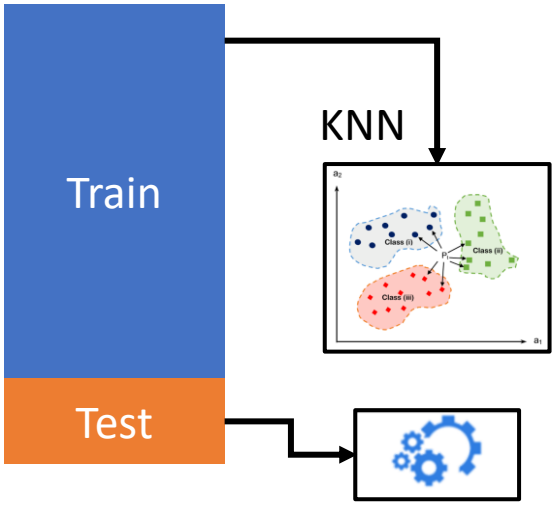
Ch9. Les algorithmes ensembliste – Bagging & boosting

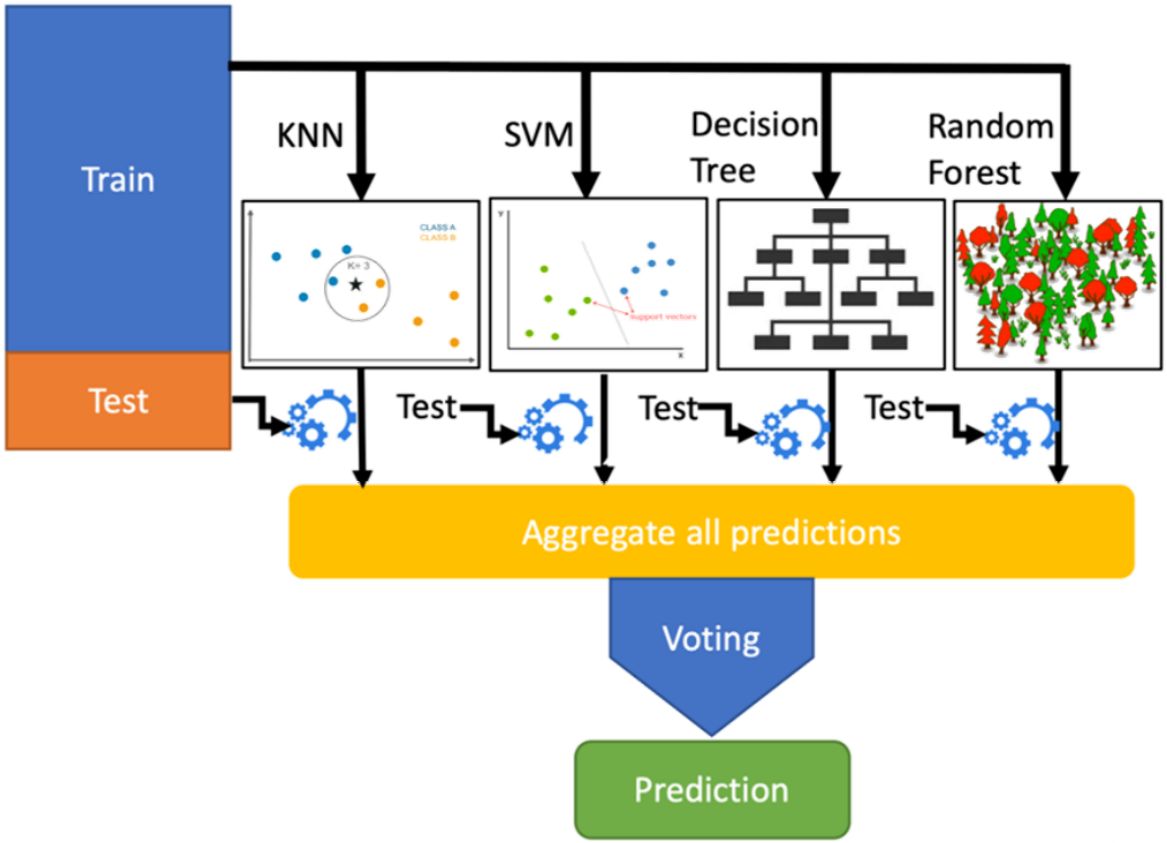
- Introduction aux algorithmes ensembliste
- Bagging & Boosting
- Random Forest

Introduction

Les méthodes ensemblistes combinent plusieurs modèles pour améliorer les performances prédictives au-delà de ce qu'un modèle unique peut atteindre.



Prédiction



Bagging

Définition du Bagging

Bootstrap Aggregating (Bagging) est une méthode ensembliste qui crée plusieurs versions d'un prédicteur et les utilise pour obtenir un prédicteur agrégé.

1 Échantillonnage bootstrap

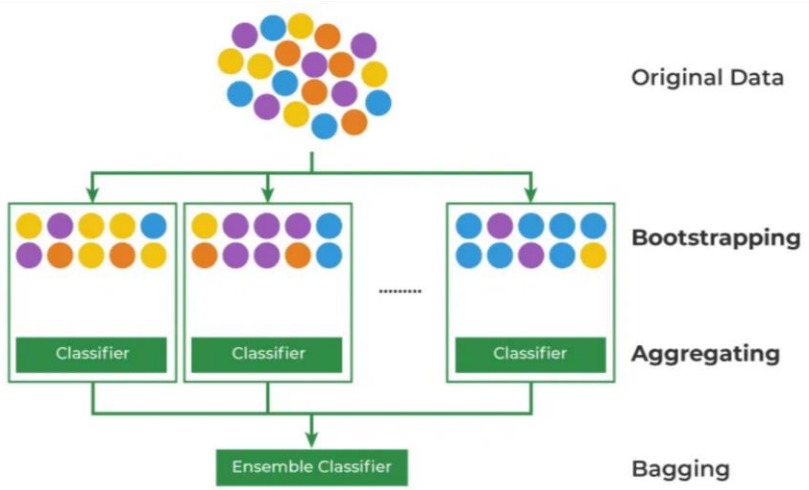
Création de N échantillons d'entraînement en tirant avec remise à partir du jeu de données original. Chaque échantillon a la même taille que l'ensemble original.

2 Entraînement indépendant

Entraînement indépendant et parallèle de N modèles de base, chacun sur un échantillon bootstrap différent. Les modèles peuvent être des arbres de décision, des réseaux de neurones, etc.

3 Agrégation des prédictions

Les prédictions individuelles sont combinées par vote majoritaire (classification) ou moyenne (régression).



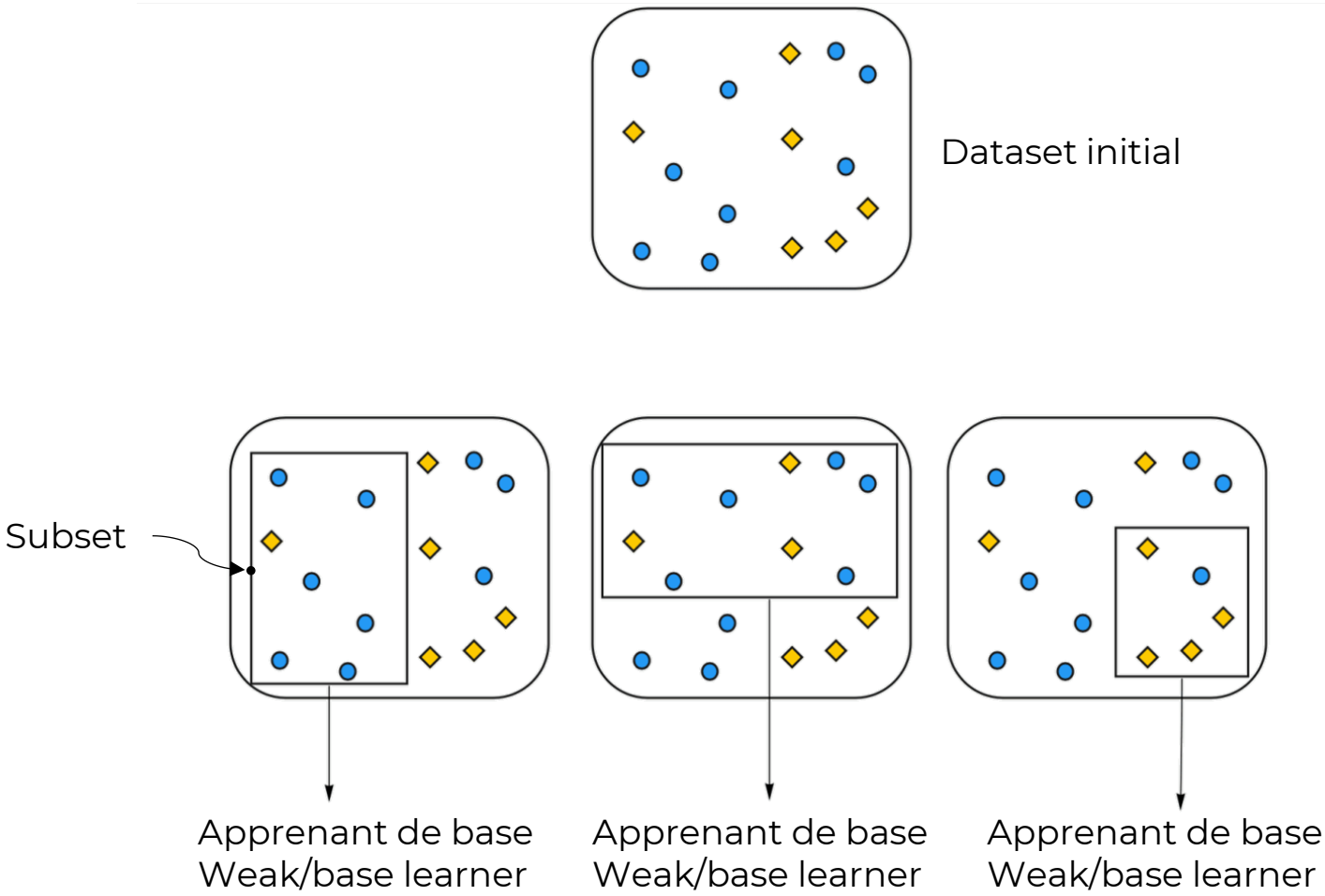
Pour la régression

$$f_{bag}(x) = \frac{1}{N}\sum f_i(x)$$

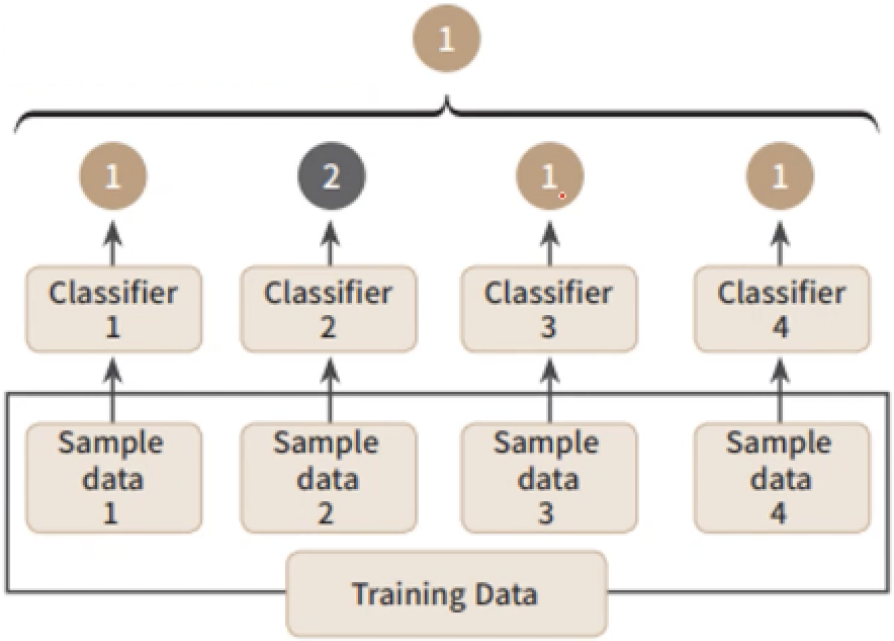
Pour la classification

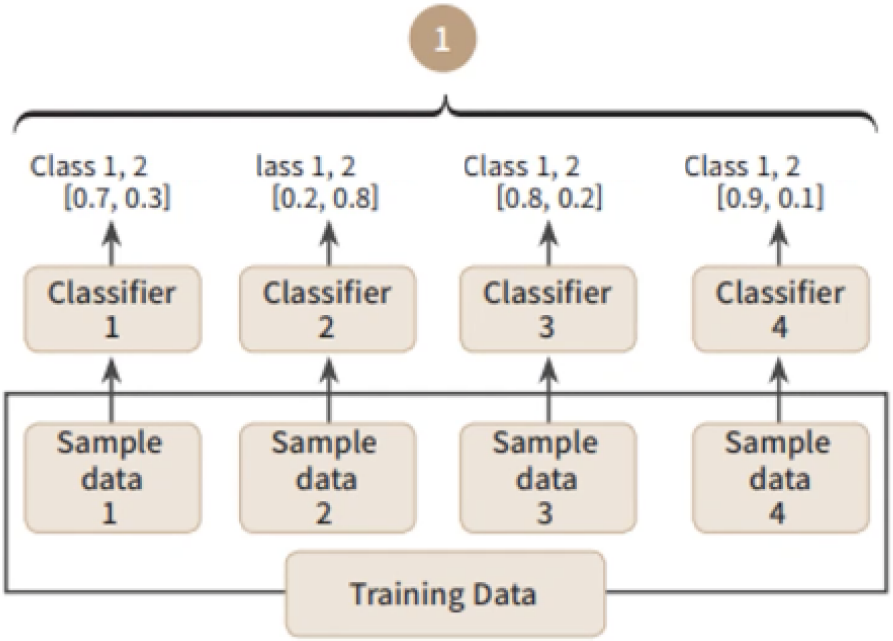
$$f_{bag}(x) = \text{mode}\{f_1(x), f_2(x), \ldots, f_n(x)\}$$

Bagging

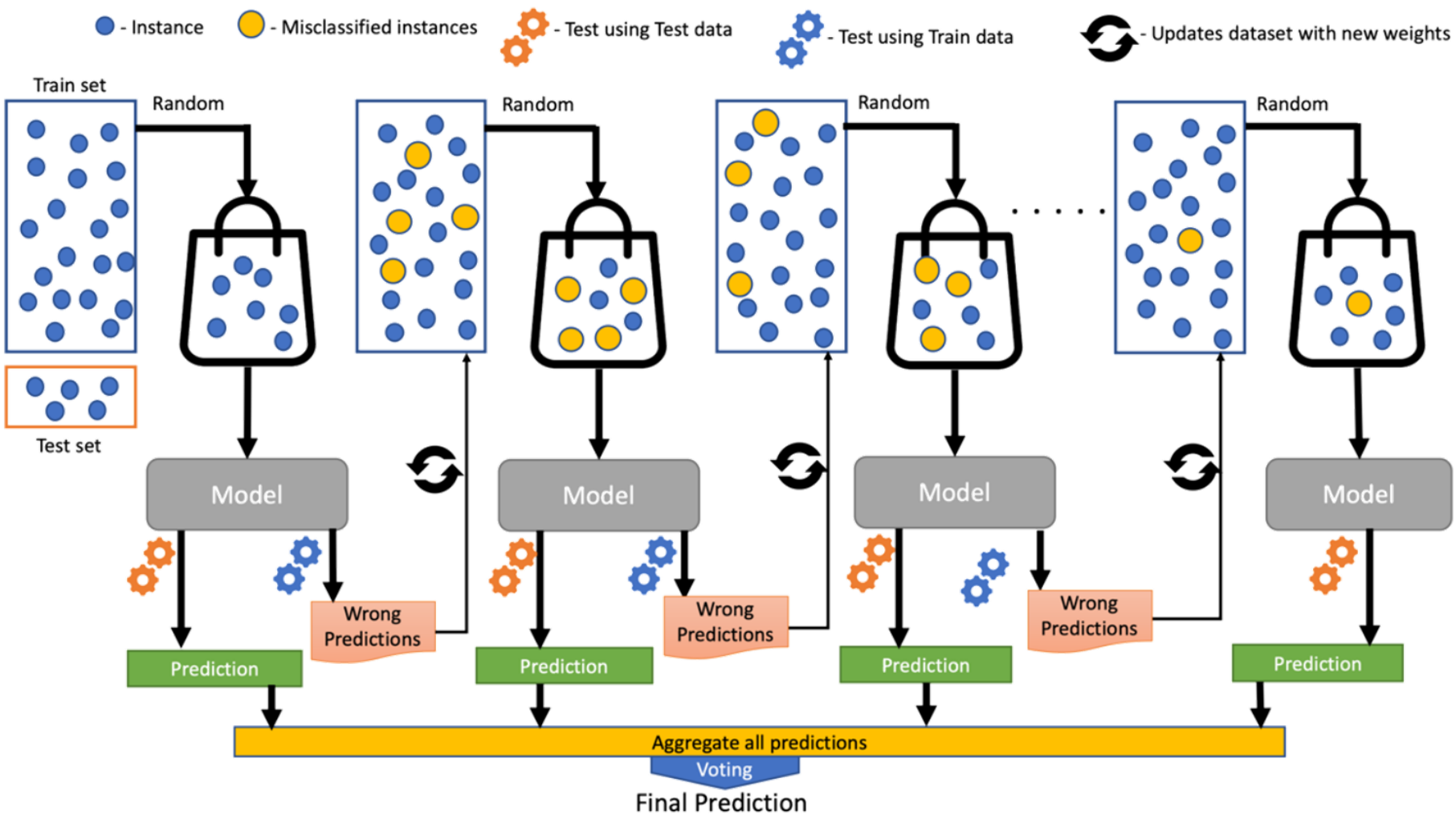

Dataset initial
Subset
Apprenant de base
Weak/base learner
Apprenant de base
Weak/base learner
Apprenant de base
Weak/base learner

Bagging


1
1
2
1
1
Classifier 1
Classifier 2
Classifier 3
Classifier 4
Sample data 1
Sample data 2
Sample data 3
Sample data 4
Training Data


1
Class 1, 2 [0.7, 0.3]
lass 1, 2 [0.2, 0.8]
Class 1, 2 [0.8, 0.2]
Class 1, 2 [0.9, 0.1]
Classifier 1
Classifier 2
Classifier 3
Classifier 4
Sample data 1
Sample data 2
Sample data 3
Sample data 4
Training Data

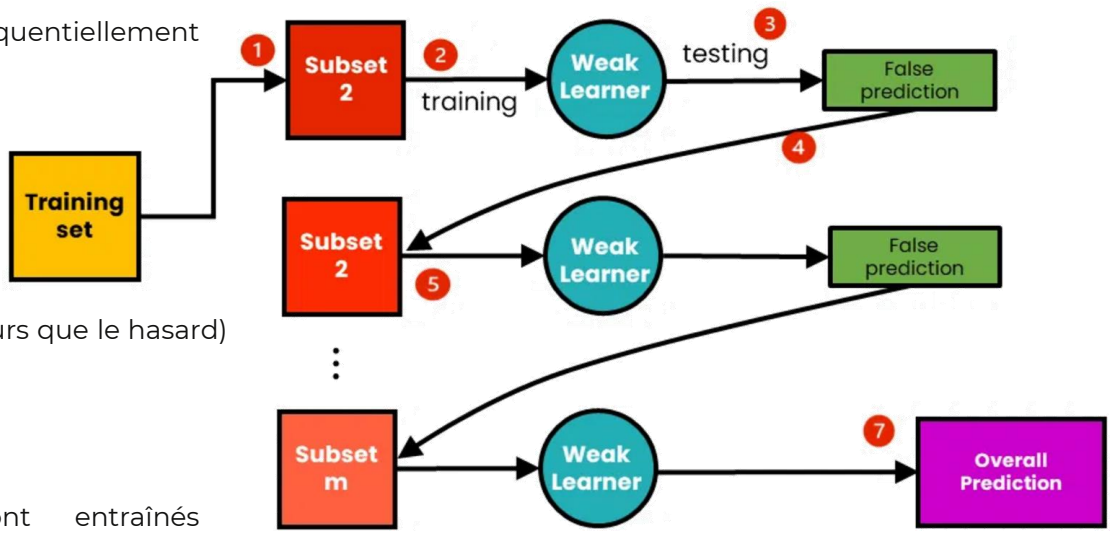
Boosting


- Instance
- Misclassified instances
- Test using Test data
- Test using Train data
- Updates dataset with new weights
Train set
Test set
Random
Random
Random
Random
Model
Model
Model
Model
Wrong Predictions
Wrong Predictions
Wrong Predictions
Prediction
Prediction
Prediction
Prediction
Aggregate all predictions
Voting
Final Prediction

Boosting

Qu'est-ce que le Boosting?

Méthode d'apprentissage ensembliste qui combine séquentiellement plusieurs modèles faibles pour créer un modèle fort.

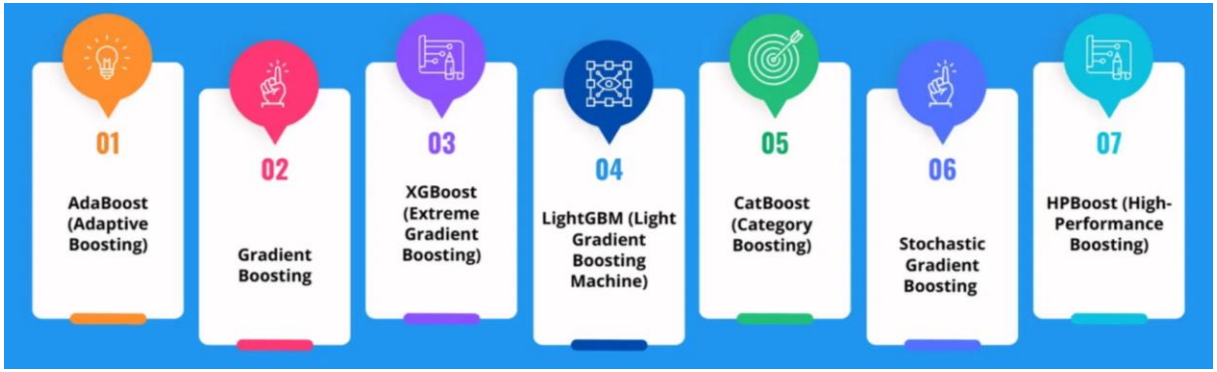


Objectif

Transformer des apprenants faibles (légèrement meilleurs que le hasard) en un apprenant fort avec une précision élevée.

Approche séquentielle

Contrairement au bagging, les modèles sont entraînés séquentiellement, chacun tentant de corriger les erreurs du précédent.



Boosting

1. **Initialisation**

   Attribution de poids égaux à toutes les observations du jeu de données d'entraînement.

2. **Entraînement d'un modèle faible**

   Entraînement d'un apprenant faible (souvent un arbre de décision peu profond) sur les données pondérées.

   → **Rééchantillonnage adaptatif**

   Le boosting modifie la distribution des données d'entraînement à chaque itération en fonction des erreurs précédentes.

3. **Calcul de l'erreur et mise à jour des poids**

   Évaluation de la performance du modèle et augmentation des poids des observations mal classées.

   → **Importance des observations mal classées**

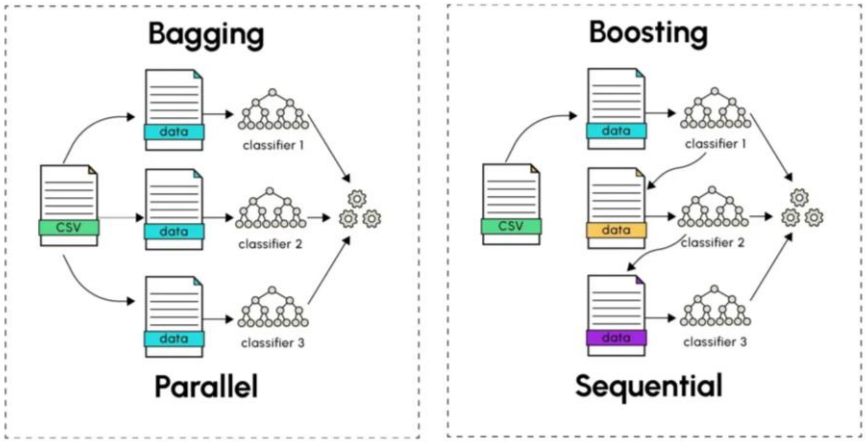
   Les observations difficiles à classer reçoivent progressivement plus de poids, forçant les modèles suivants à se concentrer sur elles.

4. **Combinaison des modèles**

   Après plusieurs itérations, combinaison pondérée des modèles faibles pour former le modèle final.

$$w_i^{(t+1)} = w_i^{(t)} \times \exp^{(\alpha_t \times I(y_i \neq h_t(X_i)))}$$

Bagging vs Boosting



Caractéristique	Bagging (ex: Random Forest)	Boosting (ex: AdaBoost, XGBoost)
Apprentissage	Parallèle (indépendant)	Séquentiel (dépendant)
Objectif principal	Réduction de la variance	Réduction du biais
Pondération	Égale pour tous les modèles	Basée sur la performance
Sensibilité au bruit	Faible	Élevée

MARRAKECH
جامعة القاضي عياض
UNIVERSITÉ CADI AYYAD

كلية العلوم
السملالية – مراكش
FACULTÉ DES SCIENCES
SEMLALIA – MARRAKECH

Random Forest – RF

Random Forest

Extension du Bagging

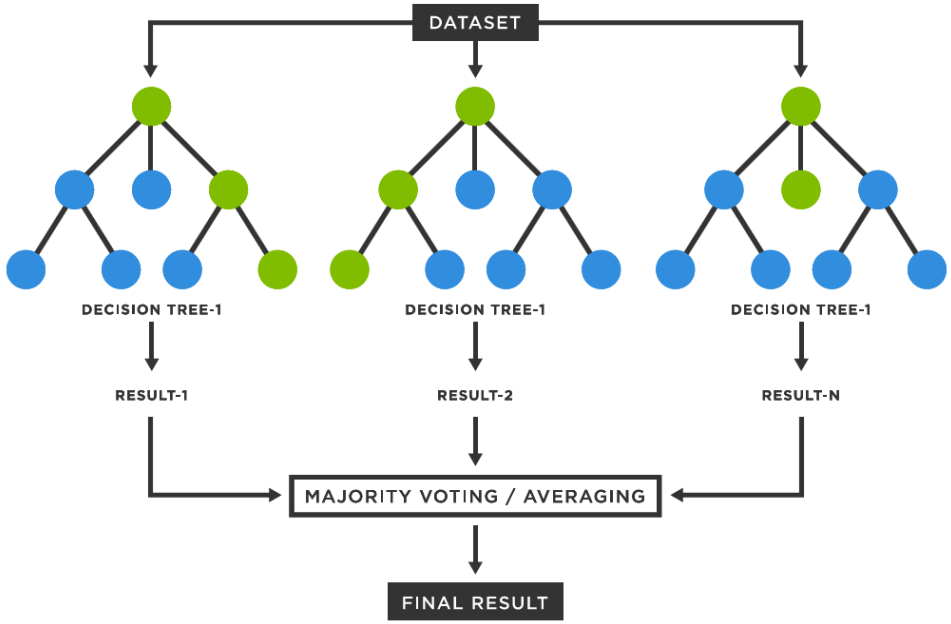
Random Forest est une extension du Bagging qui ajoute une couche supplémentaire de randomisation en sélectionnant aléatoirement un sous-ensemble de variables à chaque division d'arbre.

Sélection aléatoire des variables

À chaque nœud, seul un sous-ensemble aléatoire de m variables (typiquement $m \approx \sqrt{p}$, où p est le nombre total de variables) est considéré pour la division.

Arbres complets

Les arbres sont généralement développés à leur profondeur maximale sans élagage, permettant de capturer des relations complexes dans les données.



L'algorithme Random Forest est un algorithme d'apprentissage automatique supervisé. Il est capable d'effectuer à la fois des tâches de régression et de classification. Conformément à son nom, « Random Forest », il crée une forêt composée d'un certain nombre d'arbres de décision.

Random Forest

Étape 1 : Création d'un ensemble de données **bootstrappées**

échantillons de données avec remplacement

Dataset initial

Stinky	Green	Indoor	Age	Plant Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped dataset

Stinky	Green	Indoor	Age	Plant Disease

➔ Pour créer un ensemble de données « bootstrapped » de même taille que l'original, il suffit de sélectionner des échantillons dans l'ensemble de données original.

⚠ Nous sommes autorisés à choisir le même échantillon « exemplaire » plus d'une fois

Random Forest

Étape 2 : créer un **arbre de décision** à l'aide de l'ensemble de données **bootstrappées**, tout en utilisant un **sous-ensemble aléatoire de variables** (ou de colonnes) à chaque étape.

➔ Dans cet exemple nous considérons 2 variables / étape

⚠ Il existe des règles pour bien choisir le nombre optimal des variables a considérer

Bootstrapped dataset

Stinky	Green	Indoor	Age	Plant Disease
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes
Yes	Yes	Yes	180	Yes

➔ Donc, au lieu de considérer les 4 variables pour déterminer comment diviser le nœud racine …

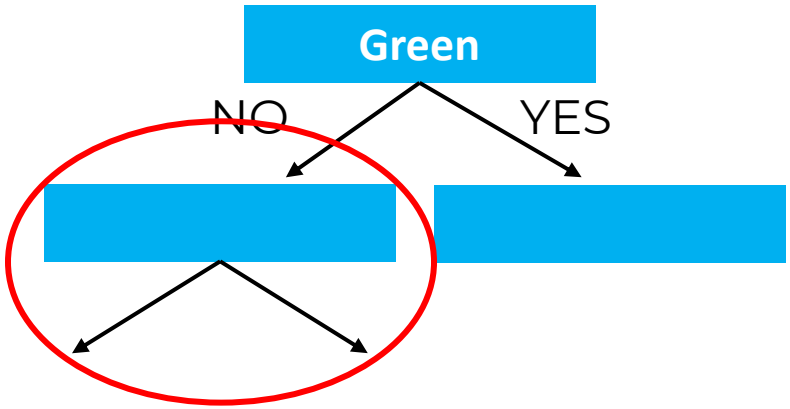
… nous sélectionnons au hasard 2

Random Forest

Étape 2 : créer un **arbre de décision** à l'aide de l'ensemble de données **bootstrappées**, tout en utilisant un **sous-ensemble aléatoire de variables** (ou de colonnes) à chaque étape.

Bootstrapped dataset

Stinky	Green	Indoor	Age	Plant Disease
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes
Yes	Yes	Yes	180	Yes



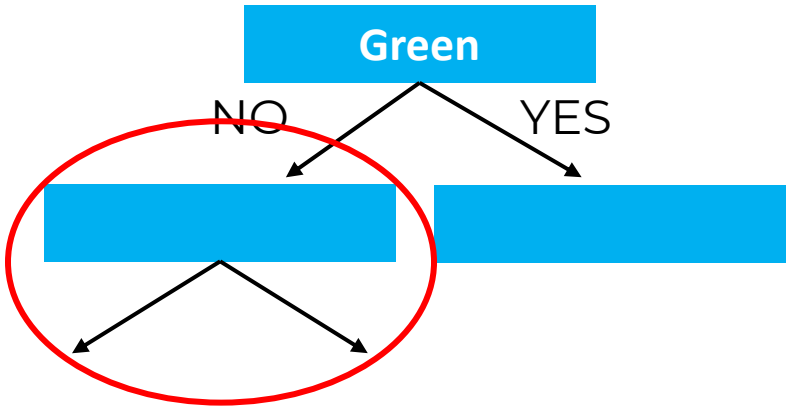
➔ Maintenant, nous devons déterminer comment diviser les échantillons au niveau de ce nœud

Random Forest

Étape 2 : créer un **arbre de décision** à l'aide de l'ensemble de données **bootstrappées**, tout en utilisant un **sous-ensemble aléatoire de variables** (ou de colonnes) à chaque étape.

Bootstrapped dataset

Stinky	Green	Indoor	Age	Plant Disease
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes
Yes	Yes	Yes	180	Yes



➔ De la même manière que pour la racine, nous sélectionnons au hasard 2 variables comme candidates, au lieu des 3 colonnes restantes.

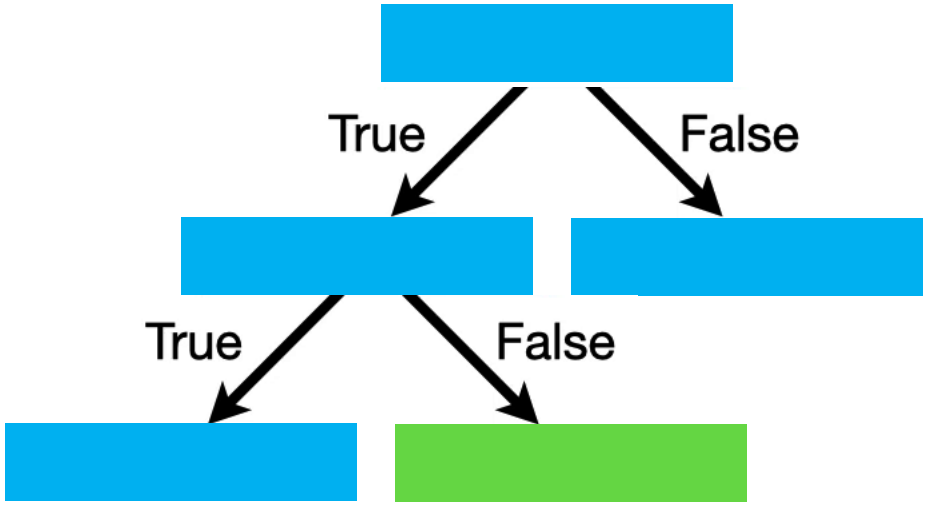
Random Forest

Étape 2 : créer un **arbre de décision** à l'aide de l'ensemble de données **bootstrappées**, tout en utilisant un **sous-ensemble aléatoire de variables** (ou de colonnes) à chaque étape.

Bootstrapped dataset

Stinky	Green	Indoor	Age	Plant Disease
Yes	Yes	Yes	180	Yes
Yes	Yes	Yes	180	Yes

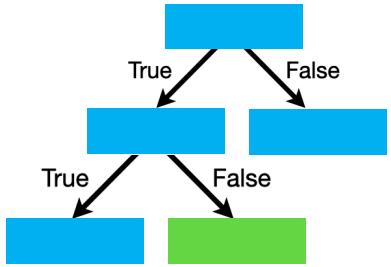
➔ Et nous construisons l'arbre comme d'habitude, mais en ne considérant qu'un sous-ensemble aléatoire de variables à chaque étape



Random Forest

Récap : nous construisons un arbre :

1. à l'aide d'un ensemble de données bootstrapées
2. en ne considérant qu'un sous-ensemble aléatoire de variables à chaque étape

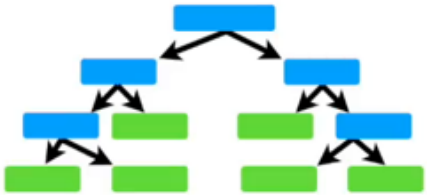


Bootstrapped dataset

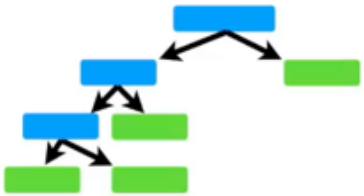
Stinky	Green	Indoor	Age	Plant Disease
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes
Yes	Yes	Yes	180	Yes

Ensuite, revenez à **l'étape 1** et répétez l'opération : créez un nouvel ensemble de données bootstrappé et construisez un arbre en considérant un sous-ensemble de variables à chaque étape.

➔ idéalement, vous feriez cela des centaines de fois, mais ici, juste pour avoir une idée



La variété est ce qui rend les algorithmes RF plus efficaces que les arbres de décision individuels.



Comment l'utiliser ?

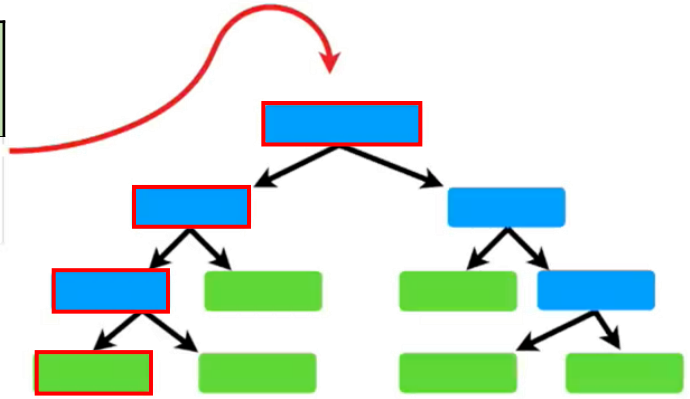
Random Forest

Pour commencer, nous recevons une nouvelle plante

dont nous avons pris toutes les mesures ...

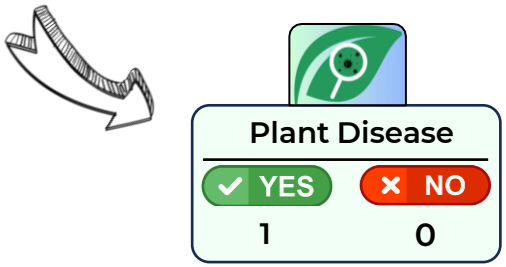
et nous souhaitons savoir si elle est malade ou non.

Stinky	Green	Indoor	Age	Plant Disease
Yes	No	No	168	???



➔ Nous prenons donc les données et nous les faisons descendre dans le premier arbre que nous avons construit ...

le premier arbre dit « oui »



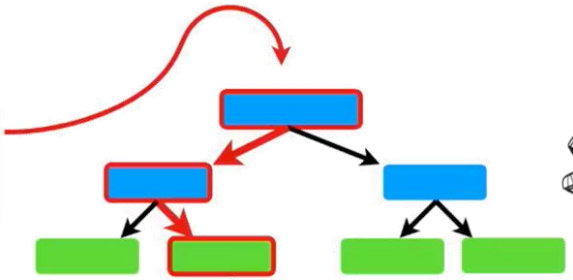
Nous gardons le résultat ici

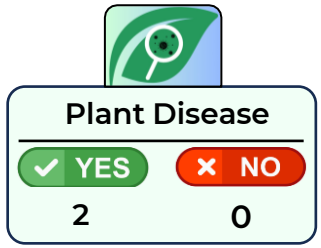
Random Forest

➔ Nous prenons donc les données et nous les faisons descendre dans le 2ème arbre …

… 2ème arbre dit aussi « oui »

Stinky	Green	Indoor	Age	Plant Disease
Yes	No	No	168	???

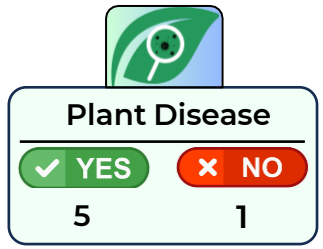




Nous gardons le résultat ici

➔ Après avoir parcouru les données dans tous les arbres de la forêt aléatoire, nous voyons quelle est l'option qui recueille le plus de votes.

Stinky	Green	Indoor	Age	Plant Disease
YES	NO	NO	168	YES



➔ Dans ce cas, c'est le « YES » qui a recueilli le plus grand nombre de votes, ce qui nous permet de conclure que cette plante est malade.

Random Forest

nous avons autorisé des doublons dans l'ensemble de données bootstrappées ...

... Par conséquent, cette entrée n'a pas été intégrée dans l'ensemble de données bootstrappé.

➔ En général, environ 1/3 des données originales ne se retrouve pas dans l'ensemble de données obtenu.

Bootstrapped dataset

Stinky	Green	Indoor	Age	Plant Disease
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes
Yes	Yes	Yes	180	Yes

Dataset initial

Stinky	Green	Indoor	Age	Plant Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

L'entrée qui n'a pas été retenue dans l'ensemble de données bootstrappé. On l'appelle « Out-Of-Bag Dataset »

Out-Of-Bag Dataset

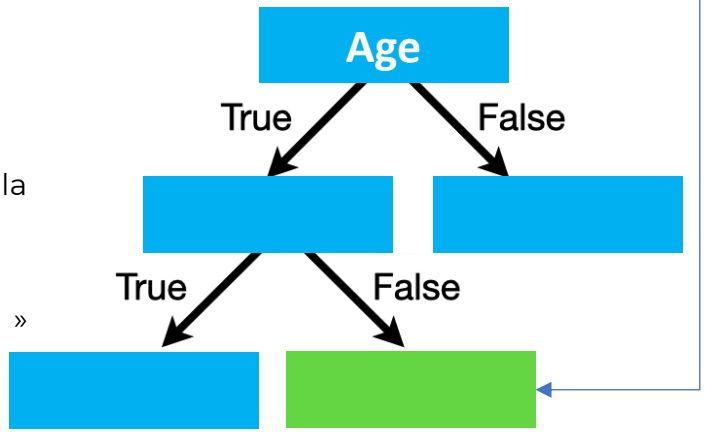
Stinky	Green	Indoor	Age	Plant Disease
No	No	No	125	No

Random Forest

L'entrée qui n'a pas été retenue dans l'ensemble de données bootstrappé. On l'appelle « Out-Of-Bag Dataset »

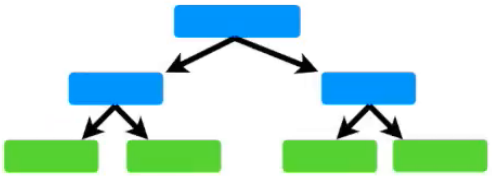
Out-Of-Bag Dataset

Stinky	Green	Indoor	Age	Plant Disease
No	No	No	125	No

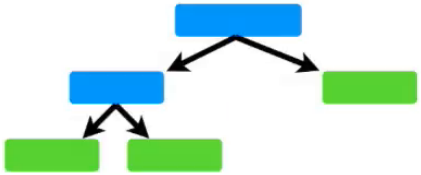


- Vu que les données Out-Of-Bag n'ont pas été utilisées pour créer cet arbre
- Nous utilisons cet arbre Et voir s'il permet de classer l'échantillon dans la catégorie « Absence de maladie des plantes »
- Dans ce cas, l'arbre étiquette correctement l'échantillon « Out of Bag » comme étant « No ».

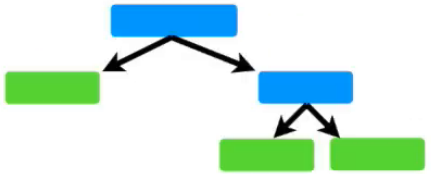
Puis nous passons cet échantillon « out of bag » à travers tous les autres arbres qui ont été construits sans lui.



cet arbre a incorrectement étiqueté l'échantillon



cet arbre a correctement étiqueté l'échantillon



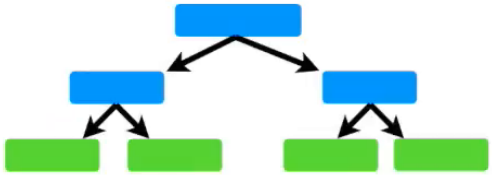
cet arbre a correctement étiqueté l'échantillon

Random Forest

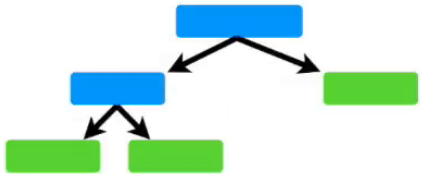
Out-Of-Bag Dataset

Stinky	Green	Indoor	Age	Plant Disease
No	No	No	125	No

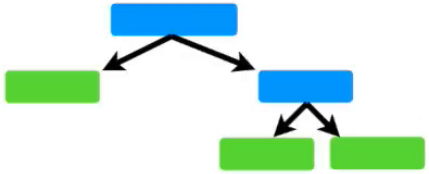
Puis nous passons cet échantillon « out of bag » à travers tous les autres arbres qui ont été construits sans lui.



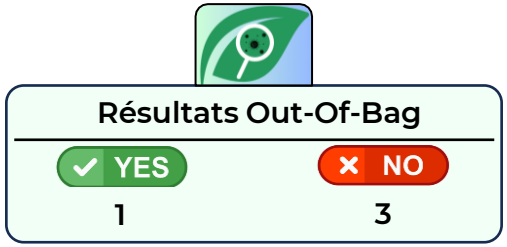
cet arbre a incorrectement étiqueté l'échantillon



cet arbre a correctement étiqueté l'échantillon



cet arbre a correctement étiqueté l'échantillon



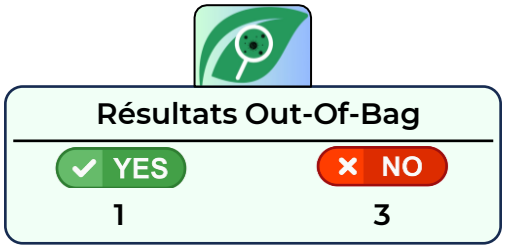
- ✓ Puisque l'étiquette qui obtient le plus grand nombre de votes gagne, c'est l'étiquette que nous attribuons à l'échantillon out-of-bag
- ✓ Dans ce cas, l'échantillon out-of-bag est correctement étiqueté par la RF.

Random Forest

Out-Of-Bag Dataset

Stinky	Green	Indoor	Age	Plant Disease
No	No	No	125	No

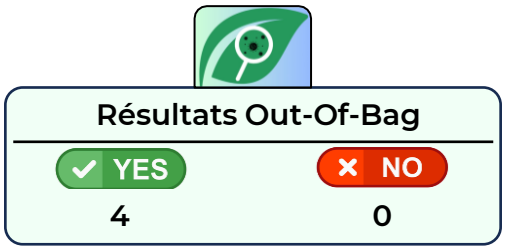
✓ nous faisons ensuite la même chose pour tous les autres échantillons « out of bag » pour tous les arbres.



Out-Of-Bag Dataset

Stinky	Green	Indoor	Age	Plant Disease
YES	YES	YES	180	YES

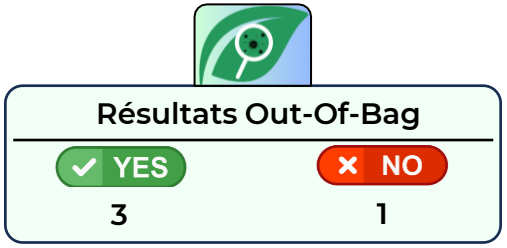
✓ Cet échantillon a également été correctement étiqueté



Out-Of-Bag Dataset

Stinky	Green	Indoor	Age	Plant Disease
NO	NO	NO	125	NO

✓ Cet échantillon a été incorrectement étiqueté.



➔ En fin de compte, nous pouvons mesurer la précision de notre RF par la proportion d'échantillons Out-Of-Bag qui ont été correctement classés.

➔ La proportion d'échantillons Out-Of-Bag qui ont été incorrectement classés est « l'erreur Out-Of-Bag »

Random Forest

```
from sklearn.ensemble import RandomForestClassifier
RC = RandomForestClassifier(n_estimators=100, max_features=10)
RC = RC.fit(X_train, y_train)
y_predict = RC.predict(X_test)
```

Random Forest

1. **Chargement des données :**
   Importer et charger les jeux de données que vous avez choisis pour le mini projet
2. **Prétraitement des données :** préparer et diviser les données pour la modélisation.
3. **Entraînement du modèle :** entraîner deux modèles du Random Forest à l'aide des données prétraitées. Un pour la régression et l'autre pour la classification
4. **Évaluation du modèle :** Évaluer les performances en calculant les indices de performances correspondant de chaque modèle entraîné et effectuer des ajustements.
5. **Sauvegarde du modèle entraîné.**

Questions à réfléchir

1. Quel modèle donne la meilleure précision entre Bagging et RF ? Pourquoi ?
2. En quoi RF améliore le simple Bagging ?
3. Essayez de visualiser l'importance des features (feature_importances_) avec le modèle RF.
4. Changez le nombre d'estimateurs (par exemple 10, 100, 500) pour observer l'impact.