TER

Etude d'un algorithme de Machine Learning

Léo LAMOUREUX
Lola PIRES PINTO

Encadrants : Luc Bouganim et Ludovic Javet

Master 1
Année 2021 - 2022

Table des matières

1 Introduction

L'objectif de ce TER est de réaliser l'étude d'un algorithme de Machine Learning afin d'en améliorer nos connaissances. Nous avons choisi de travailler sur un algorithme de type Random Forest (Forêt aléatoire), qui est applicable à des problèmes de classification (RandomForestClassifier) comme à des problèmes de régression (RandomForestRegressor).
Ayant étudié les structures arborescentes durant notre cursus, et sachant que c'est un outil très utilisé dans beaucoup de domaines, nous étions d'abord intéressés par les arbres de décision. Mais nous avons ensuite découvert les algorithmes de type forêts aléatoires, qui nous ont paru plus complexes et enrichissants à étudier que les seuls arbres de décision. Ils sont d'ailleurs apparemment plus fiables et plus précis que ces derniers. De plus, ces algorithmes ont l'air d'être assez souvent utilisés et reconnus (ils ont par exemple été utilisés pour la reconnaissance des mouvements du Kinect sur la console Xbox 360). C'est pourquoi les étudier pour mieux les comprendre nous paraît important, car nous y serons sûrement de nouveau confrontés.

2 Premier test avec des jeux de données fournis par Scikit-learn

Pour notre première approche des algorithmes, nous avons choisis deux datasets fournis par Scikit-learn :

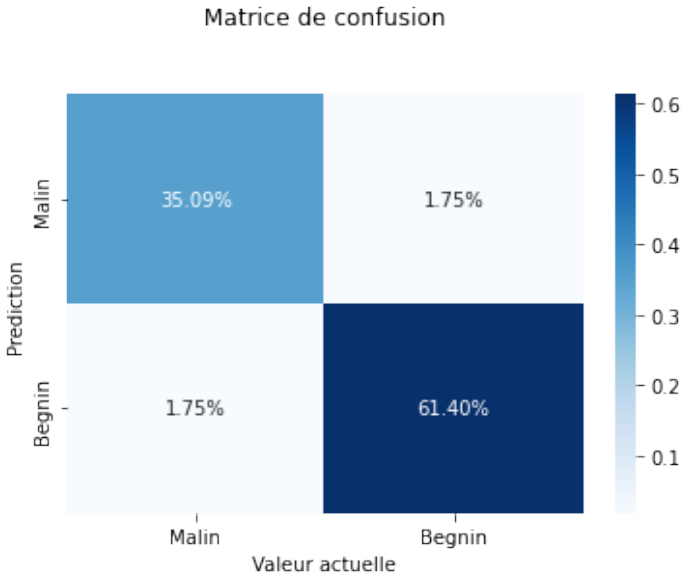
— En classification, un dataset sur les tumeurs pour le cancer du sein.
— en Régression, un dataset sur des taux de glycémie pour le diabète.

Nous avons choisi ces Datasets car parmi ceux proposés sur Scikit-learn, il s'agit de ceux que nous trouvons les plus clairs. Pour une première approche nous préférions donc des datasets que nous comprenons bien afin d'apprendre à les manipuler correctement, ainsi que de bien comprendre les principes de base de nos algorithmes et comment les appliquer. De plus, ils ont des applications que nous trouvons importantes : classification de tumeurs pour le cancer du sein, et prédiction de taux de glycémie pour le diabète.

2.1 Résultats obtenus

2.1.1 Classification [1]

Pour évaluer notre modèle, nous avons choisi d'utiliser une matrice de confusion. Celle-ci compare, pour la variable cible, les données réelles à celles prédites par le modèle. Elle nous permet de visualiser les rapports entre prédictions correctes et incorrectes.



On observe 1,75% de faux positifs et 1,75% de faux négatifs, donc un total d'environ 3,5% de fausses prédictions contre environ 96% de bonnes prédictions au total.
À partir de cette matrice nous pouvons calculer toutes les métriques nécessaires pour évaluer notre modèle. La matrice graphique étant normalisée pour afficher des pourcentages, on calcule ces métriques à partir des valeurs exactes.

Nous avons choisi d'utiliser les **métriques** suivantes :

— *recall_score* : le recall donne le pourcentage de positifs bien prédits par le modèle. Plus il est élevé, moins le modèle ne rate de cas positifs (tumeur maligne). C'est donc un indicateur important dans le cadre du cancer, ou minimiser les faux négatifs peut sauver des vies. Ici le recall à un score de : $40/(40+2) = 0,952\%$.

— *accuracy_score* : il donne le pourcentage de prédictions correctes (positives comme négatives). Il est donc aussi important à prendre en compte, et à associer avec le recall pour avoir des informations précises. Il nous apprend ici que nous avons environ 97% de prédictions correctes (négatives comme positives). Valeur exactes $= (40+70)/114 = 110/114 = 0,965\%$ de précision. Nous avons donc environ 95% de cas positifs qui sont bien prédits par le modèle.

Le modèle obtenu est donc très précis sur ce dataset.

2.1.2 Régression [3]

Les métriques de régression usuellement utilisées sont MAE (erreur absolue moyenne), MSE (erreur quadratique moyenne), RMSE (erreur quadratique moyenne) et R^2.

— $\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \bar{y}|^2$

C'est la moyenne effectuée sur la différence absolue entre les valeurs réelles et celles prédites. Il mesure la moyenne des résidus [1] dans l'ensemble des données.

1. Résidus = à quel point les valeurs réelles sont écartées de la droite de régression. Ce sont donc les erreurs observées.

— MSE = $\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2$

C'est la somme des différences entre valeur réelle et prédiction au carré le tout divisé par le nombre de valeurs. C'est donc la moyenne des différences au carré. Il mesure la variance des résidus.
Plus la somme des différences est proche de 0 (plus les valeurs sont ajustées à la droite de régression), plus le modèle est précis. Donc plus la valeur de MSE est proche de 0, plus le modèle est précis.

Ces deux métriques apportent donc des informations sur la précision du modèle. MSE pénalise beaucoup plus les grandes erreurs que MAE. En effet, une grande différence entre valeur prédite et réelle sera élevée au carré dans MSE et aura donc plus d'impact dans la valeur finale. C'est pourquoi dans notre cas (prédiction d'un taux de glycémie pour détecter le diabète) on utilisera la MSE car une grande erreur de prédiction peut être grave. Les grandes erreurs de prédictions doivent être détectées (pénalisées).

— RMSE = $\sqrt{MSE}$, ce qui correspond à une mesure d'écart type des résidus (erreur de prévisions), qui ramène la valeur à la même échelle que celle de l'erreur de prédiction, c'est pourquoi nous préférons choisir RMSE. Tout comme MSE, plus il est élevé par rapport aux observations faites sur les données test, moins le modèle est précis (on peut aussi l'exprimer comme un pourcentage des observations sur les données test, plus ce pourcentage est grand moins le modèle est précis).

Nous avons aussi utilisé le R^2(ou coefficient de détermination).

— $R^2 = 1 - \frac{\sum(y_i - \bar{y}^2)}{\sum(y_i - \bar{y})^2}$

R^2 calcule donc les erreurs par rapport au niveau de variation présent dans les données. Plus il est proche de 1, plus la fraction est proche de 0, donc plus les erreurs du modèle sont petites par rapport à la variance des données : le modèle est précis. À l'inverse, si il est proche de 0 le modèle n'est pas précis.
Si les erreurs sont plus grandes que la variance des données on peut obtenir un R^2 négatif. C'est un indicateur très utilisé en statistiques et assez visuel pour définir la qualité d'une régression (associable à un pourcentage pour une valeur entre 0 et 1). Ici le score est très bas, donc le modèle est imprécis.

— RMSE = 62.1097.. et $R^2 = 0.34$.. RMSE est élevé et R^2 est petit, notre modèle en régression n'est donc pas précis du tout.

2.2 Conclusion de ce premier test

Les modèles obtenus avec les algorithmes de type Random Forest sont efficaces sur notre premier dataset dans un but de classification, mais ne le sont pas sur notre deuxième dataset en régression. Après cette première approche, on peut donc penser que ce type d'algorithme est plus efficace pour des problèmes de classification.

3 Étude des paramètres de l'algorithme

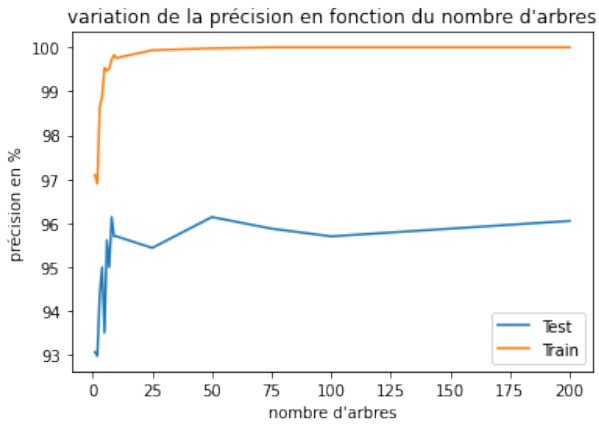
Nous allons nous concentrer sur les hyper-paramètres suivants et les faire varier :

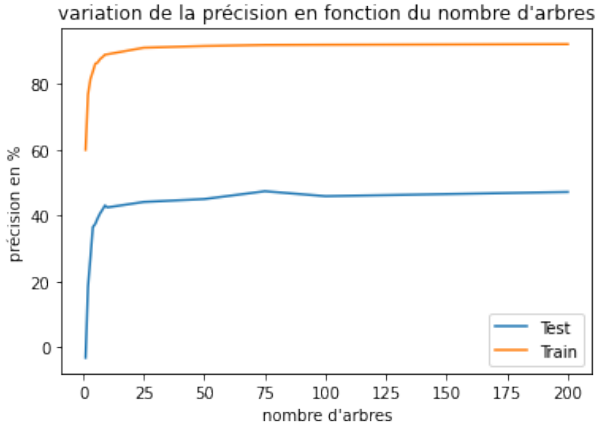
1. n_estimators
2. max_depth
3. max_samples
4. max_leaf_nodes

Pour chaque valeur des paramètres, nous avons effectué 10 tests de précision puis retenu les moyennes obtenues, pour avoir une courbe plus précise et s'assurer de ne pas avoir obtenu une valeur aberrante.
Nous testons l'algorithme sur les données sur lesquelles il à été entraîné et sur les données qu'il ne connaît pas encore (celles de test), pour vérifier la cohérence des résultats.

3.1 n_estimators

Il s'agit du nombre d'arbres utilisés dans la forêt.



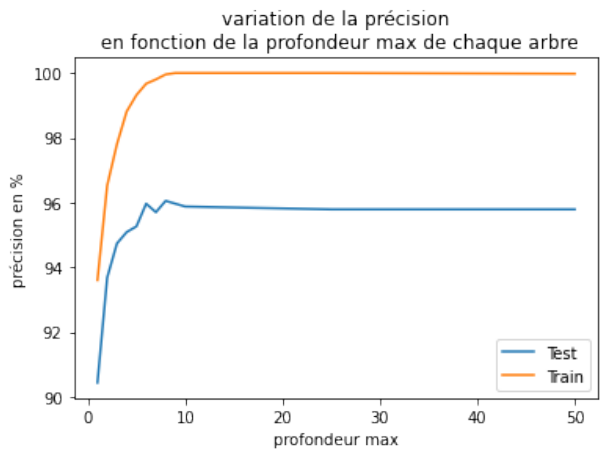


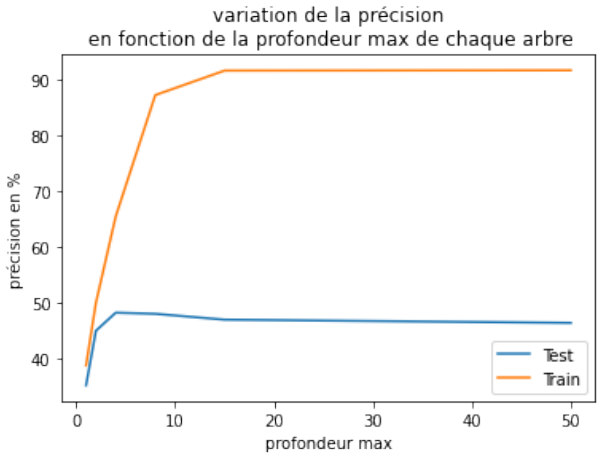
Pour un petit nombre d'arbres (≤ 10) la précision du modèle est légèrement plus faible. Ce qui est logique car moins il y a d'arbres, moins il y a de prédictions faites sur différentes parties du dataset donc moins le résultat sera précis.
En augmentant le nombre d'arbres, la précision du modèle augmente et à partir d'une quinzaine d'arbres environ, la précision oscille autour de la moyenne. On ne gagne plus rien à augmenter la valeur. En effet, chaque arbre se concentrant sur une vision du problème, plus il y a d'arbres, plus ils risquent de traiter les mêmes données (max-samples n'est pas défini ici) et plus il y a de chance qu'ils effectuent les mêmes estimations. Ce qui n'apporte rien au modèle et ne ferait que le ralentir.
Pour conclure, un nombre trop petit d'arbres impacte de manière négative la précision du modèle mais utiliser toujours de plus en plus d'arbres est aussi inutile.

3.2 max_depth

Il s'agit de la profondeur maximale des arbres utilisés. (la hauteur maximale de chaque arbre de décision). Si le paramètre n'est pas explicité, chaque arbre arrête de se développer

lorsque toutes ses feuilles sont pures.

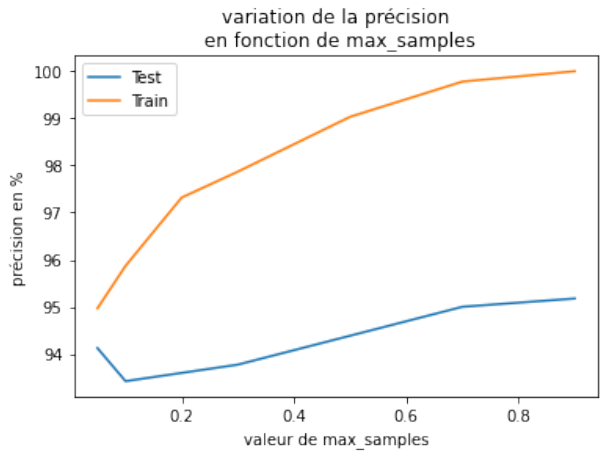


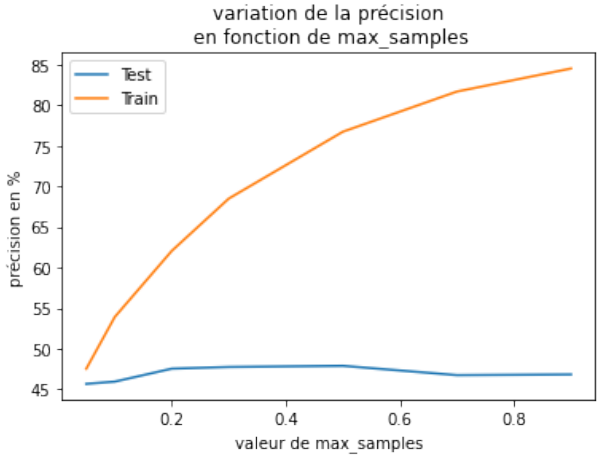


Pour une profondeur faible (≤ 10) la précision du modèle est légèrement plus faible que la moyenne.
Pour être performant l'algorithme a besoin d'une profondeur qui ne soit pas trop faible. Ce qui semble logique car plus la profondeur est élevée, plus l'arbre traite d'informations sur les données, donc plus la prédiction est précise.
En augmentant cette profondeur, la précision augmente, car l'arbre peut diviser ses noeuds comme il le souhaite de manière à ce que chaque feuille soit pure dans le résultat final.
La précision se stabilise ensuite et n'évolue plus au-delà d'un certain point.
Il est possible qu'en atteignant une certaine profondeur, l'arbre se spécialise trop (les critères de divisions deviennent trop spécifiques) et qu'il n'arrive donc plus à généraliser les classes de données.
Par exemple, l'arbre pourrait diviser des noeuds déjà purs, sur des critères inutiles par rapport au problème (trop précis). C'est ce qu'on appelle le sur-ajustement.
Ceci pourrait expliquer que la précision sur les données du train-set soit aussi élevée. Dans ce cas, on pourrait s'attendre à ce que la précision sur les données de test chute, or ici elle reste plutôt constante.

3.3 max_samples

Max_samples correspond à la fraction de données du dataset qui sera utilisée par chaque arbre de décision.

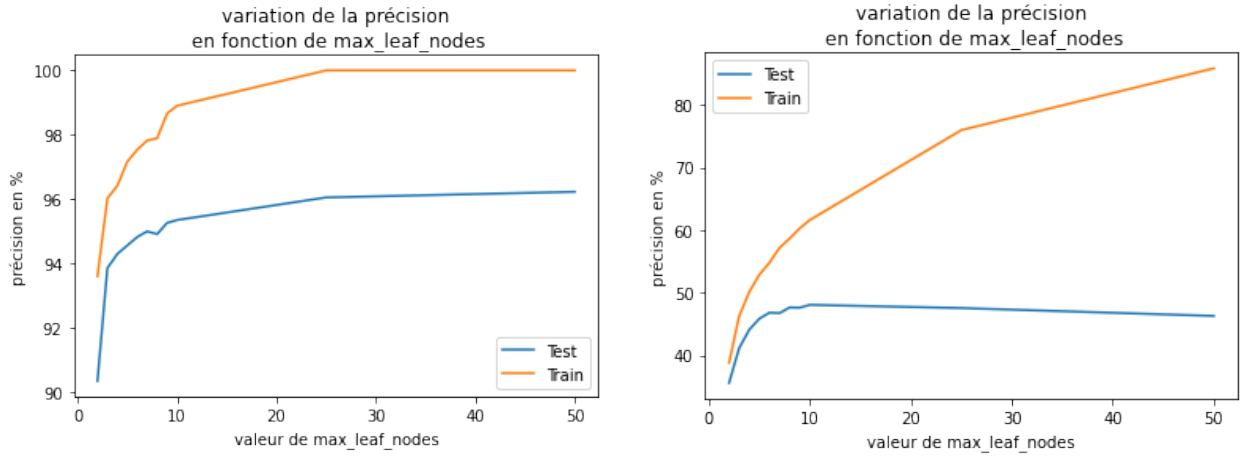




Au fur et à mesure de l'augmentation de la fraction de données, la précision stagne en régression et augmente légérement en classification. Ce qui semble logique car plus l'arbre utilise de données du dataset, plus il aura d'informations pour établir de bonnes prédictions. Cependant, à partir d'une certaine fraction de dataset, les arbres contiennent chacun des données aussi contenues dans les autres arbres. Ces données pourraient être prises en compte que par un seul arbre, cette redondance n'apporte donc pas plus de précision. Ce qui explique pourquoi le modèle en Régression stagne et que la précision en Classification n'augmente que très légèrement. On peut en conclure qu'il vaut mieux ne pas donner toutes les données du dataset à chaque arbre de décision mais seulement une partition équitable. En effet, donner une trop grande partie des données à chaque arbre ne ferait que ralentir l'algorithme.

3.4 max_leaf_nodes

Il s'agit du nombre maximal de feuilles par arbre. Ce paramètre permet de restreindre la croissance de chaque arbre en conditionnant la division d'un noeud. En effet, un noeud ne peut se fractionner que si le nombre de feuilles n'a pas atteint le *max_leaf_nodes* défini.



On constate que la précision du modèle est plus basse pour un nombre maximal de feuilles très petit : peut-être que l'arbre ne peut pas se développer assez pour que le modèle s'ajuste au mieux sur les données. Dans ce cas, des feuilles prises en compte (des noeuds n'ayant pas pu se diviser) seront moins pures. C'est peut-être un cas de sous-ajustement.
Au fur et à mesure de l'augmentation du nombre de feuilles, la précision augmente puis stagne autour de la moyenne. Les arbres peuvent développer assez de feuilles pour permettre un meilleur ajustement.
On voit néanmoins que la précision du train-set évolue de plus en plus vers un cas de sur-ajustement.
Cela peut-être dû au fait que les arbres commencent à trop se fractionner et se spécialiser. Dans ce cas, on pourrait s'attendre à ce que la précision du test chute.

3.5 Grid Search CV

Nous effectuons ensuite un Grid Search avec des valeurs pour lesquelles la précision semble maximale selon nos analyses précédentes. Le Grid Search s'occupera de choisir les

meilleures valeurs ainsi que les meilleures associations de paramètres. Voici les résultats que nous obtenons :

- **En classification** :
  - max-depth = 15, max-leaf-nodes = 20, max-samples = None, n-estimators = 80.
  - précision sur les données de test = 0.94%
  - accuracy-score = 93.86% de prédictions correctes.
  - recall-score = 93.86% de positifs bien prédits par le modèle.
- **En régression** :
  - max-depth = None, max-leaf-nodes = 10, max-samples = 0.2, n-estimators = 80.
  - précision sur les données de test = 0.43%
  - RMSE = 57.91, moyenne des observation = 145.34
  - RMSE correspond à 39.84% des observations
  - $R^2 = 42.75$

4 Second test avec des jeux de données choisis

4.1 Les datasets [2] [4]

Le Dataset choisi en classification [2] concerne le prix des téléphones selon leurs caractéristiques. Il nous à été proposé par l'un de nos encadrant. Celui en régréssion [4] est choisi dans la même continuité, il concerne le prix des ordinateurs selon leurs caractéristiques.

Features : Les caractéristiques du téléphone ou de l'ordinateur (puissance de la batterie, la capacité de la mémoire, taille de l'écran...).
La cible est la catégorie de prix du téléphone pour le dataset de classification (de 0 à 3), et le prix en euros pour celui en régression.

Nettoyage des données :

1. **Classification :**
   - Aucune donnée manquante dans le dataset.
   - Toutes nos données sont des valeurs numériques.
   - Il n'y a pas de données aberrantes.
2. **Régression :**
   - Aucune donnée manquante dans le dataset.
   - Nous avons ici aussi remplacé les données catégoriques par des données numériques.
   - Nous avons supprimé du dataset la colonne donnant l'ID de l'ordinateur sur la table car elle était inutile pour notre modèle.
   - Nous retirons du dataset les tuples pour lesquels le prix de l'ordinateur est aberrant. Ces prix ne feraient que fausser le modèle car ils sont inhabituels et non représentatifs de l'échantillon. Nous les supprimons car nous ne voulons

pas les remplacer par d'autres prix (exemple par le prix moyen, le prix qui revient le plus souvent...) qui ne correspondent pas aux caractéristiques du tuple et pourraient fausser le modèle.

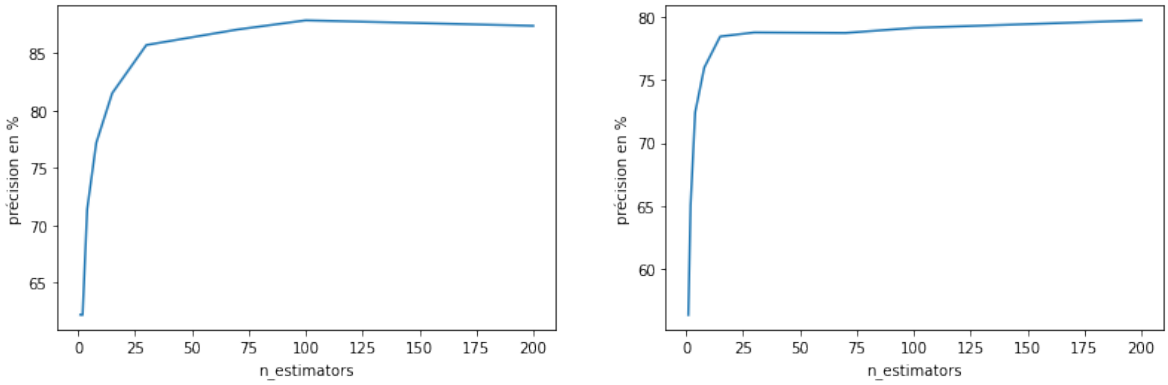
Analyse des paramètres :

En effectuant un premier test avec les paramètres par défaut de l'algorithme nous obtenons une précision $\approx 85\%$ en classification et comprise entre $\approx 75\% \leq x \leq 80\%$ en régression.

Nous allons faire varier les paramètres suivants pour tenter d'améliorer la précision du modèle :

1. **n estimators**
   Il définit le nombre d'arbres de la forêt. Par défaut il est calibré à 100, nous le ferons donc varier sur des petites valeurs, des valeurs proches de 100 et plus grande que 100.

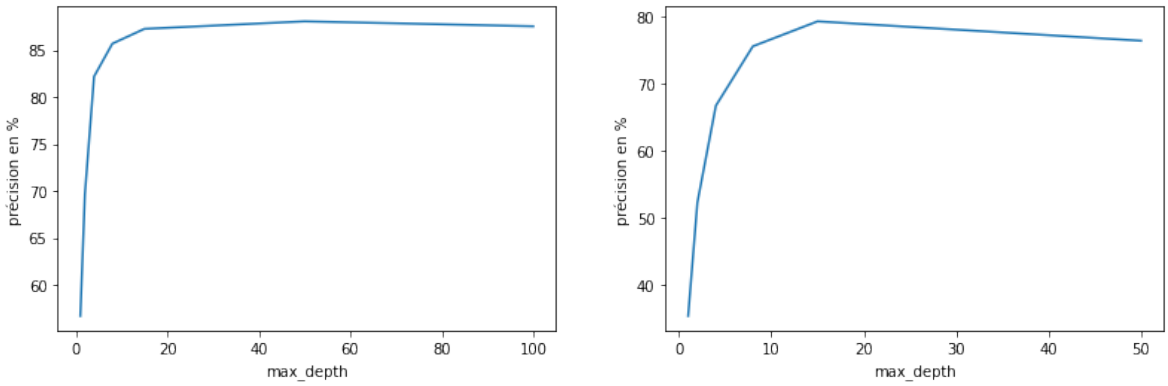
   n_estimators = [1, 2, 4, 8, 15, 30, 70, 100, 200]



   La meilleure précision semble être pour $x \geq 25$ arbres. La précision stagne autour de la valeur par défaut = des valeurs élevées.

2. **max depth**
   Il définit la profondeur maximale de nos arbres de décision. Par défaut, il se développe jusqu'à ce que les feuilles soient pures, nous voulons donc voir si un petit nombre de feuilles apporte une meilleure précision plutôt que de laisser l'arbre grandir jusqu'à ce que les feuilles soient pures.

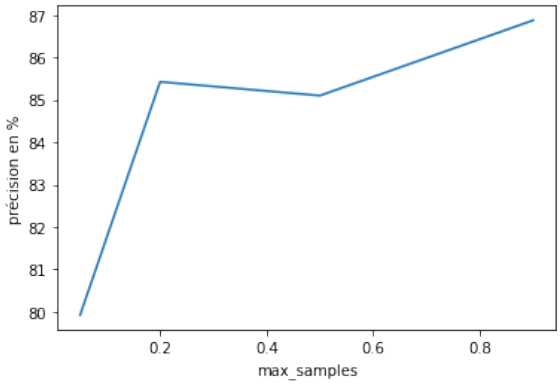
   max_depth = [1, 2, 4, 8, 15, 50, (100)]

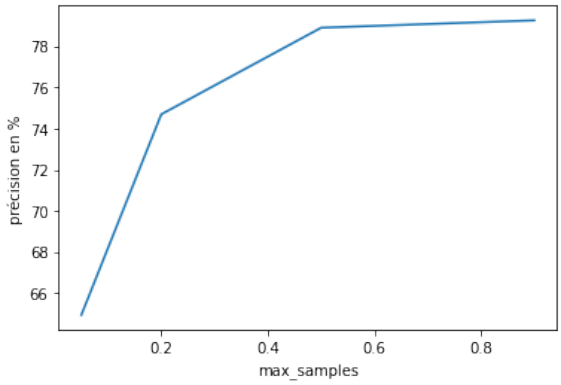


   La meilleure précision semble être pour une profondeur ≥ 15.

3. **max samples**
   Il définit la fraction de données du dataset qui sera utilisée par chaque arbre de décision. Nous le faisons donc varier selon des fractions de tailles du dataset.

max_samples = [0.05,0.2,0.5,0.9]

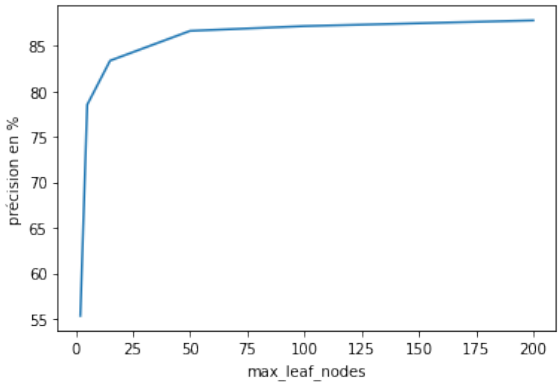


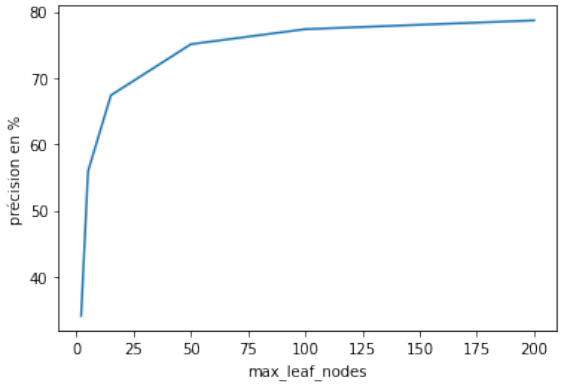


La meilleure précision semble être obtenue pour la plus grande fraction du dataset.

4. **max leaf nodes**
   Il définit le nombre maximal de feuilles par arbre. Par défaut, un nombre infini de feuilles peut être développé. Nous voulons aussi voir si un petit nombre de feuilles apporte plus de précision qu'un trop grand nombre de feuilles possible.

max_leaf_nodes = [2, 5, 15, 50, 100, 200]





La meilleure précision semble être obtenue pour un grand nombre de feuilles, laisser le paramètre par défaut peut être une bonne solution, car au-dessus de 50, la précision continue d'augmenter mais très légèrement.

4.2 Calibrage des paramètres

Nous effectuons un Grid Search pour connaitre la meilleur combinaison de paramètres ainsi que le meilleur calibrage pour chacun d'entre eux. Voici les résultats obtenus :

1. En Classification :
   - max-depth = 25, max-leaf-nodes = 200, max-samples = 0.8, n-estimators = 200.
   - précision sur les données de test = 87.5%.
   - accuracy_score : 87.50% de prédictions correctes.
   - recall_score : 87.50% de positifs bien prédits par le modèle.
2. En Régression

— max-depth = None, max-leaf-nodes = None, max-samples = None, n-estimators = 200.

— précision sur les données de test = 78%

— RMSE = 260.91, moyenne des observation = 1052.93

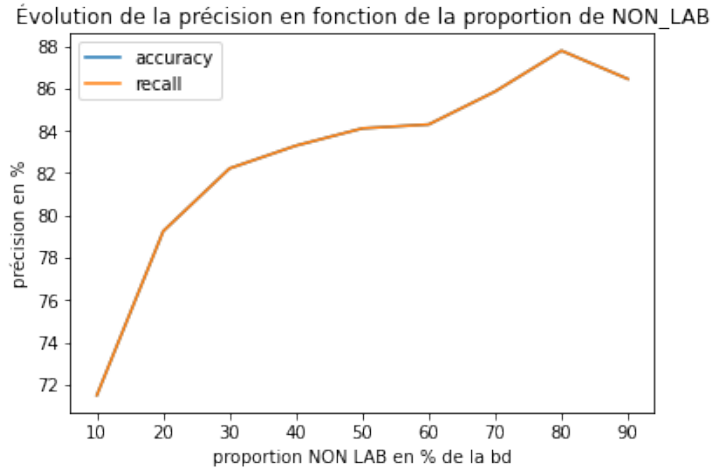
— RMSE correspond à 24.71% des observations

— $R^2 = 78.00\%$

Dans les deux cas, Nous constatons que la précision obtenue est proche de celle obtenue avec les paramètres par défaut. La précision est donc sensiblement la même avec ou sans calibrage des paramètres. Cela signifie que les valeurs par défaut de nos paramètres s'avèrent être optimales. C'est d'ailleurs bien ce que nous observons avec les résultats du Grid Search en Régression, la plupart des paramètres sont à "None". Cela semble normal car les meilleures précisions que nous observons sont obtenues pour les plus grandes valeurs des paramètres et que par défaut, ils sont développés sur des grandes valeurs.

5 Étude sur l'algorithme de classification en utilisant l'apprentissage semi-supervisé

Dans cette étape, nous entraînons notre modèle en apprentissage supervisé sur une partie "labellisée" des données et en semi-supervisé sur une autre "non-labellisée". Nous faisons varier la proportions de données de "non-labellisées" pour observer les effets sur la précision du modèle.



2

On constaste que la précision augmente proportionnellement avec la quantité de données non-labellisées utilisée. Cela paraît logique car plus le modèle a de données sur lesquelles s'entraîner plus il sera précis.
Il sera toujours moins précis qu'avec une grande quantité de données uniquement labellisées mais, en pratique, on ne dispose pas forcément d'autant de données labellisées. C'est donc une bonne technique pour obtenir un modèle plus précis sans avoir besoin de labelliser les données, ce qui peut être plus couteux. Nous observons donc le même comportement en variant la quantité de données labellisées. Plus elle est grande, plus la précision est élevée dès le départ, mais la courbe garde la même tendance.

2. Les deux courbes se superposent parfaitement

Références

[1] Classification breast cancer. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html.

[2] Classification prix des téléphones. https://www.kaggle.com/datasets/iabhishekofficial/mobile-price-classification.

[3] Régression diabetes. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_diabetes.html.

[4] Régression prix des ordinateurs. https://www.kaggle.com/datasets/muhammetvarl/laptop-price.