```
pip install --upgrade pip
```

```
Requirement already satisfied: pip in /usr/local/lib/python3.7/dist-packages (21.1.3)
```

```
!pip install keras-tuner
```

```
Collecting keras-tuner
  Downloading keras_tuner-1.0.3-py3-none-any.whl (96 kB)
     |████████████████████████████████| 96 kB 6.3 MB/s
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from ker
Collecting kt-legacy
  Downloading kt-legacy-1.0.3.tar.gz (5.8 kB)
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: tensorboard in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: ipython in /usr/local/lib/python3.7/dist-packages (from k
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from ker
Requirement already satisfied: prompt-toolkit<2.0.0,>=1.0.4 in /usr/local/lib/python3.7/
Requirement already satisfied: simplegeneric>0.8 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: setuptools>=18.5 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: decorator in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: traitlets>=4.2 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: pexpect in /usr/local/lib/python3.7/dist-packages (from i
Requirement already satisfied: pygments in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: pickleshare in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: wcwidth in /usr/local/lib/python3.7/dist-packages (from p
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: ipython-genutils in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: pyparsing>=2.0.2 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: grpcio>=1.24.3 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: absl-py>=0.4 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: google-auth<2,>=1.6.3 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: protobuf>=3.6.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.7
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/lib/p
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dist
```

```
Building wheels for collected packages: kt-legacy
  Building wheel for kt-legacy (setup.py) ... done
  Created wheel for kt-legacy: filename=kt_legacy-1.0.3-py3-none-any.whl size=9568 sha25
  Stored in directory: /root/.cache/pip/wheels/38/5c/e0/13003e68c17f403af40b92a24d20171b
Successfully built kt-legacy
Installing collected packages: kt-legacy, keras-tuner
Successfully installed keras-tuner-1.0.3 kt-legacy-1.0.3
```

```
!pip install imutils
```

```
Requirement already satisfied: imutils in /usr/local/lib/python3.7/dist-packages (0.5.4)
```

```
import tensorflow as tf
from tensorflow import keras
import numpy as np
import os
#import cv2
from imutils import paths
from keras.utils.np_utils import to_categorical
from sklearn.model_selection import train_test_split
import pandas as pd
import matplotlib.pyplot as plt
from math import sqrt
from tqdm import tqdm
from PIL import Image, ImageChops, ImageEnhance
from sklearn.metrics import roc_curve, roc_auc_score, auc, mean_squared_error, classification
from sklearn.metrics import accuracy_score
```

```
print(tf.__version__)
```

```
2.5.0
```

```
tf.test.gpu_device_name()
```

```
'/device:GPU:0'
```

```
def ErrorLevelAnalysis(path, quality):
    filename = path
    resaved_filename = filename.split('.')[0] + '.resaved.jpg'
    ELA_filename = filename.split('.')[0] + '.ela.png'

    im = Image.open(filename).convert('RGB')
    im.save(resaved_filename, 'JPEG', quality=quality)
    resaved_im = Image.open(resaved_filename)
```

```
        os.remove(resaved_filename)
        ela_im = ImageChops.difference(im, resaved_im)


        extrema = ela_im.getextrema()
        max_diff = max([ex[1] for ex in extrema])
        if max_diff == 0:
            max_diff = 1
        scale = 255.0 / max_diff

        ela_im = ImageEnhance.Brightness(ela_im).enhance(scale)

        return ela_im




#Image.open("/kaggle/input/imgtst/imgtst/org/Im_2.jpg")


#ErrorLevelAnalysis("/kaggle/input/imgtst/imgtst/org/Im_2.jpg")


import io,sys



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

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
```

```
Image.open("drive/My Drive/collab/csa/original.jpg")


from IPython.display import Image
Image('drive/My Drive/collab/csa/original.jpg')
Image('drive/My Drive/collab/csa/original.jpg', width=100, height=100)


path_org="drive/My Drive/collab/csa/ogr/"
path_fak="drive/My Drive/collab/csa/fke/"




org = os.listdir(path_org)
fak = os.listdir(path_fak)
```

```
images_names = []

for imgname in tqdm(os.listdir(path_org)):
    try:
        if imgname.endswith('png') or imgname.endswith('jpg'):

                imgnamefinal = path_org +'/'+ imgname  + ',0\n'
                images_names.append(imgnamefinal)

    except:
           print(path_org+imgname)


for imgname in tqdm(os.listdir(path_fak)):
    try:
        if imgname.endswith('png') or imgname.endswith('jpg') :

                    imgnamefinal = path_fak +'/'+imgname + ',1\n'
                    images_names.append(imgnamefinal)

    except:
          print(path_fak+imgname)
```

```
 100%|██████████| 800/800 [00:00<00:00, 244316.53it/s]
 100%|██████████| 921/921 [00:00<00:00, 345863.91it/s]
```

```
len(images_names)
```

```
 1711
```

```
image_name = []
label = []
for i in tqdm(range(len(images_names))):
    image_name.append(images_names[i][0:-3])
    label.append(images_names[i][-2])

dataset = pd.DataFrame({'image':image_name,'output':label})
dataset['output'].value_counts()
```

```
 100%|██████████| 1711/1711 [00:00<00:00, 513594.37it/s]
 1    921
 0    790
 Name: output, dtype: int64
```

```
dataset.to_csv('DLdataset.csv',index=False)
```

```
dataset = pd.read_csv('DLdataset.csv')
```

```
dataset=dataset.sample(frac=1)
```

```
dataset.shape
```

```
(1711, 2)
```

```
dataset.head(10)
```

	image	output
109	drive/My Drive/collab/csa/ogr//Au_arc_0093.jpg	0
1454	drive/My Drive/collab/csa/fke//Sp_S_NNN_R_txt0...	1
361	drive/My Drive/collab/csa/ogr//Au_cha_0078.jpg	0
680	drive/My Drive/collab/csa/ogr//Au_sec_0085.jpg	0
45	drive/My Drive/collab/csa/ogr//Au_ani_0027.jpg	0
1390	drive/My Drive/collab/csa/fke//Sp_S_NNN_C_txt0...	1
255	drive/My Drive/collab/csa/ogr//Au_art_0100.jpg	0
1147	drive/My Drive/collab/csa/fke//Sp_D_NRN_A_cha0...	1
1011	drive/My Drive/collab/csa/fke//Sp_D_NNN_A_txt0...	1
149	drive/My Drive/collab/csa/ogr//Au_ani_0044.jpg	0

```
x=[]
y=[]
```

```
for index, data in tqdm(dataset.iterrows()):
```

```
    x.append(np.array(ErrorLevelAnalysis(data[0],90).resize((192, 192))).flatten() / 255.0)
    y.append(data[1])
```

```
1711it [15:58,  1.78it/s]
```

```
X = np.array(x)
Y = np.array(y)



X = X.reshape(-1, 192, 192, 3)
Y = to_categorical(Y, 2)
X.shape,Y.shape
```

```
((1711, 192, 192, 3), (1711, 2))
```

```
from numpy import save
## save all the data
save('X_.npy', X)
save('Y_.npy',Y)



from numpy import load
x_ = load('X_.npy')
y_ = load('Y_.npy')
x_.shape,y_.shape
```

```
((1711, 192, 192, 3), (1711, 2))
```

```
X_train, X_test, Y_train, Y_test = train_test_split(x_, y_, test_size = 0.3,shuffle=True, ran


X_train.shape,Y_train.shape,X_test.shape,Y_test.shape
```

```
((1197, 192, 192, 3), (1197, 2), (514, 192, 192, 3), (514, 2))
```

```
def build_model(hp):
    model = keras.Sequential([

    keras.layers.Conv2D(
    filters=hp.Int('conv_1_filter', min_value=32, max_value=96, step=16),
    kernel_size=hp.Choice('conv_1_kernel', values = [5,5]),
    activation='relu',
    input_shape=X_train.shape[1:]
    ),
    keras.layers.MaxPooling2D(
    pool_size=hp.Choice('2d2',values=[2,2]), strides=None,padding="valid", data_format=None
    ),
    keras.layers.Conv2D(
    filters=hp.Int('conv_2_filter', min_value=32, max_value=64, step=16),
    kernel_size=hp.Choice('conv_2_kernel', values = [5,5]),
    activation='relu'
    ),
    keras.layers.MaxPooling2D( pool_size=hp.Choice('2d2',values=[2,2]), strides=None,padding=
    ),
```

```
    keras.layers.Dropout(0.5, noise_shape=None, seed=None),
    keras.layers.Flatten(),
    keras.layers.Dense(
    units=hp.Int('dense_1_units', min_value=32, max_value=256, step=16),
    activation='relu'
    ),
    keras.layers.Dropout(0.5, noise_shape=None, seed=None),
    keras.layers.Dense(2, activation='softmax')
    ])

    model.compile(optimizer=keras.optimizers.Adam(hp.Choice('learning_rate', values=[1e-2, 1e
                                                          metrics=['accuracy'])



#   keras.optimizers.RMSprop(learning_rate=0.001,rho=0.9, momentum=0.0,epsilon=1e-07, centere


    return model


from kerastuner import RandomSearch
from kerastuner.engine.hyperparameters import HyperParameters



tf.test.gpu_device_name()

     '/device:GPU:0'


tuner_search=RandomSearch(build_model,
                          objective='val_accuracy',
                          max_trials=5,directory='output',project_name="fake image")



tuner_search.search(X_train,Y_train,epochs=30,validation_data = (X_test, Y_test))

     Trial 5 Complete [00h 01m 24s]
     val_accuracy: 0.5564202070236206

     Best val_accuracy So Far: 0.8346303701400757
     Total elapsed time: 00h 12m 46s
     INFO:tensorflow:Oracle triggered exit


model=tuner_search.get_best_models(num_models=1)[0]


model.summary()

     Model: "sequential"
     _________________________________________________________________
```

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 188, 188, 96)      7296
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 94, 94, 96)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 90, 90, 64)        153664
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 45, 45, 64)        0
_________________________________________________________________
dropout (Dropout)            (None, 45, 45, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 129600)            0
_________________________________________________________________
dense (Dense)                (None, 48)                6220848
_________________________________________________________________
dropout_1 (Dropout)          (None, 48)                0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 98
=================================================================
Total params: 6,381,906
Trainable params: 6,381,906
Non-trainable params: 0
_________________________________________________________________
```

```
model.fit(X_train,Y_train, batch_size=100, epochs=30 , validation_data = (X_test, Y_test))
```

```
Epoch 1/30
12/12 [==============================] - 15s 927ms/step - loss: 0.4142 - accuracy: 0.
Epoch 2/30
12/12 [==============================] - 6s 520ms/step - loss: 0.4213 - accuracy: 0.7
Epoch 3/30
12/12 [==============================] - 6s 517ms/step - loss: 0.3998 - accuracy: 0.8
Epoch 4/30
12/12 [==============================] - 6s 518ms/step - loss: 0.4167 - accuracy: 0.7
Epoch 5/30
12/12 [==============================] - 6s 520ms/step - loss: 0.3960 - accuracy: 0.8
Epoch 6/30
12/12 [==============================] - 6s 518ms/step - loss: 0.3837 - accuracy: 0.8
Epoch 7/30
12/12 [==============================] - 6s 522ms/step - loss: 0.3989 - accuracy: 0.7
Epoch 8/30
12/12 [==============================] - 6s 520ms/step - loss: 0.3735 - accuracy: 0.8
Epoch 9/30
12/12 [==============================] - 6s 520ms/step - loss: 0.3824 - accuracy: 0.8
Epoch 10/30
12/12 [==============================] - 6s 523ms/step - loss: 0.3937 - accuracy: 0.7
Epoch 11/30
12/12 [==============================] - 6s 517ms/step - loss: 0.3637 - accuracy: 0.8
Epoch 12/30
12/12 [==============================] - 6s 518ms/step - loss: 0.3787 - accuracy: 0.7
Epoch 13/30
12/12 [==============================] - 6s 523ms/step - loss: 0.3549 - accuracy: 0.8
```

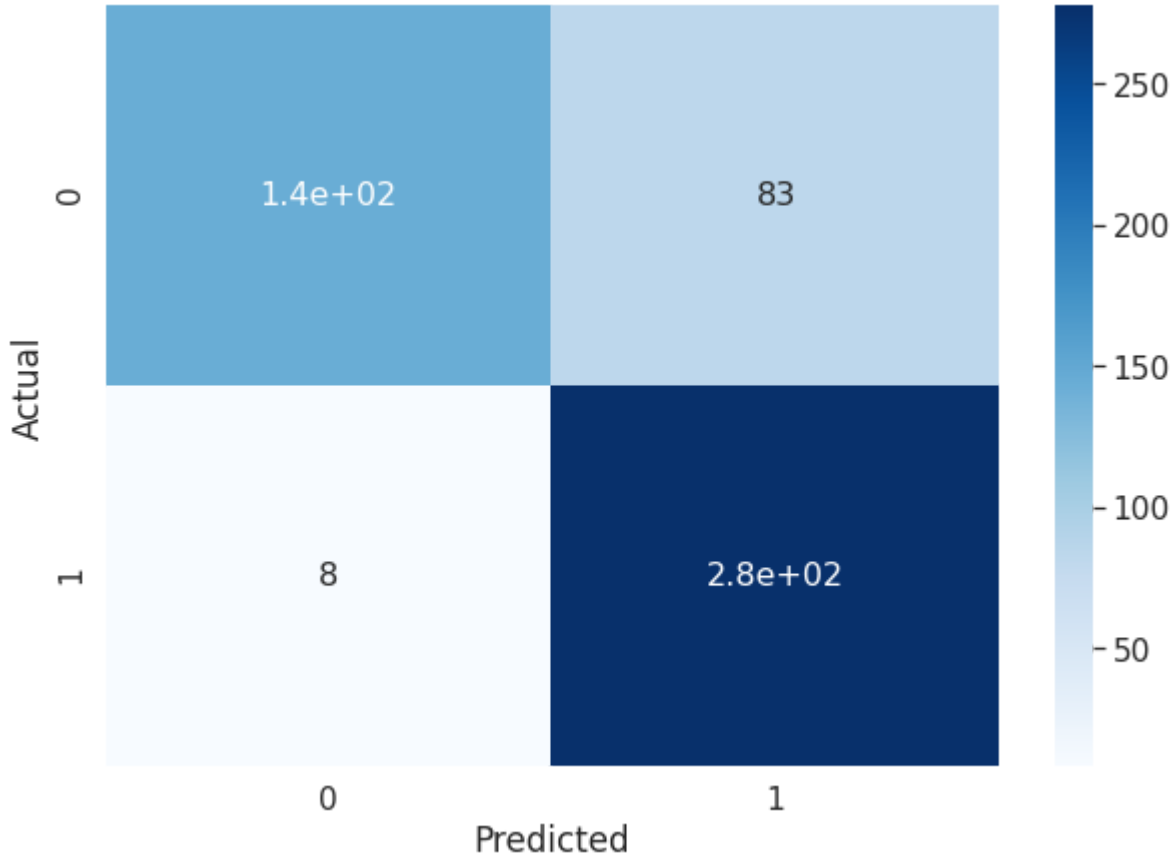
```
Epoch 14/30
12/12 [==============================] - 6s 521ms/step - loss: 0.3384 - accuracy: 0.8
Epoch 15/30
12/12 [==============================] - 6s 521ms/step - loss: 0.3616 - accuracy: 0.8
Epoch 16/30
12/12 [==============================] - 6s 520ms/step - loss: 0.3684 - accuracy: 0.7
Epoch 17/30
12/12 [==============================] - 6s 520ms/step - loss: 0.3668 - accuracy: 0.8
Epoch 18/30
12/12 [==============================] - 6s 521ms/step - loss: 0.3563 - accuracy: 0.8
Epoch 19/30
12/12 [==============================] - 6s 521ms/step - loss: 0.3468 - accuracy: 0.8
Epoch 20/30
12/12 [==============================] - 6s 518ms/step - loss: 0.3355 - accuracy: 0.8
Epoch 21/30
12/12 [==============================] - 6s 521ms/step - loss: 0.3391 - accuracy: 0.8
Epoch 22/30
12/12 [==============================] - 6s 519ms/step - loss: 0.3326 - accuracy: 0.8
Epoch 23/30
12/12 [==============================] - 6s 522ms/step - loss: 0.3273 - accuracy: 0.8
Epoch 24/30
12/12 [==============================] - 6s 518ms/step - loss: 0.3159 - accuracy: 0.8
Epoch 25/30
12/12 [==============================] - 6s 523ms/step - loss: 0.3376 - accuracy: 0.8
Epoch 26/30
12/12 [==============================] - 6s 522ms/step - loss: 0.3241 - accuracy: 0.8
Epoch 27/30
12/12 [==============================] - 6s 519ms/step - loss: 0.3051 - accuracy: 0.8
Epoch 28/30
12/12 [==============================] - 6s 518ms/step - loss: 0.3018 - accuracy: 0.8
Epoch 29/30
12/12 [==============================] - 6s 521ms/step - loss: 0.3025 - accuracy: 0.8
```

```
from sklearn import metrics
from sklearn.metrics import confusion_matrix, classification_report
y_pred_cnn1 = model.predict(X_test)
y_pred_cnn = np.argmax(y_pred_cnn1,axis = 1)

Y_true = np.argmax(Y_test,axis = 1)
score = accuracy_score(Y_true,  y_pred_cnn)
print("Accuracy score: {}".format(score))
score = metrics.precision_score(Y_true,y_pred_cnn, average= "weighted")
print("Precision score: {}".format(score))
score = metrics.recall_score(Y_true, y_pred_cnn, average= "weighted")
print("Recall score: {}".format(score))
score_lr1 = metrics.f1_score(Y_true, y_pred_cnn, average= "weighted")
print("F1 score: {}".format(score_lr1))
```

```
Accuracy score: 0.8229571984435797
Precision score: 0.8488758729291215
Recall score: 0.8229571984435797
F1 score: 0.8157930937912765
```

```
import seaborn as sns
cm = confusion_matrix(Y_true, y_pred_cnn)
print('Confusion matrix:\n',cm)

print(classification_report(Y_true, y_pred_cnn))

print('Plot of Confusion Matrix')
df_cm = pd.DataFrame(cm, columns=np.unique(Y_true), index = np.unique(Y_true))
df_cm.index.name = 'Actual'
df_cm.columns.name = 'Predicted'
plt.figure(figsize = (10,7))
sns.set(font_scale=1.4)#for label size
sns.heatmap(df_cm, cmap="Blues", annot=True,annot_kws={"size": 16})# font size
```

```
Confusion matrix:
 [[145  83]
 [  8 278]]
              precision    recall  f1-score   support

           0       0.95      0.64      0.76       228
           1       0.77      0.97      0.86       286

    accuracy                           0.82       514
   macro avg       0.86      0.80      0.81       514
weighted avg       0.85      0.82      0.82       514

Plot of Confusion Matrix
<matplotlib.axes._subplots.AxesSubplot at 0x7f905f8476d0>
```



```
cnn_score = model.evaluate(X_test, Y_test, verbose=3)
print ('Test loss:', cnn_score[0])
print ('Test accuracy:', cnn_score[1])
```

```
Test loss: 0.36998090147972107
Test accuracy: 0.8229572176933289
```

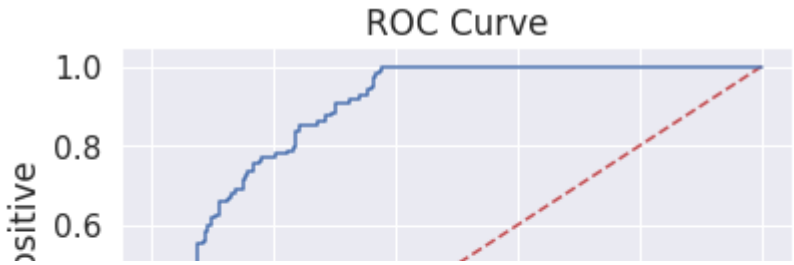
```
def plot_roc_curve(y_true, y_pred, y_proba):
    rmse = sqrt(mean_squared_error(y_true, y_pred))
    print('RMSE', rmse)
    from sklearn.metrics import roc_auc_score
    print('ROC_AUC score:',roc_auc_score(Y_true,y_pred_cnn))


    FPR, TPR, thresholds = roc_curve(y_true, y_proba)
    roc_auc = auc(FPR, TPR)
    plt.plot([0, 1], [0, 1], 'r--')
    plt.plot(FPR, TPR, label='' % roc_auc)
    plt.title('ROC Curve')
    plt.legend(loc='lower right')
    plt.ylabel('True Positive')
    plt.xlabel('False Positive')
    plt.show()
```

```
cnn_model_y_proba=model.predict(X_test,verbose=3)
cnn_model_y_proba
```

```
array([[6.0350806e-02, 9.3964916e-01],
       [3.0467451e-01, 6.9532549e-01],
       [9.9999952e-01, 4.2751927e-07],
       ...,
       [3.2478549e-02, 9.6752143e-01],
       [2.1233760e-01, 7.8766245e-01],
       [8.0852872e-03, 9.9191475e-01]], dtype=float32)
```

```
plot_roc_curve(Y_true, y_pred_cnn, cnn_model_y_proba[:,1])
```

```
No handles with labels found to put in legend.
RMSE 0.42076454408186564
ROC_AUC score: 0.8039964421543369
```




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

```
Mounted at /content/gdrive
```

```
model.save("/content/gdrive/My Drive/collab/my_model.h5")
```

```
new_model=keras.models.load_model("/content/gdrive/My Drive/collab/my_model.h5")
```

```
WARNING:tensorflow:Error in loading the saved optimizer state. As a result, your model i
```

```
cnn_score = new_model.evaluate(X_test, Y_test, verbose=3)
print ('Test loss:', cnn_score[0])
print ('Test accuracy:', cnn_score[1])
```

```
Test loss: 0.36998090147972107
Test accuracy: 0.8229572176933289
```

```
import pickle

from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Dense
from tensorflow.python.keras.layers import deserialize, serialize
from tensorflow.python.keras.saving import saving_utils


def unpack(model, training_config, weights):
    restored_model = deserialize(model)
    if training_config is not None:
        restored_model.compile(
            **saving_utils.compile_args_from_training_config(
                training_config
            )
        )
    restored_model.set_weights(weights)
    return restored_model

# Hotfix function
```

```
# Model.reduce function
def make_keras_picklable():

    def __reduce__(self):
        model_metadata = saving_utils.model_metadata(self)
        training_config = model_metadata.get("training_config", None)
        model = serialize(self)
        weights = self.get_weights()
        return (unpack, (model, training_config, weights))

    cls = Model
    cls.__reduce__ = __reduce__

# Run the function
make_keras_picklable()


# Save
with open('model.pkl', 'wb') as f:
    pickle.dump(model, f)


# # open a file, where you stored the pickled data
# file = open('model.pkl', 'rb')

# # dump information to that file
# modell = pickle.load(file)


# y_pred_cnn1 = modell.predict(X_test)
# y_pred_cnn = np.argmax(y_pred_cnn1,axis = 1)

# Y_true = np.argmax(Y_test,axis = 1)
# score = accuracy_score(Y_true,  y_pred_cnn)
# print("Accuracy score: {}".format(score))









from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials

# 1. Authenticate and create the PyDrive client.
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)
```

```
# get the folder id where you want to save your file
file = drive.CreateFile({'parents':[{u'id': '1dfeVOBXHsgegh5zn0R7JlrQAyjwTefUO'}]})
file.SetContentFile('model.pkl')
file.Upload()

import pickle

class MyClass:
    def __init__(self, name):
        self.name = name

if __name__ == '__main__':
    o = MyClass('test')
    with open('model.pkl', 'wb') as f:
        pickle.dump(o, f)
```

```
Go to the following link in your browser:

    https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=32555940559.a

Enter verification code: 4/1AY0e-g7qLfiagouPU7FYEnk8umJua5CXYBLob9tZrsysz6NnIvzCDy-nD44
```

```
DATA_PATH =
infile = open(DATA_PATH+'/model.pkl','rb')
best_model2 = pickle.load(infile)
```

```
  File "<ipython-input-56-a3ab88082dc9>", line 1
    DATA_PATH =
              ^
SyntaxError: invalid syntax
```

SEARCH STACK OVERFLOW

```
from google.colab import files
mode=files.download('model.pkl')
```

```
y_pred_cnn1 = best_model2.predict(X_test)
y_pred_cnn = np.argmax(y_pred_cnn1,axis = 1)

Y_true = np.argmax(Y_test,axis = 1)
score = accuracy_score(Y_true,  y_pred_cnn)
print("Accuracy score: {}".format(score))
```

✓ 0s completed at 11:38 PM