IrisRecognition

October 28, 2020

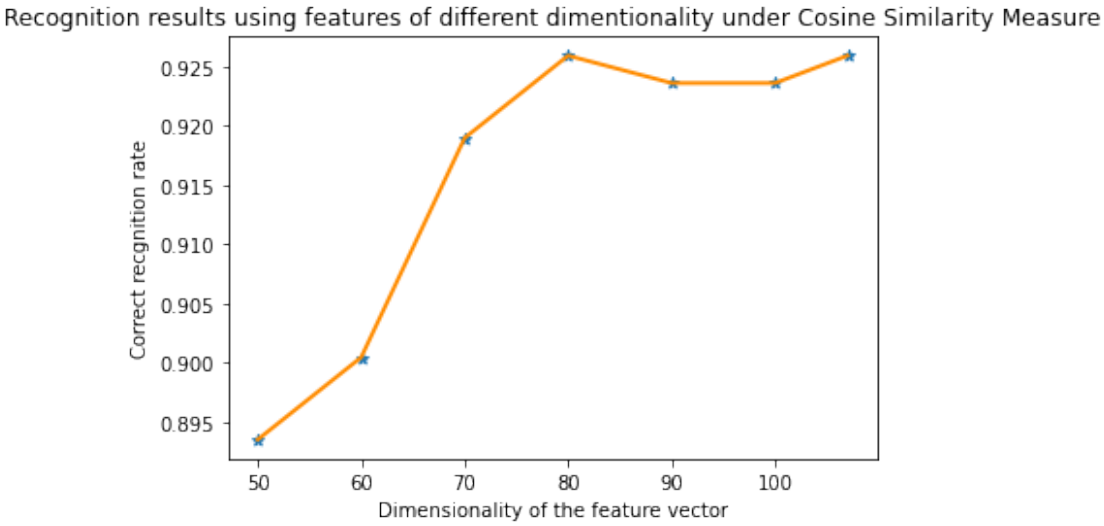
[3]:
```
import numpy as np
import cv2
import matplotlib.pyplot as plt
import pandas as pd
import scipy.signal

from sklearn.neighbors.nearest_centroid import NearestCentroid
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
from sklearn import metrics

from IrisLocalization  import *
from IrisNormalization import *
from ImageEnhancement import *
from FeatureExtraction import *
from IrisMatching      import *
from PerformanceEnvaluation import *
```

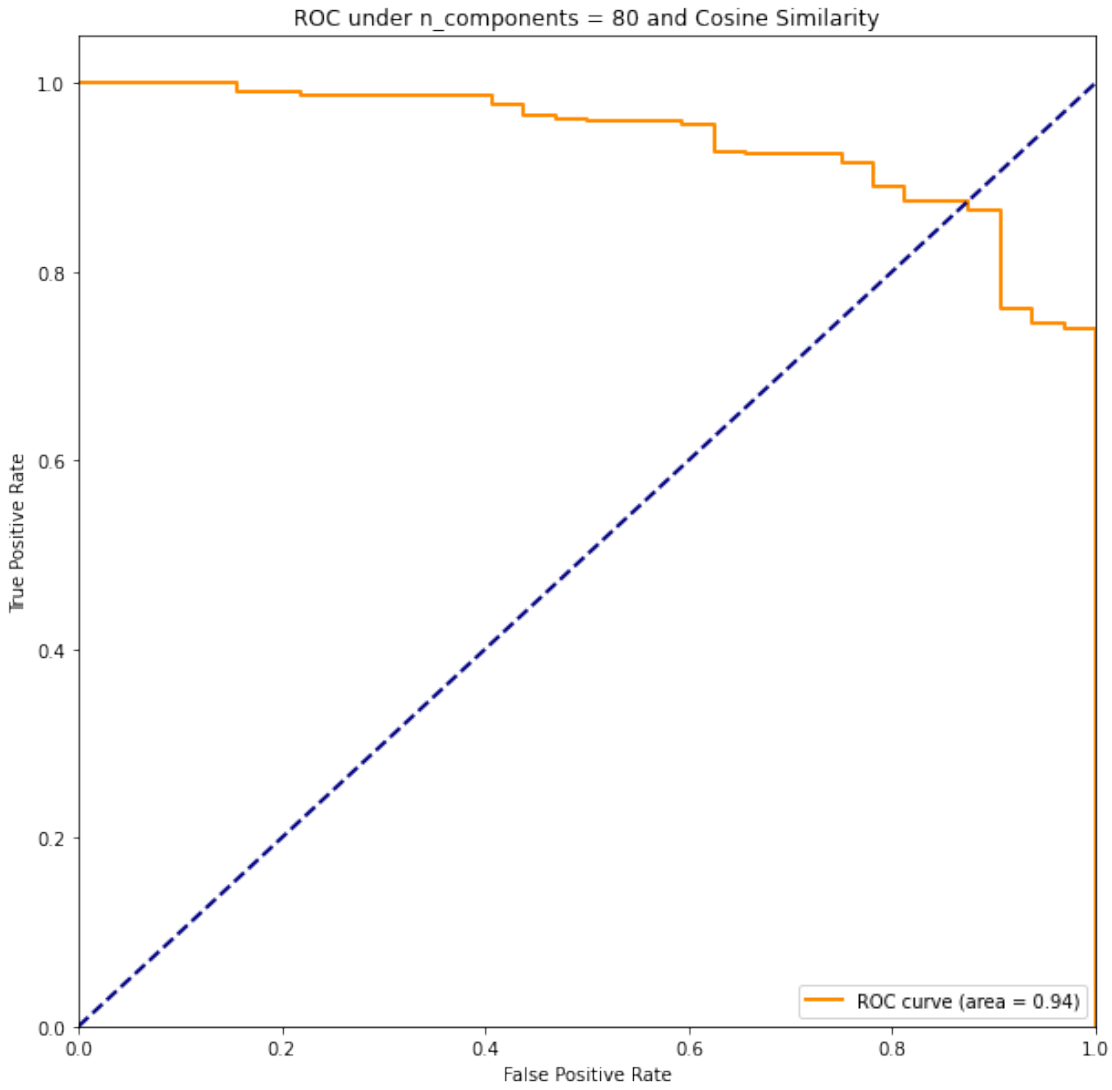
[4]:
```
def main():
    #get the feature of all training images and testing images
    #save the features since it takes very long time to run and we will␣
 ↪directly use it to draw plots and make accuracy table
    testBase = getDatabase(2)
    irisTest = np.array(testBase)
    np.save('irisTest',irisTest)

    trainBase = getDatabase(1)
    irisTrain = np.array(trainBase)
    np.save('irisTrain',irisTrain)



    train = np.load('irisTrain.npy')
    test = np.load('irisTest.npy')


    # Plot accuracy curve for different dimensionality of the LDA
    getCRRCurve(train,test)
```

```
    # Draw a table for recognition results using different similarity measures
    a = getTable(train,test)




#main()
# We have run this function to get the training features and testing features␣
 ↪and saved them in .npy files.
# We will load the data directly when drawing the plots and calculating the␣
 ↪accuracy without running main() again.
```

[5]:
```
def runAllReduced():

    # Load train and test from data file saved before
    train = np.load('irisTrain.npy')
    test = np.load('irisTest.npy')
    irisY = np.arange(1,109) # training labels
    trainY = np.repeat(irisY,3*7)
    testY = np.repeat(irisY,4)

    # Plot accuracy curve for different dimensionality of the LDA
    getCRRCurve(train,test)

    # plot ROC curve
    ROC_curve(train, trainY, test, testY)
    # Draw a table for recognition results using different similarity measures
    a = getTable(train,test)


runAllReduced()
```

```
Currently computing dimention 50
Currently computing dimention 60
Currently computing dimention 70
Currently computing dimention 80
Currently computing dimention 90
Currently computing dimention 100
Currently computing dimention 107
```


Recognition results using features of different dimentionality under Cosine Similarity Measure
Correct recgnition rate
0.925
0.920
0.915
0.910
0.905
0.900
0.895
50
60
70
80
90
100
Dimensionality of the feature vector




```
Currently computing distance measure number 1
Currently computing distance measure number 2
Currently computing distance measure number 3
                          Original Feature Set  Reduced Feature Set
L1 distance measure                   0.865741             0.872685
L2 distance measure                   0.872685             0.872685
Cosine similarity measure             0.923611             0.925926
```

[ ]: