Master Informatique Specialités EID², PLS

Traitement numérique des données

TP1
Introduction au module scikit-learn

Les TP sont notés. Les étudiants doivent sauvegarder le code de chaque exercice et doivent présenter le résultat au chargé du TP.

scikit-learn est un module Python entièrement consacré aux traitements de données. Les ensembles de données (ou d'objets) sont ici représentés par des vecteurs dans un espace de caractéristiques (des **variables** ou « **features** »). Les données sont stockées sous la forme d'une matrice de taille **m x k** ; où **m** sont les vecteurs lignes représentant les objets caractérisés par **k** variables. A cette matrice s'ajoute aussi un ensemble de **m** étiquettes (qui peuvent être des chaînes de caractères ou des nombres), une pour chaque donnée et un ensemble **k** de noms de variables (également des chaînes ou des nombres), un nom pour chaque variable. Les données avec la même étiquette appartiennent à la même classe.

Vous pouvez utiliser *help(nom de la fonction)* pour avoir une aide sur une fonction Python.

A. Importation des libraires

1- Importez **scikit-learn :**

Dans Python, la plupart des fonctions sont incluses dans des librairies, qu'il faut importer pour pourvoir les utiliser. Par exemple pour importer **scikit-learn** il faut écrire :

*from sklearn import **

2- Importez les librairies **numpy** (calcul scientifique) et **matplotlib.pyplot** (figures).

B. Manipulation d'un jeu de données

1- Chargez les données Iris avec la commande :

iris = datasets.load_iris()

La variable *iris* est un objet qui contient la matrice des données (*iris.data*), un vecteur de numéro de classe (*target*), ainsi que les noms des variables (*feature_names*) et le nom des classes (*target_names*).

2- Affichez les données, les noms des variables et le nom des classes (utilisez *print*).

3- Affichez le nom des classes pour chaque donnée

iris.data est une matrice, mais c'est aussi un objet, comme toute variable en Python, qui inclut des méthodes. Par exemple, *iris.data.mean(0)* donne la moyenne de chaque variable, et *iris.data.mean(1)* donne la moyenne de chaque donnée. Il existe aussi des attributs associé aux objets. Par exemple : i*ris.target.size* donne le nombre d'objets dans *target.*

4- Affichez la moyenne (*mean*), l'ecart-type (*std*), le *min* et le *max* pour chaque variable.

5- En utilisant les attributs *size* et *shape*, affichez le nombre de données, le nombre de variables et le nombre de classes.

C. Téléchargement et importation de données

Il est possible de charger un jeu de donnée directement depuis le site mldata.org qui contient de nombreux jeux de données gratuits, grâce à la fonction *datasets.fetch_mldata*.

1- Importez les données 'MNIST original'.

2- Affichez la matrice des données, le nombre de données et de variables, les numéros de classes pour chaque donnée, ainsi que la moyenne, l'écart type, les valeurs min et max pour chaque variable ; enfin donnez le nombre de classe avec la fonction *unique*.

D. Génération de données et affichage

1. Utiliser l'aide (*help*) pour voir comment utiliser la fonction *datasets.make_blobs.*
2. Générez 1000 données de deux variables réparties en 4 groupes.
3. Utilisez les fonctions *figure, scatter, title, xlim, ylim, xlabel, ylabel* et *show* pour afficher les données avec des couleurs correspondant aux classes. Les axes x et y seront dans l'intervalle [-15, 15] et devrons avoir un titre. La figure doit aussi avoir un titre.
4. Générez 100 données de deux variables réparties en 2 groupes, puis 500 données de deux variables réparties en 3 groupes. Concaténez (*vstack* et *hstack*) les deux jeux de données et les numéros de classe pour fusionner les deux jeux de données. Affichez les trois ensembles avec *scatter* comme précédemment.