TP : Application de l'Algorithme KNN pour la Classification et la Régression

Mohamed Mahmoud EL BENANY

January 20, 2025

1 Introduction

L'objectif de ce TP est d'appliquer l'algorithme KNN (K-Nearest Neighbors) pour résoudre des problèmes de classification et de régression. Vous allez travailler avec différents ensembles de données et tester les concepts de KNN sur des cas réels comme la prédiction du prix d'une maison et la classification de la maladie d'un individu.

2 Exemple 1 : Prédiction du Prix d'une Maison (Régression KNN)

Imaginons un ensemble de données avec trois attributs pour chaque maison :

- Superficie (en m^2)
- Nombre de chambres
- Proximité des transports publics (en mètres)

L'objectif est de prédire le prix de la maison en fonction de ces attributs à l'aide de KNN.

2.1 Données

Voici les données d'exemple :

Superficie (m^2)	Chambres	Proximité Transports (m)	Prix (en milliers d'€)
70	2	300	150
120	3	500	250
80	2	800	180
200	4	200	400
100	3	700	220
90	2	600	210
150	3	400	350
60	1	1000	120

2.2 Code Python pour Régression KNN (Prix d'une Maison)

```
import numpy as np

# Donn es d'entr e
data = np.array([[70, 2, 300], [120, 3, 500], [80, 2, 800], [200, 4, 200],
                 [100, 3, 700], [90, 2, 600], [150, 3, 400], [60, 1, 1000]])
labels_regression = [150, 250, 180, 400, 220, 210, 350, 120]
# Prix de la maison en milliers d'

# Fonction pour calculer la distance Euclidienne
def euclidean_distance(x1, x2):
    return np.sqrt(np.sum((x1 - x2) ** 2))

# Fonction KNN pour r gression
def knn_regression(data, labels, test_point, k=3):
    distances = []

    # Calculer la distance entre le point test et chaque point dans le jeu de do
    for i in range(len(data)):
        dist = euclidean_distance(test_point, data[i])
        distances.append((dist, labels[i]))

    # Trier par distance croissante
    distances.sort(key=lambda x: x[0])

    # S lectionner les k voisins les plus proches
    nearest_neighbors = distances[:k]

    # Calculer la moyenne des cibles des k voisins pour la r gression
    predicted_value = np.mean([neighbor[1] for neighbor in nearest_neighbors])

    return predicted_value
```

```
# Exemple d'application du KNN pour pr dire le prix d'une maison
test_point_regression = np.array([110, 3, 600])  # Point test
k = 3

# Appliquer KNN pour pr dire le prix de la maison
predicted_value = knn_regression(data, labels_regression, test_point_regression,
print(f"Le prix pr dit pour la maison {test_point_regression} est: {predicted_v
```

3 Exemple 2 : Prédiction de la Classe d'une Maladie (Classification KNN)

Imaginons un ensemble de données sur les habitudes de vie des individus qui peuvent influencer la probabilité d'avoir une maladie, comme le diabète par exemple :

- Niveau d'activité physique (en heures par semaine)
- Consommation de sucre (en grammes par jour)
- Poids corporel (en kg)

L'objectif est de prédire si un individu a la maladie (1) ou non (0) en fonction de ces attributs à l'aide de KNN.

3.1 Données

Voici les données d'exemple :

Activité physique (h/semaine)	Consommation de sucre (g/jour)	Poids corporel (kg)	Maladie (0: Non,
10	50	70	0
5	80	85	1
15	40	60	0
3	120	95	1
8	70	75	0
2	150	100	1
12	30	65	0
4	100	90	1

3.2 Code Python pour Classification KNN (Maladie)

```
import numpy as np

# Donn es d'entr e
data = np.array([[10, 50, 70], [5, 80, 85], [15, 40, 60], [3, 120, 95],
```

```
                    [8, 70, 75], [2, 150, 100], [12, 30, 65], [4, 100, 90]])
labels_classification = [0, 1, 0, 1, 0, 1, 0, 1]  # Maladie (0: Non, 1: Oui)

# Fonction pour calculer la distance Euclidienne
def euclidean_distance(x1, x2):
    return np.sqrt(np.sum((x1 - x2) ** 2))

# Fonction KNN pour classification
def knn_classification(data, labels, test_point, k=3):
    distances = []

    # Calculer la distance entre le point test et chaque point dans le jeu de do
    for i in range(len(data)):
        dist = euclidean_distance(test_point, data[i])
        distances.append((dist, labels[i]))

    # Trier par distance croissante
    distances.sort(key=lambda x: x[0])

    # S lectionner les k voisins les plus proches
    nearest_neighbors = distances[:k]

    # D terminer la classe majoritaire parmi les k voisins
    classes = [neighbor[1] for neighbor in nearest_neighbors]
    predicted_class = max(set(classes), key=classes.count)

    return predicted_class

# Exemple d'application du KNN pour pr dire la classe de la maladie
test_point_classification = np.array([7, 60, 80])  # Point test
k = 3

# Appliquer KNN pour classer le point test
predicted_class = knn_classification(data, labels_classification, test_point_cla
print(f"L'individu {test_point_classification} est class  comme : {'Malade' if
```

4 Exemple 3 : KNN avec un Jeu de Données Fictif (Classification et Régression)

Un autre exemple générique de l'utilisation de KNN avec un jeu de données fictif de deux types de prédiction : régression et classification.

4.1 Données et Code Python

Les détails de cet exemple peuvent être similaires à ceux ci-dessus, en suivant la même structure.

5 Conclusion

Dans ce TP, nous avons appliqué l'algorithme KNN pour la régression et la classification, en utilisant deux cas concrets : la prédiction du prix d'une maison et la classification de la maladie d'un individu. Vous avez vu comment KNN fonctionne en utilisant des distances Euclidiennes pour déterminer les voisins les plus proches et prédire des résultats.