Taller 2 Pablo Micolta Lopez

```
[17] from google.colab import drive
     drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[18] !pwd
```

```
/content
```

```
[19] cd '/content/drive/MyDrive/Proyecto_I-MachineLearning'
```

```
/content/drive/MyDrive/Proyecto_I-MachineLearning
```

```
[20] dataPath
     '/content/drive/MyDrive/Proyecto_I-MachineLearning/Data/cats_vs_dogs_small'
```

```
'/content/drive/MyDrive/Proyecto_I-MachineLearning/Data/cats_vs_dogs_small'
```

```
[21] import tensorflow as tf
     from tensorflow import  keras
     from tensorflow.keras.preprocessing import  image_dataset_from_directory
```

```
[22] from PIL import Image
     from IPython.display import display
     import matplotlib.pyplot as plt
     import numpy as np
     import os
     #importar libs de pros imagenes
```

```
[23] setName='train'
     className="dog"
     fileName='dog.1.jpg'
     filePath= os.path.join(dataPath,setName,className,fileName)
```

```
trainPath= os.path.join(dataPath,'train')
trainSet= image_dataset_from_directory(trainPath, shuffle=True, batch_size=32, image_size=(150,150), validation_split=0.2, subset='training', seed=1234)
validationSet= image_dataset_from_directory(trainPath, shuffle=True, batch_size=32, image_size=(150,150), validation_split=0.2, subset='validation', seed=1
#limito y defino los sets de entrenar y validar
```

```
[ ] baseModel = keras.applications.InceptionV3(weights='imagenet', input_shape=(150,150,3),include_top=False)
    #baseModel = keras.applications.Xception(weights='imagenet', input_shape=(150,150,3),include_top=False)
    baseModel.trainable=False
    #definir el modelo a utilizar (en este caso la Xception) con pre entrada de imagenet para extraccion de caracteristicas
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_v3/inception_v3_weights_tf_dim_ordering_
87916544/87910968 [==============================] - 3s 0us/step
87924736/87910968 [==============================] - 3s 0us/step
```

```
inputs= keras.Input(shape=(150,150,3))
x=tf.keras.applications.inception_v3.preprocess_input(inputs) #para cambiar al modelo de Xception simplemente cambiar por xception
x=baseModel(x, training=False)
x=keras.layers.GlobalAveragePooling2D()(x)
x=keras.layers.Dropout(0.2)(x)
outputs=keras.layers.Dense(1)(x)
model=keras.Model(inputs, outputs)
# definir entradas de tamaño del input shape cuyos valores de pixeles etan entre 0 y 255
#se confirma que la primera parte del modelo sera el base y que no se modifican sus paramentros (como seguro)
# el output se promedia conservando el tamaño del batch se agrega capa dopout para regularizar
```

```
model.compile(optimizer='adam', loss= tf.keras.losses.BinaryCrossentropy(from_logits=True),metrics= keras.metrics.BinaryAccuracy())
model.fit(trainSet, epochs=20, validation_data= validationSet)
#entrena el modelo
```

```
jsonConfig=model.to_json()
with open('model_config2.json','w') as json_file:
  json_file.write(jsonConfig)

model.save_weights('pets_InceptionV3_transferlearning.h5')
#desplegar modelo y almacenar parametros en un json y sus pesos en el archivo del final
```

Welcome to PetLlasifier App

Seleccionar archivo | Sin archivos seleccionados

upload

dog prob 1.4366075902216835e-06, cat prob0.9999985694885254