TP12 – Classification par la méthode k-NN

Année 2022-2023

L'objectif de ce TP est de coder la méthode des k plus proches voisins et de l'appliquer à des données réelles. Ainsi vous travaillerez sur les données réelles de malades Parkinsoniens récupérées sur le site : `https://archive.ics.uci.edu/ml/index.php` (Machine Learning Repositery). Ces données sont déposées sur Moodle. Elles représentent des grandeurs extraites de signaux de parole enregistrés sur des patients parkinsoniens. Ce `Data Set` (fichier .csv) est décrit en détail dans le fichier `parkinsons.names`. Les données sont réelles, multivariées (22 attributs pour chaque exemple ou *sample* + la classe qui vaut 0 ou 1). Il y a au total 197 *samples*. Il est impératif de bien comprendre le contenu et la structuration de ce `Data Set`.
Attention : Vous n'utiliserez pas les fonctions existantes de la librairie Python `Scikit-Learn` pour la méthode k-NN.

1 Classification par la méthode des k plus proches voisins

Rappels – L'algorithme de classification dit "des Plus Proches Voisins" (k-nearest neighbors ou k-NN) s'applique sur des données d'entraînement. Pour un ensemble de `m` données d'entraînement, il est possible de partitionner l'espace selon l'appartenance des données aux classes. Par exemple, sur la figure 1, les points représentés en rouge (étoiles) appartiennent à la classe A et les points en vert (triangles) appartiennent à la classe B. Un nouveau point peut être classifié dans l'une des deux classes A ou B, en fonction de ses plus proches voisins (le nombre k constituant un paramètre de l'algorithme). Pour ce nouveau point qui n'appartient pas aux données d'entraînement, on détermine parmi les plus proches voisins quelle est la classe majoritaire. Par exemple, si $k = 3$, et que l'on trouve 3 plus proches voisins associés aux classes respectives (A, B, B), alors ce nouveau point sera classé B.

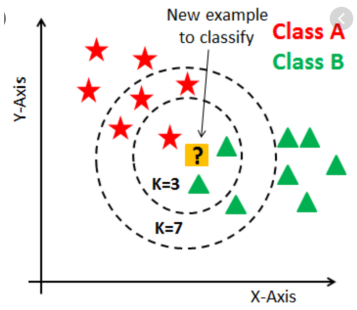

Figure 1: Exemple de classification par méthode k-NN dans un plan

1. Écrivez les fonctions et méthodes nécessaires pour télécharger les données et les stocker dans des vecteurs X et y. Vous pourrez utiliser des classes pour modéliser les données.
   Indications : Pour la lecture des données vous pourrez vous appuyer sur la librairie `csv`.

2. Écrivez une fonction `distance` qui permet de retourner la distance entre deux patients de l'ensemble d'apprentissage. Les données étant réelles, vous pourrez prendre la distance euclidienne.

3. Dans la classe `KNN`, écrivez la méthode `get_neighbors` qui prend en entrée un patient `p_test` et un entier `k` et retourne la liste des `k` patients les plus proches du patient `p_test`.

4. Dans la classe `KNN`, écrivez la méthode `predict_knn` qui prend en entrée un patient `p_test` et un entier `k` et retourne la classe prédite du patient `p_test`.

5. Effectuez des tests sur ce `DataSet` et proposez une évaluation de la méthode.

Indications

- Vous pourrez utiliser la librairie `metrics` qui permet de déterminer la matrice de confusion et les différentes mesures de performance vues en cours et dans le TP précedent.

  ```
  from sklearn.metrics import accuracy_score
  from sklearn.metrics import classification_report
  from sklearn.metrics import confusion_matrix
  ```

- Vous pourrez effectuer de la cross-validation sur k.
- Vous pourrez également proposer plusieurs sous-ensemble d'attributs pour caractériser les patients (en privilégiant certains paramètres par exemple, en normalisant, ...).