```
# 1) DOWNLOAD DATA SET AND UNZIP
!unzip '/content/Flowers-Datasets.zip'
```

```
  inflating: flowers/Testing/rose/33184832190_e5411d91a4_n.jpg
  inflating: flowers/Testing/rose/33185001420_dc8f571c4f_n.jpg
  inflating: flowers/Testing/rose/33411423082_8150d9254e_n.jpg
  inflating: flowers/Testing/rose/33568244695_7ec846bcc6_n.jpg
  inflating: flowers/Testing/rose/33568345265_e4f7d0fe45_n.jpg
   creating: flowers/Testing/sunflower/
  inflating: flowers/Testing/sunflower/5015462205_440898fe41_n.jpg
  inflating: flowers/Testing/sunflower/5018120483_cc0421b176_m.jpg
  inflating: flowers/Testing/sunflower/5020805135_1219d7523d.jpg
  inflating: flowers/Testing/sunflower/5020805619_6c710793f7.jpg
  inflating: flowers/Testing/sunflower/5025805406_033cb03475_n.jpg
  inflating: flowers/Testing/sunflower/5027895361_ace3b731e5_n.jpg
  inflating: flowers/Testing/sunflower/5028817729_f04d32bac8_n.jpg
  inflating: flowers/Testing/sunflower/5032376020_2ed312306c.jpg
  inflating: flowers/Testing/sunflower/5037531593_e2daf4c7f1.jpg
  inflating: flowers/Testing/sunflower/5037790727_57c527494f.jpg
  inflating: flowers/Testing/sunflower/5042785753_392cc4e74d_n.jpg
  inflating: flowers/Testing/sunflower/5043404000_9bc16cb7e5_m.jpg
  inflating: flowers/Testing/sunflower/5043409092_5b12cc985a_m.jpg
  inflating: flowers/Testing/sunflower/5043409856_395300dbe5_m.jpg
  inflating: flowers/Testing/sunflower/5067864967_19928ca94c_m.jpg
  inflating: flowers/Testing/sunflower/5076821914_c21b58fd4c_m.jpg
  inflating: flowers/Testing/sunflower/5091281256_648c37d7c1_n.jpg
  inflating: flowers/Testing/sunflower/5115925320_ed9ca5b2d1_n.jpg
  inflating: flowers/Testing/sunflower/5139969631_743880e01d_n.jpg
  inflating: flowers/Testing/sunflower/5139969871_c9046bdaa7_n.jpg
  inflating: flowers/Testing/sunflower/5139971615_434ff8ed8b_n.jpg
  inflating: flowers/Testing/sunflower/5139977283_530c508603_n.jpg
  inflating: flowers/Testing/sunflower/5139977423_d413b23fde_m.jpg
  inflating: flowers/Testing/sunflower/5139977579_ea2dd6a322_m.jpg
  inflating: flowers/Testing/sunflower/5180260869_1db7ff98e4_n.jpg
  inflating: flowers/Testing/sunflower/5180859236_60aa57ff9b_n.jpg
  inflating: flowers/Testing/sunflower/5180861654_0741222c62_n.jpg
  inflating: flowers/Testing/sunflower/5223643767_d8beb7e410.jpg
  inflating: flowers/Testing/sunflower/5231868667_f0baa71feb_n.jpg
  inflating: flowers/Testing/sunflower/5293283002_9b17f085f7_m.jpg
  inflating: flowers/Testing/sunflower/5330608174_b49f7a4c48_m.jpg
  inflating: flowers/Testing/sunflower/5339004958_a0a6f385fd_m.jpg
  inflating: flowers/Testing/sunflower/5357144886_b78f4782eb.jpg
  inflating: flowers/Testing/sunflower/5437996076_cf7e2ac32e_n.jpg
  inflating: flowers/Testing/sunflower/5492906452_80943bfd04.jpg
  inflating: flowers/Testing/sunflower/5526324308_b333da0e57_n.jpg
  inflating: flowers/Testing/sunflower/5556633113_0a04f5ed8a_n.jpg
  inflating: flowers/Testing/sunflower/5738580862_e128192f75.jpg
  inflating: flowers/Testing/sunflower/5830614551_e460a1215c.jpg
  inflating: flowers/Testing/sunflower/5896354497_6a19162741.jpg
  inflating: flowers/Testing/sunflower/5917253022_4e3142d48b_n.jpg
  inflating: flowers/Testing/sunflower/5923085671_f81dd1cf6f.jpg
  inflating: flowers/Testing/sunflower/5923085891_27617463fe.jpg
  inflating: flowers/Testing/sunflower/5923649444_a823e534e9.jpg
  inflating: flowers/Testing/sunflower/5927432662_3ffd2461c2_n.jpg
  inflating: flowers/Testing/sunflower/5933438337_b26a81ea81_n.jpg
  inflating: flowers/Testing/sunflower/5933438461_7607cf06e2_n.jpg
  inflating: flowers/Testing/sunflower/5933438547_0dea1fddd6_n.jpg
  inflating: flowers/Testing/sunflower/5937355165_1dc7b2cbf9_n.jpg
  inflating: flowers/Testing/sunflower/5937914300_bfca430439_n.jpg
```

```
inflating: flowers/Testing/sunflower/5951665793_8ae4807cbd_n.jpg
inflating: flowers/Testing/sunflower/5952223760_85972671d6_n.jpg
```

```
# 2) IMAGE AUGMENTATION
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale=1./255, zoom_range=0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)

xtrain = train_datagen.flow_from_directory('/content/flowers/Training', target_size=(64,64

xtest = test_datagen.flow_from_directory('/content/flowers/Testing', target_size=(64,64),
```

```
Found 4317 images belonging to 5 classes.
Found 750 images belonging to 5 classes.
```

```
# 3) CREATE MODEL
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense

# Initializing the model
model = Sequential()



# 4) ADD LAYERS

# There are 4 layers - Covolution layer, Max pooling layer, Flatten layer, Hidden layer, O


# Covolution layer
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))


# Max pooling layer
model.add(MaxPooling2D(pool_size=(2,2)))


# Flatten layer
model.add(Flatten())


# Hidden layer
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))


# Output layer
model.add(Dense(5,activation='softmax'))


# 5) COMPILE THE MODEL

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


# 6) FIT THE MODEL
```

```
# 6) FIT THE MODEL

model.fit_generator(xtrain,
                    steps_per_epoch=len(xtrain),
                    epochs=10,
                    validation_data=xtest,
                    validation_steps=len(xtest))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: UserWarning: `Model.f
  import sys
Epoch 1/10
44/44 [==============================] - 29s 634ms/step - loss: 1.7077 - accuracy: 0
Epoch 2/10
44/44 [==============================] - 28s 636ms/step - loss: 1.0900 - accuracy: 0
Epoch 3/10
44/44 [==============================] - 28s 623ms/step - loss: 1.0195 - accuracy: 0
Epoch 4/10
44/44 [==============================] - 28s 626ms/step - loss: 0.9325 - accuracy: 0
Epoch 5/10
44/44 [==============================] - 28s 632ms/step - loss: 0.8705 - accuracy: 0
Epoch 6/10
44/44 [==============================] - 28s 640ms/step - loss: 0.8384 - accuracy: 0
Epoch 7/10
44/44 [==============================] - 28s 628ms/step - loss: 0.7914 - accuracy: 0
Epoch 8/10
44/44 [==============================] - 28s 627ms/step - loss: 0.7457 - accuracy: 0
Epoch 9/10
44/44 [==============================] - 28s 632ms/step - loss: 0.7190 - accuracy: 0
Epoch 10/10
44/44 [==============================] - 29s 657ms/step - loss: 0.6893 - accuracy: 0
<keras.callbacks.History at 0x7fd6651654d0>
```

```
# 7) SAVE THE MODEL
model.save('IbmFlowers.h5')
```

```
# 8) TEST THE MODEL
import numpy as np
from tensorflow.keras.preprocessing import image
op = ['daisy', 'dandelion','rose', 'sunflower', 'tulip']


img = image.load_img('/content/flowers/Testing/daisy/11642632_1e7627a2cc.jpg',target_size=
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
print(pred, model.predict(x))
print(op[pred])

# wrong prediction
```

```
1 [[0. 1. 0. 0. 0.]]
dandelion
```

```
img = image.load_img('/content/flowers/Testing/rose/15498482197_8878cdfb07_n.jpg',target_s
```

```
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
print(pred, model.predict(x))
print(op[pred])
```

```
2 [[0. 0. 1. 0. 0.]]
rose
```

```
img = image.load_img('/content/flowers/Testing/tulip/14084211971_0f921f11fe_n.jpg',target_
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
print(pred, model.predict(x))
print(op[pred])

# wrong prediction
```

```
2 [[0.0000000e+00 0.0000000e+00 1.0000000e+00 0.0000000e+00 1.6980128e-12]]
rose
```

```
img = image.load_img('/content/flowers/Testing/sunflower/5180859236_60aa57ff9b_n.jpg',targ
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
print(pred, model.predict(x))
print(op[pred])
```

```
3 [[0. 0. 0. 1. 0.]]
sunflower
```

```
# TUNING THE MODEL FOR MORE ACCURATE PREDICTION

from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau

early_stop = EarlyStopping(monitor='val_accuracy',
                           patience=5)

lr = ReduceLROnPlateau(monitor='val_accuracy',
                       factor=0.5,
                       patience=5,
                       min_lr=0.00001)

callbacks = [early_stop,lr]

model.fit_generator(xtrain,
                    steps_per_epoch=len(xtrain),
                    epochs=100,
                    callbacks=callbacks,
                    validation_data=xtest,
                    validation_steps=len(xtest),)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:20: UserWarning: `Model
Epoch 1/100
```

```
44/44 [==============================] - 28s 625ms/step - loss: 0.6726 - accuracy: 0
Epoch 2/100
44/44 [==============================] - 27s 616ms/step - loss: 0.6292 - accuracy: 0
Epoch 3/100
44/44 [==============================] - 28s 623ms/step - loss: 0.5962 - accuracy: 0
Epoch 4/100
44/44 [==============================] - 28s 623ms/step - loss: 0.5785 - accuracy: 0
Epoch 5/100
44/44 [==============================] - 28s 640ms/step - loss: 0.5489 - accuracy: 0
Epoch 6/100
44/44 [==============================] - 29s 655ms/step - loss: 0.5140 - accuracy: 0
Epoch 7/100
44/44 [==============================] - 28s 643ms/step - loss: 0.4971 - accuracy: 0
Epoch 8/100
44/44 [==============================] - 29s 651ms/step - loss: 0.5052 - accuracy: 0
Epoch 9/100
44/44 [==============================] - 28s 643ms/step - loss: 0.4848 - accuracy: 0
Epoch 10/100
44/44 [==============================] - 28s 634ms/step - loss: 0.4612 - accuracy: 0
Epoch 11/100
44/44 [==============================] - 28s 637ms/step - loss: 0.4254 - accuracy: 0
Epoch 12/100
44/44 [==============================] - 28s 637ms/step - loss: 0.4008 - accuracy: 0
Epoch 13/100
44/44 [==============================] - 28s 635ms/step - loss: 0.3699 - accuracy: 0
Epoch 14/100
44/44 [==============================] - 28s 638ms/step - loss: 0.3492 - accuracy: 0
<keras.callbacks.History at 0x7fd665137cd0>
```

```
img = image.load_img('/content/flowers/Testing/tulip/14084211971_0f921f11fe_n.jpg',target_
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
print(pred, model.predict(x))
print(op[pred])

# correct prediction after tuning the model
```

```
4 [[0. 0. 0. 0. 1.]]
tulip
```

```
img = image.load_img('/content/flowers/Testing/daisy/11642632_1e7627a2cc.jpg',target_size=
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))
print(pred, model.predict(x))
print(op[pred])

# after tuning the model - correct prediction
```

```
0 [[1. 0. 0. 0. 0.]]
daisy
```