Cats vs Dogs : Image Classification using EfficienNet

Data Description

The training archive contains 25,000 images of dogs and cats.

Install

```
!pip install efficientnet
```

```
Collecting efficientnet
  Downloading efficientnet-1.1.1-py3-none-any.whl.metadata (6.4 kB)
Collecting keras-applications<=1.0.8,>=1.0.7 (from efficientnet)
  Downloading Keras_Applications-1.0.8-py3-none-any.whl.metadata (1.7 kB)
Requirement already satisfied: scikit-image in /usr/local/lib/python3.10/dist-packages (from efficientnet) (0.24.0)
Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.10/dist-packages (from keras-applications<=1.0.8,>=1.0.7->effi
Requirement already satisfied: h5py in /usr/local/lib/python3.10/dist-packages (from keras-applications<=1.0.8,>=1.0.7->efficientnet
Requirement already satisfied: scipy>=1.9 in /usr/local/lib/python3.10/dist-packages (from scikit-image->efficientnet) (1.13.1)
Requirement already satisfied: networkx>=2.8 in /usr/local/lib/python3.10/dist-packages (from scikit-image->efficientnet) (3.4.2)
Requirement already satisfied: pillow>=9.1 in /usr/local/lib/python3.10/dist-packages (from scikit-image->efficientnet) (11.0.0)
Requirement already satisfied: imageio>=2.33 in /usr/local/lib/python3.10/dist-packages (from scikit-image->efficientnet) (2.36.1)
Requirement already satisfied: tifffile>=2022.8.12 in /usr/local/lib/python3.10/dist-packages (from scikit-image->efficientnet) (202
Requirement already satisfied: packaging>=21 in /usr/local/lib/python3.10/dist-packages (from scikit-image->efficientnet) (24.2)
Requirement already satisfied: lazy-loader>=0.4 in /usr/local/lib/python3.10/dist-packages (from scikit-image->efficientnet) (0.4)
Downloading efficientnet-1.1.1-py3-none-any.whl (18 kB)
Downloading Keras_Applications-1.0.8-py3-none-any.whl (50 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.7/50.7 kB 1.7 MB/s eta 0:00:00
Installing collected packages: keras-applications, efficientnet
Successfully installed efficientnet-1.1.1 keras-applications-1.0.8
```

Loading Libraries

> TensorFlow

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

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

```
# Basic
import os
from os import makedirs
from os import listdir
from shutil import copyfile
from random import seed
from random import random
import numpy as np
import pandas as pd
from PIL import Image, ImageFile

# visuals
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.image import imread
from PIL import Image

# Scikit-learn
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix,ConfusionMatrixDisplay

# Tensorflow
!pip install tensorflow==2.14.0
import tensorflow as tf
from tensorflow.keras import callbacks
from keras.callbacks import Callback
from keras import layers
from keras import models
from tensorflow.keras import optimizers
from keras.optimizers import Adam
from efficientnet.keras import center_crop_and_resize, preprocess_input
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.15,>=2.14->tensorfl
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.15,>=2.14->tens
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.15,>=2.14->tensorfl
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->ten
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tens
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib<1.1
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->ten
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.1
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboa
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from werkzeug>=1.0.1->tensorboard<2.
Requirement already satisfied: pyasn1<0.7.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->googl
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-
Downloading tensorflow-2.14.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (489.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 489.8/489.8 MB 3.0 MB/s eta 0:00:00
Downloading ml_dtypes-0.2.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.0/1.0 MB 15.7 MB/s eta 0:00:00
Downloading keras-2.14.0-py3-none-any.whl (1.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.7/1.7 MB 21.3 MB/s eta 0:00:00
```

```
tensorstore 0.1.69 requires ml_dtypes>=0.3.1, but you have ml_dtypes 0.2.0 which is incompatible.
tf-keras 2.17.0 requires tensorflow<2.18,>=2.17, but you have tensorflow 2.14.0 which is incompatible.
Successfully installed google-auth-oauthlib-1.0.0 keras-2.14.0 ml-dtypes-0.2.0 tensorboard-2.14.1 tensorflow-2.14.0 tensorflow-es
```

```
import tensorflow as tf
print(tf.__version__)
```

```
2.14.0
```

```
dataset_url = 'https://download.microsoft.com/download/3/E/1/3E1C3F21-ECDB-4869-8368-6DEBA77B919F/kagglecatsanddogs_5340.zip'
```

Data Extraction

```
data_dir = '/content/sample_data'
%cd '/content/sample_data'
!pwd
```

```
/content/sample_data
/content/sample_data
```

```
path_to_downloaded_file = tf.keras.utils.get_file(
    origin=dataset_url,
    extract=True,
)
```

```
Downloading data from https://download.microsoft.com/download/3/E/1/3E1C3F21-ECDB-4869-8368-6DEBA77B919F/kagglecatsanddogs_5340.zip
824887076/824887076 [==============================] - 7s 0us/step
```

```
path_to_downloaded_file
```

```
'/root/.keras/datasets/kagglecatsanddogs_5340.zip'
```

```
import zipfile

with zipfile.ZipFile(path_to_downloaded_file, 'r') as zipp:
    zipp.extractall(data_dir)
```

```
import glob

Cat_count = len(list(glob.glob(f'{data_dir}/PetImages/Cat/*.jpg')))
print('Cat: ', Cat_count)

Dog_count = len(list(glob.glob(f'{data_dir}/PetImages/Dog/*.jpg')))
print('Dog: ', Dog_count)
```

```
Cat:  12500
Dog:  12500
```

Loading Images in a Dataframe

```
df_dada = pd.DataFrame(columns=['filename', 'label'])
list_dir = ["Cat", "Dog"]
for label in list_dir:
    filenames = list(glob.glob(f'{data_dir}/PetImages/{label}/*.jpg'))
    #labels = [x.split("/")[3] for x in filenames]
    data = pd.DataFrame({"filename": filenames, "label": label})
    data = data.iloc[:1250,]
    df_dada = pd.concat([df_dada, data], ignore_index=True)

print(df_dada.shape)
df_dada.head()
```

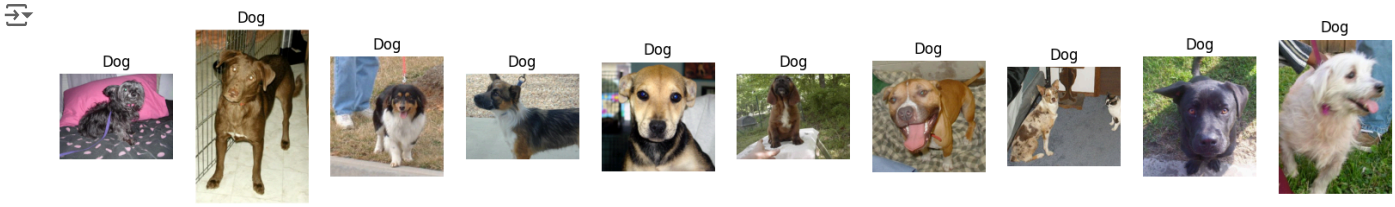
```
(2500, 2)
```

	filename	label
0	/content/sample_data/PetImages/Cat/9508.jpg	Cat
1	/content/sample_data/PetImages/Cat/621.jpg	Cat
2	/content/sample_data/PetImages/Cat/8825.jpg	Cat
3	/content/sample_data/PetImages/Cat/3714.jpg	Cat
4	/content/sample_data/PetImages/Cat/12054.jpg	Cat

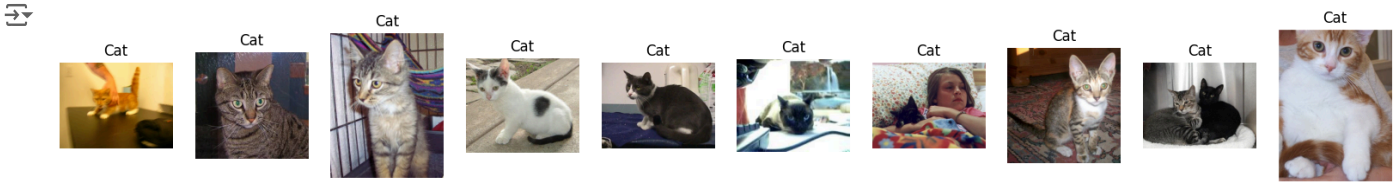
ขั้นตอนถัดไป: สร้างโค้ดด้วย df_dada | ดูแผนภูมิที่แนะนำ | New interactive sheet

Data Exploration

```
plt.figure(figsize=(20,20)) # specifying the overall grid size
plt.subplots_adjust(hspace=0.4)


for i in range(10):

    plt.subplot(1,10,i+1)    # the number of images in the grid is 10*10 (100)
    filename = f'{data_dir}/PetImages/Dog/{str(i)}.jpg'
    image = imread(filename)
    plt.imshow(image)
    plt.title('Dog',fontsize=12)
    plt.axis('off')

plt.show()
```




```
plt.figure(figsize=(20,20)) # specifying the overall grid size
plt.subplots_adjust(hspace=0.4)


for i in range(10):

    plt.subplot(1,10,i+1)    # the number of images in the grid is 10*10 (100)
    filename = f'{data_dir}/PetImages/Cat/{str(i)}.jpg'
    image = imread(filename)
    plt.imshow(image)
    plt.title('Cat',fontsize=12)
    plt.axis('off')
```



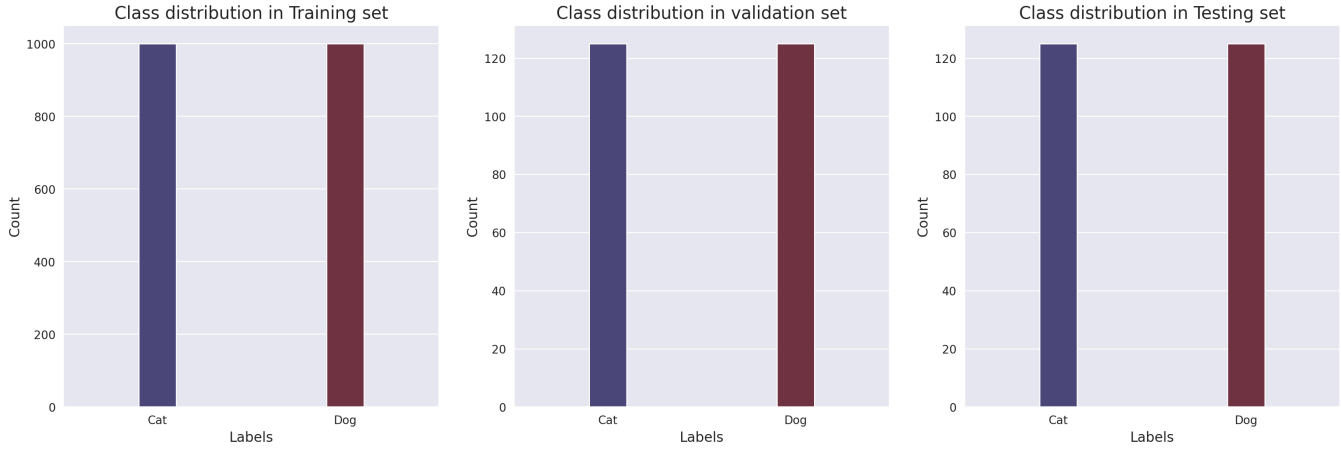
Train Test Split

```
# train test split using dataframe

labels = df_dada['label']

X_train, X_temp = train_test_split(df_dada, test_size=0.2, stratify=labels, random_state = 42)

label_test_val = X_temp['label']

X_test, X_val = train_test_split(X_temp, test_size=0.5, stratify=label_test_val, random_state = 42)

print('The shape of train data',X_train.shape)
print('The shape of test data',X_test.shape)
print('The shape of validation data',X_val.shape)
```

```
The shape of train data (2000, 2)
The shape of test data (250, 2)
The shape of validation data (250, 2)
```

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

label1,count1 = np.unique(X_train.label,return_counts=True)
label2,count2 = np.unique(X_val.label,return_counts=True)
label3,count3 = np.unique(X_test.label,return_counts=True)

uni1 = pd.DataFrame(data=count1,index=labels,columns=['Count1'])
uni2 = pd.DataFrame(data=count2,index=labels,columns=['Count2'])
uni3 = pd.DataFrame(data=count3,index=labels,columns=['Count3'])


plt.figure(figsize=(20,6),dpi=200)
sns.set_style('darkgrid')

plt.subplot(131)
sns.barplot(data=uni1,x=uni1.index,y='Count1',palette='icefire',width=0.2).set_title('Class distribution in Training set',fontsize=15)
plt.xlabel('Labels',fontsize=12)
plt.ylabel('Count',fontsize=12)

plt.subplot(132)
sns.barplot(data=uni2,x=uni2.index,y='Count2',palette='icefire',width=0.2).set_title('Class distribution in validation set',fontsize=15)
plt.xlabel('Labels',fontsize=12)
plt.ylabel('Count',fontsize=12)


plt.subplot(133)
sns.barplot(data=uni3,x=uni3.index,y='Count3',palette='icefire',width=0.2).set_title('Class distribution in Testing set',fontsize=15)
plt.xlabel('Labels',fontsize=12)
plt.ylabel('Count',fontsize=12)

plt.show()
```

```
<ipython-input-14-418fb063fcee>:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(data=uni1,x=uni1.index,y='Count1',palette='icefire',width=0.2).set_title('Class distribution in Training set',fontsize
<ipython-input-14-418fb063fcee>:21: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(data=uni2,x=uni2.index,y='Count2',palette='icefire',width=0.2).set_title('Class distribution in validation set',fontsi
<ipython-input-14-418fb063fcee>:27: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(data=uni3,x=uni3.index,y='Count3',palette='icefire',width=0.2).set_title('Class distribution in Testing set',fontsize=
```



Data Preparation

```
# parameters
target_size = (456, 456)
batch_size = 16
```

Image Data Generator

- The data for will used by flow_from_dataframe.
- The batch size is 16 and the image size is (456, 456).

```
# Creating image data generator
train_datagen = ImageDataGenerator(rescale=1./255,
                                   rotation_range = 15,
                                   horizontal_flip = True,
                                   zoom_range = 0.2,
                                   shear_range = 0.1,
                                   fill_mode = 'nearest',
                                   width_shift_range = 0.1,
                                   height_shift_range = 0.1)

test_datagen = ImageDataGenerator(rescale=1./255)


# Applying image data gernerator to train and test data

train_generator = train_datagen.flow_from_dataframe(X_train,
                                                    directory = None,
                                                    x_col= 'filename',
                                                    y_col= 'label',
                                                    batch_size = batch_size,
                                                    target_size = target_size,
                                                    class_mode='categorical',
                                                    color_mode= 'rgb'
                                                   )
val_generator = test_datagen.flow_from_dataframe(X_val,
                                                 directory = None,
                                                 x_col= 'filename',
                                                 y_col= 'label',
                                                 batch_size = batch_size,
                                                 target_size = target_size,
                                                 class_mode='categorical',
                                                 color_mode= 'rgb'
                                                )

test_generator = test_datagen.flow_from_dataframe(X_test,
                                                  directory = None,
                                                  x_col= 'filename',
                                                  y_col= 'label',
                                                  batch_size = batch_size,
                                                  target_size = target_size,
                                                  class_mode='categorical',
                                                  color_mode= 'rgb'
                                                 )
```

```
Found 2000 validated image filenames belonging to 2 classes.
Found 250 validated image filenames belonging to 2 classes.
Found 250 validated image filenames belonging to 2 classes.
```

```
#label
labels = (train_generator.class_indices)
print(labels)
```

```
{'Cat': 0, 'Dog': 1}
```

```
import cv2
import PIL
from PIL import Image

%matplotlib inline
from matplotlib import pyplot as plt

w = 10
h = 10
fig = plt.figure(figsize=(12, 15))
columns = 4
rows = 4
x, y = train_generator.next()
for i in range(0, columns*rows):
    image = x[i]
    fig.add_subplot(rows, columns, i+1)
    plt.imshow(image)
plt.show()
```

EfficienNet Model

```
# loading pretrained conv base model
from efficientnet.keras import EfficientNetB5 as Net

input_shape = (456, 456, 3)
conv_base = Net(weights='imagenet', include_top=False, input_shape=input_shape)
print(f"Input Shape: {input_shape}")

# create new model with a new classification layer
x = conv_base.output
global_average_layer = layers.GlobalAveragePooling2D(name = 'head_pooling')(x)
dropout_layer = layers.Dropout(0.20,name = 'head_dropout')(global_average_layer)
prediction_layer = layers.Dense(2, activation='softmax',name = 'predict_Cat_Dog')(dropout_layer)

### FC layer
model = models.Model(inputs= conv_base.input, outputs=prediction_layer, name = 'EffNet_Cat_Dog')
```

```
Downloading data from https://github.com/Callidior/keras-applications/releases/download/efficientnet/efficientnet-b5_weights_tf_dim
115515256/115515256 [==============================] - 1s 0us/step
Input Shape: (456, 456, 3)
```

```
### Unfreeze FC layer
print('[INFO]: This is the number of trainable layers '
      'before freezing the conv base:', len(model.trainable_weights))
print('[INFO]: Freezing hidden layers...')
for layer in conv_base.layers:
    layer.trainable = False

print('[INFO]: This is the number of trainable layers '
        'after freezing the conv base:', len(model.trainable_weights))
print('-'*125)
```

```
[INFO]: This is the number of trainable layers before freezing the conv base: 506
[INFO]: Freezing hidden layers...
[INFO]: This is the number of trainable layers after freezing the conv base: 2
-----------------------------------------------------------------------------------------------------------------------------
```

```
model.summary()
```

```
block7c_expand_activation   (None, 15, 15, 3072)        0         ['block7c_expand_bn[0][0]']
(Activation)

block7c_dwconv (DepthwiseC  (None, 15, 15, 3072)        27648     ['block7c_expand_activation[0]
onv2D)                                                            [0]']

block7c_bn (BatchNormaliza  (None, 15, 15, 3072)        12288     ['block7c_dwconv[0][0]']
tion)

block7c_activation (Activa  (None, 15, 15, 3072)        0         ['block7c_bn[0][0]']
tion)

block7c_se_squeeze (Global  (None, 3072)                0         ['block7c_activation[0][0]']
AveragePooling2D)

block7c_se_reshape (Reshap  (None, 1, 1, 3072)          0         ['block7c_se_squeeze[0][0]']
e)

block7c_se_reduce (Conv2D)  (None, 1, 1, 128)           393344    ['block7c_se_reshape[0][0]']

block7c_se_expand (Conv2D)  (None, 1, 1, 3072)          396288    ['block7c_se_reduce[0][0]']

block7c_se_excite (Multipl  (None, 15, 15, 3072)        0         ['block7c_activation[0][0]',
y)                                                                 'block7c_se_expand[0][0]']

block7c_project_conv (Conv  (None, 15, 15, 512)         1572864   ['block7c_se_excite[0][0]']
2D)

block7c_project_bn (BatchN  (None, 15, 15, 512)         2048      ['block7c_project_conv[0][0]']
ormalization)

block7c_drop (FixedDropout  (None, 15, 15, 512)         0         ['block7c_project_bn[0][0]']
)

block7c_add (Add)           (None, 15, 15, 512)         0         ['block7c_drop[0][0]',
                                                                   'block7b_add[0][0]']

top_conv (Conv2D)           (None, 15, 15, 2048)        1048576   ['block7c_add[0][0]']

top_bn (BatchNormalization  (None, 15, 15, 2048)        8192      ['top_conv[0][0]']
)

top_activation (Activation  (None, 15, 15, 2048)        0         ['top_bn[0][0]']
)

head_pooling (GlobalAverag  (None, 2048)                0         ['top_activation[0][0]']
ePooling2D)

head_dropout (Dropout)      (None, 2048)                0         ['head_pooling[0][0]']

predict_Cat_Dog (Dense)     (None, 2)                   4098      ['head_dropout[0][0]']

==============================================================================================
Total params: 28517618 (108.79 MB)
Trainable params: 4098 (16.01 KB)
Non-trainable params: 28513520 (108.77 MB)
_____________________________________________________________________________________________
```

Callbacks

- ReduceLROnPlateau : Reduce learning rate when a metric has stopped improving.

```
from tensorflow.keras.callbacks import ReduceLROnPlateau,EarlyStopping

learning_rate_reduction = ReduceLROnPlateau(monitor = 'val_accuracy',
                                            patience=2,
                                            factor=0.5,
                                            min_lr = 0.00001,
                                            verbose = 1)
```

Compile the model

```
from keras.optimizers import Adam

lr=1e-4
#Training
model.compile(loss='categorical_crossentropy',
              optimizer=optimizers.RMSprop(learning_rate=lr),
              metrics=['acc'])
```

Fit the model

```
num_epochs = 20
history_eff = model.fit(train_generator,
                        epochs = num_epochs,
                        validation_data = val_generator,
                        callbacks=learning_rate_reduction)
```

```
Epoch 1/20
125/125 [==============================] - ETA: 0s - loss: 0.1114 - acc: 0.9745WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 160s 1s/step - loss: 0.1114 - acc: 0.9745 - val_loss: 0.0909 - val_acc: 0.9840 - lr: 1
Epoch 2/20
125/125 [==============================] - ETA: 0s - loss: 0.0857 - acc: 0.9825WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 142s 1s/step - loss: 0.0857 - acc: 0.9825 - val_loss: 0.0765 - val_acc: 0.9840 - lr: 1
Epoch 3/20
125/125 [==============================] - ETA: 0s - loss: 0.0757 - acc: 0.9805WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 141s 1s/step - loss: 0.0757 - acc: 0.9805 - val_loss: 0.0692 - val_acc: 0.9880 - lr: 1
Epoch 4/20
125/125 [==============================] - ETA: 0s - loss: 0.0708 - acc: 0.9810WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 141s 1s/step - loss: 0.0708 - acc: 0.9810 - val_loss: 0.0650 - val_acc: 0.9880 - lr: 1
Epoch 5/20
125/125 [==============================] - ETA: 0s - loss: 0.0573 - acc: 0.9875WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 145s 1s/step - loss: 0.0573 - acc: 0.9875 - val_loss: 0.0620 - val_acc: 0.9880 - lr: 1
Epoch 6/20
125/125 [==============================] - ETA: 0s - loss: 0.0615 - acc: 0.9840WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 141s 1s/step - loss: 0.0615 - acc: 0.9840 - val_loss: 0.0603 - val_acc: 0.9880 - lr: 1
Epoch 7/20
125/125 [==============================] - ETA: 0s - loss: 0.0562 - acc: 0.9865WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 136s 1s/step - loss: 0.0562 - acc: 0.9865 - val_loss: 0.0581 - val_acc: 0.9880 - lr: 1
Epoch 8/20
125/125 [==============================] - ETA: 0s - loss: 0.0505 - acc: 0.9870WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 142s 1s/step - loss: 0.0505 - acc: 0.9870 - val_loss: 0.0573 - val_acc: 0.9840 - lr: 1
Epoch 9/20
125/125 [==============================] - ETA: 0s - loss: 0.0482 - acc: 0.9880WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 141s 1s/step - loss: 0.0482 - acc: 0.9880 - val_loss: 0.0568 - val_acc: 0.9880 - lr: 1
Epoch 10/20
125/125 [==============================] - ETA: 0s - loss: 0.0508 - acc: 0.9855WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 141s 1s/step - loss: 0.0508 - acc: 0.9855 - val_loss: 0.0563 - val_acc: 0.9840 - lr: 1
Epoch 11/20
125/125 [==============================] - ETA: 0s - loss: 0.0476 - acc: 0.9855WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 149s 1s/step - loss: 0.0476 - acc: 0.9855 - val_loss: 0.0557 - val_acc: 0.9840 - lr: 1
Epoch 12/20
125/125 [==============================] - ETA: 0s - loss: 0.0444 - acc: 0.9885WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 145s 1s/step - loss: 0.0444 - acc: 0.9885 - val_loss: 0.0548 - val_acc: 0.9840 - lr: 1
Epoch 13/20
125/125 [==============================] - ETA: 0s - loss: 0.0415 - acc: 0.9910WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 141s 1s/step - loss: 0.0415 - acc: 0.9910 - val_loss: 0.0550 - val_acc: 0.9840 - lr: 1
Epoch 14/20
125/125 [==============================] - ETA: 0s - loss: 0.0470 - acc: 0.9870WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 142s 1s/step - loss: 0.0470 - acc: 0.9870 - val_loss: 0.0546 - val_acc: 0.9880 - lr: 1
Epoch 15/20
125/125 [==============================] - ETA: 0s - loss: 0.0405 - acc: 0.9885WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 142s 1s/step - loss: 0.0405 - acc: 0.9885 - val_loss: 0.0544 - val_acc: 0.9840 - lr: 1
Epoch 16/20
125/125 [==============================] - ETA: 0s - loss: 0.0410 - acc: 0.9905WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 142s 1s/step - loss: 0.0410 - acc: 0.9905 - val_loss: 0.0549 - val_acc: 0.9840 - lr: 1
Epoch 17/20
125/125 [==============================] - ETA: 0s - loss: 0.0451 - acc: 0.9880WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 145s 1s/step - loss: 0.0451 - acc: 0.9880 - val_loss: 0.0549 - val_acc: 0.9840 - lr: 1
Epoch 18/20
125/125 [==============================] - ETA: 0s - loss: 0.0416 - acc: 0.9865WARNING:tensorflow:Learning rate reduction is conc
```

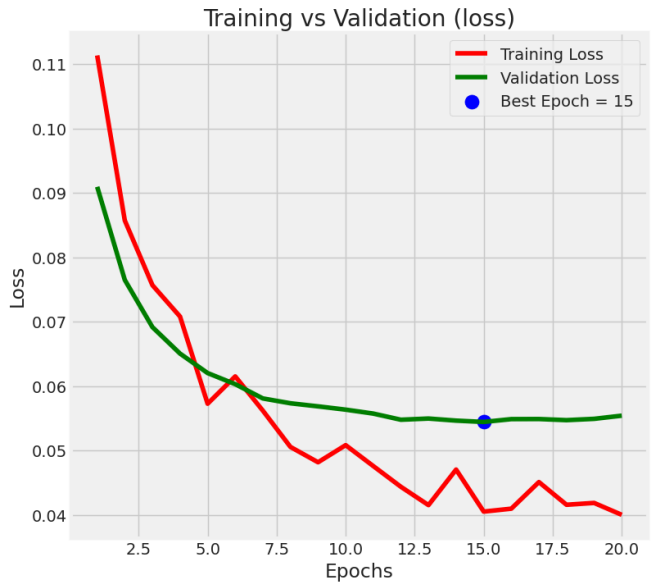
```
125/125 [==============================] - 145s 1s/step - loss: 0.0416 - acc: 0.9865 - val_loss: 0.0547 - val_acc: 0.9840 - lr: 1
Epoch 19/20
125/125 [==============================] - ETA: 0s - loss: 0.0418 - acc: 0.9880WARNING:tensorflow:Learning rate reduction is conc
125/125 [==============================] - 141s 1s/step - loss: 0.0418 - acc: 0.9880 - val_loss: 0.0549 - val_acc: 0.9840 - lr: 1
```

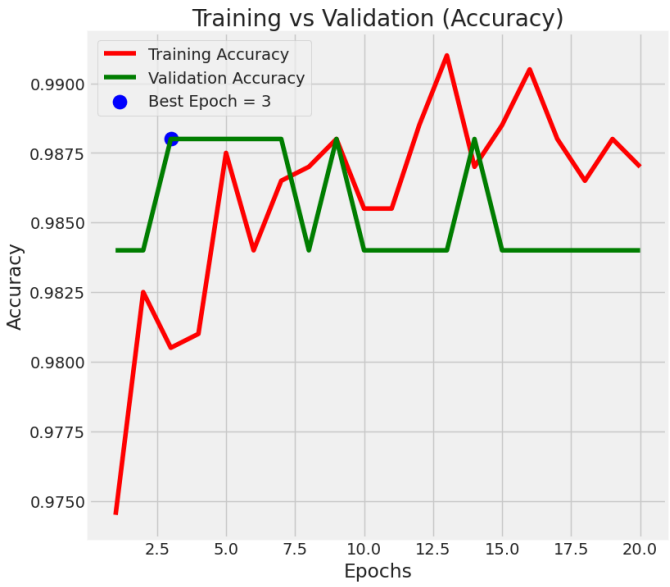
```
## Set up model path
#Categorical Crossentropy
modelName  = "EffNetB5_CategoricalCrossentropy_Cat_Dog_Classes.h5"
Model2save = f"/content/drive/MyDrive/{modelName}"
model.save(Model2save)
### print
print(f"[INFO]: Save Model as: {Model2save}")
print(f"*"*100)
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079: UserWarning: You are saving your model as an HDF5 file vi
  saving_api.save_model(
[INFO]: Save Model as: /content/drive/MyDrive/EffNetB5_CategoricalCrossentropy_Cat_Dog_Classes.h5
****************************************************************************************************
```

Plot the results

```
train_acc = history_eff.history['acc']
train_loss = history_eff.history['loss']

val_acc = history_eff.history['val_acc']
val_loss = history_eff.history['val_loss']

index_loss = np.argmin(val_loss)
index_acc = np.argmax(val_acc)

val_lowest = val_loss[index_loss]
val_highest = val_acc[index_acc]

Epochs = [i+1 for i in range(len(train_acc))]

loss_label = f'Best Epoch = {str(index_loss + 1)}'
acc_label = f'Best Epoch = {str(index_acc + 1)}'

plt.figure(figsize= (20,8))
plt.style.use('fivethirtyeight')

plt.subplot(1,2,1)
plt.plot(Epochs , train_loss , 'r', label = 'Training Loss')
plt.plot(Epochs , val_loss , 'g' , label = 'Validation Loss')
plt.scatter(index_loss +1 , val_lowest , s = 150 , c = 'blue' , label = loss_label)
plt.title('Training vs Validation (loss)')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.subplot(1,2,2)
plt.plot(Epochs , train_acc , 'r', label= 'Training Accuracy')
plt.plot(Epochs , val_acc , 'g' , label = 'Validation Accuracy')
plt.scatter(index_acc + 1 , val_highest , s= 150 , c = 'blue' , label= acc_label)
plt.title('Training vs Validation (Accuracy)')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.tight_layout
plt.show();
```





Evaluation

```
# Evaluvate for train generator
loss,acc = model.evaluate(train_generator,batch_size = val_generator, verbose = 0)

print('The accuracy of the model for training data is:',acc*100)
print('The Loss of the model for training data is:',loss)

# Evaluvate for validation generator
loss,acc = model.evaluate(val_generator,batch_size = val_generator, verbose = 0)

print('The accuracy of the model for validation data is:',acc*100)
print('The Loss of the model for validation data is:',loss)
```

```
The accuracy of the model for training data is: 99.40000176429749
The Loss of the model for training data is: 0.028960317373275757
The accuracy of the model for validation data is: 98.4000027179718
The Loss of the model for validation data is: 0.055402692407369614
```

Prediction

```
import os
import numpy as np
import efficientnet.tfkeras
from tensorflow.keras.models import load_model
import tensorflow as tf
import pandas as pd

from tensorflow.keras.preprocessing import image
from tensorflow.keras.utils import get_file
```

Load model

```
model_dir = "/content/drive/MyDrive/EffNetB5_CategoricalCrossentropy_Cat_Dog_Classes.h5"
model_B5 = load_model(model_dir)
height = width = model_B5.input_shape[1]
print(height, width)
```

```
456 456
```

```
model_B5.summary()
```

```
Model: "EffNet_Cat_Dog"
_________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
=================================================================================================
 input_3 (InputLayer)        [(None, 456, 456, 3)]     0         []

 stem_conv (Conv2D)          (None, 228, 228, 48)      1296      ['input_3[0][0]']

 stem_bn (BatchNormalizatio  (None, 228, 228, 48)      192       ['stem_conv[0][0]']
 n)

 stem_activation (Activatio  (None, 228, 228, 48)      0         ['stem_bn[0][0]']
 n)

 block1a_dwconv (DepthwiseC  (None, 228, 228, 48)      432       ['stem_activation[0][0]']
 onv2D)

 block1a_bn (BatchNormaliza  (None, 228, 228, 48)      192       ['block1a_dwconv[0][0]']
 tion)

 block1a_activation (Activa  (None, 228, 228, 48)      0         ['block1a_bn[0][0]']
 tion)

 block1a_se_squeeze (Global  (None, 48)                0         ['block1a_activation[0][0]']
 AveragePooling2D)

 block1a_se_reshape (Reshap  (None, 1, 1, 48)          0         ['block1a_se_squeeze[0][0]']
 e)

 block1a_se_reduce (Conv2D)  (None, 1, 1, 12)          588       ['block1a_se_reshape[0][0]']

 block1a_se_expand (Conv2D)  (None, 1, 1, 48)          624       ['block1a_se_reduce[0][0]']

 block1a_se_excite (Multipl  (None, 228, 228, 48)      0         ['block1a_activation[0][0]',
 y)                                                               'block1a_se_expand[0][0]']

 block1a_project_conv (Conv  (None, 228, 228, 24)      1152      ['block1a_se_excite[0][0]']
 2D)

 block1a_project_bn (BatchN  (None, 228, 228, 24)      96        ['block1a_project_conv[0][0]']
 ormalization)

 block1b_dwconv (DepthwiseC  (None, 228, 228, 24)      216       ['block1a_project_bn[0][0]']
 onv2D)

 block1b_bn (BatchNormaliza  (None, 228, 228, 24)      96        ['block1b_dwconv[0][0]']
 tion)

 block1b_activation (Activa  (None, 228, 228, 24)      0         ['block1b_bn[0][0]']
 tion)

 block1b_se_squeeze (Global  (None, 24)                0         ['block1b_activation[0][0]']
 AveragePooling2D)

 block1b_se_reshape (Reshap  (None, 1, 1, 24)          0         ['block1b_se_squeeze[0][0]']
 e)

 block1b_se_reduce (Conv2D)  (None, 1, 1, 6)           150       ['block1b_se_reshape[0][0]']
```

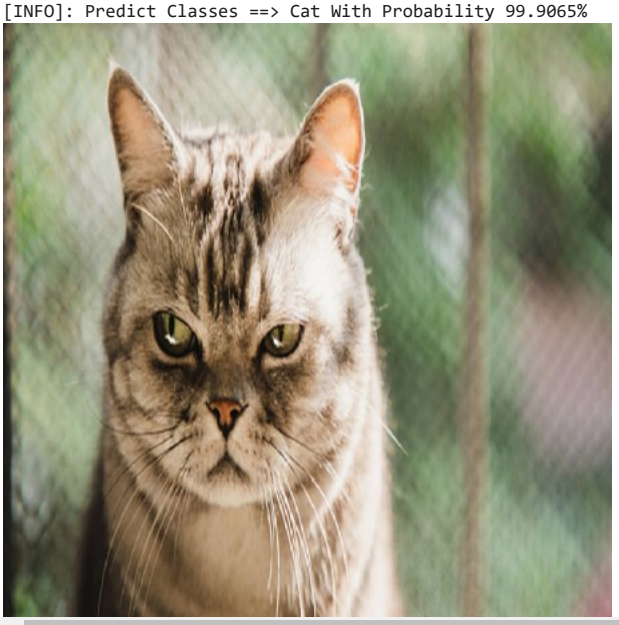
```
# label_dict = {0: 'Cat', 1: 'Dog'}
# print(label_dict)

label_dict = dict((v,k) for k,v in labels.items())
print(label_dict)
```

```
{0: 'Cat', 1: 'Dog'}
```

```
def process_image(img_path):
    # Read the image and resize it
    img = image.load_img(img_path, target_size=(height, width))
    # Convert it to a Numpy array with target shape.
    x = image.img_to_array(img)
    # Reshape
    x = x.reshape((1,) + x.shape)
    x /= 255.

    return x


def predict(img_path, model_B5):
    img_x = process_image(img_path)
    predict = model_B5.predict([img_x])
    predict0 = predict[0]
    result = np.argmax(predict0)
    pred_label = label_dict[result]
```

```
    pred_prob = predict0[result]

    print(f"[INFO]: Predict Classes ==> {pred_label} With Probability {pred_prob*100:.4f}%")


img_path1 = "/content/drive/MyDrive/_119932207_indifferentcatgettyimages.jpg"
predict(img_path1, model_B5)
image.load_img(img_path1, target_size=(height, width))
```

```
1/1 [==============================] - 0s 307ms/step
[INFO]: Predict Classes ==> Cat With Probability 99.9065%
```



```
img_path2 = "/content/drive/MyDrive/images.jpg"
predict(img_path2, model_B5)
image.load_img(img_path2, target_size=(height, width))
```

```
1/1 [==============================] - 0s 136ms/step
[INFO]: Predict Classes ==> Dog With Probability 97.3297%
```

```
img_path3 = "/content/drive/MyDrive/Hero Pedigree Cats.jpg"
predict(img_path3, model_B5)
image.load_img(img_path3, target_size=(height, width))
```

```
1/1 [==============================] - 0s 93ms/step
[INFO]: Predict Classes ==> Cat With Probability 99.9849%
```

เริ่มเขียนโค้ดหรือสร้างโดยใช้ AI

เริ่มเขียนโค้ดหรือสร้างโดยใช้ AI

เริ่มเขียนโค้ดหรือสร้างโดยใช้ AI

Exercise

ทดลองนำภาพ นก รถยนต์ เสือ สิงโต และ สุนัขจิ้งจอก มาทำการทำนายและรายงานผลการทำนายแต่ละภาพ

```
img_path4 = "/content/drive/MyDrive/black-maned-lion-shem-compion-786x500.jpg"
predict(img_path4, model_B5)
image.load_img(img_path4, target_size=(height, width))
```

```
1/1 [==============================] - 0s 99ms/step
[INFO]: Predict Classes ==> Dog With Probability 87.8515%
```

```
img_path5 = "/content/sample_data/PetImages/About_WildBird-mobile.jpg"
predict(img_path5, model_B5)
image.load_img(img_path4, target_size=(height, width))
```

```
img_path6 = "/content/sample_data/PetImages/best-porsche-concept-cars-5-porsche-mission-e-concept-2015-goodwood-11092020.jpg"
predict(img_path6, model_B5)
image.load_img(img_path4, target_size=(height, width))
```

```
img_path7 = "/content/sample_data/PetImages/tiger_pho_0000.jpg"
predict(img_path7, model_B5)
image.load_img(img_path4, target_size=(height, width))


img_path8 = "/content/sample_data/PetImages/Lion_pose_6649531395.jpg"
predict(img_path8, model_B5)
image.load_img(img_path4, target_size=(height, width))


img_path9 = "/content/sample_data/PetImages/fox.jpg"
predict(img_path9, model_B5)
image.load_img(img_path4, target_size=(height, width))
```

Exercise

Print output ของ network

```
label_dict
img_x = process_image(img_path4)
predict4 = model_B5.predict(img_x)
predict4
img_x = process_image(img_path5)
predict5 = model_B5.predict(img_x)
predict5
img_x = process_image(img_path6)
predict6 = model_B5.predict(img_x)
predict6
img_x = process_image(img_path7)
predict7 = model_B5.predict(img_x)
predict7
img_x = process_image(img_path8)
predict8 = model_B5.predict(img_x)
predict8
img_x = process_image(img_path9)
predict9 = model_B5.predict(img_x)
predict9
```

Evaluating Model Performance

- Confusion matrix

```
print(X_test.shape)
X_test.head()
```

```
(250, 2)
```

	filename	label
1095	/content/sample_data/PetImages/Cat/184.jpg	Cat
1179	/content/sample_data/PetImages/Cat/11194.jpg	Cat
925	/content/sample_data/PetImages/Dog/8486.jpg	Dog
1019	/content/sample_data/PetImages/Dog/5322.jpg	Dog
556	/content/sample_data/PetImages/Cat/9778.jpg	Cat

```
# label_dict = {0: 'Cat', 1: 'Dog'}
# print(label_dict)

label_dict = dict((v,k) for k,v in labels.items())
print(label_dict)
```

```
{0: 'Cat', 1: 'Dog'}
```

```
from tensorflow.keras.preprocessing import image

def predict_image(img_path):
    # Read the image and resize it
    img = image.load_img(img_path, target_size=(height, width))
    # Convert it to a Numpy array with target shape.
    x = image.img_to_array(img)
```

```
    # Reshape
    x = x.reshape((1,) + x.shape)
    x /= 255.
    result = model.predict([x])

    return result[0]


#Predict
pred_list = list()
prob_list = list()
img_path=X_test['filename'].tolist()
for i in range(0,len(img_path)):
    predict = predict_image(img_path[i])
    result = np.argmax(predict)
    pred_list.append(label_dict[result])
    prob_list.append(predict[result])
```

```
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 47ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 47ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 42ms/step
```

```
predict
```

```
array([0.00679942, 0.9932006 ], dtype=float32)
```

```
print(len(pred_list))
print(len(prob_list))

X_test['category'] = pred_list
X_test['Prob'] = prob_list
X_test.head()
```

```
250
250
```

	filename	label	category	Prob
1095	/content/sample_data/PetImages/Cat/184.jpg	Cat	Cat	0.999494
1179	/content/sample_data/PetImages/Cat/11194.jpg	Cat	Cat	0.812446

```
import numpy as np
from sklearn.metrics import confusion_matrix

act = X_test['label'].array
pred = X_test['category'].array

cmat = confusion_matrix(act, pred)
print('classifier accuracy = {}%'.format((100.*np.trace(cmat))/(np.sum(cmat))))

#Marking the Confusion Matrix
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(act, pred))#performance
```

```
classifier accuracy = 99.2%
              precision    recall  f1-score   support

         Cat       0.99      0.99      0.99       125
         Dog       0.99      0.99      0.99       125

    accuracy                           0.99       250
   macro avg       0.99      0.99      0.99       250
weighted avg       0.99      0.99      0.99       250
```

Confusion matrix

```
#create CF
data = {'Actual': act,'Predicted' : pred,}
df = pd.DataFrame(data, columns=['Actual','Predicted'])
conf_mat = pd.crosstab(df['Actual'],df['Predicted'],rownames=['Actual'],colnames=['Predicted'])

#Confusion matrix
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
cm = confusion_matrix(act, pred)

#plot Confusion matrix
import seaborn as sns
sns.set()
fig, ax = plt.subplots(figsize=(8, 5))

ax = sns.heatmap(conf_mat, annot=True, fmt="d", cmap="YlGnBu") #Blues,Oranges,Reds
ax.set_title('Confusion matrix',fontsize=20)
ax.set_ylabel('True label',fontsize=18)
ax.set_xlabel('Predicted label',fontsize=18)
```

```
Text(0.5, 1.2499999999999805, 'Predicted label')
```