CentraleSupélec

APPRENTISSAGE AUTOMATIQUE — MENTION IA — 2020/2021

Projet final

Gianluca QUERCINI, Yassine OUALI, Myriam TAMI

gianluca.quercini@centralesupelec.fr, yassine.ouali@centralesupelec.fr,
myriam.tami@centralesupelec.fr

1 PRÉLIMINAIRES

Ce projet a pour objectif la résolution d'un problème de reconnaissance faciale. Dans un premier temps, on utilisera le jeu de données nommé *Labeled Faces in the Wild* (LFW) [1]. Ce jeu de données consiste en une collection de photos de célébrités au format JPEG obtenues sur Internet. Chaque photo est centrée sur une seule face. Le jeu de données contient :

- 13233 photos
- 5749 personnes
- 1680 personnes avec plus de deux photos.

Dans ce projet, il s'agira d'appliquer les algorithmes de classification que vous avez appris pour entraîner des classifieurs à reconnaître les personnes dans ce jeu de données.

2 LES DONNÉES

Pour récupérer les données, la librairie `scikit-learn` met à disposition une fonction appelée `fetch_lfw_people`, dont la documentation est disponible à l'adresse suivante :

`https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_lfw_people.html`

Cette fonction permet de télécharger le jeu de données automatiquement. La valeur renvoyée par la fonction est un objet `dataset` avec les attributs suivants :

[1] `http://vis-www.cs.umass.edu/lfw/`

- `dataset.data`. Ce sont les attributs extraites des images. `dataset.data`[i] est le vecteur des attributs de la i-ème photo.
- `dataset.target`. Ce sont les identifiants numériques des classes associées aux photos. `dataset.target`[i] est l'identifiant de la classe (cad, de la personne) associée à la i-ème photo.
- `dataset.target_names`. Ce sont les noms des classes.

La fonction `fetch_lfw_people` permet de spécifier un certain nombre de paramètres :

- `min_faces_per_person`. Le jeu de données téléchargé contiendra seulement les photos des personnes qui ont au moins `min_faces_per_person` photos différentes. Par défaut, on ne met pas de limite.
- `color`. Permet de spécifier si on souhaite garder les informations sur la couleur des photos. Par défaut, la valeur de ce paramètre est `False`.
- `slice_`. Permet de couper une photo pour garder seulement la partie contenant la face. Cela permet d'éviter le bruit du fond de l'image. Par défaut, la valeur de ce paramètre est $(slice(70,195,None), slice(78,172,None))$ (valeurs qui identifient une face dans les photos de ce jeu de données).

3 TRAVAIL À FAIRE

Exercice 1

Votre premier objectif sera de faire une **analyse descriptive** du jeu de données. Cette analyse visera à déterminer des statistiques sur le jeu de données, telles que :

- Nombre d'instances par classe.
- Prise en compte des valeurs manquantes.
- Analyse en composantes principales.
- Analyse de l'échelle des valeurs pour chaque attribut.
- ...

Le but de cette étape est de vous familiariser avec le jeu de données et éventuellement vous donner des idées sur les méthodes à utiliser.

Vous n'oublierez pas à cette étape de diviser le jeu de données en deux pour en dériver un jeu de *training* et un jeu de *test*.

Exercice 2

Entraînez une SVM linéaire sur le jeu de données. Analysez l'erreur de prédiction.

- Faites varier le paramètre `min_faces_per_person` de 20 à 200. Quel est l'impact sur l'exactitude (*accuracy*) du classifieur ?
- Fixez le paramètre `min_faces_per_person` à 100. Faites varier la taille de l'ensemble d'entraînement et observez l'impact sur l'exactitude du classifieur.

Exercice 3

Entraînez une SVM linéaire sur le jeu de données en utilisant une analyse en composantes principales (ACP) pour réduire les dimensions.

- Quel est le nombre de dimensions qui garantit le meilleur résultat en termes d'exactitude ?
- Par rapport à l'exercice précédent, que pourriez-vous dire par rapport aux temps d'entraînement et de prédiction ?
- Chargez le jeu de données en spécifiant `slice_`=None et `resize`=1. Observez-vous un changement dans l'exactitude et les temps d'entraînement ?

Exercice 4

Sur la base des observations dérivées des exercices précédents, entrainez d'autres modèles avec les autres méthodes d'apprentissage que vous avez vu en cours, en particulier:

- Naive Bayes.
- SVM à noyau polynomial et RBF.
- Arbres de décisions.
- Méthodes ensemblistes (bagging, boosting, stacking, random forest).

Pour chaque méthodes, vous penserez à choisir les valeurs optimales des hyperparamètres. Comparez les résultats (précision, rappel f-mesure) de ces modèles. Discutez les résultats.

Exercice 5

Optionnel. Que se passe-t-il si vous essayez un algorithme de *clustering* (par ex., *k-means*) sur ce jeu de données. Arrivez-vous à trouver des résultats interprétables ?