Apprentissage supervisé - Régression, DecisionTreeRegressor, RandomForestRegressor

1 Préparation des données

L'étude porte sur un fichier csv dans lequel on retrouve les prix des logements en Californie ainsi que de nombreuses informations telles que des informations géographiques comme la longitude, la latitude, la proximité de l'océan mais également le revenu moyen, le nombre de chambres…

L'objectif de cette étude est de prédire le prix des logements en fonction des autres variables du fichier de données.
Avant d'établir un modèle de prévisions, il convient de préparer les données afin qu'elles soient exploitables pour la modélisation.

L'observation du fichier nous a permis de constater que la variable total_bedrooms contient des valeurs manquantes et que la variable ocean_proximity est la seule variable qualitative (catégorielle) du fichier.

La visualisation des données nous a permis de démontrer que le prix des logements en Californie est plus élevé en zone côtière.

Le tableau de corrélation des variables nous montre une corrélation plus forte entre la Target (variable à expliquer) median_house_value et la variable explicative median_income. on a donc un lien plus fort entre ces deux variables.

Afin de rendre le fichier exploitable, on a remplacé les valeurs manquantes de total_bedrooms par la valeur médiane des valeurs de la colonne et on a encodé la variable catégorielle ocean_proximity pour n'avoir que des variables numériques.

Le fichier de données peut ainsi être utilisé pour la modélisation.

2 Sélection, apprentissage et évaluation du modèle

La target y est median_house_value et les features, le reste des variables.

On fractionne nos données en base d'apprentissage (80% des données) et en base de test (20%).

On applique la régression linéaire sur les données d'apprentissage, puis on effectue les prédictions des résultats du modèle sur l'ensemble de test.

On établit l'évaluation de la performance du modèle grâce au MSE et au coefficient de détermination. Le coefficient de détermination du modèle est de 0.65, ce qui est un résultat assez moyen.

La modélisation grâce aux arbres de décision peut elle nous apporter de meilleurs résultats de performance du modèle?

On construit un arbre de décision sur les données d'apprentissage.

On obtient une performance sur le modèle égale à 1. Au vu des résultats, il est difficile de conclure que le modèle est parfait. On va donc utiliser la méthode du K-Fold (validation croisée) pour s'assurer de la bonne performance du modèle.

La validation croisée nous donne 10 k-fold compris entre 0.6 et 0.7, soit une moyenne à 0.65. L'écart-type des K-fold est de 0.025. On a donc une faible dispersion de valeurs.

Le modèle d'arbre de décision révèle un problème de surapprentissage. Ce problème peut être dû au fait que l'arbre est trop complexe généralisant mal l'ensemble d'apprentissage ou que l'arbre soit mal équilibré accordant trop d'importance à une classe au détriment des autres.

3 Fine-Tunning

Afin de résoudre le problème de surapprentissage notamment du modèle d'arbre de décision, nous allons jouer sur les différents paramètres du modèle pour pouvoir le rendre plus performant. Dans un premier temps on va utiliser la fonction RandomForestRegressor,pour le modèle de régression de forêt aléatoire. Cette fonction permet de faire à la fois de la régression (pour les variables numériques) et de la classification (pour les variables catégorielles) grâce à plusieurs arbres de décision où chaque arbre de décision est formé sur un échantillon de données différent. L'échantillonnage est effectué avec remplacement.

Chaque arbre décisionnel est créé à partir de portions des données d'origine (entraînement) générées de manière aléatoire. Chaque arbre génère sa propre prévision et ses votes relatifs à un résultat. Le modèle de forêt tient compte des votes de tous les arbres décisionnels dans la prévision ou le classement du résultat d'un échantillon inconnu. Ceci est important en raison des problèmes de sur-ajustement à un modèle que peut rencontrer individuellement chaque arbre. La combinaison de plusieurs arbres dans une forêt pour formuler des prévisions peut éventuellement régler le problème de sur-ajustement associé à un seul arbre.

On obtient de meilleurs résultats avec la régression de forêt aléatoire avec un rmse moyen de presque 48747 contre 68433 pour régression linéaire.

Afin d'optimiser notre modèle, on peut jouer sur les paramètres du modèle (Hyperparamètres). On peut faire varier manuellement ces hyperparamètres mais on peut le faire aussi de façon automatique grâce à la fonction GridSearchCV. Pour un Random Forest, on doit choisir le nombre d'arbres à créer et le nombre de variables à utiliser à chaque division d'un nœud. La méthode Grid search est une méthode d'optimisation (hyperparameter optimization) qui va nous permettre de tester une série de paramètres et de comparer les performances pour en déduire le meilleur paramétrage. dans le cas d'un Random Forest on peut tester : Par exemple:

Nombre d'arbres : {100, 150, 200, 250}

Nombre de variables sélectionnées : {8, 10, 12, 14, 16}

Le Grid Search croise simplement chacune de ces hypothèses et va créer un modèle pour chaque combinaison de paramètres. L'inconvénient d'une utilisation abusive du Grid Search augmente considérablement les temps de calcul.

Cette méthode permet non seulement de répondre à une problématique de surapprentissage du modèle mais facilite l'obtention d'un modèle qui soit facilement optimisé.