# Algorithme des k plus proches voisins

## 

## 1. Introduction

En intelligence artificielle, il est souvent nécessaire de savoir classifier un objet selon une liste de catégories connues.

Par exemple:

* une image contient-elle un chat ou une voiture ?
* un email qu’on reçoit contient-il du spam ou pas ?

Il existe de nombreux algorithmes pour faire cela.

L’un d’eux s’appelle *Algorithme des k plus proches voisins* (*k-****N****earest* ***N****eighbors* en anglais ou *kNN*) où *k* est un nombre entier strictement positif correspondant au nombre de voisins.

## 

## 2. Principe du 1-NN

On considère un ensemble de objets dont on connaît la *classe* , , … .

On dispose d’une méthode de mesure de distance entre deux objets : .

On introduit un nouvel objet et on souhaite lui attribuer une classe selon l’algorithme suivant:

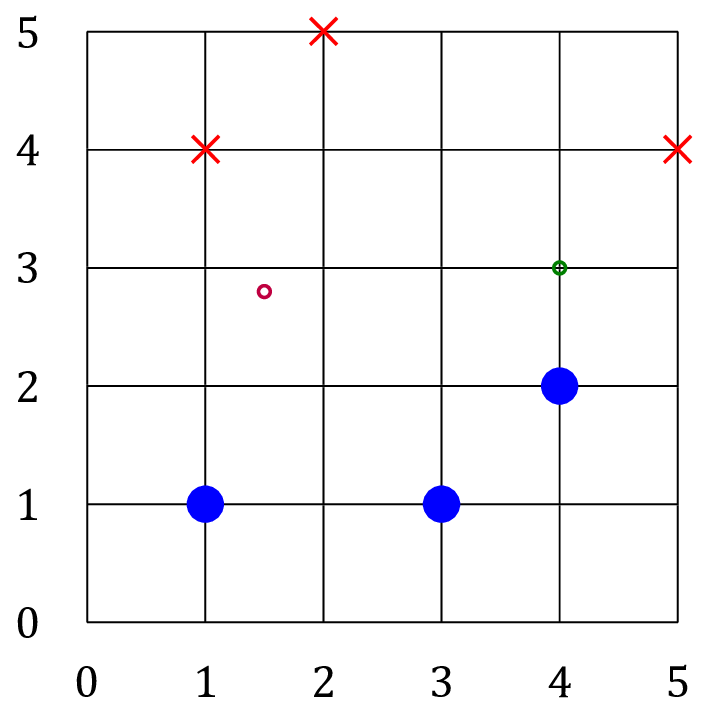
1. Recherche de l’objet (pour entre et ) qui est le plus proche de .
2. Attribution de la classe à .

Remarque : la recherche de l’objet le plus proche de consiste à trouver le minimum dans une liste de nombres réels correspondant à la distance entre et chacun des objet de l’ensemble .

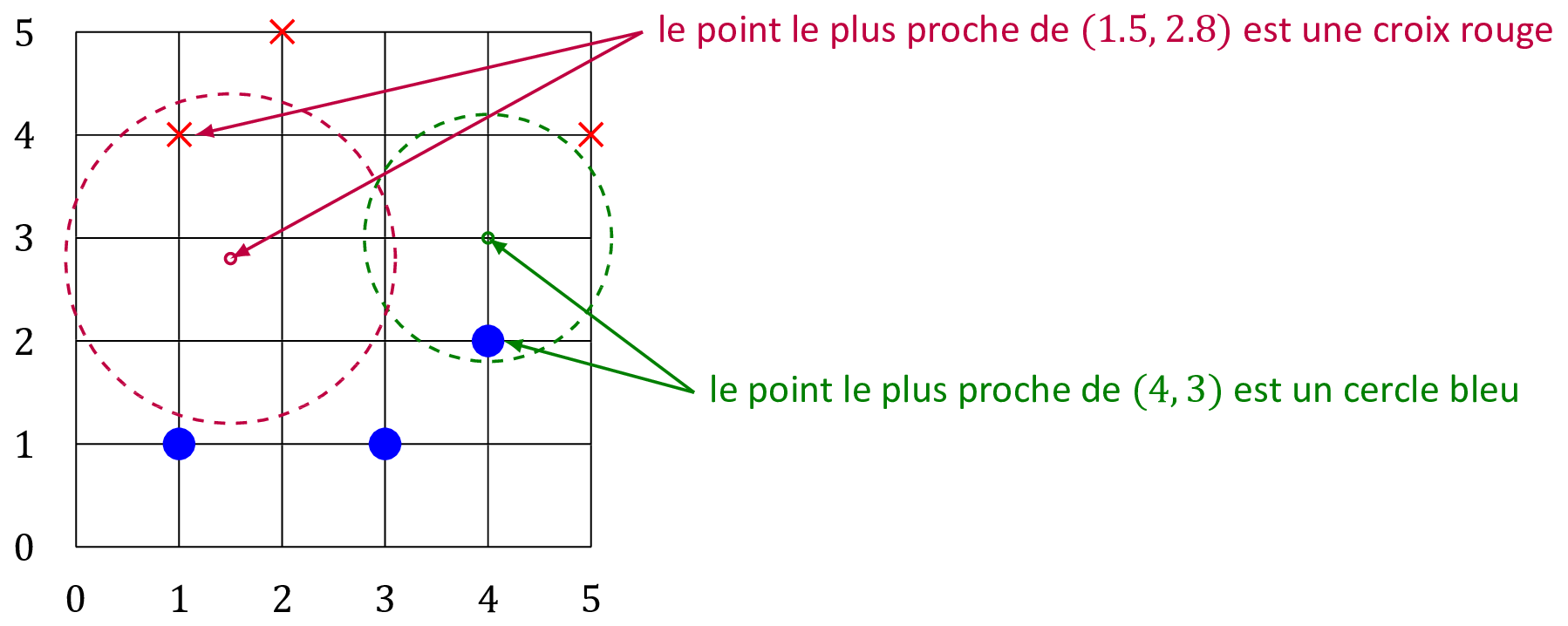
Fonction classe\_1nn(o)  
   
 distance\_min = Infini  
 voisin = Nul  
  
 Pour j allant 1 à n:  
 distance = d(o,o\_j)  
 Si distance < distance\_min Alors  
 distance\_min = distance  
 voisin = o\_j  
 FinSi  
 FinPour  
  
 Renvoyer c(voisin)

*Exemple :* Dans la figure ci-dessous, on connait la classe de 6 points (les cercles pleins bleus et les croix rouges)

On souhaite classer les points et



Pour les deux points à classer, on cherche parmi les points connus celui qui est le plus proche (= 1 voisin)



* doit être classé “cercle bleu”
* doit être classé “croix rouge”

Exercices n°1 question 1, n°2 et n°3 de la feuille d’exercices.

## 3. Principe du k-NN

Le contexte de travail est le même que pour l’algorithme 1-NN, mais on procède de la façon suivante :

1. Parmi les objets de , on recherche les voisins les plus proches de .
2. Parmi ces voisins, on compte les effectifs de chaque classe et on attribue à la classe majoritaire.

Remarque : dans le cas d’égalité d’effectifs entre deux classes, la classe attribuée dépendra de l’implémentation du point 2.

Algorithme naïf : on suppose

* qu’on dispose d’une fonction distance\_max(objet, liste) renvoyant la plus grande distance entre un objet et les objets de la liste
* qu’il n’y a que deux classes d’objets *c1* et *c2*

Fonction k\_voisins(o)  
 Pour j allant 1 à k:  
 ajouter o\_j à voisins  
 FinPour  
  
 dmax = distance\_max(o, voisins)  
  
 Pour j allant k+1 à n:  
 distance = d(o,o\_j)  
 Si distance < dmax Alors  
 Enlever le voisin le plus éloigné de o  
 Ajouter o\_j à voisins  
 dmax = distance\_max(o, voisins)  
 FinSi  
 FinPour  
  
 Renvoyer voisins  
  
Fonction classe\_knn(0)  
 voisins = k\_voisins(o)  
 effectif\_c1 = 0  
 effectif\_c2 = 0  
  
 Pour objet dans voisins:  
 Si c(objet) = c1  
 Augmenter effectif\_c1  
 Sinon  
 Augmenter effectif\_c2  
 FinSi  
 FinPour  
  
 Si effectif\_c1 > effectif\_c2  
 Renvoyer c1  
 Sinon  
 Renvoyer c2  
 FinSi

Exercices n°1 questions 2 et 3, n°4 et n°5 de la feuille d’exercice