# Assignment-3

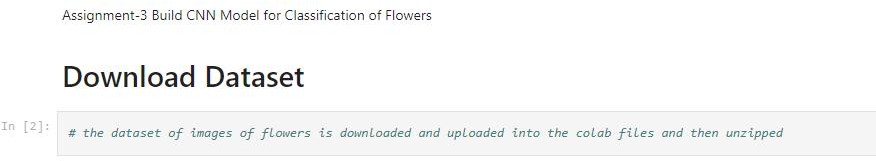
**Build CNN Model for Classification Of Flowers**

|  |  |  |
| --- | --- | --- |
| Assignment Date | : | 15 October 2022 |
| Student Name | : | Kiruthiga M |
| Student Roll Number | : | 212219040115 |
| Maximum Marks | : | 2 Marks |

**Task 1:**

**Question- 1:**

Download the dataset



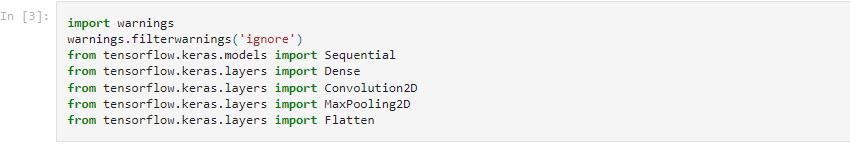
Solution:

import warnings warnings.filterwarnings('ignore')

from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense

from tensorflow.keras.layers import Convolution2D from tensorflow.keras.layers import MaxPooling2D from tensorflow.keras.layers import Flatten

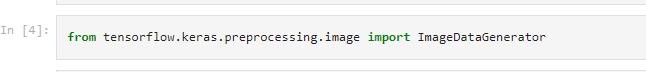
Output:



Solution:

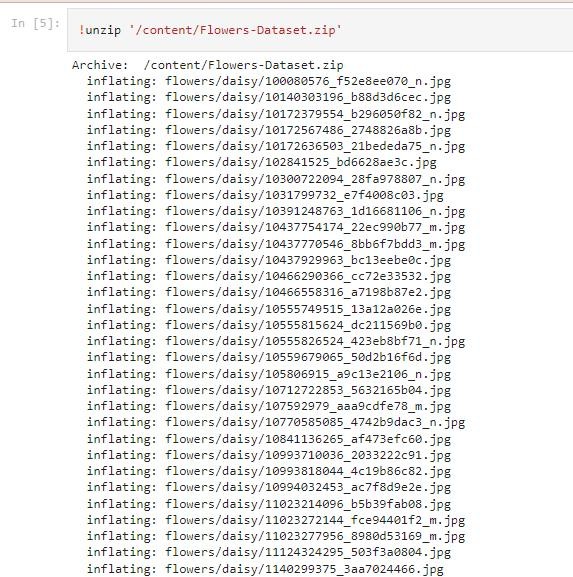
from tensorflow.keras.preprocessing.image import ImageDataGenerator

Output:



Solution:

!unzip '/content/Flowers-Dataset.zip' Output:



Task 2:

**Question- 2:**

Image Augmentation

Solution:

#Image Augmentation on training varaible train\_datagen = ImageDataGenerator(rescale=1./255,

zoom\_range=0.2, horizontal\_flip=True)

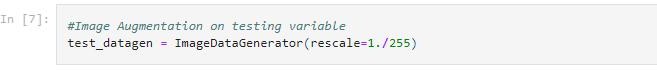
Output:



Solution:

#Image Augmentation on testing variable test\_datagen = ImageDataGenerator(rescale=1./255)

Output:

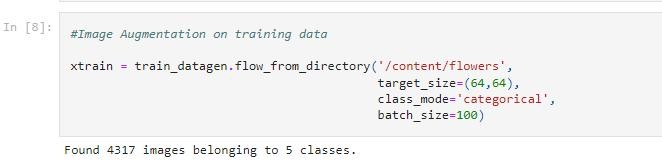


Solution:

### #Image Augmentation on training data

Xtrain=train\_datagen.flow\_from\_directory('/content/ flowers',target\_size=(64,64),class\_mode='cate gorical',batch\_size=100)

Output:

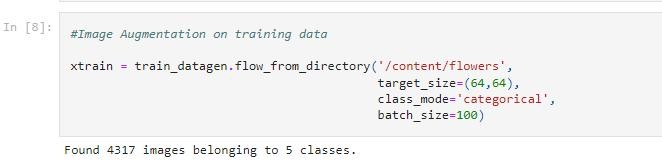


Solution:

### #Image Augmentation on training data

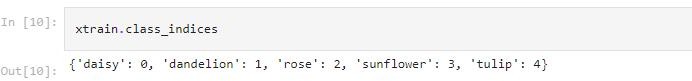
Xtrain=train\_datagen.flow\_from\_directory('/content/ flowers',target\_size=(64,64),class\_mode='cate gorical',batch\_size=100)

Output:



Solution:

xtrain.class\_indices Output:



Task 3:

**Question- 3:**

Create Model

Solution:

### # Initializing sequential model

model = Sequential()

Output:



# Task 4:

**Question- 4:**

Add Layers

A .Convolution Layer

Solution:

model.add(Convolution2D(32,(3,3),activation='relu', input\_shape=(64,64,3)))

Output:



B .Max-Pooling Layer

Solution:

model.add(MaxPooling2D(pool\_size=(2, 2))) Output:

C .Flatten Layer

Solution:

model.add(Flatten()) Output:

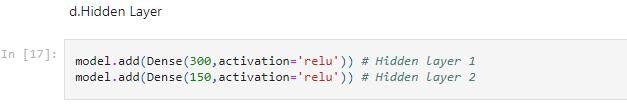


D .Hidden Layer

Solution:

model.add(Dense(300,activation='relu')) # Hidden layer 1 model.add(Dense(150,activation='relu')) # Hidden layer 2

Output:

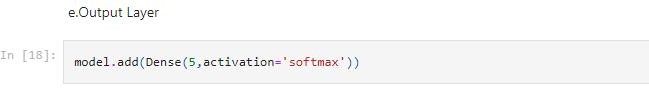


E .Output Layer

Solution:

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

Output:



## Task 5:

**Question- 5:**

Add Layers

Solution:

### from tensorflow.keras.models import Sequential

Output:



Solution:

### from tensorflow.keras.layers import Dense, Convolution2D, MaxPooling2D, Flatten

Output:



Solution:

### model = Sequential()

Output:



Solution:

### model.add(Convolution2D(32, (3,3), input\_shape=(64,64,3), activation = 'relu'))

Output:

Solution:

### model.add(MaxPooling2D(pool\_size = (2,2)))

Output:



Solution:

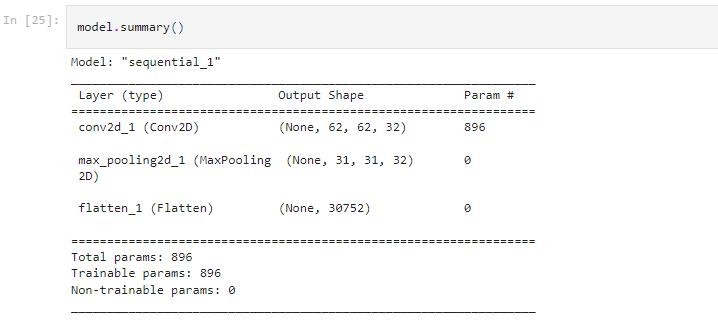
### model.add(Flatten())

Output:



Solution:

model.summary()



# Task 6:

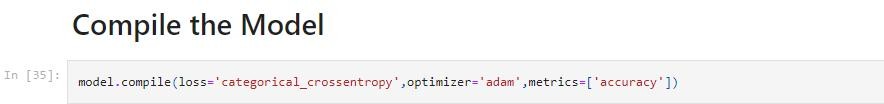
**Question- 6:**

Compile the Model

Solution:

### model.compile(loss='categorical\_crossentropy', optimizer'adam',metrics=['accuracy'])

Output:



# Task 7:

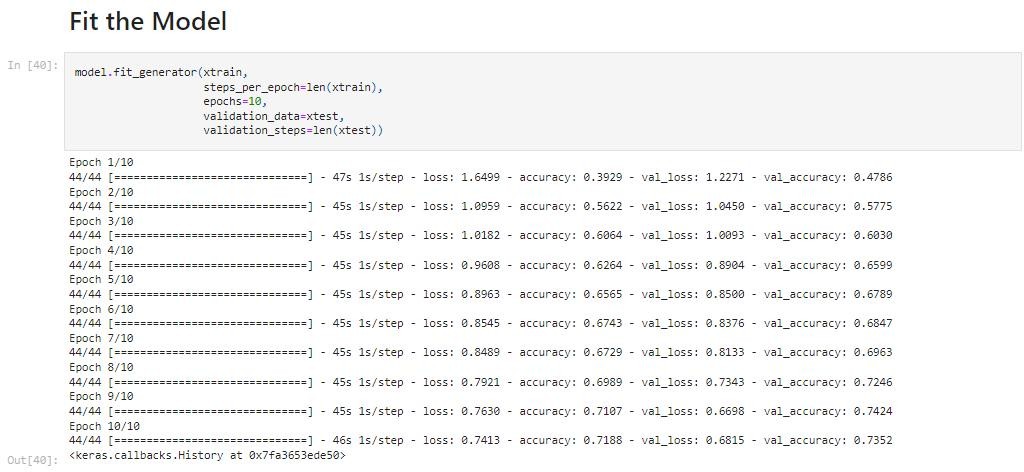
**Question- 7:**

Fit the Model

Solution:

model.fit\_generator(xtrain,steps\_per\_epoch=len(xtrain), epochs=10,validation\_data=xtest,validation\_steps=len (xtest)))

Output:



# Task 8:

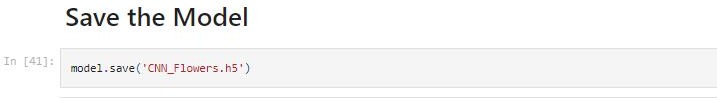
**Question- 8:**

Save the Model

Solution:

### model.save('CNN\_Flowers.h5')

Output:



Solution:

ls Output:

# Task 9:

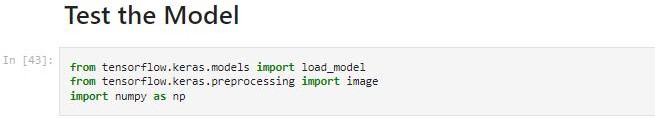
**Question- 9:**

Test the Model

Solution:

### from tensorflow.keras.models import load\_model from tensorflow.keras.preprocessing import image import numpy as np

Output:

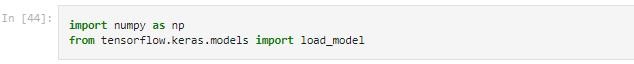


Solution:

### import numpy as np

from tensorflow.keras.models import load\_model

Output:



Solution:

from tensorflow.keras.preprocessing import image Output:



Solution:

### model=load\_model('CNN\_Flowers.h5')

Output:



Solution:

### pwd

Output:



Solution:

### img = image.load\_img('/content/flowers/rose/1233 8444334\_72fcc2fc58\_m.jpg')

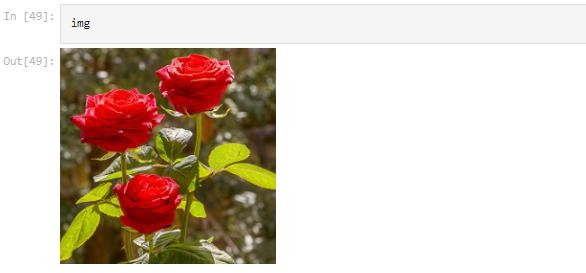
Output:



Solution:

### Img

Output:



Solution:

### img =

image.load\_img('/content/flowers/dandelion/ 14003401241\_543535b385.jpg',target\_size=(64, 64))

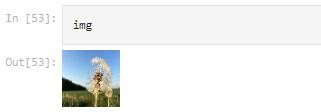
Output:



Solution:

### Img

Output:



Solution:

### x=image.img\_to\_array(img)

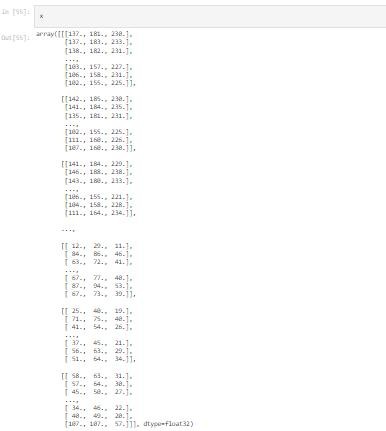
Output:



Solution:

### X

Output:



Solution:

### x.shape

Output:



Solution:

### x=np.expand\_dims(x,axis=0)

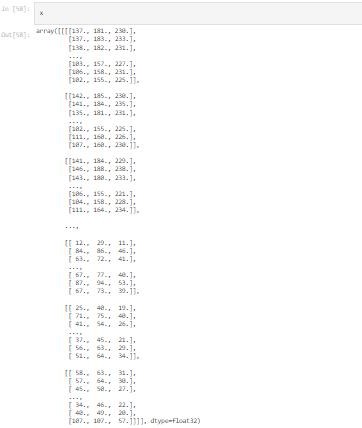
Output:



Solution:

X

Output:



Solution:

### x.shape

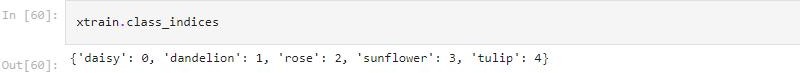
Output:



Solution:

### xtrain.class\_indices

Output:



Solution:

index=["Daisy","Dandelion","Rose","Sunflower","Tulip"] Output:



Solution:

### index[0]

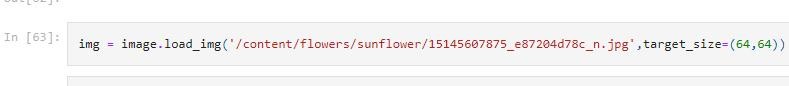
Output:



Solution:

### img = image.load\_img('/content/flowers/sunflower/151 45607875\_e87204d78c\_n.jpg',target\_size=(64,64))

Output:



Solution:

### img

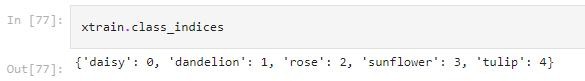
Output:



Solution:

### xtrain.class\_indices

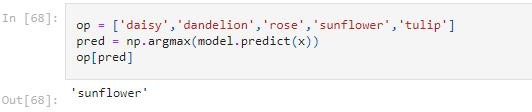
Output:



Solution:

op = ['daisy','dandelion','rose','sunflower','tulip'] pred = np.argmax(model.predict(x))

op[pred] Output:



Solution:

dandelion =

image.load\_img('/content/flowers/dandelion/1356015 2823\_9da5e48c87\_m.jpg',target\_size=(64,64))

x = image.img\_to\_array(dandelion) x = np.expand\_dims(x,axis=0)

pred = np.argmax(model.predict(x)) op[pred]

Output:

