Download the Dataset

```
!wget --no-check-certificate https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip -O /tmp/cats_and_dogs_filtered.zip
```

```
--2019-12-11 20:36:44--  https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip
Resolving storage.googleapis.com (storage.googleapis.com)... 74.125.204.128, 2404:6800:4008:c06::80
Connecting to storage.googleapis.com (storage.googleapis.com)|74.125.204.128|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 68606236 (65M) [application/zip]
Saving to: ‘/tmp/cats_and_dogs_filtered.zip’

/tmp/cats_and_dogs_ 100%[===================>]  65.43M  27.7MB/s    in 2.4s

2019-12-11 20:36:47 (27.7 MB/s) - ‘/tmp/cats_and_dogs_filtered.zip’ saved [68606236/68606236]
```

Import the Packages

```
import os
import zipfile
import shutil
import numpy as np
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import Model
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.inception_v3 import InceptionV3
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from google.colab import files
import matplotlib.pyplot as plt
%matplotlib inline
```

Extract the Images Dataset

```
local_zip = '/tmp/cats_and_dogs_filtered.zip'
zip_ref = zipfile.ZipFile(local_zip, 'r')
zip_ref.extractall('/tmp')
zip_ref.close()
```

Setup the Images Directories

```
base_dir = '/tmp/cats_and_dogs_filtered'
train_dir = os.path.join(base_dir, 'train')
validation_dir = os.path.join(base_dir, 'validation')
```

Configure the Inception V3 Model

Download the weights of the Model

```
!wget --no-check-certificate https://storage.googleapis.com/mledu-datasets/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5 -O ,
```

```
--2019-12-11 20:36:53--  https://storage.googleapis.com/mledu-datasets/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5
Resolving storage.googleapis.com (storage.googleapis.com)... 64.233.187.128, 2404:6800:4008:c00::80
Connecting to storage.googleapis.com (storage.googleapis.com)|64.233.187.128|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 87910968 (84M) [application/x-hdf]
Saving to: ‘/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5’

/tmp/inception_v3_w 100%[===================>]  83.84M  69.2MB/s    in 1.2s

2019-12-11 20:36:54 (69.2 MB/s) - ‘/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5’ saved [87910968/87910968]
```

Configure the Inception V3 Model

```
local_weights_file = '/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5'
```

```
pre_trained_model = InceptionV3(input_shape=(150, 150, 3), include_top=False, weights=None)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/resource_variable_ops.py:1630: calling Bas
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
```

```
pre_trained_model.load_weights(local_weights_file)
```

```
pre_trained_model.summary()
```

```
Model: "inception_v3"
_________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
=================================================================================================
input_1 (InputLayer)            [(None, 150, 150, 3) 0
_________________________________________________________________________________________________
conv2d (Conv2D)                 (None, 74, 74, 32)   864         input_1[0][0]
_________________________________________________________________________________________________
batch_normalization (BatchNorma (None, 74, 74, 32)   96          conv2d[0][0]
_________________________________________________________________________________________________
activation (Activation)         (None, 74, 74, 32)   0           batch_normalization[0][0]
_________________________________________________________________________________________________
conv2d_1 (Conv2D)               (None, 72, 72, 32)   9216        activation[0][0]
_________________________________________________________________________________________________
batch_normalization_1 (BatchNor (None, 72, 72, 32)   96          conv2d_1[0][0]
_________________________________________________________________________________________________
activation_1 (Activation)       (None, 72, 72, 32)   0           batch_normalization_1[0][0]
_________________________________________________________________________________________________
conv2d_2 (Conv2D)               (None, 72, 72, 64)   18432       activation_1[0][0]
_________________________________________________________________________________________________
batch_normalization_2 (BatchNor (None, 72, 72, 64)   192         conv2d_2[0][0]
_________________________________________________________________________________________________
activation_2 (Activation)       (None, 72, 72, 64)   0           batch_normalization_2[0][0]
_________________________________________________________________________________________________
max_pooling2d (MaxPooling2D)    (None, 35, 35, 64)   0           activation_2[0][0]
_________________________________________________________________________________________________
conv2d_3 (Conv2D)               (None, 35, 35, 80)   5120        max_pooling2d[0][0]
_________________________________________________________________________________________________
batch_normalization_3 (BatchNor (None, 35, 35, 80)   240         conv2d_3[0][0]
_________________________________________________________________________________________________
activation_3 (Activation)       (None, 35, 35, 80)   0           batch_normalization_3[0][0]
_________________________________________________________________________________________________
conv2d_4 (Conv2D)               (None, 33, 33, 192)  138240      activation_3[0][0]
_________________________________________________________________________________________________
batch_normalization_4 (BatchNor (None, 33, 33, 192)  576         conv2d_4[0][0]
_________________________________________________________________________________________________
activation_4 (Activation)       (None, 33, 33, 192)  0           batch_normalization_4[0][0]
_________________________________________________________________________________________________
max_pooling2d_1 (MaxPooling2D)  (None, 16, 16, 192)  0           activation_4[0][0]
_________________________________________________________________________________________________
conv2d_8 (Conv2D)               (None, 16, 16, 64)   12288       max_pooling2d_1[0][0]
_________________________________________________________________________________________________
batch_normalization_8 (BatchNor (None, 16, 16, 64)   192         conv2d_8[0][0]
_________________________________________________________________________________________________
activation_8 (Activation)       (None, 16, 16, 64)   0           batch_normalization_8[0][0]
_________________________________________________________________________________________________
conv2d_6 (Conv2D)               (None, 16, 16, 48)   9216        max_pooling2d_1[0][0]
_________________________________________________________________________________________________
conv2d_9 (Conv2D)               (None, 16, 16, 96)   55296       activation_8[0][0]
_________________________________________________________________________________________________
batch_normalization_6 (BatchNor (None, 16, 16, 48)   144         conv2d_6[0][0]
_________________________________________________________________________________________________
batch_normalization_9 (BatchNor (None, 16, 16, 96)   288         conv2d_9[0][0]
_________________________________________________________________________________________________
activation_6 (Activation)       (None, 16, 16, 48)   0           batch_normalization_6[0][0]
_________________________________________________________________________________________________
activation_9 (Activation)       (None, 16, 16, 96)   0           batch_normalization_9[0][0]
_________________________________________________________________________________________________
```

```
# Freeze the Layers
for layer in pre_trained_model.layers:
  layer.trainable = False
```

```
# Retrieve the last layer
last_layer = pre_trained_model.get_layer(name='mixed7')
print('Last Layer Output Shape: ', last_layer.output_shape)
```

```
Last Layer Output Shape:  (None, 7, 7, 768)
```

```
# Output of the last layer
last_output = last_layer.output
```

Setup the Model

```
x = layers.Flatten()(last_output)

x = layers.Dense(units=1024, activation='relu')(x)

x = layers.Dropout(rate=0.2)(x)

x = layers.Dense(units=1, activation='sigmoid')(x)
```

```
model = Model(pre_trained_model.input, x)
```

```
model.compile(optimizer=RMSprop(learning_rate=0.0001), loss='binary_crossentropy', metrics=['acc'])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/nn_impl.py:183: where (from tensorflow.py
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
```

Configure the ImageDataGenerator

```
train_datagen = ImageDataGenerator(rescale=1/255,
                                   rotation_range=40,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1/255)
```

```
train_generator = train_datagen.flow_from_directory(directory=train_dir, target_size=(150, 150), batch_size=20, class_mode='binary')

test_generator = test_datagen.flow_from_directory(directory=validation_dir, target_size=(150, 150), batch_size=20, class_mode='binary')
```

```
Found 2000 images belonging to 2 classes.
Found 1000 images belonging to 2 classes.
```

Train the model

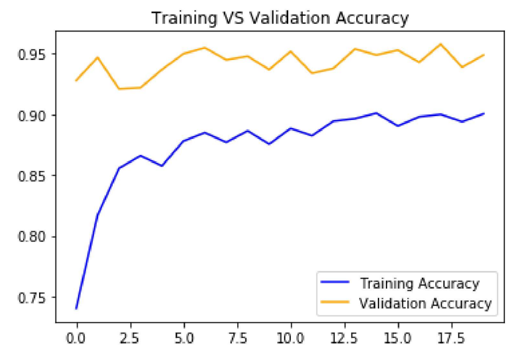
```
history = model.fit_generator(generator=train_generator, steps_per_epoch=100, epochs=20, validation_data=test_generator, validation_step
```

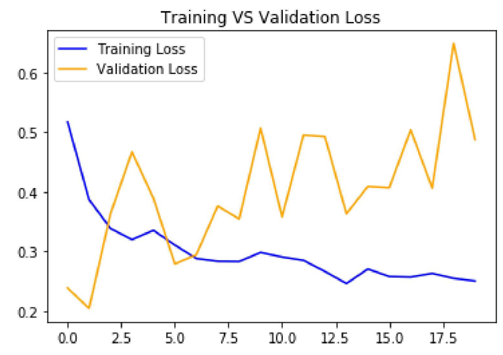
```
Epoch 1/20
 99/100 [============================>.] - ETA: 0s - loss: 0.5159 - acc: 0.7404Epoch 1/20
100/100 [==============================] - 28s 285ms/step - loss: 0.5167 - acc: 0.7400 - val_loss: 0.2385 - val_acc: 0.9280
Epoch 2/20
 99/100 [============================>.] - ETA: 0s - loss: 0.3876 - acc: 0.8162Epoch 1/20
100/100 [==============================] - 23s 227ms/step - loss: 0.3869 - acc: 0.8170 - val_loss: 0.2047 - val_acc: 0.9470
Epoch 3/20
 99/100 [============================>.] - ETA: 0s - loss: 0.3386 - acc: 0.8556Epoch 1/20
100/100 [==============================] - 23s 229ms/step - loss: 0.3384 - acc: 0.8555 - val_loss: 0.3632 - val_acc: 0.9210
Epoch 4/20
 99/100 [============================>.] - ETA: 0s - loss: 0.3186 - acc: 0.8662Epoch 1/20
100/100 [==============================] - 23s 228ms/step - loss: 0.3199 - acc: 0.8660 - val_loss: 0.4668 - val_acc: 0.9220
Epoch 5/20
 99/100 [============================>.] - ETA: 0s - loss: 0.3361 - acc: 0.8571Epoch 1/20
100/100 [==============================] - 23s 227ms/step - loss: 0.3354 - acc: 0.8575 - val_loss: 0.3892 - val_acc: 0.9370
Epoch 6/20
 99/100 [============================>.] - ETA: 0s - loss: 0.3089 - acc: 0.8783Epoch 1/20
100/100 [==============================] - 22s 224ms/step - loss: 0.3107 - acc: 0.8780 - val_loss: 0.2792 - val_acc: 0.9500
Epoch 7/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2898 - acc: 0.8848Epoch 1/20
100/100 [==============================] - 23s 229ms/step - loss: 0.2882 - acc: 0.8850 - val_loss: 0.2945 - val_acc: 0.9550
Epoch 8/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2846 - acc: 0.8763Epoch 1/20
100/100 [==============================] - 22s 223ms/step - loss: 0.2835 - acc: 0.8770 - val_loss: 0.3761 - val_acc: 0.9450
Epoch 9/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2842 - acc: 0.8864Epoch 1/20
```

```
100/100 [==============================] - 22s 223ms/step - loss: 0.2832 - acc: 0.8865 - val_loss: 0.3544 - val_acc: 0.9480
Epoch 10/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2998 - acc: 0.8753Epoch 1/20
100/100 [==============================] - 22s 224ms/step - loss: 0.2984 - acc: 0.8755 - val_loss: 0.5067 - val_acc: 0.9370
Epoch 11/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2929 - acc: 0.8874Epoch 1/20
100/100 [==============================] - 23s 226ms/step - loss: 0.2907 - acc: 0.8885 - val_loss: 0.3579 - val_acc: 0.9520
Epoch 12/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2871 - acc: 0.8813Epoch 1/20
100/100 [==============================] - 23s 228ms/step - loss: 0.2851 - acc: 0.8825 - val_loss: 0.4949 - val_acc: 0.9340
Epoch 13/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2649 - acc: 0.8949Epoch 1/20
100/100 [==============================] - 23s 227ms/step - loss: 0.2665 - acc: 0.8945 - val_loss: 0.4923 - val_acc: 0.9380
Epoch 14/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2456 - acc: 0.8970Epoch 1/20
100/100 [==============================] - 23s 226ms/step - loss: 0.2462 - acc: 0.8965 - val_loss: 0.3631 - val_acc: 0.9540
Epoch 15/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2700 - acc: 0.9010Epoch 1/20
100/100 [==============================] - 23s 226ms/step - loss: 0.2706 - acc: 0.9010 - val_loss: 0.4090 - val_acc: 0.9490
Epoch 16/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2587 - acc: 0.8904Epoch 1/20
100/100 [==============================] - 23s 226ms/step - loss: 0.2580 - acc: 0.8905 - val_loss: 0.4068 - val_acc: 0.9530
Epoch 17/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2576 - acc: 0.8975Epoch 1/20
100/100 [==============================] - 23s 226ms/step - loss: 0.2571 - acc: 0.8980 - val_loss: 0.5038 - val_acc: 0.9430
Epoch 18/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2643 - acc: 0.9000Epoch 1/20
100/100 [==============================] - 23s 226ms/step - loss: 0.2632 - acc: 0.9000 - val_loss: 0.4060 - val_acc: 0.9580
Epoch 19/20
 99/100 [============================>.] - ETA: 0s - loss: 0.2557 - acc: 0.8939Epoch 1/20
100/100 [==============================] - 23s 227ms/step - loss: 0.2550 - acc: 0.8940 - val_loss: 0.6481 - val_acc: 0.9390
Epoch 20/20
```

Evaluating Accuracy and Loss

```
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(len(acc))

plt.plot(epochs, acc, color='blue', label='Training Accuracy')
plt.plot(epochs, val_acc, color='orange', label='Validation Accuracy')
plt.title('Training VS Validation Accuracy')
plt.legend()
plt.show()

plt.plot(epochs, loss, color='blue', label='Training Loss')
plt.plot(epochs, val_loss, color='orange', label='Validation Loss')
plt.title('Training VS Validation Loss')
plt.legend()
plt.show()
```





Test the model on new images [Use google colab]

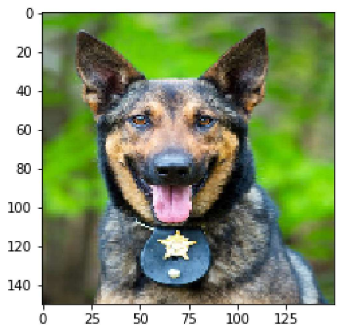
```
class_names = ['Cat', 'Dog']
```

Note : *You might have to run the below cell twice.*

```
uploaded = files.upload()

for file_name in uploaded.keys():

  path = file_name

  img = image.load_img(path, target_size=(150, 150))

  x = image.img_to_array(img)

  x = np.expand_dims(x, axis=0)

  x /= 255.

  images = np.vstack([x])

  classes = model.predict(images, batch_size=10)

  plt.imshow(img)
  plt.grid(False)
  plt.show()

  if classes[0][0] > 0.6:
    print('Prediction: {}'.format(class_names[1]))
  else:
    print('Prediction: {}'.format(class_names[0]))
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving dog_5.jpeg to dog_5 (1).jpeg
```



```
Prediction: Dog
```

Save the model

```
export_dir = "./saved_model_v1"
tf.saved_model.save(model, export_dir)
```

Create a zip file to download the model locally

```
shutil.make_archive(base_name="saved_model_v1",format="zip",root_dir="saved_model_v1")
```

```
'/content/saved_model_v1.zip'
```

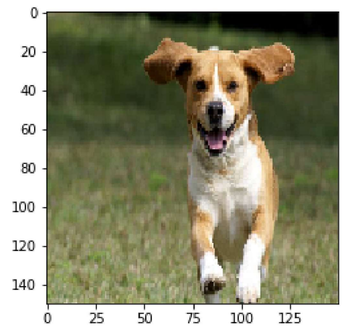
Load the saved model

```
keras_model = tf.keras.models.load_model(export_dir)
```

```
uploaded = files.upload()

for file_name in uploaded.keys():

  path = file_name

  img = image.load_img(path, target_size=(150, 150))

  x = image.img_to_array(img)

  x = np.expand_dims(x, axis=0)

  x /= 255.

  images = np.vstack([x])

  classes = keras_model.predict(images, batch_size=10)

  plt.imshow(img)
  plt.grid(False)
  plt.show()

  if classes[0][0] > 0.6:
    print('Prediction: {}'.format(class_names[1]))
  else:
    print('Prediction: {}'.format(class_names[0]))
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving dog_4.jpeg to dog_4 (4).jpeg
```



```
Prediction: Dog
```

Start coding or generate with AI.