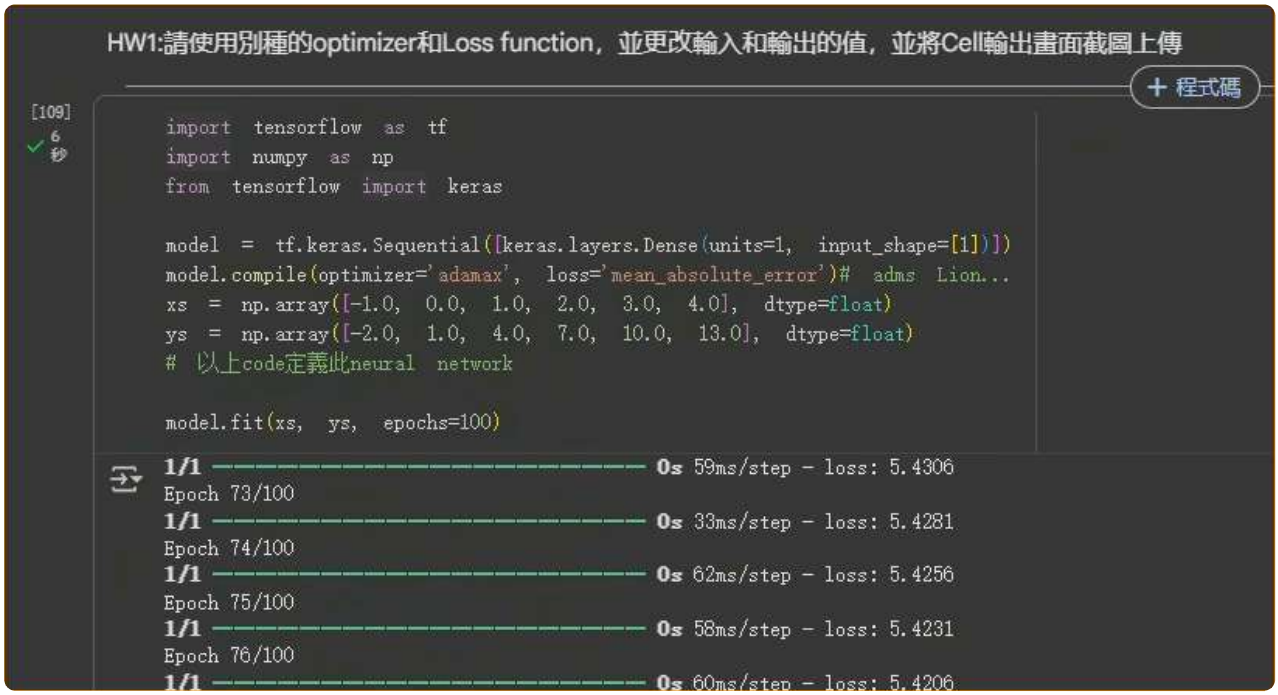
1.請嘗試修改xs與ys元素值並經過神經網路訓練求出y=ax+b線性回歸方程式之a、b最佳解。



HW1:請使用別種的optimizer和Loss function，並更改輸入和輸出的值，並將Cell輸出畫面截圖上傳

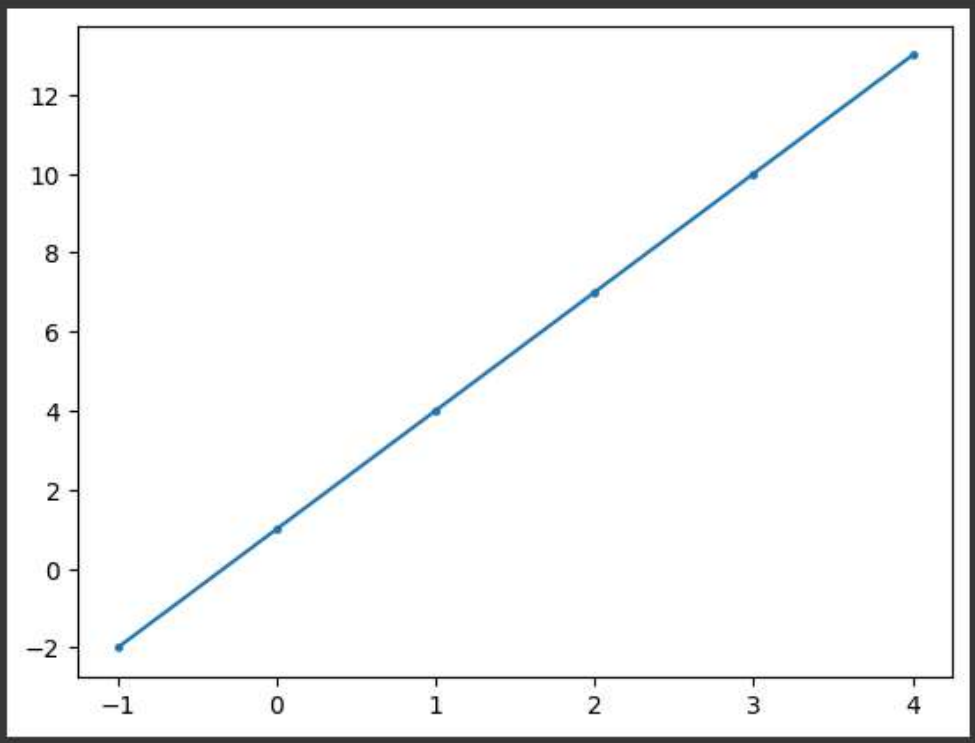
+ 程式碼

[109] ✓ 6 秒

```
import tensorflow as tf
import numpy as np
from tensorflow import keras

model = tf.keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])
model.compile(optimizer='adamax', loss='mean_absolute_error')# adms Lion...
xs = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0], dtype=float)
ys = np.array([-2.0, 1.0, 4.0, 7.0, 10.0, 13.0], dtype=float)
# 以上code定義此neural network

model.fit(xs, ys, epochs=100)
```

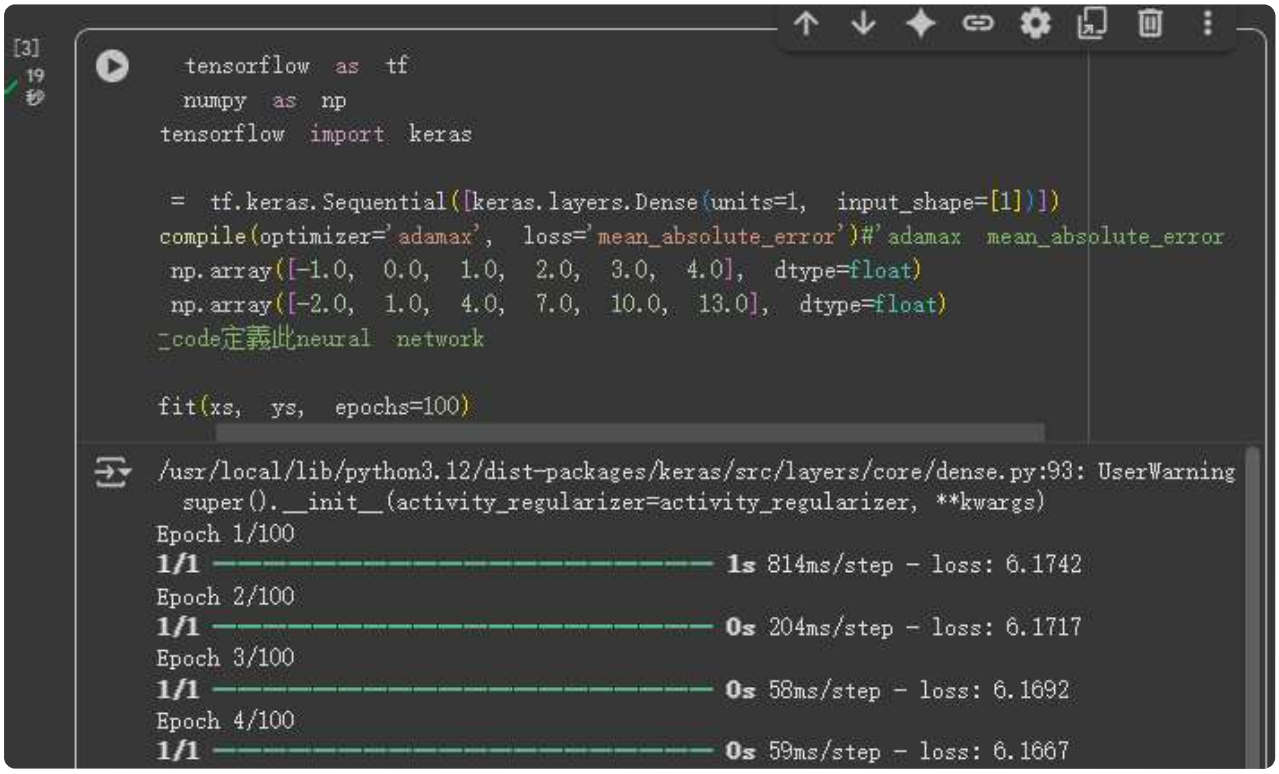
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 59ms/step - loss: 5.4306
Epoch 73/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step - loss: 5.4281
Epoch 74/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 62ms/step - loss: 5.4256
Epoch 75/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step - loss: 5.4231
Epoch 76/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 60ms/step - loss: 5.4206
```

```
[ ]
model.layers[0].get_weights()

[array([[1.8068321]], dtype=float32), array([0.09852193], dtype=float32)]

[ ]
import matplotlib.pyplot as plt
from scipy import stats
slope, intercept, r, p, std_err = stats.linregress(xs, ys)

def myfunc(v):
  return slope * v + intercept

mymodel = list(map(myfunc, xs))
plt.scatter(xs,ys,6)
plt.plot(xs, mymodel)
plt.show()
```




```
[ ]
print(model.predict(x=np.array([10.0])))

1/1 ———————————————————— 0s 165ms/step
[[18.166843]]
```

2.請更換優化器(optimizer)和損失函數(Loss function)。



```
[3]
19
秒
  tensorflow as tf
  numpy as np
tensorflow import keras

 = tf.keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])
compile(optimizer='adamax', loss='mean_absolute_error')#'adamax mean_absolute_error
 np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0], dtype=float)
 np.array([-2.0, 1.0, 4.0, 7.0, 10.0, 13.0], dtype=float)
_code定義此neural network

fit(xs, ys, epochs=100)

/usr/local/lib/python3.12/dist-packages/keras/src/layers/core/dense.py:93: UserWarning
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 814ms/step - loss: 6.1742
Epoch 2/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 204ms/step - loss: 6.1717
Epoch 3/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step - loss: 6.1692
Epoch 4/100
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 59ms/step - loss: 6.1667
```

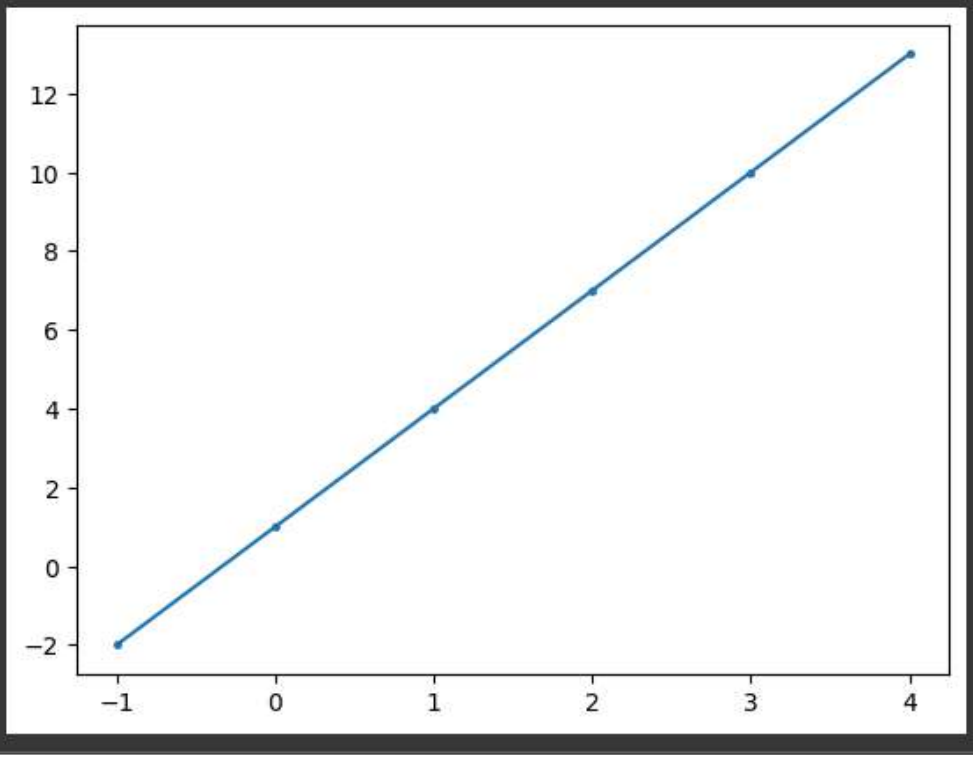
[4] 0秒

```
model.layers[0].get_weights()
```

```
[array([[0.09587549]], dtype=float32), array([0.10000002], dtype=float32)]
```

[5] 0秒

```
import matplotlib.pyplot as plt
from scipy import stats
slope, intercept, r, p, std_err = stats.linregress(xs, ys)

def myfunc(v):
    return slope * v + intercept

mymodel = list(map(myfunc, xs))
plt.scatter(xs,ys,6)
plt.plot(xs, mymodel)
plt.show()
```



3.根據最佳解預測當x=10時，y的值為何?

[6] 0秒

```
print(model.predict(x=np.array([10.0])))
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 130ms/step
[[1.0587549]]
```