

```
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, horizontal_flip=True)

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

```
In [6]: #Applying ImageDataGenerator functionality to trainset
        x_train=train_datagen.flow_from_directory('/content/Dataset/Dataset/train_set', target_size=(128,128), batch_size=32, class_mode='binary')
```

Found 436 images belonging to 2 classes.

```
In [7]: #Applying ImageDataGenerator functionality to testset
        x_test=test_datagen.flow_from_directory('/content/Dataset/Dataset/test_set', target_size=(128,128), batch_size=32, class_mode='binary')
```

Found 121 images belonging to 2 classes.

```
In [8]: #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 [9]: #initializing the model
        model=Sequential()
```

```
In [10]: #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 [11]: #add hidden layer
        model.add(Dense(150, activation='relu'))
        #add output layer
        model.add(Dense(1, activation='sigmoid'))
```

```
In [12]: #configure the Learning process
        model.compile(loss='binary_crossentropy', optimizer="adam", metrics=["accuracy"])
```

```
In [13]: #Training the model
        model.fit_generator(x_train, steps_per_epoch=14, epochs=10, validation_data=x_test, validation_steps=4)
```

```
Epoch 1/10
14/14 [=====] - 29s 2s/step - loss: 2.1856 - accuracy: 0.7156 - val_loss: 0.3046 - val_accuracy: 0.9256
Epoch 2/10
14/14 [=====] - 25s 2s/step - loss: 0.3005 - accuracy: 0.8899 - val_loss: 0.0900 - val_accuracy: 0.9669
Epoch 3/10
14/14 [=====] - 24s 2s/step - loss: 0.3225 - accuracy: 0.8830 - val_loss: 0.0665 - val_accuracy: 0.9752
Epoch 4/10
14/14 [=====] - 25s 2s/step - loss: 0.2286 - accuracy: 0.9083 - val_loss: 0.0653 - val_accuracy: 0.9835
Epoch 5/10
14/14 [=====] - 24s 2s/step - loss: 0.2062 - accuracy: 0.9106 - val_loss: 0.0727 - val_accuracy: 0.9752
Epoch 6/10
14/14 [=====] - 24s 2s/step - loss: 0.1593 - accuracy: 0.9335 - val_loss: 0.0804 - val_accuracy: 0.9669
Epoch 7/10
14/14 [=====] - 24s 2s/step - loss: 0.1552 - accuracy: 0.9335 - val_loss: 0.0777 - val_accuracy: 0.9669
Epoch 8/10
14/14 [=====] - 24s 2s/step - loss: 0.1445 - accuracy: 0.9335 - val_loss: 0.0795 - val_accuracy: 0.9669
```

```
Epoch 9/10
14/14 [=====] - 24s 2s/step - loss: 0.1577 - accuracy: 0.9381 - val_loss: 0.0851 - val_accuracy: 0.9752
Epoch 10/10
14/14 [=====] - 24s 2s/step - loss: 0.1690 - accuracy: 0.9289 - val_loss: 0.0647 - val_accuracy: 0.9752
```

Out[13]:

```
In [14]: model.save("forest1.h5")
```

In [ ]: