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

```
tf.__version__
```

```
'2.4.0'
```

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

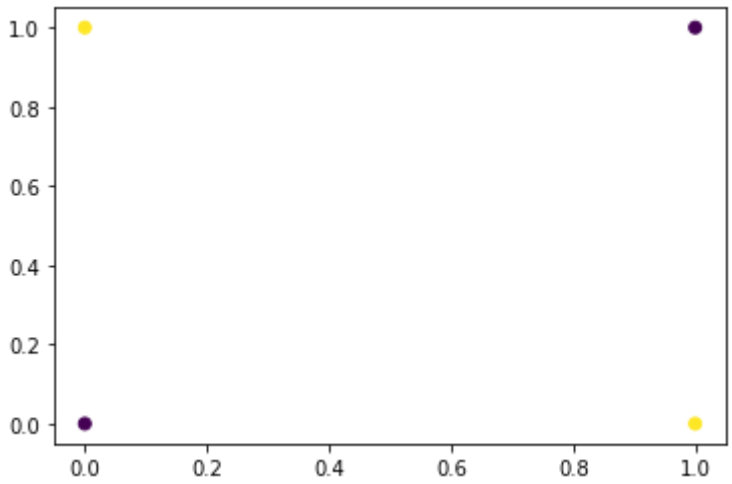
```
x_data = [[0, 0],
          [0, 1],
          [1, 0],
          [1, 1]]

y_data = [0,
          1,
          1,
          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 0x7fc7f57094e0>
```




```
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'))
```

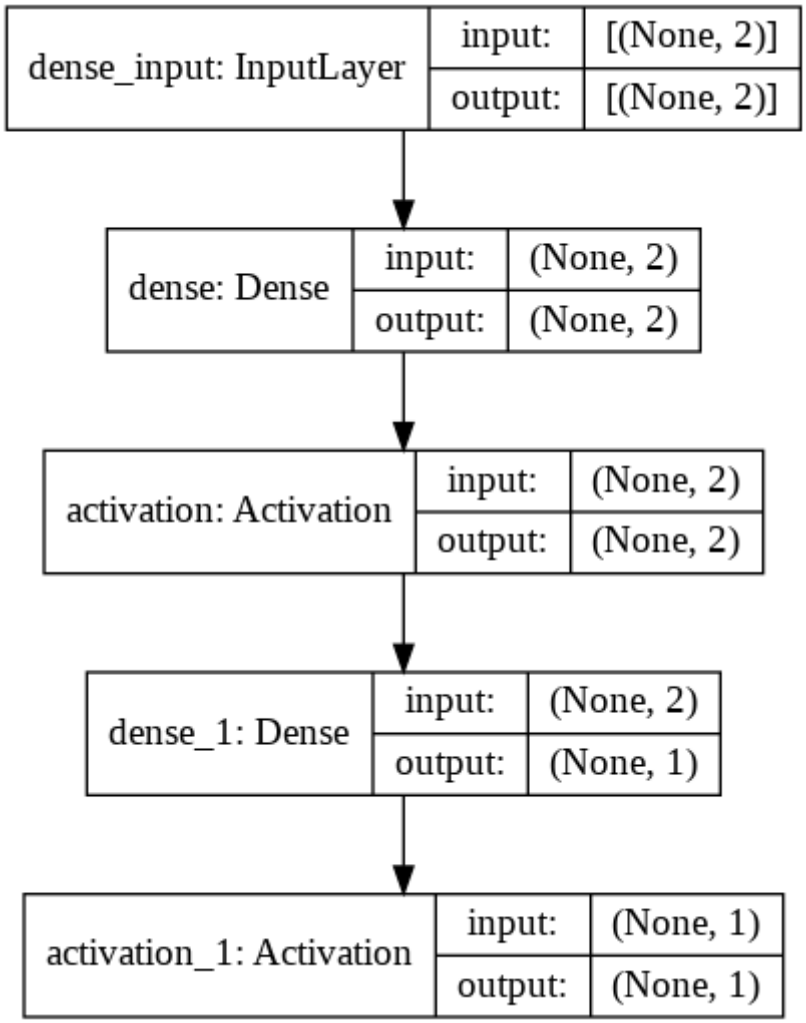
```
model.add(tf.keras.layers.Activation('sigmoid'))
```

```
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_xor.png', show_shapes=True)
```



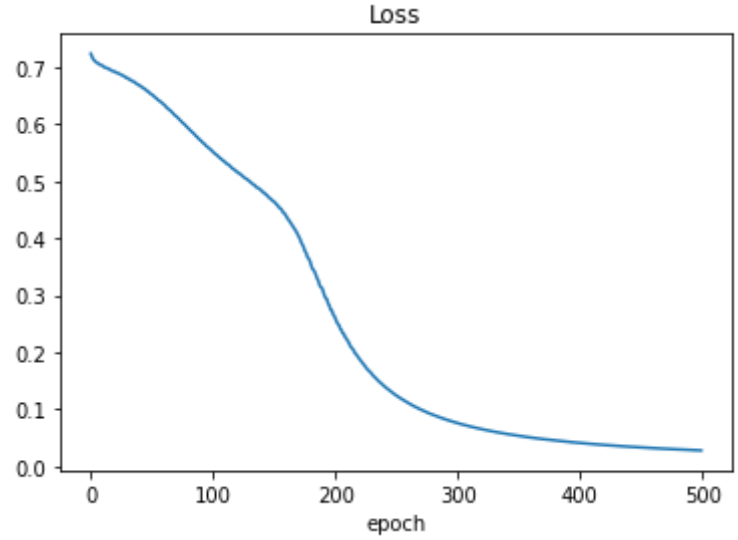
```
history = model.fit(x_data, y_data,  batch_size=1, epochs=500)
```

```
Epoch 467/500
4/4 [==============================] - 0s 8ms/step - loss: 0.0344 - accuracy: 1.0000
Epoch 468/500
4/4 [==============================] - 0s 6ms/step - loss: 0.0323 - accuracy: 1.0000
Epoch 469/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0307 - accuracy: 1.0000
Epoch 470/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0306 - accuracy: 1.0000
Epoch 471/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0339 - accuracy: 1.0000
Epoch 472/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0348 - accuracy: 1.0000
Epoch 473/500
4/4 [==============================] - 0s 6ms/step - loss: 0.0347 - accuracy: 1.0000
Epoch 474/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0286 - accuracy: 1.0000
Epoch 475/500
4/4 [==============================] - 0s 3ms/step - loss: 0.0344 - accuracy: 1.0000
Epoch 476/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0260 - accuracy: 1.0000
Epoch 477/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0270 - accuracy: 1.0000
Epoch 478/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0278 - accuracy: 1.0000
Epoch 479/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0258 - accuracy: 1.0000
Epoch 480/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0256 - accuracy: 1.0000
Epoch 481/500
4/4 [==============================] - 0s 6ms/step - loss: 0.0255 - accuracy: 1.0000
Epoch 482/500
4/4 [==============================] - 0s 3ms/step - loss: 0.0322 - accuracy: 1.0000
Epoch 483/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0277 - accuracy: 1.0000
Epoch 484/500
4/4 [==============================] - 0s 5ms/step - loss: 0.0249 - accuracy: 1.0000
Epoch 485/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0309 - accuracy: 1.0000
Epoch 486/500
4/4 [==============================] - 0s 3ms/step - loss: 0.0270 - accuracy: 1.0000
Epoch 487/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0251 - accuracy: 1.0000
Epoch 488/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0301 - accuracy: 1.0000
Epoch 489/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0245 - accuracy: 1.0000
Epoch 490/500
4/4 [==============================] - 0s 5ms/step - loss: 0.0267 - accuracy: 1.0000
Epoch 491/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0244 - accuracy: 1.0000
Epoch 492/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0324 - accuracy: 1.0000
Epoch 493/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0264 - accuracy: 1.0000
Epoch 494/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0308 - accuracy: 1.0000
Epoch 495/500
4/4 [==============================] - 0s 3ms/step - loss: 0.0266 - accuracy: 1.0000
Epoch 496/500
```

```
4/4 [==============================] - 0s 2ms/step - loss: 0.0293 - accuracy: 1.0000
```

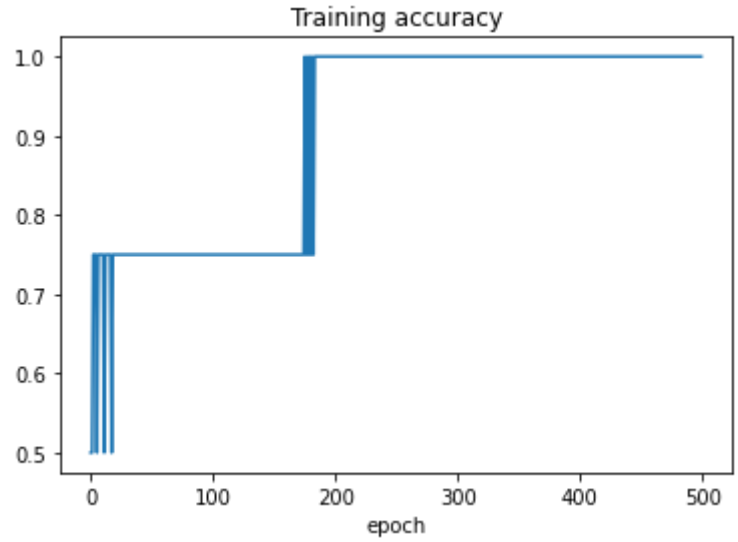
```
plt.plot(history.history['loss'])
plt.title('Loss')
plt.xlabel('epoch')
```

```
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)
```

```
[[0.03231364]
 [0.98077446]
 [0.98058975]
 [0.03650239]]
```

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

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