Intelligence artificielle : Théorie & Algorithme

Examen final

Session principale

Durée : 2 heures

Calculatrices autorisées. Deux feuilles R/V sont autorisées.

Exercice I (Jeux)

On considère le labyrinthe défini ci-dessous (Figure 1). Le départ se situe en haut du labyrinthe et l'arrivée en bas.

1) Appliquer l'algorithme A* avec comme heuristique la distance de Manhattan, pour trouver le chemin optimal permettant de sortir de ce labyrinthe. Vous détaillerez toutes les cases visitées, en spécifiant pour chacune les valeurs de g et h telles que définies dans le cours.
2) Si on avait utilisé la distance euclidienne comme heuristique, celle-ci aurait-elle été admissible (on dit aussi minorante) ? Justifiez votre réponse.

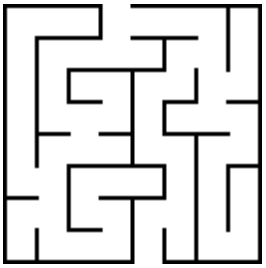


Figure 1

Exercice II (Jeux)

On considère l'arbre d'un jeu (à deux joueurs) défini comme suit :

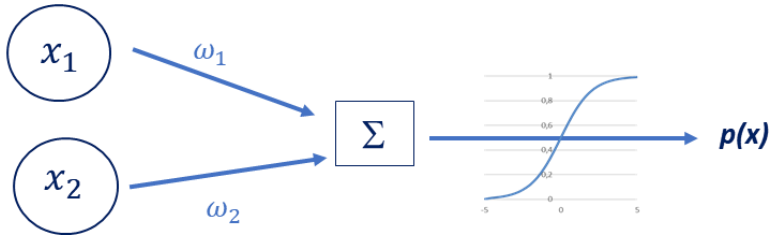
- Il comporte 4 niveaux (racine=niveau 1 → feuilles=niveau 4).
- La racine correspond au joueur Max.
- Chaque noeud non terminal possède 3 fils.
- Les feuilles sont étiquetées par les valeurs suivantes (de gauche à droite): 8, 7, 3, 9, 1, 6, 2, 4, 1, 1, 3, 5, 3, 9, 2, 6, 5, 2, 1, 2, 3, 9, 7, 2, 1, 6, 4.

1) Construire l'arbre.
2) Rappelez la signification de la racine de l'arbre ainsi que celle de ses feuilles.
3) Expliquer la signification des valeurs des feuilles et la manière dont elles ont pu être obtenues.

4) Appliquer l'algorithme Minimax. Que signifie la valeur de la racine ainsi obtenue?
5) Comment le joueur Max utilisera-t-il le résultat de l'application de Minimax.
6) Appliquer l'algorithme MiniMax avec élagage Alpha-Beta. Donnez une évaluation du temps que cette version de l'algorithme nous a fait gagner.

Exercice III (Data mining)

Soit Y une variable cible binaire. Considérons le réseau de neurones suivant,



où Σ est la combinaison linéaire à laquelle on applique la fonction d'activation sigmoïde, $f(x)=1/(1+e^{-x})$.

1) Que représente la sortie du réseau de neurones ?
2) Ecrire la sortie du réseau de neurones en fonction des entrées x_1 et x_2.

Dans l'algorithme de rétropragation du gradient, on initialise les poids à $w_1=w_2=1$ et on considère la formule de mise à jour des poids

$$\Delta w_i = \alpha\,[\,y - p(\boldsymbol{x})\,]\,x_i$$
$$w_i \leftarrow w_i + \Delta w_i$$

avec un taux d'apprentissage $\alpha = 0.9$.

Soit la base d'apprentissage suivante,

	x_1	x_2	Y
1	0.5	0.6	1
2	2.6	1.2	0

3) Quelle est la sortie du réseau de neurones pour la ligne 1 (arrondi à 10^{-2})?
4) Que deviennent les poids à l'issue de la 1ère itération (ligne 1) ?
5) Que deviennent les poids à l'issue de la 2ème itération (ligne 2) ?
6) Soit la matrice de confusion suivante obtenue sur une base de 125 observations

		Predicted values		
		A	B	C
True values	A	51	2	1
	B	3	58	0
	C	2	3	5

a) Quel est le taux de bien classés ?
b) Que pouvez-vous dire sur la performance du modèle ?

Exercice IV (Data mining)

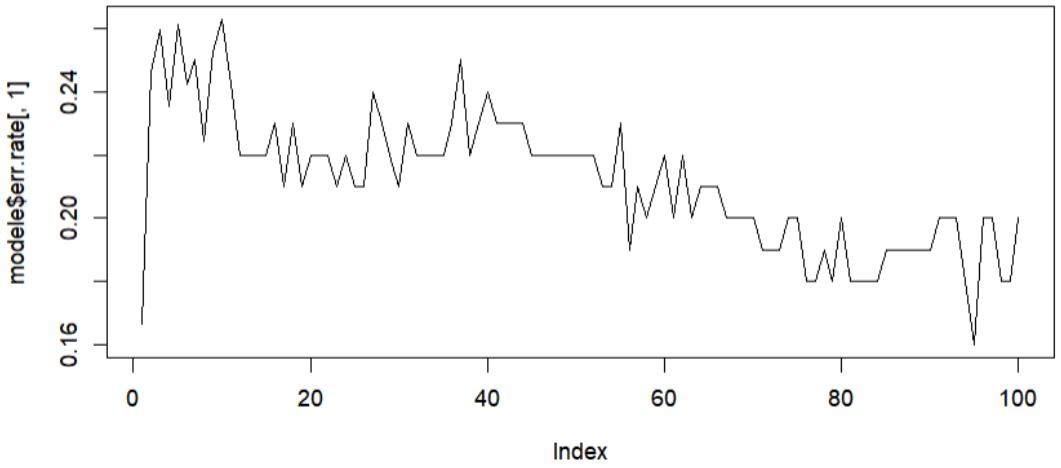
Afin de construire un modèle pour prédire le cancer de la prostate, on considère une base d'apprentissage de 100 patients caractérisés par 8 variables numériques (Radius, Texture, Perimeter, Area, Smoothness, Compactness, Symmetry, Fractal dimension) et une variable cible binaire représentant la présence ou l'absence de cellules cancéreuses (Y=diagnostic_result). La classe positive (Y=1) est notée « M » pour Malignant et la classe négative (Y=0) est notée « B » pour Bénigne.

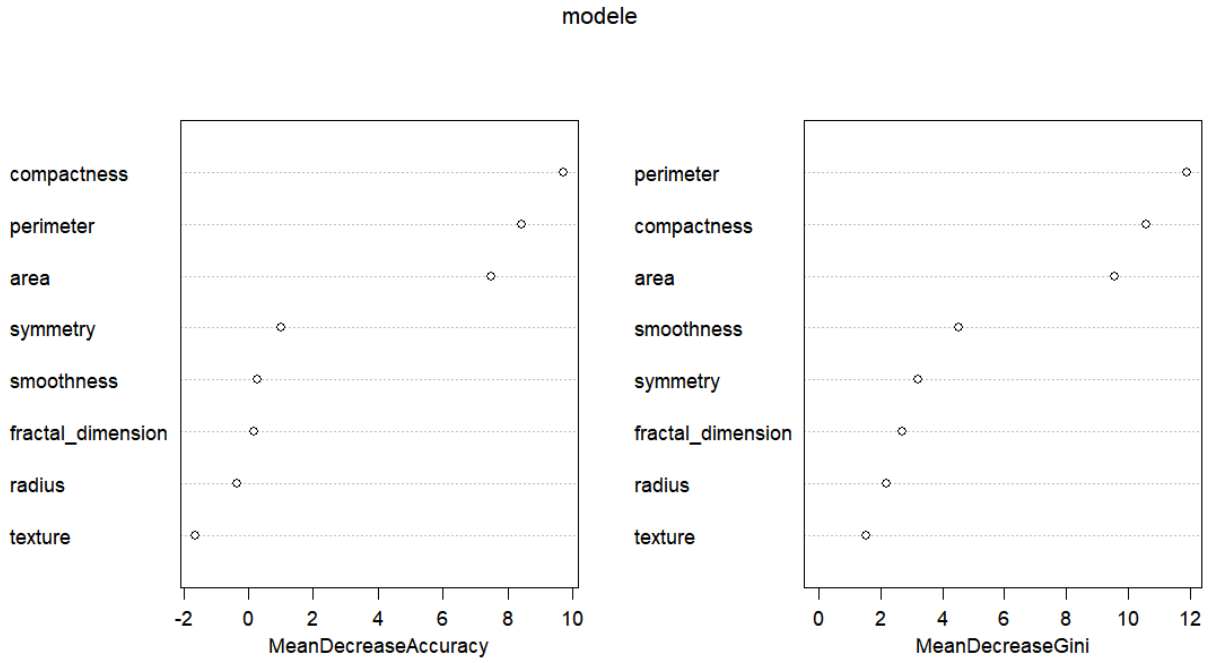
On ajuste une forêt aléatoire avec le package RandomForest de R dont les résultats sont donnés ci-dessous.

1) Combien y-a-t-il d'arbres dans la forêt ? Pensez-vous que cela est suffisant ou bien faut-il ajouter des arbres (justifiez votre réponse) ?
2) Combien de variables sont testées (mises en concurrence) à chaque nœud de chaque arbre ? Expliquez ce chiffre.
3) Quelles sont les variables les plus importantes pour prédire la présence ou l'absence de cellules cancéreuses ?
4) Expliquez pourquoi les forêts aléatoires sont des algorithmes stochastiques (aléatoires) ?
5) Donnez 3 hyperparamètres qui ont un impact sur le temps de calcul.
6) Calculer le taux de bien classés, la specificity (recall) et la précision.

```
Call:
 randomForest(formula = diagnosis_result ~ ., data = tab, importance = T,      ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 20%
Confusion matrix:
   B  M class.error
B 29  9   0.2368421
M 11 51   0.1774194
```





Exercice V (réflexion)

Si vous disposez de plusieurs nœuds de calcul, quel(s) algorithme(s) vu(s) en cours (Naive Bayes, Arbre de décision, Foret aléatoire, Réseau de neurones), pouvez-vous simplement distribués sur ces nœuds afin de diminuer son temps de calculs ? Expliquer pourquoi ?