import numpy as np

from keras.datasets import cifar10

from keras.models import Sequential

from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

from keras.utils import to\_categorical

**# Load CIFAR-10 dataset**

(x\_train, y\_train), (x\_test, y\_test) = cifar10.load\_data()

**# Normalize pixel values to be between 0 and 1**

x\_train = x\_train.astype('float32') / 255

x\_test = x\_test.astype('float32') / 255

**# Convert class labels to one-hot encoded vectors**

y\_train = to\_categorical(y\_train, num\_classes=10)

y\_test = to\_categorical(y\_test, num\_classes=10)

**# Create a Sequential model**

model = Sequential()

**# Add a Convolutional layer with 32 filters, kernel size 3x3, and ReLU activation function**

model.add(Conv2D(32, (3, 3), activation='relu', input\_shape=(32, 32, 3)))

**# Add a MaxPooling layer with pool size 2x2**

model.add(MaxPooling2D(pool\_size=(2, 2)))

**# Add another Convolutional layer with 64 filters, kernel size 3x3, and ReLU activation function**

model.add(Conv2D(64, (3, 3), activation='relu'))

**# Add another MaxPooling layer with pool size 2x2**

model.add(MaxPooling2D(pool\_size=(2, 2)))

**# Flatten the output from the convolutional layers**

model.add(Flatten())

**# Add a fully connected layer with 64 neurons and ReLU activation function**

model.add(Dense(64, activation='relu'))

**# Add the output layer with 10 neurons (one for each class) and softmax activation function**

model.add(Dense(10, activation='softmax'))

**# Compile the model with Adam optimizer, categorical crossentropy loss, and accuracy metric**

model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

**# Train the model on the training data**

model.fit(x\_train, y\_train, batch\_size=32, epochs=10, validation\_data=(x\_test, y\_test))

**# Evaluate the model on the test data**

loss, accuracy = model.evaluate(x\_test, y\_test)

print(f'Test Loss: {loss}')

print(f'Test Accuracy: {accuracy}')