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

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True)
```

```
!unzip "/content/drive/MyDrive/Flowers-Dataset.zip"
```

```
Archive:  /content/drive/MyDrive/Flowers-Dataset.zip
replace flowers/daisy/100080576_f52e8ee070_n.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
  inflating: flowers/daisy/100080576_f52e8ee070_n.jpg
replace flowers/daisy/10140303196_b88d3d6cec.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename:
```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

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

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

```
x_train=train_datagen.flow_from_directory(r"/content/flowers",target_size=(64,64),class_mode='categorical',batch_size=24)
```

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

```
x_test=test_datagen.flow_from_directory(r"/content/flowers",target_size=(64,64),class_mode='categorical',batch_size=24)
```

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

```
x_train.class_indices
```

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

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Convolution2D,MaxPooling2D,Flatten

model=Sequential()


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


model.add(MaxPooling2D(pool_size=(2,2)))


model.add(Flatten())


model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 flatten (Flatten)           (None, 30752)             0

=================================================================
Total params: 896
Trainable params: 896
Non-trainable params: 0
_________________________________________________________________
```

```
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))


model.add(Dense(5,activation='softmax'))
```

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

```
len(x_train)
```

```
180
```

```
model.fit_generator(x_train,steps_per_epoch=len(x_train), validation_data=x_test, validation_steps=len(x_test), epochs= 30)
```

```
180/180 [==============================] - 26s 143ms/step - loss: 1.0267 - accuracy: 0.5833 - val_loss: 0.9598 - val_accuracy
Epoch 3/30
180/180 [==============================] - 24s 136ms/step - loss: 0.9379 - accuracy: 0.6423 - val_loss: 0.9262 - val_accuracy
Epoch 4/30
180/180 [==============================] - 24s 136ms/step - loss: 0.8576 - accuracy: 0.6725 - val_loss: 0.8078 - val_accuracy
Epoch 5/30
180/180 [==============================] - 24s 135ms/step - loss: 0.8050 - accuracy: 0.6921 - val_loss: 0.8061 - val_accuracy
Epoch 6/30
180/180 [==============================] - 24s 135ms/step - loss: 0.7563 - accuracy: 0.7114 - val_loss: 0.6680 - val_accuracy
Epoch 7/30
180/180 [==============================] - 24s 136ms/step - loss: 0.7305 - accuracy: 0.7216 - val_loss: 0.6000 - val_accuracy
Epoch 8/30
180/180 [==============================] - 24s 135ms/step - loss: 0.6909 - accuracy: 0.7334 - val_loss: 0.5393 - val_accuracy
Epoch 9/30
180/180 [==============================] - 24s 135ms/step - loss: 0.6470 - accuracy: 0.7589 - val_loss: 0.5462 - val_accuracy
Epoch 10/30
180/180 [==============================] - 26s 144ms/step - loss: 0.6093 - accuracy: 0.7739 - val_loss: 0.5949 - val_accuracy
Epoch 11/30
180/180 [==============================] - 25s 138ms/step - loss: 0.5642 - accuracy: 0.7899 - val_loss: 0.5021 - val_accuracy
Epoch 12/30
180/180 [==============================] - 24s 135ms/step - loss: 0.5053 - accuracy: 0.8119 - val_loss: 0.6559 - val_accuracy
Epoch 13/30
180/180 [==============================] - 24s 135ms/step - loss: 0.4798 - accuracy: 0.8228 - val_loss: 0.4023 - val_accuracy
Epoch 14/30
180/180 [==============================] - 24s 135ms/step - loss: 0.4667 - accuracy: 0.8249 - val_loss: 0.3324 - val_accuracy
Epoch 15/30
180/180 [==============================] - 25s 136ms/step - loss: 0.4226 - accuracy: 0.8513 - val_loss: 0.2972 - val_accuracy
Epoch 16/30
180/180 [==============================] - 25s 139ms/step - loss: 0.4039 - accuracy: 0.8494 - val_loss: 0.3337 - val_accuracy
Epoch 17/30
```

```
180/180 [==============================] - 25s 139ms/step - loss: 0.3625 - accuracy: 0.8689 - val_loss: 0.2777 - val_accuracy
Epoch 18/30
180/180 [==============================] - 24s 134ms/step - loss: 0.3315 - accuracy: 0.8805 - val_loss: 0.3107 - val_accuracy
Epoch 19/30
180/180 [==============================] - 24s 134ms/step - loss: 0.3316 - accuracy: 0.8821 - val_loss: 0.2876 - val_accuracy
Epoch 20/30
180/180 [==============================] - 24s 133ms/step - loss: 0.2999 - accuracy: 0.8953 - val_loss: 0.1872 - val_accuracy
Epoch 21/30
180/180 [==============================] - 24s 133ms/step - loss: 0.2825 - accuracy: 0.9020 - val_loss: 0.2619 - val_accuracy
Epoch 22/30
180/180 [==============================] - 24s 134ms/step - loss: 0.2588 - accuracy: 0.9073 - val_loss: 0.2719 - val_accuracy
Epoch 23/30
180/180 [==============================] - 24s 134ms/step - loss: 0.2611 - accuracy: 0.9094 - val_loss: 0.1305 - val_accuracy
Epoch 24/30
180/180 [==============================] - 26s 142ms/step - loss: 0.2490 - accuracy: 0.9117 - val_loss: 0.2254 - val_accuracy
Epoch 25/30
180/180 [==============================] - 24s 135ms/step - loss: 0.2324 - accuracy: 0.9187 - val_loss: 0.1225 - val_accuracy
Epoch 26/30
180/180 [==============================] - 24s 134ms/step - loss: 0.2169 - accuracy: 0.9249 - val_loss: 0.1343 - val_accuracy
Epoch 27/30
180/180 [==============================] - 24s 134ms/step - loss: 0.2130 - accuracy: 0.9252 - val_loss: 0.0891 - val_accuracy
Epoch 28/30
180/180 [==============================] - 24s 135ms/step - loss: 0.1814 - accuracy: 0.9384 - val_loss: 0.1247 - val_accuracy
Epoch 29/30
180/180 [==============================] - 24s 134ms/step - loss: 0.1709 - accuracy: 0.9416 - val_loss: 0.1662 - val_accuracy
Epoch 30/30
180/180 [==============================] - 24s 135ms/step - loss: 0.1792 - accuracy: 0.9333 - val_loss: 0.1622 - val_accuracy
<keras.callbacks.History at 0x7f005d77b710>
```

```
model.save('Flowers_classification_model1.h5')
```

```
ls
```

```
drive/  flowers/  Flowers_classification_model1.h5  sample_data/
```

```
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

```
# Load the model
model=load_model('Flowers_classification_model1.h5')


img=image.load_img(r"/content/flowers/rose/10503217854_e66a804309.jpg",target_size=(64,64))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
y=np.argmax(model.predict(x),axis=1)
# x_train.class_indices
index=['daisy','dandelion','rose','sunflower','tulip']
index[y[0]]
```

```
1/1 [==============================] - 0s 124ms/step
'tulip'
```

Colab paid products - Cancel contracts here

✓ 0s completed at 2:51 AM