TP1 – Classification par kppv

Dans ce TP, nous allons utiliser une partie de la base de visages "Labeled Faces in the Wild" provenant de http://vis-www.cs.umass.edu/lfw/. Cette base contient 5749 personnes et 13233 images de taille 62 x 47 pixels. Certaines personnes ne sont représentées qu'une seule fois tandis que d'autres sont représentées très souvent (plus de 80 fois). Nous utiliserons ici seulement 7 personnes représentées 1288 fois.

I. Chargement des données

a. Charger les données

Chargez les données, puis affichez les en utilisant les fonctions plotHistoClasses() et plot_gallery() fournies.

```
from display import plotHistoClasses, plotGallery
[X, y, name]=np.load("TP1.npy",allow_pickle=True )
plotGallery(imgs)
plotHistoClasses(lbls)
```

Question

Sachant que X représente les features, y les labels et name le nom des classes, déterminez la taille des images, le nombre d'images et le nombre de classes.

Retrouvez l'identité des 12 personnes affichées. Est-ce que les classes sont équiprobables ? Retrouvez le nombre d'exemples par classe.

b. Partitionnement de la base d'apprentissage

Partitionnez la base en une base d'apprentissage et une base de test en mettant 25% des données en test (fonction train_test_split()) pour obtenir les variables X_train, X_test, y_train et y_test. Pour reproductibilité lors de la correction, la variable random_state correspondra aux trois derniers chiffres de votre numéro d'étudiant.

Question

Combien y a-t-il d'images en train et en test ? Quelles sont les dimensions des quatre variables X_train, X_test, y_train et y_test ?

II. Prétraitement des données

a. Redimensionnement des données

Pour réaliser une classification par kppv, on utilise un codage rétinien. Chaque image est donc représentée par un vecteur de caractéristiques de dimension $n = 2914$. Redimensionnez X_train et X_test de façon à ce qu'ils aient pour dimension N x n (np.reshape()) où N est le nombre d'exemples.

b. Mise en forme des données pour la classification

Mettez en forme les données (train et test) en utilisant la classe StandardScaler. On estimera la moyenne et l'écart-type de chaque dimension sur les données d'apprentissage, puis on transformera les données (train et test) en utilisant ces valeurs. Allez sur la documentation en ligne de StandardScaler pour voir quelle méthode de cette classe utiliser.

On fera une attention particulière sur les données que l'on utilise pour estimer la mise en forme des données et celles sur lesquelles on applique la mise en forme.

Question

A quoi consiste la mise en forme des données ? Comment sont-elles transformées ?

III. Classification par les KPPV

c. Classifieur 1PPV

Définissez le classifieur 1PPV en utilisant la classe KNeighborsClassifier(). On souhaite utiliser la distance euclidienne et le 1PPV.

Réalisez la classification des exemples de test en utilisant la méthode predict().

Affichez la matrice de confusion (fonction confusion_matrix()) et estimez le taux de reconnaissance à partir des éléments de cette matrice. Vérifiez que le taux est identique à celui renvoyé par la fonction accuracy_score().

Questions

Que représente la matrice de confusion ? Que vaut sa somme ? Que vaut la somme des lignes ? Est-ce que les classes sont équilibrées dans la base de test ?

d. Classifieur KPPV

Faites varier le K des KPPV et tracez l'évolution du taux de reconnaissance en fonction de K.

Questions

Conclusion ? Interprétez l'évolution des résultats en fonction de K

IV. Test sur une image du Web

En fixant K à la valeur obtenant les meilleures performances sur la base de test, réalisez toutes les étapes permettant de reconnaitre les personnes des images Bush.jpg, Powell.jpg et Blair.jpg. On pourra lire les images avec :

```
I = Image.open("Bush.jpg")
I = np.array(I)
I = rgb2gray(I)
```