

고급프로그래밍및실습

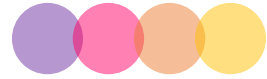
/ 16. 기계 학습 2

-

이 정 진

조교수, 숭실대 글로벌미디어학부

jungjinlee@ssu.ac.kr, 정보과학관 623호



기계 학습 (Machine Learning)

Summary

글로벌미디어학부 <고급프로그래밍및실습>, 이정진



Keywords

기계학습 / 선형 회귀 / 인공신경망

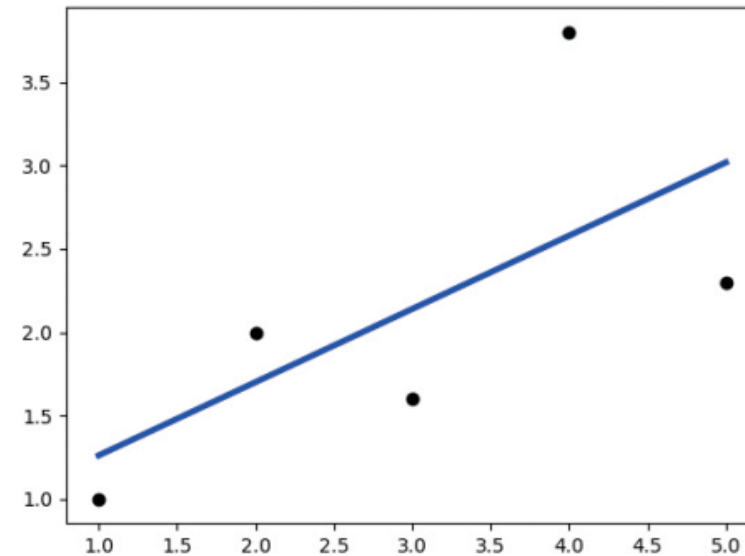
- 모델
- 손실 함수 (Loss Function)
- 경사하강법 (Gradient Descent Algorithm)
- 학습 (Training)



Linear Regression

- 모델
- 손실 함수 (Loss Function)
- 경사하강법 (Gradient Descent Algorithm)
- 학습 (Training)

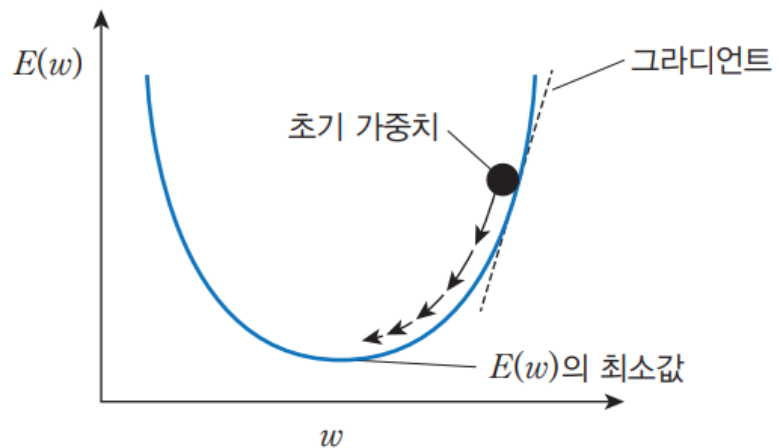
X	y
1.0	1.0
2.0	2.0
3.0	1.6
4.0	3.8
5.0	2.3





Linear Regression

- 모델
- 손실 함수 (Loss Function)
- 경사하강법 (Gradient Descent Algorithm)
- 학습 (Training)



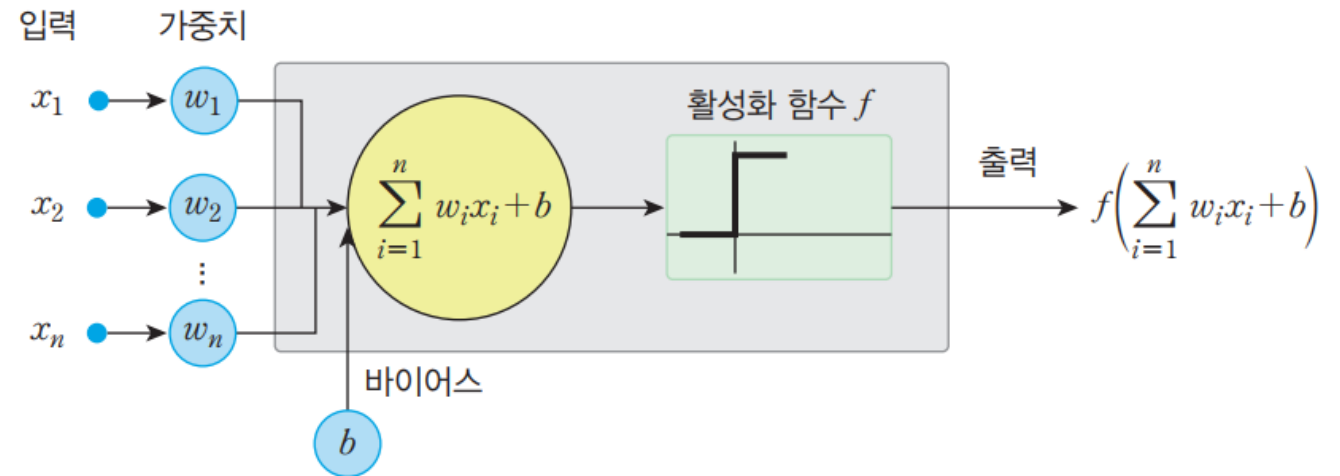
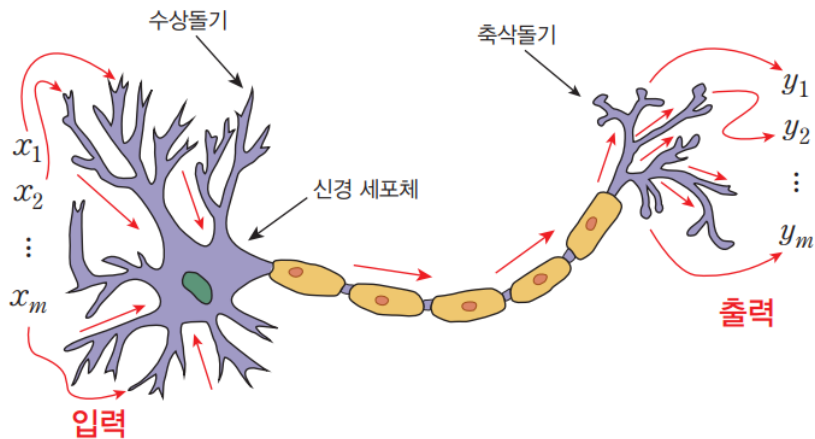
그라디언트는 접선의 기울기로 이해해도 됩니다. 접선의 기울기가 양수이면 반대로 w 를 감소시킵니다.





인공신경망 (Keras, TensorFlow)

- 모델
- 손실 함수 (Loss Function)
- 경사하강법 (Gradient Descent Algorithm)
- 학습 (Training)

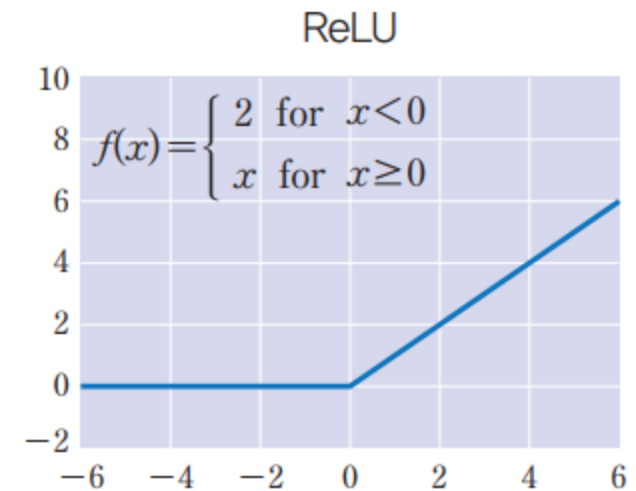
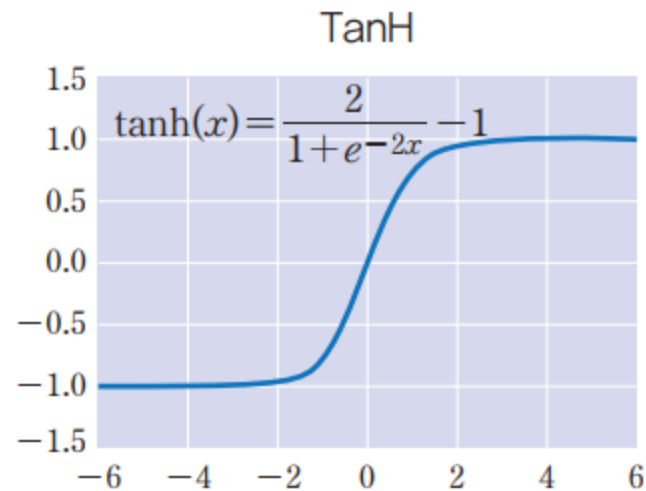
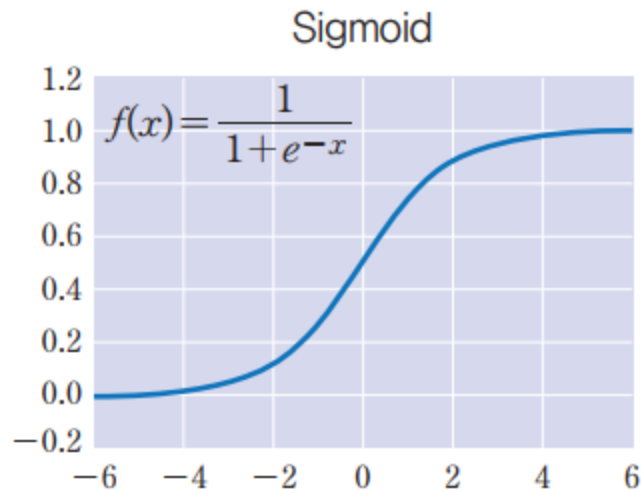




Summary

인공신경망 (Keras, TensorFlow)

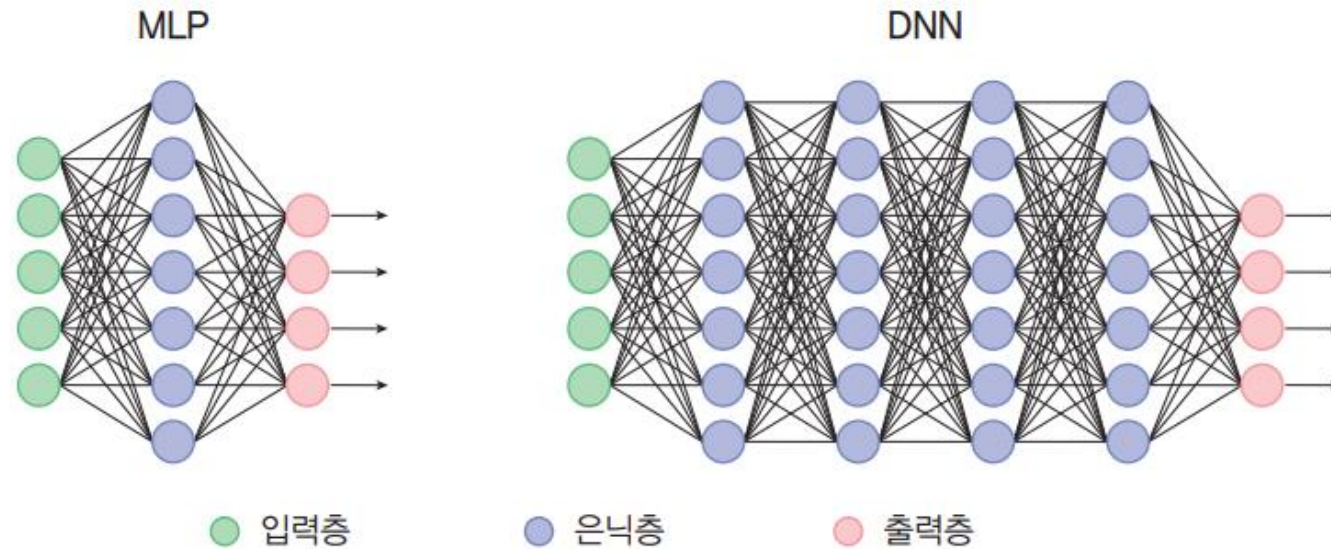
- 모델
- 손실 함수 (Loss Function)
- 경사하강법 (Gradient Descent Algorithm)
- 학습 (Training)





인공신경망 (Keras, TensorFlow)

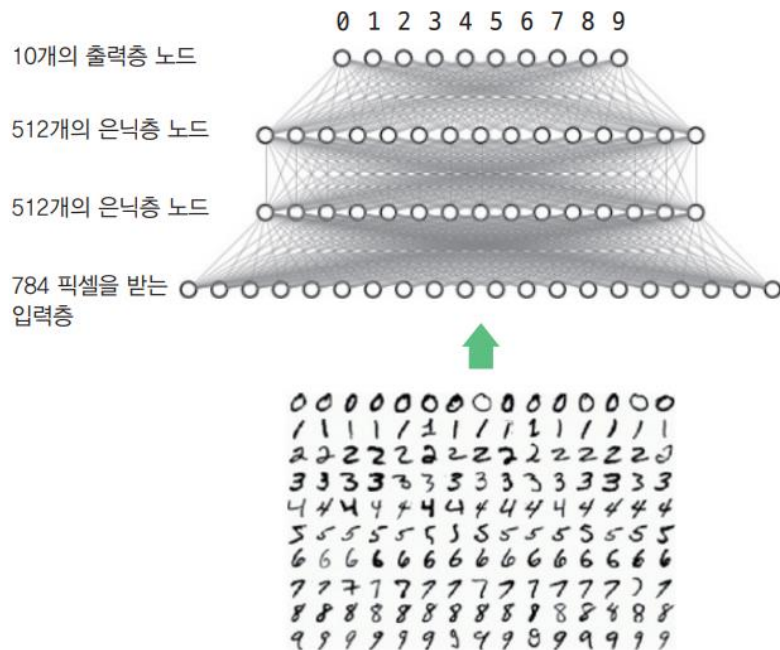
- 모델
- 손실 함수 (Loss Function)
- 경사하강법 (Gradient Descent Algorithm)
- 학습 (Training)





인공신경망 (Keras, TensorFlow)

- MNIST



```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt

mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential()

model.add(tf.keras.layers.Flatten(input_shape=(28, 28)))
model.add(tf.keras.layers.Dense(512, activation='relu'))
model.add(tf.keras.layers.Dropout(0.2))
model.add(tf.keras.layers.Dense(10, activation='softmax'))

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)

print(model.predict(x_test[:1]))
```



기계 학습 (Machine Learning)

패션 MNIST

글로벌미디어학부 <고급프로그래밍및실습>, 이정진



패션 MNIST

- 손 글씨 MNIST 데이터 셋과 유사
- 운동화나 셔츠같은 옷과 신발의 이미지와 이 이미지에 대한 레이블을 제공





패션 MNIST

- 데이터 불러오기



```
# tensorflow와 tf.keras를 임포트
import tensorflow as tf
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt
```

```
# 패션 MNIST 데이터는 keras의 데이터셋에 있는데 이를 읽어와서 학습용, 테스트 데이터로 구분
fashion_mnist = keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```



```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/tra
32768/29515 [=====] - 0s 0us/step
```

```
...
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/tra
4423680/4422102 [=====] - 0s 0us/step
```



패션 MNIST

- 데이터 형태 확인



```
print(train_images.shape) # 학습 이미지의 형태와 레이블을 출력한다
print(train_labels)      # 레이블에는 0에서 9까지의 숫자가 있다
print(test_images.shape)
```



```
(60000, 28, 28)
[9 0 0 ... 3 0 5]
(10000, 28, 28)
```



패션 MNIST

- 데이터 형태 확인

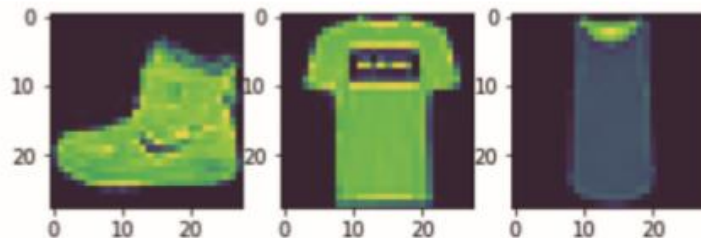


```
fig = plt.figure()
ax1 = fig.add_subplot(1, 3, 1)
ax2 = fig.add_subplot(1, 3, 2)
ax3 = fig.add_subplot(1, 3, 3)

ax1.imshow(train_images[0])      # 첫 번째 훈련용 데이터
ax2.imshow(train_images[1])      # 두 번째 훈련용 데이터
ax3.imshow(train_images[2])      # 세 번째 훈련용 데이터
plt.show()
```



<matplotlib.image.AxesImage at 0x7fe3e6d6c320>





패션 MNIST

- 레이블 의미

레이블	설명
0	티셔츠/상의
1	바지
2	점퍼
3	드레스
4	코트
5	샌들
6	셔츠
7	운동화
8	가방
9	앵클 부츠



패션 MNIST

- 신경망 구조 모델링

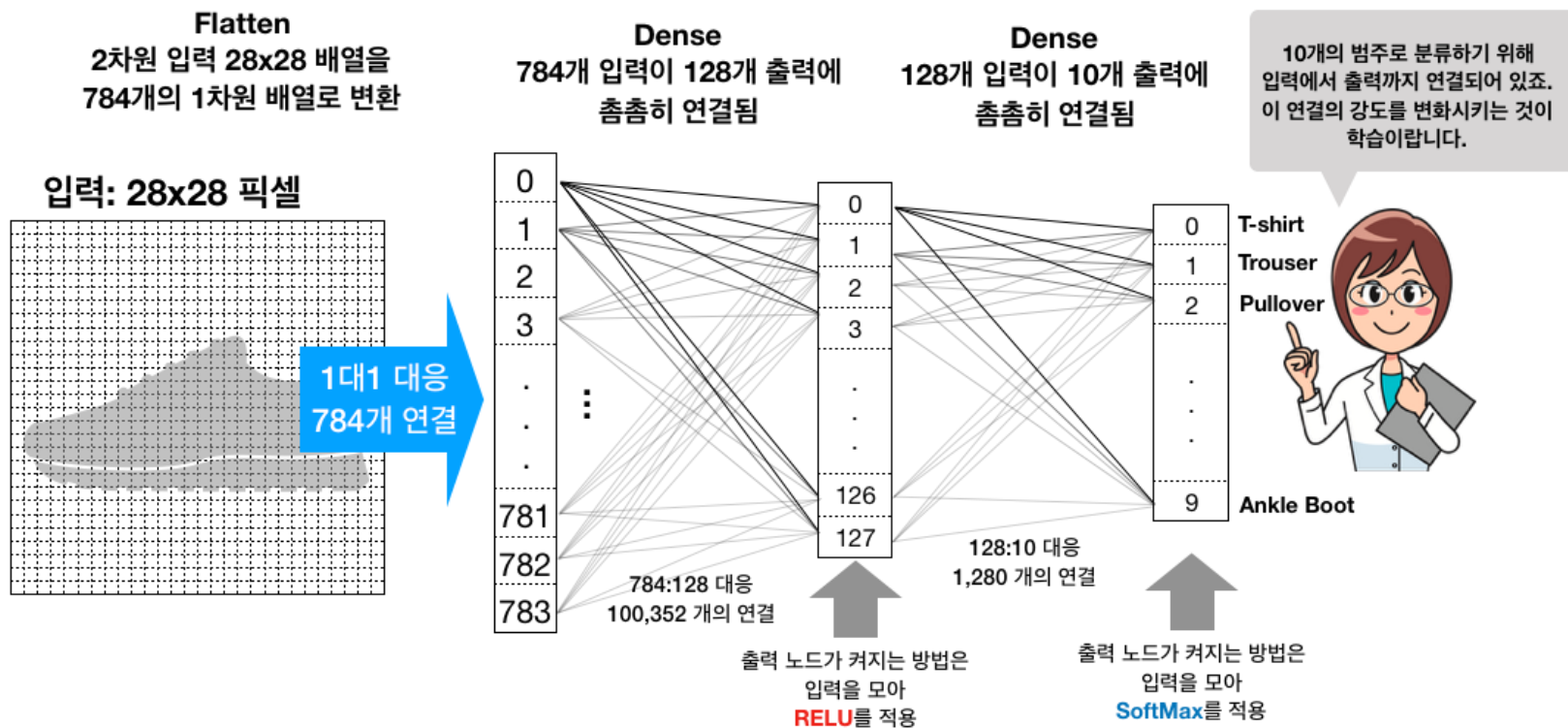


```
model = keras.Sequential([  
    keras.layers.Flatten(input_shape=(28, 28)),  
    keras.layers.Dense(128, activation='relu'),  
    keras.layers.Dense(10, activation='softmax')  
])
```




패션 MNIST

• 신경망 구조 모델링





패션 MNIST

- 손실 함수와 경사하강법 알고리즘 (최적화 기법) 설정



```
model.compile(optimizer='adam',  
              loss='sparse_categorical_crossentropy',  
              metrics=['accuracy'])
```



패션 MNIST

- 학습

```
▶ model.fit(train_images, train_labels, epochs=5)
```

```
→ Epoch 1/5  
1875/1875 [=====] - 3s 2ms/step - loss: 3.6095 - accuracy:  
0.7010  
Epoch 2/5  
1875/1875 [=====] - 3s 2ms/step - loss: 0.6739 - accuracy:  
0.7566  
Epoch 3/5  
1875/1875 [=====] - 3s 2ms/step - loss: 0.6113 - accuracy:  
0.7844  
Epoch 4/5  
1875/1875 [=====] - 3s 2ms/step - loss: 0.5659 - accuracy:  
0.8045  
Epoch 5/5  
1875/1875 [=====] - 3s 2ms/step - loss: 0.5352 - accuracy: 0.8150
```



패션 MNIST

- 테스트



```
test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2)
print('\n테스트 정확도:', test_acc)
```



```
313/313 - 0s - loss: 0.6141 - accuracy: 0.7907
```

```
테스트 정확도: 0.7907000184059143
```



패션 MNIST

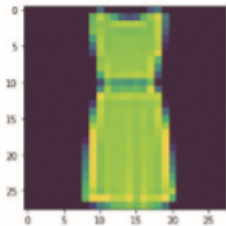
- 학습된 모델 사용해서 예측하기



```
import numpy as np
randIdx = np.random.randint(0, 1000)
plt.imshow(test_images[randIdx])
```



<matplotlib.image.AxesImage at 0x7fe3d54e6358>



```
yhat = model.predict( test_images[randIdx][np.newaxis, :, :])
yhat
```



```
array([[2.0086157e-04, 2.7483705e-04, 4.5031801e-09, 9.9949753e-01,
        2.1863398e-05, 8.0804257e-22, 4.6949867e-06, 0.0000000e+00,
        1.6868819e-07, 0.0000000e+00]], dtype=float32)
```



패션 MNIST

- 학습된 모델 사용해서 예측하기

```
yhat = np.argmax( model.predict( test_images[randIdx][np.newaxis, :, :]) )
yhat
```

3

```
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
               'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']
```

출력 없음

```
yhat = np.argmax( model.predict( test_images[randIdx][np.newaxis, :, :]) )
print(class_names[yhat])
```

Dress



이미지 파일 데이터 적용





이미지 파일 데이터 적용

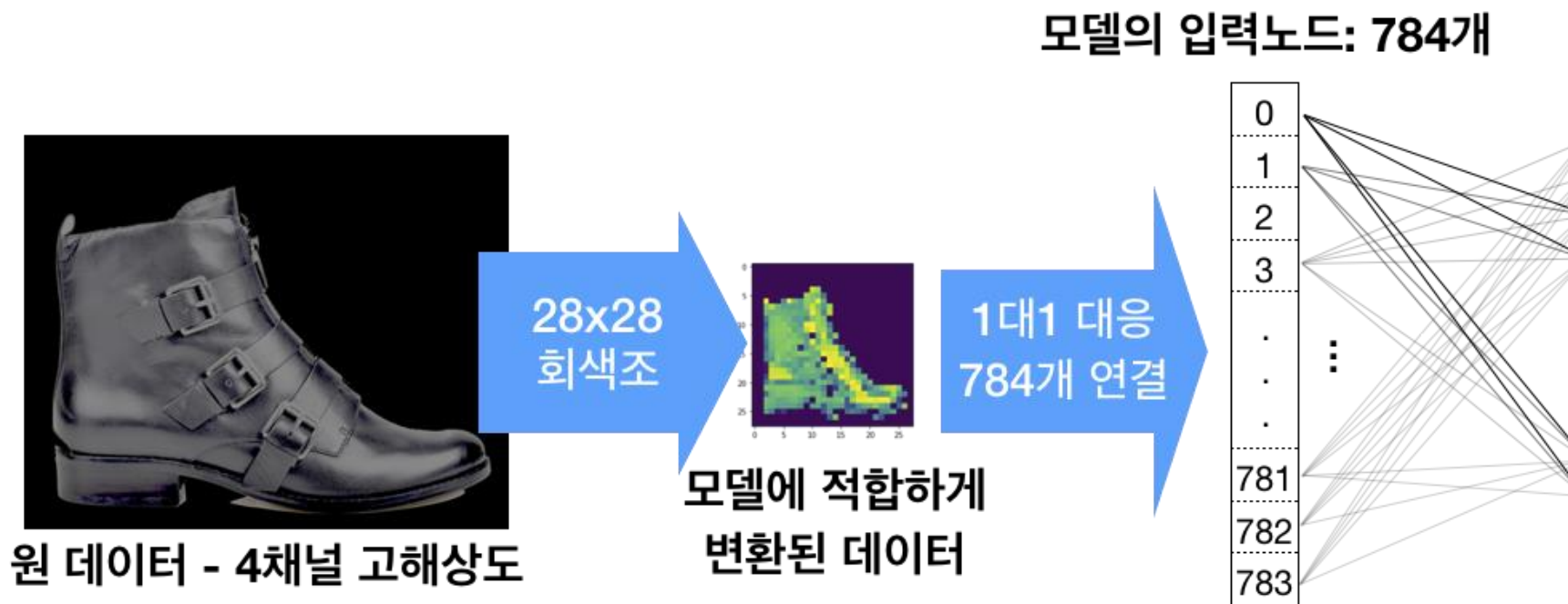
- 맷플롯립을 이용하여 이미지 파일 읽기





이미지 파일 데이터 적용

- 신경망 모델의 입력 데이터 형식에 맞게 조정이 필요!





이미지 파일 데이터 적용

- OpenCV 라이브러리 활용

https://docs.opencv.org/master/d6/d00/tutorial_py_root.html

- 컴퓨터 비전 및 영상 처리를 위한 기능 제공
- 설치법: Anaconda Prompt 실행
conda install -c conda-forge opencv 입력 후 엔터

```
import cv2
img = cv2.imread('./drive/My Drive/myData.png', cv2.IMREAD_GRAYSCALE)
img = cv2.resize(img, (28, 28) )
plt.imshow(img)
```

matplotlib.image.AxesImage at 0x7fe3d3f6a0f0>



이미지 파일 데이터 적용

- $(n, 28, 28)$ 형태의 3차원 배열이어야 하는데, 현재 가지고 있는 이미지는 $(28, 28)$ 이다.
- 이를 넘파이의 `newaxis`를 이용하여 3차원으로 바꾸어 보자.



```
input_data = img[np.newaxis, :, :]  
input_data.shape
```



```
(1, 28, 28)
```



이미지 파일 데이터 적용

- 이제 모델을 이용해 예측해보자

```
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',  
               'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']  
yhat = np.argmax( model.predict( input_data ) )  
  
print(class_names[yhat])
```

Bag

이처럼 학습된 모델이 잘못된 결과를 출력할 수도 있다.



이미지 파일 데이터 적용

- 이미지를 좌우로 뒤집어서 다시 해보자

```
▶ input_mirror = input_data[:, :, ::-1]
  plt.imshow(input_mirror[0])
```

→ <matplotlib.image.AxesImage at 0x7fe3d3f6a0f0>



```
▶ class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
                  'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']
  yhat = np.argmax( model.predict( input_mirror ) )

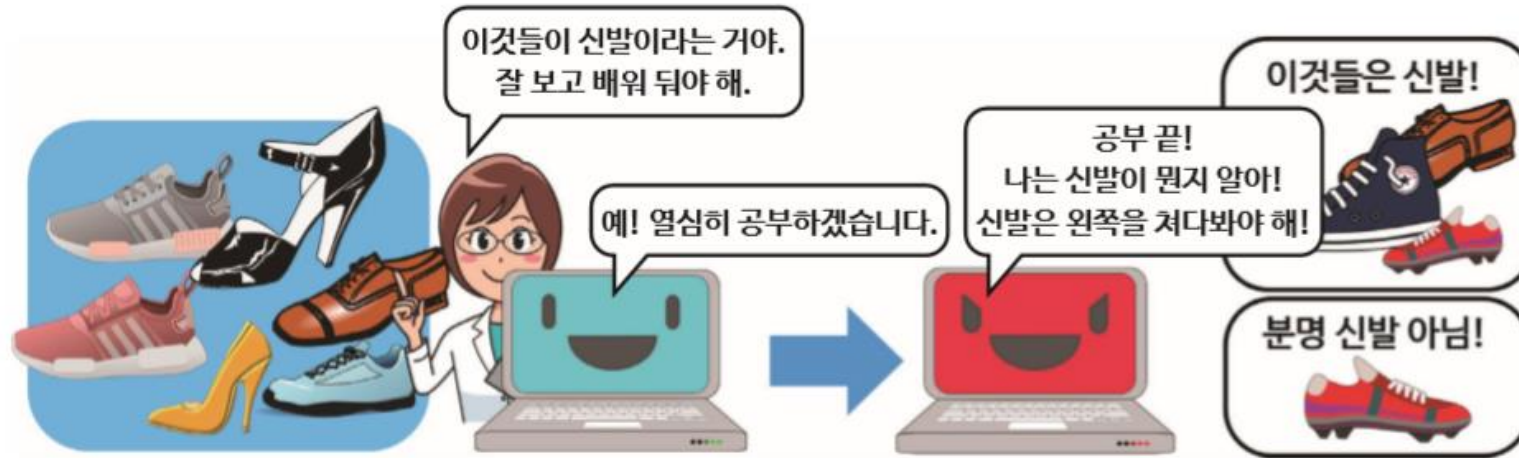
  print(class_names[yhat])
```

→ Ankle boot



신경망 학습과 편향 : 학습의 한계를 인식하자

- 학습이라는 것은 입력으로 주어진 데이터에 따라 한계를 가질 수 밖에 없다.

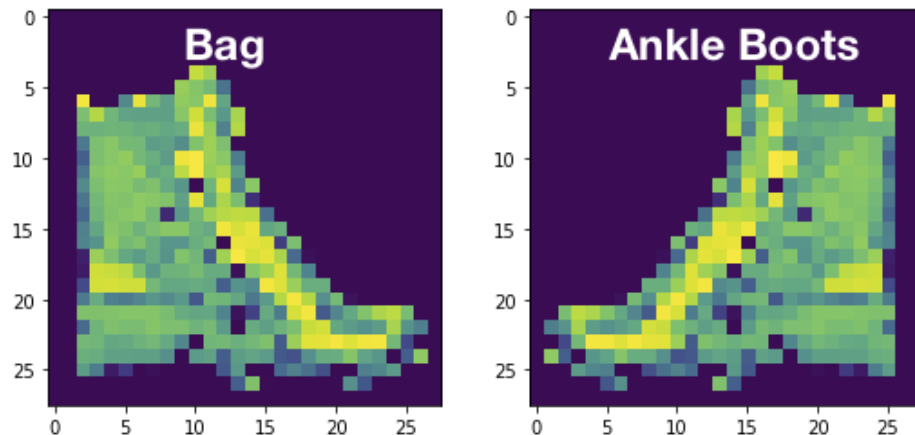


- 훈련용으로 주어진 데이터에서 모든 신발은 왼쪽을 쳐다보고 있었기 때문에 오른쪽을 쳐다보는 발목 부츠가 들어왔을 때, 이것이 신발류라는 것을 인식하지 못하는 것이다.



신경망 학습과 편향 : 학습의 한계를 인식하자

- 전체를 대표하지 못하고 특정한 특징이나 경향을 가진 데이터만 지나치게 학습에 많이 사용되는 것을 **데이터 편향**(data bias)이라고 한다.



- 확보된 데이터를 다양하게 변형하여 편향을 제거하는 방법이 있다. 이것을 **데이터 증강**(data augmentation)이라고 한다.



학습된 모델 저장하기

- Keras에서 제공(.h5)

```
model.compile(optimizer='adam',  
              loss='sparse_categorical_crossentropy',  
              metrics=['accuracy'])  
  
model.fit(x_train, y_train, epochs=5)  
  
model.save("c:\\Dev\\PyP_data\\model.h5")
```




저장된 모델 읽어오기

- Keras에서 제공(.h5)

```
In [45]: model_imported = tf.keras.models.load_model("c:\\Dev\\PyP_data\\model.h5")
```

```
In [46]: model_imported.summary()
```

```
Model: "sequential_12"
```

Layer (type)	Output Shape	Param #
=====		
flatten_12 (Flatten)	(None, 784)	0
dense_24 (Dense)	(None, 512)	401920
dropout_12 (Dropout)	(None, 512)	0
dense_25 (Dense)	(None, 10)	5130
=====		
Total params: 407,050		
Trainable params: 407,050		
Non-trainable params: 0		



기계 학습 (Machine Learning)

코랩에서 구글 드라이브 데이터 접근하기

글로벌미디어학부 <고급프로그래밍및실습>, 이정진



구글 드라이브에 올린 파일 접근하기

- 가장 먼저 해야 할 일은 자신의 드라이브를 코랩에서 사용할 수 있도록 마운트mount하는 것이다. 마운트는 어떤 장치를 컴퓨터 시스템에서 접근할 수 있도록 등록하는 일이다.



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



Go to this URL in a browser: https://accounts.google.com/*****

Enter your authorization code:

4/3wH*****9UA0ng

Mounted at /content/drive



구글 드라이브에 올린 파일 접근하기


- 드라이브를 코랩에 마운트시키려고 하면 이 작업에 대한 승인을 요청하는 페이지의 주소가 아래 그림과 같이 나타난다.


이 부분을 클릭하면 구글 계정 액세스를 요청하는 페이지가 나타난다. 이 요청을 승인하여 인증 코드를 얻는다.

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


... Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=...

Enter your authorization code:

 **Google Drive File Stream**에서 내 Google 계정에 액세스하려고 합니다

 dgpark@gs.cwnu.ac.kr

이렇게 하면 **Google Drive File Stream**에서 다음 작업을 할 수 있습니다.

 Google 드라이브 파일 보기, 수정, 생성, 삭제 ⓘ



구글 드라이브에 올린 파일 접근하기

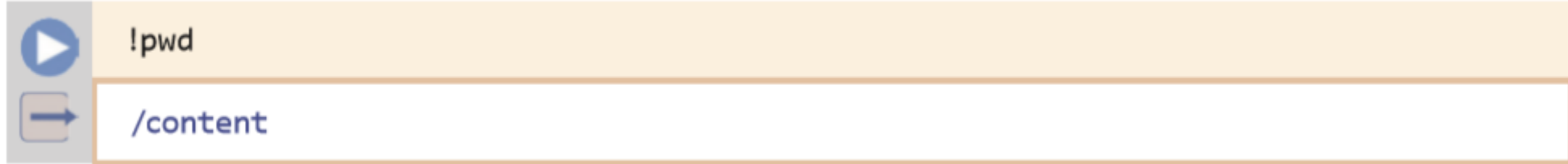
- 승인에 필요한 코드를 복사하여 구글 코랩에 실행결과 셀의 입력창에 넣으면 된다. 이 과정을 마치면 각자의 개인 구글 드라이브를 코랩에서 /content/drive라는 이름으로 사용할 수 있다.





구글 드라이브에 올린 파일 접근하기

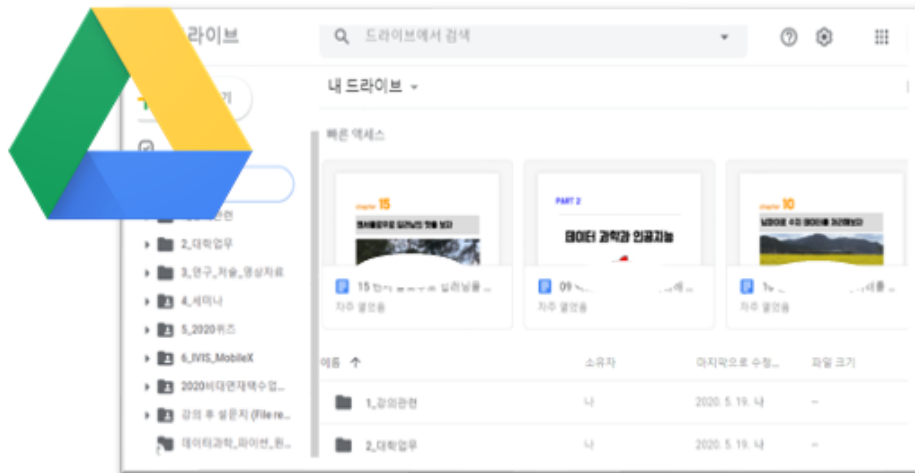
- 구글 코랩에서 현재 작업하고 있는 위치가 어디인지 알고 싶으면 !pwd 를 아래와 같이 입력하면 된다. !은 코랩의 시스템 명령을 입력하는 기호이고, pwd는 “print working directory”의 약어로 현재 작업중인 위치를 출력한다.





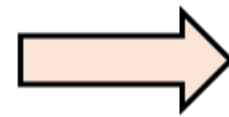
구글 클라우드의 자료를 코랩에서 사용하기

- 구글은 구글 드라이브라는 클라우드 저장소를 제공하고 있다. 그리고 이 저장소에 있는 콘텐츠를 또 다른 서비스인 코랩에서 불러서 이용할 수 있다.



구글 클라우드 저장소

클라우드 저장소를
마운트하여 사용가능

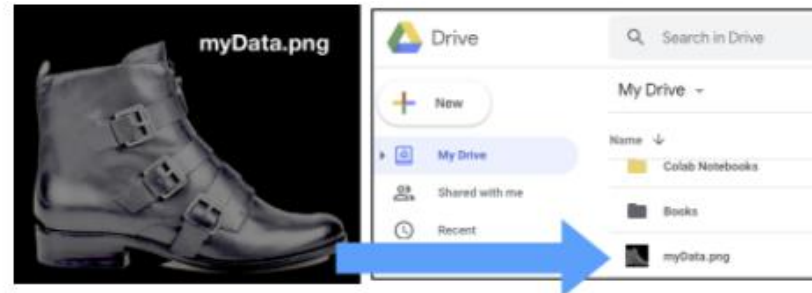


colab

구글 코랩 개발 플랫폼



구글 클라우드의 자료를 코랩에서 사용하기



이 데이터를 업로드 한 후 코랩에서 확인되는지 보자. 코랩에서 여러분이 사용하고 있는 시스템을 관리하는 명령은 ! 기호를 붙여서 입력할 수 있다는 점을 이미 설명했다. 저장 장치의 내용을 볼 때 사용하는 리눅스 명령은 ls 인데, list를 의미한다. 특정 위치의 파일 목록을 나열하여 보이라는 뜻이다.

```
!ls ./drive/'My Drive' -la
```

total	1247						
drwx-----	2	root	root	4096	Sep 4 2019	Books	
drwx-----	2	root	root	4096	Apr 8 2019	DiveIntoDeepLearning	
drwx-----	2	root	root	4096	Sep 18 2018	'massspring lecturenotes'	
-rw-----	1	root	root	1226500	Sep 9 05:20	myData.png	
drwx-----	2	root	root	4096	Nov 19 2019	Pandas	



구글 드라이브의 이미지 파일을 읽어 화면에 표시해 보기

- 맷플롯립을 이용하여 읽어 보도록 하자.



```
import matplotlib.image as mpimg
import matplotlib.pyplot as plt

img = mpimg.imread('./drive/My Drive/myData.png')
plt.imshow(img)
```



<matplotlib.image.AxesImage at 0x7fe3d365fda0>





구글 드라이브의 이미지 파일을 읽어 화면에 표시해 보기

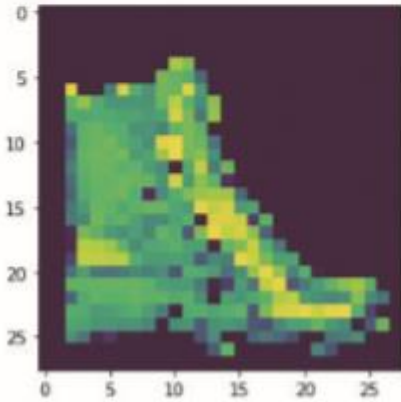
- OpenCV를 사용해 볼 수 있다.



```
import cv2
img = cv2.imread('./drive/My Drive/myData.png', cv2.IMREAD_GRAYSCALE)
img = cv2.resize(img, (28, 28) )
plt.imshow(img)
```



matplotlib.image.AxesImage at 0x7fe3d3f6a0f0>





기계 학습 (Machine Learning)

수고하셨습니다 😊