TP6 : Récap. classification supervisée

M1 IAA

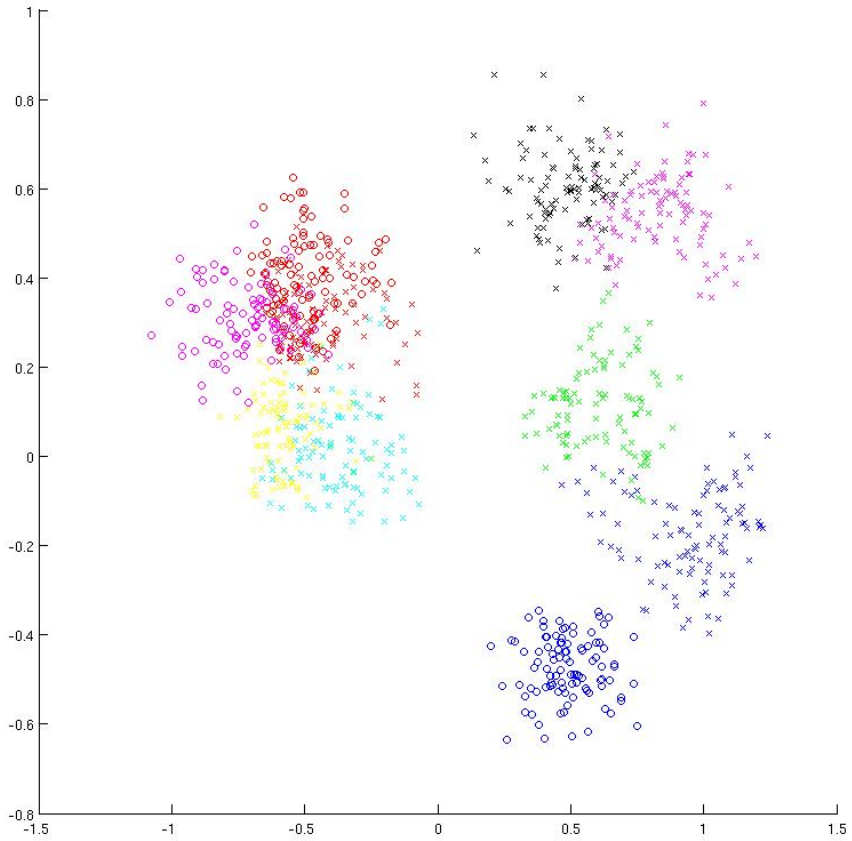
L'objectif est de comparer deux méthodes de classification supervisée sur un même jeu de données. Vous allez :

- Utiliser le module d'apprentissage automatique open-source `sklearn` pour expérimenter une nouvelle méthode sans l'implémenter,
- Adapter, tester et évaluer une méthode que vous avez implémentée dans les séances précédentes,
- Rédiger votre rapport sous la forme d'un mini-article de recherche,
- Attention, cette fois-ci nous vous demandons de travailler en binôme et de rédiger un compte-rendu et de déposer votre code par binôme. Des instructions à ce sujet sont données à la fin de ce document.

1 Jeu de données

Vous disposez pour réaliser vos expériences de signal audio qui a été enregistré en studio. Le locuteur prononce 10 voyelles. Il y a 100 occurrences de chaque voyelle. Une paramétrisation cepstrale a été extraite de ces échantillons aboutissant à 12 coefficients pour chaque échantillon. Une Analyse en Composantes Principales a été réalisée afin de réduire la dimension des données à 2 et à 3.

Vous disposez donc de trois fichiers `data2.csv`, `data3.csv`, `data12.csv` pour ces données en différentes dimensions arrangées de manière CSV standard, à savoir un échantillon par ligne, comportant en premier les paramètres acoustiques et la classe étant en fin de ligne. Tous les champs sont séparés par une virgule.



Jeu de 10 voyelles en 2-d avec 100 échantillons par voyelle

2 Scikit-learn

Le site officiel du logiciel Scikit-learn est :
`http://scikit-learn.org/stable/index.html`
La documentation en ligne est complète, et devra être consultée chaque fois que nécessaire :
`http://scikit-learn.org/stable/documentation.html`
Des tutoriaux sont disponibles a l'adresse suivante :
`http://scikit-learn.org/stable/tutorial/index.html`

Scikit-learn ou sklearn est un paquet Python dédié à l'apprentissage automatique. Il permet de :

- Charger un jeu de données intégré ou bien votre propre jeu de données facilement, générer des données synthétiques, avec le paquet `sklearn.dataset`
- Réaliser des pré-traitements sur les données avec le paquet `sklearn.preprocessing` : standardisation, ré-échelonnage, normalisation, transformations et encodage des labels (par ex., encodage multinomial à encode *one-hot*, etc.
- Construire des modèles sous la forme d'un *estimator* qui est un objet qui implémente la méthode `fit(X, y)` pour estimer un modèle et `predict(T)` afin d'utiliser ce modèle pour prendre des décisions i.e. faire des prédictions. Exemple : le k-plus proche voisin ou *k-nearest neighbor*.
- Évaluer vos modèles avec trois outils à choisir selon les besoins :
  - la méthode `score` de votre *estimator*, documentée dans la page d'aide de chaque *estimator*,

— le paquet `sklearn.model_selection` pour mettre en place facilement un protocole d'évaluation adapté, comme par exemple une validation croisée,
— le paquet `sklearn.metrics` qui mis à disposition de nombreuses métriques si celle de la méthode `score` n'est pas suffisante.

Nous vous proposons de tester tout d'abord un classifieur de type k plus proches voisins ou k-NN, fondé sur les voisins les plus proches de chaque point à classer, où k est une valeur entière spécifiée par l'utilisateur, i.e. un hyperparamètre.
Sous sklearn, il s'agit de l'estimator `KNeighborsClassifier`. Vous trouverez un exemple d'utilisation ici : `https://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html`

3 Consignes

1. Séparer vos données en sous-ensembles disjoints : les données d'apprentissage et les données de test. Prendre pour commencer 80 % des fichiers pour l'apprentissage et 20 % pour les tests.
2. Mettre en place une classification bayésienne avec Gaussienne avec le code réalisé dans les TPs précédents
3. Mettre en place une classification k-NN à l'aide de `sklearn`. Vous utiliserez l'option de poids uniformes pour cette méthode.
4. Évaluer vos systèmes : produire une matrice de confusion et un taux de reconnaissance pour chacune des deux méthodes précédentes.
5. Quels sont avantages/inconvénients des deux méthodes, en prenant par exemple comme éléments de comparaison : temps de calcul, pré-requis sur les données, efficacité ?
6. Faire varier le nombre de données dans les sous-ensembles et comparer les résultats. Vous pouvez produire des courbes sur le taux d'erreur en fonction de la taille des sous-ensembles.
7. Si l'on souhaite utiliser un maximum de données pour réaliser l'apprentissage, il est possible d'effectuer une évaluation par validation croisée : quel est le principe de cette méthode ? Sous quelles appellations anglaises peut-on trouver cette technique ?
8. Reprendre la classification supervisée en utilisant les données de dimension supérieure `data3.csv` et `data12.csv`. Est-ce que les performances diffèrent en fonction de la dimension ?

4 Compte-rendu sous forme d'article scientifique

Dans le dépôt prévu pour ce TP et pour votre groupe, chaque binôme doit fournir une seule archive ZIP par binôme contenant le rapport de TP au format pdf et le code python.
Le nom des fichiers et de l'archive doivent comporter le nom des deux personnes composant le binôme.
Le rapport devra être présenté sous une forme particulière : il ne s'agit pas d'un compte rendu classique comme aux séances précédentes, il s'agit d'écrire un article scientifique de 4 pages maximum en français en utilisant le format d'article RFIA fourni.
L'objectif de cet article est de présenter une introduction du sujet du TP, une description succincte des données et des méthodes (par exemple un paragraphe par méthode, k-NN et Gaussienne), vos résultats en terme de performance ainsi qu'une analyse personnelle de vos résultats pour conclure. Veillez à donner tous les détails nécessaires pour qu'une tierce personne puisse reproduire les résultats. Des images d'illustration sont toujours bienvenues également.