In [271]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import cv2
import os
import PIL
import time
import wikipedia
%matplotlib inline
```

In [147]:
```
face_cascade=cv2.CascadeClassifier(r"D:\ML Model\xmlfiles\haarcascade_frontalf
eye_cascade=cv2.CascadeClassifier(r"D:\ML Model\xmlfiles\haarcascade_eye.xml")
```

In [148]:
```
def get_cropped(image_path):
    img=cv2.imread(image_path)
    gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    face_cascade=cv2.CascadeClassifier(r"D:\ML Model\xmlfiles\haarcascade_fror
    eye_cascade=cv2.CascadeClassifier(r"D:\ML Model\xmlfiles\haarcascade_eye.x
    faces=face_cascade.detectMultiScale(gray,1.3,5)
    roi_gray = gray
    roi_color = img
    for(x,y,w,h)in faces:
        roi_gray=gray[y:y+h,x:x+w]
        roi_color=img[y:y+h,x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    if len(eyes)>=2:
        for (ex, ey, ew, eh) in eyes:
            cv2.rectangle(roi_color, (ex,ey),(ex+ew,ey+eh),(0,255,0),2)
    return roi_color
```

In [149]:
```
path_to_data=r"D:\ML Model\modelpics"
path_to_cropdata=r"D:\ML Model\Cropimages"
```

In [150]:
```
import os
img_dir=[]
for items in os.scandir(path_to_data):
    if items.is_dir():
        img_dir.append(items.path)
img_dir
```

Out[150]:
```
['D:\\ML Model\\modelpics\\CristianoRonaldo',
 'D:\\ML Model\\modelpics\\LionelMessi',
 'D:\\ML Model\\modelpics\\MahendraSinghDhoni']
```

In [151]:
```
import shutil
if os.path.exists(path_to_cropdata):
     shutil.rmtree(path_to_cropdata)
os.mkdir(path_to_cropdata)
```

In [152]:
```
cropped_image_dirs = []
file_names_dict = {}

for img in img_dir:
    name=img.split('\\')[-1]
    print(name)
    file_names_dict[name] = []
    count=1
    for item in os.scandir(img):
        roi_color=get_cropped(item.path)
        if roi_color is not None:
            cropped_folder=path_to_cropdata+'\\'+name
            if not os.path.exists(cropped_folder):
                os.mkdir(cropped_folder)
                cropped_image_dirs.append(cropped_folder)
                print("generating images for cropped_folder",cropped_folder)
            file_name=name+str(count)+".png"
            file_path=cropped_folder+'\\'+file_name
            roi_color = cv2.cvtColor(roi_color, cv2.COLOR_BGR2RGB)
            cv2.imwrite(file_path,roi_color)
            file_names_dict[name].append(file_path)
            count+=1
```

```
CristianoRonaldo
generating images for cropped_folder D:\ML Model\Cropimages\CristianoRonaldo
LionelMessi
generating images for cropped_folder D:\ML Model\Cropimages\LionelMessi
MahendraSinghDhoni
generating images for cropped_folder D:\ML Model\Cropimages\MahendraSinghDho
ni
```

In [153]:
```
file_names_dict
```

Out[153]:

```
{'CristianoRonaldo': ['D:\\ML Model\\Cropimages\\CristianoRonaldo\\Cristiano
Ronaldo1.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo2.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo3.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo4.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo5.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo6.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo7.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo8.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo9.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo10.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo11.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo12.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo13.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo14.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo15.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo16.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo17.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo18.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo19.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo20.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo21.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo22.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo23.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo24.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo25.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo26.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo27.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo28.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo29.png',
  'D:\\ML Model\\Cropimages\\CristianoRonaldo\\CristianoRonaldo30.png'],
 'LionelMessi': ['D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi1.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi2.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi3.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi4.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi5.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi6.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi7.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi8.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi9.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi10.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi11.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi12.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi13.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi14.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi15.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi16.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi17.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi18.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi19.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi20.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi21.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi22.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi23.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi24.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi25.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi26.png',
```

```
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi27.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi28.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi29.png',
  'D:\\ML Model\\Cropimages\\LionelMessi\\LionelMessi30.png'],
 'MahendraSinghDhoni': ['D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\Mahen
draSinghDhoni1.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni2.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni3.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni4.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni5.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni6.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni7.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni8.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni9.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni10.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni11.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni12.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni13.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni14.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni15.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni16.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni17.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni18.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni19.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni20.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni21.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni22.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni23.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni24.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni25.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni26.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni27.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni28.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni29.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni30.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni31.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni32.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni33.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni34.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni35.png',
  'D:\\ML Model\\Cropimages\\MahendraSinghDhoni\\MahendraSinghDhoni36.png']}
```

In [155]:
```
import pywt
import numpy as np
def cwd(img,mode='haar',level=1):
    imarr=img
    imarr=cv2.cvtColor(imarr,cv2.COLOR_RGB2GRAY)
    imarr=np.float32(imarr)
    imarr /= 255
     # compute coefficients
    coeffs=pywt.wavedec2(imarr, mode, level=level)

    #Process Coefficients
    coeffs_H=list(coeffs)
    coeffs_H[0] *= 0

    # reconstruction
    imArray_H=pywt.waverec2(coeffs_H, mode)
    imArray_H *= 255;
    imArray_H =  np.uint8(imArray_H)

    return imArray_H
```

In [156]:
```
per_dict={}
count=1
for person_names in file_names_dict.keys():
    per_dict[person_names]=count
    count+=1
per_dict
```

Out[156]:
```
{'CristianoRonaldo': 1, 'LionelMessi': 2, 'MahendraSinghDhoni': 3}
```

In [157]:
```
x=[]
y=[]
for person_names,training_files in file_names_dict.items():
    for image in training_files:
        raw_img=cv2.imread(image)
        scaled_raw_img=cv2.resize(raw_img,(32,32))
        haar_img=cwd(raw_img,'db1',5)
        scaled_haar_img=cv2.resize(haar_img,(32,32))
        combined_img=np.vstack((scaled_raw_img.reshape(32*32*3,1),scaled_haar_
        x.append(combined_img)
        y.append(per_dict[person_names])
```

In [158]:
```
x=np.array(x).reshape(len(x),4096).astype(float)
x.shape
x[0]
```

Out[158]:
```
array([127., 135., 131., ...,  11., 137.,  78.])
```

In [159]:
```
len(x)
```

Out[159]:
```
96
```

In [160]:
```
len(y)
```

Out[160]: 96

In [161]:
```
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
```

In [162]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=
```

In [163]:
```
model=Pipeline([('scaler' ,StandardScaler()) ,('svc' ,SVC(kernel='rbf',C=10))]
model.fit(x_train,y_train)
model.score(x_test,y_test)
print(classification_report(y_test,model.predict(x_test)))
```

```
              precision    recall  f1-score   support

           1       0.80      0.80      0.80        10
           2       0.67      0.75      0.71         8
           3       0.90      0.82      0.86        11

    accuracy                           0.79        29
   macro avg       0.79      0.79      0.79        29
weighted avg       0.80      0.79      0.80        29
```

In [164]:
```
model.score(x_test,y_test)
```

Out[164]: 0.7931034482758621

In [165]:
```
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
```

In [166]:
```
model_params={'svm':{'model':SVC(gamma='auto',probability=True),'param':{'svc_
              'random_forest':{'model':RandomForestClassifier(),'param':{'n_es
             'logistic_Regression':{'model':LogisticRegression(solver='libline
```

In [167]:
```
#svc
```

In [168]:
```
clf=GridSearchCV(SVC(gamma='auto'),{'C':[1,10,20],'kernel':['rbf','linear']},c
clf.fit(x_train,y_train)
```

Out[168]:
```
▸ GridSearchCV
▸ estimator: SVC
    ▸ SVC
```

In [169]:
```
clf.best_params_
```

Out[169]:
```
{'C': 1, 'kernel': 'linear'}
```

In [170]:
```
model=Pipeline([('scaler' ,StandardScaler()) ,('svc' ,SVC(kernel='linear',C=1)
model.fit(x_train,y_train)
model.score(x_test,y_test)
```

Out[170]:
```
0.7931034482758621
```

In [171]:
```
sel_model=SVC(kernel='linear',C=1)
sel_model.fit(x_train,y_train)
```

Out[171]:
```
▾ SVC
SVC(C=1, kernel='linear')
```

In [172]:
```
sel_model.score(x_test,y_test)
```

Out[172]:
```
0.7931034482758621
```

In [173]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model,x,y,cv=3,scoring='f1_macro')
score
```

Out[173]:
```
array([0.713868  , 0.80656085, 0.66269841])
```

In [174]:
```
len(x)
```

Out[174]:
```
96
```

In [175]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model,x,y,cv=5,scoring='f1_macro')
score
```

Out[175]:
```
array([0.80952381, 0.8964369 , 0.8964369 , 0.70238095, 0.78821179])
```

In [176]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model,x,y,cv=7,scoring='f1_macro')
score
```

Out[176]:
```
array([0.72222222, 0.92592593, 0.85925926, 0.85541126, 0.82621083,
       0.92592593, 0.76623377])
```

In [177]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model,x,y,cv=10,scoring='f1_macro')
score
```

Out[177]:
```
array([0.61111111, 1.        , 0.88571429, 0.9047619 , 0.9047619 ,
       0.8962963 , 0.63888889, 0.75      , 0.88571429, 0.73809524])
```

In [178]:
```
#gaussianNB with scaling
from sklearn.naive_bayes import GaussianNB
model1=Pipeline([('scaler' ,StandardScaler()) ,('gausiannb' ,GaussianNB(priors
model1.fit(x_train,y_train)
model1.score(x_test,y_test)
```

Out[178]:
```
0.7586206896551724
```

In [179]:
```
from sklearn.naive_bayes import GaussianNB
model1=GaussianNB(priors=None)
model1.fit(x_train,y_train)
```

Out[179]:
```
GaussianNB
GaussianNB()
```

In [180]:
```
model1.get_params()
```

Out[180]:
```
{'priors': None, 'var_smoothing': 1e-09}
```

In [181]:
```
model1.score(x_test,y_test)
```

Out[181]:
```
0.7586206896551724
```

In [182]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model1,x,y,cv=3,scoring='f1_macro')
score
```

Out[182]:
```
array([0.68686869, 0.76190476, 0.65079365])
```

In [183]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model1,x,y,cv=5,scoring='f1_macro')
score
```

Out[183]:
```
array([0.58566434, 0.88571429, 0.84420677, 0.60714286, 0.61105169])
```

In [184]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model1,x,y,cv=7,scoring='f1_macro')
score
```

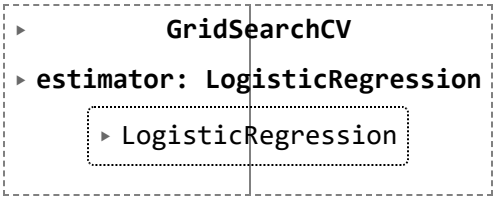
Out[184]:
```
array([0.71851852, 0.85978836, 0.92592593, 0.85978836, 0.67936508,
       0.77813853, 0.45512821])
```

In [185]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model1,x,y,cv=10,scoring='f1_macro')
score
```

Out[185]:
```
array([0.80555556, 0.65343915, 0.77777778, 0.88571429, 0.88571429,
       0.8962963 , 0.44444444, 0.65555556, 0.67936508, 0.46666667])
```

In [186]:
```
#gridsearchcv on logisticregression
clf=GridSearchCV(LogisticRegression(solver='liblinear'),{'C':[1,10,20]},cv=2,r
clf.fit(x_train,y_train)
```

Out[186]:



In [187]:
```
clf.best_params_
```

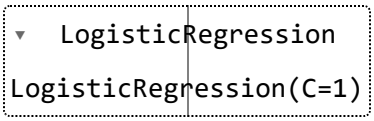
Out[187]:
```
{'C': 1}
```

In [188]:
```
#logisticRegression with scaling
model4=Pipeline([('scaler' ,StandardScaler()) ,('logisticregression' ,Logistic
model4.fit(x_train,y_train)
model4.score(x_test,y_test)
```

Out[188]:
```
0.7931034482758621
```

In [189]:
```
from sklearn.linear_model import LogisticRegression
model2=LogisticRegression(C=1)
model2.fit(x_train,y_train)
```

Out[189]:



In [190]:
```
model2.get_params()
```

Out[190]:
```
{'C': 1,
 'class_weight': None,
 'dual': False,
 'fit_intercept': True,
 'intercept_scaling': 1,
 'l1_ratio': None,
 'max_iter': 100,
 'multi_class': 'auto',
 'n_jobs': None,
 'penalty': 'l2',
 'random_state': None,
 'solver': 'lbfgs',
 'tol': 0.0001,
 'verbose': 0,
 'warm_start': False}
```

In [191]:
```
model2.score(x_test,y_test)
```

Out[191]:
```
0.7241379310344828
```

In [192]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model2,x,y,cv=3,scoring='f1_macro')
score
```

Out[192]:
```
array([0.77777778, 0.84131054, 0.65894737])
```

In [193]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model2,x,y,cv=5,scoring='f1_macro')
score
```

Out[193]:
```
array([0.69047619, 0.94405594, 0.83809524, 0.65555556, 0.78821179])
```

In [194]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model2,x,y,cv=7,scoring='f1_macro')
score
```

Out[194]:
```
array([0.64478114, 0.85      , 0.93265993, 0.76666667, 0.75793651,
       0.75343915, 0.75343915])
```

In [195]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model2,x,y,cv=10,scoring='f1_macro')
score
```

Out[195]:
```
array([0.71428571, 0.66904762, 0.88571429, 0.9047619 , 0.9047619 ,
       0.74867725, 0.54761905, 0.65555556, 0.73809524, 0.73809524])
```

In [196]:
```
#randomforestclassifier
```

In [197]:
```
rf_classifier = RandomForestClassifier(random_state=42)

# Define the parameter grid for GridSearchCV
param_grid = {
    'n_estimators': [100, 200, 300],        # Number of trees in the forest
    'max_depth': [None, 10, 20, 30],        # Maximum depth of the trees
    'min_samples_split': [2, 5, 10],         # Minimum number of samples requi
    'min_samples_leaf': [1, 2, 4],           # Minimum number of samples requi
    'bootstrap': [True, False]               # Method of selecting samples for
}
```

In [198]:
```
grid_search = GridSearchCV(estimator=rf_classifier, param_grid=param_grid, cv=
```

In [199]:
```
grid_search.fit(x_train, y_train)
```

```
Fitting 3 folds for each of 216 candidates, totalling 648 fits
```

Out[199]:

GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier

In [200]:
```
grid_search.best_params_
```

Out[200]:
```
{'bootstrap': True,
 'max_depth': None,
 'min_samples_leaf': 1,
 'min_samples_split': 5,
 'n_estimators': 200}
```

In [ ]:
```
model3=RandomForestClassifier(n_estimators=100)
model3.fit(x_train,y_train)
```

In [202]:
```
model3.score(x_test,y_test)
```

Out[202]:
```
0.7931034482758621
```

In [203]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model3,x,y,cv=3,scoring='f1_macro')
score
```

Out[203]:
```
array([0.65106442, 0.72709552, 0.66615832])
```

In [204]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model3,x,y,cv=5,scoring='f1_macro')
score
```

Out[204]:
```
array([0.8965812 , 0.84126984, 0.84747475, 0.65555556, 0.78632479])
```

In [205]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model3,x,y,cv=7,scoring='f1_macro')
score
```

Out[205]:
```
array([0.6540404 , 0.6995671 , 0.85925926, 0.64957265, 0.71428571,
       0.92592593, 0.68677249])
```

In [206]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model3,x,y,cv=10,scoring='f1_macro')
score
```

Out[206]:
```
array([0.49206349, 1.        , 0.79365079, 0.88571429, 0.71428571,
       0.71428571, 0.55555556, 0.75      , 0.88571429, 0.73809524])
```

In [207]:
```
#DECISION TREE
from sklearn.tree import DecisionTreeClassifier
model5=DecisionTreeClassifier()
model5.get_params()
```

Out[207]:
```
{'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': None,
 'max_features': None,
 'max_leaf_nodes': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'random_state': None,
 'splitter': 'best'}
```

In [208]:
```
#parameters for decision tree for grid search cv
param_grid = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': [None, 'sqrt', 'log2'],
}
```

In [209]:
```
clf=GridSearchCV(model5,param_grid,cv=3)
```

In [210]:
```
clf.fit(x_train,y_train)
```

Out[210]:
```
GridSearchCV
estimator: DecisionTreeClassifier
DecisionTreeClassifier
```

In [211]:
```
clf.best_params_
```

Out[211]:
```
{'criterion': 'entropy',
 'max_depth': 30,
 'max_features': None,
 'min_samples_leaf': 1,
 'min_samples_split': 5,
 'splitter': 'best'}
```

In [212]:
```
model5=DecisionTreeClassifier(criterion='gini',max_depth=20,max_features=None,
model5.fit(x_train,y_train)
```

Out[212]:
```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=20, min_samples_split=5)
```

In [213]:
```
model5.score(x_test,y_test)
```

Out[213]:
```
0.6206896551724138
```

In [214]:
```
from sklearn.model_selection import cross_val_score
score=cross_val_score(model5,x,y,cv=3,scoring='f1_macro')
score
```

Out[214]:
```
array([0.56467236, 0.53037037, 0.57801673])
```

In [215]:
```
score=cross_val_score(model5,x,y,cv=5,scoring='f1_macro')
score
```

Out[215]:
```
array([0.57634033, 0.52614053, 0.67948718, 0.79059829, 0.61363636])
```

In [216]:
```
score=cross_val_score(model5,x,y,cv=7,scoring='f1_macro')
score
```

Out[216]:
```
array([0.48412698, 0.55723906, 0.63290043, 0.7       , 0.69688645,
       0.7       , 0.69444444])
```

In [217]:
```
score=cross_val_score(model5,x,y,cv=10,scoring='f1_macro')
score
```

Out[217]:
```
array([0.33333333, 0.62380952, 0.57777778, 0.40714286, 0.39814815,
       0.68888889, 0.19444444, 0.75      , 0.56825397, 0.78333333])
```

In [218]:
```
if not os.path.exists(r"D:\ML Model\modelfiles\sel_model.bin"):
    os.makedirs(r"D:\ML Model\modelfiles\sel_model.bin")
```

In [219]:
```

!pip install joblib
import joblib
filename=r"D:\ML Model\modelfiles\sel_model.pkl"

joblib.dump(sel_model, filename)
```

```
Defaulting to user installation because normal site-packages is not writeabl
e
Requirement already satisfied: joblib in c:\programdata\anaconda3\lib\site-p
ackages (1.1.1)

WARNING: Ignoring invalid distribution -pencv-python (c:\users\ramas\appdata
\roaming\python\python310\site-packages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\ramas\appdata
\roaming\python\python310\site-packages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\ramas\appdata
\roaming\python\python310\site-packages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\ramas\appdata
\roaming\python\python310\site-packages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\ramas\appdata
\roaming\python\python310\site-packages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\ramas\appdata
\roaming\python\python310\site-packages)
```

Out[219]:
```
['D:\\ML Model\\modelfiles\\sel_model.pkl']
```

In [220]:
```
print(sel_model)
```

```
SVC(C=1, kernel='linear')
```

In [221]:
```

import json
with open(r"D:\ML Model\modelfiles\class_directory.json",'w') as f:
    f.write(json.dumps(per_dict))
```

In [222]:
```
global __class_name_to_number
global __class_number_to_name

with open(r"D:\ML Model\modelfiles\class_directory.json", "r") as f:
    __class_name_to_number = json.load(f)
    __class_number_to_name = {v:k for k,v in __class_name_to_number.items()}
```

In [223]:
```
__class_number_to_name
```

Out[223]:
```
{1: 'CristianoRonaldo', 2: 'LionelMessi', 3: 'MahendraSinghDhoni'}
```

In [224]:
```
def class_number_to_name(num):
    return __class_number_to_name[num]
```

In [225]:
```
print( class_number_to_name(2))
```

```
LionelMessi
```

In [226]:
```
import joblib

with open(r"D:\ML Model\modelfiles\sel_model.pkl",'rb') as f:
        __model=joblib.load(f)
```

In [227]:
```
__model
```

Out[227]:
```
SVC
SVC(C=1, kernel='linear')
```

In [269]:
```
def predict_imagess(image_path):
    out=[]
    imag=get_cropped(image_path)
    scalled_raw=cv2.resize(imag,(32,32))
    img_har=cwd(imag,'db1',5)
    scalled_img_har=cv2.resize(img_har,(32,32))
    combined_img=np.vstack((scalled_raw.reshape(32*32*3,1),scalled_img_har.res
    len_image_array = 32*32*3 + 32*32
    final = combined_img.reshape(1,len_image_array).astype(float)
    result=class_number_to_name(__model.predict(final)[0])
    about=wikipedia.summary(result,sentences=2)
    out.append({'Name':result,
    'Information':about})
    return out
```

In [270]:
```
print(predict_imagess(r"D:\ML Model\Pictures\dhoni\8.jpg"))
```

```
[{'Name': 'MahendraSinghDhoni', 'Information': "Mahendra Singh Dhoni ( ; bor
n 7 July 1981) is an Indian professional cricketer, who plays as a wicket-ke
eper-batsman. Widely regarded as one of the world's greatest wicketkeeper ba
tsmen and captains in the history of the sport, he is known for his explosiv
e batting, wicket-keeping and leadership skills."}]
```