```
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials

auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)


downloaded = drive.CreateFile({'id':"1NwrithRqi1lcpAPadM5Rz0AQrqmr-2DD"})
downloaded.GetContentFile('CatvsDogs.rar')

!unrar x "/content/CatvsDogs.rar" "/content/"
```

```
Streaming output truncated to the last 5000 lines.
Extracting  /content/train/dog.55.jpg                                 OK
Extracting  /content/train/dog.550.jpg                                OK
Extracting  /content/train/dog.5500.jpg                               OK
Extracting  /content/train/dog.5501.jpg                               OK
Extracting  /content/train/dog.5502.jpg                               OK
Extracting  /content/train/dog.5503.jpg                               OK
Extracting  /content/train/dog.5504.jpg                               OK
Extracting  /content/train/dog.5505.jpg                               OK
Extracting  /content/train/dog.5506.jpg                               OK
Extracting  /content/train/dog.5507.jpg                               OK
Extracting  /content/train/dog.5508.jpg                               OK
Extracting  /content/train/dog.5509.jpg                               OK
Extracting  /content/train/dog.551.jpg                                OK
Extracting  /content/train/dog.5510.jpg                               OK
Extracting  /content/train/dog.5511.jpg                               OK
Extracting  /content/train/dog.5512.jpg                               OK
Extracting  /content/train/dog.5513.jpg                               OK
Extracting  /content/train/dog.5514.jpg                               OK
Extracting  /content/train/dog.5515.jpg                               OK
Extracting  /content/train/dog.5516.jpg                               OK
Extracting  /content/train/dog.5517.jpg                               OK
Extracting  /content/train/dog.5518.jpg                               OK
Extracting  /content/train/dog.5519.jpg                               OK
Extracting  /content/train/dog.552.jpg                                OK
Extracting  /content/train/dog.5520.jpg                               OK
Extracting  /content/train/dog.5521.jpg                               OK
Extracting  /content/train/dog.5522.jpg                               OK
Extracting  /content/train/dog.5523.jpg                               OK
Extracting  /content/train/dog.5524.jpg                               OK
Extracting  /content/train/dog.5525.jpg                               OK
Extracting  /content/train/dog.5526.jpg                               OK
Extracting  /content/train/dog.5527.jpg                               OK
Extracting  /content/train/dog.5528.jpg                               OK
Extracting  /content/train/dog.5529.jpg                               OK
Extracting  /content/train/dog.553.jpg                                OK
Extracting  /content/train/dog.5530.jpg                               OK
Extracting  /content/train/dog.5531.jpg                               OK
Extracting  /content/train/dog.5532.jpg                               OK
Extracting  /content/train/dog.5533.jpg                               OK
Extracting  /content/train/dog.5534.jpg                               OK
Extracting  /content/train/dog.5535.jpg                               OK
Extracting  /content/train/dog.5536.jpg                               OK
Extracting  /content/train/dog.5537.jpg                               OK
Extracting  /content/train/dog.5538.jpg                               OK
Extracting  /content/train/dog.5539.jpg                               OK
Extracting  /content/train/dog.554.jpg                                OK
Extracting  /content/train/dog.5540.jpg                               OK
Extracting  /content/train/dog.5541.jpg                               OK
Extracting  /content/train/dog.5542.jpg                               OK
Extracting  /content/train/dog.5543.jpg                               OK
Extracting  /content/train/dog.5544.jpg                               OK
Extracting  /content/train/dog.5545.jpg                               OK
Extracting  /content/train/dog.5546.jpg                               OK
Extracting  /content/train/dog.5547.jpg                               OK
Extracting  /content/train/dog.5548.jpg                               OK
Extracting  /content/train/dog.5549.jpg                               OK
Extracting  /content/train/dog.555.jpg                                OK
```

```
import pandas as pd
import tensorflow as tf
from tensorflow.keras import models, Sequential,layers,preprocessing
import os
file_names=os.listdir("/content/train")


dogorcat=[]
for name in file_names:
  category=name.split('.')[0]
  if category=='dog':
```

```
    dogorcat.append("DOG")
  else:
    dogorcat.append("CAT")
train=pd.DataFrame({ 'filename':file_names, 'category':dogorcat})
train
```

	filename	category
0	cat.6534.jpg	CAT
1	cat.1431.jpg	CAT
2	cat.2233.jpg	CAT
3	dog.5920.jpg	DOG
4	dog.6674.jpg	DOG
...	...	...
24995	dog.7299.jpg	DOG
24996	cat.1508.jpg	CAT
24997	cat.2850.jpg	CAT
24998	dog.5058.jpg	DOG
24999	cat.6405.jpg	CAT

25000 rows × 2 columns

```
model=models.Sequential()
model.add(layers.BatchNormalization())
model.add(layers.Conv2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
#model.add(layers.Conv2D(32,(3,3),activation='relu'))
model.add(layers.BatchNormalization())
model.add(layers.MaxPooling2D(2,2))
model.add(layers.Conv2D(64,(3,3),activation='relu'))
model.add(layers.BatchNormalization())
#model.add(layers.Conv2D(64,(3,3),activation='relu'))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.Conv2D(128,(3,3),activation='relu'))
model.add(layers.BatchNormalization())
#model.add(layers.Conv2D(128,(3,3),activation='relu'))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.Flatten())
model.add(layers.Dense(512,activation='relu'))
model.add(layers.Dense(256,activation="relu"))
model.add(layers.Dense(128,activation="relu"))
model.add(layers.Dense(2,activation="softmax"))
```

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

```
training = preprocessing.image.ImageDataGenerator(rotation_range=15, rescale=1./255, shear_range=0.1, zoom_range=0.2, horizontal_flip=Tru
validation=preprocessing.image.ImageDataGenerator(rotation_range=15, rescale=1./255, shear_range=0.1, zoom_range=0.2, horizontal_flip=Tru
```

```
trainingdata = training.flow_from_dataframe(train,"/content/train",x_col='filename',y_col='category',target_size=(64,64),class_mode='cate
```

```
Found 25000 validated image filenames belonging to 2 classes.
```

```
model.fit(trainingdata,epochs=10)
```

```
Epoch 1/10
782/782 [==============================] - 75s 87ms/step - loss: 0.5879 - accuracy: 0.6924
Epoch 2/10
782/782 [==============================] - 68s 87ms/step - loss: 0.4724 - accuracy: 0.7738
Epoch 3/10
782/782 [==============================] - 69s 89ms/step - loss: 0.4074 - accuracy: 0.8114
Epoch 4/10
782/782 [==============================] - 67s 85ms/step - loss: 0.3645 - accuracy: 0.8356
Epoch 5/10
782/782 [==============================] - 67s 86ms/step - loss: 0.3382 - accuracy: 0.8506
Epoch 6/10
782/782 [==============================] - 67s 86ms/step - loss: 0.3133 - accuracy: 0.8630
Epoch 7/10
782/782 [==============================] - 67s 85ms/step - loss: 0.2998 - accuracy: 0.8684
Epoch 8/10
782/782 [==============================] - 68s 87ms/step - loss: 0.2834 - accuracy: 0.8766
Epoch 9/10
782/782 [==============================] - 67s 86ms/step - loss: 0.2706 - accuracy: 0.8837
Epoch 10/10
782/782 [==============================] - 63s 81ms/step - loss: 0.2648 - accuracy: 0.8835
<keras.src.callbacks.History at 0x79f5d0621b70>
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 batch_normalization (Batch  (None, None, None, 3)     12
 Normalization)

 conv2d (Conv2D)             (None, None, None, 32)    896

 batch_normalization_1 (Bat  (None, None, None, 32)    128
 chNormalization)

 max_pooling2d (MaxPooling2  (None, None, None, 32)    0
 D)

 conv2d_1 (Conv2D)           (None, None, None, 64)    18496

 batch_normalization_2 (Bat  (None, None, None, 64)    256
 chNormalization)

 max_pooling2d_1 (MaxPoolin  (None, None, None, 64)    0
 g2D)

 conv2d_2 (Conv2D)           (None, None, None, 128)   73856

 batch_normalization_3 (Bat  (None, None, None, 128)   512
 chNormalization)

 max_pooling2d_2 (MaxPoolin  (None, None, None, 128)   0
 g2D)

 flatten (Flatten)           (None, None)              0

 dense (Dense)               (None, 512)               2359808

 dense_1 (Dense)             (None, 256)               131328

 dense_2 (Dense)             (None, 128)               32896

 dense_3 (Dense)             (None, 2)                 258

=================================================================
Total params: 2618446 (9.99 MB)
Trainable params: 2617992 (9.99 MB)
Non-trainable params: 454 (1.77 KB)
_________________________________________________________________
```