

Algorithme des K plus proches voisins (KNN)

L'algorithme K-NN (K-nearest neighbors) est une méthode d'apprentissage supervisé. Il peut être utilisé aussi bien pour la régression que pour la classification. Son fonctionnement peut être assimilé à l'analogie suivante "dis moi qui sont tes voisins, je te dirais qui tu es...".

Méthode de prédiction

Pour effectuer une prédiction, l'algorithme K-NN va se baser sur le jeu de données en entier. En effet, pour une observation, qui ne fait pas parti du jeu de données, qu'on souhaite prédire, l'algorithme va chercher les K instances du jeu de données les plus proches de notre observation. Ensuite pour ces K voisins, l'algorithme se basera sur leurs variables de sortie (output variable), pour calculer la valeur de la variable de l'observation qu'on souhaite prédire.

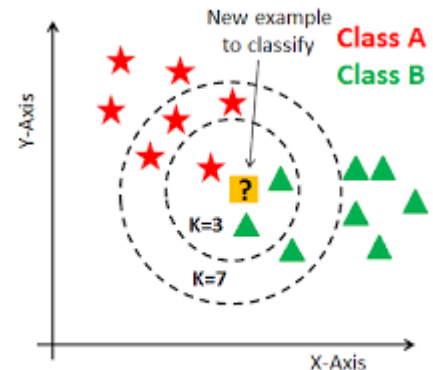


Figure 1: Méthode de prédiction

Programme Python

```

26 def distance(p1,p2):
27     """
28     Aide de la fonction distance du module knn :
29
30     distance(p1,p2)
31         Calcule la distance Euclidienne separant deux points p1 et p2
32
33     ->Parametres : p1 : coordonnes de p1 (liste d'entiers)
34                   p2 : coordonnes de p2 (liste d'entiers)
35
36     -> Retour : distance separant p1 et p2 (float)
37     Exemple : distance([0,1] , [1,1]) renvoie 1.0
38     """
39     somme=0
40     for i in range(len(p1)):
41         somme+=(p1[i]-p2[i])**2
42     return sqrt(somme)
43
44 def distance_voisins(p1,jeu2donnees):
45     """
46     Aide de la fonction distance_voisins du module knn :
47
48     distance(p1,dataset)
49         Calcule la distance Euclidienne separant le point p1 et des diffrents points du jeu de donnees (dataset)
50
51     ->Parametres : p1 : coordonnes de p1 (liste d'entiers)
52                   jeu2donnees : liste de tuples : [('nom du joueur','poste du joueur','taille','masse')....]
53
54     -> Retour : liste de listes contenant l'les caractéristiques du joueur et la distance entre ce joueur et le joueur p1
55     Exemple : distance_voisins([180,90] , [('joueur1','arriere','175','75'), ('joueur2','avant','190','90')]) renvoie
56     [[('joueur1','arriere','175','75'), 15.811388300841896], [('joueur2','avant','190','90'), 10.0]]
57     """
58     result = []
59     i=0
60     for i in range(len(jeu2donnees)):
61         result.append([jeu2donnees[i],distance(p1,[int(jeu2donnees[i][2]),int(jeu2donnees[i][3])])])
62     return result
63

```

```

63
64 def k_voisins(distances,k) :
65     '''
66     Aide la fonction K_voisins du module Knn :
67
68     k_voisins(jeu2donnees,k)
69     Renvoie la liste des K plus proches voisins à partir de la liste des distances
70     Parametres : distances : liste de listes contenant les caracteristiques du joueur et la distance entre ce joueur et le joue
71                  K : nombre de voisins considérés (int)
72     Retour : Liste des K plus proches joueurs
73     '''
74     k_voisins = []
75     for i in range(k) :
76         dmin = distances[0][1]
77         index = 0
78         for i in range(len(distances)) :
79             if distances[i][1] < dmin :
80                 dmin = distances[i][1]
81                 index = i
82         k_voisins.append(distances.pop(index))
83     return k_voisins
84
85 def predire_classe(k_voisins):
86     '''
87     Aide de la fonction predire_classe du module knn :
88
89     predire_classe(proches joueurs)
90     Renvoie la classe plus proche voisin du point p1
91
92     ->Parametres : k_voisins : liste des k voisins
93
94     -> Retour : Poste (classe) du plus proche voisin de p1
95     '''
96     postes = {}
97     #Creer dictionnaire avec classes et nombre d'occurrence de chaque classe
98     for joueur in k_voisins :
99         if joueur[0][1] in postes :
100             postes[joueur[0][1]] +=1
101         else :
102             postes[joueur[0][1]] = 1
103     #Cherhe maximum du nombre d'occurence
104     n = 0
105     for classe in postes :
106         if postes[classe] > n :
107             n = postes[classe]
108             classe_majoritaire = classe
109     return classe_majoritaire
110
111
112
113 dataset = extractionDonnees("joueursToulouse.csv")
114 distances = distance_voisins([185,114],dataset)
115 knn = k_voisins(distances,10)
116 prediction = predire_classe(knn)

```