

Les méthodes d'ensemble

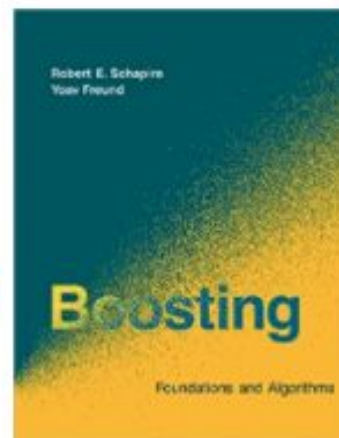
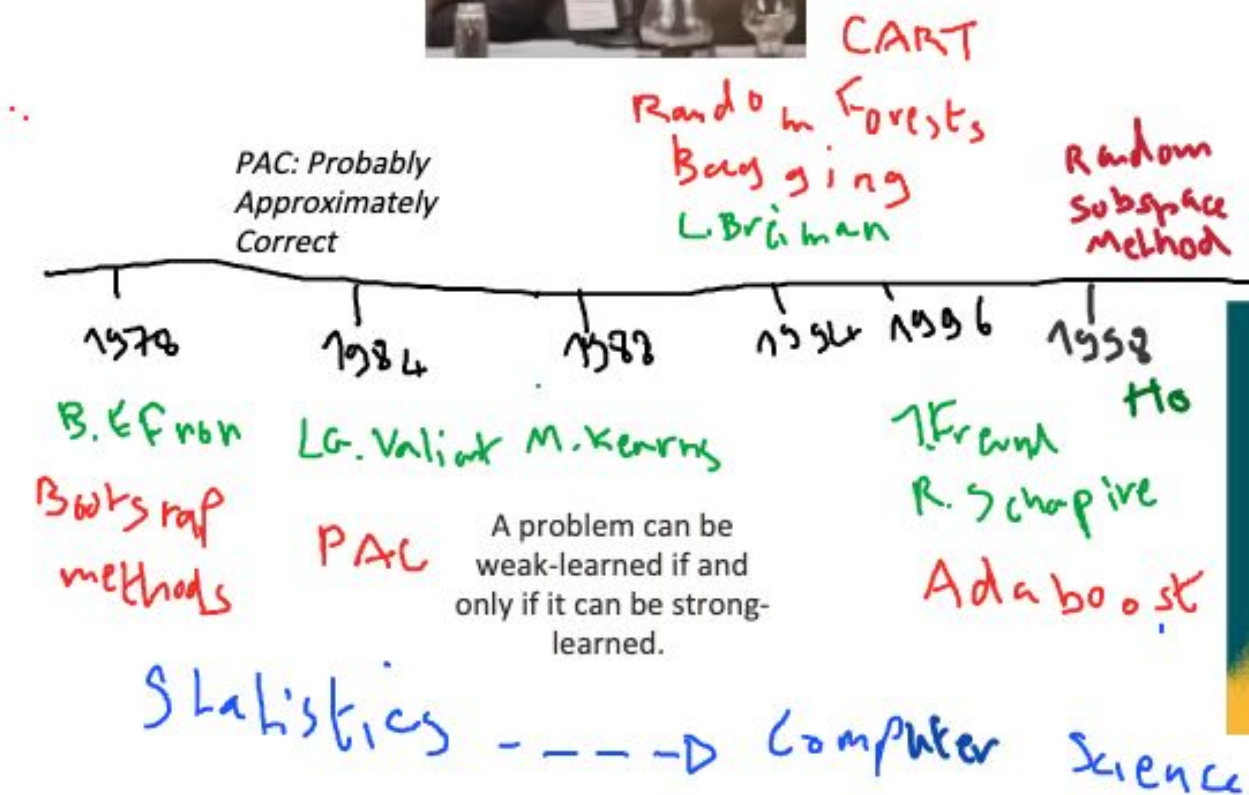
Christelle, Laurent, Stéphanie, Thibault

Random Forest et Régression...

SOMMAIRE

- Quel est le principe général des méthodes d'ensemble ?
- Que sont le "Bagging" et le "Pasting" ?
- Qu'est ce que l'évaluation "Out-Of-Bag" ?
- Qu'est ce que la méthode de "Random Patches" ?
- Qu'est ce que la méthode de "Random Subspaces" ?
- Décrire plus particulièrement ce qu'est une RandomForest et lesquels des concepts présentés précédemment sont utilisés par ce modèle.
- Expliciter enfin l'ensemble des paramètres de la fonction "RandomForestRegressor" de la librairie Scikit-Learn.

Histoire



Principe général des méthodes d'ensemble

La théorie des méthodes ensemblistes :
améliorer des apprenants faibles en les **combinants**.

apprenant faible > modèle aléatoire.

prédicteurs très différents



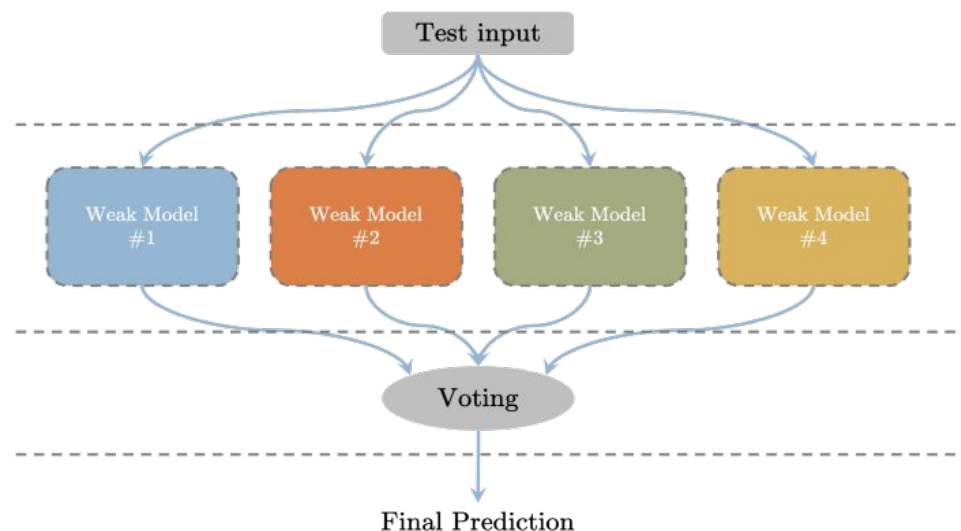
erreurs très différentes



mêmes erreurs peu probables



précision de l'ensemble des prédicteurs améliorée



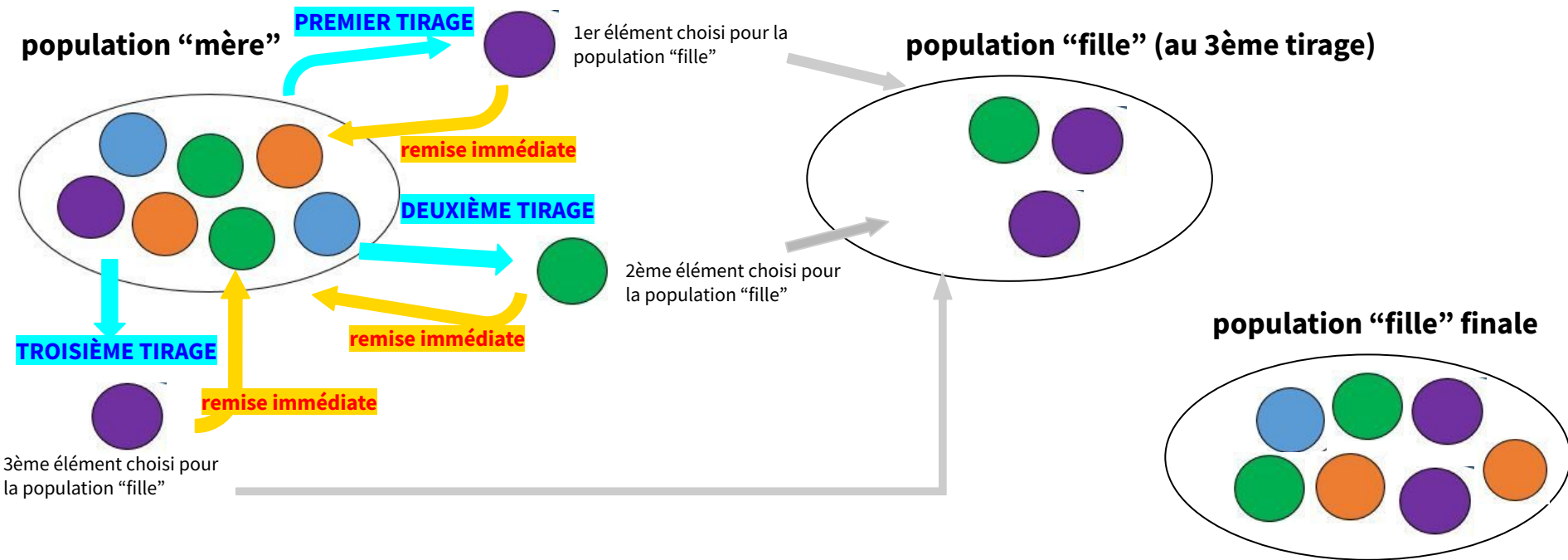
Exemple : – Estimation du poids d'un panier dans un marché

787 participants

- Le meilleur = plus d'un centième d'erreur
- Moyenne : moins d'un millième d'erreur

[Francis Galton¹, 1906 (85 ans)]

Petit rappel : l'échantillonnage avec remise



Le **bootstrap** applique ce processus pour obtenir une population "fille" qui peut avoir la même taille N que la population "mère" et ce pour chaque apprenant faible.

le "Bagging" (Bootstrap Aggregation)

Parmi les méthodes d'ensemble parallèles, il y a le **bagging** (voire d'autres exemples à la fin)

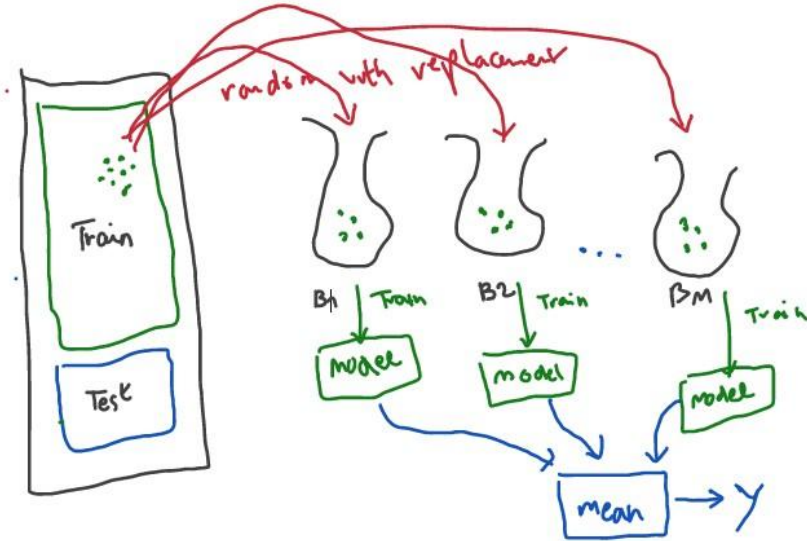
Le bagging combine des apprenants faibles(=overfitting ou underfitting), en créant un subset différent pour chacun d'eux et ainsi obtenir des apprenants très différents avec des erreurs très différentes.

La création des subset par **bootstrap** :

- échantillonnage avec remise,
- les subsets peuvent être de même taille que le dataset original.

Après avoir entraîné les apprenants faibles, la prédiction sera effectuée par :

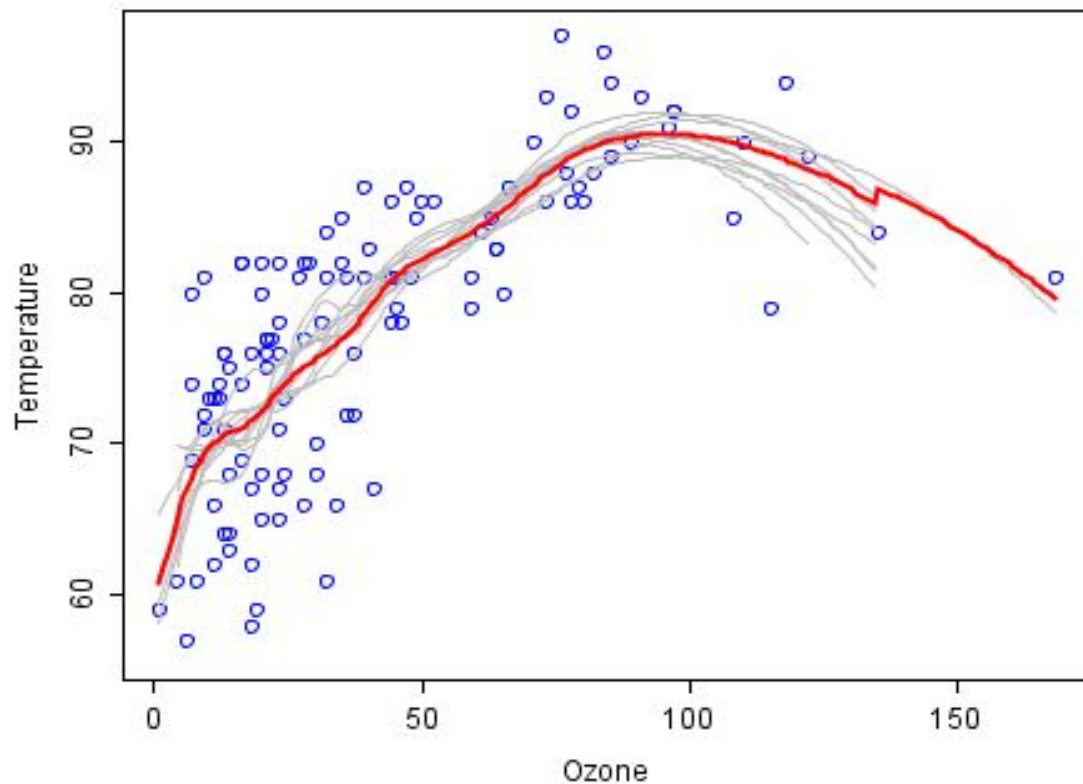
- **vote à la majorité pour la classification,**
- **calcul de la moyenne des résultats pour la régression.**



plus on a d'apprenants faibles (=avec une forte variance), plus on aura une prédiction performante et stable.

Le bagging donne d'excellents résultats, en particulier avec la random forest.

le "Bagging", exemple sur une régression



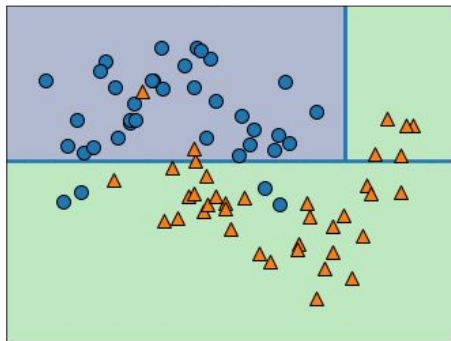
bleu = les données,

gris = les régressions des apprenants faibles,

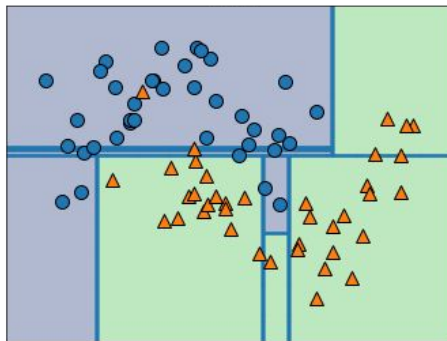
rouge = la régression finale (=moyenne des régressions des apprenants faibles)

le "Bagging", exemple de la fonction `BaggingClassifier` (`bootstrap default=True`) => **bagging**
de sklearn, avec 5 estimateurs (=apprenants faibles) et le résultat du bagging

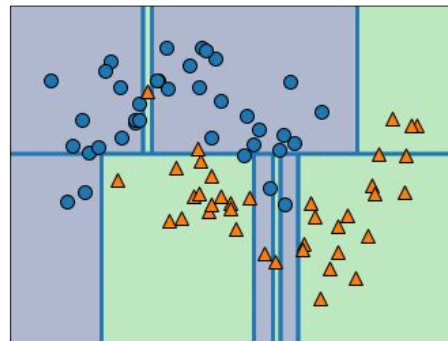
Tree 0



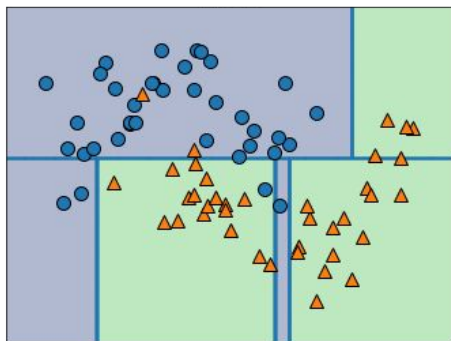
Tree 1



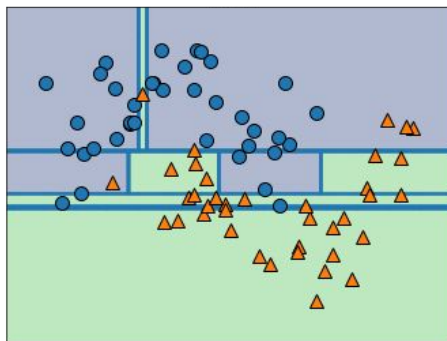
Tree 2



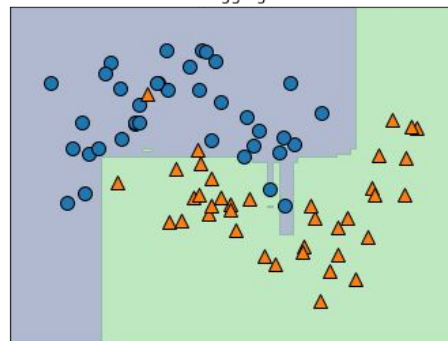
Tree 3



Tree 4



Bagging



Les erreurs des différents classifieurs (ici, overfitting) sont toutes différentes.

Elles sont gommées par le bagging qui les combine entre elles (ici par le vote des 5 classifieurs) pour obtenir le classifieur final.

le "Pasting"

Le **pasting** va permettre de générer des subsets en utilisant cette fois-ci **l'échantillonnage SANS remise** (ou sans remplacement).

L'échantillonnage SANS remise :

Un élément E d'une population "mère" de taille N est choisi pour une population "fille" **ET** est **retiré** immédiatement de la population "mère" **AVANT** de choisir le prochain élément.

De sorte que cet élément E ne peut être choisi qu'une seule fois au maximum.

La population "fille" sera au maximum de taille N-1.

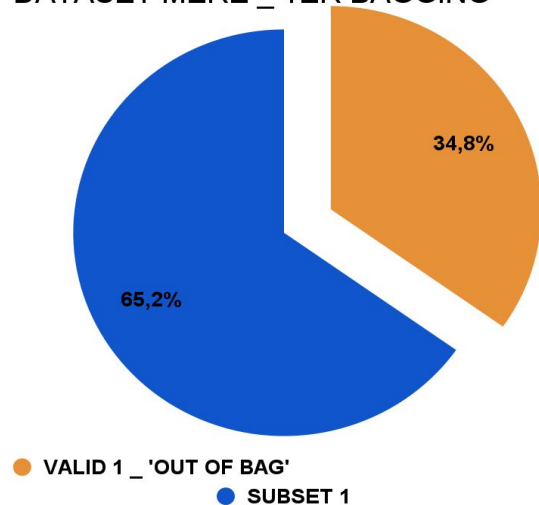
BaggingClassifier (bootstrap default=False) => *pasting*

Qu'est ce que l'évaluation "Out-Of-Bag" ?

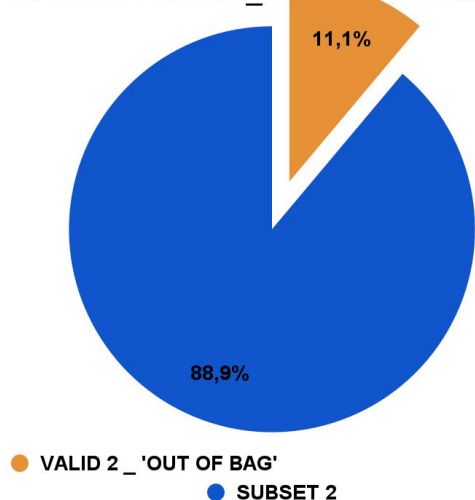
Lors d'un **bagging** ➤ environ 37% du dataset mère "laissés de côté"

➤ Ces données "out-of-bag" vont être **utilisées pour faire un set de validation de chaque subset**

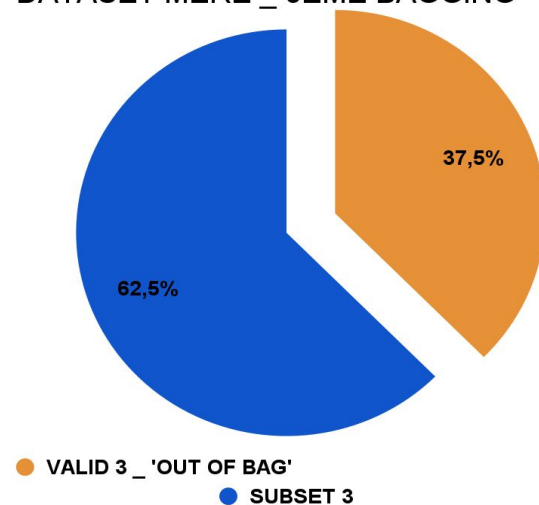
DATASET MÈRE _ 1ER BAGGING



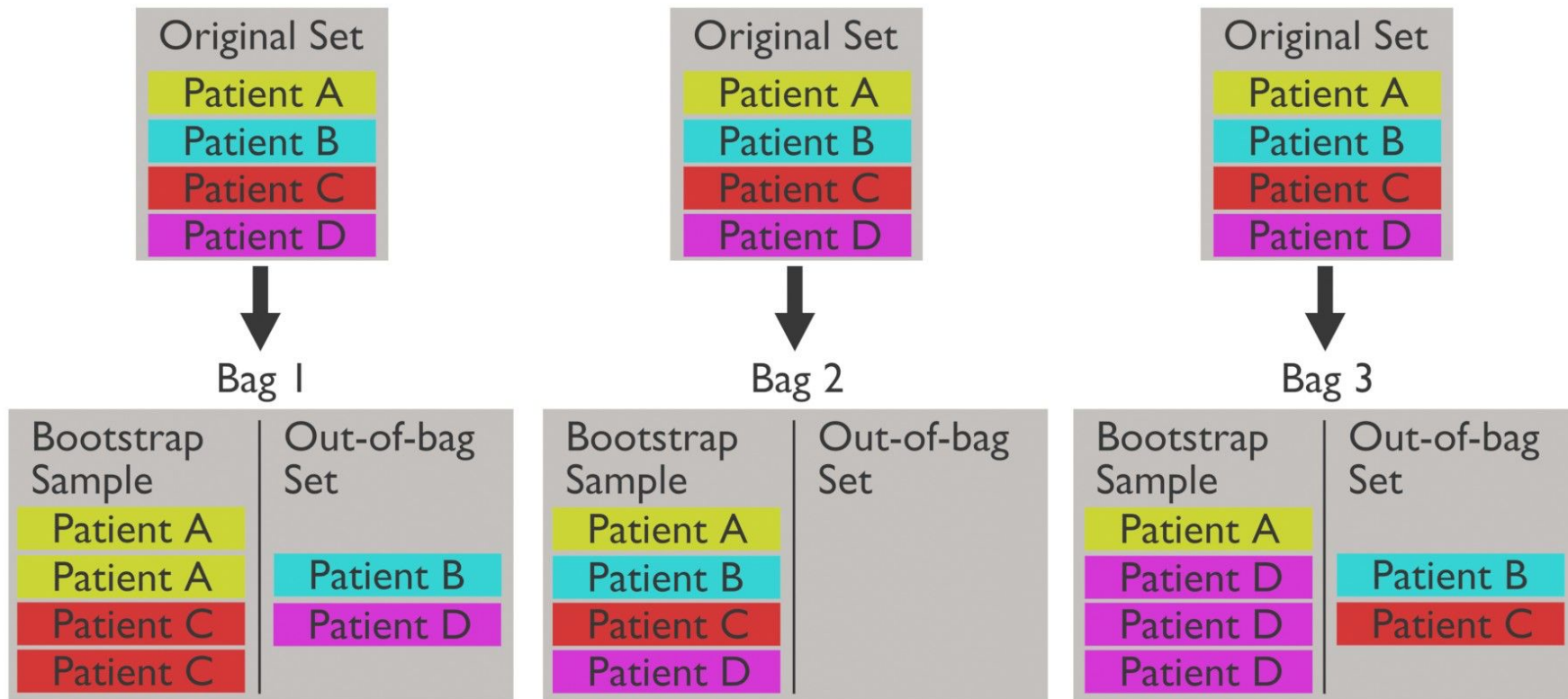
DATASET MÈRE _ 2EME BAGGING



DATASET MÈRE _ 3EME BAGGING



Exemple d'évaluation "Out-Of-Bag"



The Random Subspace Method

"Une grande partie de l'intérêt porté aux arbres de décision se concentre sur les critères de division et l'optimisation de la taille des arbres. Le dilemme entre le surapprentissage et l'obtention d'une précision maximale est rarement résolu. Une méthode pour construire un classificateur basé sur un arbre de décision est proposée qui maintient la plus haute précision sur les données d'apprentissage et améliore la précision de la généralisation à mesure qu'elle augmente en complexité. Le classificateur consiste en plusieurs arbres construits systématiquement en sélectionnant de manière pseudo-aléatoire des sous-ensembles de composants du vecteur de caractéristiques, c'est-à-dire des arbres construits dans des sous-espaces choisis au hasard. La méthode du sous-espace est comparée aux classificateurs à arbre unique et à d'autres méthodes de construction de forêts par des expériences sur des ensembles de données accessibles au public, où la supériorité de la méthode est démontrée. Nous discutons également de l'indépendance entre les arbres d'une forêt et la relation à la précision de la classification combinée." (Ho, 1998)

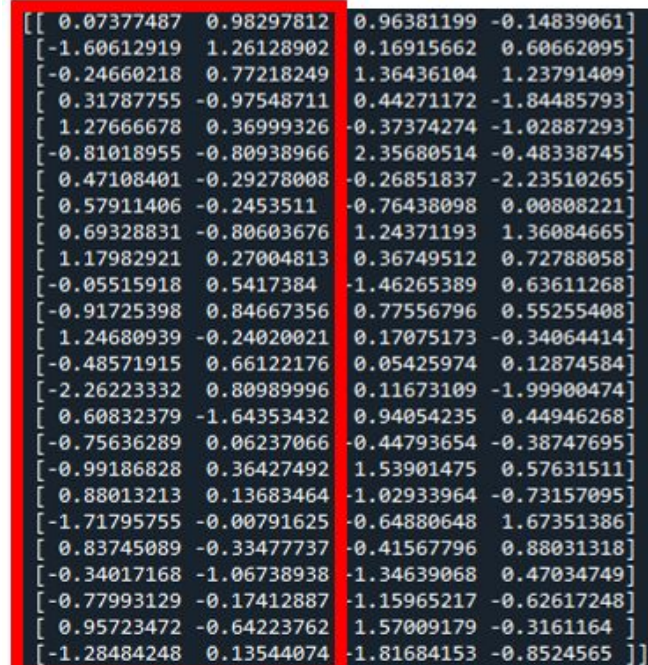
Qu'est ce que la méthode de "Random Subspaces" ?

Son but : introduire de la variance entre les apprenants faibles, pour augmenter les performances du modèle d'ensemble.

Comment : en diminuant la corrélation entre les apprenants :

- Création des **sous-ensembles/subsets** ne contenant **que certaines caractéristiques**(par échantillonnage avec remise)

La méthode de Random Subspaces est également appelée *attribut bagging* ou *feature bagging*.



[0.07377487	0.98297812	0.96381199	-0.14839061]
[-1.60612919	1.26128902	0.16915662	0.60662095]
[-0.24660218	0.77218249	1.36436104	1.23791409]
[0.31787755	-0.97548711	0.44271172	-1.84485793]
[1.27666678	0.36999326	-0.37374274	-1.02887293]
[-0.81018955	-0.80938966	2.35680514	-0.48338745]
[0.47108401	-0.29278008	-0.26851837	-2.23510265]
[0.57911406	-0.2453511	-0.76438098	0.00808221]
[0.69328831	-0.80603676	1.24371193	1.36084665]
[1.17982921	0.27004813	0.36749512	0.72788058]
[-0.05515918	0.5417384	-1.46265389	0.63611268]
[-0.91725398	0.84667356	0.77556796	0.55255408]
[1.24680939	-0.24020021	0.17075173	-0.34064414]
[-0.48571915	0.66122176	0.05425974	0.12874584]
[-2.26223332	0.80989996	0.11673109	-1.99900474]
[0.60832379	-1.64353432	0.94054235	0.44946268]
[-0.75636289	0.06237066	-0.44793654	-0.38747695]
[-0.99186828	0.36427492	1.53901475	0.57631511]
[0.88013213	0.13683464	-1.02933964	-0.73157095]
[-1.71795755	-0.00791625	-0.64880648	1.67351386]
[0.83745089	-0.33477737	-0.41567796	0.88031318]
[-0.34017168	-1.06738938	-1.34639068	0.47034749]
[-0.77993129	-0.17412887	-1.15965217	-0.62617248]
[0.95723472	-0.64223762	1.57009179	-0.3161164]
[-1.28484248	0.13544074	-1.81684153	-0.8524565]

Figure 1. Une illustration d'un ensemble de données créé par la méthode de sous-espace aléatoire.

Qu'est ce que la méthode de "Random Patches" ?

Random Patches est une méthode de random Subspace qui est utilisée avec le Pasting ou le Bagging

Le terme Random Patches désigne la méthode qui combine l'utilisation de Random Subset avec le Bagging ou le Pasting dans une méthode d'ensemble parallèle.

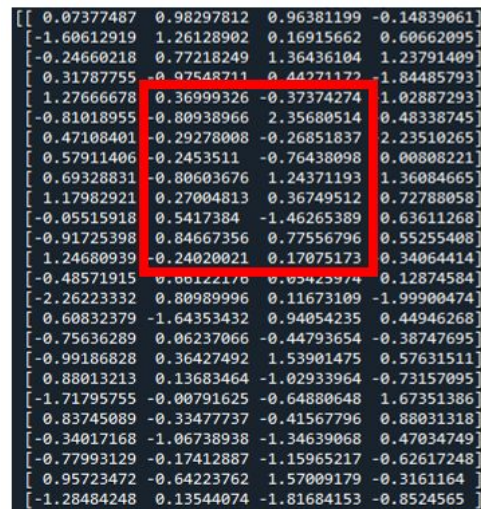


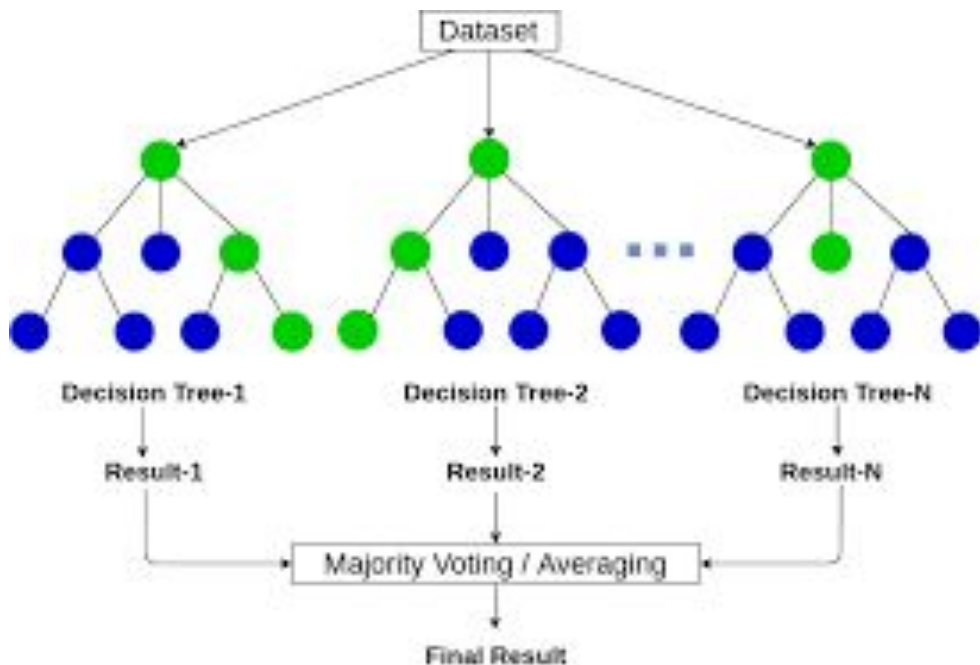
Figure 2. Illustration d'un ensemble de données créé par la méthode des patches aléatoires.

Décrire plus particulièrement ce qu'est une RandomForest et lesquels des concepts présentés précédemment sont utilisés par ce modèle.

Le Random Forest est un algorithme utilisant plusieurs arbres de décision.

Chaque arbre est indépendant des autres et produit une prédiction

Random Forest utilise le bagging via le `n_estimators`, et Random subspaces via le `max_features`., une cross validation via `oob_score`



Hyperparamètres

max_depth

Par défaut, les arbres croissent jusqu'à ce que toutes ses feuilles soient pures ou contiennent moins d'échantillons que min_samples_leaf (soit 2 par défaut).

Il est question ici de contrôler le surapprentissage des arbres.

n_estimators

Plus d'arbres signifie plus de précision au détriment de la rapidité d'apprentissage.

Le choix est ici une question de puissance et de temps disponibles

max_features

Spécifie le nombre maximum de variables indépendantes à sélectionner au hasard à chaque split de l'arbre.

On peut se guider sur le principe suivant : si les variables sont hautement corrélées, on peut vouloir diminuer ce nombre. Au contraire si les variables sont très décorélées et que notre modèle souffre d'un manque de précision, on voudra augmenter celui-ci. Sqrt, 0.5 et log2 sont de bons candidats pour ce paramètre.

warm_start

Pour maximiser la précision, le nombre d'estimateurs devraient être le plus élevé possible. Malgré tout, on aimerait utiliser une valeur raisonnable afin de limiter les temps d'entraînement et de prédiction.

Le `warm_starting` est disponible pour plusieurs modèles de Scikit-Learn en regard parfois d'aspect différents. Il s'agit de sauvegarder les attributs des modèles au cours de la phase d'apprentissage pour les réutiliser, typiquement dans un `gridSearch` mais aussi :

```
rfc = RandomForestClassifier(n_estimators=10, warm_start=True)
rfc.fit(X[:50], y[:50])
rfc.n_estimators += 10
rfc.fit(X[51:100], y[51:100])
rfc.n_estimators += 10
rfc.fit(X[101:150], y[101:150])
```

Pour une recherche par grille, spécifier 200 et 500 comme valeurs de `n_estimators` reviendra à réutiliser les attributs découverts pour les 200 premiers arbres avec les 300 autres nécessaires pour atteindre la valeur 500.

criterion

Critère de sélection des features lors d'un split.
Par défaut, on va chercher à diminuer la variance en utilisant `squared_error` (`absolute_error` sera recommandée pour les `TimeSeries`).
La réduction de la déviance de Poisson sera elle adaptée aux problèmes de dénombrements.

max_samples

Il n'est pas nécessaire de donner à chaque arbre les données complètes. Les performances du modèle atteignent souvent leur maximum lorsque les données fournies sont inférieures à 20% de l'ensemble de données d'origine suivant leur distribution initiale. C'est assez étonnant !

random_state

Contrôle le caractère aléatoire du bootstrapping et de l'échantillonnage des caractéristiques

bootstrap

La tâche principale du statisticien est de mener des études basées sur un échantillon afin de généraliser les résultats à la population parente non ?
La méthode de bootstrap est activée par défaut.

oob_score

$\lim_{n \rightarrow \infty} (1 - 1/n)^n = 0.368$

Ainsi 36,8 % des données d'entraînement sont utilisées pour la validation de chaque arbre.
Ne peut être activée que si `bootstrap=True`

n_jobs

Nombre de processeurs alloués au calcul du modèle

Autres méthodes

Scheme	Architecture	Trainable	Adaptive
Voting	Parallel	No	No
Sum, mean, median	Parallel	No	No
Product, min, max	Parallel	No	No
Generalized ensemble	Parallel	Yes	No
Adaptive weighting	Parallel	Yes	Yes
Stacking	Parallel	Yes	No
Borda count	Parallel	Yes	No
Logistic regression	Parallel	Yes	No
Class set reduction	Parallel cascading	Yes/No	No
Dempster-Shafer	Parallel	Yes	No
Fuzzy integrals	Parallel	Yes	No
Mixture of local experts (MLE)	Gated parallel	Yes	Yes
Hierarchical MLE	Gated parallel hierarchical	Yes	Yes
Associative switch	Parallel	Yes	Yes
Bagging	Parallel	Yes	No
Boosting	Parallel hierarchical	Yes	No
Random subspace	Parallel	Yes	No
Neural trees	Hierarchical	Yes	No

Bibliographie

Méthodes ensemblistes :

<http://www2.agroparistech.fr/ufr-info/membres/cornuejols/Teaching/ENSTA/Tr-boosting-LSI-10x4.pdf>

Bagging:

<https://openclassrooms.com/fr/courses/4470521-modelisez-vos-donnees-avec-les-methodes-ensemblistes/4664687-controlez-la-variance-a-l-aide-du-bagging>

Out-of-bag :

<https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.3712&rep=rep1&type=pdf>

Random Forest:

<https://www.journaldunet.fr/web-tech/guide-de-l-intelligence-artificielle/1501905-random-forest-ou-foret-aleatoire-definition-et-cas-d-usage/>

<https://ai-pool.com/a/s/random-forests-understanding>

Random Subspaces et Random Patches:

<https://machinelearningjourney.com/index.php/2020/03/24/random-patches-and-random-subspaces/>

Criterion :

<https://mljar.com/blog/feature-importance-in-random-forest/>

Déviance de Poisson :

<https://peijin.medium.com/the-poisson-deviance-for-regression-d469b56959ce>

<https://ichi.pro/fr/un-guide-illustre-du-modele-de-regression-de-poisson-88840519899480>

http://www.perrin33.com/incertitudes/gum/ufc_poissong2.html

Bibliographie

BREIMAN, LEO. "Bagging Predictors," 1996.

Efron, B. "Bootstrap Methods: Another Look at the Jackknife." *The Annals of Statistics* 7, no. 1 (January 1979): 1–26.

<https://doi.org/10.1214/aos/1176344552>.

Freund, Yoav, and Robert E. Schapire. "A Short Introduction to Boosting," 1999.

———. "Experiments with a New Boosting Algorithm," 1996.

"Sewell - Ensemble Learning.Pdf." Accessed November 10, 2021.

<http://www.machine-learning.martinsewell.com/ensembles/ensemble-learning.pdf>.

Sewell, Martin. "Ensemble Learning," n.d., 16.

Statistics, Leo Breiman, and Leo Breiman. "Out-Of-Bag Estimation," n.d.

Tin Kam Ho. "The Random Subspace Method for Constructing Decision Forests." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, no. 8 (August 1998): 832–44. <https://doi.org/10.1109/34.709601>.

Valiant, L. G. "A Theory of the Learnable." *Communications of the ACM* 27, no. 11 (November 5, 1984): 1134–42.

<https://doi.org/10.1145/1968.1972>.

Vapnik, Vladimir N. "The Nature of Statistical Learning Theory," 1999.

Expliciter l'ensemble des paramètres de la fonction "RandomForestRegressor" de la librairie Scikit-Learn.

`n_estimators int, default=100`

=> nombre d'arbres dans la forêt

`criterion {"squared_error", "absolute_error", "poisson"}, default="squared_error"`

=> quelle mesure pour choisir la bonne caractéristique à création d'un noeud interne(=split), en général la sélection se fait sur la base de la diminution de la variance ("squared_error")

"squared_error": l'erreur quadratique moyenne, équivalente à utiliser la réduction de la variance en tant que caractéristique de sélection pour mesurer la qualité d'un split

"absolute_error": l'erreur absolue moyenne (moy. des différences entre les valeurs prédites et réelles), méthode d'entraînement plus lente que "squared_error", recommandée pour les TimeSeries

"poisson": utilise la réduction de la déviance de Poisson pour trouver des splits. adapté pour les problèmes de dénombrements (évolution des colonies bactériennes, nombre de buts d'une équipe par match en fonction d'être à domicile ou non)

`max_depth int, default=None`

=> profondeur de chaque arbre

"None": par défaut, on ne va pas définir de profondeur maximale car on cherche à obtenir des arbres overfittés

`max_features{"auto", "sqrt", "log2"}, int or float, default="auto"`

=> nombre de features considérées pour chaque split, le nombre de caractéristique examinés à chaque création d'un noeud

"default" = "auto" = "None" = "n_features"

"int" = on va regarder parmi l'ensemble de features à chaque split

"float" = max_features devient une fraction et c'est le nombre= $\text{round}(\text{max_features} * \text{n_features})$ de features considérées à chaque split

"sqrt" = $\text{max_features} = \text{sqrt}(\text{n_features})$

"log2" = $\text{max_features} = \text{log2}(\text{n_features})$

Expliciter l'ensemble des paramètres de la fonction "RandomForestRegressor" de la librairie Scikit-Learn.

bootstrap bool, default=True

Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree.

oob_score bool, default=False

Whether to use out-of-bag samples to estimate the generalization score. Only available if bootstrap=True.

n_jobs int, default=None

The number of jobs to run in parallel. `fit`, `predict`, `decision_path` and `apply` are all parallelized over the trees. `None` means 1 unless in a `joblib.parallel_backend` context. `-1` means using all processors. See [Glossary](#) for more details.

random_state int, RandomState instance or None, default=None

Controls both the randomness of the bootstrapping of the samples used when building trees (if `bootstrap=True`) and the sampling of the features to consider when looking for the best split at each node (if `max_features < n_features`). See [Glossary](#) for details.

warm_start bool, default=False

When set to `True`, reuse the solution of the previous call to fit and add more estimators to the ensemble, otherwise, just fit a whole new forest. See [the Glossary](#).

max_samples int or float, default=None

If bootstrap is True, the number of samples to draw from X to train each base estimator.

- If None (default), then draw `X.shape[0]` samples.
- If int, then draw `max_samples` samples.
- If float, then draw `max_samples * X.shape[0]` samples. Thus, `max_samples` should be in the interval `(0.0, 1.0]`.

New in version 0.22.