In [1]:

```
from tensorflow import keras
```

In [2]:

```
import numpy as np
```

In [3]:

```
#xor data
x_data = [
    [0,0],
    [0,1],
    [1,0],
    [1,1]
]

y_data = [
    [0],
    [1],
    [1],
    [0]
]
```

In [4]:

```
x_data = np.array(x_data)
y_data = np.array(y_data)
```

In [5]:

```
print(x_data.shape)
```

```
(4, 2)
```

In [6]:

```
model= keras.Sequential()
```

In [7]:

```
model.add(keras.layers.Dense(32,activation = "sigmoid",input_shape = (2,)))
model.add(keras.layers.Dense(1,activation = "sigmoid"))
```

In [8]:

```
optimizer = keras.optimizers.SGD(lr=0.1)
#0.001, 0.01
model.compile(optimizer=optimizer, loss="binary_crossentropy", metrics=['accuracy'])
```

In [9]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense (Dense)                (None, 32)                96        
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 33        
=================================================================
Total params: 129
Trainable params: 129
Non-trainable params: 0
_________________________________________________________________
```

In [14]:

```
model.fit(x_data, y_data, batch_size=4, epochs=10000)
```

```
Epoch 9992/10000
1/1 [==============================] - 0s 3ms/step - loss: 0.0145 - accuracy: 1.0000
Epoch 9993/10000
1/1 [==============================] - 0s 4ms/step - loss: 0.0145 - accuracy: 1.0000
Epoch 9994/10000
1/1 [==============================] - 0s 2ms/step - loss: 0.0144 - accuracy: 1.0000
Epoch 9995/10000
1/1 [==============================] - 0s 997us/step - loss: 0.0144 - accuracy: 1.0000
Epoch 9996/10000
1/1 [==============================] - 0s 3ms/step - loss: 0.0144 - accuracy: 1.0000
Epoch 9997/10000
1/1 [==============================] - 0s 3ms/step - loss: 0.0144 - accuracy: 1.0000
Epoch 9998/10000
1/1 [==============================] - 0s 10ms/step - loss: 0.0144 - accuracy: 1.0000
Epoch 9999/10000
1/1 [==============================] - 0s 999us/step - loss: 0.0144 - accuracy: 1.0000
Epoch 10000/10000
1/1 [==============================] - 0s 2ms/step - loss: 0.0144 - accuracy: 1.0000
```

Out[14]:

```
<tensorflow.python.keras.callbacks.History at 0x13b5f9e5700>
```

In [16]:

```
predict=model.predict(x_data)
print(np.round(predict))
```

```
[[0.]
 [1.]
 [1.]
 [0.]]
```

In [1]:

```
# tensorflow
# TensorFlow and tf.keras
import tensorflow as tf

# Helper libraries
import numpy as np
import matplotlib.pyplot as plt

print(tf.__version__)
```

```
2.3.1
```

In [15]:

```
from keras.models import Sequential
from keras.layers import Dense
model = Sequential()
model.add(Dense(2, input_dim=3, activation='relu')) # 2*(3+1) op*(ip+1)
model.add(Dense(2, activation='relu'))  # 2*(2+1)
model.add(Dense(1, activation='sigmoid')) # 1*(2+1)
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_2 (Dense)              (None, 2)                 8         
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 6         
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 3         
=================================================================
Total params: 17
Trainable params: 17
Non-trainable params: 0
_________________________________________________________________
```

In [17]:

```
model = Sequential()
hidden_layer_1=Dense(2, input_dim=3, activation='relu')
model.add(hidden_layer_1)
hidden_layer_2=Dense(2, activation='relu')
model.add(hidden_layer_2)
output_layer=Dense(1, activation='sigmoid')
model.add(output_layer)
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_5 (Dense)              (None, 2)                 8         
_________________________________________________________________
dense_6 (Dense)              (None, 2)                 6         
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 3         
=================================================================
Total params: 17
Trainable params: 17
Non-trainable params: 0
_________________________________________________________________
```

In [18]:

```
print("*******Details of Hidden Layer 1*******")
print("hidden_layer_1 : Config")
print(hidden_layer_1.get_config())
print("hidden_layer_1: Weights & Bias")
print(hidden_layer_1.get_weights())
```

```
*******Details of Hidden Layer 1*******
hidden_layer_1 : Config
{'name': 'dense_5', 'trainable': True, 'batch_input_shape': (None, 3), 'dtype': 'float32', 'units': 2, 'act
ivation': 'relu', 'use_bias': True, 'kernel_initializer': {'class_name': 'GlorotUniform', 'config': {'see
d': None}}, 'bias_initializer': {'class_name': 'Zeros', 'config': {}}, 'kernel_regularizer': None, 'bias_re
gularizer': None, 'activity_regularizer': None, 'kernel_constraint': None, 'bias_constraint': None}
hidden_layer_1: Weights & Bias
[array([[0.00473118, 0.49211693],
       [0.6397805 , 1.0157771 ],
       [0.21997285, 0.85423636]], dtype=float32), array([0., 0.], dtype=float32)]
```

In [ ]:

```
```