

```
import tensorflow as tf
from tensorflow import keras
```

```
tf.__version__
```

```
'2.4.0'
```

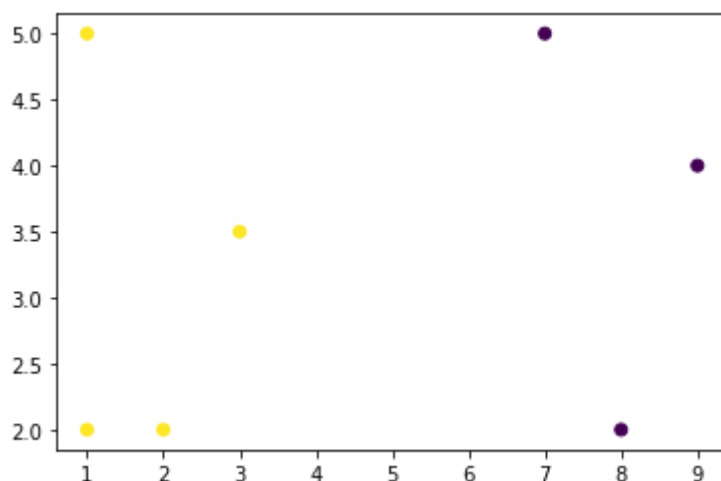
```
import numpy as np
import matplotlib.pyplot as plt
```

```
x_data = np.array([[1, 2], [1, 5], [2, 2], [3, 3.5], [7, 5], [9, 4], [8, 2]])
y_data = [1, 1, 1, 1, 0, 0, 0]
```

```
x_data = np.array(x_data, dtype=np.float32)
y_data = np.array(y_data, dtype=np.float32)
```

```
plt.scatter(x_data[:, 0], x_data[:, 1], c=y_data)
```

<matplotlib.collections.PathCollection at 0x7f4123ac8400>



```
from tensorflow.keras import layers
from tensorflow.keras import activations
from tensorflow.keras import optimizers
from tensorflow.keras import models
```

```
model = models.Sequential()
model.add(layers.Dense(2, input_dim=2))
model.add(layers.Activation('tanh'))
model.add(layers.Dense(1))
model.add(layers.Activation('sigmoid'))
```

```
sgd = optimizers.SGD(lr=0.1)
model.compile(loss='binary_crossentropy', optimizer=sgd, metrics=['accuracy'])
```

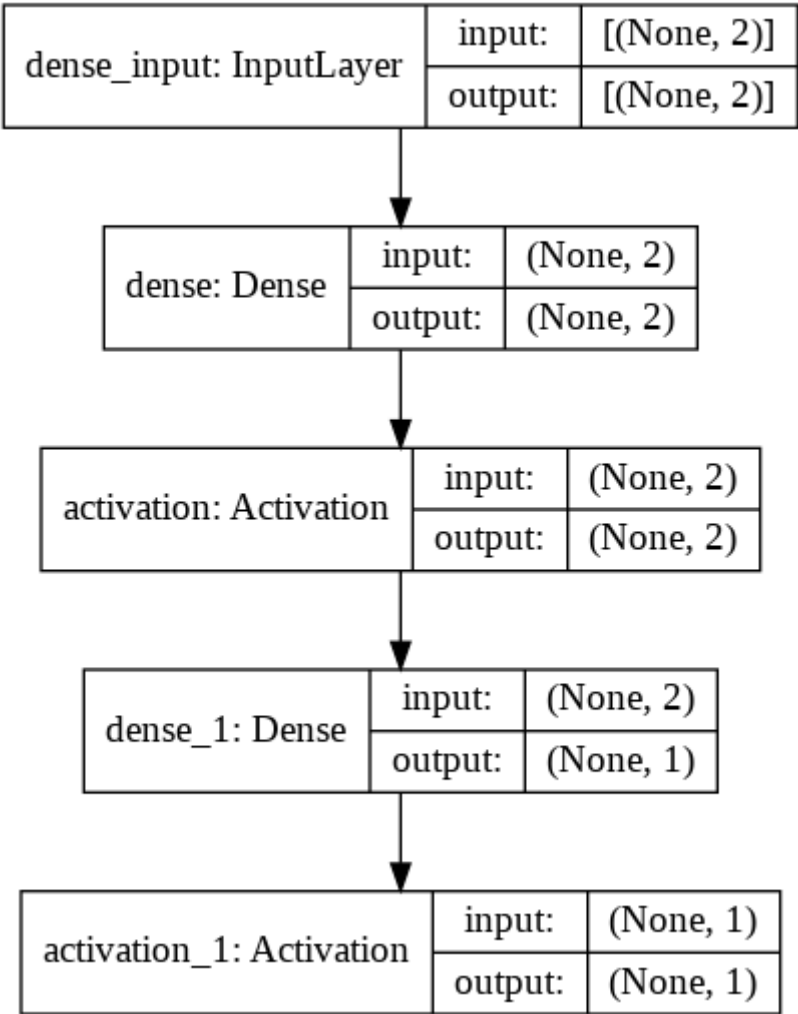
```
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 2)	6
activation (Activation)	(None, 2)	0
dense_1 (Dense)	(None, 1)	3
activation_1 (Activation)	(None, 1)	0

Total params: 9
Trainable params: 9
Non-trainable params: 0

```
from tensorflow.keras.utils import plot_model
plot_model(model, to_file='model_and.png', show_shapes=True)
```



```
history = model.fit(x_data, y_data, batch_size=1, epochs=500)
```

Epoch 467/500
7/7 [=====] - 0s 3ms/step - loss: 0.0016 - accuracy: 1.0000
Epoch 468/500
7/7 [=====] - 0s 2ms/step - loss: 0.0019 - accuracy: 1.0000
Epoch 469/500

```

/// [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 470/500
7/7 [=====] - 0s 3ms/step - loss: 0.0015 - accuracy: 1.0000
Epoch 471/500
7/7 [=====] - 0s 3ms/step - loss: 0.0017 - accuracy: 1.0000
Epoch 472/500
7/7 [=====] - 0s 2ms/step - loss: 0.0015 - accuracy: 1.0000
Epoch 473/500
7/7 [=====] - 0s 3ms/step - loss: 0.0015 - accuracy: 1.0000
Epoch 474/500
7/7 [=====] - 0s 2ms/step - loss: 0.0018 - accuracy: 1.0000
Epoch 475/500
7/7 [=====] - 0s 3ms/step - loss: 0.0018 - accuracy: 1.0000
Epoch 476/500
7/7 [=====] - 0s 2ms/step - loss: 0.0018 - accuracy: 1.0000
Epoch 477/500
7/7 [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 478/500
7/7 [=====] - 0s 3ms/step - loss: 0.0019 - accuracy: 1.0000
Epoch 479/500
7/7 [=====] - 0s 3ms/step - loss: 0.0017 - accuracy: 1.0000
Epoch 480/500
7/7 [=====] - 0s 4ms/step - loss: 0.0021 - accuracy: 1.0000
Epoch 481/500
7/7 [=====] - 0s 3ms/step - loss: 0.0017 - accuracy: 1.0000
Epoch 482/500
7/7 [=====] - 0s 2ms/step - loss: 0.0020 - accuracy: 1.0000
Epoch 483/500
7/7 [=====] - 0s 3ms/step - loss: 0.0017 - accuracy: 1.0000
Epoch 484/500
7/7 [=====] - 0s 2ms/step - loss: 0.0016 - accuracy: 1.0000
Epoch 485/500
7/7 [=====] - 0s 3ms/step - loss: 0.0018 - accuracy: 1.0000
Epoch 486/500
7/7 [=====] - 0s 2ms/step - loss: 0.0014 - accuracy: 1.0000
Epoch 487/500
7/7 [=====] - 0s 2ms/step - loss: 0.0014 - accuracy: 1.0000
Epoch 488/500
7/7 [=====] - 0s 2ms/step - loss: 0.0018 - accuracy: 1.0000
Epoch 489/500
7/7 [=====] - 0s 3ms/step - loss: 0.0018 - accuracy: 1.0000
Epoch 490/500
7/7 [=====] - 0s 5ms/step - loss: 0.0017 - accuracy: 1.0000
Epoch 491/500
7/7 [=====] - 0s 2ms/step - loss: 0.0018 - accuracy: 1.0000
Epoch 492/500
7/7 [=====] - 0s 2ms/step - loss: 0.0014 - accuracy: 1.0000
Epoch 493/500
7/7 [=====] - 0s 2ms/step - loss: 0.0013 - accuracy: 1.0000
Epoch 494/500
7/7 [=====] - 0s 4ms/step - loss: 0.0017 - accuracy: 1.0000
Epoch 495/500
7/7 [=====] - 0s 6ms/step - loss: 0.0020 - accuracy: 1.0000
Epoch 496/500

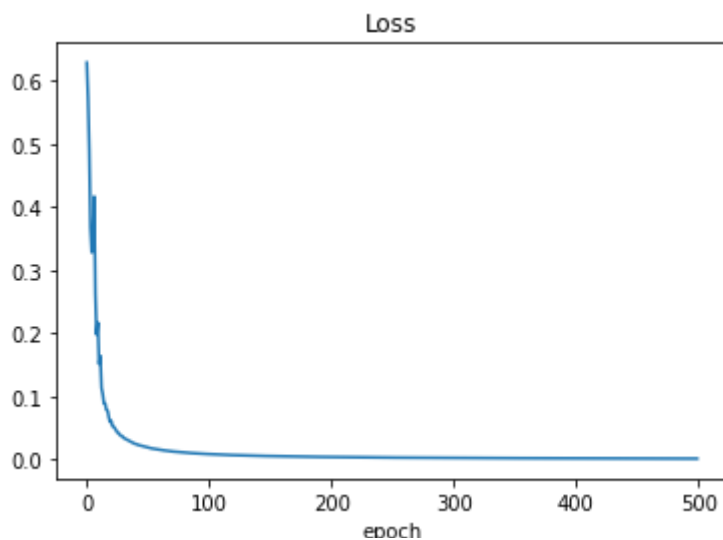
```

```

plt.plot(history.history['loss'])
plt.title('Loss')
plt.xlabel('epoch')

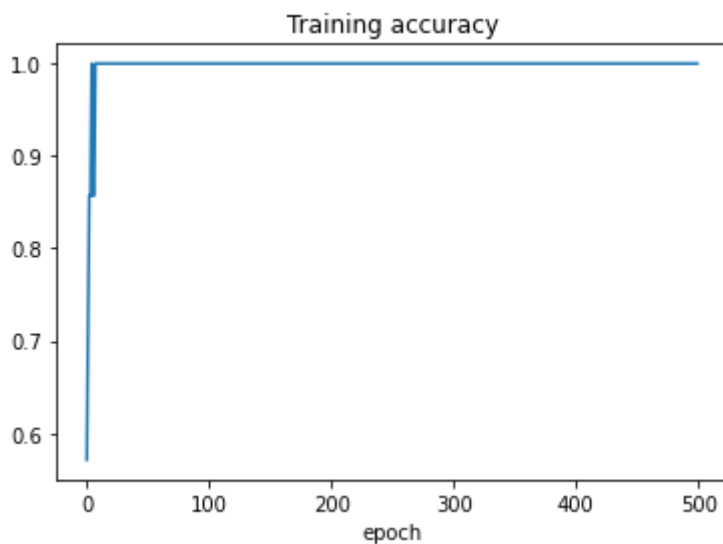
```

```
Text(0.5, 0, 'epoch')
```



```
plt.plot(history.history['accuracy'])  
plt.title('Training accuracy')  
plt.xlabel('epoch')
```

```
Text(0.5, 0, 'epoch')
```



```
hypothesis = model.predict(x_data)  
print(hypothesis)
```

```
[[0.9992269 ]  
 [0.9993099 ]  
 [0.997231  ]  
 [0.99893   ]  
 [0.00263828]  
 [0.00150385]  
 [0.00150359]]
```

```
predicted = hypothesis > 0.5  
print(predicted)
```

```
[[ True]  
 [ True]  
 [ True]  
 [ True]]
```

```
[False]  
[False]  
[False]]
```