KNN - K-Nearest Neighbors

KNN (K-Nearest Neighbors)

Théorie

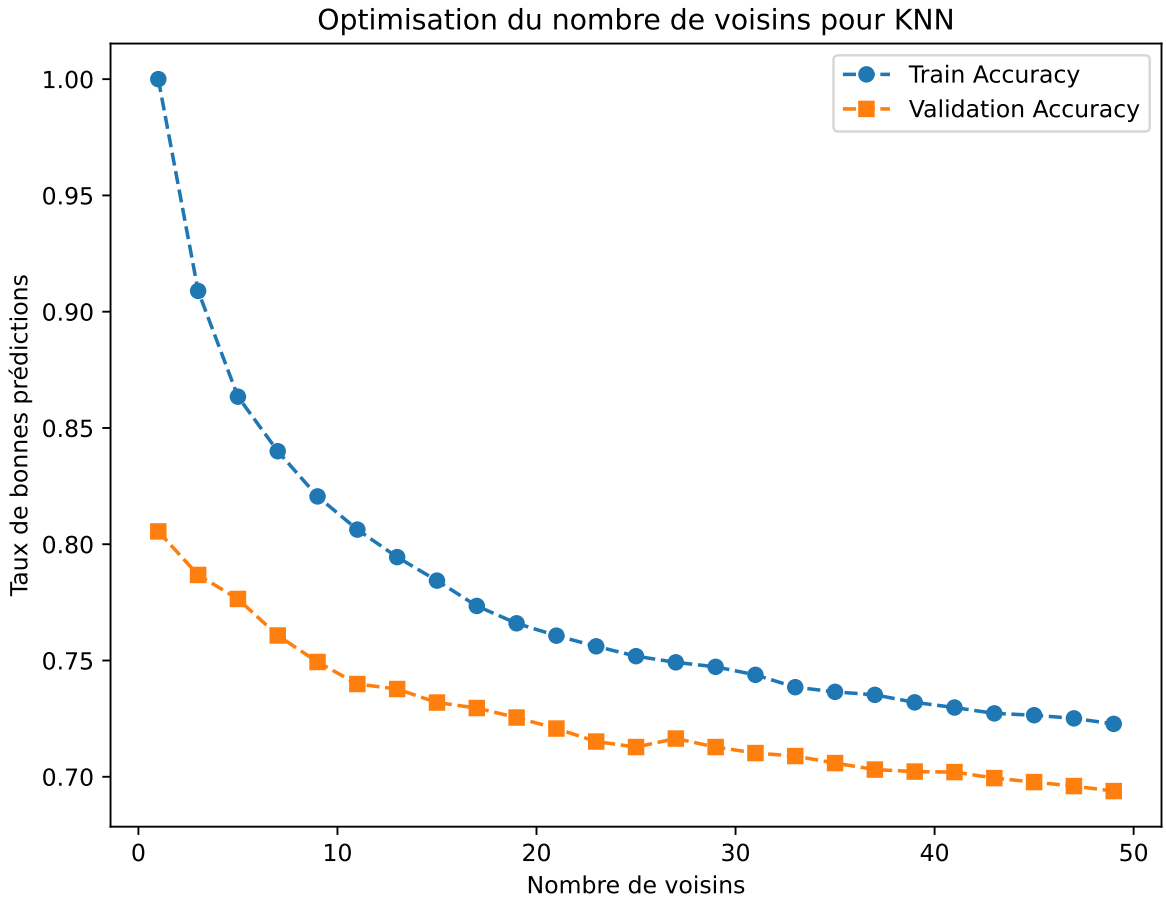
Le KNN est un algorithme de classification basé sur la proximité des données dans un espace multidimensionnel.

Exemple en Python

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.model_selection import train_test_split

# Chargement des ensembles de données déjà préparés
train_data = pd.read_csv('covertype_train.csv')
val_data = pd.read_csv('covertype_val.csv')
test_data = pd.read_csv('covertype_test.csv')

# Préparation des données
X_train = train_data.drop('Cover_Type', axis=1)
y_train = train_data['Cover_Type']

X_val = val_data.drop('Cover_Type', axis=1)
y_val = val_data['Cover_Type']

X_test = test_data.drop('Cover_Type', axis=1)
y_test = test_data['Cover_Type']
```

```
# Recherche du meilleur nombre de voisins en utilisant uniquement l'ensemble d'entraînement
neighbors = range(1, 51, 2)
train_accuracies = []
val_accuracies = []

for k in neighbors:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)

    # Évaluation sur l'ensemble d'entraînement
    y_train_pred = knn.predict(X_train)
    train_accuracies.append(accuracy_score(y_train, y_train_pred))

    # Évaluation sur l'ensemble de validation
    y_val_pred = knn.predict(X_val)
    val_accuracies.append(accuracy_score(y_val, y_val_pred))

# Sélection du meilleur hyperparamètre basé sur l'ensemble de validation
best_k = neighbors[val_accuracies.index(max(val_accuracies))]
print(f"Meilleur nombre de voisins: {best_k}")

# Affichage du graphique comparant l'entraînement et la validation
plt.figure(figsize=(8, 6))
plt.plot(neighbors, train_accuracies, marker='o', linestyle='dashed', label='Train Accuracy')
plt.plot(neighbors, val_accuracies, marker='s', linestyle='dashed', label='Validation Accurac
plt.xlabel("Nombre de voisins")
plt.ylabel("Taux de bonnes prédictions")
plt.title("Optimisation du nombre de voisins pour KNN")
plt.legend()
plt.show()

# Modèle final avec le meilleur k
knn = KNeighborsClassifier(n_neighbors=best_k)
knn.fit(X_train, y_train)

# Évaluation sur l'ensemble de test
y_test_pred = knn.predict(X_test)
print("\nÉvaluation sur l'ensemble de test")
print(confusion_matrix(y_test, y_test_pred))
print(classification_report(y_test, y_test_pred))
```

```
Meilleur nombre de voisins: 1
```




```
Évaluation sur l'ensemble de test
[[1413  263    0    0    6    2   24]
 [ 267 1914   39    0   20   16    5]
 [   0   44  210    3    0   24    0]
 [   0    1    8    8    0    4    0]
 [   8   23    2    0   41    0    0]
 [   1   14   28    1    0  100    0]
 [  27    4    0    0    0    0  128]]
              precision    recall  f1-score   support

           1       0.82      0.83      0.83      1708
           2       0.85      0.85      0.85      2261
```

3	0.73	0.75	0.74	281
4	0.67	0.38	0.48	21
5	0.61	0.55	0.58	74
6	0.68	0.69	0.69	144
7	0.82	0.81	0.81	159
accuracy			0.82	4648
macro avg	0.74	0.69	0.71	4648
weighted avg	0.82	0.82	0.82	4648