TP ozone : analyses discriminantes linéaire et quadratique, k plus proches voisins

Résumé

Prévision du pic d'ozone par analyse discriminante et k plus proches voisins.

1 Objectif

L'objectif est de comparer les trois méthodes d'analyses discriminantes disponibles dans R : `lda` paramétrique linéaire (homoscédasticité), `lqa` paramétrique quadratique (hétéroscédasticité) sous hypothèse gaussienne et celle non-paramétrique des k plus proches voisins.

Attention, ces techniques n'acceptent par principe que des variables explicatives ou prédictives quantitatives. Néanmoins, une variable qualitative à deux modalités, par exemple le type de jour, peut être considérée comme quantitative sous la forme d'une fonction indicatrice prenant ses valeurs dans $\{0, 1\}$ et, de façon plus "abusive", une variable ordinale est considérée comme "réelle". Dans ce dernier cas, il ne faut pas tenter d'interpréter les fonctions de discrimination, juste considérer des erreurs de prévision. La variable `Station` n'est pas prise en compte.

La bibliothèque standard de R (MASS) pour l'analyse discriminante ne propose pas de procédure automatique de choix de variable contrairement à la procédure `stepdisc` de SAS mais, dans cet exemple, les variables sont peu nombreuses.

2 Estimations

```
library(MASS) # chargement des librairies
library(class) # pour kNN
```

```
# analyse discriminante linéaire
disc.lda=lda(DepSeuil~.,data=datappq[,-4])
# analyse discriminante quadratique
disc.qda=qda(DepSeuil~.,data=datappq[,-4])
# k plus proches voisins
disc.knn=knn(datappq[,c(-4,-10)],datappq[,c(-4,-10)],
   datappq$DepSeuil,k=10)

# erreur apparente de prévision
table(datappq[,"DepSeuil"],
    predict(disc.lda,datappq)$class)
table(datappq[,"DepSeuil"],
    predict(disc.qda,datappq)$class)
table(datappq[,"DepSeuil"],disc.knn)
```

Noter le manque d'homogénéité des commandes de R issues de librairies différentes. L'indice de colonne négatif (-10) permet de retirer la colonne contenant la variable à prédire de type facteur. Celle-ci est mentionnée en troisième paramètre pour les données d'apprentissage.

Estimatin de l'erreur par validation croisée

Pour `knn`, le choix du nombre de voisins k peut être optimisé par validation croisée mais la procédure proposée par la bibliothèque `class` est celle *leave-one-out*, donc trop coûteuse en calcul pour des gros fichiers. Il serait simple de la programmer mais une autre bibliothèque (`e1071`) propose déjà une batterie de fonctions de validation croisée pour de nombreuses techniques de discrimination.

```
# erreur par validation croisée
# analyse discriminante linéaire
disc.lda=lda(DepSeuil~.,data=datappq[,-4],CV=T)
# analyse discriminante quadratique
disc.qda=qda(DepSeuil~.,data=datappq[,-4],CV=T)

# estimer le taux d'erreur à partir des
# matrices de confusion
table(datappq[,"DepSeuil"],disc.lda$class)
```

```
table(datappq[,"DepSeuil"],disc.qda$class)

# k plus proches voisins
library(e1071)
plot(tune.knn(as.matrix(datappq[,c(-4,-10)]),
   as.factor(datappq[,10]),k=2:20))
```

Remarquer que chaque exécution de la commande précédente donne des résultats différents donc très instables. Faire plusieurs exécutions et déterminer une valeurs “raisonnables” de k.

Comparer le taux d’erreur minima obtenu avec ceux, toujours estimés par validation croisée, des autres versions d’analyse dicriminante.

3 Erreur sur l’échantillon test

Les commandes suivantes calculent la matrice de confusion pour la "meilleure" méthode d’analyse discriminante au sens de la validation croisée.

```
# analyse discriminante linéaire
disc.lda=lda(DepSeuil~.,data=datappq[,-4])
table(datestq[,"DepSeuil"],
    predict(disc.lda,datestq[,-4])$class)
```

A titre indicatif, voici l’estimation de l’erreur sur l’échantillon test pour la méthode des k plus proches voisins.

```
disc.knn=knn(as.matrix(datappq[,c(-4,-10)]),
    as.matrix(datestq[,c(-4,-10)]),
    datappq$DepSeuil,k=15)
table(disc.knn,datestq$DepSeuil)
```

Noter le meilleur taux d’erreur en vue d’une comparaison.

4 Comparaison des courbes ROC

```
library(ROCR)
ROCdiscrim=predict(disc.lda,
   datestq[,c(-4)])$posterior[,2]
preddiscrim=prediction(ROCdiscrim,datestq$DepSeuil)
perfdiscrim=performance(preddiscrim,"tpr","fpr")
# tracer les courbes ROC en les superposant
# pour mieux comparer
plot(perflogit,col=1)
plot(perfdiscrim,col=2,add=TRUE)
```

Une méthode de prévisoin d’occurence de dépassement du pic de pollution est-elle globalement meilleure ?