In [34]:
```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import normalize
from tensorflow import keras
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.utils import to_categorical
import numpy as np
```

In [35]:
```
X, y = load_iris(return_X_y=True)
X = normalize(X, axis=0)
y = to_categorical(y, num_classes=3)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [36]:
```
print(f'X.shape: {X.shape}, y.shape {y.shape}')
```

```
X.shape: (150, 4), y.shape (150, 3)
```

In [37]:
```
model=Sequential()
model.add(Dense(1000,input_dim=4,activation='relu'))
model.add(Dense(500,activation='relu'))
model.add(Dense(300,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(3,activation='softmax'))
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

In [38]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_4 (Dense)              (None, 1000)              5000
_________________________________________________________________
dense_5 (Dense)              (None, 500)               500500
_________________________________________________________________
dense_6 (Dense)              (None, 300)               150300
_________________________________________________________________
dropout_1 (Dropout)          (None, 300)               0
_________________________________________________________________
dense_7 (Dense)              (None, 3)                 903
=================================================================
Total params: 656,703
Trainable params: 656,703
```

```
Non-trainable params: 0
_________________________________________________________________
```

In [39]:
```
model.fit(X_train,y_train,validation_data=(X_test,y_test),batch_size=20,epochs=10,verbose=1)
```

```
Epoch 1/10
6/6 [==============================] - 1s 35ms/step - loss: 1.0777 - accuracy: 0.4417 - val_loss: 1.0340 - val_accuracy:
0.7000
Epoch 2/10
6/6 [==============================] - 0s 12ms/step - loss: 0.9955 - accuracy: 0.6167 - val_loss: 0.8957 - val_accuracy:
0.7000
Epoch 3/10
6/6 [==============================] - 0s 11ms/step - loss: 0.8306 - accuracy: 0.7000 - val_loss: 0.6957 - val_accuracy:
0.7000
Epoch 4/10
6/6 [==============================] - 0s 12ms/step - loss: 0.6152 - accuracy: 0.7917 - val_loss: 0.4568 - val_accuracy:
0.8333
Epoch 5/10
6/6 [==============================] - 0s 11ms/step - loss: 0.4344 - accuracy: 0.8583 - val_loss: 0.3384 - val_accuracy:
0.9667
Epoch 6/10
6/6 [==============================] - 0s 11ms/step - loss: 0.3453 - accuracy: 0.9083 - val_loss: 0.2424 - val_accuracy:
0.9667
Epoch 7/10
6/6 [==============================] - 0s 13ms/step - loss: 0.2543 - accuracy: 0.9333 - val_loss: 0.1676 - val_accuracy:
1.0000
Epoch 8/10
6/6 [==============================] - 0s 17ms/step - loss: 0.2213 - accuracy: 0.9333 - val_loss: 0.1479 - val_accuracy:
0.9667
Epoch 9/10
6/6 [==============================] - 0s 12ms/step - loss: 0.1689 - accuracy: 0.9500 - val_loss: 0.2362 - val_accuracy:
0.8667
Epoch 10/10
6/6 [==============================] - 0s 13ms/step - loss: 0.1767 - accuracy: 0.9250 - val_loss: 0.0839 - val_accuracy:
1.0000
```

Out[39]:
```
<keras.callbacks.History at 0x2542ba16b50>
```

In [40]:
```
prediction=model.predict(X_test)
length=len(prediction)
y_label=np.argmax(y_test,axis=1)
predict_label=np.argmax(prediction,axis=1)

accuracy=np.sum(y_label==predict_label)/length * 100
print("Accuracy of the dataset",accuracy )
```

```
Accuracy of the dataset 100.0
```

In [ ]: