Chapter 1

Algorithmes des k plus proches voisins

L'algorithme des k plus proches voisins appartient à la famille des algorithmes d'apprentissage automatique ou *machine learning*. Ces algorithme utilisés pour la première fois vers 1959 ont connu un essor considérable au début des années 2000.

C'est ce type d'algorithme qu'utilisent de nombreuses sociétés comme les GAFAM pour cerner leurs utilisateurs en termes de consommation.

A partir d'un jeu de données, l'algorithme des k plus proches voisins détermine les k données les plus proches de la donnée *cible* à traiter. Pour prédire la classe d'un nouvel élément, il nous faut donc:

1. un échantillon de données ;
2. un nouvel élément dont on connaît les caractéristiques et dont on veut prédire le type ;
3. la valeur de k : nombre de voisins étudiés.

Il s'utilise aussi bien pour:

La classification où on va prédire la classe d'un nouvel élément

Exemple: Déterminer la couleur d'un fruit en fonction de sa largeur et de sa hauteur.

La régression où on va trouver une valeur réelle pour le nouvel élément.

Exemple: Estimer le poids d'une personne en fonction de sa taille.

1.1 Etude d'un exemple

1.1.1 Premier cas

On dispose d'un échantillon de 56 zombies dans un jeu dont on done un extrait sur la figure 1.1. Ils possèdent deux attributs compris entre 0 et 20: dangerosité et rapidité. Ces zombies sont rangés dans trois classes: normaux, améliorés ou spéciaux. On a utilisé ces attributs pour les représenter dans le repère orthonormé ci-dessous. On introduit un nouveau zombie de dangerosité 15 et de rapidité 15. On ne connaît pas sa classe et on veut la déterminer par la méthode des k plus proches voisins la classe de ce zombie cible.

1. Placer la cible sur le repère.
2. Il existe différentes méthodes pour évaluer la distance entre deux points, voir Calcul de distance. Nous choisissons ici la distance euclidienne, qui apparaît comme la plus naturelle. Comment représenter l'ensemble des points situés à même distance de la cible avec cette distance ?

Zombie	Vitesse	Danger	Classe
11	2.4	15.3	normal
23	2.5	13.1	enhanced
38	19.6	17.3	special
7	3.3	12.5	normal

Figure 1.1: Extrait de l'échantillon

3. Déterminer la classe de la cible si k=1, si k =3, si k=7.
4. Que se passe-t-il si on ne considère que la dangerosité avec k = 3? k = 5 ?

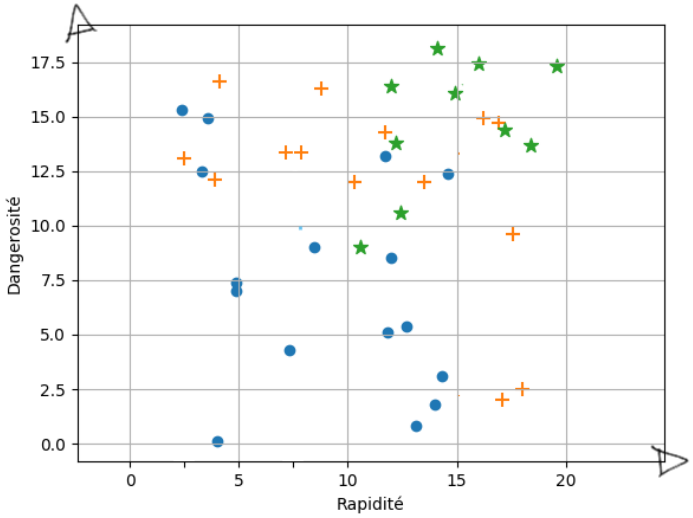

Figure 1.2: Représentation de l'échantillon, cible à placer

Remarque:

La valeur de k doit être choisie judicieusement :

trop faible: on comprend que la qualité de la prédiction diminue.

trop grande: il suffit d'imaginer qu'il existe une classe prédominante en nombre; avec une grande valeur de k, cette classe remporterait la prédiction à chaque fois. La prédiction diminue donc aussi.

1.1.2 Second cas

On travaille avec k = 5.

1. On choisit comme cible le zombie de vitesse 7,7 et de danger 10. Placer cette cible dans le repère.
2. Déterminer la classe de cette cible pour la distance Euclidienne.
3. Déterminer la classe de cette cible pour la distance de Tchebychev.

1.2 Implémentation en Python

Pour implémenter cet algorithme, il nous faut :

1. Une table de données. Cette table peut être une liste ou un dictionnaire.
2. Une distance entre deux données.
3. Une cible.

1.2.1 Calcul de distance

La notion de distance est un élément central de cet algorithme. Considérons deux données d_1 et d_2 de coordonnées (x_1, y_1) et (x_2, y_2). Voici quelques distances possibles :

La distance Euclidienne: (uniquement en repère orthonormé):

$$distance(d_1, d_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

La distance de Manhattan:

$$distance(d_1, d_2) = |x_2 - x_1| + |y_2 - y_1|$$

La distance de Tchebychev:

$$distance(d_1, d_2) = max(|x_2 - x_1|, |y_2 - y_1|)$$

1.2.2 Algorithme

Récupérez le fichier *Zombies.csv*, il vous faudra importer le module `csv` déjà rencontré dans le chapitre *données en table*. Pour rappel, on peut importer un fichier csv à l'aide d'un objet *Dictreader*. Utilisez la fonction `importer` qui renvoie la liste des zombies. Il est important de bien comprendre les objets manipulés. Les attributs Zombie, Vitesse, Danger, Classe sont les clés de chaque dictionnaire représentant un zombie.

```
def importer(echantillon):
    with open(echantillon, newline='') as fichier:
        # lecture est un objet DictReader
        lecture = csv.DictReader(fichier, delimiter=',')
        return [dict(ligne) for ligne in lecture]
>>> liste_zombie = importer('Zombies.csv')
[{'Zombie':'11', 'Vitesse':'2.4', 'Danger':'15.3',  'Classe': 'normale'},
 {'Zombie':'23', 'Vitesse':'2.5', 'Danger':'13.1',  'Classe': 'enhanced', ...}, ........]
```

A faire pour écrire l'agorithme KNN: On doit calculer les distances entre la cible et chaque point de l'échantillon et ne conserver que les classes des k points les plus proches.

- Sur le premier élément de `liste_zombie`, indiquer les instructions permettant de renvoyer les valeurs des clés `'Vitesse'` et `'Danger'`.
- Pourquoi peut-on se contenter de calculer les carrés des distances ?
- Ecrire une fonction `distance` prenant en paramètre les coordonnées de deux points et renvoyant la distance entre ces points.
- Les ranger dans l'ordre croissant de leur distance à la cible; en utilisant éventuellement la fonction `sorted` avec la commande `key=distance`.
- Placer les k premiers dans une liste.