Name: Mohan R

Roll no: 611219106303

Date: 08/10/2022

1. Unzip Dataset

```
!unzip '/content/Flowers-Dataset.zip'
```

```
       inflating: flowers/daisy/14350958832_29bdd3a254.jpg
       inflating: flowers/daisy/14354051035_1037b30421_n.jpg
       inflating: flowers/daisy/14372713423_61e2daae88.jpg
       inflating: flowers/daisy/14399435971_ea5868c792.jpg
       inflating: flowers/daisy/14402451388_56545a374a_n.jpg
       inflating: flowers/daisy/144076848_57e1d662e3_m.jpg
       inflating: flowers/daisy/144099102_bf63a41e4f_n.jpg
       inflating: flowers/daisy/1441939151_b271408c8d_n.jpg
       inflating: flowers/daisy/14421389519_d5fd353eb4.jpg
       inflating: flowers/daisy/144603918_b9de002f60_m.jpg
       inflating: flowers/daisy/14471433500_cdaa22e3ea_m.jpg
       inflating: flowers/daisy/14485782498_fb342ec301.jpg
       inflating: flowers/daisy/14507818175_05219b051c_m.jpg
       inflating: flowers/daisy/14523675369_97c31d0b5b.jpg
       inflating: flowers/daisy/14551098743_2842e7a004_n.jpg
       inflating: flowers/daisy/14554906452_35f066ffe9_n.jpg

       inflating: flowers/daisy/14564545365_1f1d267bf1_n.jpg
       inflating: flowers/daisy/14569895116_32f0dcb0f9.jpg
       inflating: flowers/daisy/14591326135_930703dbed_m.jpg
       inflating: flowers/daisy/14600779226_7bbc288d40_m.jpg
       inflating: flowers/daisy/14613443462_d4ed356201.jpg
       inflating: flowers/daisy/14621687774_ec52811acd_n.jpg
       inflating: flowers/daisy/14674743211_f68b13f6d9.jpg
       inflating: flowers/daisy/14698531521_0c2f0c6539.jpg
       inflating: flowers/daisy/147068564_32bb4350cc.jpg
       inflating: flowers/daisy/14707111433_cce08ee007.jpg
       inflating: flowers/daisy/14716799982_ed6d626a66.jpg
       inflating: flowers/daisy/14816364517_2423021484_m.jpg
       inflating: flowers/daisy/14866200659_6462c723cb_m.jpg
       inflating: flowers/daisy/14907815010_bff495449f.jpg
       inflating: flowers/daisy/14921511479_7b0a647795.jpg
       inflating: flowers/daisy/15029936576_8d6f96c72c_n.jpg
       inflating: flowers/daisy/15100730728_a450c5f422_n.jpg
       inflating: flowers/daisy/15207766_fc2f1d692c_n.jpg
       inflating: flowers/daisy/15306268004_4680ba95e1.jpg
       inflating: flowers/daisy/153210866_03cc9f2f36.jpg
       inflating: flowers/daisy/15327813273_06cdf42210.jpg
       inflating: flowers/daisy/154332674_453cea64f4.jpg
       inflating: flowers/daisy/15760153042_a2a90e9da5_m.jpg
       inflating: flowers/daisy/15760811380_4d686c892b_n.jpg
       inflating: flowers/daisy/15784493690_b1858cdb2b_n.jpg
       inflating: flowers/daisy/15813862117_dedcd1c56f_m.jpg
       inflating: flowers/daisy/15853110333_229c439e7f.jpg
       inflating: flowers/daisy/158869618_f1a6704236_n.jpg
       inflating: flowers/daisy/16020253176_60f2a6a5ca_n.jpg
       inflating: flowers/daisy/16025261368_911703a536_n.jpg
       inflating: flowers/daisy/16056178001_bebc2153fe_n.jpg
       inflating: flowers/daisy/16121105382_b96251e506_m.jpg
       inflating: flowers/daisy/16161045294_70c76ce846_n.jpg
       inflating: flowers/daisy/162362896_99c7d851c8_n.jpg
       inflating: flowers/daisy/162362897_1d21b70621_m.jpg
       inflating: flowers/daisy/16291797949_a1b1b7c2bd_n.jpg
       inflating: flowers/daisy/16323838000_3818bce5c6_n.jpg
       inflating: flowers/daisy/16360180712_b72695928c_n.jpg
       inflating: flowers/daisy/163978992_8128b49d3e_n.jpg
       inflating: flowers/daisy/16401288243_36112bd52f_m.jpg
       inflating: flowers/daisy/16482676953_5296227d40_n.jpg
       inflating: flowers/daisy/16492248512_61a57dfec1_m.jpg
```

```
#import libraies

import warnings
warnings.filterwarnings("ignore")


import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Activation,Dropout,Conv2D,Flatten,MaxPool2D,Reshape,InputLayer
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img,img_to_array
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
```

2.Image Augmentation

```
path = 'flowers/'


train_data_gen = ImageDataGenerator(rescale = 1./255,
                                    shear_range = 0.2,
                                    zoom_range = 0.2,
                                    horizontal_flip = True,
                                    validation_split = 0.30)
test_data_gen = ImageDataGenerator(rescale = 1./255,validation_split = 0.30)


training_set = train_data_gen.flow_from_directory(path,
                                                  target_size=(64,64),
                                                  batch_size=100,
                                                  class_mode='categorical',
                                                  shuffle=True,
```

```
                                                color_mode='rgb',
                                                subset = 'training')

testing_set = test_data_gen.flow_from_directory(path,
                                                target_size=(64,64),
                                                batch_size=100,
                                                class_mode='categorical',
                                                shuffle=True,
                                                color_mode='rgb',
                                                subset = 'validation')
```

```
Found 3024 images belonging to 5 classes.
Found 1293 images belonging to 5 classes.
```

3. Create Model

```
model = Sequential()
```

4. Add Layers (Convolution,MaxPooling,Flatten,Dense-(Hidden Layers),Output)

```
#convolution and Pooling layer 1
model.add(Conv2D(filters=48,kernel_size=3,activation='relu',input_shape=(64,64,3)))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Dropout(0.2))

#convolution and Pooling layer 2
model.add(Conv2D(filters=32,kernel_size=3,activation='relu'))
model.add(MaxPool2D(pool_size=2,strides=2))
model.add(Dropout(0.2))

#Flattening the images
model.add(Flatten())

#Fully Connected layers
model.add(Dense(64,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(5,activation='softmax'))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 48)        1344

 max_pooling2d (MaxPooling2D  (None, 31, 31, 48)       0
 )

 dropout (Dropout)           (None, 31, 31, 48)        0

 conv2d_1 (Conv2D)           (None, 29, 29, 32)        13856

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 dropout_1 (Dropout)         (None, 14, 14, 32)        0

 flatten (Flatten)           (None, 6272)              0

 dense (Dense)               (None, 64)                401472

 dropout_2 (Dropout)         (None, 64)                0

 dense_1 (Dense)             (None, 5)                 325

=================================================================
Total params: 416,997
Trainable params: 416,997
Non-trainable params: 0
_________________________________________________________________
```

5. Compile the model

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

6. Fit the model

```
early_stop = EarlyStopping(monitor='val_accuracy',
                           patience=5,verbose=1,mode='auto')

lr = ReduceLROnPlateau(monitor='val_accuracy',
                       factor=0.2,patience=5,
                       min_lr=0.00001)

callback = [early_stop,lr]
```

Train the model

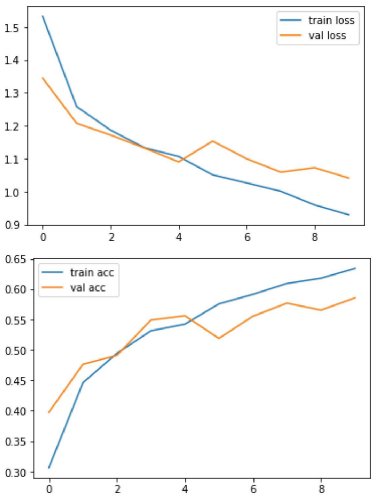
```
result = model.fit(x=training_set, validation_data=testing_set, epochs=10)
```

```
Epoch 1/10
31/31 [==============================] - 30s 927ms/step - loss: 1.5322 - accuracy: 0.3065 - val_loss: 1.3445 - val_accuracy: 0.3975
Epoch 2/10
31/31 [==============================] - 28s 910ms/step - loss: 1.2579 - accuracy: 0.4461 - val_loss: 1.2074 - val_accuracy: 0.4764
Epoch 3/10
31/31 [==============================] - 28s 906ms/step - loss: 1.1863 - accuracy: 0.4944 - val_loss: 1.1717 - val_accuracy: 0.4911
Epoch 4/10
```

```
31/31 [==============================] - 28s 908ms/step - loss: 1.1329 - accuracy: 0.5314 - val_loss: 1.1320 - val_accuracy: 0.5491
Epoch 5/10
31/31 [==============================] - 30s 949ms/step - loss: 1.1066 - accuracy: 0.5423 - val_loss: 1.0902 - val_accuracy: 0.5561
Epoch 6/10
31/31 [==============================] - 28s 911ms/step - loss: 1.0508 - accuracy: 0.5757 - val_loss: 1.1533 - val_accuracy: 0.5189
Epoch 7/10
31/31 [==============================] - 28s 907ms/step - loss: 1.0263 - accuracy: 0.5913 - val_loss: 1.0996 - val_accuracy: 0.5553
Epoch 8/10
31/31 [==============================] - 28s 934ms/step - loss: 1.0011 - accuracy: 0.6091 - val_loss: 1.0592 - val_accuracy: 0.5770
Epoch 9/10
31/31 [==============================] - 28s 910ms/step - loss: 0.9595 - accuracy: 0.6177 - val_loss: 1.0725 - val_accuracy: 0.5654
Epoch 10/10
31/31 [==============================] - 29s 943ms/step - loss: 0.9296 - accuracy: 0.6339 - val_loss: 1.0412 - val_accuracy: 0.5855
```

Loss and Accuracy check using plot

```
#plot the loss
plt.plot(result.history['loss'], label='train loss')
plt.plot(result.history['val_loss'], label='val loss')
plt.legend()
plt.show()

# plot the accuracy
plt.plot(result.history['accuracy'], label='train acc')
plt.plot(result.history['val_accuracy'], label='val acc')
plt.legend()
plt.show()
```



7. Save the model

```
model.save('flower.h5')
```

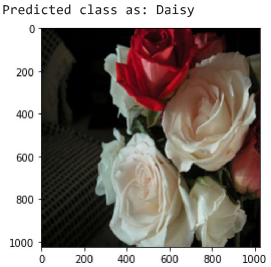
8. Test the model

```
training_set.class_indices
```

```
{'daisy': 0, 'dandelion': 1, 'rose': 2, 'sunflower': 3, 'tulip': 4}
```

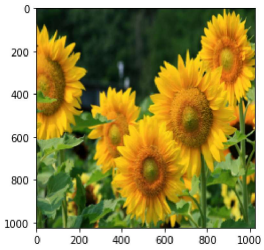
```
classes = ['Daisy','Dandelion','Rose','Sunflower','Tulip']
def testing(img):
    img = image.load_img(img,target_size=(64,64))
    x = image.img_to_array(img)
    x = np.expand_dims(x,axis=0)
    pred = np.argmax(model.predict(x))
    return print("Predicted class as:",classes[pred])

def img_show(img):
    img1 = image.load_img(img,target_size=(1024,1024))
    plt.imshow(img1)


#test1
img_show('/content/flowers/rose/102501987_3cdb8e5394_n.jpg')
testing('/content/flowers/rose/102501987_3cdb8e5394_n.jpg')
```

```
Predicted class as: Daisy
```



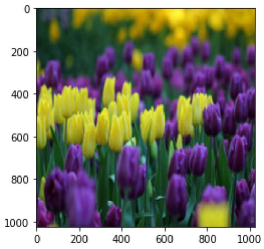
```
#test2
img_show('/content/flowers/sunflower/1008566138_6927679c8a.jpg')
testing('/content/flowers/sunflower/1008566138_6927679c8a.jpg')
```

```
Predicted class as: Sunflower
```



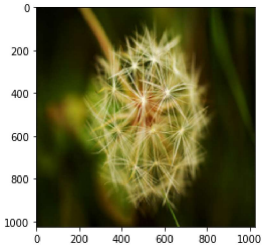
```
#test3
img_show('/content/flowers/tulip/14039129738_cc3ac0a623_n.jpg')
testing('/content/flowers/tulip/14039129738_cc3ac0a623_n.jpg')
```

```
Predicted class as: Tulip
```



```
#test4
img_show('/content/flowers/dandelion/10043234166_e6dd915111_n.jpg')
testing('/content/flowers/dandelion/10043234166_e6dd915111_n.jpg')
```

```
Predicted class as: Daisy
```



```
#test5
img_show('/content/flowers/daisy/10172567486_2748826a8b.jpg')
testing('/content/flowers/daisy/10172567486_2748826a8b.jpg')
```

```
Predicted class as: Daisy
```