

```
%tensorflow_version 1.x
```

## ▼ IRIS 데이터 분류

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
import tensorflow as tf
```

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

```
!wget https://raw.githubusercontent.com/dhrim/wiset_2020_06/master/material/deep_learning/iris.csv
```

```
--2020-06-22 05:33:33-- https://raw.githubusercontent.com/dhrim/wiset_2020_06/master/material/deep_learning/iris.csv
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.0.133, 151.101.64.
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.0.133|:443... connected
HTTP request sent, awaiting response... 200 OK
Length: 2720 (2.7K) [text/plain]
Saving to: 'iris.csv.4'
```

```
iris.csv.4          100%[=====>]    2.66K  --.-KB/s    in 0s
```

```
2020-06-22 05:33:33 (54.4 MB/s) - 'iris.csv.4' saved [2720/2720]
```

```
iris = pd.read_csv("iris.csv")
iris.head()
```

```

  sepal_length  sepal_width  petal_length  petal_width  setosa  versicolor  virginica
0         5.1         3.5         1.4           0.2         0         1         0
1         4.9         3.0         1.5           0.2         0         1         0
2         4.7         3.2         1.6           0.2         0         1         0
3         5.2         3.4         1.6           0.2         0         1         0
4         5.2         3.5         1.5           0.2         0         1         0
5         5.0         3.4         1.4           0.2         0         1         0
6         4.6         3.3         1.3           0.2         0         1         0
7         5.4         3.7         1.5           0.2         0         1         0
8         4.8         3.4         1.4           0.2         0         1         0
9         5.0         3.5         1.6           0.2         0         1         0
10        5.1         3.6         1.4           0.2         0         1         0
11        4.9         3.5         1.4           0.1         0         1         0
12        5.4         3.6         1.4           0.1         0         1         0
13        5.2         3.5         1.5           0.2         0         1         0
14        5.2         3.6         1.4           0.2         0         1         0
15        5.0         3.4         1.5           0.2         0         1         0
16        5.1         3.5         1.4           0.2         0         1         0
17        4.9         3.6         1.4           0.1         0         1         0
18        5.4         3.6         1.3           0.1         0         1         0
19        5.2         3.5         1.5           0.2         0         1         0
20        5.2         3.6         1.4           0.2         0         1         0
21        5.0         3.5         1.6           0.2         0         1         0
22        5.1         3.6         1.4           0.2         0         1         0
23        4.9         3.4         1.5           0.1         0         1         0
24        5.4         3.6         1.3           0.1         0         1         0
25        5.2         3.5         1.5           0.2         0         1         0
26        5.2         3.6         1.4           0.2         0         1         0
27        5.0         3.5         1.6           0.2         0         1         0
28        5.1         3.6         1.4           0.2         0         1         0
29        4.9         3.5         1.4           0.1         0         1         0
30        5.4         3.6         1.3           0.1         0         1         0
31        5.2         3.5         1.5           0.2         0         1         0
32        5.2         3.6         1.4           0.2         0         1         0
33        5.0         3.5         1.6           0.2         0         1         0
34        5.1         3.6         1.4           0.2         0         1         0
35        4.9         3.4         1.5           0.1         0         1         0
36        5.4         3.6         1.3           0.1         0         1         0
37        5.2         3.5         1.5           0.2         0         1         0
38        5.2         3.6         1.4           0.2         0         1         0
39        5.0         3.5         1.6           0.2         0         1         0
40        5.1         3.6         1.4           0.2         0         1         0
41        4.9         3.4         1.5           0.1         0         1         0
42        5.4         3.6         1.3           0.1         0         1         0
43        5.2         3.5         1.5           0.2         0         1         0
44        5.2         3.6         1.4           0.2         0         1         0
45        5.0         3.5         1.6           0.2         0         1         0
46        5.1         3.6         1.4           0.2         0         1         0
47        4.9         3.4         1.5           0.1         0         1         0
48        5.4         3.6         1.3           0.1         0         1         0
49        5.2         3.5         1.5           0.2         0         1         0
50        5.2         3.6         1.4           0.2         0         1         0
51        5.0         3.5         1.6           0.2         0         1         0
52        5.1         3.6         1.4           0.2         0         1         0
53        4.9         3.4         1.5           0.1         0         1         0
54        5.4         3.6         1.3           0.1         0         1         0
55        5.2         3.5         1.5           0.2         0         1         0
56        5.2         3.6         1.4           0.2         0         1         0
57        5.0         3.5         1.6           0.2         0         1         0
58        5.1         3.6         1.4           0.2         0         1         0
59        4.9         3.4         1.5           0.1         0         1         0
60        5.4         3.6         1.3           0.1         0         1         0
61        5.2         3.5         1.5           0.2         0         1         0
62        5.2         3.6         1.4           0.2         0         1         0
63        5.0         3.5         1.6           0.2         0         1         0
64        5.1         3.6         1.4           0.2         0         1         0
65        4.9         3.4         1.5           0.1         0         1         0
66        5.4         3.6         1.3           0.1         0         1         0
67        5.2         3.5         1.5           0.2         0         1         0
68        5.2         3.6         1.4           0.2         0         1         0
69        5.0         3.5         1.6           0.2         0         1         0
70        5.1         3.6         1.4           0.2         0         1         0
71        4.9         3.4         1.5           0.1         0         1         0
72        5.4         3.6         1.3           0.1         0         1         0
73        5.2         3.5         1.5           0.2         0         1         0
74        5.2         3.6         1.4           0.2         0         1         0
75        5.0         3.5         1.6           0.2         0         1         0
76        5.1         3.6         1.4           0.2         0         1         0
77        4.9         3.4         1.5           0.1         0         1         0
78        5.4         3.6         1.3           0.1         0         1         0
79        5.2         3.5         1.5           0.2         0         1         0
80        5.2         3.6         1.4           0.2         0         1         0
81        5.0         3.5         1.6           0.2         0         1         0
82        5.1         3.6         1.4           0.2         0         1         0
83        4.9         3.4         1.5           0.1         0         1         0
84        5.4         3.6         1.3           0.1         0         1         0
85        5.2         3.5         1.5           0.2         0         1         0
86        5.2         3.6         1.4           0.2         0         1         0
87        5.0         3.5         1.6           0.2         0         1         0
88        5.1         3.6         1.4           0.2         0         1         0
89        4.9         3.4         1.5           0.1         0         1         0
90        5.4         3.6         1.3           0.1         0         1         0
91        5.2         3.5         1.5           0.2         0         1         0
92        5.2         3.6         1.4           0.2         0         1         0
93        5.0         3.5         1.6           0.2         0         1         0
94        5.1         3.6         1.4           0.2         0         1         0
95        4.9         3.4         1.5           0.1         0         1         0
96        5.4         3.6         1.3           0.1         0         1         0
97        5.2         3.5         1.5           0.2         0         1         0
98        5.2         3.6         1.4           0.2         0         1         0
99        5.0         3.5         1.6           0.2         0         1         0

```

```
iris.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 120 entries, 0 to 119
Data columns (total 7 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   sepal_length    120 non-null    float64
1   sepal_width     120 non-null    float64
2   petal_length    120 non-null    float64
```

```
data = iris.to_numpy()
print(data.shape)
print(data[:5])
```

```
(120, 7)
[[6.4 2.8 5.6 2.2 0.  0.  1. ]
 [5.  2.3 3.3 1.  0.  1.  0. ]
 [4.9 2.5 4.5 1.7 0.  0.  1. ]
 [4.9 3.1 1.5 0.1 1.  0.  0. ]
 [5.7 3.8 1.7 0.3 1.  0.  0. ]]
```

```
x = data[:, :4]
y = data[:, 4:]
```

```
split_index = 100
```

```
train_x, test_x = x[:split_index], x[split_index:]
train_y, test_y = y[:split_index], y[split_index:]
```

```
model = keras.Sequential([
    keras.layers.Dense(10, activation='relu', input_shape=(4,)),
    keras.layers.Dense(10, activation='relu'),
    keras.layers.Dense(3, activation='softmax')
])
```

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

```
model.fit(train_x, train_y, epochs=200, verbose=0)
```

```
<tensorflow.python.keras.callbacks.History at 0x7f5dc4a0c668>
```

```
loss, acc = model.evaluate(test_x, test_y)
```

```
print("loss :", loss)
print("acc :", acc)
```

```
20/20 [=====] - 0s 2ms/sample - loss: 0.1240 - acc: 1.0000
loss : 0.12404048442840576
acc : 1.0
```

```
y_hat = model.predict(test_x)
```

```
y_ = model.predict(test_x)
print(y_)
print(np.argmax(y_, axis=1))
```

```
[[9.88803208e-01 1.09390430e-02 2.57815002e-04]
 [2.84629297e-02 7.84012079e-01 1.87524974e-01]
 [9.38488066e-01 5.87515011e-02 2.76041962e-03]
 [1.25919478e-02 8.15516770e-01 1.71891272e-01]
 [9.89015162e-01 1.07451221e-02 2.39656292e-04]
 [9.93278921e-01 6.59793336e-03 1.23124933e-04]
 [9.95374620e-01 4.55506053e-03 7.04094346e-05]
 [9.94435310e-01 5.46773337e-03 9.69887478e-05]
 [1.98012628e-02 7.82107472e-01 1.98091239e-01]
 [9.88133848e-01 1.16105787e-02 2.55611085e-04]
 [3.29409057e-04 3.10191125e-01 6.89479470e-01]
 [5.44510130e-03 7.38748789e-01 2.55806088e-01]
 [9.87312376e-01 1.24126561e-02 2.74882099e-04]
 [2.73284859e-05 1.07583381e-01 8.92389357e-01]
 [9.85827208e-01 1.38146207e-02 3.58257676e-04]
 [3.46858823e-03 6.92536116e-01 3.03995341e-01]
 [1.33099398e-02 8.07060063e-01 1.79630011e-01]
 [9.84834313e-01 1.47596225e-02 4.06097912e-04]
 [9.89928901e-01 9.85415187e-03 2.16860702e-04]
 [1.03893066e-02 7.78468311e-01 2.11142391e-01]]
[0 1 0 1 0 0 0 0 1 0 2 1 0 2 0 1 1 0 0 1]
```

## loss categorical\_crossentropy

2가지 crossentropy 사용 방법

- categorical\_crossentropy
- sparse\_categorical\_crossentropy

## categorical\_crossentropy

y의 값이 one hot encoding인 경우

```
1,0,0
0,1,0
0,0,1
```

출력 레이어 설정

```
model.add(Dense(3, activation="softmax")) # 출력 레이어
```

loss 설정

```
model.compile(..., loss='categorical_crossentropy')
```

## sparse\_categorical\_crossentropy

y의 값이 one hot encoding인 경우

```
0
1
2
```

출력 레이어 설정

```
model.add(Dense(3, activation="softmax")) # 출력 레이어. 10이 아니라 클래스 수 3
```

loss 설정

```
model.compile(..., loss='sparse_categorical_crossentropy')
```

## ▶ iris\_dnn with category index

아래의 코드는 iris\_dnn\_and\_optimizer.ipynb의 코드를 기반으로 한다.

```
!wget https://raw.githubusercontent.com/dhrim/wiset_2020_06/master/material/deep_learning/iris_with
```

```
--2020-06-22 05:36:34-- https://raw.githubusercontent.com/dhrim/wiset_2020_06/master/material/deep_learning/iris_with
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.0.133, 151.101.64.
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.0.133|:443... con
HTTP request sent, awaiting response... 200 OK
Length: 2218 (2.2K) [text/plain]
Saving to: 'iris_with_category_index.csv'
```

```
iris_with_category_ 100%[=====>] 2.17K --.-KB/s in 0s
```

```
2020-06-22 05:36:35 (41.6 MB/s) - 'iris_with_category_index.csv' saved [2218/2218]
```

```
iris = pd.read_csv("iris_with_category_index.csv")
iris.head()
```

```

sepal_length  sepal_width  petal_length  petal_width  class
0             6.4         2.8           5.6          2.2      2
1             5.0         2.3           3.3          1.0      1
2             4.9         2.5           4.5          1.7      2
3             4.9         3.1           1.5          0.1      0
4             5.7         3.8           1.7          0.3      0
```

```
data = iris.to_numpy()
```

```
print(data.shape)
print(data[:5])
```

```
(120, 5)
[[6.4 2.8 5.6 2.2 2. ]
 [5.  2.3 3.3 1.  1. ]
 [4.9 2.5 4.5 1.7 2. ]
 [4.9 3.1 1.5 0.1 0. ]
 [5.7 3.8 1.7 0.3 0. ]]
```

```
x = data[:, :4]
y = data[:, 4:]
```

```
split_index = 100
```

```
train_x, test_x = x[:split_index], x[split_index:]
train_y, test_y = y[:split_index], y[split_index:]
```

```
print(train_x.shape)
print(train_y.shape)
print(test_x.shape)
print(test_y.shape)
```

```
(100, 4)
(100, 1)
(20, 4)
(20, 1)
```

```
model = keras.Sequential()
model.add(Dense(10, activation='relu', input_shape=(4,)))
model.add(Dense(10, activation='relu'))
model.add(Dense(3, activation="softmax")) # 10이 아니고 클래스 수 3이다
```

```
# model.compile(optimizer="SGD", loss="categorical_crossentropy", metrics=["accuracy"])
model.compile(optimizer="SGD", loss="sparse_categorical_crossentropy", metrics=["accuracy"])
model.summary()
```

```
model.fit(train_x, train_y, epochs=1000, verbose=0, batch_size=20)
```

```
loss, acc = model.evaluate(test_x, test_y)
print("loss=", loss)
print("acc=", acc)
```



Model: "sequential\_5"

Layer (type)	Output Shape	Param #
dense_15 (Dense)	(None, 10)	50
dense_16 (Dense)	(None, 10)	110

```
y_ = model.predict(test_x)
print(y_)
print(np.argmax(y_, axis=1))
```



```
[[9.9858761e-01 1.4124756e-03 1.1726086e-16]
 [1.0649669e-02 9.8910856e-01 2.4171591e-04]
 [9.9112213e-01 8.8779069e-03 1.1856175e-13]
 [2.0122137e-03 9.9777240e-01 2.1540190e-04]
 [9.9797207e-01 2.0279635e-03 2.6774227e-16]
 [9.9942064e-01 5.7937839e-04 2.7495446e-18]
 [9.9952018e-01 4.7986573e-04 8.9114990e-19]
 [9.9934214e-01 6.5783510e-04 5.4727638e-18]
 [7.0785196e-03 9.9284768e-01 7.3791525e-05]
 [9.9791247e-01 2.0875691e-03 4.8763486e-16]
 [1.4534585e-06 2.3011844e-01 7.6988006e-01]
 [7.3111284e-04 9.9648261e-01 2.7862696e-03]
 [9.9775296e-01 2.2470313e-03 3.3707470e-16]
 [1.0055204e-08 2.9165525e-02 9.7083449e-01]
 [9.9653178e-01 3.4682492e-03 1.4455989e-15]
 [1.3583555e-04 9.2979109e-01 7.0072994e-02]
 [1.1672439e-03 9.9793506e-01 8.9770218e-04]
 [9.9759477e-01 2.4051964e-03 2.8108217e-15]
 [9.9894172e-01 1.0582770e-03 6.1290894e-17]
 [3.7317239e-03 9.9603599e-01 2.3226807e-04]]
[0 1 0 1 0 0 0 0 1 0 2 1 0 2 0 1 1 0 0 1]
```

