TP2 : K plus proches voisins

Exercice 1 :

Nous allons utiliser pour cet exercice la base de données d'Iris. Cette base contient des Iris qu'un botaniste, Ronald Fisher, a classé en 1936 à l'aide d'une clef d'identification des plantes (type de pétales, sépale, type des feuilles, forme des feuilles, ...). Puis, pour chaque fleur classée il a mesuré les longueurs et largeurs des sépales et pétales.

L'objectif de cet exercice est d'utiliser l'algorithme des k plus proches voisins afin de déterminer automatiquement l'espèce d'une nouvelle plante en fonction de la mesure des dimensions de ses sépales et pétales.

Chargement et analyse de la base

```
> data <- read.csv("iris.csv", header=TRUE)
> head(data)
> summary(data)
> str(data)
```

Analyser votre data en indiquant sa taille (nombre d'exemples), le nombre d'attributs et le nom de l'attribut cible ? Quel est le type de data$Species ?

```
> table(data$Species)
```

Que retroune la fonction table appliquée sur data$Species ?

On voudrait construire un nouveau dataframe "dataR" en mélangeant aléatoirement les exemples. Pour cela, générer un vecteur v de n nombres aléatoires tel que n est le nombre total d'exemples puis utiliser la fonction order pour réordonner les lignes de "data" selon les valeurs de v. Le résultat sera affecté à "dataR".

Analyser le data frame "dataR".

Normaliser

Nous avons besoin de normaliser les valeurs d'attributs pour qu'ils aient le même ordre de grandeur. Créer une fonction *normalize* qui retourne le vecteur passé en paramètre normalisé.

$$v = \frac{v - min(v)}{max(v) - min(v)}$$

Appliquer cette fonction sur les 4 premières colonnes de "dataR" (utilisez la fonction lapply) et affecter le résultat à $dataN$.

Apprentissage avec KNN

Afin d'appliquer l'algorithme des K plus proches voisins (KNN) . Il faudrait :

- construire un ensemble d'apprentissage "$dataTrain$" avec les 100 premiers exemples des données normalisées.
- construire un vecteur "$labelTrain$" avec les étiquettes de ces 100 premiers exemples.
- construire un ensemble de test "$dataTest$" avec les 50 exemples restants
- construire un vecteur "$labelTest$" avec les étiquettes des 50 exemples restants
- utiliser la librairie class
- Appliquer la fonction knn

```
> dataTrain <-  dataN[c(1:100),]
> dataTest <-  dataN[c(101:150),]

> labelTrain <-  dataR[c(1:100),c(6)]

> library(class)

> model <- knn(train=dataTrain, test=dataTest,
    cl = labelTrain,k=3);
```

- Analyser l'objet *model* retourné
- Afficher la matrice de confusion *conf* qui sert à mesurer la qualité d'un système de classification. Que contient cette matrice ?

```
> table(model)
> table(labelTest)
> conf<-table(model,labelTest)
> conf
```

— Calculer le taux d'erreur de l'algorithme KNN
— Faites varier k et analyser son impact

Exercice 2 : Reconnaissance de caractères manuscrits

Nous allons utiliser pour cet exercice un dataset très célèbre appelé MNIST. Il est constitué d'un ensemble de 70000 images 28x28 pixels en noir et blanc étiqueté du chiffre correspondant (entre 0 et 9). Ce dataset utilise des données réelles qui ont déjà été pré-traitées.

Notre objectif sera donc d'entraîner un modèle qui sera capable de reconnaître les chiffres écrits sur ce type d'images.

Les données sont constituées d'un échantillon d'apprentissage "mnist_train.csv" : 60000 images de 754 pixels à niveau de gris et d'un échantillon test "mnist_test.csv" de 10000 images

— Construire un data frame *Dtrain* à partir du fichier "mnist_train.csv"
— Utilisez la fonction dim pour afficher les dimensions de votre data frame *Dtrain*
— Quelle est la dimension de votre problème d'apprentissage
— Quelle est la colonne qui contient l'étiquette des caractères (attribut cible) ?

Afin de visualiser la première image de l'ensemble de test, on utilise le code suivant :

```
im<-matrix(Dtrain[1,-1], nrow=28, ncol=28)
im_numbers <- apply(im, 2, as.numeric)
image(1:28, 1:28, im_numbers, col=gray((0:255)/255))
```

— Visualisez les 5 premieres images. Que constatez vous ?
— Modifiez le code pour pouvoir afficher l'image correctement

Les données ont déjà été normalisées centrées et sont complètes. On va extraire un sous échantillon de l'ensemble d'apprentissage car le temps d'exécution sur les 600000 exemples sera trop long. Utilisez la fonction sample pour construire un échantillon de 10000 images.

```
ech <-sample(1:nrow(Dtrain),10000)
SubDtrain <- Dtrain[ech,c(2:785)]
SubLabelTrain <- as.factor(Dtrain[ech, 1])
```

— Créer de la même manière un échantillon de test avec 1000 images à partir de $Dtest$
— Appliquer le modèle KNN avec $k = 2$
— Afficher la matrice de confusion
— Calculer le taux d'erreur sur l'échantillon de test
— Quels sont les indices des images mal classées ? Visualiser les et afficher à chaque fois la prédiction du modèle KNN ainsi que l'étiquette réelle du caractère.