분류기로서의 DNN

```
import numpy as np
import pandas as pd

import tensorflow as tf
from tensorflow import keras

import numpy as np
import matplotlib.pyplot as plt

import time


def build_square_data(all_low, all_high, square_low, square_high, zero_count, one_count):

  x1 = np.random.rand(one_count)
  x1 = x1*(square_high-square_low) + square_low

  x0 = np.random.rand(zero_count*100)
  x0 = x0[ (x0<square_low) | (square_high<x0) ]
  x0 = x0[:zero_count]

  x = np.append(x1, x0)

  plt.hist(x)
  plt.xlim(0, 1)
  plt.show()

  # 범위 안의 것을 1, 범위 밖의 것을 0으로 하고
  all_data = np.ones((len(x),2))
  all_data[:,0] = x

  all_data[:len(x1),1] = 1
  all_data[len(x1):,1] = 0

  # 섞는다
  np.random.shuffle(all_data)

  # x, y로 분리하고
  x = all_data[:,0]
  y = all_data[:,1]

  plt.xlim(0, 1)

  plt.scatter(x, y)

  return x, y
```

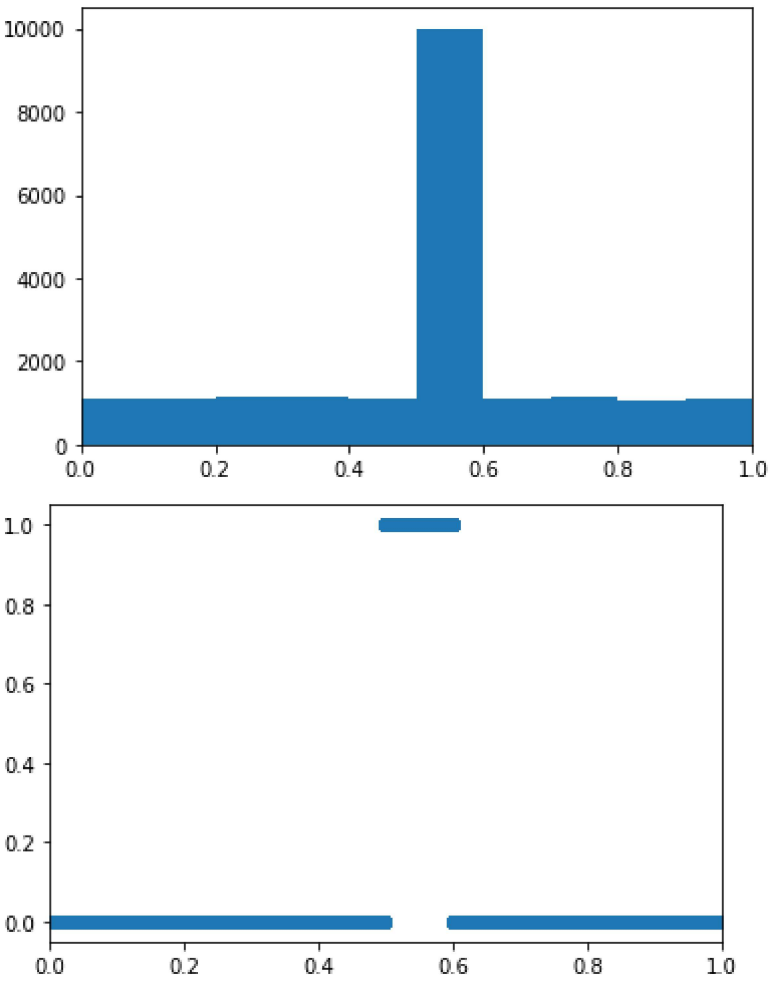
네모 함수의 학습

특정 영역은 1, 이외의 영역은 0을 출력

1개의 출력노드를 갖는다

```
all_low = 0
all_high = 1
square_low = 0.5
square_high = 0.6
zero_count = 10000
one_count = 10000

train_x, train_y = build_square_data(all_low, all_high, square_low, square_high, zero_count, one_co
```




```
from tensorflow import keras
from tensorflow.keras import optimizers
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import SGD


model = keras.Sequential()
model.add(Dense(10, activation='relu', input_shape=(1,)))
model.add(Dense(10, activation='relu'))
model.add(Dense(10, activation='relu'))
model.add(Dense(1, activation="sigmoid"))
```

```
model.compile(optimizer="SGD", loss="mse")
model.fit(train_x, train_y, epochs=100, verbose=1, batch_size=128, validation_split=0.1)

x_ = np.arange(all_low, all_high, 0.01)
y_ = model.predict(x_)
plt.scatter(x_, y_)
```

```
Epoch 3/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2465 - val_loss: 0.2460
Epoch 4/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2457 - val_loss: 0.2452
Epoch 5/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2449 - val_loss: 0.2444
Epoch 6/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2440 - val_loss: 0.2435
Epoch 7/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2431 - val_loss: 0.2425
Epoch 8/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2421 - val_loss: 0.2415
Epoch 9/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2410 - val_loss: 0.2403
Epoch 10/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2397 - val_loss: 0.2390
Epoch 11/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2383 - val_loss: 0.2377
Epoch 12/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2370 - val_loss: 0.2363
Epoch 13/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2356 - val_loss: 0.2349
Epoch 14/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2341 - val_loss: 0.2333
Epoch 15/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2325 - val_loss: 0.2316
Epoch 16/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2307 - val_loss: 0.2298
Epoch 17/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2289 - val_loss: 0.2279
Epoch 18/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2268 - val_loss: 0.2257
Epoch 19/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2246 - val_loss: 0.2234
Epoch 20/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2221 - val_loss: 0.2208
Epoch 21/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2194 - val_loss: 0.2181
Epoch 22/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2165 - val_loss: 0.2150
Epoch 23/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2132 - val_loss: 0.2117
Epoch 24/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2097 - val_loss: 0.2080
Epoch 25/100
141/141 [==============================] - 0s 1ms/step - loss: 0.2059 - val_loss: 0.2041
Epoch 26/100
141/141 [==============================] - 0s 2ms/step - loss: 0.2018 - val_loss: 0.1998
Epoch 27/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1973 - val_loss: 0.1952
Epoch 28/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1925 - val_loss: 0.1903
Epoch 29/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1874 - val_loss: 0.1851
Epoch 30/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1820 - val_loss: 0.1796
Epoch 31/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1764 - val_loss: 0.1739
```

```
Epoch 32/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1704 - val_loss: 0.1678
Epoch 33/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1644 - val_loss: 0.1617
Epoch 34/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1581 - val_loss: 0.1555
Epoch 35/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1518 - val_loss: 0.1492
Epoch 36/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1455 - val_loss: 0.1430
Epoch 37/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1392 - val_loss: 0.1368
Epoch 38/100
141/141 [==============================] - 0s 2ms/step - loss: 0.1331 - val_loss: 0.1308
Epoch 39/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1271 - val_loss: 0.1250
Epoch 40/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1213 - val_loss: 0.1193
Epoch 41/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1158 - val_loss: 0.1139
Epoch 42/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1105 - val_loss: 0.1088
Epoch 43/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1055 - val_loss: 0.1040
Epoch 44/100
141/141 [==============================] - 0s 1ms/step - loss: 0.1008 - val_loss: 0.0996
Epoch 45/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0964 - val_loss: 0.0953
Epoch 46/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0923 - val_loss: 0.0913
Epoch 47/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0885 - val_loss: 0.0877
Epoch 48/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0849 - val_loss: 0.0843
Epoch 49/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0816 - val_loss: 0.0810
Epoch 50/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0785 - val_loss: 0.0780
Epoch 51/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0756 - val_loss: 0.0753
Epoch 52/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0728 - val_loss: 0.0726
Epoch 53/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0703 - val_loss: 0.0702
Epoch 54/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0679 - val_loss: 0.0680
Epoch 55/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0657 - val_loss: 0.0658
Epoch 56/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0636 - val_loss: 0.0638
Epoch 57/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0617 - val_loss: 0.0620
Epoch 58/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0598 - val_loss: 0.0602
Epoch 59/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0581 - val_loss: 0.0586
Epoch 60/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0565 - val_loss: 0.0570
```

```
Epoch 61/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0549 - val_loss: 0.0555
Epoch 62/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0535 - val_loss: 0.0541
Epoch 63/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0521 - val_loss: 0.0528
Epoch 64/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0508 - val_loss: 0.0515
Epoch 65/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0495 - val_loss: 0.0504
Epoch 66/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0483 - val_loss: 0.0493
Epoch 67/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0472 - val_loss: 0.0481
Epoch 68/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0461 - val_loss: 0.0472
Epoch 69/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0451 - val_loss: 0.0462
Epoch 70/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0442 - val_loss: 0.0452
Epoch 71/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0432 - val_loss: 0.0443
Epoch 72/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0423 - val_loss: 0.0434
Epoch 73/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0415 - val_loss: 0.0426
Epoch 74/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0407 - val_loss: 0.0419
Epoch 75/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0399 - val_loss: 0.0411
Epoch 76/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0391 - val_loss: 0.0403
Epoch 77/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0384 - val_loss: 0.0397
Epoch 78/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0377 - val_loss: 0.0390
Epoch 79/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0370 - val_loss: 0.0385
Epoch 80/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0364 - val_loss: 0.0377
Epoch 81/100
141/141 [==============================] - 0s 2ms/step - loss: 0.0358 - val_loss: 0.0372
Epoch 82/100
141/141 [==============================] - 0s 1ms/step - loss: 0.0352 - val_loss: 0.0367
Epoch 83/100
```

특정 패턴인 경우 1, 다른 경우인 경우 0으로 분류한다.

출력된 값이 0.5보다 크면 1로, 0.5보다 작으면 0으로 간주한다.

```
141/141 [==============================] - 0s 1ms/step - loss: 0.0335 - val_loss: 0.0349
```

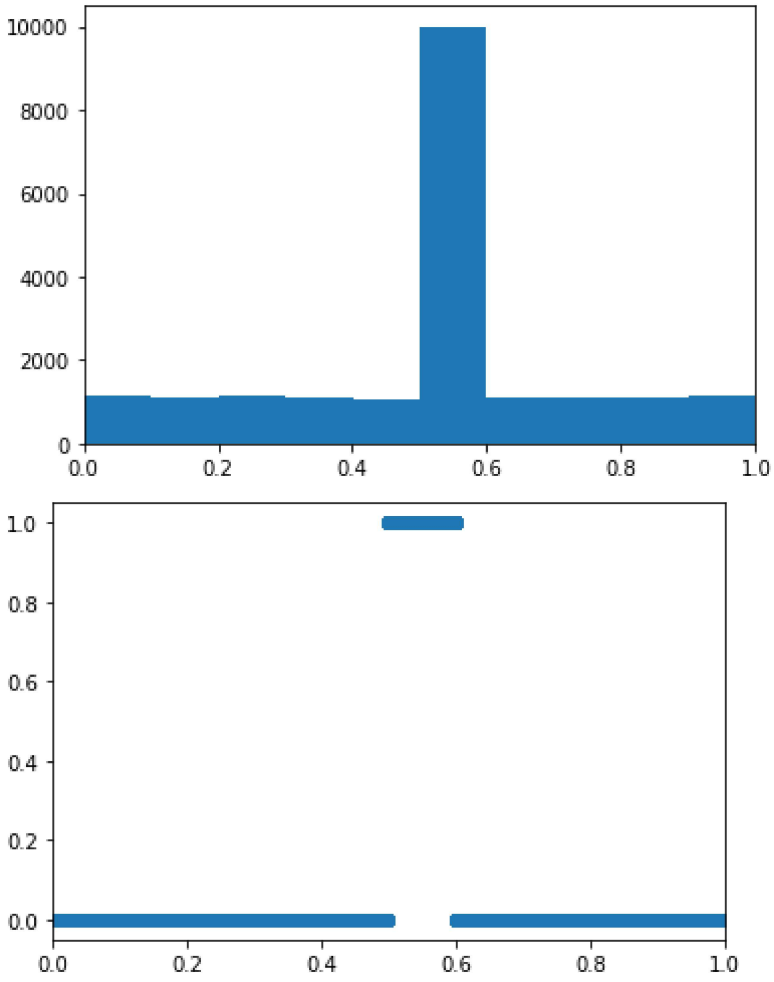
클래스 2개, 출력 노드 2개

분류 카테고리의 개수 대로 출력노드를 갖고, one-hot-encoding된 값을 출력하면, 변별력이 커진다.

```
141/141 [==============================] - 0s 1ms/step - loss: 0.0316 - val_loss: 0.0329
```

```
all_low = 0
all_high = 1
square_low = 0.5
square_high = 0.6
zero_count = 10000
one_count = 10000

train_x, train_y = build_square_data(all_low, all_high, square_low, square_high, zero_count, one_co
```




```
# ADD START
reshaped_y = train_y.reshape((train_y.shape[0],1))
train_y = np.append(reshaped_y, reshaped_y, axis=1)
train_y[:,1] = 1-train_y[:,0]
# ADD END


print(train_y[:10])
```

```
[[0. 1.]
 [1. 0.]
 [1. 0.]
 [1. 0.]
 [1. 0.]
 [1. 0.]
 [1. 0.]
 [0. 1.]
 [0. 1.]
 [1. 0.]]
```

```
from tensorflow import keras
from tensorflow.keras import optimizers
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import SGD


model = keras.Sequential()
model.add(Dense(10, activation='relu', input_shape=(1,)))
model.add(Dense(10, activation='relu'))
model.add(Dense(10, activation='relu'))
# model.add(Dense(1))
model.add(Dense(2))

model.compile(optimizer="SGD", loss="mse")
model.fit(train_x, train_y, epochs=100, verbose=1, batch_size=32, validation_split=0.1)

x_ = np.arange(all_low, all_high, 0.01)
y_ = model.predict(x_)
plt.scatter(x_, y_[:,0])
```

```
Epoch 1/100
563/563 [==============================] - 1s 1ms/step - loss: 0.2449 - val_loss: 0.2244
Epoch 2/100
563/563 [==============================] - 1s 1ms/step - loss: 0.2100 - val_loss: 0.1926
Epoch 3/100
563/563 [==============================] - 1s 1ms/step - loss: 0.1713 - val_loss: 0.1453
Epoch 4/100
563/563 [==============================] - 1s 1ms/step - loss: 0.1191 - val_loss: 0.0948
Epoch 5/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0810 - val_loss: 0.0710
Epoch 6/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0671 - val_loss: 0.0630
Epoch 7/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0618 - val_loss: 0.0589
Epoch 8/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0585 - val_loss: 0.0566
Epoch 9/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0560 - val_loss: 0.0544
Epoch 10/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0541 - val_loss: 0.0530
Epoch 11/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0524 - val_loss: 0.0528
Epoch 12/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0510 - val_loss: 0.0498
Epoch 13/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0498 - val_loss: 0.0483
Epoch 14/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0487 - val_loss: 0.0485
Epoch 15/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0476 - val_loss: 0.0461
Epoch 16/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0466 - val_loss: 0.0455
Epoch 17/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0457 - val_loss: 0.0444
Epoch 18/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0446 - val_loss: 0.0432
Epoch 19/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0436 - val_loss: 0.0423
Epoch 20/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0425 - val_loss: 0.0412
Epoch 21/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0414 - val_loss: 0.0413
Epoch 22/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0403 - val_loss: 0.0390
Epoch 23/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0391 - val_loss: 0.0390
Epoch 24/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0377 - val_loss: 0.0367
Epoch 25/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0364 - val_loss: 0.0355
Epoch 26/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0349 - val_loss: 0.0339
Epoch 27/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0335 - val_loss: 0.0330
Epoch 28/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0320 - val_loss: 0.0309
Epoch 29/100
563/563 [==============================] - 1s 1ms/step - loss: 0.0304 - val_loss: 0.0302
```