

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from tensorflow import keras
from tensorflow.keras import layers
import matplotlib.pyplot as plt
```

```
iris = load_iris()
X = iris.data
y = iris.target
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
model = keras.Sequential([
    layers.Dense(10, activation='relu', input_shape=(4,)),
    layers.Dense(8, activation='relu'),
    layers.Dense(3, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

```
/usr/local/lib/python3.12/dist-packages/keras/src/layers/core/dense.py:93: UserWarning: Do not pass an `input_shape`/`input_dim`
super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
history = model.fit(X_train, y_train, epochs=50, validation_split=0.2, verbose=1)
```

```
Epoch 1/50
3/3 ————— 2s 226ms/step - accuracy: 0.4740 - loss: 1.1598 - val_accuracy: 0.4167 - val_loss: 1.2929
Epoch 2/50
3/3 ————— 0s 66ms/step - accuracy: 0.4557 - loss: 1.2011 - val_accuracy: 0.4167 - val_loss: 1.2671
Epoch 3/50
3/3 ————— 0s 67ms/step - accuracy: 0.5052 - loss: 1.1710 - val_accuracy: 0.4167 - val_loss: 1.2434
Epoch 4/50
3/3 ————— 0s 68ms/step - accuracy: 0.5716 - loss: 1.1581 - val_accuracy: 0.4167 - val_loss: 1.2207
Epoch 5/50
3/3 ————— 0s 89ms/step - accuracy: 0.6224 - loss: 1.1088 - val_accuracy: 0.4167 - val_loss: 1.1997
Epoch 6/50
3/3 ————— 0s 75ms/step - accuracy: 0.6315 - loss: 1.1150 - val_accuracy: 0.4167 - val_loss: 1.1784
Epoch 7/50
3/3 ————— 0s 66ms/step - accuracy: 0.6042 - loss: 1.1092 - val_accuracy: 0.4167 - val_loss: 1.1579
Epoch 8/50
3/3 ————— 0s 81ms/step - accuracy: 0.6302 - loss: 1.0871 - val_accuracy: 0.4167 - val_loss: 1.1379
Epoch 9/50
3/3 ————— 0s 65ms/step - accuracy: 0.6146 - loss: 1.0869 - val_accuracy: 0.4583 - val_loss: 1.1174
Epoch 10/50
3/3 ————— 0s 53ms/step - accuracy: 0.5977 - loss: 1.0653 - val_accuracy: 0.5417 - val_loss: 1.0975
Epoch 11/50
3/3 ————— 0s 51ms/step - accuracy: 0.6016 - loss: 1.0656 - val_accuracy: 0.5833 - val_loss: 1.0782
Epoch 12/50
3/3 ————— 0s 34ms/step - accuracy: 0.6094 - loss: 1.0483 - val_accuracy: 0.5833 - val_loss: 1.0594
Epoch 13/50
3/3 ————— 0s 37ms/step - accuracy: 0.6549 - loss: 1.0312 - val_accuracy: 0.5833 - val_loss: 1.0415
Epoch 14/50
3/3 ————— 0s 36ms/step - accuracy: 0.6862 - loss: 1.0204 - val_accuracy: 0.5833 - val_loss: 1.0239
Epoch 15/50
3/3 ————— 0s 34ms/step - accuracy: 0.6914 - loss: 1.0110 - val_accuracy: 0.5833 - val_loss: 1.0067
Epoch 16/50
3/3 ————— 0s 33ms/step - accuracy: 0.6745 - loss: 1.0099 - val_accuracy: 0.6667 - val_loss: 0.9892
Epoch 17/50
3/3 ————— 0s 35ms/step - accuracy: 0.6888 - loss: 0.9848 - val_accuracy: 0.7917 - val_loss: 0.9727
Epoch 18/50
3/3 ————— 0s 41ms/step - accuracy: 0.6654 - loss: 0.9860 - val_accuracy: 0.8333 - val_loss: 0.9563
```

```

Epoch 19/50
3/3 ————— 0s 33ms/step - accuracy: 0.7201 - loss: 0.9651 - val_accuracy: 0.8333 - val_loss: 0.9402
Epoch 20/50
3/3 ————— 0s 35ms/step - accuracy: 0.7253 - loss: 0.9562 - val_accuracy: 0.8333 - val_loss: 0.9245
Epoch 21/50
3/3 ————— 0s 34ms/step - accuracy: 0.7982 - loss: 0.9364 - val_accuracy: 0.8750 - val_loss: 0.9090
Epoch 22/50
3/3 ————— 0s 35ms/step - accuracy: 0.7591 - loss: 0.9302 - val_accuracy: 0.8750 - val_loss: 0.8942
Epoch 23/50
3/3 ————— 0s 37ms/step - accuracy: 0.7760 - loss: 0.9127 - val_accuracy: 0.8750 - val_loss: 0.8794
Epoch 24/50
3/3 ————— 0s 38ms/step - accuracy: 0.8086 - loss: 0.9040 - val_accuracy: 0.8750 - val_loss: 0.8647
Epoch 25/50
3/3 ————— 0s 35ms/step - accuracy: 0.7318 - loss: 0.9004 - val_accuracy: 0.8750 - val_loss: 0.8505
Epoch 26/50
3/3 ————— 0s 35ms/step - accuracy: 0.7747 - loss: 0.8888 - val_accuracy: 0.8333 - val_loss: 0.8365
Epoch 27/50
3/3 ————— 0s 35ms/step - accuracy: 0.7826 - loss: 0.8749 - val_accuracy: 0.8333 - val_loss: 0.8232
Epoch 28/50
3/3 ————— 0s 57ms/step - accuracy: 0.7865 - loss: 0.8579 - val_accuracy: 0.8333 - val_loss: 0.8098
Epoch 29/50
3/3 ————— 0s 35ms/step - accuracy: 0.7474 - loss: 0.8715 - val_accuracy: 0.8750 - val_loss: 0.7671

```

```

test_loss, test_acc = model.evaluate(X_test, y_test)
print(f"\n Test Accuracy: {test_acc:.2f}")

```

```

1/1 ————— 0s 86ms/step - accuracy: 0.9000 - loss: 0.5875

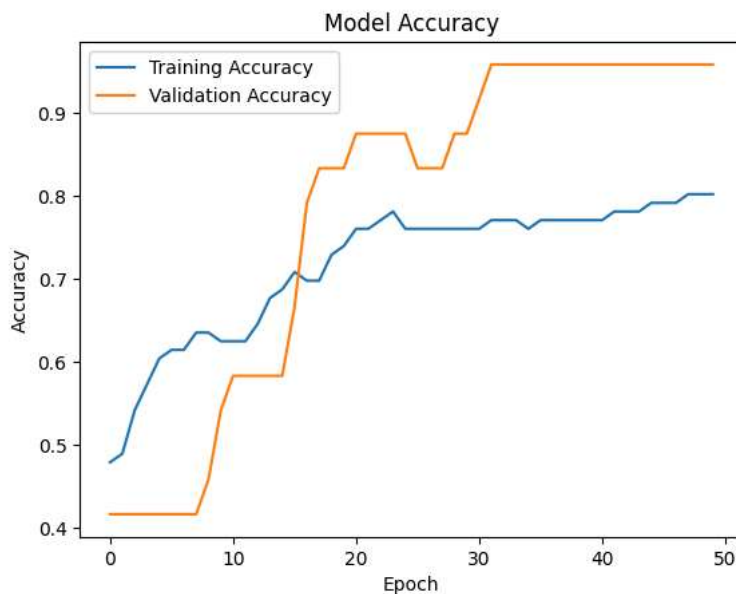
```

Test Accuracy: 0.90

```

plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

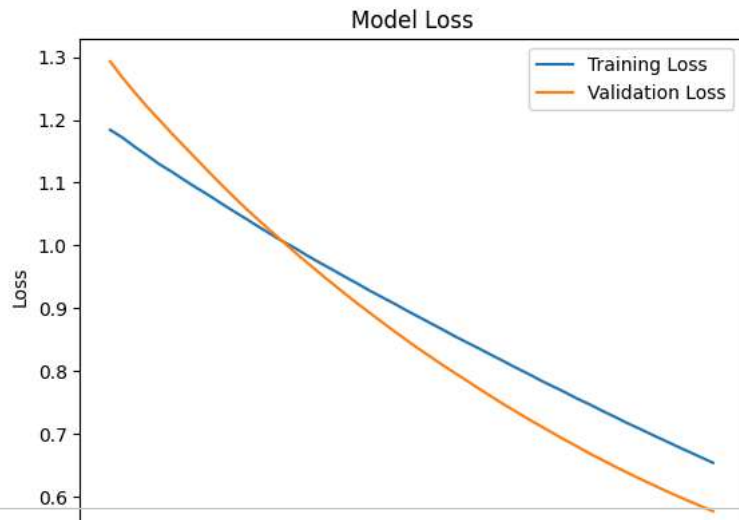
```



```

plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()

```



```
predictions = np.argmax(model.predict(X_test), axis=1)
```

```
print("Predicted:", predictions[:10])  
print("Actual:   ", y_test[:10])
```

1/1 ————— 0s 202ms/step
Predicted: [1 0 2 2 1 0 1 2 1 1]
Actual: [1 0 2 1 1 0 1 2 1 1]