Université de Carthage
Ecole Supérieure de la Statistique et de l'Analyse de l'Information

Examen de Data Mining

3 ème année du cycle de formation d'ingénieurs

Durée de l'épreuve : 1 heure 30 - Documents non autorisés
Nombre de pages : 3 - Date de l'épreuve : 6 janvier 2022

Exercice 1 : On considère le tableau de données ci-dessous contenant les valeurs observées de deux variables quantitatives x^1 et x^2, et d'une variable qualitative y possédant les deux modalités A et B, sur un échantillon I de huit individus notés $i_1, \ldots, i_8$. Par la suite, on cherche à expliquer y en fonction de x^1 et x^2.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
X^1	4	3	1	0	4	3	5	4
X^2	5	4	2	1	4	3	3	2
Y	A	A	A	A	B	B	B	B

Par la suite, on applique différentes méthodes de classification supervisèe à ces données afin d'expliquer Y en fonction de X^1 et X^2. Pour cela, on utilise les commandes du logiciel R.

A- Analyse factorielle discriminante (AFD)

1- Calculer le centre de gravité g ainsi que les centres de gravité des classes A et B.

On effectue l'AFD linéaire du tableau de données avec la commande `lda`.
2- Expliquer pourquoi il n'existe qu'un seul axe factoriel discriminant non trivial.
3- Sachant que le facteur discriminant a pour coordonnées :

```
X1 1.279204
X2 -1.066004
```

Compléter la liste suivante qui indique les scores de chaque individu (un score manquant étant signalé par un " ?") :

```
I1 -0.8528029        I5 0.2132007
I2 ?                 I6 ?
I3 -1.4924050        I7 2.5584086
I4 -1.7056057        I8 2.3452079
```

4- Indiquer les scores des 2 centres de gravité.
5- Déterminer la classe d'affectation de chacun des individus et en déduire le taux de bien classés.

6- En notant "don" le data.frame dans lequel sont enregistrées les données, écrire un code R qui effectue une AFD linéaire avec une validation croisée dont l'échantillon d'apprentissage contient 80% des données.

B- Arbre de décision (AD)

On applique la commande : `mod2 <- rpart(Y∼., don, minsplit=?)`
Le résultat est un arbre de décision que l'on peut décrire de la façon suivante où chaque noeud est identifié par une valeur de m :

- La racine, noeud $m = 1$, est divisée en $\{X^1 < 2\}$ $(m = 2)$ et $\{X^1 \geq 2\}$ $(m = 3)$.

- Le noeud $m = 3$ est divisé en $\{X^2 \geq 3\}$ $(m = 4)$ et $\{X^2 < 3\}$ $(m = 5)$.

7- Dessiner l'arbre de décision ainsi défini. Pour chaque noeud non terminal, on indiquera la coupure utilisée pour diviser ce noeud. Pour chaque noeud terminal, on indiquera les individus appartenant à ce noeud et la classe d'affectation des individus qui appartiennent à ce noeud.

8- Déterminer la plus grande valeur de `minsplit` à utiliser afin d'obtenir cet arbre.

9- Sachant que la commande `rpart` utilise l'indice de Gini pour construire l'arbre, quelle est la qualité de la coupure (appelée aussi réduction de l'impureté) effectuée pour diviser le noeud $m = 1$?

Exercice 2 : On voudrait expliquer une variable catégorielle `y` par un tableau de variables explicatives `X` en utilisant la méthode Random forest sous Python. On voudrait effectuer le choix des meilleures paramètres de cette méthode à l'aide de la fonction `GridSearchCV` de Python. Compléter le code suivant en indiquant pour chacune des 15 lettres entre parentèses le numéro de l'expression donnée dans la liste ci-dessous qui devra la remplacer.

```
### Début du code Python  ###
(a) , (b)  = train_test_split((c), (d) , (e) , (f) )

rfc=RandomForestClassifier( random_state=42 )

param_grid = {
    (g) ,
    (h) ,
    (i) ,
    (j)
}

grid_search = GridSearchCV(estimator= (k) , param_grid , (l) , scoring= (m))
(n).fit( (o) )

### Fin du code Python  ###
```

Liste des 20 expressions proposées pour compléter le code Python.

1. X_train,X_test,
2. y_train,y_test
3. X
4. y
5. test_size=0.2
6. random_state = 42
7. 'n_estimators': [200, 500]
8. 'max_features': ['auto', 'sqrt', 'log2']
9. 'max_depth' : [4,5,6,7,8]
10. 'criterion' :['gini', 'entropy']
11. rfc
12. cv= 5
13. 'accuracy'
14. grid_search
15. X_train,y_train
16. GridSearchCV
17. 'C': [0.1,1, 10, 100],
18. 'gamma': [1,0.1,0.01,0.001]
19. 'fit_intercept': [True, False]
20. 'penalty' : ['none', 'l1', 'l2', 'elasticnet']