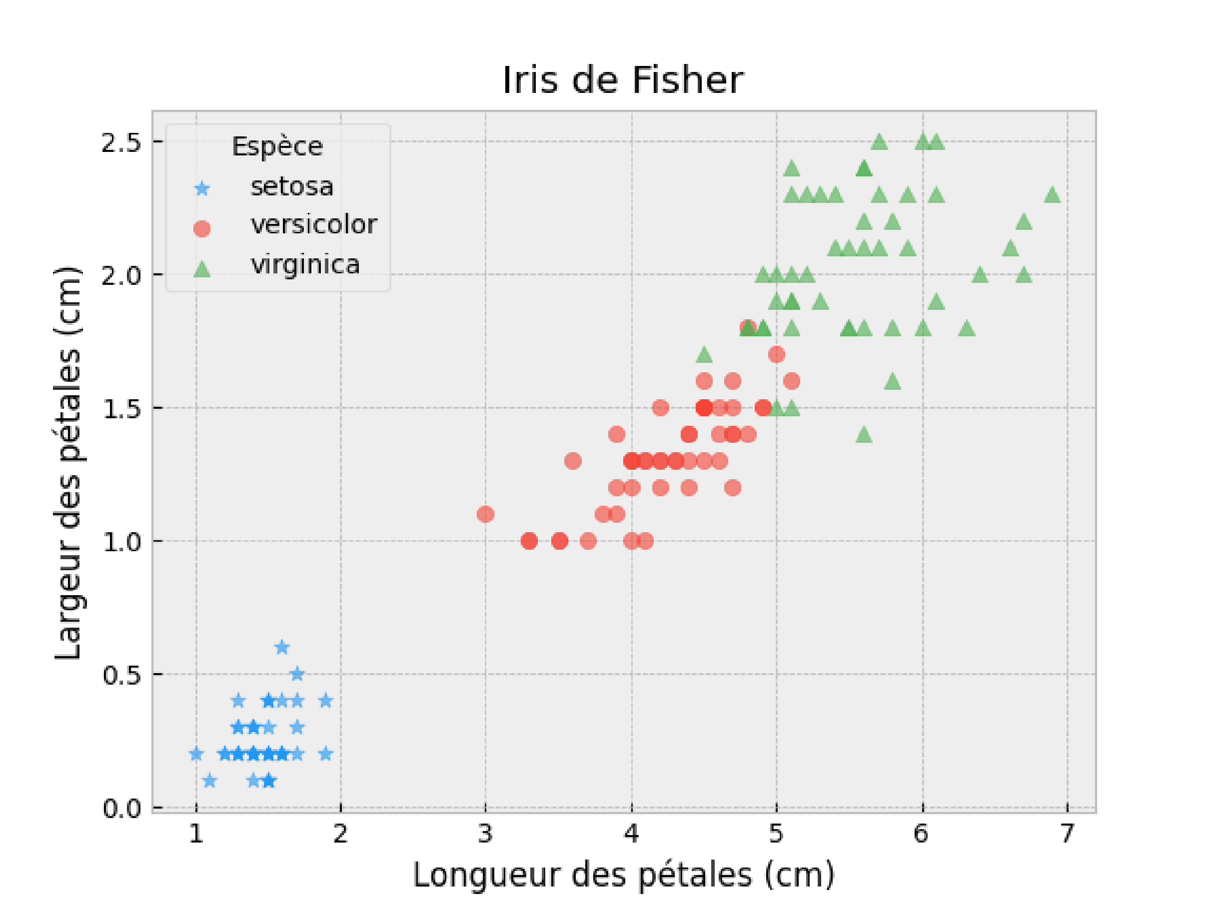
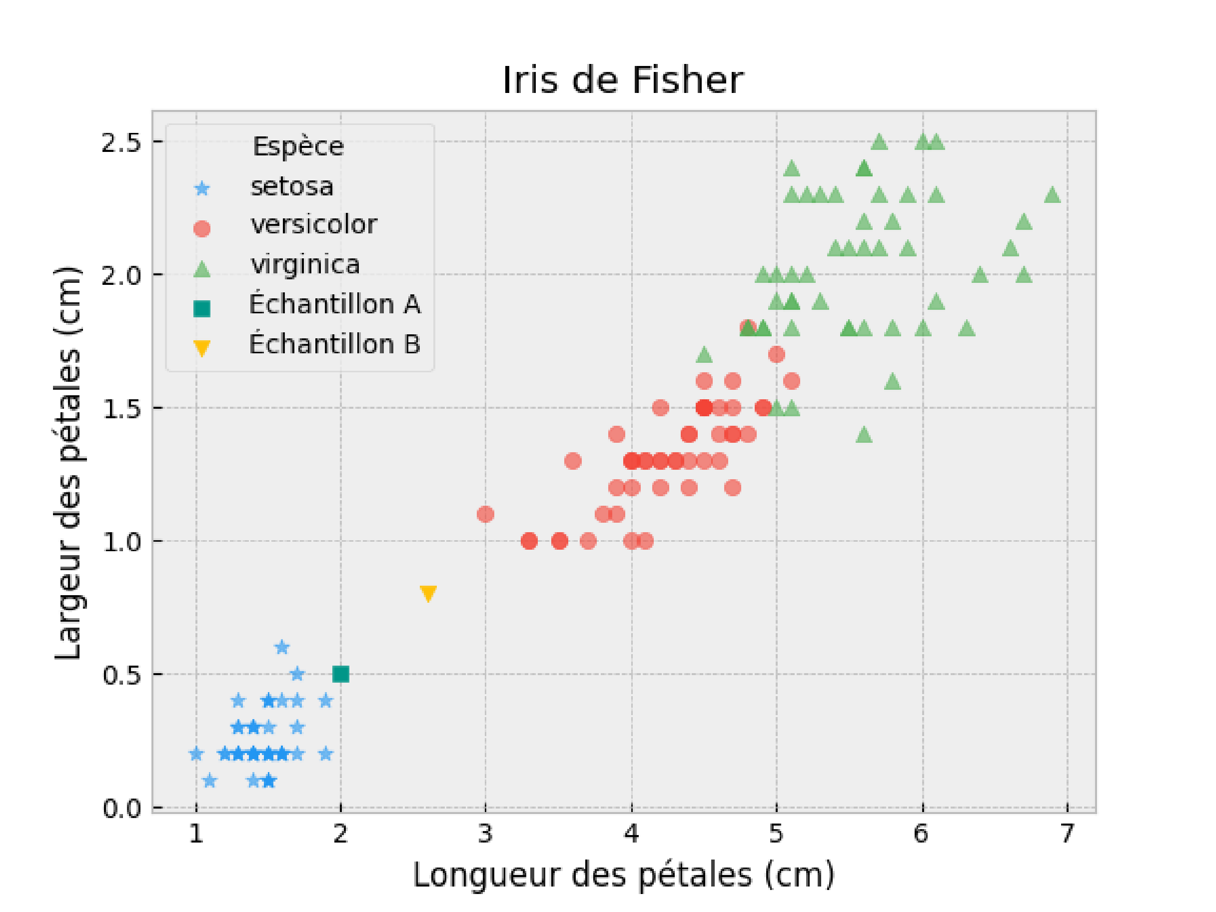
Classer la flore

Algorithme des k plus proches voisins

Un botaniste classe des iris suivant leur espèce



Et ramasse deux autres fleurs.
À quelle espèce appartiennent-elles ?



Objectif

- Écrire un algorithme qui prédit la classe d’un élément en fonction de la classe majoritaire de ses k plus proches voisins.

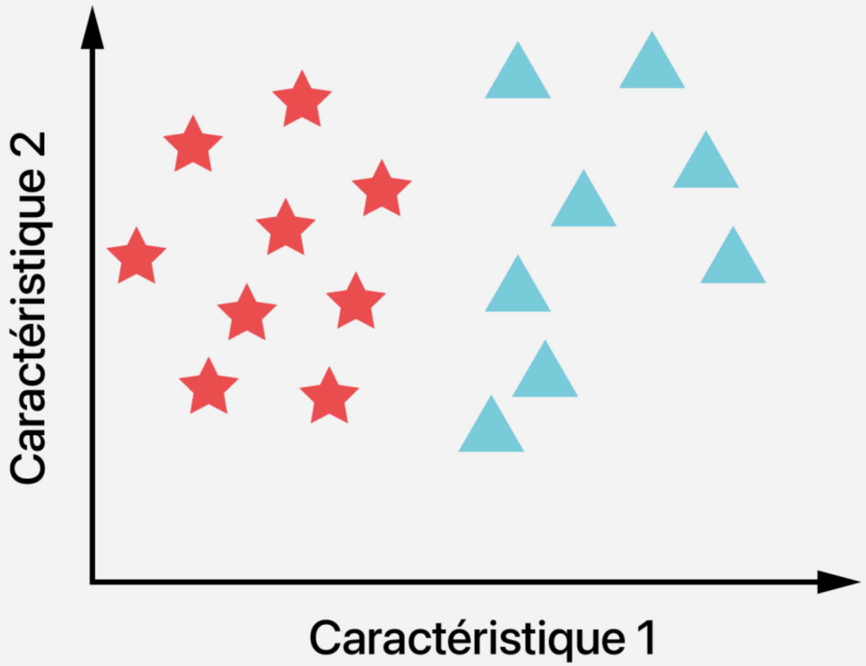
Présentation du problème

Algorithme des k plus proches voisins

Présentation du problème

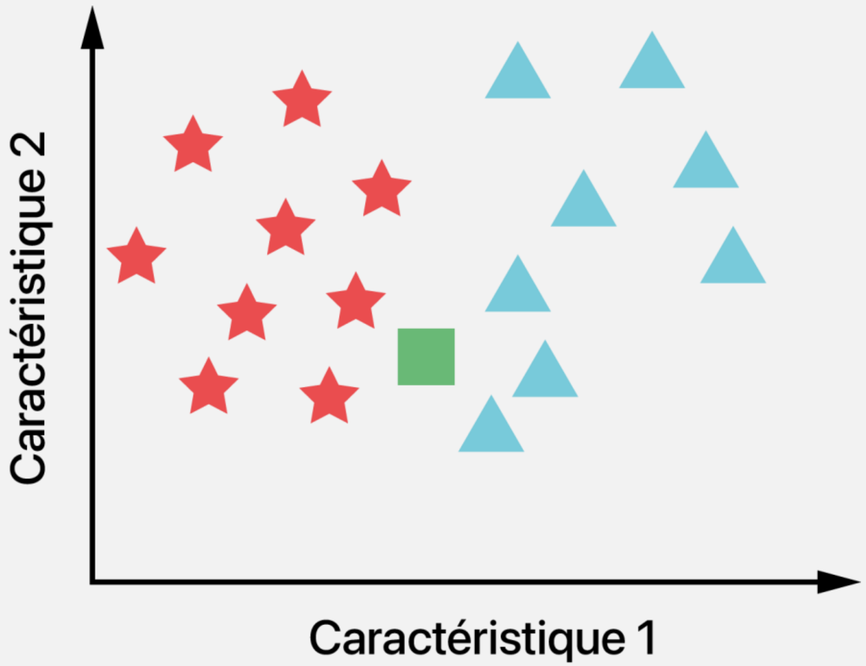
- **L'algorithme des k plus proches voisins** (en anglais « k nearest neighbors », KNN) appartient à la famille des **algorithmes d'apprentissage automatique** (**machine learning**).
- C'est un algorithme d'apprentissage supervisé, car il est nécessaire d'avoir des données **labellisées**.
- C'est un algorithme de **classification.** A partir d'un ensemble de données préalablement labellisées, il sera possible de classer (déterminer un label) une nouvelle donnée.

Un premier exemple



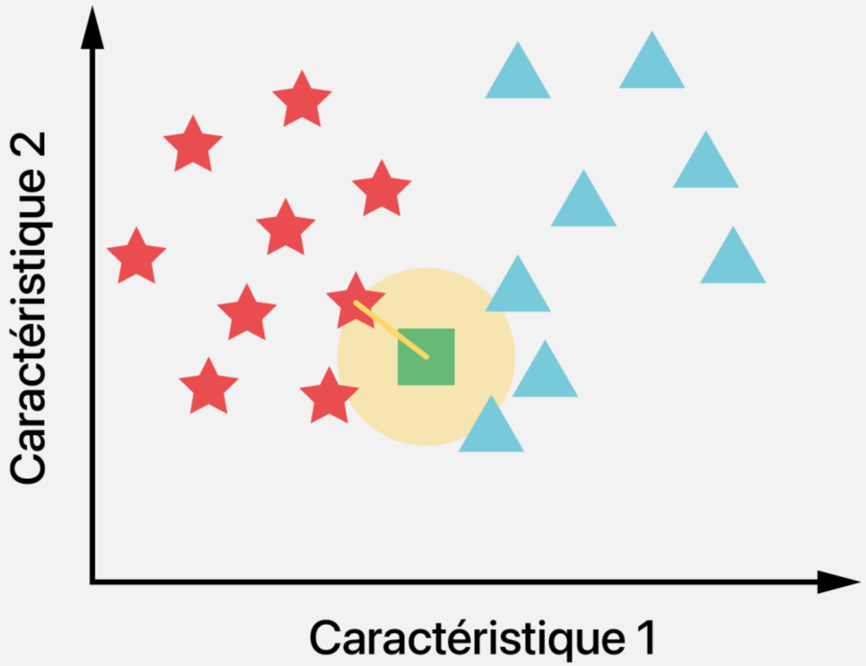
- On considère un **ensemble de données**, selon 2 labels (ou classes) : les **étoiles rouge** et les **triangles bleus**.
- Chacune des données a 2 **caractéristiques** (taille/poids, ou longueur/largeur par exemple)
- On peut donc les représenter dans un repère orthonormé.

Un premier exemple



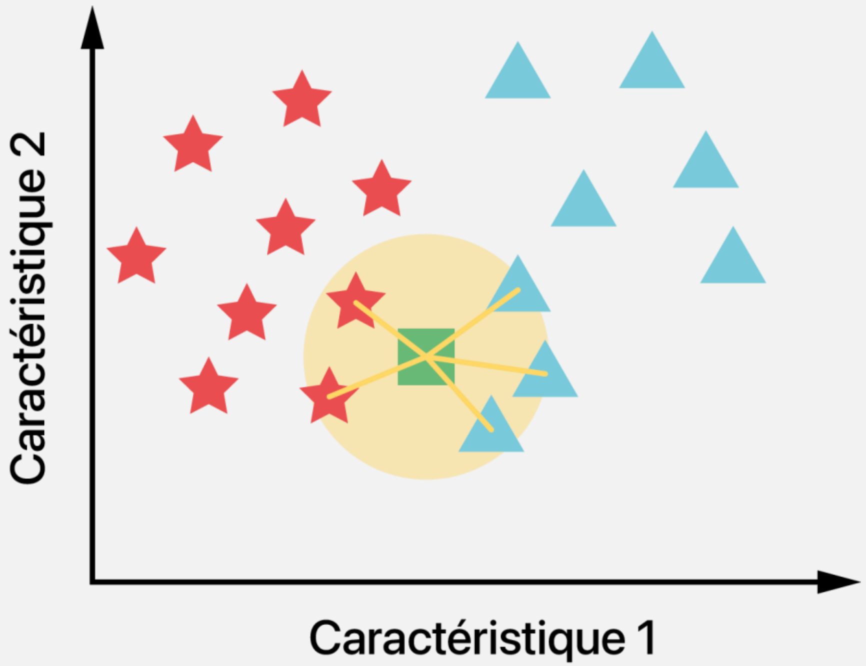
- Un **nouveau point** (une nouvelle donnée) dont on connaît les 2 caractéristiques se présente.
- Sa classe est **inconnue**.
- L'objectif est de lui **attribuer** une classe : étoile ou triangle.

Un premier exemple



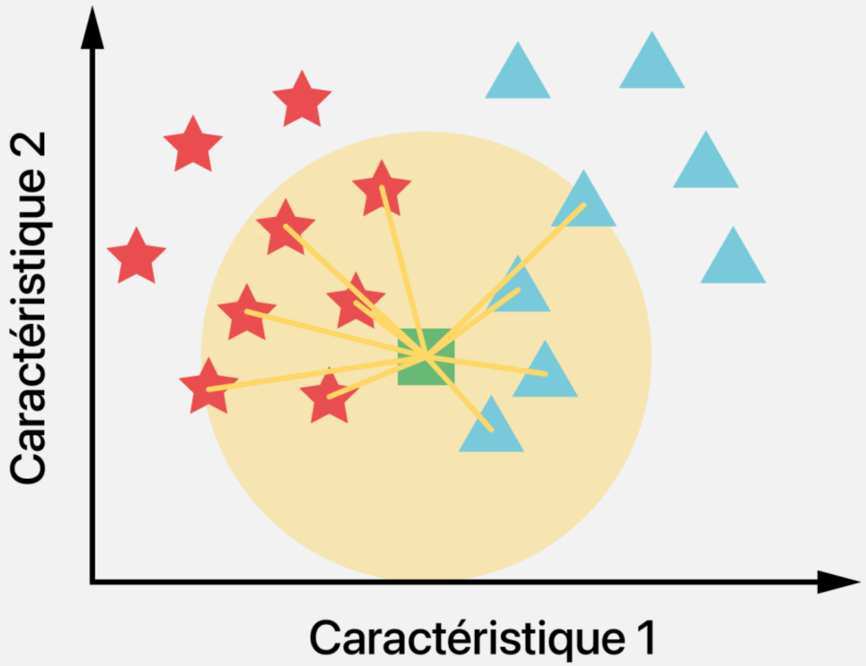
- Une méthode consiste à attribuer à ce nouveau point la même classe que le plus proche des points appartenant au nuage initial.
- C’est la méthode des **plus proches voisins**.
- Il reste à déterminer combien des k plus proches voisins on va considérer.

Un premier exemple – $k = 5$



- On considère les **5** plus proches voisins.
- Il y a 3 triangles et 2 étoiles.
- Il y a plus de triangle que d'étoiles, donc on affectera au carré la classe **triangle**.

Un premier exemple – $k = 10$



- On considère les **10** plus proches voisins.
- Il y a plus d'étoiles que de triangles, on donc affecte au carré la classe étoile.

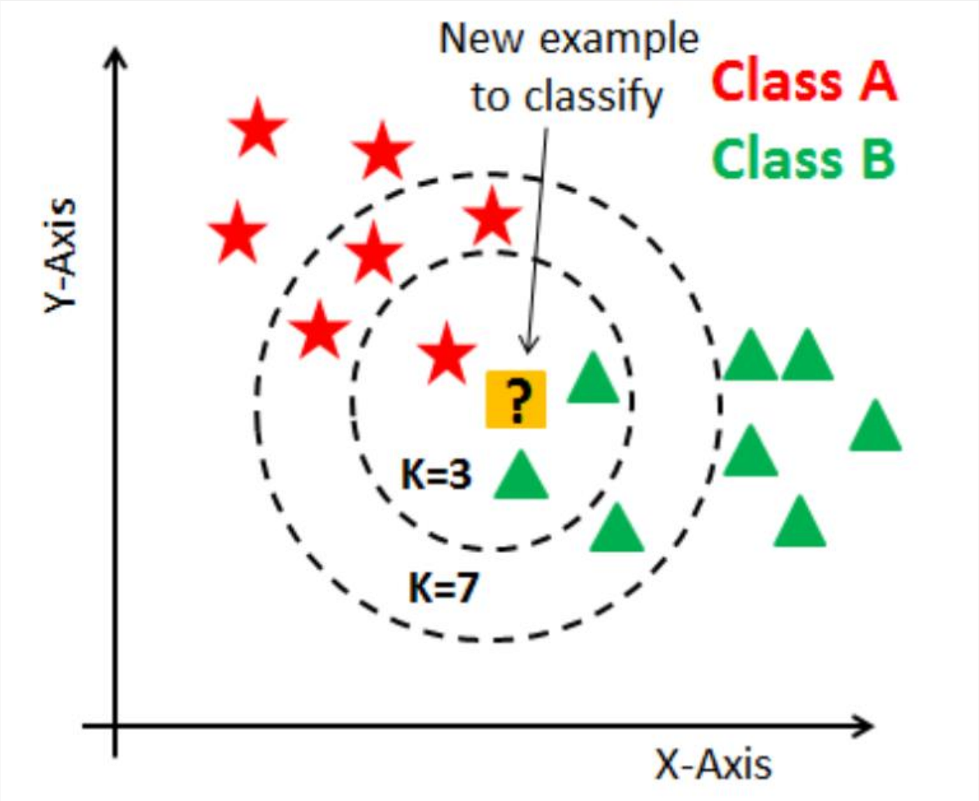
Principe de l'algorithme

Algorithme des k plus proches voisins

Les données du problème

- Soit E un **ensemble** contenant n **données préalablement labellisées**.
- Soit une **nouvelle donnée** C qui n'appartient pas à E dont on connaît ses caractéristiques (taille, poids, couleur etc.)
- Soit d la **fonction** qui renvoie la distance entre la donnée C et une donnée quelconque de E.
- Soit un **entier** $k \leq n$.

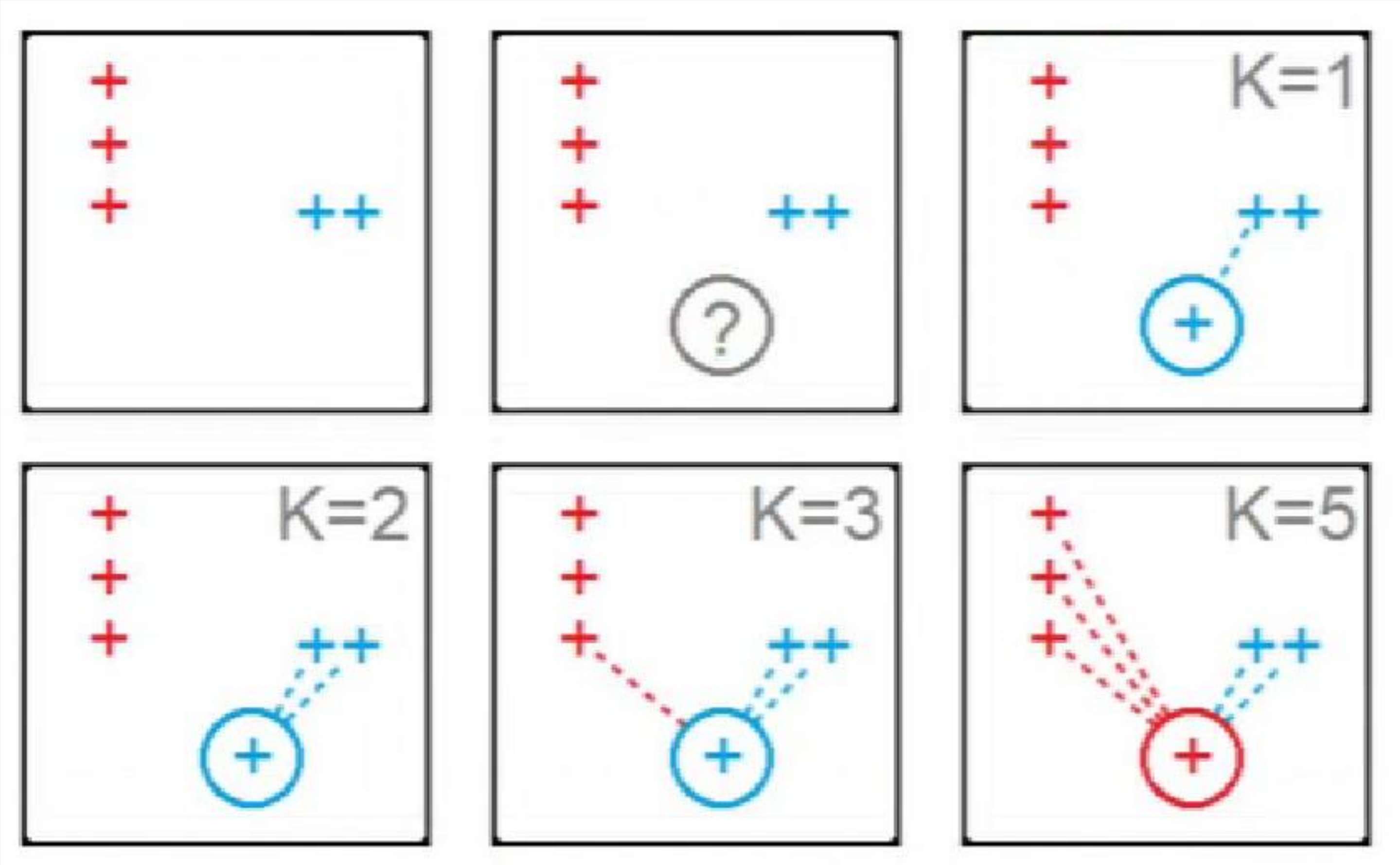
L'algorithme des k plus proches voisins



1. On calcule les **distances** entre la nouvelle donnée C et chaque donnée appartenant à E avec la fonction distance d.
2. On retient les $\boldsymbol{k}$ **éléments** de E les plus proches de C.
3. On attribue à C la classe qui est la plus **fréquente** parmi les k données les plus proches.

Le choix de k

- Le choix du paramètre k est **crucial**.



Compléments

Algorithme des k plus proches voisins

Le choix de k

- On détermine le paramètre k par des **tests successifs**.
- On introduit une donnée dont **on connaît à priori sa classe**.
- On regarde à partir de quel **seuil** k l'algorithme fonctionne **correctement**.

Le nombre de caractéristiques

- Les données peuvent avoir plus de 2 caractéristiques.
- Dans ce cas, on utilise la **formule générale** de la distance euclidienne :
- Soient **deux points** A et B dans un espace à n dimensions :

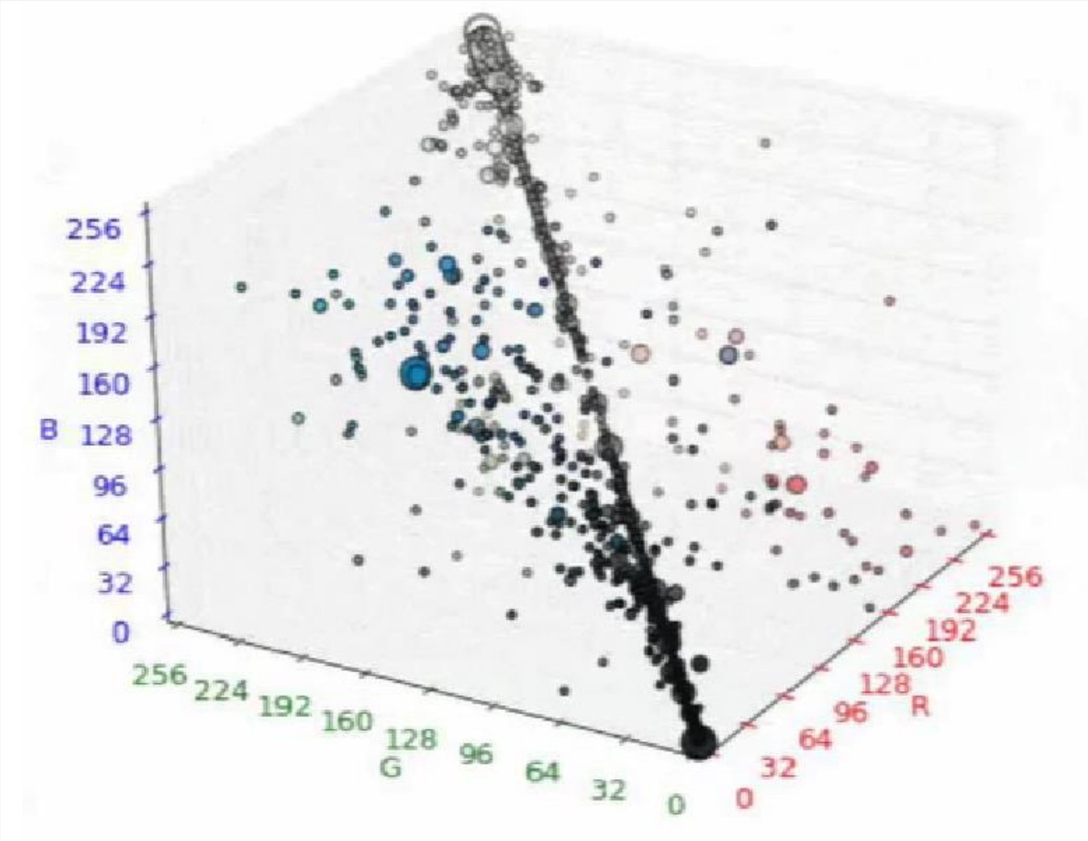
$$\begin{cases} A = (a_1, a_2, \dots, a_n) \\ B = (b_1, b_2, \dots, b_n) \end{cases}$$

- La **distance euclidienne** entre ces deux points est :

$$d(A, B) = \sqrt{(b_1 - a_1)^2 + (b_2 - a_2)^2 + \dots + (b_n - a_n)^2}$$

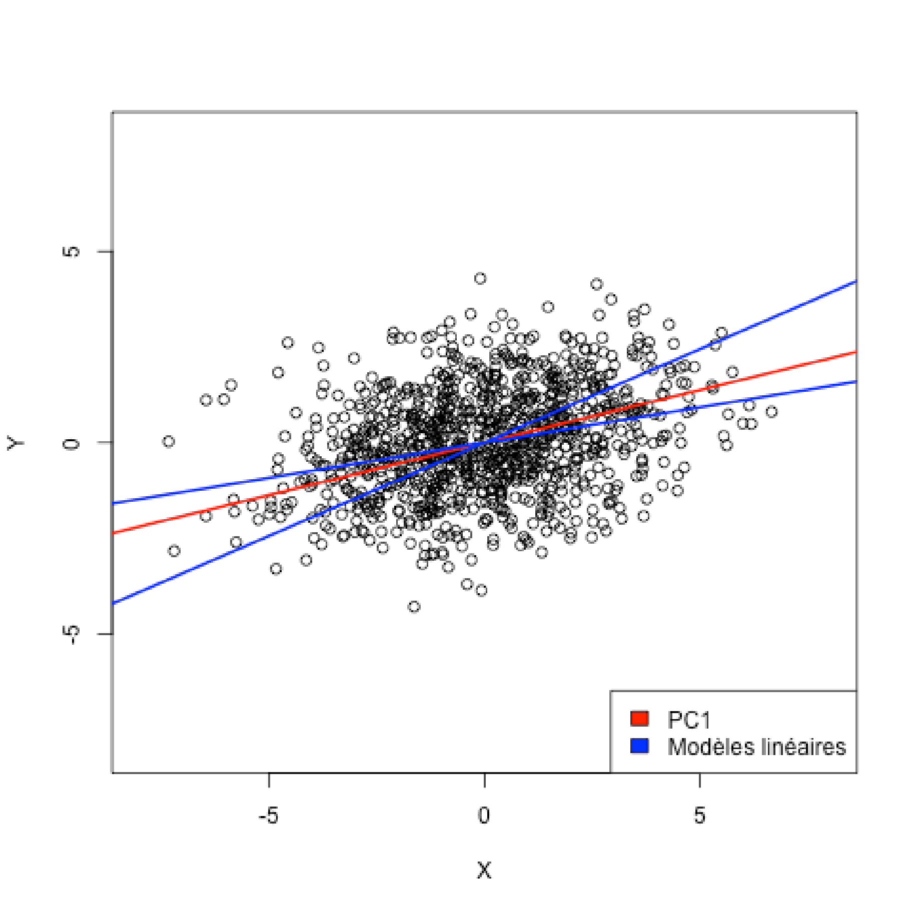
Le nombre de caractéristiques

- La **représentation** de telles données peut être complexe.
- A droite, on a représenté la couleur utilisée sur plusieurs pages webs.



Représentation des données où $n \geq 3$

- Pour un nombre de caractéristiques supérieurs à 3, on procède à une **Analyse en Composantes Principales** (ACP).
- Il s'agit de trouver une projection (« une photo en deux dimensions ») du **nuage du points** qui va le « déformer le moins possible ».
- On aura alors construit 2 nouvelles caractéristiques qui vont **agréger et résumer** l'information des caractéristiques initiales.

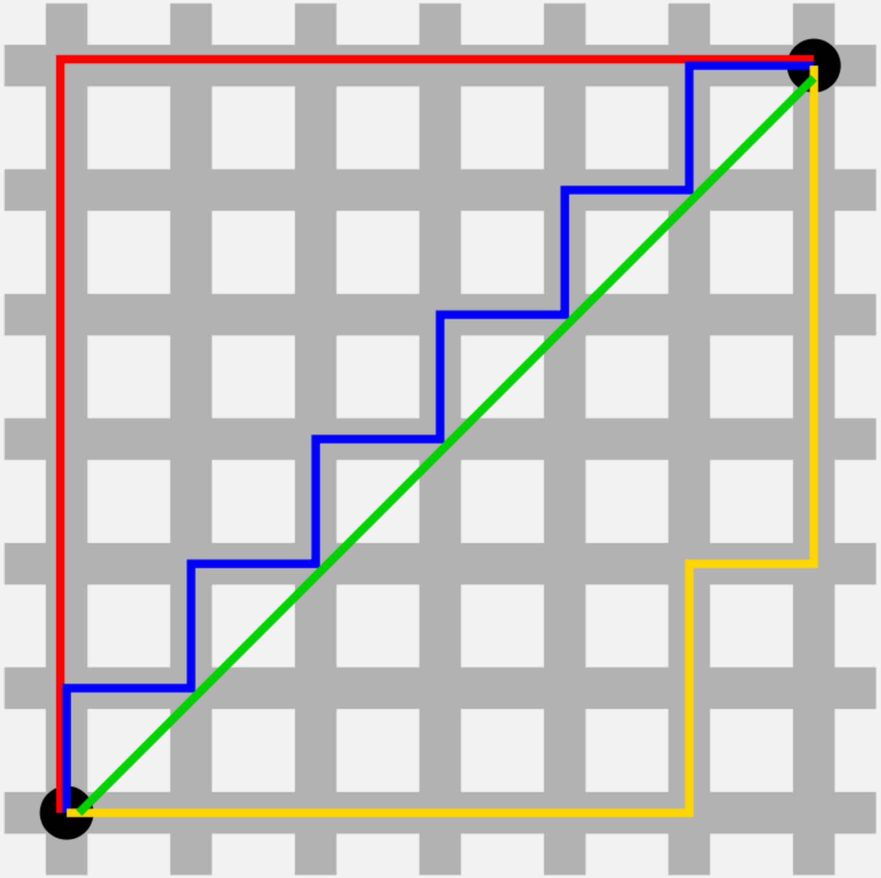


D’autres fonctions distances

- Il existe aussi d’autres distances : distance de **Manhattan**, de **Minkowski**, de **Tchebychev** etc.

La distance de Manhattan

- La **distance de Manhattan** est la distance entre 2 points parcourue par un taxi lorsqu’il se déplace dans une ville où les rues sont agencées selon un quadrillage.
- Ici, en vert la distance de euclidienne et en rouge, bleu ou jaune des distances de **Manhattan**. Celle qu’on utilise le plus communément est en rouge.



La distance de Manhattan

- La **distance de Manhattan** est définie comme :

$$d(A, B) = |a_1 - b_1| + |a_2 - b_2| + \cdots + |a_n - b_n|$$