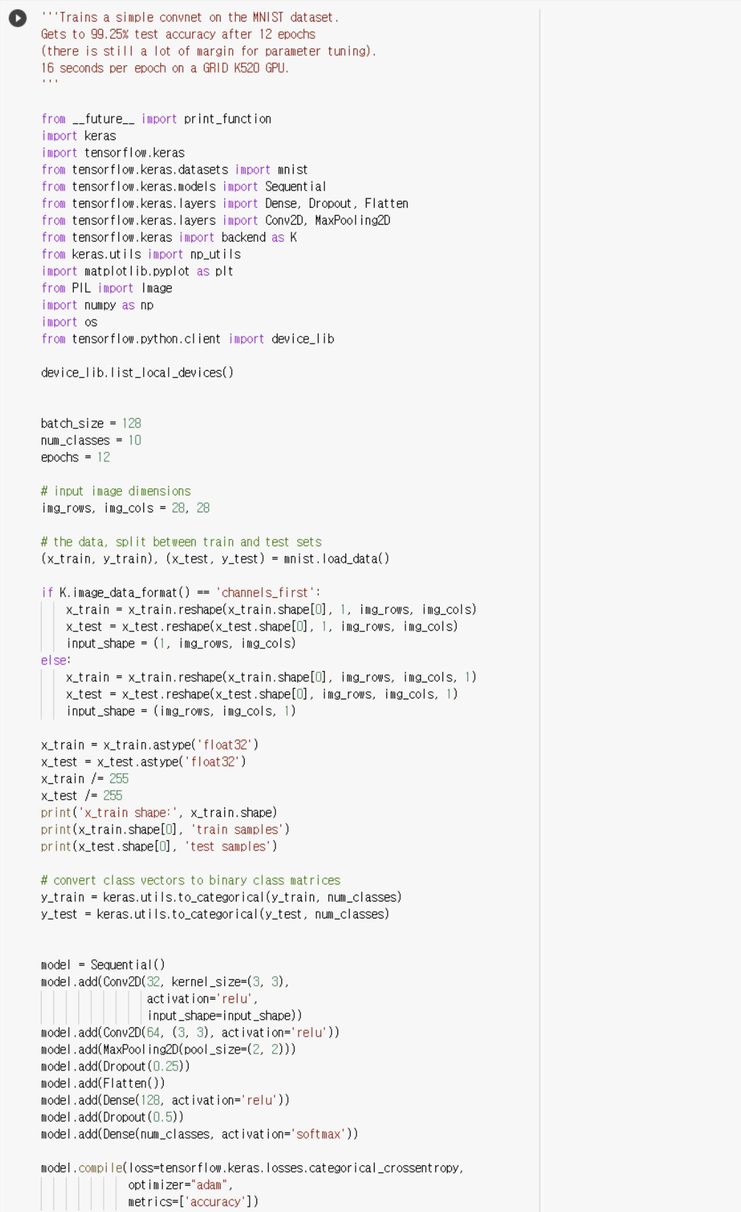
drive.google.com/drive/folders/1lMoEzkNypAXlSj0fFaO2z4AGCN2lQhpi

내 드라이브 > ••• > dataset_test > testimgs

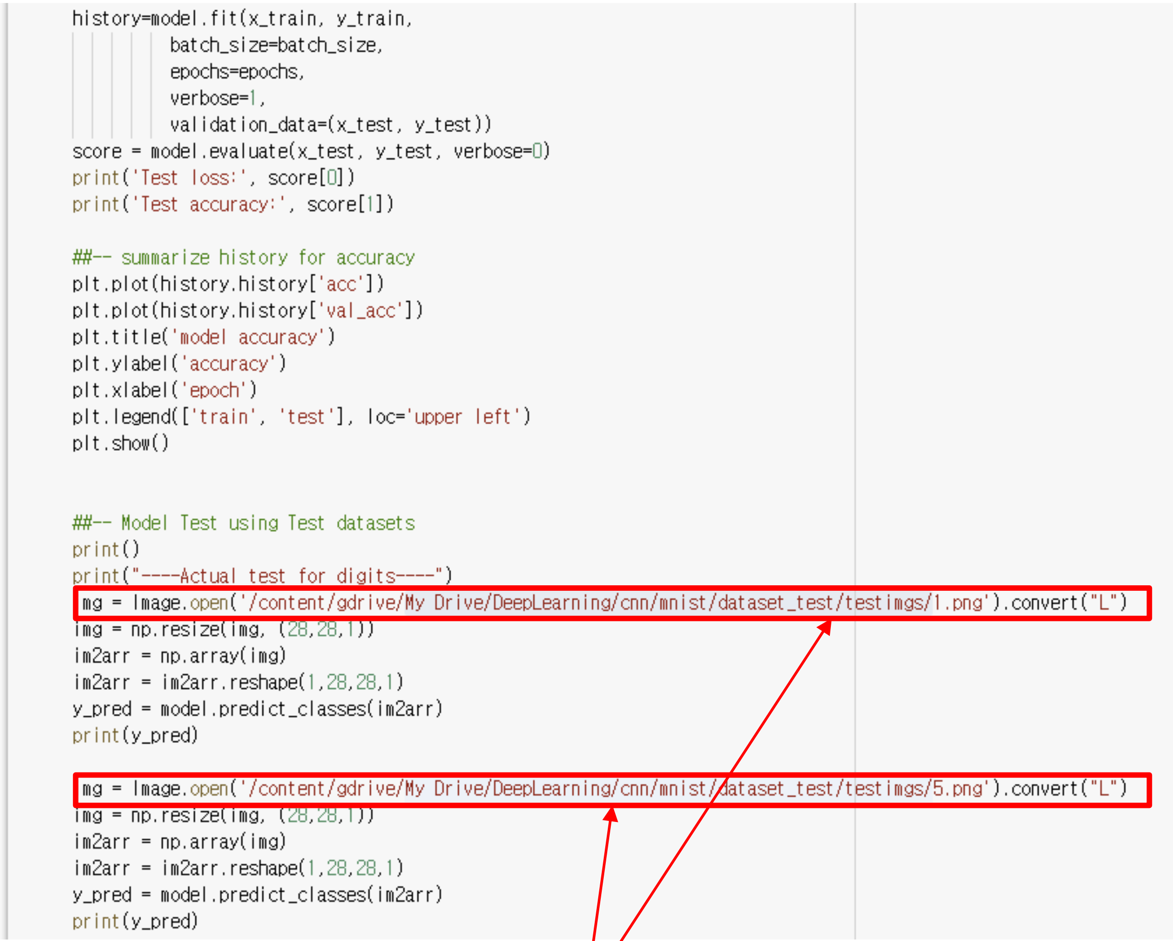
이름 ↓	소유자	마지막으로 수정한 날짜
210.png	나	오전 11:59 나
209.png	나	오전 11:59 나
208.png	나	오전 11:59 나
207.png	나	오전 11:59 나
206.png	나	오전 11:59 나
205.png	나	오전 11:59 나
204.png	나	오전 11:59 나
203.png	나	오전 11:59 나
202.png	나	오전 11:59 나
201.png	나	오전 11:59 나
200.png	나	오전 11:59 나
199.png	나	오전 11:59 나

Mnist : Digit recognition project (3)

❖ 3) mnist deep learning 코드를 준비한다.



```
'''Trains a simple convnet on the MNIST dataset.
Gets to 99.25% test accuracy after 12 epochs
(there is still a lot of margin for parameter tuning).
16 seconds per epoch on a GRID K520 GPU.
'''

from __future__ import print_function
import keras
import tensorflow.keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras import backend as K
from keras.utils import np_utils
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
import os
from tensorflow.python.client import device_lib

device_lib.list_local_devices()


batch_size = 128
num_classes = 10
epochs = 12

# input image dimensions
img_rows, img_cols = 28, 28

# the data, split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)

x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)


model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=tensorflow.keras.losses.categorical_crossentropy,
              optimizer="adam",
              metrics=['accuracy'])
```

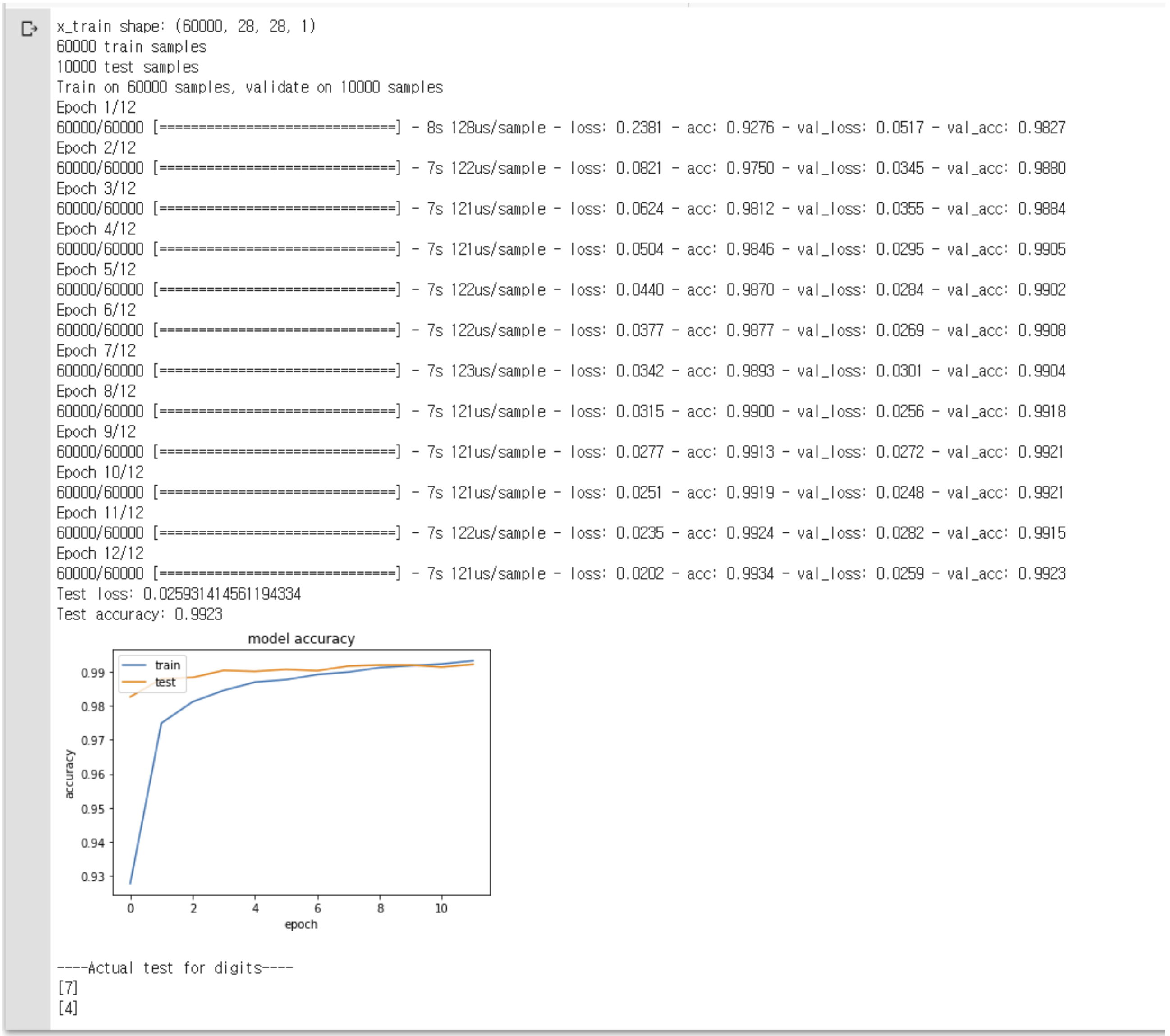


```
history=model.fit(x_train, y_train,
                  batch_size=batch_size,
                  epochs=epochs,
                  verbose=1,
                  validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

##-- summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()


##-- Model Test using Test datasets
print()
print("----Actual test for digits----")
img = Image.open('/content/gdrive/My Drive/DeepLearning/cnn/mnist/dataset_test/testimgs/1.png').convert("L")
img = np.resize(img, (28,28,1))
im2arr = np.array(img)
im2arr = im2arr.reshape(1,28,28,1)
y_pred = model.predict_classes(im2arr)
print(y_pred)

img = Image.open('/content/gdrive/My Drive/DeepLearning/cnn/mnist/dataset_test/testimgs/5.png').convert("L")
img = np.resize(img, (28,28,1))
im2arr = np.array(img)
im2arr = im2arr.reshape(1,28,28,1)
y_pred = model.predict_classes(im2arr)
print(y_pred)
```

코드 내에 내가 필요한 데이터 폴더를 정확히 명시하여 준다.

mnist : Digit recognition project (4)

❖ 4) mnist deep learning 코드 실행해 본다 (실행 결과 아래와 같음).



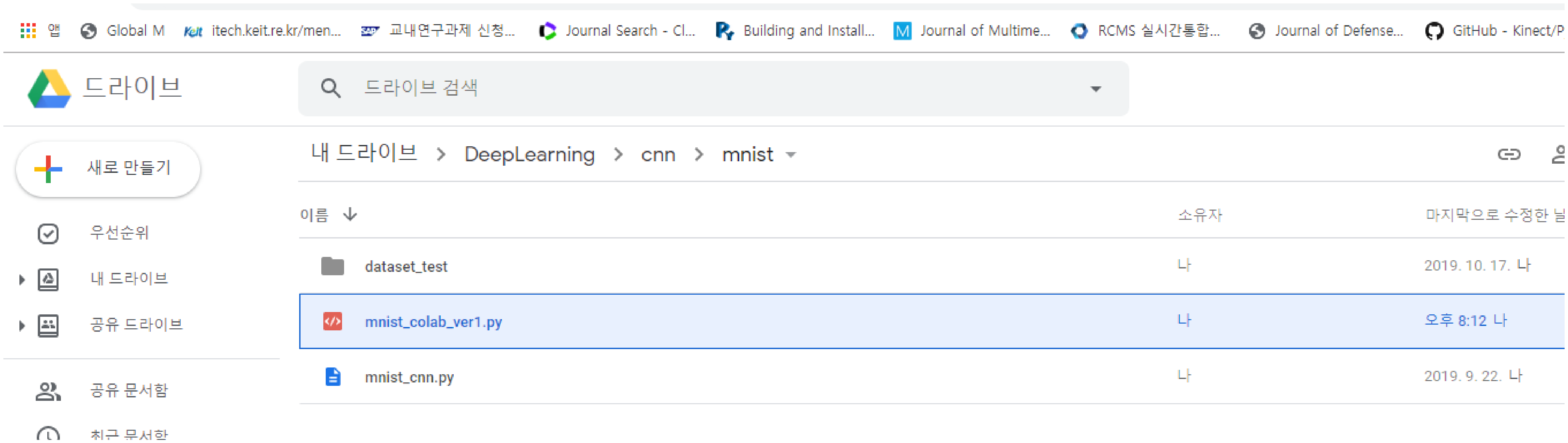
```
x_train shape: (60000, 28, 28, 1)
60000 train samples
10000 test samples
Train on 60000 samples, validate on 10000 samples
Epoch 1/12
60000/60000 [==============================] - 8s 128us/sample - loss: 0.2381 - acc: 0.9276 - val_loss: 0.0517 - val_acc: 0.9827
Epoch 2/12
60000/60000 [==============================] - 7s 122us/sample - loss: 0.0821 - acc: 0.9750 - val_loss: 0.0345 - val_acc: 0.9880
Epoch 3/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0624 - acc: 0.9812 - val_loss: 0.0355 - val_acc: 0.9884
Epoch 4/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0504 - acc: 0.9846 - val_loss: 0.0295 - val_acc: 0.9905
Epoch 5/12
60000/60000 [==============================] - 7s 122us/sample - loss: 0.0440 - acc: 0.9870 - val_loss: 0.0284 - val_acc: 0.9902
Epoch 6/12
60000/60000 [==============================] - 7s 122us/sample - loss: 0.0377 - acc: 0.9877 - val_loss: 0.0269 - val_acc: 0.9908
Epoch 7/12
60000/60000 [==============================] - 7s 123us/sample - loss: 0.0342 - acc: 0.9893 - val_loss: 0.0301 - val_acc: 0.9904
Epoch 8/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0315 - acc: 0.9900 - val_loss: 0.0256 - val_acc: 0.9918
Epoch 9/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0277 - acc: 0.9913 - val_loss: 0.0272 - val_acc: 0.9921
Epoch 10/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0251 - acc: 0.9919 - val_loss: 0.0248 - val_acc: 0.9921
Epoch 11/12
60000/60000 [==============================] - 7s 122us/sample - loss: 0.0235 - acc: 0.9924 - val_loss: 0.0282 - val_acc: 0.9915
Epoch 12/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0202 - acc: 0.9934 - val_loss: 0.0259 - val_acc: 0.9923
Test loss: 0.025931414561194334
Test accuracy: 0.9923
```

```
----Actual test for digits----
[7]
[4]
```

How to run the developed CNN code? (1)

❖ If already you have your own CNN code, how to run that python code?

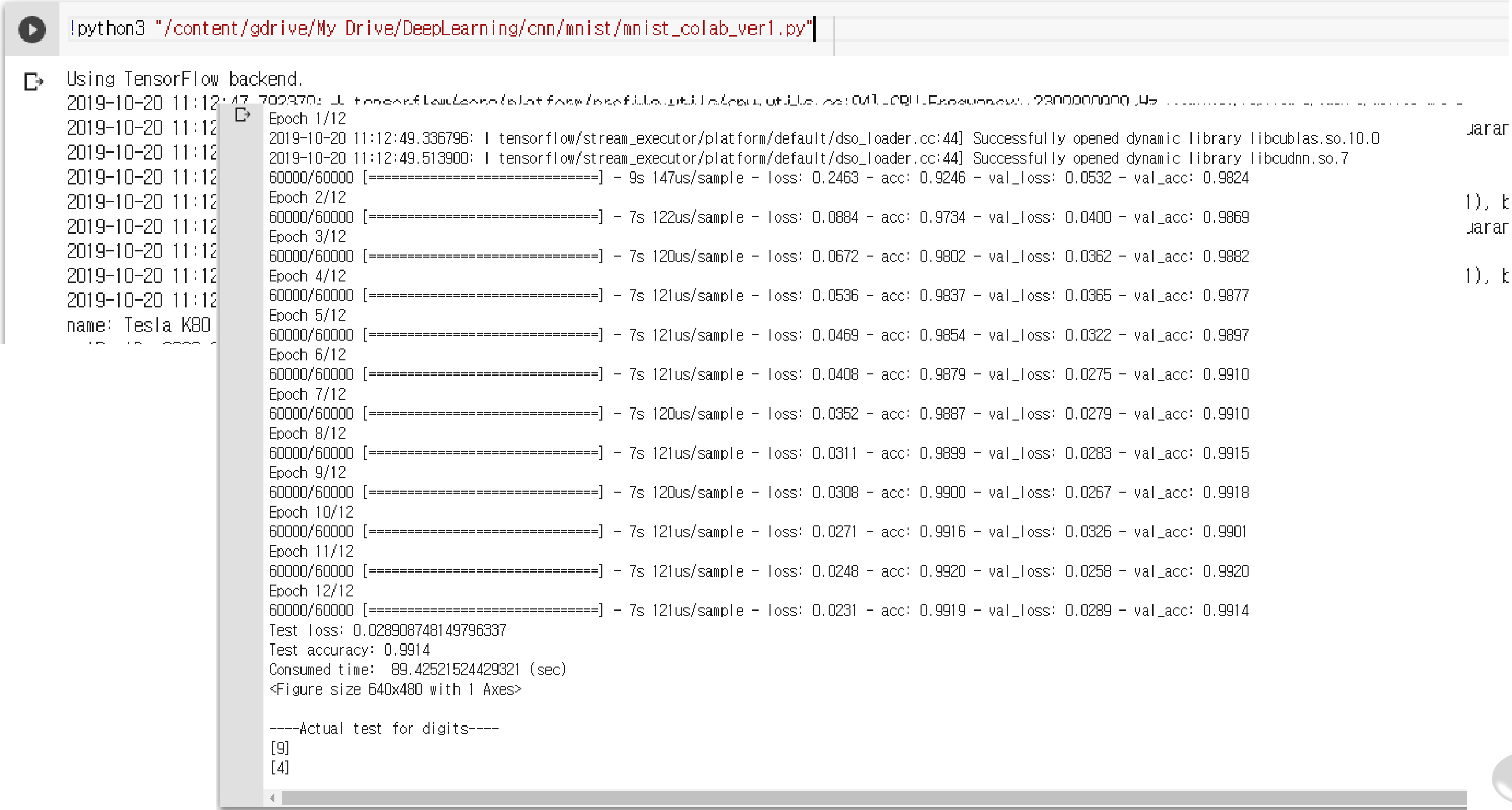
```
!python3 "/content/gdrive/My Drive/DeepLearning/cnn/mnist/mnist_colab_ver1.py"
```



이름 ↓	소유자	마지막으로 수정한 날
dataset_test	나	2019. 10. 17. 나
mnist_colab_ver1.py	나	오후 8:12 나
mnist_cnn.py	나	2019. 9. 22. 나

How to run the developed CNN code? (2)

- In your Jupyter NoteBook, the following, “**!python3 (your python code)**” and run.



```
!python3 "/content/gdrive/My Drive/DeepLearning/cnn/mnist/mnist_colab_ver1.py"
```

```
Using TensorFlow backend.
2019-10-20 11:12:47.792370: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 2300000000 Hz
2019-10-20 11:12
2019-10-20 11:12
2019-10-20 11:12
2019-10-20 11:12
2019-10-20 11:12
2019-10-20 11:12
2019-10-20 11:12
2019-10-20 11:12
name: Tesla K80
```

```
Epoch 1/12
2019-10-20 11:12:49.336796: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcublas.so.10.0
2019-10-20 11:12:49.513900: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcudnn.so.7
60000/60000 [==============================] - 9s 147us/sample - loss: 0.2463 - acc: 0.9246 - val_loss: 0.0532 - val_acc: 0.9824
Epoch 2/12
60000/60000 [==============================] - 7s 122us/sample - loss: 0.0884 - acc: 0.9734 - val_loss: 0.0400 - val_acc: 0.9869
Epoch 3/12
60000/60000 [==============================] - 7s 120us/sample - loss: 0.0672 - acc: 0.9802 - val_loss: 0.0362 - val_acc: 0.9882
Epoch 4/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0536 - acc: 0.9837 - val_loss: 0.0365 - val_acc: 0.9877
Epoch 5/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0469 - acc: 0.9854 - val_loss: 0.0322 - val_acc: 0.9897
Epoch 6/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0408 - acc: 0.9879 - val_loss: 0.0275 - val_acc: 0.9910
Epoch 7/12
60000/60000 [==============================] - 7s 120us/sample - loss: 0.0352 - acc: 0.9887 - val_loss: 0.0279 - val_acc: 0.9910
Epoch 8/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0311 - acc: 0.9899 - val_loss: 0.0283 - val_acc: 0.9915
Epoch 9/12
60000/60000 [==============================] - 7s 120us/sample - loss: 0.0308 - acc: 0.9900 - val_loss: 0.0267 - val_acc: 0.9918
Epoch 10/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0271 - acc: 0.9916 - val_loss: 0.0326 - val_acc: 0.9901
Epoch 11/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0248 - acc: 0.9920 - val_loss: 0.0258 - val_acc: 0.9920
Epoch 12/12
60000/60000 [==============================] - 7s 121us/sample - loss: 0.0231 - acc: 0.9919 - val_loss: 0.0289 - val_acc: 0.9914
Test loss: 0.028908748149796337
Test accuracy: 0.9914
Consumed time:  89.42521524429321 (sec)
<Figure size 640x480 with 1 Axes>

----Actual test for digits----
[9]
[4]
```

IVPL
Intelligent Vision Processing Lab

How to run the developed CNN code? (3)

- If you got the following syntax error when “**!python3 (your python code)**”:
  - **“from __future__ import print_function” should be in the first import line.** That is, all comments and some sentences should be removed in your python source file.

```
'''Trains a simple convnet on the MNIST dataset.
Gets to 99.25% test accuracy after 12 epochs
(there is still a lot of margin for parameter tu
ning).
16 seconds per epoch on a GRID K520 GPU.
'''
##-- google drive mounting to this project
#from google.colab import drive
#drive.mount('/content/gdrive')


from __future__ import print_function
import keras
import tensorflow.keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
      ( ~~~~~ )
```

```
from __future__ import print_function
import keras
import tensorflow.keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, F
latten
from tensorflow.keras.layers import Conv2D, MaxPoolin
g2D
from tensorflow.keras import backend as K
from keras.utils import np_utils
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
import os, time

                  ( ~~~~~~  )
```

Thank you for your attention.!!!
QnA

http://ivpl.sookmyung.ac.kr