A Single Neuron

學習了神經網路的建構模組：線性單元。了解到，僅包含一個線性單元的模型就能將線性函數擬合到資料集（相當於線性迴歸）。在本練習中，將建立一個線性模型，並練習在 Keras 中運用這些模型。

設定

```
# Setup plotting
import matplotlib.pyplot as plt

plt.style.use('seaborn-whitegrid')
# Set Matplotlib defaults
plt.rc('figure', autolayout=True)
plt.rc('axes', labelweight='bold', labelsize='large',
       titleweight='bold', titlesize=18, titlepad=10)

# Setup feedback system
from learntools.core import binder
binder.bind(globals())
from learntools.deep_learning_intro.ex1 import *
```

The Red Wine Quality 數據集包含約 1600 款葡萄牙紅葡萄酒的理化測量數據。此外，還包含每種葡萄酒的盲品品質評級。

Import pandas，讀檔

[5]:
```
import pandas as pd

red_wine = pd.read_csv('../input/dl-course-data/red-wine.csv')
red_wine.head()
```

[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

您可以使用 shape 屬性來取得資料框（或 Numpy 陣列）的行數和列數。

```
red_wine.shape # (rows, columns)
```

```
[6]: (1599, 12)
```

1) Input shape(輸入 shape)

我們能多好地根據理化測量結果預測葡萄酒的感知品質？

目標是 'quality'，其餘列是特徵。執行此任務時，如何設定 Keras 模型的 input_shape 參數？

```
[9]: # YOUR CODE HERE
input_shape = [11]
```

2) Define a linear model(定義線性模型)

現在定義一個適合此任務的線性模型。注意模型應該有多少個輸入和輸出。

```
from tensorflow import keras
from tensorflow.keras import layers

# YOUR CODE HERE
model = keras.Sequential([
    layers.Dense(units=1,input_shape=[11])
])

# Check your answer
q_2.check()
```

3） Look at the weights(查看權重)

在 Keras 內部，它以張量(Tensors)表示神經網路的權重(weights)。張量(Tensors)本質上是 TensorFlow 版本的 Numpy 數組，但有一些區別，使其更適合深度學習。其中最重要的區別之一是張量與 GPU 和 TPU 加速器(accelerators)相容(compatible)。事實上，TPU 是專為張量計算(tensor computations)而設計的。

模型的權重以張量列表的形式保存在其 weights 屬性中。取得您上面定義的模型的權重。（如果需要，可以使用以下命令顯示權重：
print("Weights\n{}\n\nBias\n{}".format(w, b)))

```
w, b = model.weights
print("Weights\n{}\nBias\n{}".format(w,b))

# Check your answer
q_3.check()
```

```
Weights
<tf.Variable 'dense/kernel:0' shape=(11, 1) dtype=float32, numpy=
array([[ 0.18608642],
       [ 0.5240769 ],
       [-0.15714455],
       [ 0.42611665],
       [ 0.16991639],
       [-0.30281886],
       [-0.09536821],
       [-0.09713954],
       [-0.27164117],
       [ 0.4787597 ],
       [ 0.24558747]], dtype=float32)>
Bias
<tf.Variable 'dense/bias:0' shape=(1,) dtype=float32, numpy=array([0.], dtype=float32)>
```

你注意到每個輸入都有一個權重（以及一個偏差(bias)）嗎？不過請注意，權重的值似乎沒有任何規律。在模型訓練之前，權重被設定為隨機數（偏差設定為 0.0）。神經網路透過尋找更優的權重值來學習。

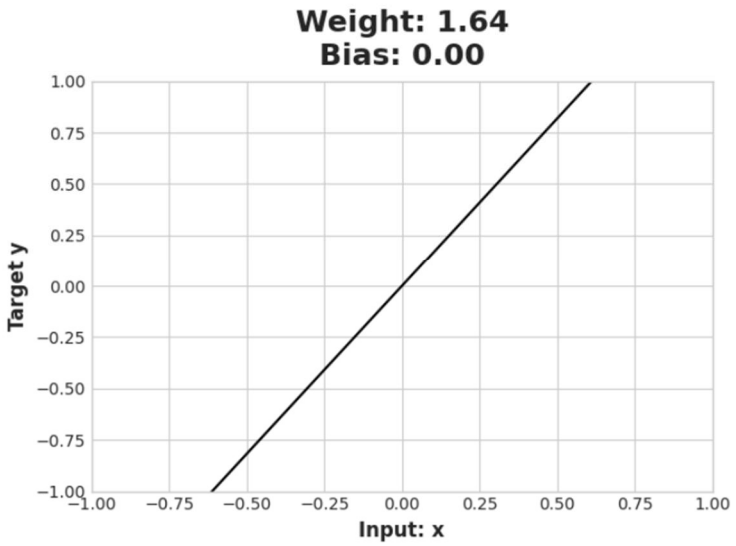
Optional: Plot the output of an untrained linear model(繪製未經訓練的線性模型的輸出)

在第 5 課中，我們將要研究的問題是迴歸(regression)問題，其目標是預測某個數值目標。迴歸問題類似於「曲線擬合」("curve-fitting")問題：我們試著找出一條與資料最匹配的曲線。讓我們來看看線性模型產生的「曲線」("curve")。（你可能已經猜到了，它是一條直線！）

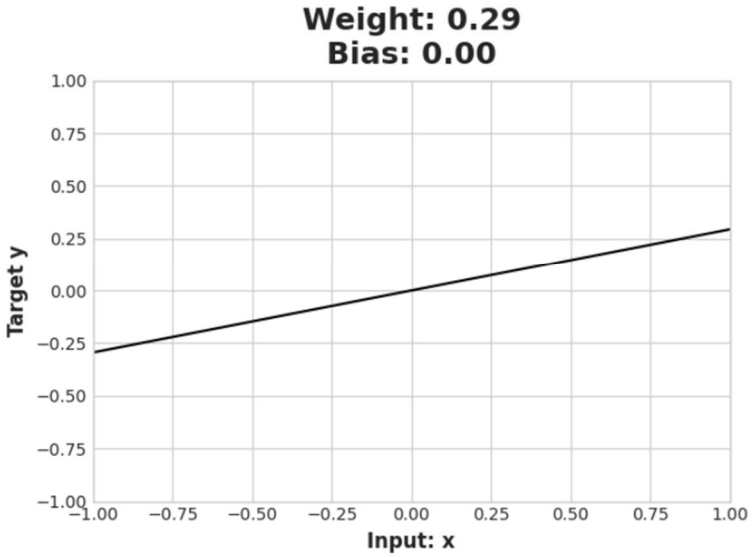
在訓練之前，模型的權重是隨機設定的。運行下面的 cell，看看隨機初始化產生的不同直線。

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

model = keras.Sequential([
    layers.Dense(1, input_shape=[1]),
])

x = tf.linspace(-1.0, 1.0, 100)
y = model.predict(x)

plt.figure(dpi=100)
plt.plot(x, y, 'k')
plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.xlabel("Input: x")
plt.ylabel("Target y")
w, b = model.weights # you could also use model.get_weights() here
plt.title("Weight: {:0.2f}\nBias: {:0.2f}".format(w[0][0], b[0]))
plt.show()
```

```
4/4 [==============================] - 0s 3ms/step
```




```
4/4 [==============================] - 0s 2ms/step
```




```
4/4 [==============================] - 0s 2ms/step
```