Apprentissage Supervisée

Outils : Logiciel R, Environnement de Développement Intégré RStudio

Ce cas d'utilisation vise à utiliser un algorithme basé sur l'approche des arbres de décision pour une application de prédiction d'appétence de clients.

L'objectif est de construire un modèle de prédiction de la variable de classe et de l'appliquer ensuite à de nouveaux clients (prospects) afin d'évaluer leur propension à acheter l'article (prédiction d'appétence).

Rappel : la liste des instructions exécutées lors de la précédente session est stockée dans le fichier `.Rhistory` automatiquement créé dans le répertoire de travail lorsque vous quittez l'environnement R. Les instructions `help(cmd)` et `? cmd` permettent d'afficher l'aide sur la commande `cmd`.

1. Ensemble de données *Achat*

L'ensemble de données `Data Achat.csv` contient des données sur l'achat d'un article qui a été proposé à des clients. La variable *Achat* indique pour chaque client s'il a ou non acheté l'article.

Caractéristiques de l'ensemble de données :

- Instances : 600 clients
- Nombre de variables : 12
- Valeurs manquantes : aucune
- Séparateur de colonnes : virgule
- Séparateur de décimales : point

Dictionnaire des données

Variable	Type	Description	Domaine de valeurs
ID	Entier	Numéro identifiant du client	[12101, 12400]
Age	Entier	Age en années	[18, 67]
Sexe	Catégoriel	Sexe	Homme, Femme
Habitat	Catégoriel	Type d'habitat	Centre_Ville, Petite_Ville, Rural, Banlieue
Revenus	Entier	Revenus annuels en dollars US	[60392, 505040]
Marie	Booléen	Statut marital	Oui, Non
Enfants	Entier	Nombres d'enfants	[0, 3]
Voiture	Booléen	Possède une voiture	Oui, Non
Compte_Epargne	Booléen	Possède un compte épargne	Oui, Non
Compte_Courant	Booléen	Possède un compte courant	Oui, Non
Emprunt	Booléen	Emprunt en cours	Oui, Non
Achat	Booléen	Client acquéreur de l'article Variable de classe	Oui, Non

Nous allons utiliser les techniques de classification supervisée sur cet ensemble de données afin de générer un modèle de prédiction d'appétence, c-à-d permettant de prédire si un nouveau client sera fortement susceptible ou non d'acheter l'article en fonction de ses caractéristiques socio-démographiques (*Age*, *Revenus*, etc.).

La variable *Produit* constituera donc la variable de classe, c-à-d celle dont on veut prédire la valeur ; les deux classes sont donc *Produit=Oui* et *Produit=Non*.

La variable *ID*, qui est l'identifiant unique de chaque exemple, sera ignorée durant l'apprentissage.

Les 10 autres variables seront les variables prédictives, i.e. celles dont l'algorithme analysera les cooccurrences de valeurs afin d'apprendre le modèle de prédiction de la classe.

2. Chargement des données

- Téléchargez l'ensemble de données `Data Produit.csv` et définissez le répertoire de travail comme le dossier contenant le fichier téléchargé.
- Chargez les données du fichier `Data Produit.csv` dans un data frame `achat` par la commande :

```
> achat <- read.csv("Data Achat.csv", header = TRUE, sep = ",", dec = ".",
                    stringsAsFactors = TRUE)
```

- Identifiez dans l'aide de la fonction `read.csv()` à quoi correspondent les paramètres `header`, `sep` et `dec`.
- Vérifiez le chargement des données dans le data frame `achat` en affichant la liste des variables et leur type (appelé *mode* en R) à l'aide de la fonction `str()` par la commande :

```
> str(achat)
```

- Affichez la distribution des deux classes *Achat=Oui* et *Achat=Non dans le* data frame `achat` par la commande :

```
> table(achat$Achat)
```

L'ensemble d'apprentissage `achat_EA` sur lequel sera construit le classifieur sera constitué des deux tiers de l'ensemble de données.

- Créez un data frame `achat_EA` constitué des deux premiers tiers du data frame `achat`. Utilisez pour cela l'opérateur de sélection *`nom_data_frame[lignes, colonnes]`* en définissant le paramètre *`lignes`* pour sélectionner les 400 premières lignes, par la commande :

```
> achat_EA <- achat[1:400,]
```

L'ensemble de test `achat_ET` sur lequel seront testés les classifieurs sera constitué du dernier tiers de l'ensemble de données `Data_Achat.csv`.

- Créez un data frame `achat_ET` constitué de 200 dernières lignes c-à-d des lignes 401 à 600, du data frame `achat`.

Le numéro identifiant les clients ne constituant pas une information utile pour l'objectif poursuivi, la variable correspondante `ID` doit être supprimée de l'ensemble d'apprentissage afin qu'elle ne soit pas utilisée durant l'apprentissage de l'arbre de décision.

- Supprimez la variable `ID` du data frame `achat_EA` par l'une de ces deux méthodes :
  - Soit en utilisant l'opérateur de sélection *`data_frame[lignes, colonnes]`* et en indiquant par un paramètre *`colonnes`* négatif la suppression de la première colonne :

```
> achat_EA <- achat_EA[,-1]
```

  - Soit en utilisant la fonction `subset()` et en référençant la variable par son nom dans la sélection :

```
> achat_EA <- subset(achat_EA, select = -ID)
```

- Utilisez la fonction `summary()` afin d'afficher les caractéristiques principales des data frames `achat_EA` et `achat_ET`.
- Observez la répartition des classes *Achat=Oui* et *Achat=Non* dans les deux ensembles à l'aide de la commande `table()`.

3. Apprentissage d'un arbre de décision *rpart*

La librairie *rpart* (*Recursive Partitioning and Regression Trees*) contient des fonctions pour la construction de modèles de classification supervisée et régression.

- Installez la librairie *rpart* dans R par la commande :

```
> install.packages("rpart")
```

- Activez la librairie *rpart* dans votre session R par la commande :

```
> library("rpart")
```

Note : Afin d'installer et activer une librairie, vous pouvez utiliser :

- Soit les fonctions `install.packages("`*`nom_librairie`*`")` puis `library(`*`nom_librairie`*`)` en ligne de commande.
- Soit l'interface de RStudio (menu *Tools* pour installer et l'onglet *Packages* pour activer).

Si lors de l'installation d'une librairie vous obtenez un message d'erreur « `Installation of package`

'*nom_librairie*' had **non zero exit status** », faites une mise à jour de R à partir de RStudio ou en ligne de commandes en :

- Installant et activant la librairie *installr*.
- Exécutant la fonction `updateR()` de cette librairie.

☛ Affichez la documentation de la librairie *rpart* en allant sur l'onglet *Packages* dans la zone bas-droite de RStudio et en cliquant sur le nom de la librairie.

Nous allons utiliser la fonction `rpart()` de création d'arbres de décision pour la prédiction de variable de classe.

☛ Affichez l'aide de la fonction `rpart()` en cliquant sur le lien *rpart* dans la documentation de la librairie.

Nous allons créer un arbre de décision `rpart()` avec les paramètres par défaut.

☛ Construisez un arbre de décision `tree1` pour la prédiction de la variable `Achat` à partir du data frame `achat_EA` par la commande :

```
> tree1 <- rpart(Achat ~ ., achat_EA)
```

Note : Le paramètre « `Achat ~ .` » indique que la variable à prédire est *Achat* et le terme « **.** » que toutes les autres variables du data frame (*Age*, *Sexe*, …, *Emprunt*) sont les variables prédictives.

Il est également possible d'indiquer explicitement la liste des variables prédictives à utiliser par le paramètre « `Achat ~ Age + Sexe + … + Emprunt` ».

4. Représentation graphique de l'arbre

La librairie *rpart* fournit les fonctions `plot()` et `text()` qui permettent respectivement de dessiner la structure (nœuds et arcs) d'un arbre de décision `rpart()` et d'ajouter des informations textuelles (tests de variables et classes prédites) sur la structure dessinée.

☛ Tracez la structure de l'arbre de décision `tree1` par la commande :

```
> plot(tree1)
```

☛ Ajoutez le texte, avec les libellés pour les variables catégorielles, par la commande :

```
> text(tree1, pretty=0)
```

☛ Cliquez sur le bouton *Zoom* afin d'ouvrir une fenêtre d'affichage adaptée à la taille de l'arbre.

5. Évaluation des performances de l'arbre

L'évaluation d'un arbre de décision consiste à l'appliquer à un ensemble de test, et comparer la prédiction faite par l'arbre avec la « classe réelle », c-à-d la valeur de la variable de classe dans l'ensemble de test.

L'ensemble de test doit être un ensemble de données distinct de l'ensemble d'apprentissage (autres exemples) mais contenant les mêmes variables, y compris la variable de classe.

Les prédictions identiques à la valeur réelle constitueront les succès du classifieur et les prédictions différentes de la valeur réelle constitueront les échecs.

Nous allons utiliser pour tester l'arbre `tree1` la fonction générique `predict()` qui permet l'application d'un classifieur à un ensemble de données.

Afin de générer la classe prédite pour chaque exemple avec un arbre créé par la fonction `rpart()`, le paramètre à utiliser est `type="class"`.

La commande à utiliser pour appliquer le classifieur *nom_arbre* sur l'ensemble de test *data_frame* et stocker le résultat dans un vecteur *test_arbre* est donc de la forme :

```
> test_arbre <- predict(nom_arbre, data_frame, type = "class")
```

☛ Appliquez le classifieur `tree1` à l'ensemble de test `achat_ET` en stockant le résultat dans un objet nommé `test_tree1` par la commande :

```
> test_tree1 <- predict(tree1, achat_ET, type="class")
```

Note : le paramètre `type` permet de définir le type de résultat généré pour chaque exemple de test. (valeur de la classe pour `type="class"` ou probabilité pour chaque classe pour `type="prob"`).

☛ Affichez le contenu du vecteur `test_tree1` généré par la commande :

```
> test_tree1
```

☛ Affichez le nombre de prédictions dans chaque classe pour le résultat `test_tree1` par la commande :

```
> table(test_tree1)
```

- Afin de comparer ces prédictions et la classe réelle (valeur de la colonne `Achat`), ajoutez le vecteur `test_tree1` contenant les prédictions au data frame `achat_ET` comme une nouvelle colonne nommée `Prediction` par la commande :

```
> achat_ET$Prediction <- test_tree1
```

- Affichez le data frame `achat` par la commande :

```
> View(achat)
```

Rappel : la sélection d'exemples dans un data frame *`data_frame`* se fait à l'aide d'une instruction de la forme *`data_frame[selecteur_lignes, ]`* qui sélectionne tous les exemples qui vérifient la condition définie dans *`selecteur_lignes`*.

- Affichez la liste des exemples correctement prédits, c'est-à-dire pour lesquels la classe réelle (colonne `Achat`) et la prédiction (colonne `Prediction`) sont identiques, par la commande :

```
> achat_ET[achat_ET$Achat==achat_ET$Prediction, ]
```

- Calculez le nombre de succès `nbr_succes` en comptant le nombre d'exemples de test correctement prédits avec la fonction `nrow()` de comptage du nombre de lignes d'un data frame par la commande :

```
> nbr_succes <- nrow(achat_ET[achat_ET$Achat==achat_ET$Prediction,])
```

- Calculez le taux de succès (appelé *Classification Accuracy*) en divisant le nombre de succès par le nombre total d'exemples de test dans `achat_ET` par la commande :

```
> taux_succes <- nbr_succes/nrow(achat_ET)
```

- Calculez le nombre d'échecs `nbr_echecs` en comptant le nombre d'exemples de test incorrectement prédits avec la fonction `nrow()`.
- Calculez le taux d'échecs (appelé *Error Rate*) en divisant le nombre d'échecs par le nombre total d'exemples de test dans `achat_ET`.

6. Application de l'arbre pour la prédiction

Nous allons maintenant appliquer l'arbre de décision `tree1` pour prédire la classe des exemples d'un nouvel ensemble de données `Data Achat Prospects.csv` concernant de nouveaux clients dont la classe est inconnue. Cet ensemble de données ne contient donc pas de variable `Achat`.

- Chargez les données du fichier `Data Achat Prospects.csv` dans un data frame `achat_pro` par la commande :

```
> achat_pro <- read.csv("Data Achat Prospects.csv", header = TRUE, sep = ",",
  dec = ".")
```

- Vérifiez le chargement des données dans le data frame `achat_pro` en affichant la liste des variables et leur type (appelé *mode* en R) à l'aide de la fonction `str()`.
- Appliquez le classifieur `tree1` à l'ensemble à prédire `achat_pro` à l'aide de la fonction `predict()` en stockant le résultat dans un objet `pred_tree1` par la commande :

```
> pred_tree1 <- predict(tree1, achat_pro, type="class")
```

- Affichez le nombre de prédictions pour chacune des deux classes à l'aide de la fonction `table()` par la commande :

```
> table(pred_tree1)
```

- Ajouter au data frame `achat_pro` une nouvelle colonne `Prediction` contenant la prédiction (classe `Oui` ou classe `Non`) faite pour chaque exemple prospect dans `pred_tree1` par la commande :

```
> achat_pro$Prediction <- pred_tree1
```

- Créez un data frame `achat_pro_oui` en sélectionnant dans `achat_pro` les exemples prédits dans la classe *Achat=Oui* par la commande :

```
> achat_pro_oui <- achat_pro[achat_pro$Prediction=="Oui",]
```

- Créez un data frame `achat_pro_non` en sélectionnant dans `achat_pro` les exemples prédits dans la classe *Achat=Non*.