Assignment 3

1. Download And unzip dataset

```
# run this to download the dataset directly to the kernal

!gdown 1xkynpL15pt6KT3YSlDimu4A5iRU9qYck
```

```
    Downloading...
    From: https://drive.google.com/uc?id=1xkynpL15pt6KT3YSlDimu4A5iRU9qYck
    To: /content/Flowers-Dataset.zip
    100% 236M/236M [00:00<00:00, 286MB/s]
```

```
# Unzip

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

```
  inflating: flowers/daisy/13826249325_f61cb15f86_n.jpg
  inflating: flowers/daisy/13901930939_a7733c03f0_n.jpg
  inflating: flowers/daisy/1392131677_116ec04751.jpg
  inflating: flowers/daisy/1392946544_115acbb2d9.jpg
  inflating: flowers/daisy/13953307149_f8de6a768c_m.jpg
  inflating: flowers/daisy/1396526833_fb867165be_n.jpg
  inflating: flowers/daisy/13977181862_f8237b6b52.jpg
  inflating: flowers/daisy/14021430525_e06baf93a9.jpg
  inflating: flowers/daisy/14073784469_ffb12f3387_n.jpg
  inflating: flowers/daisy/14087947408_9779257411_n.jpg
  inflating: flowers/daisy/14088053307_1a13a0bf91_n.jpg
  inflating: flowers/daisy/14114116486_0bb6649bc1_m.jpg
  inflating: flowers/daisy/14147016029_8d3cf2414e.jpg
  inflating: flowers/daisy/14163875973_467224aaf5_m.jpg
  inflating: flowers/daisy/14167534527_781ceb1b7a_n.jpg
  inflating: flowers/daisy/14167543177_cd36b54ac6_n.jpg
  inflating: flowers/daisy/14219214466_3ca6104eae_m.jpg
  inflating: flowers/daisy/14221836990_90374e6b34.jpg
  inflating: flowers/daisy/14221848160_7f0a37c395.jpg
  inflating: flowers/daisy/14245834619_153624f836.jpg
  inflating: flowers/daisy/14264136211_9531fbc144.jpg
  inflating: flowers/daisy/14272874304_47c0a46f5a.jpg
  inflating: flowers/daisy/14307766919_fac3c37a6b_m.jpg
  inflating: flowers/daisy/14330343061_99478302d4_m.jpg
  inflating: flowers/daisy/14332947164_9b13513c71_m.jpg
  inflating: flowers/daisy/14333681205_a07c9f1752_m.jpg
  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
```

Importing Necessary Libs

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense
from tensorflow.keras.preprocessing import image
import numpy as np
import matplotlib.pyplot as plt
```

2. Data Augmnetaion

```
# For training

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

```
# for testing

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

```
# To split the dataset into Train and test

!pip install split_folders
```

```
import splitfolders

input_folder = "/content/flowers"
output = "/content/Dataset"

splitfolders.ratio(input_folder, output=output, seed=42, ratio=(0.7,0.3))
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
Requirement already satisfied: split_folders in /usr/local/lib/python3.7/dist
Copying files: 4317 files [00:01, 3937.44 files/s]
```

```
# data generation

xtrain = train_datagen.flow_from_directory('/content/Dataset/train',
                                           target_size=(64,64),
                                           class_mode='categorical',
                                           batch_size=100)




xtest = test_datagen.flow_from_directory('/content/Dataset/val',
                                         target_size=(64,64),
                                         class_mode='categorical',
                                         batch_size=100)
```

```
Found 3019 images belonging to 5 classes.
Found 1298 images belonging to 5 classes.
```

3. Build Model

Adding layers

```
# Build a CNN block

model = Sequential() # Initializing sequential model
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3))) # convo
model.add(MaxPooling2D(pool_size=(2, 2))) # Max pooling layer
model.add(Flatten()) # Flatten layer
model.add(Dense(300,activation='relu')) # Hidden layer 1
model.add(Dense(150,activation='relu')) # Hidden layer 2
model.add(Dense(5,activation='softmax')) # Output layer
```

Compiling Model

```
# Compiling the model
```

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

Fit Model

```
# Train model

model.fit(xtrain,
          steps_per_epoch=len(xtrain),
          epochs=50,
          validation_data=xtest,
          validation_steps=len(xtest))
```

```
    Epoch 1/50
    31/31 [==============================] - 23s 394ms/step - loss: 2.2765 - accu
    Epoch 2/50
    31/31 [==============================] - 12s 393ms/step - loss: 1.2581 - accu
    Epoch 3/50
    31/31 [==============================] - 16s 508ms/step - loss: 1.1536 - accu
    Epoch 4/50
    31/31 [==============================] - 12s 393ms/step - loss: 1.0853 - accu
    Epoch 5/50
    31/31 [==============================] - 13s 414ms/step - loss: 1.0715 - accu
    Epoch 6/50
    31/31 [==============================] - 12s 389ms/step - loss: 1.0135 - accu
    Epoch 7/50
    31/31 [==============================] - 13s 415ms/step - loss: 0.9690 - accu
    Epoch 8/50
    31/31 [==============================] - 14s 456ms/step - loss: 0.9308 - accu
    Epoch 9/50
    31/31 [==============================] - 14s 446ms/step - loss: 0.8920 - accu
    Epoch 10/50
    31/31 [==============================] - 14s 451ms/step - loss: 0.8516 - accu
    Epoch 11/50
    31/31 [==============================] - 13s 432ms/step - loss: 0.8382 - accu
    Epoch 12/50
    31/31 [==============================] - 13s 418ms/step - loss: 0.8205 - accu
    Epoch 13/50
    31/31 [==============================] - 13s 431ms/step - loss: 0.7762 - accu
    Epoch 14/50
    31/31 [==============================] - 12s 392ms/step - loss: 0.7396 - accu
    Epoch 15/50
    31/31 [==============================] - 12s 392ms/step - loss: 0.7333 - accu
    Epoch 16/50
    31/31 [==============================] - 12s 412ms/step - loss: 0.7193 - accu
    Epoch 17/50
    31/31 [==============================] - 14s 447ms/step - loss: 0.6993 - accu
    Epoch 18/50
    31/31 [==============================] - 13s 424ms/step - loss: 0.6708 - accu
    Epoch 19/50
    31/31 [==============================] - 13s 407ms/step - loss: 0.6495 - accu
    Epoch 20/50
    31/31 [==============================] - 13s 405ms/step - loss: 0.6217 - accu
    Epoch 21/50
    31/31 [==============================] - 13s 411ms/step - loss: 0.5822 - accu
    Epoch 22/50
    31/31 [==============================] - 12s 391ms/step - loss: 0.6147 - accu
    Epoch 23/50
    31/31 [==============================] - 12s 391ms/step - loss: 0.5687 - accu
    Epoch 24/50
    31/31 [==============================] - 12s 389ms/step - loss: 0.5583 - accu
    Epoch 25/50
    31/31 [==============================] - 12s 407ms/step - loss: 0.5564 - accu
    Epoch 26/50
    31/31 [==============================] - 12s 391ms/step - loss: 0.4828 - accu
    Epoch 27/50
    31/31 [==============================] - 12s 389ms/step - loss: 0.4835 - accu
    Epoch 28/50
    31/31 [==============================] - 13s 408ms/step - loss: 0.4777 - accu
    Epoch 29/50
    31/31 [==============================] - 12s 401ms/step - loss: 0.4616 - accu
```

4. Save Model

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

5. Testing The Model

```
def predict_flower(img_path):
  img = image.load_img(img_path,target_size=(64,64)) # Reading image
  x = image.img_to_array(img) # Converting image into array
  x = np.expand_dims(x,axis=0) # expanding Dimensions
  pred = np.argmax(model.predict(x)) # Predicting the higher probablity index
  op = ['Daisy','Dandelion','Rose','SunFlower','Tulip'] # Creating list
  print(op[pred]) # List indexing with output
  plt.imshow(img) # Printing the image
```

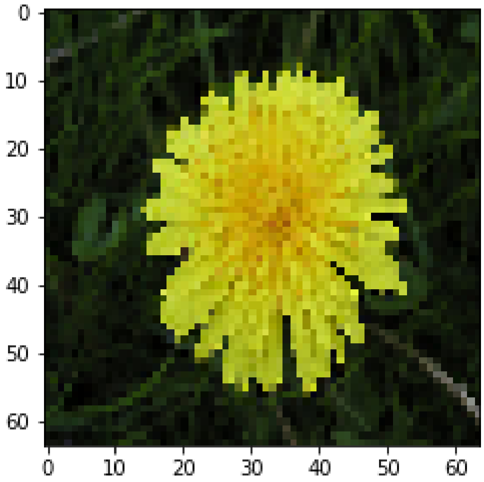
With Test Data Images

```
# Testing 1
# Daisy flower Image

predict_flower('/content/Dataset/val/daisy/1150395827_6f94a5c6e4_n.jpg') # Predict
```

```
Daisy
```

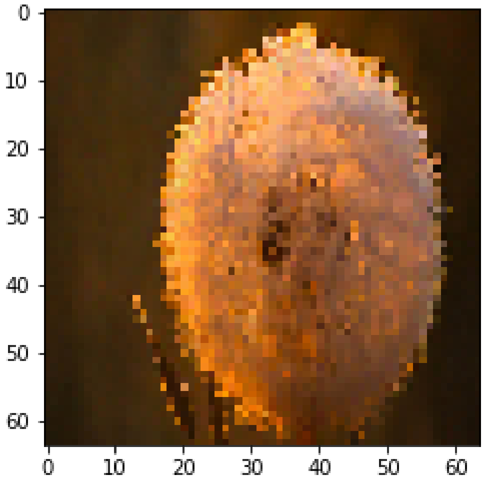
```
# Testing 2
# Dandelion flower Image

predict_flower('/content/Dataset/val/dandelion/1128626197_3f52424215_n.jpg')
```

```
Dandelion
```



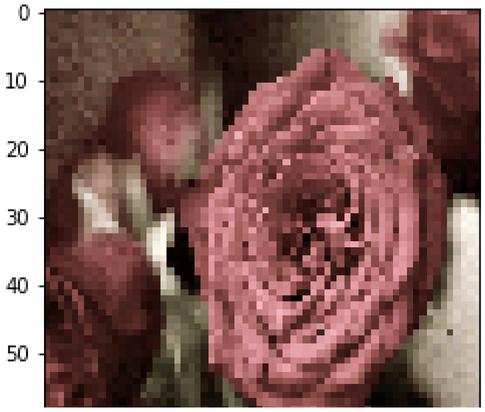
```
# Testing 3
# Dandelion flower Image

predict_flower('/content/Dataset/val/dandelion/14199664556_188b37e51e.jpg')
```

```
Rose
```



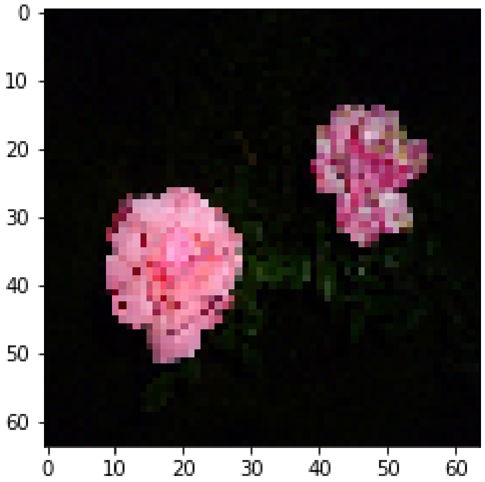
```
# Testing 4
# Rose flower Image

predict_flower('/content/Dataset/val/rose/12202373204_34fb07205b.jpg')
```

```
Rose
```



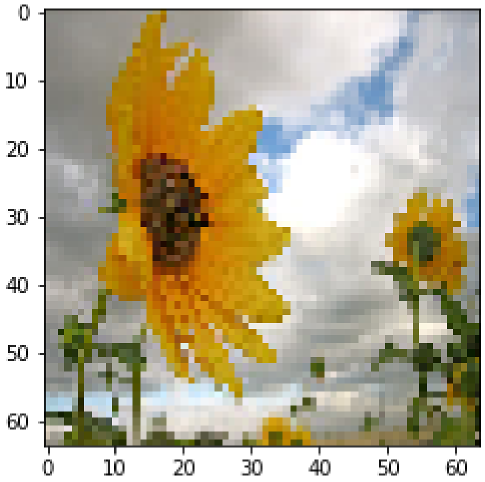
```
# Testing 5
# Rose flower Image

predict_flower('/content/Dataset/val/rose/15820572326_be2ea4a55c_n.jpg')
```

```
Rose
```



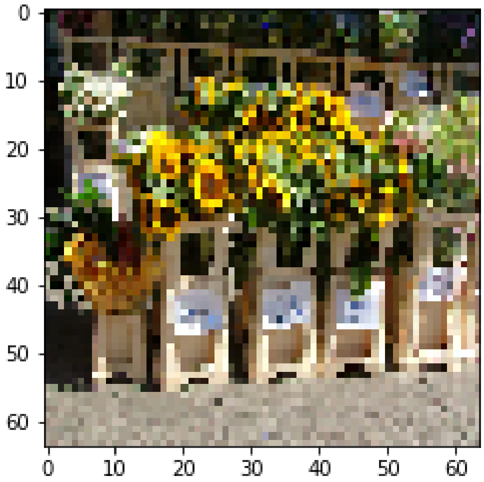
```
# Testing 6
# Sunflower Image

predict_flower('/content/Dataset/val/sunflower/1596293240_2d5b53495a_m.jpg')
```

```
SunFlower
```



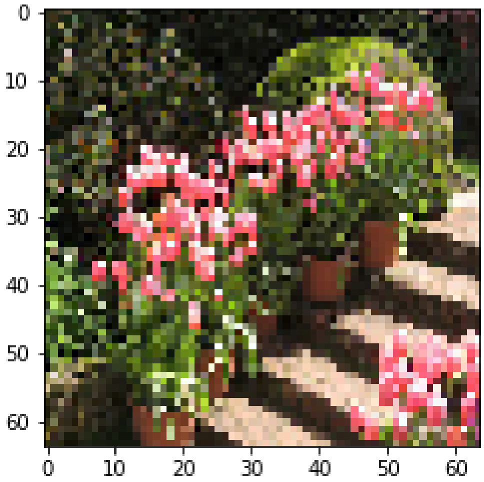
```
# Testing 7
# Sunflower Image

predict_flower('/content/Dataset/val/sunflower/210076535_80951bc5d5.jpg')
```

```
SunFlower
```



```
# Testing 8
# Tulip Flower Image

predict_flower('/content/Dataset/val/tulip/13530690445_9f1f5cf43a_n.jpg')
```

```
Rose
```

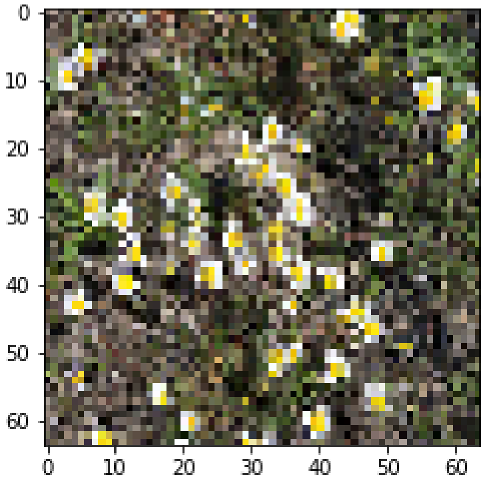


```
# Testing 9
# Tulip Flower Image

predict_flower('/content/Dataset/val/tulip/16680927427_07ca6e4552_n.jpg')
```

```
Tulip
```

```
# Testing 10
# Daisy Flower Image

predict_flower('/content/Dataset/val/daisy/34542837641_10492bf600_n.jpg')
```

```
Daisy
```



With Google Images

```
# Run To download test images

!gdown 1Q-QTRIfXjV0BbLcIvopbiYfbAD3hJfmw
```

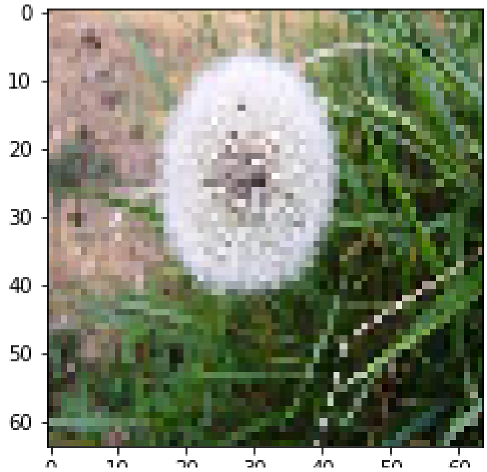
```
Downloading...
From: https://drive.google.com/uc?id=1Q-QTRIfXjV0BbLcIvopbiYfbAD3hJfmw
To: /content/IBM Flower_Test dataset.zip
100% 1.01M/1.01M [00:00<00:00, 163MB/s]
```

```
# unzip

!unzip '/content/IBM Flower_Test dataset.zip'
```

```
Archive:  /content/IBM Flower_Test dataset.zip
replace IBM Flower_Test dataset/tulip_2.jpg? [y]es, [n]o, [A]ll, [N]one, [r]e
```

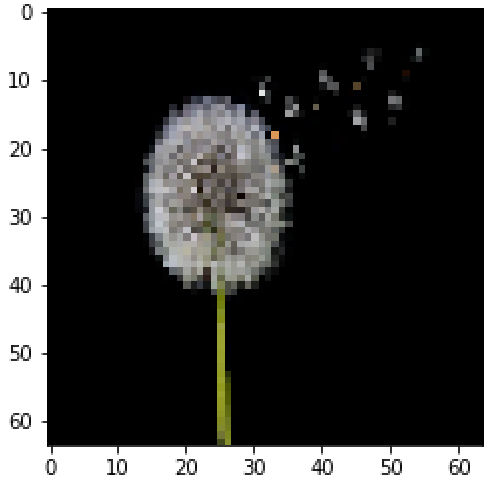
```
# Test 1
# Dandelion Flower

predict_flower('/content/IBM Flower_Test dataset/Dandelion.jpeg')
```

```
Tulip
```



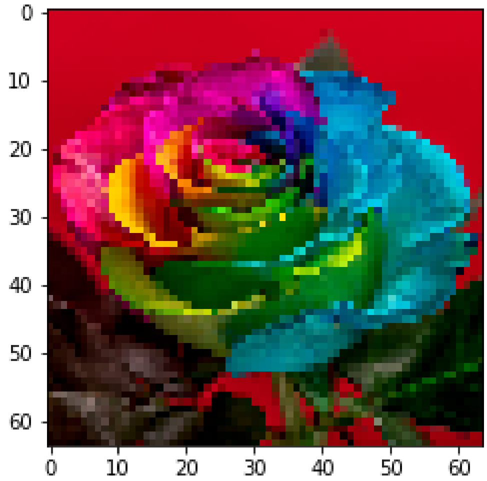
```
# Test 2
# Dandelion Flower

predict_flower('/content/IBM Flower_Test dataset/Dandelion_2.jpeg')
```

```
Daisy
```



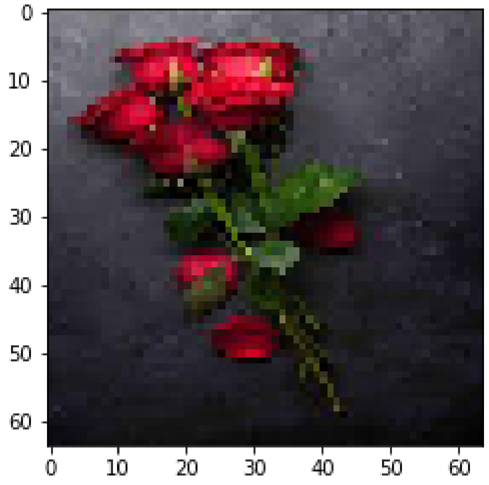
```
# Test 3
# Rose Flower

predict_flower('/content/IBM Flower_Test dataset/Rose.jpeg')
```

```
Tulip
```



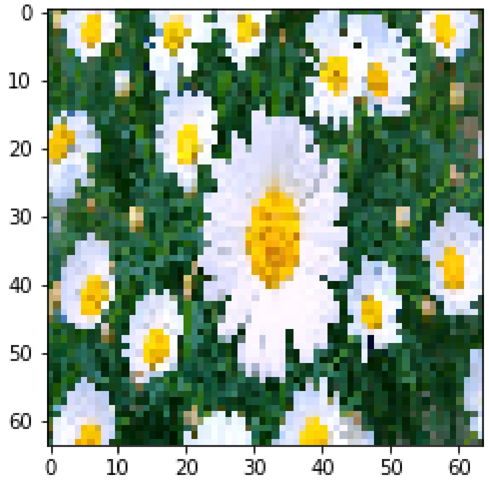
```
# Test 4
# Rose Flower

predict_flower('/content/IBM Flower_Test dataset/Rose_2.jpeg')
```

```
Rose
```

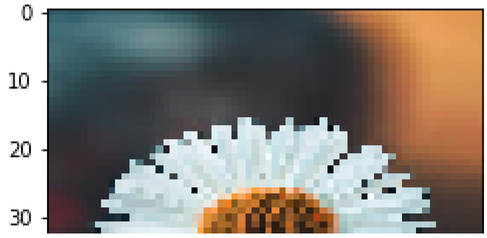


```
# Test 5
# Daisy Flower

predict_flower('/content/IBM Flower_Test dataset/daisy-flower-1532449822.jpg')
```

```
Daisy
```

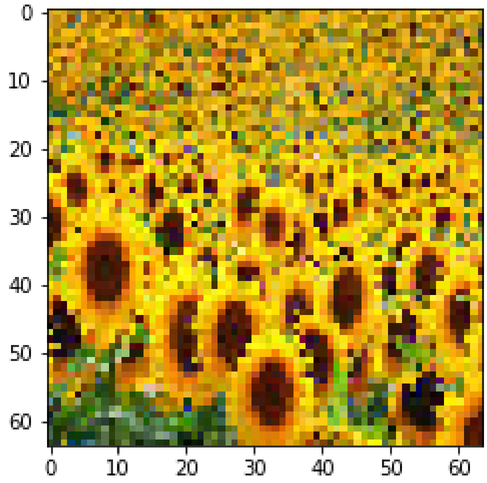


```
# Test 6
# Daisy Flower

predict_flower('/content/IBM Flower_Test dataset/photo-1606041008023-472dfb5e530f.
Rose
```



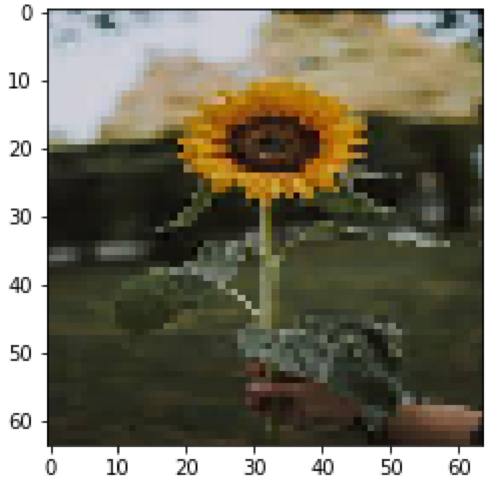
```
# Test 7
# Sun Flower

predict_flower('/content/IBM Flower_Test dataset/sunflower.jpeg')
```

```
SunFlower
```



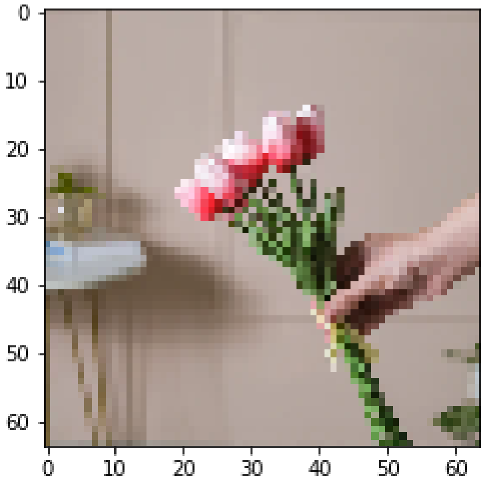
```
# Test 8
# Sun Flower

predict_flower('/content/IBM Flower_Test dataset/sunflower_2.jpeg')
```

```
SunFlower
```



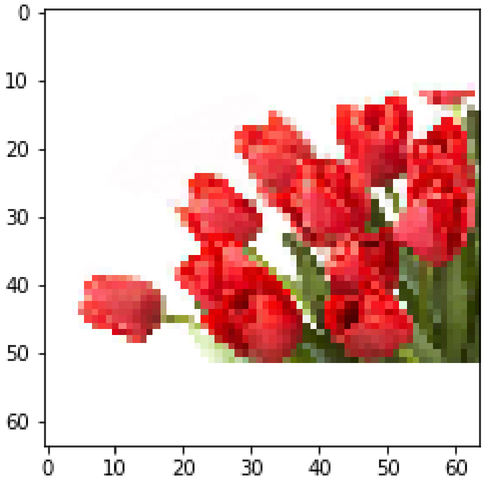
```
# Test 9
# Tulip Flower

predict_flower('/content/IBM Flower_Test dataset/tulip.webp')
```

```
Tulip
```



```
# Test 10
# Tulip Flower

predict_flower('/content/IBM Flower_Test dataset/tulip_2.jpg')
```

```
Tulip
```



Colab paid products - Cancel contracts here