tensorflow2教程-基礎MLP(多層感知器)網路

In [6]:
```
import tensorflow as tf
import tensorflow.keras as keras
import tensorflow.keras.layers as layers
print(tf.__version__)
```

```
2.3.1
```

1.回歸任務

In [7]:
```
# 導入數據
(x_train, y_train), (x_test, y_test) = keras.datasets.boston_housing.load_data()
print(x_train.shape, ' ', y_train.shape)
print(x_test.shape, ' ', y_test.shape)
```

```
(404, 13)   (404,)
(102, 13)   (102,)
```

In [8]:
```
# 構建模型

model = keras.Sequential([
    layers.Dense(32, activation='sigmoid', input_shape=(13,)),
    layers.Dense(32, activation='sigmoid'),
    layers.Dense(32, activation='sigmoid'),
    layers.Dense(1)
])

# 配置模型
model.compile(optimizer=keras.optimizers.SGD(0.1),
              loss='mean_squared_error',  # keras.losses.mean_squared_error
              metrics=['mse'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 32)                448
_________________________________________________________________
dense_1 (Dense)              (None, 32)                1056
_________________________________________________________________
dense_2 (Dense)              (None, 32)                1056
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 33
=================================================================
Total params: 2,593
Trainable params: 2,593
Non-trainable params: 0
_________________________________________________________________
```

In [9]:
```
# 訓練
model.fit(x_train, y_train, batch_size=50, epochs=50, validation_split=0.1, verbo
```

```
Epoch 1/50
8/8 [==============================] - 0s 17ms/step - loss: 363.0231 - mse: 36
3.0231 - val_loss: 43.0516 - val_mse: 43.0516
Epoch 2/50
8/8 [==============================] - 0s 3ms/step - loss: 93.0425 - mse: 93.04
25 - val_loss: 42.7334 - val_mse: 42.7334
Epoch 3/50
8/8 [==============================] - 0s 4ms/step - loss: 89.9174 - mse: 89.91
74 - val_loss: 43.9873 - val_mse: 43.9873
Epoch 4/50
8/8 [==============================] - 0s 3ms/step - loss: 89.5870 - mse: 89.58
70 - val_loss: 44.6338 - val_mse: 44.6338
Epoch 5/50
8/8 [==============================] - 0s 3ms/step - loss: 94.3780 - mse: 94.37
80 - val_loss: 48.0469 - val_mse: 48.0469
Epoch 6/50
8/8 [==============================] - 0s 3ms/step - loss: 89.1945 - mse: 89.19
45 - val_loss: 43.0508 - val_mse: 43.0508
Epoch 7/50
8/8 [==============================] - 0s 4ms/step - loss: 89.4859 - mse: 89.48
59 - val_loss: 48.2906 - val_mse: 48.2906
Epoch 8/50
8/8 [==============================] - 0s 4ms/step - loss: 91.3263 - mse: 91.32
63 - val_loss: 70.5233 - val_mse: 70.5233
Epoch 9/50
8/8 [==============================] - 0s 3ms/step - loss: 91.3456 - mse: 91.34
56 - val_loss: 45.4455 - val_mse: 45.4455
Epoch 10/50
8/8 [==============================] - 0s 3ms/step - loss: 95.7316 - mse: 95.73
16 - val_loss: 59.9354 - val_mse: 59.9354
Epoch 11/50
8/8 [==============================] - 0s 3ms/step - loss: 101.1677 - mse: 101.
1677 - val_loss: 47.3810 - val_mse: 47.3810
Epoch 12/50
8/8 [==============================] - 0s 3ms/step - loss: 93.2472 - mse: 93.24
72 - val_loss: 112.8072 - val_mse: 112.8072
Epoch 13/50
8/8 [==============================] - 0s 3ms/step - loss: 94.5911 - mse: 94.59
11 - val_loss: 42.6619 - val_mse: 42.6619
Epoch 14/50
8/8 [==============================] - 0s 3ms/step - loss: 90.1367 - mse: 90.13
67 - val_loss: 43.8587 - val_mse: 43.8587
Epoch 15/50
8/8 [==============================] - 0s 3ms/step - loss: 91.1122 - mse: 91.11
22 - val_loss: 47.8243 - val_mse: 47.8243
Epoch 16/50
8/8 [==============================] - 0s 3ms/step - loss: 92.5957 - mse: 92.59
57 - val_loss: 45.0856 - val_mse: 45.0856
Epoch 17/50
8/8 [==============================] - 0s 3ms/step - loss: 90.1292 - mse: 90.12
92 - val_loss: 46.0044 - val_mse: 46.0044
Epoch 18/50
8/8 [==============================] - 0s 3ms/step - loss: 101.1956 - mse: 101.
```

```
1956 - val_loss: 42.7588 - val_mse: 42.7588
Epoch 19/50
8/8 [==============================] - 0s 4ms/step - loss: 91.5063 - mse: 91.50
63 - val_loss: 97.8648 - val_mse: 97.8648
Epoch 20/50
8/8 [==============================] - 0s 3ms/step - loss: 102.4639 - mse: 102.
4639 - val_loss: 42.5259 - val_mse: 42.5259
Epoch 21/50
8/8 [==============================] - 0s 3ms/step - loss: 89.8824 - mse: 89.88
24 - val_loss: 50.1589 - val_mse: 50.1589
Epoch 22/50
8/8 [==============================] - 0s 4ms/step - loss: 93.0267 - mse: 93.02
67 - val_loss: 74.6408 - val_mse: 74.6408
Epoch 23/50
8/8 [==============================] - 0s 4ms/step - loss: 95.5160 - mse: 95.51
60 - val_loss: 58.1527 - val_mse: 58.1527
Epoch 24/50
8/8 [==============================] - 0s 3ms/step - loss: 91.6176 - mse: 91.61
77 - val_loss: 47.3648 - val_mse: 47.3648
Epoch 25/50
8/8 [==============================] - 0s 3ms/step - loss: 88.6908 - mse: 88.69
08 - val_loss: 52.6281 - val_mse: 52.6281
Epoch 26/50
8/8 [==============================] - 0s 3ms/step - loss: 96.8344 - mse: 96.83
44 - val_loss: 113.9869 - val_mse: 113.9869
Epoch 27/50
8/8 [==============================] - 0s 3ms/step - loss: 104.4772 - mse: 104.
4772 - val_loss: 44.8500 - val_mse: 44.8500
Epoch 28/50
8/8 [==============================] - 0s 3ms/step - loss: 90.2103 - mse: 90.21
03 - val_loss: 44.4610 - val_mse: 44.4610
Epoch 29/50
8/8 [==============================] - 0s 3ms/step - loss: 89.8695 - mse: 89.86
95 - val_loss: 44.6823 - val_mse: 44.6823
Epoch 30/50
8/8 [==============================] - 0s 3ms/step - loss: 94.8823 - mse: 94.88
23 - val_loss: 46.0264 - val_mse: 46.0264
Epoch 31/50
8/8 [==============================] - 0s 3ms/step - loss: 90.0821 - mse: 90.08
21 - val_loss: 48.4359 - val_mse: 48.4359
Epoch 32/50
8/8 [==============================] - 0s 3ms/step - loss: 90.8036 - mse: 90.80
36 - val_loss: 44.9337 - val_mse: 44.9337
Epoch 33/50
8/8 [==============================] - 0s 3ms/step - loss: 90.2533 - mse: 90.25
33 - val_loss: 45.8298 - val_mse: 45.8298
Epoch 34/50
8/8 [==============================] - 0s 3ms/step - loss: 90.9234 - mse: 90.92
34 - val_loss: 54.6863 - val_mse: 54.6863
Epoch 35/50
8/8 [==============================] - 0s 3ms/step - loss: 92.6804 - mse: 92.68
04 - val_loss: 51.2015 - val_mse: 51.2015
Epoch 36/50
8/8 [==============================] - 0s 3ms/step - loss: 91.6597 - mse: 91.65
97 - val_loss: 50.2838 - val_mse: 50.2838
Epoch 37/50
8/8 [==============================] - 0s 3ms/step - loss: 89.6119 - mse: 89.61
```

```
19 - val_loss: 65.0017 - val_mse: 65.0017
Epoch 38/50
8/8 [==============================] - 0s 3ms/step - loss: 93.0540 - mse: 93.05
40 - val_loss: 45.3023 - val_mse: 45.3023
Epoch 39/50
8/8 [==============================] - 0s 3ms/step - loss: 91.7353 - mse: 91.73
53 - val_loss: 49.3627 - val_mse: 49.3627
Epoch 40/50
8/8 [==============================] - 0s 3ms/step - loss: 93.1663 - mse: 93.16
63 - val_loss: 79.8133 - val_mse: 79.8133
Epoch 41/50
8/8 [==============================] - 0s 3ms/step - loss: 97.0535 - mse: 97.05
35 - val_loss: 50.4758 - val_mse: 50.4758
Epoch 42/50
8/8 [==============================] - 0s 3ms/step - loss: 92.9031 - mse: 92.90
31 - val_loss: 46.7236 - val_mse: 46.7236
Epoch 43/50
8/8 [==============================] - 0s 3ms/step - loss: 99.2840 - mse: 99.28
40 - val_loss: 50.0669 - val_mse: 50.0669
Epoch 44/50
8/8 [==============================] - 0s 3ms/step - loss: 96.1300 - mse: 96.13
00 - val_loss: 54.9243 - val_mse: 54.9243
Epoch 45/50
8/8 [==============================] - 0s 3ms/step - loss: 94.0857 - mse: 94.08
57 - val_loss: 75.8688 - val_mse: 75.8688
Epoch 46/50
8/8 [==============================] - 0s 3ms/step - loss: 94.2585 - mse: 94.25
85 - val_loss: 63.2968 - val_mse: 63.2968
Epoch 47/50
8/8 [==============================] - 0s 4ms/step - loss: 91.0156 - mse: 91.01
56 - val_loss: 42.7777 - val_mse: 42.7777
Epoch 48/50
8/8 [==============================] - 0s 3ms/step - loss: 92.6769 - mse: 92.67
69 - val_loss: 49.7331 - val_mse: 49.7331
Epoch 49/50
8/8 [==============================] - 0s 3ms/step - loss: 92.4118 - mse: 92.41
18 - val_loss: 42.6793 - val_mse: 42.6793
Epoch 50/50
8/8 [==============================] - 0s 3ms/step - loss: 90.7955 - mse: 90.79
55 - val_loss: 44.7894 - val_mse: 44.7894
```

Out[9]:
```
<tensorflow.python.keras.callbacks.History at 0x2196b0501c8>
```

In [10]:
```
result = model.evaluate(x_test, y_test)
```

```
4/4 [==============================] - 0s 997us/step - loss: 92.3133 - mse: 92.
3133
```

In [11]:
```
print(model.metrics_names)
print(result)
```

```
['loss', 'mse']
[92.31330108642578, 92.31330108642578]
```

2.分類任務

In [14]:
```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

whole_data = load_breast_cancer()
x_data = whole_data.data
y_data = whole_data.target

x_train, x_test, y_train, y_test = train_test_split(x_data, y_data, test_size=0.3

print(x_train.shape, ' ', y_train.shape)
print(x_test.shape, ' ', y_test.shape)
```

```
(398, 30)   (398,)
(171, 30)   (171,)
```

In [15]:
```
# 构建模型
model = keras.Sequential([
    layers.Dense(32, activation='relu', input_shape=(30,)),
    layers.Dense(32, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])

model.compile(optimizer=keras.optimizers.Adam(),
              loss=keras.losses.binary_crossentropy,
              metrics=['accuracy'])
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_4 (Dense)              (None, 32)                992
_________________________________________________________________
dense_5 (Dense)              (None, 32)                1056
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 33
=================================================================
Total params: 2,081
Trainable params: 2,081
Non-trainable params: 0
_________________________________________________________________
```

In [16]:
```
model.fit(x_train, y_train, batch_size=64, epochs=10, verbose=1)
```

```
Epoch 1/10
7/7 [==============================] - 0s 999us/step - loss: 21.8088 - accurac
y: 0.3970
Epoch 2/10
7/7 [==============================] - 0s 1ms/step - loss: 11.2308 - accuracy:
0.3693
Epoch 3/10
7/7 [==============================] - 0s 1ms/step - loss: 10.0044 - accuracy:
0.2990
Epoch 4/10
7/7 [==============================] - 0s 1ms/step - loss: 6.0738 - accuracy:
0.2940
Epoch 5/10
7/7 [==============================] - 0s 1ms/step - loss: 4.8097 - accuracy:
0.3090
Epoch 6/10
7/7 [==============================] - 0s 1ms/step - loss: 3.3228 - accuracy:
0.3216
Epoch 7/10
7/7 [==============================] - 0s 1ms/step - loss: 2.5263 - accuracy:
0.4347
Epoch 8/10
7/7 [==============================] - 0s 1ms/step - loss: 2.1083 - accuracy:
0.5075
Epoch 9/10
7/7 [==============================] - 0s 1ms/step - loss: 1.7401 - accuracy:
0.5729
Epoch 10/10
7/7 [==============================] - 0s 1ms/step - loss: 1.4697 - accuracy:
0.6583
```

Out[16]:
```
<tensorflow.python.keras.callbacks.History at 0x2197b5cc188>
```

In [17]:
```
model.evaluate(x_test, y_test)
```

```
6/6 [==============================] - 0s 997us/step - loss: 0.7271 - accuracy:
0.8246
```

Out[17]:
```
[0.7271407246589661, 0.8245614171028137]
```

In [18]:
```
print(model.metrics_names)
```

```
['loss', 'accuracy']
```

In [ ]:
```

```