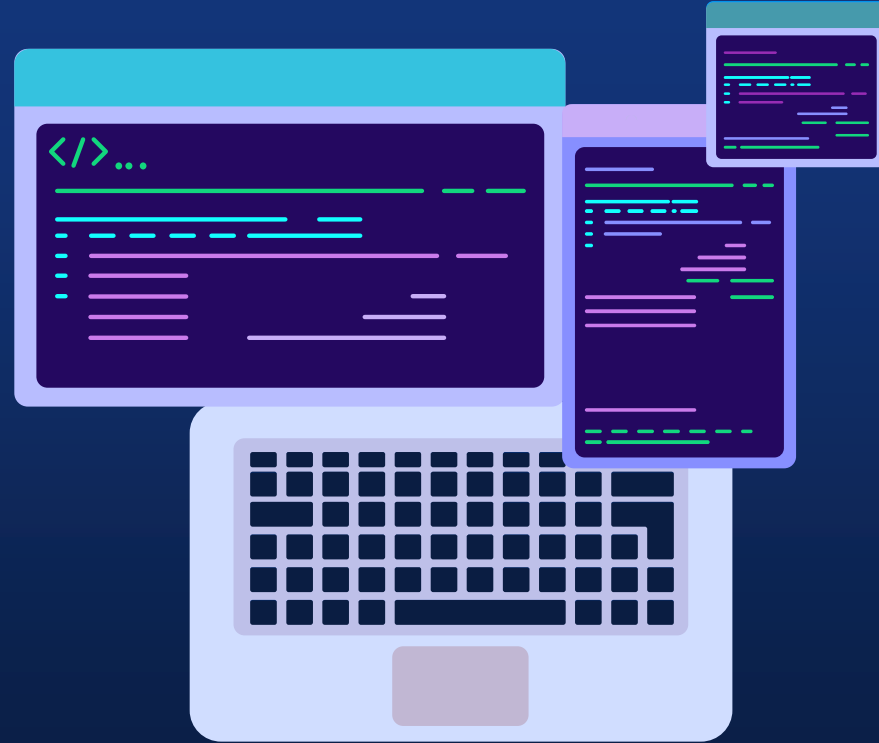


k-Nearest Neighbors algorithm

en langage C

Realise par :

- Maraoui Yassine
- Ismail Faroui



Plan :

- Introduction
- l'apprentissage supervisé
- l'algorithme K-NN
- Fonctionnement de l'algorithme
- PSEUDO CODE K-NN
- Avantages et Inconvénients
- Application





Introduction :

Le machine Learning devient la priorité de bon nombre laboratoire de recherche notamment celles de biologie et de médecine. Elles veulent modéliser d'importants volumes de données. Le choix du bon algorithme dépend des objectifs à atteindre et de la maturité de l'équipe de Data Science. le K-NN parmet les algorithmes les plus connue et utiliser dans le domaine de la l'apprentissage supervisé pour aidé les chercheur de prédicteur les cas maladies





L'apprentissage supervisé :

L'algorithme d'apprentissage supervisé est une sorte d'algorithme dans lequel il s'appuie sur une entrée étiquetée pour apprendre et prédit en fonction de la fonction lorsque des données non étiquetées sont fournies. Comme nous avons compris ce qu'est l'apprentissage supervisé, voyons ce qu'est la classification, l'algorithme de classification donne une valeur discrète en sortie, pas des valeurs continues.





l'algorithme K-NN :

L'algorithme des k plus proches voisins s'écrit souvent KNN de l'anglais K-Nearest Neighbors. Il est une méthode fait partie de la partie classification de l'apprentissage supervisé, il peut être utilisé aussi bien pour la régression que pour la classification qui essentiel dans le milieu l'intelligence artificielle. Son fonctionnement peut être assimilé à l'analogie suivante << dis moi qui sont tes voisins, je te dirais qui tu es...>>. Pour effectuer une prédiction, l'algorithme K-NN ne va pas calculer un modèle prédictif à partir d'un Training Set comme c'est le cas pour la régression logistique ou la régression linéaire. En effet K-NN ,n'a pas besoin de construire un modèle prédictif. par exemple utilisé prédire le comportement d'une personne en s'intéressant à son milieu. Les géants de la vente en ligne comme Amazon, Netflix, ... l'utilise afin de prévoir si vous seriez intéressé ou non par un produit, un film





FONCTIONNEMENT DE L'ALGORITHME :

K Nearest Neighbors est un algorithme se baser sur le jeu de données en entier. Ensuite, l'algorithme assigne l'étiquette de cette donnée d'apprentissage à la nouvelle observation qui était inconnue. Le k dans la formule "k plus proches voisins" signifie qu'à la place de se contenter du seul voisin le plus proche de l'observation inconnue, nous pouvons prendre en compte un nombre fixé k de voisins du jeu d'apprentissage. Enfin, nous pouvons faire une prédiction en nous basant sur la classe majoritaire dans ce voisinage.



PSEUDO CODE K-NN



Pseudo code :

Debut Algorithme

Pour une nouvelle observation inconnue en entrée dont on veut prédire sa variable de sortie, il faut faire :

Etape 1: Calculer toutes les distances entre cette observation en entrée et les autres observations du jeu de données,

Etape 2: Conserver les k observations du jeu de données qui sont les plus « proches » de l'observation à prédire, en utilisant la fonction de calcul de distance

Etape 3: - Prendre les valeurs des observations retenues :

Si on effectue une régression, l'algorithme calcule la moyenne des valeurs des observations retenues,

Si on effectue une classification, l'algorithme assigne le label de la classe majoritaire à la donnée qui était inconnue.

Etape 4: - Retourner la valeur calculée dans l'étape 3 comme étant la valeur qui a été prédite par l'algorithme pour l'observation en entrée qui était inconnue.

Fin Algorithme

Pour cet algorithme, le choix du nombre k et le choix de la fonction de similarité sont des étapes qui peuvent conduire à une forte variabilité des résultats.



Avantages et Inconvénients

Avantages

- L'algorithme est facile à mettre en œuvre
- Il n'est pas nécessaire de construire un modèle, d'ajuster plusieurs paramètres ou de faire des hypothèses supplémentaires.
- Il peut être utilisé pour la classification, la régression et la recherche d'informations

Inconvénients

L'algorithme ralentit considérablement à mesure que le nombre d'observations et/ou de variables dépendantes/indépendantes augmente. En effet, l'algorithme parcourt l'ensemble des observations pour calculer chaque distance.





APPLICATION :

Dans cette application on traite un exemple de l'algorithme K-NN qui permet de détecter si une cellule est infectée ou non.

Définition des données

Il existe bien évidemment plusieurs façons de définir nos éléments, On pourra dans la suite définir nos données avec des Pointeurs, Tableau et Structure.




Structure :

La structure utilisé nommer Cellule chaque cellule définie par ces cordonnée (x,y),le type (sain ou infecter) et la distance entre deux cellule :

```
typedef struct{  
    int type;  
    double x,y;  
    double distance;  
} cellule;
```

Les différent variables utilisée



```
int Type,i,k=3;
cellule arr[17],*p;
p = arr;
int n = sizeof(arr)/sizeof(arr[0]);
char buf[1024];
int line_count=0;
int case_count=0;
FILE *F;
```



Les fonctions

On met en œuvre au minimum 2 fonctions :

Une fonction `celluleClass` : pour le calcul de la distance euclidienne entre la cellule de test et tous les autres cellules.

DISTANCE

$$= \sqrt{(\mathbf{x}_i - \mathbf{x}_p)^2 + (\mathbf{Y}_i - \mathbf{Y}_p)^2}$$

détermine le résultat majoritaire des classes d'appartenance des k plus proches voisins et assigne la classe du nouvel élément à cette classe majoritaire



CelluleClass code :

```
int celluleClass(cellule *pointer,int n,int k,cellule p){
    int i,j,V_count=0,R_count=0;
    for (int i = 0; i < n; i++)
    {
        (pointer+i)->distance=sqrt(((pointer+i)->x - p.x)*((pointer+i)->x - p.x)+((pointer+i)->y - p.y)*((pointer+i)->y - p.y));
    }
    triDistance(pointer,n);
    for(j=0;j<k;j++){
        if((pointer+j)->type == 0){
            V_count++;
        }
        else
            R_count++;
    }

    return (V_count>R_count?0:1);
}
```

TriDistance :

Le procédure `triDistance` : permet de trier les distance calculer dans la fonction `celluleClass`

```
void triDistance(cellule *pointer, int n)
{
    cellule temp;
    int i, j, min_idx;
    for (i = 0; i < n-1; i++)
    {
        min_idx = i;
        for (j = i+1; j < n; j++)
            if ((pointer+j)->distance < (pointer+min_idx)->distance)
            {
                min_idx = j;
            }

        temp = *(pointer+min_idx);
        *(pointer+min_idx) = *(pointer+i);
        *(pointer+i) = temp;
    }
}
```




Choisir la bonne valeur pour k

Pour sélectionner la valeur de k qui convient à vos données, nous exécutons plusieurs fois l'algorithme KNN avec différentes valeurs de k . Puis nous choisissons le k qui réduit le nombre d'erreurs rencontrées tout en maintenant la capacité de l'algorithme à effectuer des prédictions avec précision lorsqu'il reçoit des données nouvelles



Dans notre cas on a choisir $k=3$ c'est a dire que notre programme il va ce base sur les trois plus proche voisins pour donner le résultat.



```
int Type,i,k=3;
cellule arr[17],*p;
p = arr;
int n = sizeof(arr)/sizeof(arr[0]);
char buf[1024];
int line_count=0;
int case_count=0;
FILE *F;
```


TriDistance :

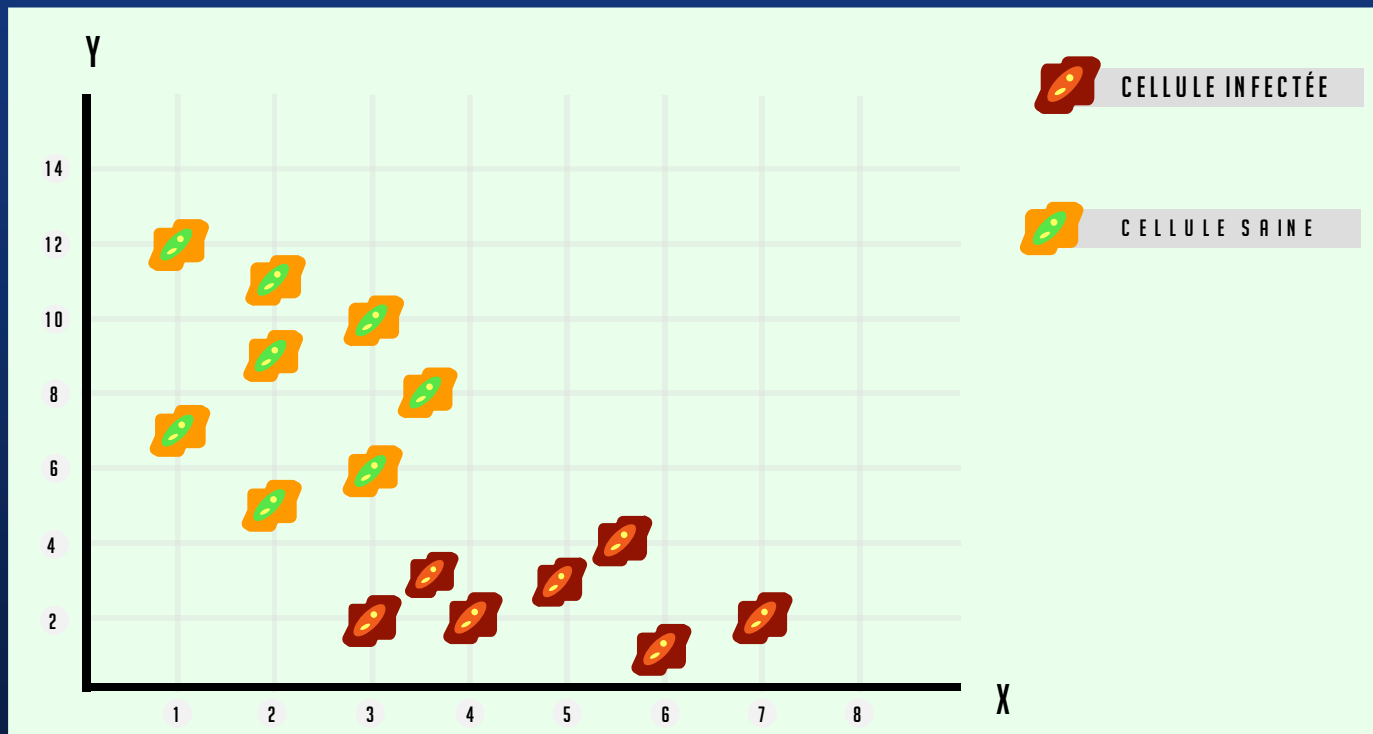
Data-Set : le data-set utilisé pour tester l'application est de format « .csv »

x	y	Type
1	12	0
2	5	0
5	3	1
3	2	1
3	6	0
1.5	9	1

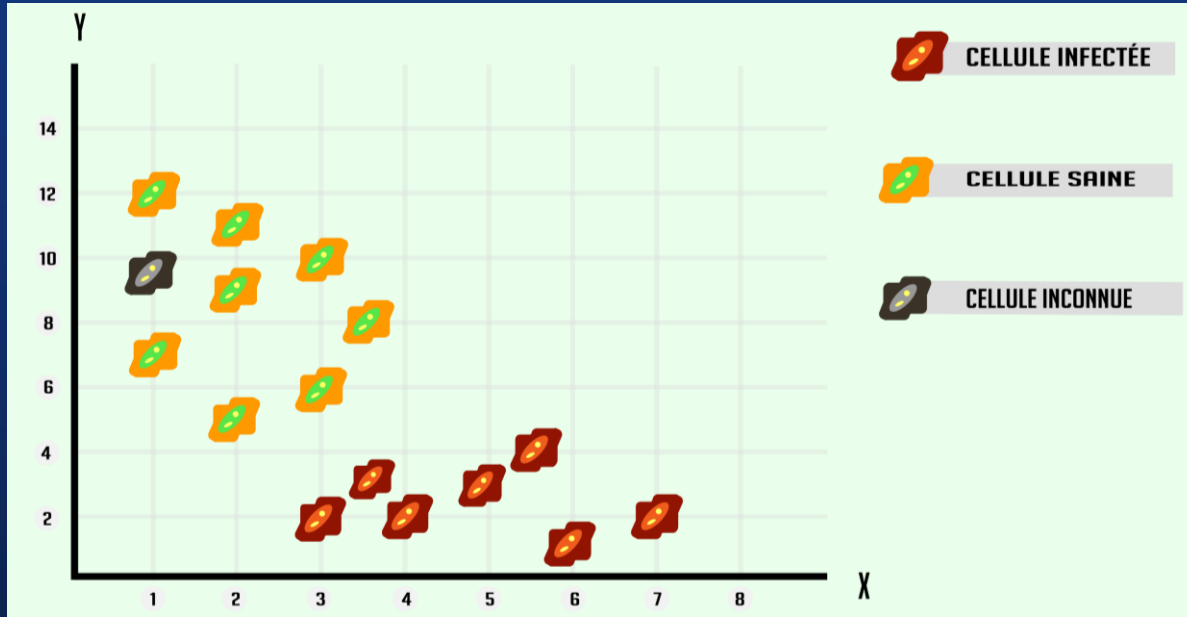
Pour les types :

- 1 signifie que la cellule est **saine**
- 2 signifie que la cellule est **infectée**


Tous les données dans le dataset sont collecter en ce basant sur c'est illustration



Test de programme :




on va tester notre programme sur la cellule de type inconnu qui a les cordonner suivantes : $X = 1$ et $Y = 9.8$



```
Application
===== DONNER LES COORDONNEES DE VOTRE CELLULE =====
X: 1
Y: 9.8
=====
Tapez 1 pour saine
Tapez 0 pour infectee
=====
Votre attente ? : _
```

Notre cellule doit être saine car il se trouve dans la zone des cellules saines.





```
Application [X] + -
===== DONNER LES COORDONNEES DE VOTRE CELLULE =====
X: 1
Y: 9.8
=====
Tapez 1 pour saine
Tapez 0 pour infectee
=====
Votre attente ? : 1
=====
La cellule est saine
=====
Tapez 1 si oui
Tapez 0 si non
Voulez-vous tester d'autre cellule ?
-
```

Le programme il nous donne que la cellule est sain.

