Examen (3 h)

Le premier exercice est à faire sur feuille, tandis que le deuxième exercice est à faire sur machine. Vous me remettrez d'une part votre copie pour l'exercice 1 et vous m'enverrez par email les réponses de l'exercice 2. L'exercice 2 peut être rédigé en Rmarkdown (pdf ou html) ou tout simplement en word avec les figures incluses dans le document (les codes peuvent également être inclus ou alors mis dans un fichier R à part). Si l'exercice 2 est rédigé en word, pensez à le convertir en pdf avant de me l'envoyer.

Exercice 1.
On considère des données d'apprentissage sur des patientes ayant eu un cancer du sein. Le but est de prédire l'apparition d'une récurrence du cancer ("oui" ou "non"), en fonction de l'âge ("<=50" ou ">50"), du statut de la ménopause ("pré-ménopause" ou "post-ménopause") et de la taille de la tumeur initiale ("<=35" ou ">35" en cm). Les données sont issues de la plateforme UCI, "Breast cancer data set", et ne contiennent qu'un sous échantillon de 10 patientes.

i	Récurrence (Y_i)	Âge (X_{i1})	Ménopause (X_{i2})	Taille de la tumeur (X_{i3})
1	non	<=50	pré-ménopause	<=35
2	non	<=50	pré-ménopause	<=35
3	non	<=50	pré-ménopause	<=35
4	non	>50	post-ménopause	<=35
5	non	<=50	pré-ménopause	<=35
6	non	>50	post-ménopause	<=35
7	oui	>50	pré-ménopause	<=35
8	oui	<=50	pré-ménopause	>35
9	oui	>50	post-ménopause	>35
10	oui	>50	pré-ménopause	<=35

Le but de l'exercice est d'obtenir une classification pour une patiente présentant les caractéristiques "age>50", "ménopause=pré-ménopause" et "tumeur initiale<=35", en utilisant différents algorithmes.

Algorithme CART

1. On a obtenu l'arbre complet T^c, sous R, présenté dans la Figure 1. Remplissez les cases blanches permettant de préciser l'étiquette prédite dans chaque feuille de l'arbre T^c. Que signifient les valeurs 0.00 et 40% dans la feuille terminale en bas à gauche de T^c ?

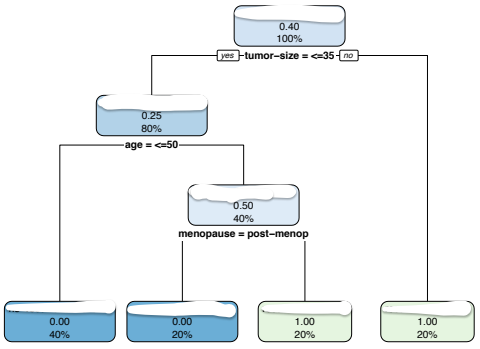

FIGURE 1 – Arbre de classification maximal T^c. Tumor-size représente la taille de la tumeur.

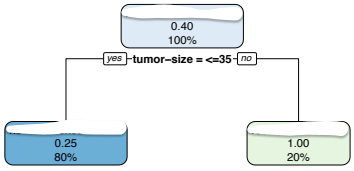

FIGURE 2 – Sous-arbre de classification $\tilde{T}$.

2. Expliquez, par le calcul, pourquoi l'algorithme CART préfère effectuer un premier découpage de l'arbre selon la variable "taille de la tumeur" plutôt que selon la variable "âge".

3. On considère le sous-arbre $\tilde{T}$ de la Figure 2. Comparez le risque de classification de l'arbre complet T^c avec celui de $\tilde{T}$.
4. Pour un paramètre $\alpha > 0$ de pénalisation, un arbre T appartenant à l'ensemble des sous-arbres de T^c, on considère alors le critère pénalisé suivant

$$\mathrm{Crit}_\alpha(T) = R(T) + \alpha|T|,$$

où $R(T)$ représente le risque de classification de T et $|T|$ représente la complexité de T, mesurée par son nombre de nœuds terminaux. Pour quelles valeurs de α, le sous-arbre $\tilde{T}$ présente-il un risque pénalisé Crit_α inférieur à celui de l'arbre maximal T^c ?

5. Quelle classification obtient-t-on pour une patiente présentant les caractéristiques "age>50", "ménopause=pré-ménopause" et "tumeur initiale<=35" en utilisant l'arbre complet T^c ? En utilisant le sous-arbre $\tilde{T}$?

Algorithme des k-plus proches voisins

6. Pour deux individus i, j, $i \neq j$, ayant comme covariables $\mathbf{X_i} = (X_{i1}, X_{i2}, X_{i3})^t$, $\mathbf{X_j} = (X_{j1}, X_{j2}, X_{j3})^t$, on définit une mesure de dissimilarité par :

$$\mu(\mathbf{X_i}, \mathbf{X_j}) = \sum_{l=1}^{3} \mathbb{1}_{X_{il} \neq X_{jl}}.$$

Par exemple, $\mu(\mathbf{X_1}, \mathbf{X_4}) = 2$. On peut ainsi définir les k plus proches voisin de x comme les k individus ayant la plus petite mesure de dissimilarité parmi les 10 individus. Calculez la mesure de dissimilarité de chaque individu par rapport à une patiente présentant les caractéristiques "age>50", "ménopause=pré-ménopause" et "tumeur initiale<=35".

7. En déduire la classification de cette patiente à partir de l'algorithme des 2 plus proches voisins. Même question à partir de l'algorithme des 8 plus proches voisins.

Classifieur naïf de Bayes

8. On rappelle que le classifieur naïf de Bayes suppose que pour tout $\mathbf{x} = (x_1, x_2, x_3)^t$ et $y \in \{0, 1\}$:

$$\mathbb{P}(\mathbf{X_i} = \mathbf{x} \mid Y_i = y) = \prod_{i=1}^{3} \mathbb{P}(X_{il} = x_l \mid Y_i = y).$$

Pour y valant "oui", $\mathbf{x} = (x_1, x_2, x_3)^t$ avec x_1 ="age>50", x_2 ="ménopause=pré-ménopause" et x_3 ="tumeur initiale<=35", calculez $\mathbb{P}(Y_i = y \mid \mathbf{X_i} = \mathbf{x})$.

9. Même question avec y valant "non".
10. En déduire la classification de la patiente présentant les caractéristiques "age>50", "ménopause=pré-ménopause" et "tumeur initiale<=35" à partir du classifieur naïf de Bayes.

Exercice 2.
Dans cet exercice, vous allez analyser la base de données `Boston` du package `MASS` (506 observations et 14 variables). Téléchargez la base de données directement depuis R, en tapant par exemple `data(Boston, package = "MASS")` puis convertissez la variable `medv` en une variable catégorielle prenant la modalité "TRUE" si `medv`$\geq$ 20 et "FALSE" sinon. Le but de cet exercice est de classifier l'étiquette `medv`$\geq$ 20 en utilisant les 13 autres covariables de la base de données.

1. Construire un sous-échantillon, tiré aléatoirement (sans remise), constitué de 450 individus de la base de données. On appellera cette sous-base de données `Boston_sub`. La base de données contenant les 56 autres individus s'appellera `Boston_test`. Faites attention : dans toute la suite de l'exercice vous travaillerez sur la sous-base de données `Boston_sub`. La base de données `Boston_test` ne sera utilisée qu'aux questions 9. et 10.
2. Implémentez un modèle de régression logistique pour classifier l'étiquette `medv`$\geq$ 20 à l'aide de toutes les covariables. Commentez la sortie R (en particulier les effets estimés des variables et les tests correspondants). Effectuez une sélection de variable (en précisant la méthode utilisée) et commentez vos résultats.
3. En jouant avec les paramètres de la fonction `rpart`, construire un arbre de classification CART maximal. A l'aide de la méthode du coude (fonction `plotcp`), sélectionnez un arbre élagué. Expliquez votre choix d'arbre élagué et commentez.

4. À partir de la sous-base de données `Boston_sub`, tirez aléatoirement (sans remise) 300 individus pour constituer un échantillon d'apprentissage (que l'on nommera `Boston_train`). Les 150 autres individus constitueront un échantillon de validation (que l'on nommera `Boston_valid`). À l'aide de la méthode du bagging, construire différents classifieurs en jouant sur la profondeur des arbres (option `maxdepth`) et le nombre d'arbres (option `mfinal`). Vous entraînerez vos arbres sur l'échantillon `Boston_train` et vous évaluerez le taux de mauvaise classification sur l'échantillon `Boston_valid`. Justifiez ainsi un choix des paramètres de `maxdepth` et `mfinal`.
5. Procédez de la même manière, cette fois à partir des forêts aléatoires. Vous justifierez d'un choix optimal du nombre d'arbres (option `ntree`) et du nombre de variables candidates à chaque découpage de l'arbre (option `mtry`) en entraînant vos arbres sur l'échantillon `Boston_train` et en évaluant le taux de mauvaise classification sur l'échantillon `Boston_valid`.

Cross-validation V-fold et super-learner

Dans toute la suite, vous garderez les variables sélectionnées par la régression logistique dans la question 2, la profondeur de l'arbre élagué de la question 3, les paramètres `maxdepth` et `mfinal` du bagging de la question 4 et les paramètres `ntree` et `mtry` des forêts aléatoires de la question 5.

Vous devez à présent évaluer la performance de ces 4 classifieurs par la méthode V-fold de cross-validation (avec V= 10). Pour cela, vous découperez la sous-base de données `Boston_sub` en 10 blocs (les "folds"), avec à chaque fois 405 observations (les observations de 9 blocs) sur lesquelles les algorithmes sont entraînés et les 45 observations restantes (les observations du bloc restant de validation) sur lesquelles les algorithmes vont classifier les observations. En faisant varier le rôle du bloc de validation et des 9 blocs d'apprentissage, vous obtiendrez, pour chacun des 4 classifieurs, une prédiction de toutes les observations de la base de données `Boston_sub`. Une illustration du découpage de la base de données en blocs apprentissage et validation est illustrée sur la Figure 3 (pour V= 5).

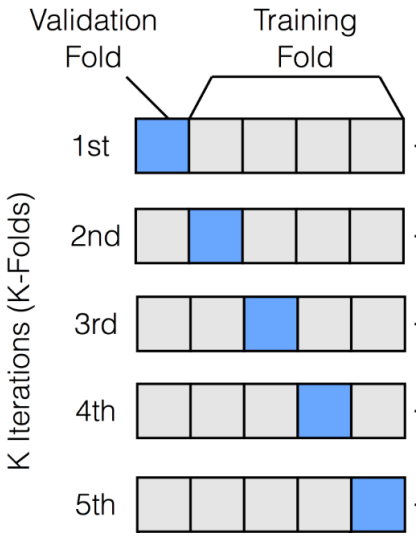

FIGURE 3 – Illustration de la procédure de cross-validation V-fold, pour V= 5.

Il existe de nombreuses façon de coder la cross-validation. Pour vous aider, vous pourrez vous inspirer du code suivant :

```
suite<-1:450
Pred_mat=data.frame(matrix(NA,450,5))
Pred_mat[,1]<-Boston_sub$medv #la premiere colonne contient les vraies etiquettes
for (v in 1:10)
{
 indices<-suite[-c((1+(v-1)*45):(v*45))]
 algo1<-algo_fun(...,data = Boston_sub[indices,],...)
 ...
 Pred_mat[c((1+(v-1)*45):(v*45)),2]<-predict(algo1,Boston_sub[c((1+(v-1)*45):(v*45)),])
 ...
}
```

6. Présentez vos résultat dans une matrice `Pred_mat` contenant 450 lignes et 5 colonnes, où la première colonne contient les vraies étiquettes et les colonnes 2 à 5 contiennent les probabilités de classifier "TRUE" pour chacun des 4 algorithmes.
7. En déduire le taux de mauvaise classification pour chacun des 4 algorithmes. Commentez.
8. On cherche à présent à construire un algorithme dit **super learner** qui combine les prédictions des 4 algorithmes de classification. Pour cela, implémentez un modèle de régression logistique avec comme variable réponse la première colonne de `Pred_mat` (les vraies étiquettes) et comme covariables les colonnes 2 à 5 de `Pred_mat` (les probabilités prédictes des 4 algorithmes de classification). Commentez les effets obtenus ainsi que les tests correspondants.
9. Pour obtenir la probabilité de classifier en "TRUE" une nouvelle observation à partir du **super learner** il suffit d'utiliser la formule classique d'un modèle de régression logistique :

$$\mathbb{P}(Y = \widehat{1 \mid X} = x) = \frac{\exp(\hat{\beta}_0 + \hat{\beta}^t x)}{1 + \exp(\hat{\beta}_0 + \hat{\beta}^t x)},$$

où $\hat{\beta}_0 \in \mathbb{R}$, $\hat{\beta} \in \mathbb{R}^4$ sont les paramètres estimés de la régression logistique de la question 8. (obtenus à partir de la commande \$coefficients du modèle `glm`). Calculez ces probabilités sur l'échantillon `Boston_test` et en déduire une classification pour ces observations. Calculez le taux de mauvaise classification.

10. Classifiez les observations de l'échantillon `Boston_test` à partir des 4 algorithmes de classification individuels (régression logistique, CART, bagging, forêt aléatoire) et calculez le taux de mauvaise classification. Comparez avec le super learner.