```
import os
import zipfile
```

```
import tensorflow as tf
from tensorflow.keras import layers, models
import tensorflow_datasets as tfds
import matplotlib.pyplot as plt
```

```
train_dataset,info = tfds.load('horses_or_humans', with_info = True, split = 'train', as_supervised=True)
val_dataset,val_info = tfds.load("horses_or_humans", with_info=True, split = 'test', as_supervised=True)
```

```
print(len(train_dataset))
print(len(val_dataset))
```

```
    1027
    256
```

```
target_size = (64, 64)  # Specify the desired height and width

# Function to resize images
def resize_images(features, label):
    resized_image = tf.image.resize(features, target_size)
    return resized_image, label

# Apply the resize function to the dataset
train_dataset = train_dataset.map(resize_images)
```

```
train_dataset = train_dataset.shuffle(100).batch(10)
val_dataset = val_dataset.batch(10)
```

```
from sklearn.model_selection import train_test_split
```

```
import numpy as np
```

```
features = []
labels = []

for batch in train_dataset:
    image, label = batch
    features.append(image.numpy())
    labels.append(label.numpy())

# Concatenate the lists or arrays
features = np.concatenate(features, axis=0)
labels = np.concatenate(labels, axis=0)
```

```
features.shape
```

```
    (1027, 64, 64, 3)
```

```
xtrain,xtest,ytrain,ytest = train_test_split(features, labels, test_size=0.2)
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D , Dense , Flatten , MaxPooling2D
model = Sequential()
```

```
#CNN Input Layer
model.add(Conv2D(32 , (3,3) , activation="relu" , input_shape = (64,64,3)))


model.add(MaxPooling2D(2,2))


#1 CNN block
model.add(Conv2D(64 , (3,3) , activation = "relu"))
model.add(MaxPooling2D(2,2))


#2 CNN block
model.add(Conv2D(128 , (3,3) , activation = "relu"))
model.add(MaxPooling2D(2,2))


#3 CNN block
#model.add(Conv2D(64 , (3,3) , activation = "relu"))
#model.add(MaxPooling2D(2,2))


#Flatten Layer
model.add(Flatten())


# ANN Layer
model.add(Dense(64 , activation = "relu"))


# ANN output layer
model.add(Dense(1 , activation = "sigmoid"))


# compile
model.compile(optimizer = "adam" , loss = "binary_crossentropy" , metrics = ["accuracy"])


#run
model.fit(xtrain, ytrain, epochs = 2 , batch_size= 32 , validation_data=(xtest,ytest))
```

```
Epoch 1/2
26/26 [==============================] - 7s 218ms/step - loss: 9.7933 - accuracy: 0.6322 - val_loss: 0.3156
Epoch 2/2
26/26 [==============================] - 7s 273ms/step - loss: 0.2060 - accuracy: 0.9294 - val_loss: 0.1688
<keras.src.callbacks.History at 0x7f92ba64bd00>
```

```
loss , accuracy = model.evaluate(xtest,ytest)
print(loss , accuracy)
```

```
7/7 [==============================] - 1s 71ms/step - loss: 0.1688 - accuracy: 0.9223
0.1688072234392166 0.9223300814628601
```

```
import cv2
import numpy as np
image_path = "/content/joseph-gonzalez-iFgRcqHznqg-unsplash-scaled-1.jpg"
image = cv2.imread(image_path)
image = cv2.resize(image, (64,64))
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
```

```
image.shape
```

```
(64, 64, 3)
```

```
image = np.expand_dims(image, axis = 0)
image.shape
```

```
    (1, 1, 64, 64, 3)
```

```
image = image / 255.0
```

```
model.predict(image)
```

```
    1/1 [==============================] - 0s 122ms/step
    array([[0.5012791]], dtype=float32)
```

```
output = model.predict(image)
print("Human") if output[0][0] > 0.5 else print("Horse")
```

```
    1/1 [==============================] - 0s 39ms/step
    Human
```

```
import cv2
import numpy as np
image_path = "/content/caballos-con-caras-graciosas.jpg"
image = cv2.imread(image_path)
image = cv2.resize(image, (64,64))
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

image = np.expand_dims(image, axis = 0)
image = image / 255.0
```

```
model.predict(image)
```

```
    1/1 [==============================] - 0s 25ms/step
    array([[0.49800333]], dtype=float32)
```

```
output = model.predict(image)
print("Human") if output[0][0] > 0.5 else print("Horse")
```

```
    1/1 [==============================] - 0s 23ms/step
    Horse
```

Start coding or generate with AI.