{

"nbformat": 4,

"nbformat\_minor": 0,

"metadata": {

"colab": {

"provenance": []

},

"kernelspec": {

"name": "python3",

"display\_name": "Python 3"

},

"language\_info": {

"name": "python"

}

},

"cells": [

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {

"id": "BD3c4d85kL\_O"

},

"outputs": [],

"source": [

"import keras\n",

"from keras.preprocessing.image import ImageDataGenerator"

]

},

{

"cell\_type": "code",

"source": [

"#Define the parameters/arguments for ImageDataGenerator class\n",

"train\_datagen=ImageDataGenerator(rescale=1./255,shear\_range=0.2,rotation\_range=180,zoom\_range=0.2,horizontal\_flip=True)\n",

"\n",

"test\_datagen=ImageDataGenerator(rescale=1./255)"

],

"metadata": {

"id": "-8114SkekYDy"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#Applying ImageDataGenerator functionality to trainset\n",

"x\_train=train\_datagen.flow\_from\_directory('/content/drive/MyDrive/Dataset/Dataset/train\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')"

],

"metadata": {

"id": "WkKnprURkdF1"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#Applying ImageDataGenerator functionality to testset\n",

"x\_test=test\_datagen.flow\_from\_directory('/content/drive/MyDrive/Dataset/Dataset/test\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')"

],

"metadata": {

"id": "Xx66sQAnkfvW"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#import model building libraries\n",

"\n",

"#To define Linear initialisation import Sequential\n",

"from keras.models import Sequential\n",

"#To add layers import Dense\n",

"from keras.layers import Dense\n",

"#To create Convolution kernel import Convolution2D\n",

"from keras.layers import Convolution2D\n",

"#import Maxpooling layer\n",

"from keras.layers import MaxPooling2D\n",

"#import flatten layer\n",

"from keras.layers import Flatten\n",

"import warnings\n",

"warnings.filterwarnings('ignore')"

],

"metadata": {

"id": "x6QvF1hnkknF"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#initializing the model\n",

"model=Sequential()"

],

"metadata": {

"id": "cy3OIPewklbn"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#add convolutional layer\n",

"model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu'))\n",

"#add maxpooling layer\n",

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

"#add flatten layer \n",

"model.add(Flatten())"

],

"metadata": {

"id": "\_cLy4UOPkn4F"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#add hidden layer\n",

"model.add(Dense(150,activation='relu'))\n",

"#add output layer\n",

"model.add(Dense(1,activation='sigmoid'))"

],

"metadata": {

"id": "pgtRSgT3lrGz"

},

"execution\_count": null,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#configure the learning process\n",

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

],

"metadata": {

"id": "G64HrbPkmrkj"

},

"execution\_count": null,

"outputs": []

}

]

}