Sujet n°5 : Arbres décisionnels et Forêts aléatoires

Elsa Azoulay, Guénolé Joubioux, Grégoire Mourre, Anaëlle Zerbib

Mai 2023

Résumé

L'apprentissage statistique constitue un ensemble de méthodes qui, à partir d'un jeu de données, permettent d'extraire des informations prévisionnelles et/ou et décisionnelles. Parmi les différentes catégories d'apprentissage statistique, on distingue l'apprentissage supervisé où l'on travaille à partir de données étiquetées et l'apprentissage non supervisé où les réponses que l'on cherche à prédire ne sont pas dans notre jeu de données. Les arbres décisionnels correspondent à une méthode d'apprentissage supervisé de prises de décisions très efficace. En effet, cette méthode permet de donner une interprétation compréhensible des décisions prises. Elle est utilisée dans de nombreux domaines tels que le domaine médical, le domaine budgétaire etc. Dans un premier temps, nous tâcherons d'analyser le fonctionnement des arbres décisionnels, des forêts aléatoires, ainsi que de la méthode bagging. Dans un second temps, nous travaillerons sur un exemple concret de prédiction du cancer du sein à partir de la base de données `load breast cancer` et nous travaillerons notamment sur le choix des hyper-paramètres de chacune des méthodes.

Table des matières

1 Présentation du cadre

Dans toutes les méthodes étudiées, les données avec lesquelles on travaille sont des échantillons d'une population de taille N. Elles sont représentées par l'ensemble :

$$D_N = \{(X_i, Y_i), i = 1, ..., N\}.$$

Nous supposerons que ce sont des réalisations *iid* d'un couple de variables aléatoires (X, Y) défini sur $(\Omega, \mathcal{F}, \mathbb{P})$ et à valeur dans $\mathcal{X} \times \mathcal{Y}$.

Les variables $X = (X^1, ..., X^p)$ sont appelées *features* ; elles constituent l'ensemble des caractéristiques sur lesquelles s'appuie notre analyse et sont au nombre de p. La variable Y est appelée *target* ; elle peut être qualitative par exemple pour la détection d'un cancer, ou quantitative comme le prix d'une habitation. Les arbres décisionnels s'adaptent aussi bien au cas discret, en appliquant le procédé de classification, qu'au cas continu à l'aide d'une régression.

On divisera le jeu de données en un set de *training* que l'on notera D_N^{train} et un set de *test*, que l'on notera D_N^{test}. On appellera également $\mathcal{X}$ l'ensemble des *features*.

2 Présentation des arbres décisionnels

Les arbres décisionnels sont une famille de méthodes d'apprentissage statistique. Dans le cas de données quantitatives, le principe de ces méthodes est de créer un arbre binaire de décision, qui, à chaque feuille (*leaf*), associe une valeur. Cette méthode assure une interprétation facilement exploitable et peut être appliquée dans une multitude de domaines.
Voici un exemple d'arbre de décision :

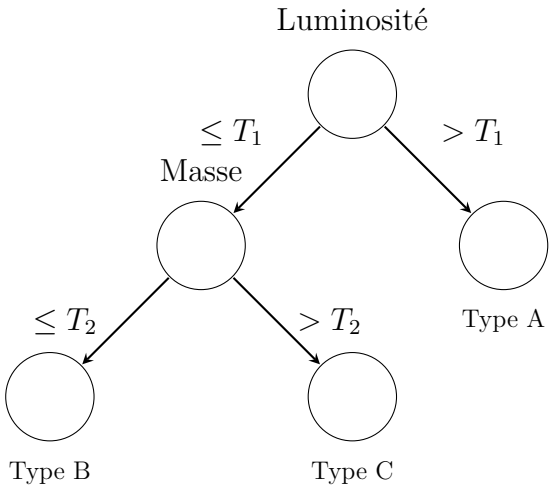

FIGURE 1 – Exemple élémentaire d'un arbre aléatoire pour la prédiction d'une *target* qualitative, avec des *features* quantitative.

Cet exemple illustre la classification des différents types d'étoile en fonction de leur luminosité et de leur masse. Le principe est assez simple : chaque noeud interne correspond à un attribut, et chaque noeud terminal engendre une classe, ici un type d'étoile. Les noeuds internes sont créés selon les différentes valeurs que peut prendre un attribut. Par exemple ici on crée un noeud dont le critère est : Luminosité $\leq$ T1 et $>$ T1. Les noeuds terminaux, appelés *leaves*, assignent l'appartenance à une classe.
Dans le cas de *features* qualitatives, les noeuds se divisent en autant de sous-noeuds qu'il n'y a de possibilités. On peut le voir en figure 2.

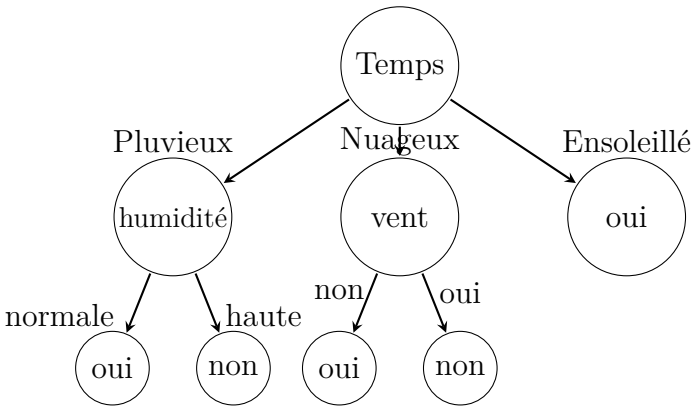

FIGURE 2 – Exemple élémentaire d'un arbre aléatoire pour des *features* qualitatives et une *target* qualitative

La première propriété importante d'un arbre décisionnel est la pureté de ses feuilles. Une feuille est pure lorsque tous les éléments de celle-ci appartiennent à la même classe. L'algorithme doit donc optimiser la pureté de chaque *leaf*.

Formulation mathématique L'idée générale est que les différentes feuilles de l'arbre partitionnent l'ensemble des *features* $\mathcal{X}$. Dans cette exemple, $\mathcal{X} = \mathbb{R}^p$. Dans un arbre de décision, on cherche justement à partitionner l'ensemble des *features* $\mathcal{X}$ en rectangles de $\mathbb{R}^p$ de la forme

$$R_k := \bigotimes_{k=1}^{p}]a_k, b_k]$$

pour $n \in \mathbb{N}^*$, $k \in \{1, \cdots, n\}$, les $(a_k)_k, (b_k)_k \in \mathbb{R} \cup \{\pm\infty\}$ et $a_k < b_k,\ \forall k \in \{1, \cdots, n\}$. On a donc :

$$\mathcal{X} = \sqcup_{k=1}^{n} R_k.$$

On associe ensuite une valeur $\hat{y}_k$ à chacun des R_k, pour $k \in \{1, \cdots, n\}$ et la prédiction de l'arbre pour $(X_1, \cdots, X_p)$ est donc $\hat{y}_k$ avec k tel que $(X_1, \cdots, X_p) \in R_k$.

Pour reprendre l'exemple donné plus haut en figure 1, la partition de $\mathcal{X}$ obtenue peut se représenter de cette manière, avec la prédiction associée à chaque ensemble :

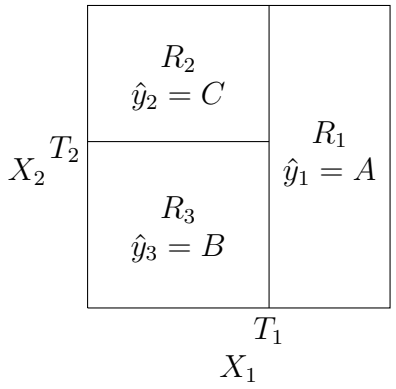

FIGURE 3 – Partition de l'ensemble des *features* avec X_1 qui représente la luminosité et X_2 qui représente la masse.

Lorsque la variable Y à prédire est quantitative, le prédicteur $\hat{y}_k$ associé à l'un des R_k est la moyenne des observations appartenant à ce R_k. Dans le cas d'une variable Y qualitative, le prédicteur $\hat{y}_k$ correspond à la réponse y la plus présente pour les données appartenant à ce R_k.

Construction récursive d'un arbre Le but est de construire les différentes régions $(R_k)_{\{k\in\{1,\cdots,p\}\}}$ de telle sorte que

$$RSS := \sum_{k=1}^{p} \sum_{x_i \in R_k} (y_i - \hat{y}_k)^2$$

soit minimal. L'approche des arbres décisionnels consiste à récursivement trouver la valeur (et le seuil dans le cas quantitatif) qui permet de minimiser le RSS de l'arbre déjà construit à l'étape précédente. On part de l'arbre à une seule région de prédiction et donc de

$$RSS_0 := \sum_{x_i \in D_N^{train}} (y_i - \hat{y})^2.$$

Puis, pour un arbre construit à la j-ième étape, avec

$$RSS_j := \sum_{k=1}^{j} \sum_{x_i \in R_k^j} (y_i - \hat{y}_k^j)^2,$$

on choisit la variable et le seuil qui minimisent le nouveau RSS_{j+1} obtenu en divisant l'un des $(R_k)_{\{k\in\{1,\cdots,j\}\}}$ en deux nouveaux rectangles. On ré-itère ce processus jusqu'à n'avoir plus que des feuilles pures. Dans le cas de variables qualitatives, le principe reste le même.

Problème d'*overfitting* Une des questions délicates concernant les arbres décisionnels est de connaître la taille optimale de l'arbre. Si l'arbre est trop grand, le problème du sur-ajustement (*overfitting*) apparait et la méthode a du mal à se généraliser à de nouveaux échantillons. D'autre part, un arbre trop court devient trop imprécis et ne permet plus d'expliquer la *target*. Pour éviter l'*overfitting*, on va chercher à « réduire » l'arbre décisionnel initial pour obtenir un sous-arbre. C'est une méthode dite d'élagage, ou *pruning*, qui consiste à rajouter lors de la construction d'un arbre un hyper-paramètre α pénalisant la longueur des arbres. Le but est donc de supprimer les branches peu représentatives tout en gardant de bonnes performances prédictives. La méthode d'élagage assure un équilibre entre performances et interprétabilité. Notons T_0 l'arbre initial. Dans le cas d'une régression, on calcule, pour tous les sous-arbres $T \subset T_0$, la fonction de coût :

$$cost = \sum_{k=1}^{|T|} \sum_{x_i \in R_k} (y_i - \hat{y}_{R_k})^2 + \alpha|T|,$$

où $|T|$ représente le nombre de feuilles (*leaf*). Dans le cas d'une classification, on calcule, pour tous les sous-arbres $T \subset T_0$, la fonction de coût :

$$cost = \sum_{k=1}^{|T|} \sum_{x_i \in R_k} \mathbb{1}_{\{y_i \neq \hat{y}_{R_k}\}} + \alpha|T|.$$

Finalement, on prend le sous-arbre qui a le plus petit coût (*cost*).

« *L'explication la plus simple est souvent la meilleure.* » A la fin de l'élagage, l'arbre résultant est moins complexe et par conséquent plus simple à interpréter.

3 Présentation des forêts

Désormais, afin d'accroître notre précision, on ne génère plus un seul arbre pour prendre notre décision mais un ensemble d'arbres qu'on appellera forêt. En effet, la méthode des arbres

décisionnels manquait de stabilité puisqu'un léger changement dans l'ensemble des données pouvait conduire à des résultats totalement différents. Il est donc naturel de résoudre ce problème en entraînant plusieurs arbres différents et en prenant comme estimateur final la moyenne des estimateurs de chaque arbre dans le cas d'un problème de régression, ou la réponse majoritaire dans le cas d'un problème de classification. Les algorithmes présentés dans cette partie sont des méthodes d'apprentissage qui reposent sur la combinaison de plusieurs algorithmes d'apprentissage, afin d'accroître les performances du modèle final. L'intérêt majeur de ces méthodes est la réduction de la variance.

3.1 Bagging

La méthode de *boostrap aggregating*, plus communément appelée *bagging* a pour but de générer un certain nombre d'arbres, de les entraîner sur différents jeux de données, pour ensuite faire la moyenne ou un vote des prédictions que ces arbres renvoient. Ce nombre d'arbres sera considéré comme un hyper-paramètre à déterminer. On ne peut prendre la même base de données pour générer tous les arbres car le but de la méthode est de réduire la variance. Nous cherchons donc plutôt à produire des arbres « quasiment décorrélés ». Puisque l'on ne dispose que d'une base de données, on utilise la méthode *boostrap* pour en produire plusieurs. Ensuite, on effectue l'agrégation de la prédiction de nos arbres si le modèle est de régression, ou la méthode du vote si le modèle est de classification.

3.2 Forêts aléatoires

Le principe des forêts aléatoires est similaire à celui de *bagging*. Il est question de générer un certain nombre d'arbres à partir de nouvelles bases de données. Cette fois, on veut des arbres encore plus « dé-corrélés ». Le nombre d'arbres est toujours un hyper-paramètre. Dans cette méthode, la construction des nouvelles bases de données se fait un peu différemment. On fait également un *bootstrap*, mais sur les bases de données obtenues, on sélectionne aléatoirement m *features* à conserver sur les p. En général on prend m de l'ordre de $\sqrt{p}$. On conclut finalement de la même façon que pour le *bagging* : par agrégation ou vote.

4 Application pour la prédiction du cancer du sein

Nous étudions une base de données de $N = 569$ observations avec $p = 30$ variables explicatives et nous essayons de déterminer si une patiente est atteinte du cancer du sein. Dans notre cas, nous avons donc la variable à prédire $Y \in \{0, 1\}$: c'est un problème de classification. Le cas d'un cancer malin est décrit avec $Y = 0$ et le cas d'un cancer bénin avec $Y = 1$. Pour construire la base de données, les chercheurs ont d'abord récolté les données pour 10 variables quantitatives sur les noyaux cellulaires prélevés. Ils en ont ensuite déterminé la moyenne, l'erreur type et la plus grande valeur de ces variables portant ainsi le nombre de variables explicatives quantitatives à 30.

Pour notre étude, nous divisons notre jeu de données en deux, 80% pour l'entraînement des modèles et 20% pour pouvoir les tester et les comparer.

4.1 Classification par arbre décisionnel

Dans un premier temps, nous utilisons la méthode classique de classification par un arbre décisionnel. Une première tentative naïve d'arbre décisionnel nous donne la figure 4. Dans cet arbre (et dans le suivant 6), la condition est noté dans chaque noeud et les données qui la vérifient sont à gauche, tandis que les données ne vérifiant pas la condition sont à droite.

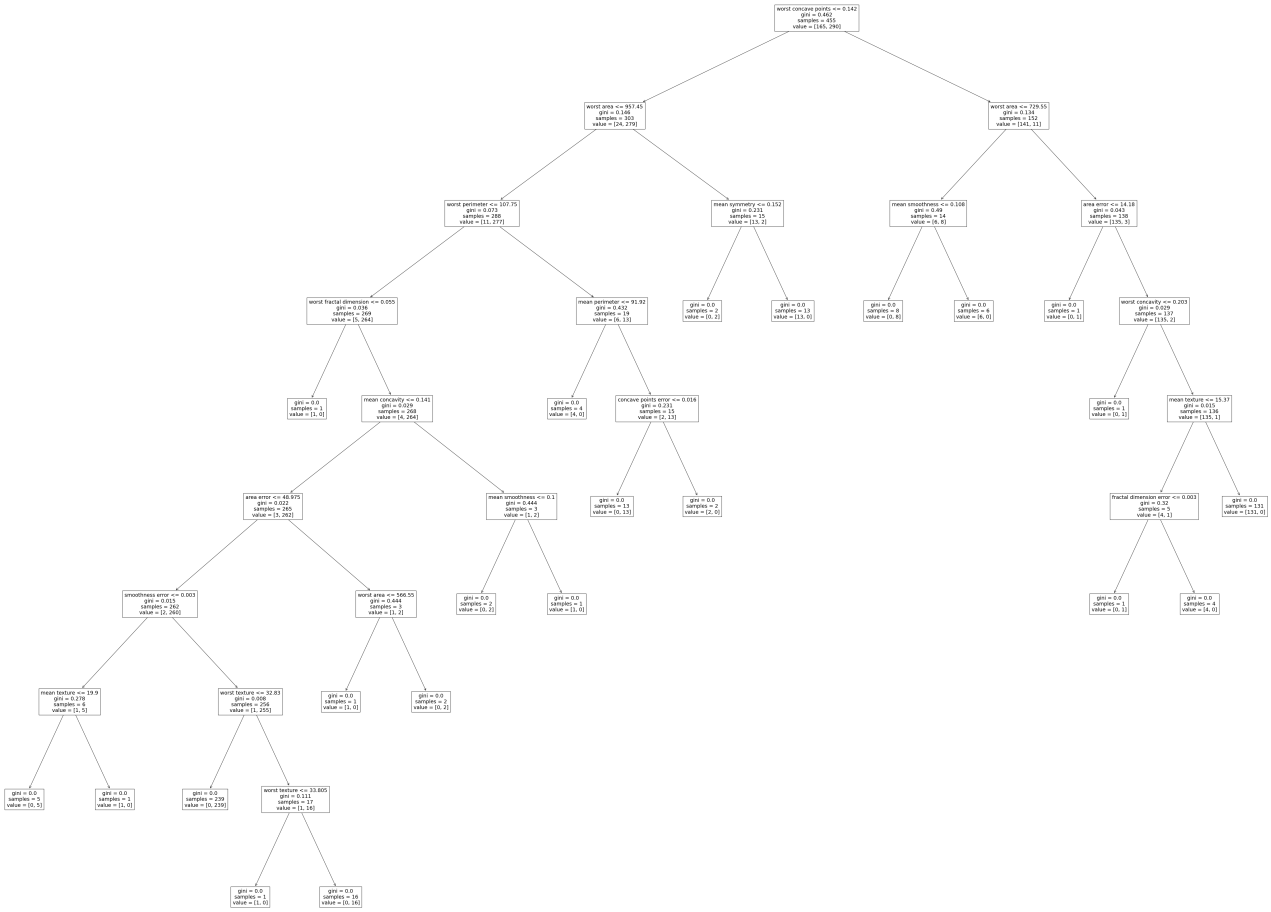

FIGURE 4 – Arbre décisionnel sans *pruning*.

On voit tout de suite que l'arbre est bien trop grand pour le peu de données que l'on a. Effectivement, toutes les feuilles terminales sont pures mais certaines n'ont que peu de données. Nous allons donc élaguer l'arbre en rajoutant un hyper-paramètre α que nous choisissons par *6th-cross-validation*.

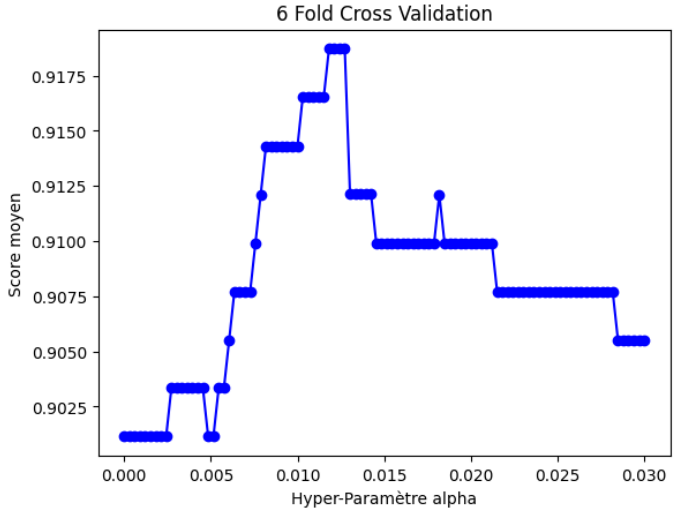

FIGURE 5 – Cross Validation pour l'hyper-paramètre α de *pruning*.

D'après le graphique 5, nous sélectionnons α proche de $0,012$ comme hyper-paramètre. On obtient alors sur le graphique 6 le nouvel arbre, résultant du *pruning*. Cet arbre est beaucoup plus lisible. Il comprend seulement 5 feuilles ce qui est déjà beaucoup plus réaliste avec le peu de données à notre disposition.

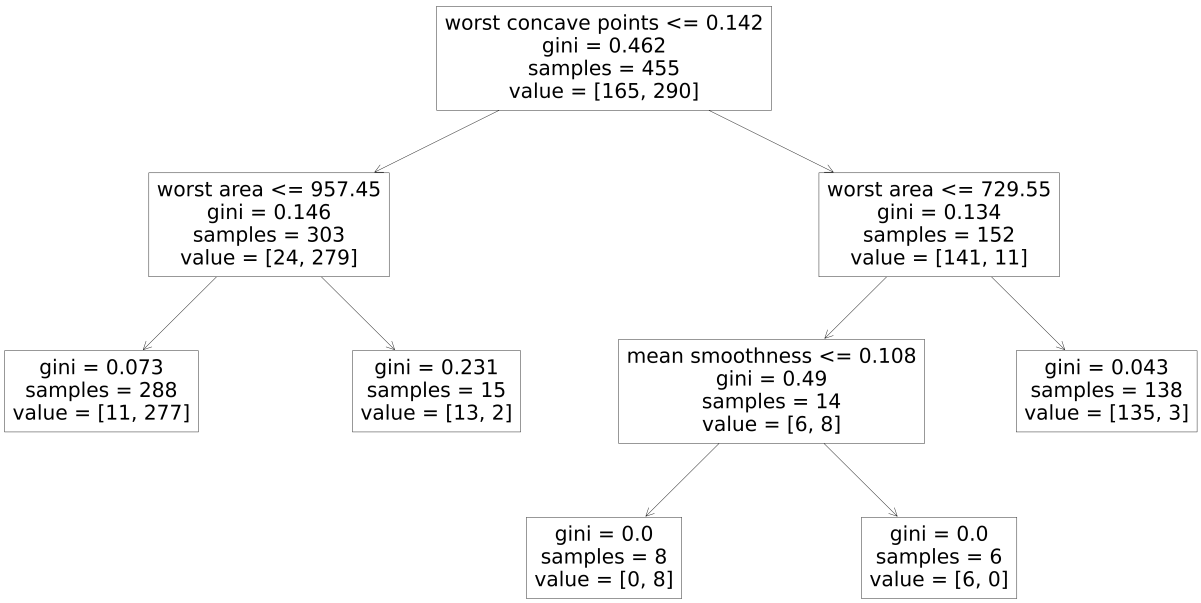

FIGURE 6 – Arbre après *pruning* avec paramètre α optimal (par cross validation).

4.2 Classification par la méthode *Bagging*

Dans la méthode du *Bagging*, nous devons choisir le nombre d'arbres. On utilisera pour cela le *Out-Of-Bag Score*, noté *OOB score*. Expliquons ce qu'il représente. Lors de la construction d'une forêt *bagging*, chaque arbre est construit avec différents sous-jeux de données construit par méthode *bootstrap*. Dans le jeu de données d'entraînement, durant la création de chaque arbre, une partie n'a donc pas été utilisée. Nous pouvons nous en servir pour tester l'arbre. Pour chaque donnée, nous récupérons les arbres qui n'ont pas été entraîné avec, et la valeur majoritairement renvoyée par ces arbres sur ces données correspond à la valeur prédite. L'*OOB score* calcule l'erreur de cette prédiction.

Il est plus rapide d'utiliser le *OOB Score* comme critère de décision pour l'hyper-paramètre que la méthode de *Cross-Validation*. Mais, puisque l'*OOB Score* s'appuie sur le principe de forêt on ne pouvait pas l'utiliser dans le cas d'un arbre décisionnel. Ici on y fera appel pour tracer le graphique 7. En l'interpretant, il semblerait que la précision du modèle augmente avec le nombre d'arbres. Il y a beaucoup de variations mais l'*OOB Score* semble se stabiliser à partir de 100 arbres. On choisit donc une forêt de cette taille.

4.3 Classification par une forêt aléatoire

Nous tentons désormais d'effectuer une classification grâce à une forêt aléatoire. Comme pour la méthode précédente, nous devons choisir le nombre d'arbres d'une forêt aléatoire. Puisqu'il y a également des données dites *Out-of-Bag*, nous réutilisons donc l'*OOB Score* pour sélectionner notre nombre d'arbres. Nous pouvons observer sur le graphique 8 qu'augmenter

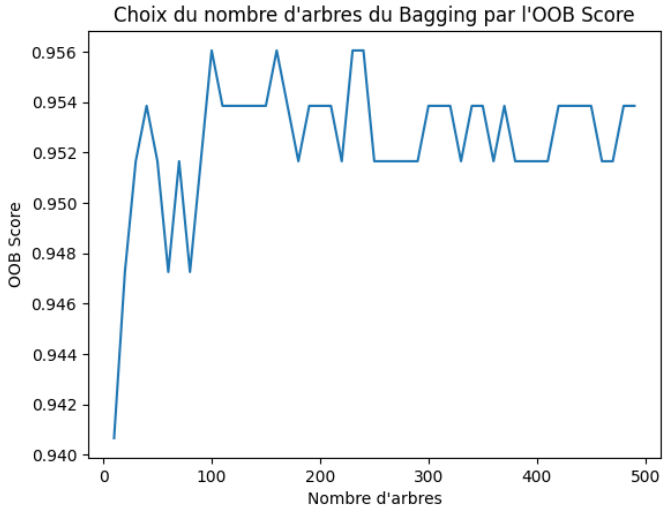

FIGURE 7 – Erreur d'estimation *OOB* (*Out-of-Bag*) pour le *bagging*.

le nombre d'arbres augmente la précision de la forêt aléatoire. A partir de l'*OOB Score*, nous choisissons une forêt aléatoire de 70 arbres.

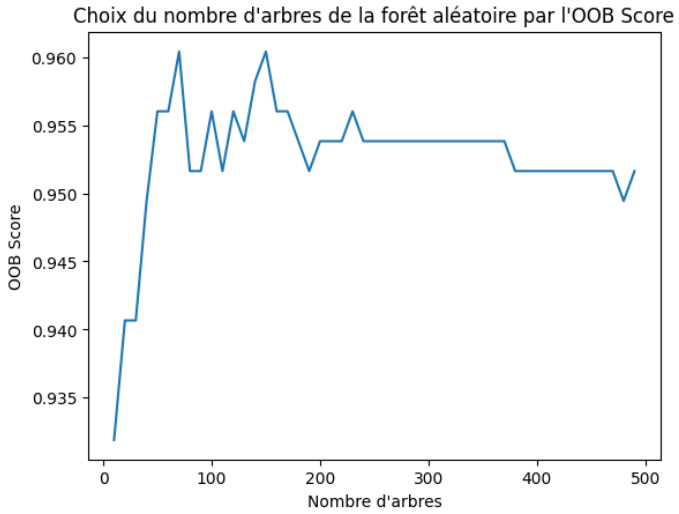

FIGURE 8 – Erreur d'estimation *OOB* (*Out-of-Bag*) pour la forêt aléatoire.

5 Comparaison des méthodes

Les *Decision Trees* avec *pruning* sont une amélioration des *Decision Trees* sans *pruning* car ils permettent de réduire le risque d'overfitting.
Le *Bagging* est une amélioration des *Decision Trees* car il permet de réduire la variance. En effet, on fait la « moyenne » de plusieurs arbres différents.
Les *Random Forests* sont une amélioration du *Bagging*. Ils représentent une méthode plus robuste car les arbres sont décorrélés entre eux : les arbres sont construits à partir de jeux de données ré-échantillonnés par *bootstrap* et dont on n'a gardé que m *features* parmi p. Cela force donc les arbres construits à être très différents les uns des autres.

Dans l'application, les trois dernières méthodes sont toutes performantes avec des taux de précision supérieurs à 96% sur le *test set* comme le montre le tableau 1. On remarque que la

classification par arbre de décision (avec *pruning*) est étonnamment meilleure que *bagging* et la forêt aléatoire. Cela est sûrement dû au fait que l'on ait peu de données dans `load breast cancer`. On note également que l'arbre de décision sans *pruning* à une précision de 100% sur le *train set* mais de seulement 91% sur le *test set*, ce qui illustre bien le problème d'*overfitting* de ce modèle.

Scores	Arbre de décision	Arbre de décision (*pruning*)	Bagging	Forêt aléatoire
Sur le *train set*	100%	96%	99%	100%
Sur le *test set*	91%	97%	96%	96%

TABLE 1 – Taux de précision par méthode de classification

Dans le cadre de notre jeu de données, les trois derniers modèles prédisent correctement comme le montre la figure 9. On remarque d'ailleurs que la méthode *Bagging* et la forêt aléatoire prédisent exactement la même chose. De plus, comme nous l'avions remarqué avec le taux de précision, l'arbre de décision simple avec *pruning* prédit mieux que les deux modèles plus complexes en réduisant l'erreur. Une explication de ce résultat peut provenir de la complexité des forêts. Lorsque l'on construit les forêts (*Bagging* et forêt aléatoire), les arbres ont des feuilles pures. Malheureusement, lorsque nous n'avons pas assez de données, les feuilles pures s'appuient sur peu d'observations. Les prédictions des arbres sont donc moins bonnes. Ainsi, traiter un nombre insuffisant de données peut amener à construire trop de mauvais arbres dans la forêt. Avec un petit jeu de données, un bon modèle d'arbre de décision peut être préférable par rapport à d'autres modèles plus compliqués.

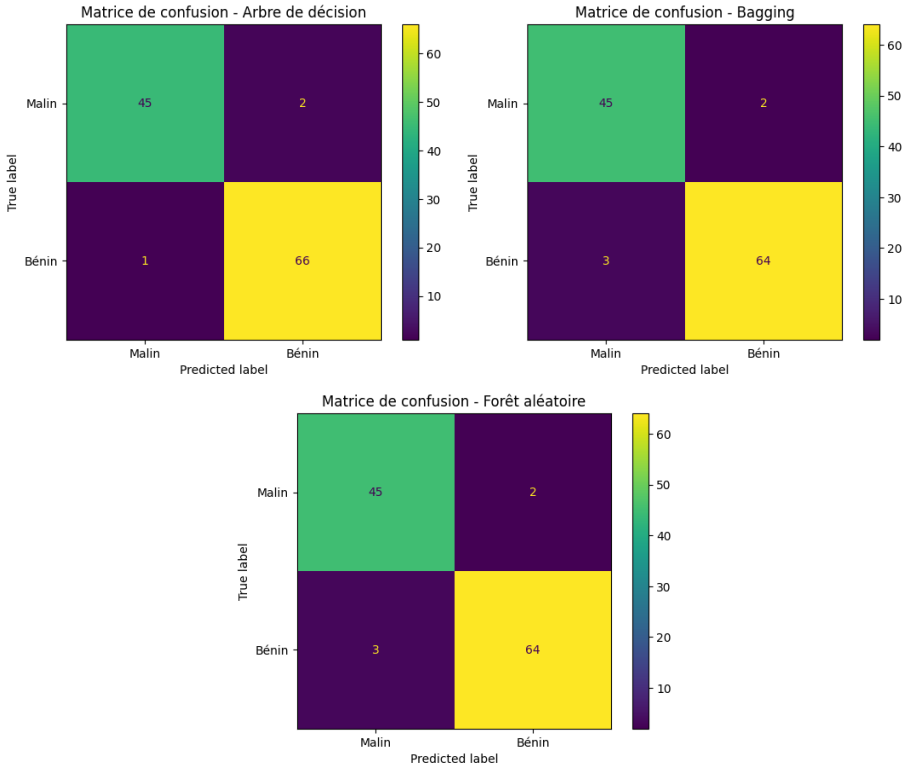

FIGURE 9 – Matrices de confusion pour nos trois bons modèles de classification (arbre de décision avec *pruning*, *bagging* et *random forest*).

6 Conclusion

Les arbres décisionnels sont un moyen efficace de classifier des données, *a fortiori* par leur simplicité à pouvoir expliciter des résultats. La méthode *Bagging* et les *Random Forests* sont des améliorations de l'arbre décisionnel car ils réduisent la variance des prédictions. Cependant, leur construction contient de l'aléa, rendant leur interprétation plus difficile. Une piste d'amélioration serait de considérer une construction optimisée de la forêt. La méthode *Boosting* pourrait être adéquate. Elle propose une construction optimisée de la forêt en formant chaque arbre de manière à partir du précédent. Lors de la construction d'un nouvel arbre dans la forêt, celui-ci a pour but de corriger les erreurs de la forêt en accordant plus d'importance à la bonne prédiction des précédentes erreurs de la forêt. Ainsi, au fur et à mesure que la forêt se construit, elle devient de plus en plus robuste. Il pourrait être intéressant d'implémenter cette méthode et de l'appliquer à notre jeu de données pour observer (ou pas) une amélioration des prédictions. Une autre idée d'amélioration de nos modèles serait d'ajouter un facteur d'élagage dans les modèles de *bagging* et de *random forests*. En effet, l'estimateur final pourrait être moins parasité par certains arbres trop longs.

Références

[Dua and Graff, 2017] Dua, D. and Graff, C. (2017). UCI machine learning repository.

[Gareth James, 2013] Gareth James, Daniela Witten, T. H. R. T. (2013). An introduction to statistical learning.