

```
In [1]: import keras
        from keras.preprocessing.image import ImageDataGenerator
```

```
In [2]: #Define the parameters/arguments for ImageDataGenerator class
        train_datagen=ImageDataGenerator(rescale=1./255, shear_range=0.2, rotation_range=180, zoom_range=0.2,
        test_datagen=ImageDataGenerator(rescale=1./255)
```

```
In [3]: #Applying ImageDataGenerator functionality to trainset
        x_train=train_datagen.flow_from_directory(r'C:\Users\dhine\Downloads\archive\Dataset\Dataset\
                                                    target_size=(128,128),
                                                    batch_size=32,
                                                    class_mode='binary')
```

Found 436 images belonging to 2 classes.

```
In [4]: #Applying ImageDataGenerator functionality to testset
        x_test=test_datagen.flow_from_directory(r'C:\Users\dhine\Downloads\archive\Dataset\Dataset\te
                                                    target_size=(128,128),
                                                    batch_size=32,
                                                    class_mode='binary')
```

Found 121 images belonging to 2 classes.

```
In [5]: #import model building libraries

        #To define Linear initialisation import Sequential
        from keras.models import Sequential
        #To add Layers import Dense
        from keras.layers import Dense
        #To create Convolution kernel import Convolution2D
        from keras.layers import Convolution2D
        #import Maxpooling Layer
        from keras.layers import MaxPooling2D
        #import flatten Layer
        from keras.layers import Flatten
        import warnings
        warnings.filterwarnings('ignore')
```

```
In [7]: #initializing the model
        model=Sequential()
```

```
In [8]: #add convolutional layer
        model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
        #add maxpooling layer
        model.add(MaxPooling2D(pool_size=(2,2)))
        #add flatten layer
        model.add(Flatten())
```

```
In [9]: #add hidden layer
        model.add(Dense(150,activation='relu'))
        #add output layer
        model.add(Dense(1,activation='sigmoid'))
```

```
In [ ]:
```