

Keras

발표자 : 201532005 김도희

1. Keras란?
2. 설치과정
3. 향후계획
4. 참고자료

Keras란?

Keras의 개념

Keras: The Python Deep Learning library

You have just found Keras.

Keras is a high-level neural networks API, written in Python and capable of running on top of **TensorFlow**, **CNTK**, or **Theano**. It was developed with a focus on enabling fast experimentation. *Being able to go from idea to result with the least possible delay is key to doing good research.*

Use Keras if you need a deep learning library that:

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Runs seamlessly on CPU and GPU.

Read the documentation at **Keras.io**.

Keras is compatible with: **Python 2.7-3.5**.

출처 : <https://keras.io/>

Keras의 개념

- ① 파이썬으로 구현된 신경망 API(Application Programming Interface)
 - 현재 Python 2.7 – 3.5 에서 호환이 가능하다.
- ② Tensorflow, CNTK, Theano를 기반으로 운영된다
 - Tensorflow의 Wrapper라이브러리
- ③ 쉽게 다층퍼셉트론 모델, 컨볼루션 신경망 모델, 순환 신경망 모델 또는 이를 조합한 모델은 물론 다중 입력 또는 다중 출력 등 다양한 구성을 할 수 있습니다
- ④ 현재 Tensorflow에서 Keras가 사용가능하며, Keras에서도 당연히 tensorflow를 지원한다.
 - Tensorflow 내에서의 Keras 사용 : `tf.contrib.keras`
 - 최근 업데이트로 인해 tensorflow만의 데이터 형식인 `tfrecord`도 keras 라이브러리로 사용가능

Keras의 특징

케라스 주요 특징

케라스는 아래 4가지의 주요 특징을 가지고 있습니다.

- 모듈화 (Modularity)
 - 케라스에서 제공하는 모듈은 독립적이고 설정 가능하며, 가능한 최소한의 제약사항으로 서로 연결될 수 있습니다. 모델은 시퀀스 또는 그래프로 이러한 모듈들을 구성한 것입니다.
 - 특히 신경망 층, 비용함수, 최적화기, 초기화기법, 활성화함수, 정규화기법은 모두 독립적인 모듈이며, 새로운 모델을 만들기 위해 이러한 모듈을 조합할 수 있습니다.
- 최소주의 (Minimalism)
 - 각 모듈은 짧고 간결합니다.
 - 모든 코드는 한 번 훑어보는 것으로도 이해가능해야 합니다.
 - 단 반복 속도와 혁신성에는 다소 떨어질 수가 있습니다.
- 쉬운 확장성
 - 새로운 클래스나 함수로 모듈을 아주 쉽게 추가할 수 있습니다.
 - 따라서 고급 연구에 필요한 다양한 표현을 할 수 있습니다.
- 파이썬 기반
 - Caffe 처럼 별도의 모델 설정 파일이 필요없으며 파이썬 코드로 모델들이 정의됩니다.

Keras란?

Keras의 특징

- ① Keras의 데이터 구조는 '모델'
- ② 원하는 레이어를 쉽게 순차적으로 쌓아올리는 형식

케라스 기본 개념

- ③ 딥러닝 모델을 만드는 과정

케라스의 가장 핵심적인 데이터 구조는 바로 **모델**입니다. 케라스에서 제공하는 시퀀스 모델로 원하는 레이어를 쉽게 순차적으로 쌓을 수 있습니다. 다중 출력이 필요하는 등 좀 더 복잡한 모델을 구성하려면 케라스 함수 API를 사용하면 됩니다. 케라스로 딥러닝 모델을 만들 때는 다음과 같은 순서로 작성합니다. 다른 딥러닝 라이브러리와 비슷한 순서이지만 훨씬 직관적이고 간결합니다.

1. 데이터셋 생성하기

- 원본 데이터를 불러오거나 시뮬레이션을 통해 데이터를 생성합니다.
- 데이터로부터 훈련셋, 검증셋, 시험셋을 생성합니다.
- 이 때 딥러닝 모델의 학습 및 평가를 할 수 있도록 포맷 변환을 합니다.

2. 모델 구성하기

- 시퀀스 모델을 생성한 뒤 필요한 레이어를 추가하여 구성합니다.
- 좀 더 복잡한 모델이 필요할 때는 케라스 함수 API를 사용합니다.

3. 모델 학습과정 설정하기

- 학습하기 전에 학습에 대한 설정을 수행합니다.
- 손실 함수 및 최적화 방법을 정의합니다.
- 케라스에서는 `compile()` 함수를 사용합니다.

4. 모델 학습시키기

- 훈련셋을 이용하여 구성된 모델로 학습시킵니다.
- 케라스에서는 `fit()` 함수를 사용합니다.

5. 학습과정 살펴보기

- 모델 학습 시 훈련셋, 검증셋의 손실 및 정확도를 측정합니다.
- 반복횟수에 따른 손실 및 정확도 추이를 보면서 학습 상황을 판단합니다.

6. 모델 평가하기

- 준비된 시험셋으로 학습한 모델을 평가합니다.
- 케라스에서는 `evaluate()` 함수를 사용합니다.

7. 모델 사용하기

- 임의의 입력으로 모델의 출력을 얻습니다.
- 케라스에서는 `predict()` 함수를 사용합니다.

설치과정

사용가능한 OS 목록

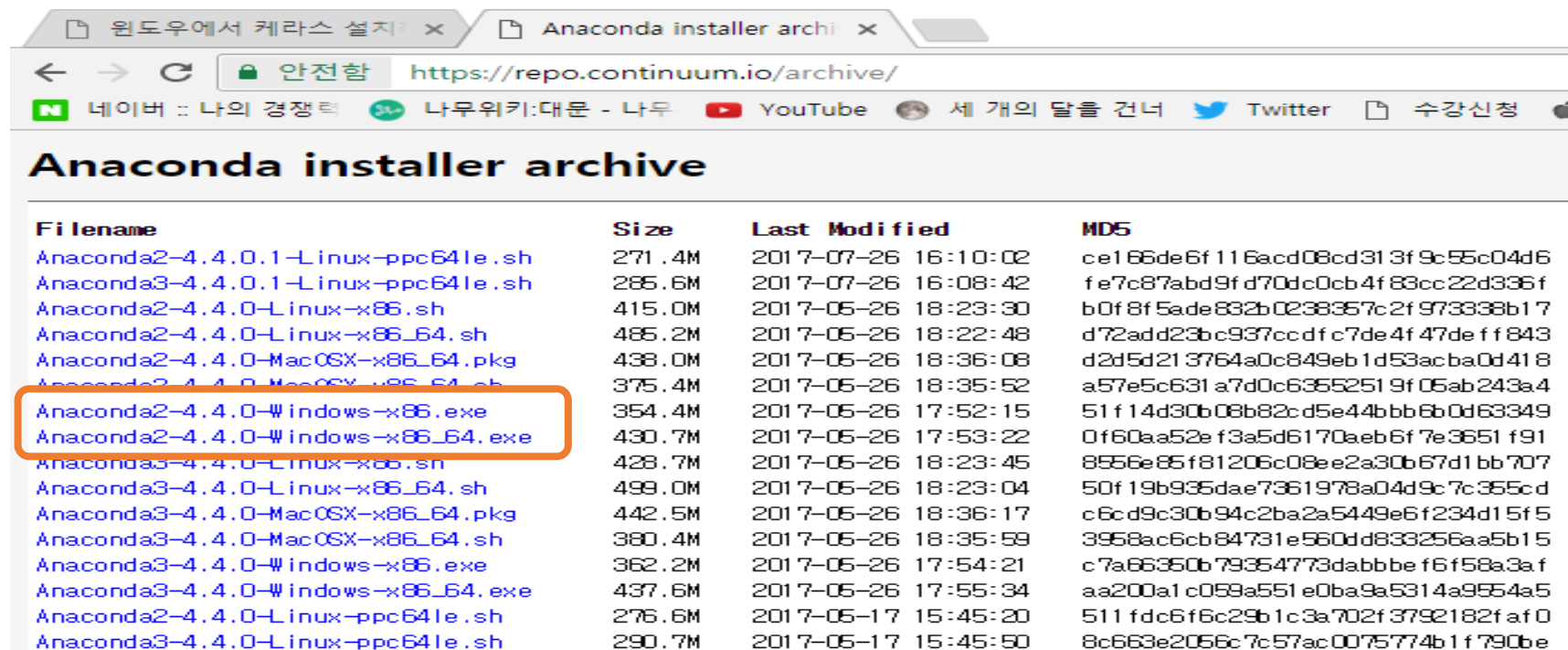
① Ubuntu

② Mac OS X

③ Windows (실제로 설치한 OS) →

에디션	Windows 10 Pro
버전	1703
OS 빌드	15063.540
제품 ID	00331-10000-00001-AA653
프로세서	Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz 2.40 GHz
설치된 RAM	4.00GB
시스템 종류	64비트 운영 체제, x64 기반 프로세서

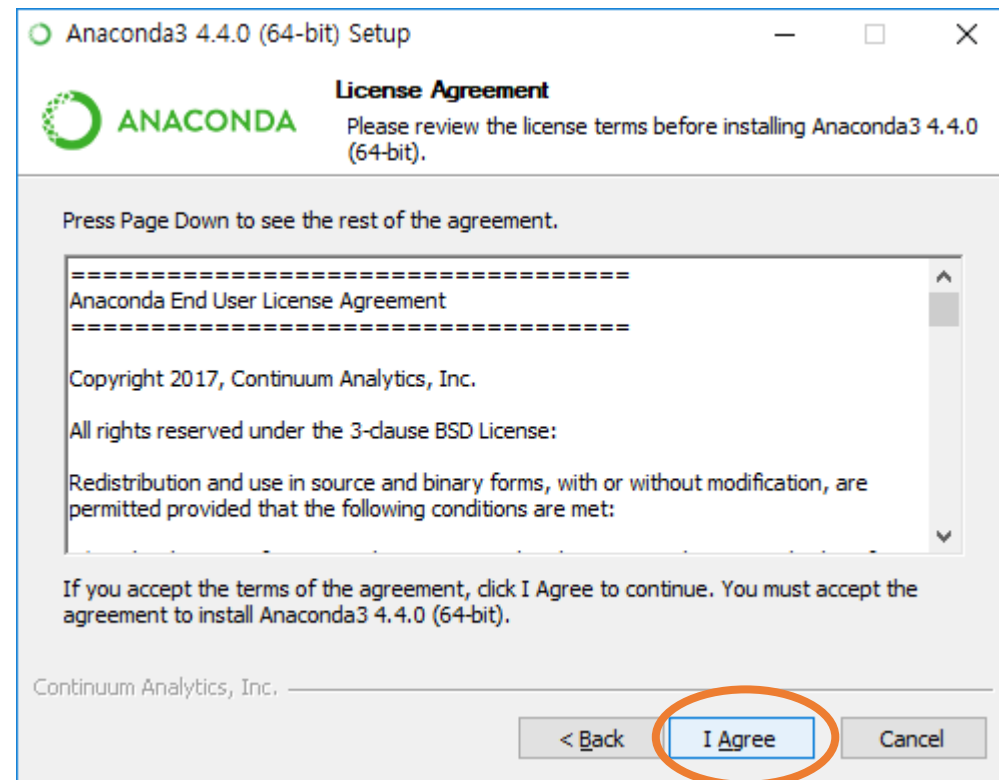
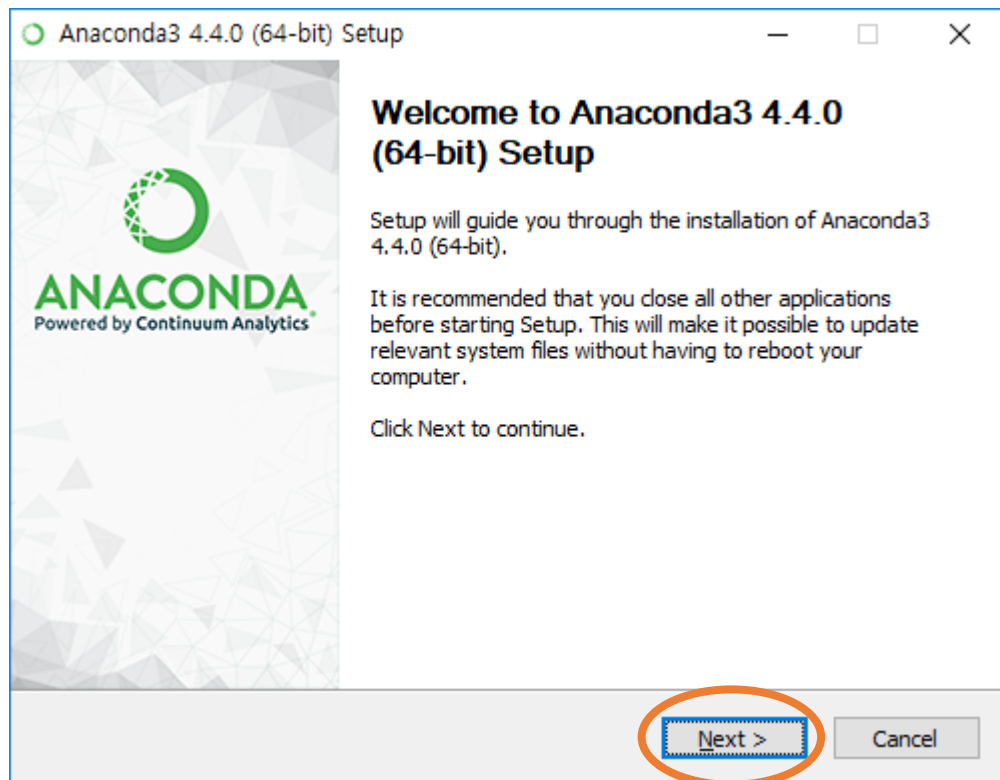
아나콘다 설치



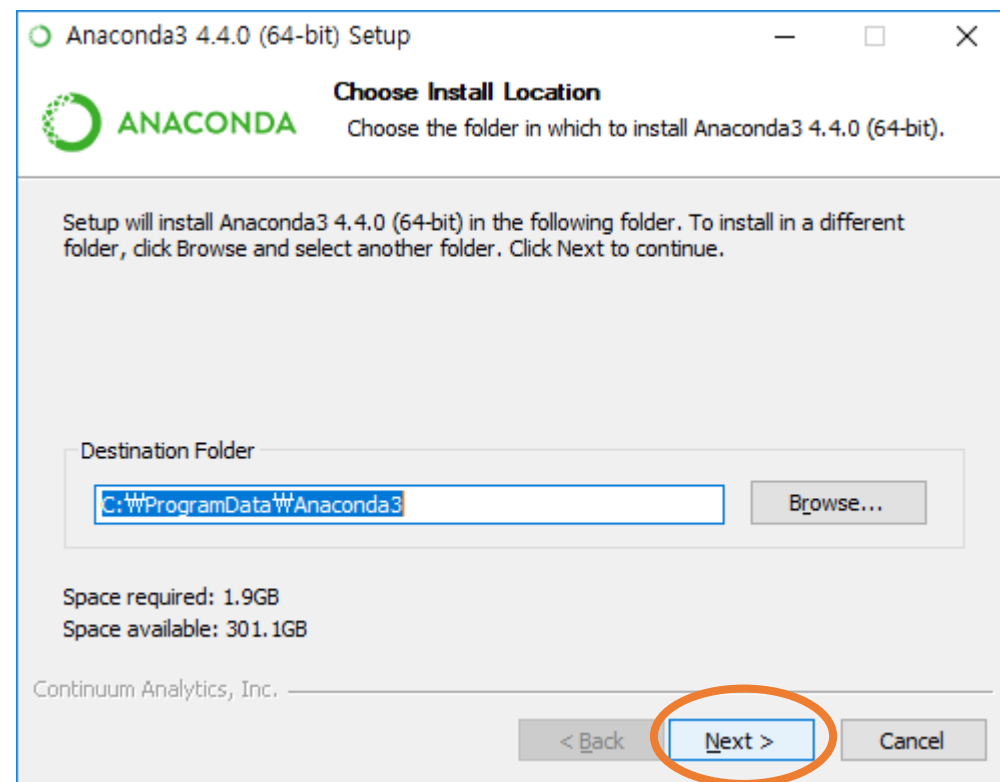
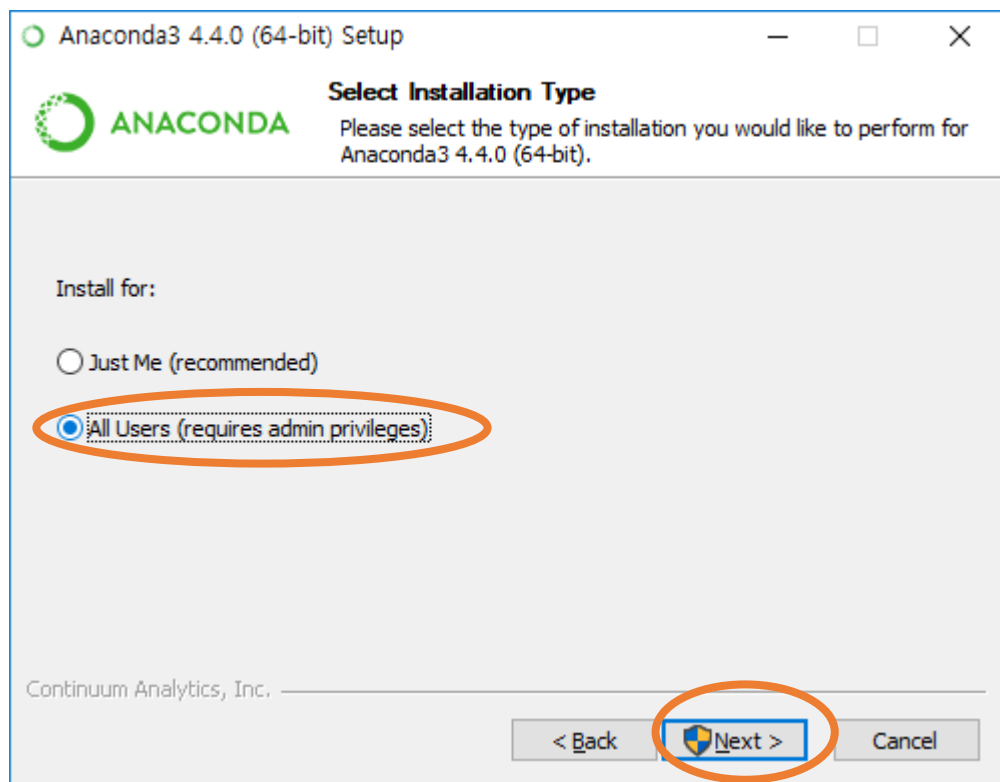
Filename	Size	Last Modified	MD5
Anaconda2-4.4.0.1-Linux-ppc64le.sh	271.4M	2017-07-26 16:10:02	ce166de6f116acd08cd313f9c55c04d6
Anaconda3-4.4.0.1-Linux-ppc64le.sh	285.6M	2017-07-26 16:08:42	fe7c87abd9fd70dc0cb4f83cc22d336f
Anaconda2-4.4.0-Linux-x86.sh	415.0M	2017-05-26 18:23:30	b0f8f5ade832b0238357c2f973338b17
Anaconda2-4.4.0-Linux-x86_64.sh	485.2M	2017-05-26 18:22:48	d72add23bc937ccdfc7de4f47deff843
Anaconda2-4.4.0-MacOSX-x86_64.pkg	438.0M	2017-05-26 18:36:08	d2d5d213764a0c849eb1d53acba0d418
Anaconda2-4.4.0-MacOSX-x86_64.sh	375.4M	2017-05-26 18:35:52	a57e5c631a7d0c63552519f05ab243a4
Anaconda2-4.4.0-Windows-x86.exe	354.4M	2017-05-26 17:52:15	51f14d30b08b82cd5e44bbb6b0d63349
Anaconda2-4.4.0-Windows-x86_64.exe	430.7M	2017-05-26 17:53:22	0f60aa52ef3a5d6170aeb6f7e3651f91
Anaconda3-4.4.0-Linux-x86.sh	428.7M	2017-05-26 18:23:45	8556e85f81206c08ee2a30b67d1bb707
Anaconda3-4.4.0-Linux-x86_64.sh	499.0M	2017-05-26 18:23:04	50f19b935dae7361978a04d9c7c355cd
Anaconda3-4.4.0-MacOSX-x86_64.pkg	442.5M	2017-05-26 18:36:17	c6cd9c30b94c2ba2a5449e6f234d15f5
Anaconda3-4.4.0-MacOSX-x86_64.sh	380.4M	2017-05-26 18:35:59	3958ac6cb84731e560dd833256aa5b15
Anaconda3-4.4.0-Windows-x86.exe	362.2M	2017-05-26 17:54:21	c7a66350b79354773dabbbe6f6f58a3af
Anaconda3-4.4.0-Windows-x86_64.exe	437.6M	2017-05-26 17:55:34	aa200a1c059a551e0ba9a5314a9554a5
Anaconda2-4.4.0-Linux-ppc64le.sh	276.6M	2017-05-17 15:45:20	511fdc6f6c29b1c3a702f3792182faf0
Anaconda3-4.4.0-Linux-ppc64le.sh	290.7M	2017-05-17 15:45:50	8c663e2056c7c57ac0075774b1f790be

- ① <https://repo.continuum.io/archive/> 에 들어가서 자신의 os와 맞는 아나콘다 설치 프로그램 다운로드

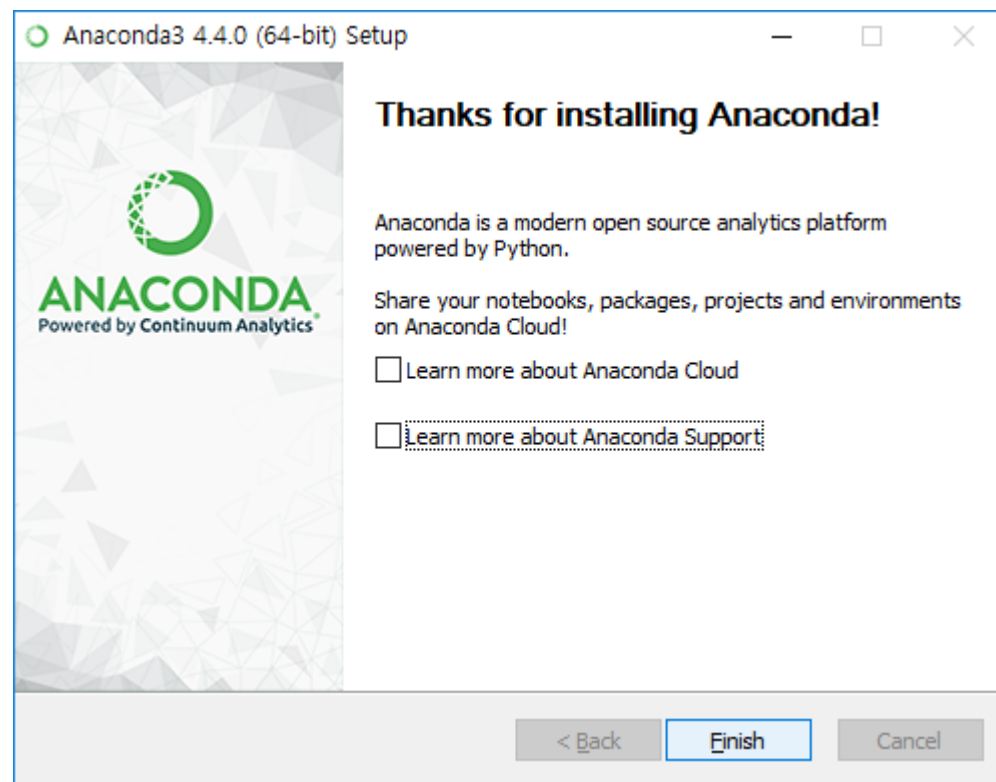
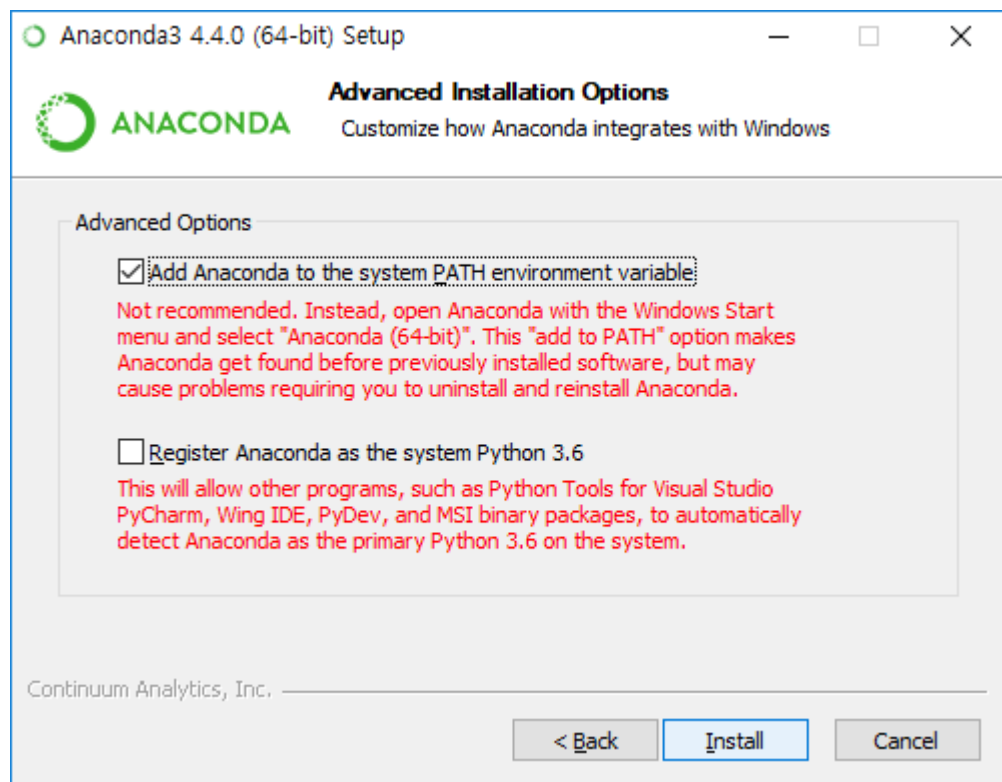
아나콘다 설치



아나콘다 설치

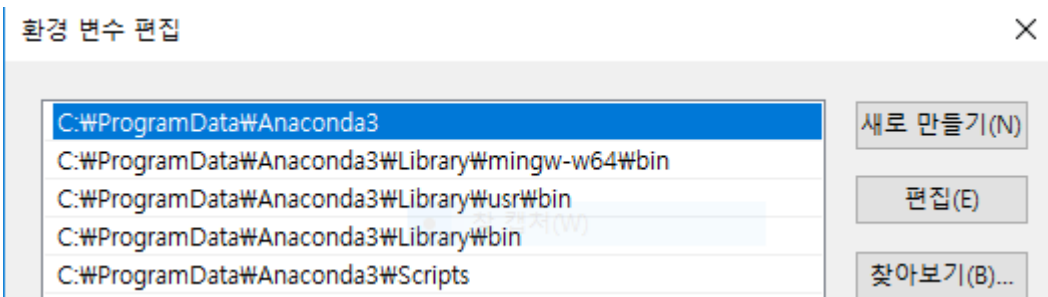


아나콘다 설치



아나콘다 설치

✓ 설치 완료 후 반드시 **PATH 경로**를 설정해야 한다.



- ✓ 제어판 > 시스템 및 보안 > 시스템 > 고급 시스템 설정 > 환경 변수에서 [시스템변수] 중 Path 에 아래 경로들을 추가
- ✓ [추가할 경로]

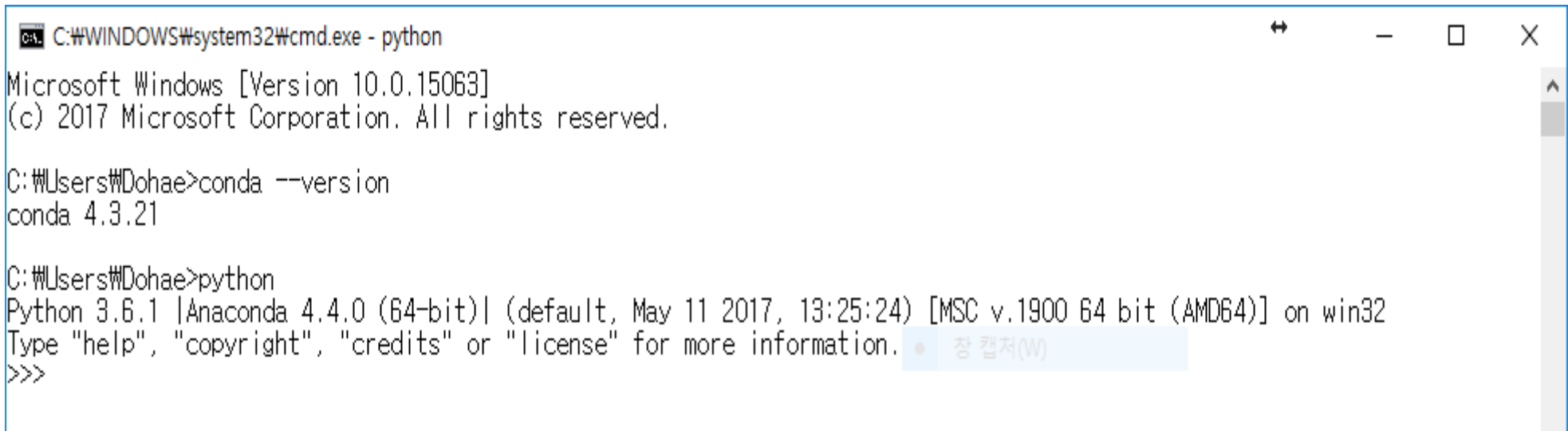
C:\ProgramData\Anaconda3

C:\ProgramData\Anaconda3\Scripts

C:\ProgramData\Anaconda3\Library\bin

아나콘다 설치

✓설치 완료 확인(프롬프트 환경)



```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Dohae>conda --version
conda 4.3.21

C:\Users\Dohae>python
Python 3.6.1 |Anaconda 4.4.0 (64-bit)| (default, May 11 2017, 13:25:24) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

conda --version [enter]

python [enter]

프로젝트 디렉토리 생성

- 여기서부터는 명령프롬프트 사용시 관리자 권한으로 실행

- 1) `cd c:\`
- 2) `mkdir Projects` (실습용 프로젝트 폴더 생성)
- 3) `cd Projects`
- 4) `mkdir keras_talk`
- 5) `cd keras_talk`

관리자: 명령 프롬프트

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd c:\Projects\keras_talk

c:\Projects\keras_talk>
```


가상 개발환경 생성

1) `conda create -n venv python=3.5 anaconda` (가상환경 생성)

- I. venv : 가상환경 이름
- II. 사용 할 python 버전

2) `activate venv` (가상환경 활성화)

C:\ 관리자: 명령 프롬프트

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd c:\Projects\keras_talk

c:\Projects\keras_talk>conda create -n venv python=3.5 anaconda

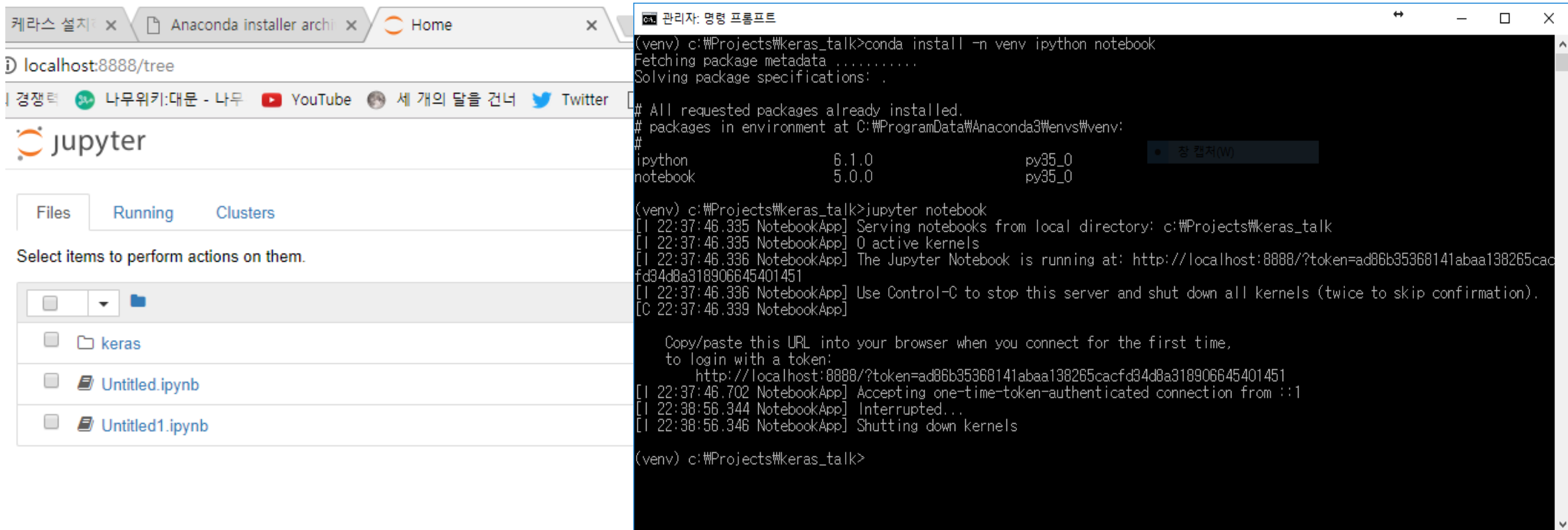
CondaValueError: prefix already exists: C:\ProgramData\Anaconda3\envs\venv

c:\Projects\keras_talk>activate venv

(venv) c:\Projects\keras_talk>
```

주피터 노트북 설치

- 1) `conda install -n venv ipython notebook`
- 2) `jupyter notebook` -> 종료는 `ctrl + C`



```
(venv) c:\Projects\keras_talk>conda install -n venv ipython notebook
Fetching package metadata .....
Solving package specifications: .

# All requested packages already installed.
# packages in environment at C:\ProgramData\Anaconda3\envs\venv:
#
ipython                6.1.0                  py35_0
notebook               5.0.0                  py35_0

(venv) c:\Projects\keras_talk>jupyter notebook
[I 22:37:46.335 NotebookApp] Serving notebooks from local directory: c:\Projects\keras_talk
[I 22:37:46.335 NotebookApp] 0 active kernels
[I 22:37:46.336 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/?token=ad86b35368141abaa138265cacf...
[I 22:37:46.336 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 22:37:46.339 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:8888/?token=ad86b35368141abaa138265cacf...
[I 22:37:46.702 NotebookApp] Accepting one-time-token-authenticated connection from ::1
[I 22:38:56.344 NotebookApp] Interrupted...
[I 22:38:56.346 NotebookApp] Shutting down kernels

(venv) c:\Projects\keras_talk>
```

주요 패키지 설치

- 1) `conda install -n venv numpy matplotlib pandas pydotplus h5py scikit-learn`
- 2) `conda install -n venv scipy mkl-service libpython m2w64-toolchain`

```
관리자: 명령 프롬프트

(venv) c:\Projects\Keras_talk>conda install -n venv numpy matplotlib pandas pydotplus h5py scikit-learn
Fetching package metadata .....
Solving package specifications: .

# All requested packages already installed.
# packages in environment at C:\ProgramData\Anaconda3\envs\venv:
#
h5py                2.7.0                np113py35_0
matplotlib          2.0.2                np113py35_0
numpy               1.13.1               py35_0
pandas              0.20.3               py35_0
pydotplus           2.0.2                py35_0
scikit-learn        0.19.0               np113py35_0

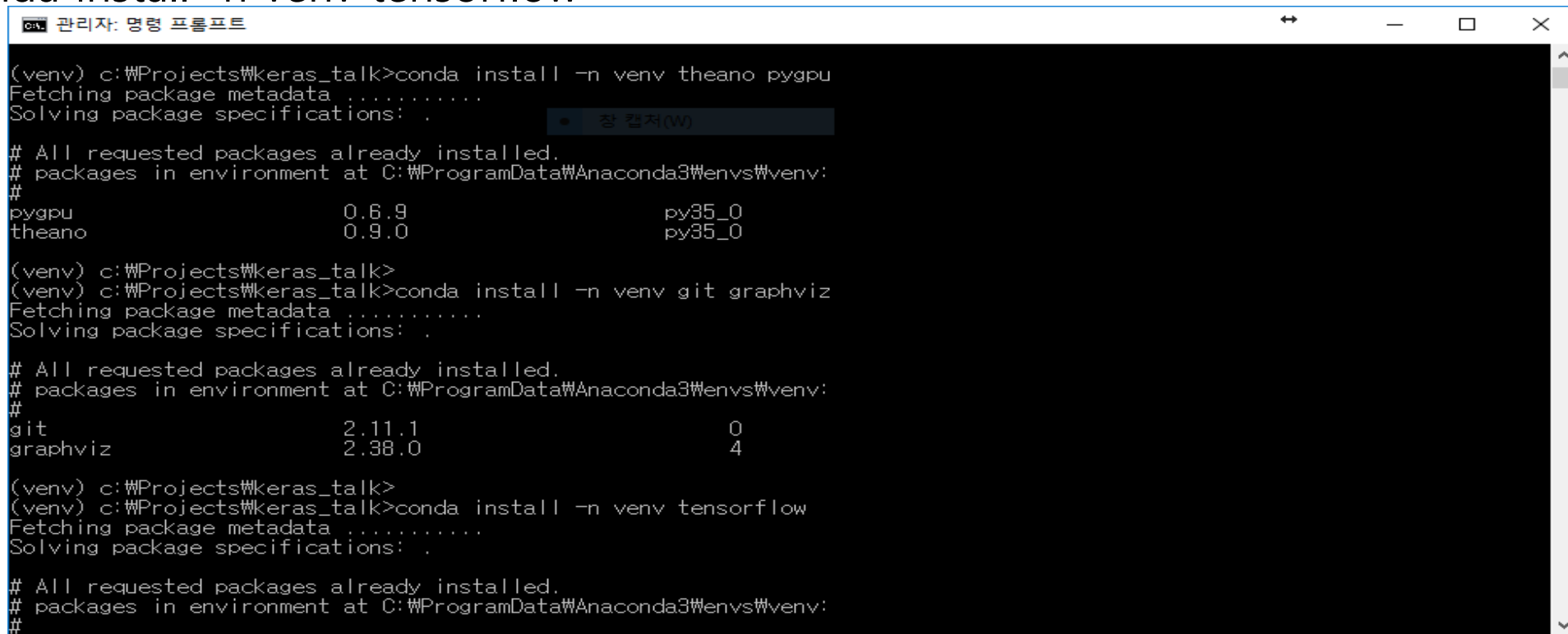
(venv) c:\Projects\Keras_talk>conda install -n venv scipy mkl-service libpython m2w64-toolchain
Fetching package metadata .....
Solving package specifications: .

# All requested packages already installed.
# packages in environment at C:\ProgramData\Anaconda3\envs\venv:
#
libpython           2.0                  py35_0
m2w64-toolchain     5.3.0                7
mkl-service         1.1.2                py35_3
scipy               0.19.1               np113py35_0

(venv) c:\Projects\Keras_talk>
```

딥러닝 라이브러리 설치

- 1) `conda install -n venv theano pygpu`
- 2) `conda install -n venv git graphviz`
- 3) `conda install -n venv tensorflow`



```
관리자: 명령 프롬프트

(venv) c:\w\Projects\keras_talk>conda install -n venv theano pygpu
Fetching package metadata .....
Solving package specifications: .

# All requested packages already installed.
# packages in environment at C:\w\ProgramData\Anaconda3\envs\venv:
#
pygpu                0.6.9                py35_0
theano               0.9.0                py35_0

(venv) c:\w\Projects\keras_talk>
(venv) c:\w\Projects\keras_talk>conda install -n venv git graphviz
Fetching package metadata .....
Solving package specifications: .

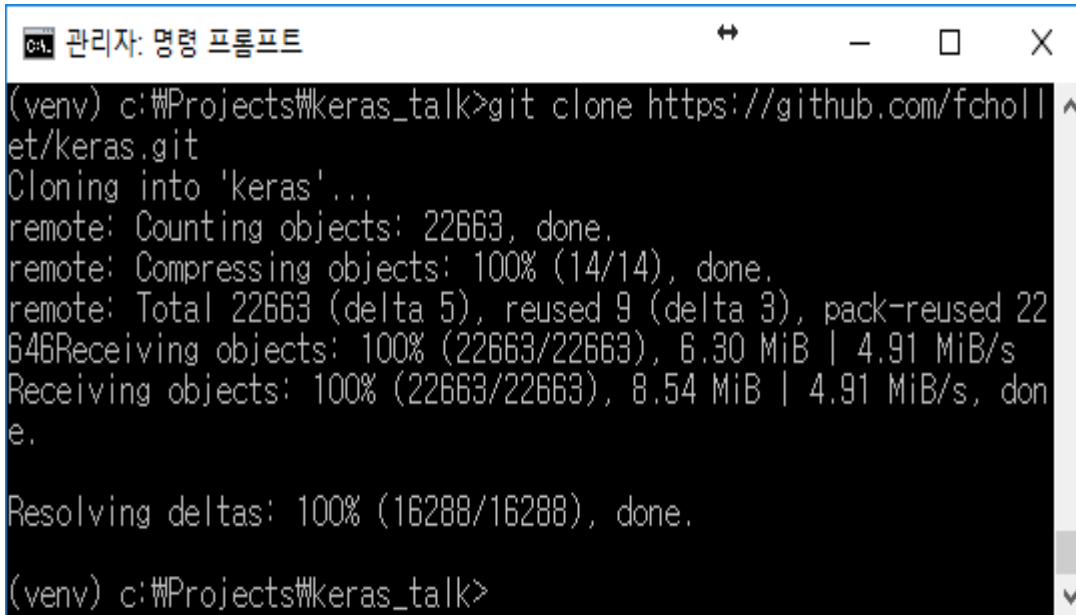
# All requested packages already installed.
# packages in environment at C:\w\ProgramData\Anaconda3\envs\venv:
#
git                  2.11.1                0
graphviz             2.38.0                4

(venv) c:\w\Projects\keras_talk>
(venv) c:\w\Projects\keras_talk>conda install -n venv tensorflow
Fetching package metadata .....
Solving package specifications: .

# All requested packages already installed.
# packages in environment at C:\w\ProgramData\Anaconda3\envs\venv:
#
```

딥러닝 라이브러리 설치

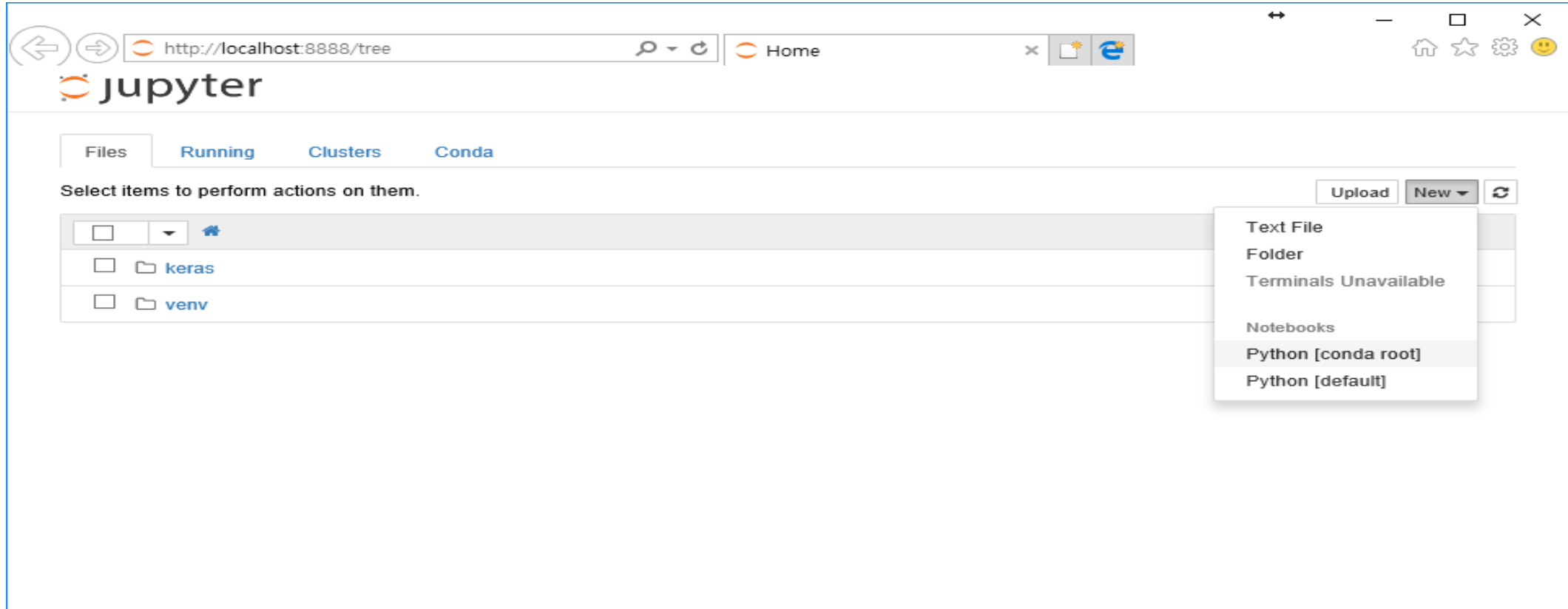
- 4) `git clone https://github.com/fchollet/keras.git` `conda install -n venv git graphviz`
- 5) `cd keras`
- 6) `python setup.py install` (Keras 설치)



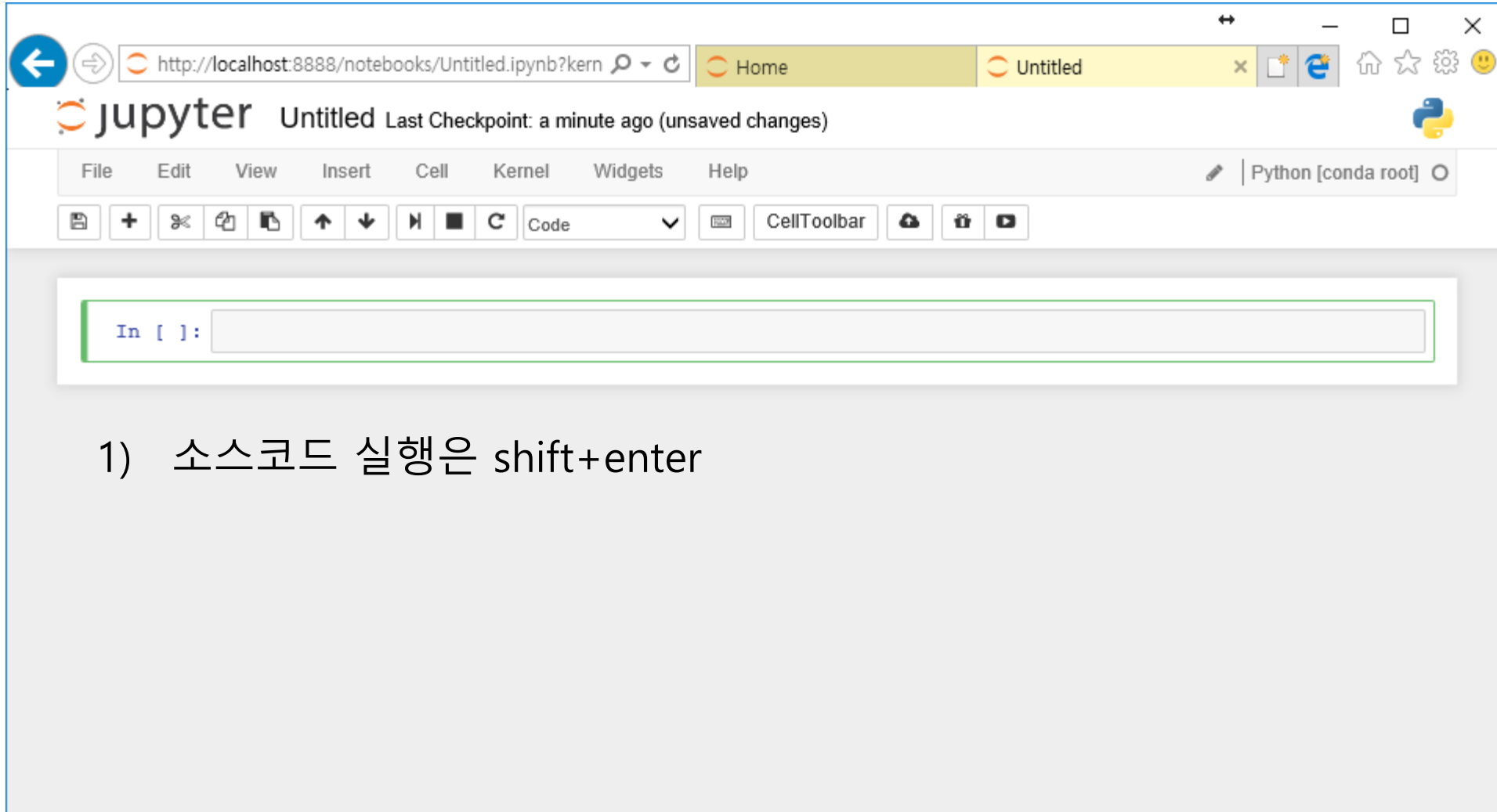
```
관리자: 명령 프롬프트
(venv) c:\Projects\keras_talk>git clone https://github.com/fchollet/keras.git
Cloning into 'keras'...
remote: Counting objects: 22663, done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 22663 (delta 5), reused 9 (delta 3), pack-reused 22646
Receiving objects: 100% (22663/22663), 6.30 MiB | 4.91 MiB/s
Receiving objects: 100% (22663/22663), 8.54 MiB | 4.91 MiB/s, done.
Resolving deltas: 100% (16288/16288), done.
(venv) c:\Projects\keras_talk>
```

설치 환경 테스트

1) 프로젝트 폴더(c:\Projects\keras_talk) 이동 후 "jupyter notebook"실행



설치 환경 테스트



1) 소스코드 실행은 shift+enter

설치 환경 테스트

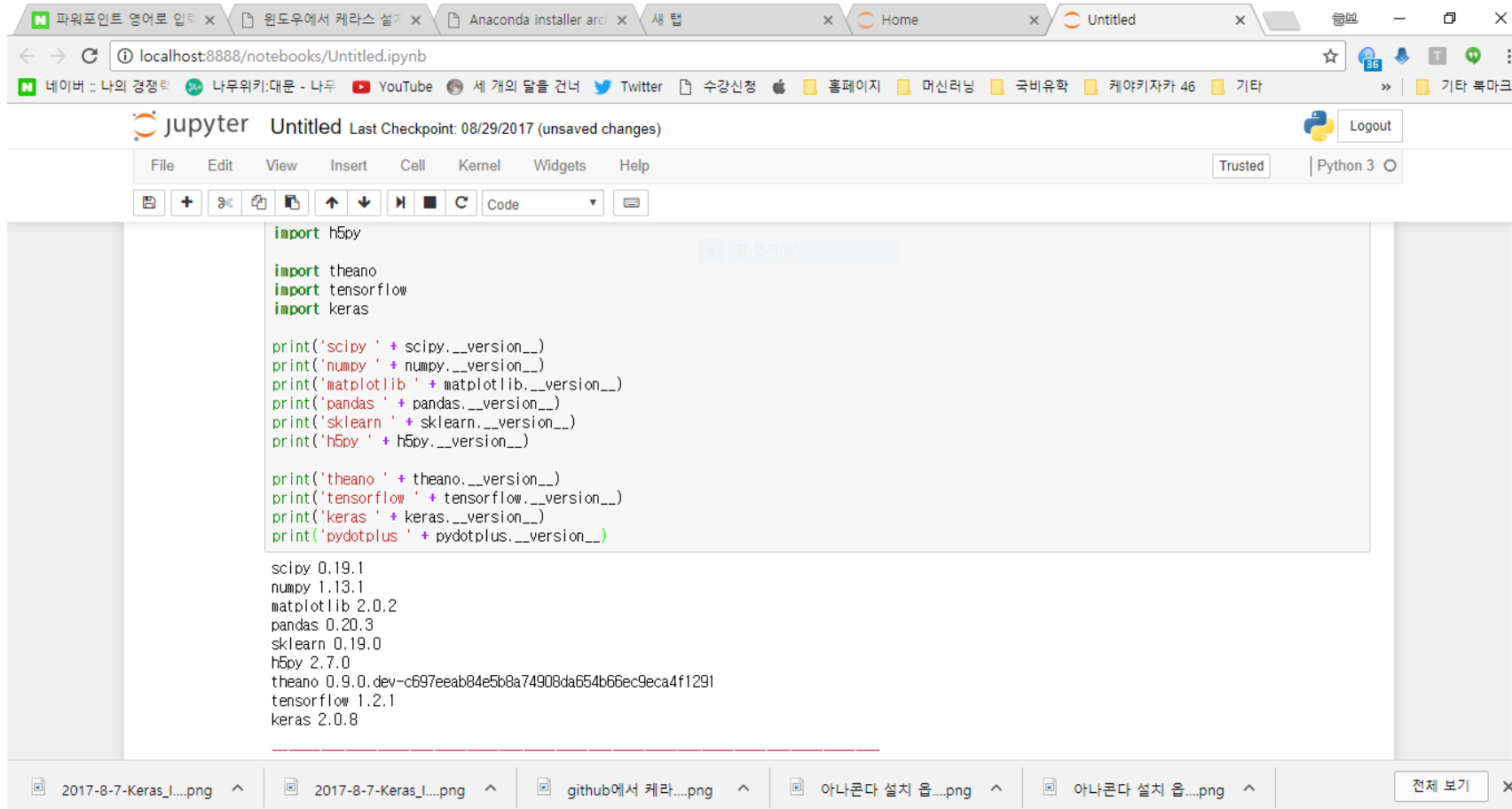
```
import scipy
import numpy
import matplotlib
import pandas
import sklearn
import pydotplus
import h5py

import theano
import tensorflow
import keras
```

```
print('scipy ' + scipy.__version__)
print('numpy ' + numpy.__version__)
print('matplotlib ' + matplotlib.__version__)
print('pandas ' + pandas.__version__)
print('sklearn ' + sklearn.__version__)
print('pydotplus ' + pydotplus.__version__)
print('h5py ' + h5py.__version__)

print('theano ' + theano.__version__)
print('tensorflow ' + tensorflow.__version__)
print('keras ' + keras.__version__)
```


설치환경 테스트 및 구동확인



```
import h5py

import theano
import tensorflow
import keras

print('scipy ' + scipy.__version__)
print('numpy ' + numpy.__version__)
print('matplotlib ' + matplotlib.__version__)
print('pandas ' + pandas.__version__)
print('sklearn ' + sklearn.__version__)
print('h5py ' + h5py.__version__)

print('theano ' + theano.__version__)
print('tensorflow ' + tensorflow.__version__)
print('keras ' + keras.__version__)
print('pydotplus ' + pydotplus.__version__)

scipy 0.19.1
numpy 1.13.1
matplotlib 2.0.2
pandas 0.20.3
sklearn 0.19.0
h5py 2.7.0
theano 0.9.0.dev-c697eeab84e5b8a74908da654b66ec9eca4f1291
tensorflow 1.2.1
keras 2.0.8
```

딥러닝 기본 모델 구현 확인

```
from keras.utils import np_utils

from keras.datasets import mnist

from keras.models import Sequential

from keras.layers import Dense, Activation
```

```
(X_train, Y_train), (X_test, Y_test) = mnist.load_data()

X_train = X_train.reshape(60000, 784).astype('float32') / 255.0

X_test = X_test.reshape(10000, 784).astype('float32') / 255.0

Y_train = np_utils.to_categorical(Y_train)

Y_test = np_utils.to_categorical(Y_test)
```

```
model = Sequential()

model.add(Dense(units=64, input_dim=28*28, activation='relu'))

model.add(Dense(units=10, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='sgd', metrics=['accuracy'])

model.fit(X_train, Y_train, epochs=5, batch_size=32)
```

```
loss_and_metrics = model.evaluate(X_test, Y_test, batch_size=32)
```

```
(X_train, Y_train), (X_test, Y_test) = mnist.load_data()
X_train = X_train.reshape(60000, 784).astype('float32') / 255.0
X_test = X_test.reshape(10000, 784).astype('float32') / 255.0
Y_train = np_utils.to_categorical(Y_train)
Y_test = np_utils.to_categorical(Y_test)

model = Sequential()
model.add(Dense(units=64, input_dim=28*28, activation='relu'))
model.add(Dense(units=10, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='sgd', metrics=['accuracy'])
model.fit(X_train, Y_train, epochs=5, batch_size=32)

loss_and_metrics = model.evaluate(X_test, Y_test, batch_size=32)

print('loss_and_metrics : ' + str(loss_and_metrics))
```

```
Epoch 1/5
60000/60000 [=====] - 3s - loss: 0.6828 - acc: 0.8233
Epoch 2/5
60000/60000 [=====] - 3s - loss: 0.3477 - acc: 0.9029
Epoch 3/5
60000/60000 [=====] - 3s - loss: 0.2994 - acc: 0.9148
Epoch 4/5
60000/60000 [=====] - 3s - loss: 0.2696 - acc: 0.9237
Epoch 5/5
60000/60000 [=====] - 3s - loss: 0.2458 - acc: 0.9311 - ETA: 2s - loss: 0.2559 - - ETA: 1s - loss: 0.2516 - ac
c: 0.929 - ETA: 1s
9760/10000 [=====>.] - ETA: 0sloss_and_metrics : [0.22699120199680328, 0.9356999999999999]
```

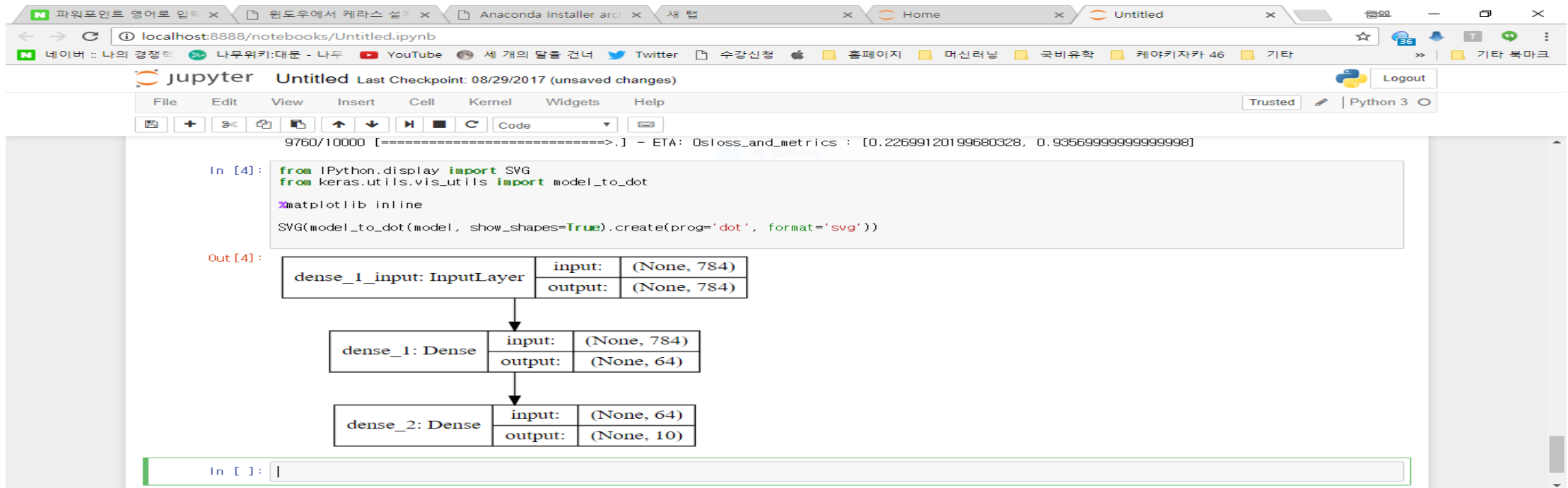
딥러닝 모델 가시화 확인

```
from IPython.display import SVG
```

```
from keras.utils.vis_utils import model_to_dot
```

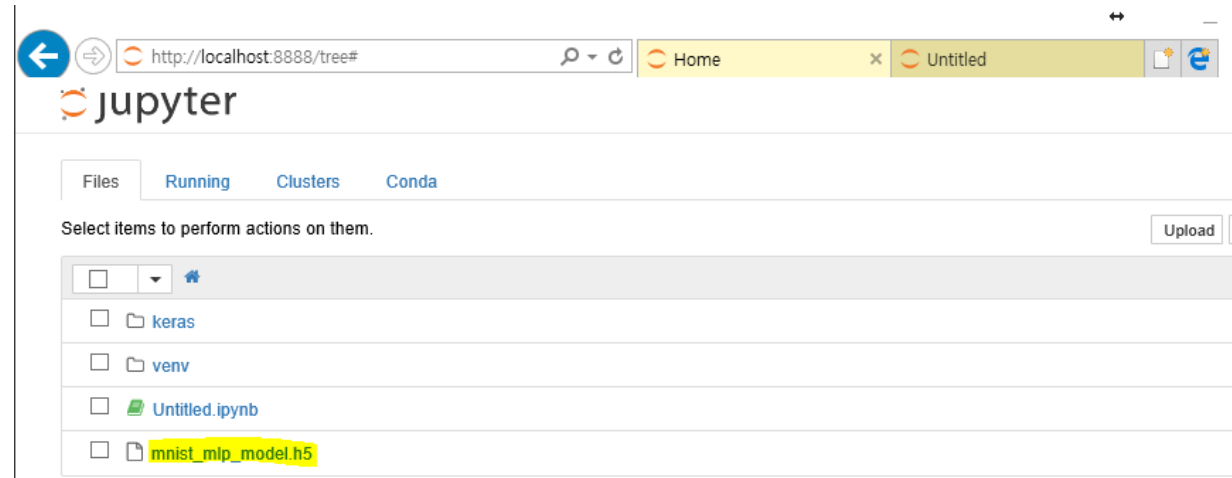
```
%matplotlib inline
```

```
SVG(model_to_dot(model, show_shapes=True).create(prog='dot', format='svg'))
```



딥러닝 모델 저장 및 엔진 바꾸기

```
from keras.models import load_model  
model.save('mnist_mlp_model.h5')  
model = load_model('mnist_mlp_model.h5')
```



‘C:/Users/사용자이름/.keras/keras.json’에서
‘backend’부분을 수정하면 딥러닝 엔진을 바꿀 수 있다.

향후계획

① <https://github.com/fchollet/keras>

- 예제 소스 분석

② <https://tykimos.github.io/Keras/lecture/>

- 강의 차근차근 따라가기

③



- 책 참고

④ https://tykimos.github.io/Keras/2017/08/09/DeepBrick_Talk/

- Keras의 기본 데이터 단위에 대한 이해

참고자료

참고자료

- ① <https://keras.io/> (Keras 공식)
- ② <https://blog.keras.io/> (Keras 공식)
- ③ <https://tykimos.github.io/Keras/lecture/> (현재 Keras강의를 진행 중)
- ④ <http://www.modulabs.co.kr/> (Keras외에도 다양한 인공지능 자료)
- ⑤ <https://github.com/fchollet/keras> (①이 운영하는 github, 다양한 예제 존재)
- ⑥ <https://www.facebook.com/groups/KerasKorea/> (③의 사람이 운영하는 페이스북 그룹)
- ⑦ <https://www.facebook.com/groups/keras.py/> (Keras및 인공지능 자료)
- ⑧ <http://blog.daum.net/goodgodgd/22>
- ⑨ <http://iostream.tistory.com/category/%EA%B0%9C%EB%B0%9C%20%EC%9D%B4%EC%95%BC%EA%B8%B0/Machine%20learning>