Le paramétrage d'une méthode des plus proches voisins

Romain Lesueur Lucas Beloin Mandir Diop

BUT Science des données — Poster Data Mining

Introduction

La méthode des k plus proches voisins (k-NN) est un algorithme simple et intuitif utilisé pour la classification et la régression [1]. Ses performances dépendent du choix des hyperparamètres comme le nombre de voisins (k), la distance utilisée et la pondération. L'objectif de ce travail est de trouver les valeurs optimales de ces paramètres pour obtenir un modèle précis et équilibré.

Principes du paramétrage

Le paramétrage d'un modèle *k-Nearest Neighbors* (k-NN) consiste à rechercher la combinaison d'hyperparamètres qui maximise la performance du modèle.

- **n_neighbors** (nombre de voisins à considérer)
- **weights** poids appliqué aux voisins : `uniform` pour poids égal à tous les voisins, `distance` pour poids inversement proportionnel à la distance
- **metric** mesure de distance utilisée : `euclidean` pour distance euclidienne (norme L2), `manhattan` pour distance de Manhattan (norme L1), etc.

Ces paramètres influencent la précision du modèle et le compromis biais-variance.

Compromis Biais-Variance

Valeur de k trop faible : modèle instable, sensible au bruit (variance élevée).

Valeur de k trop élevée : modèle trop rigide, perte de complexité locale (biais élevé).

Objectif : trouver le k optimal qui minimise l'erreur de généralisation.

Sélection du modèle

En pratique, on utilise les outils du module sklearn.model_selection de la bibliothèque sklearn :

- **cross_val_score** : validation croisée à k plis
- **GridSearchCV** : teste toutes les combinaisons
- **RandomizedSearchCV** : échantillonnage aléatoire

Objectif : maximiser la performance, limiter le surapprentissage.

Ces techniques permettent d'évaluer rigoureusement les hyperparamètres tout en évitant le surapprentissage grâce à la validation croisée. Le choix entre Grid Search et Random Search dépend de la complexité de l'espace des paramètres et des ressources computationnelles disponibles.

Comparaison des méthodes d'optimisation

Grid Search teste toutes les combinaisons (exhaustif) tandis que **Random Search** échantillonne aléatoirement l'espace des paramètres (plus rapide).

Comparaison Grid Search vs Random Search

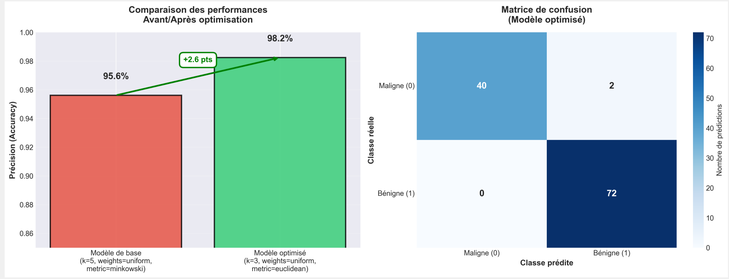
Critère	Grid Search	Random Search
Nombre de tests	40	20
Meilleur score	0.9693	0.9693
Meilleur k	6	9
Meilleur weights	uniform	uniform
Temps relatif	Plus long (×2)	Plus rapide (÷2)
Couverture	Exhaustive (100%)	Aléatoire (50%)
Garantie	Optimum garanti	Approximation

Résultat clé : Random Search atteint les mêmes performances avec 2 fois moins de tests, offrant un excellent compromis entre précision et efficacité computationnelle. Cette approche est particulièrement avantageuse lorsque l'espace de recherche est de grande dimension ou que l'entraînement des modèles est coûteux en temps de calcul.

En pratique, Random Search est souvent privilégié pour sa rapidité d'exécution et son efficacité comparable.

Application pratique avec GridSearchCV

L'optimisation du modèle **k-NN** a été réalisée à l'aide de la fonction `GridSearchCV` de la bibliothèque sklearn [2] [3]. Cette méthode permet de tester automatiquement plusieurs combinaisons d'hyperparamètres afin de trouver celle qui maximise la performance moyenne mesurée par validation croisée.



- n_neighbors = 3
- weights = uniform
- metric = euclidean

Score moyen : 98.2 % (contre 95.6 % sans optimisation)

Jeu de données : *Breast Cancer Wisconsin (Scikit-learn)* [4]

Conclusion

Le paramétrage d'un modèle **k-NN** est essentiel pour garantir de bonnes performances. L'expérimentation a démontré qu'un choix judicieux des paramètres peut améliorer la précision du modèle de 2.6 points, rendant le **k-NN** plus robuste et adaptable.

Cette approche méthodique illustre l'importance de la sélection de modèle dans tout projet de machine learning.

Références

[1] Tutoriel k-nn, section 5. Sélection de modèles.

[2] Gridsearchcv. `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html`.

[3] Cahier jupyter: knn-model-selection.

[4] Breast cancer wisconsin dataset. `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html`.