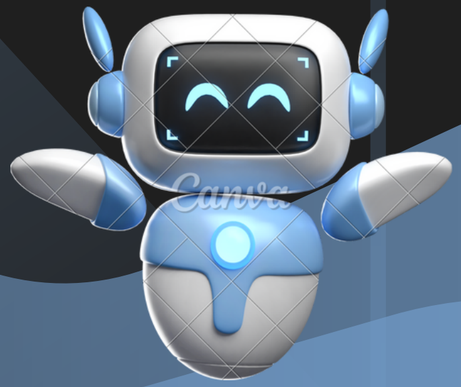
Regnier Mathilde - Le Beaudour Bastien - Fourre Mael

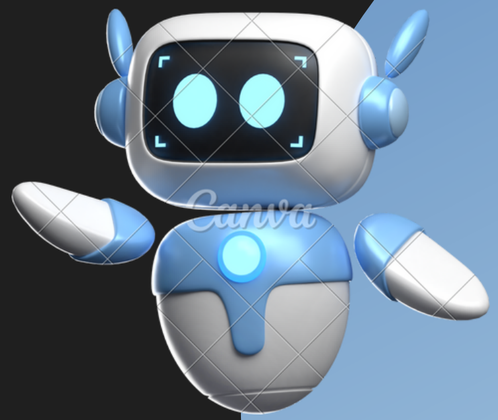
Présentation des algorithmes de KNN et K-means



2024-2025

SAE Exploration algorithmique d'un problème

Sommaire



- Objectifs
- Description
- Limites et obstacles

Regnier Mathilde - Le Beaudour Bastien - Fourre Mael

Objectifs

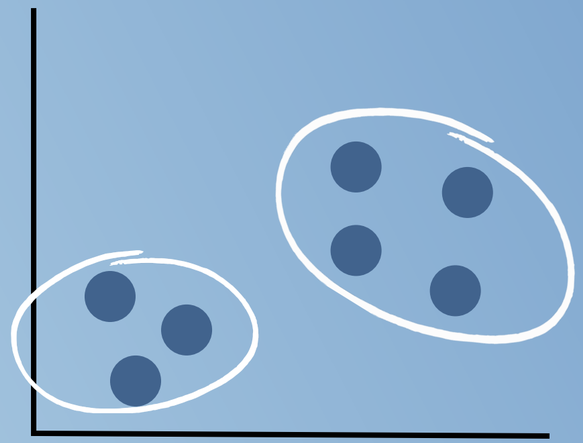
KNN

Algorithme d’apprentissage automatique supervisé

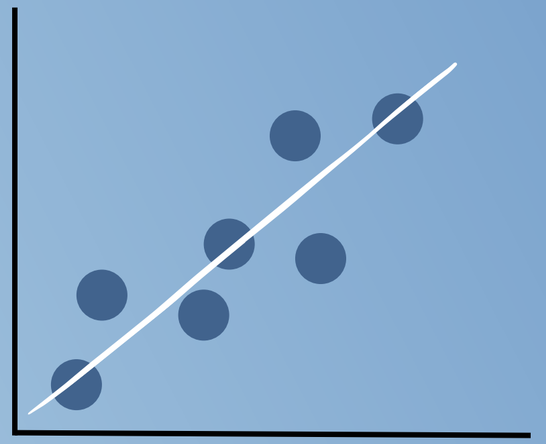
Prédire la valeur d'un point

Des objets similaires sont proches les uns des autres

Classification ou régression



Classification



Régression

Objectifs

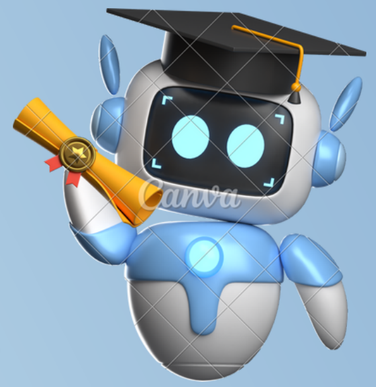
KNN

Algorithme d’apprentissage automatique supervisé

Prédire la valeur d'un point

Des objets similaires sont proches les uns des autres

Classification ou régression



K-Means

Trier des points

Cluster

Centroïde

Description

KNN

Données en tuples

Structures en liste

```
def knn(donnees, data, k, p, choix):
```

Manhattan, Euclidienne, Chebyshev, Minkowski,

Première partie :

```
for donnee_id in range(nb_donnee):
    distances.append((donnee_id, Distance(p, data[0], donnees[donnee_id][0])))

distances.sort(key=lambda x: x[1])
```

Deuxième partie :

```
for donnee_id in range(k):
    if choix=='reg':
        regression=0
        regression+=distances[donnee_id][0]
        regression=regression / k
        return regression
```

Description

K-Means

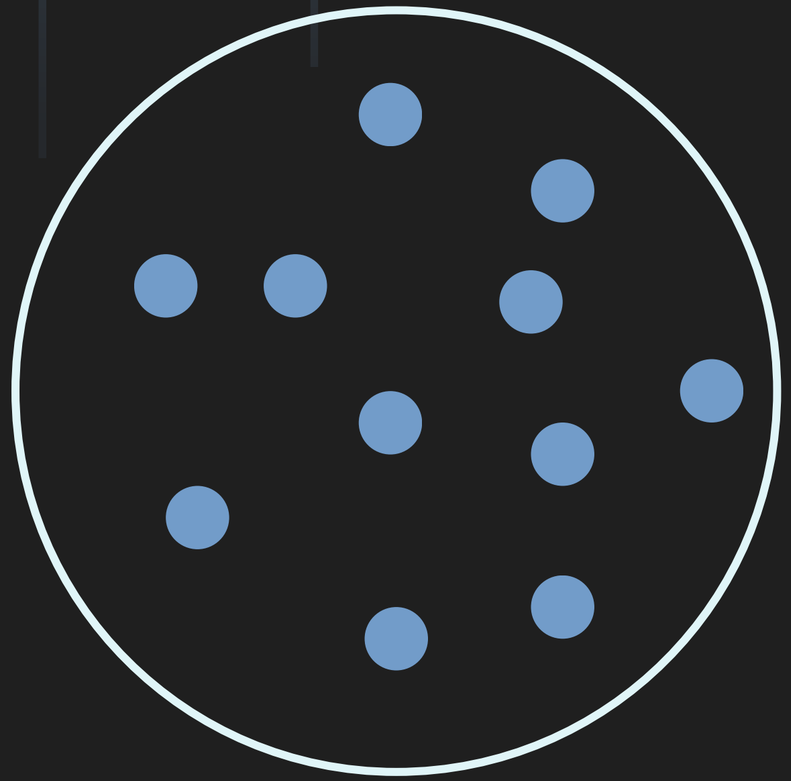
```
def Kmeans(lst : list[tuple], k : int):
```

Liste de points

nombre de centroïdes

Description

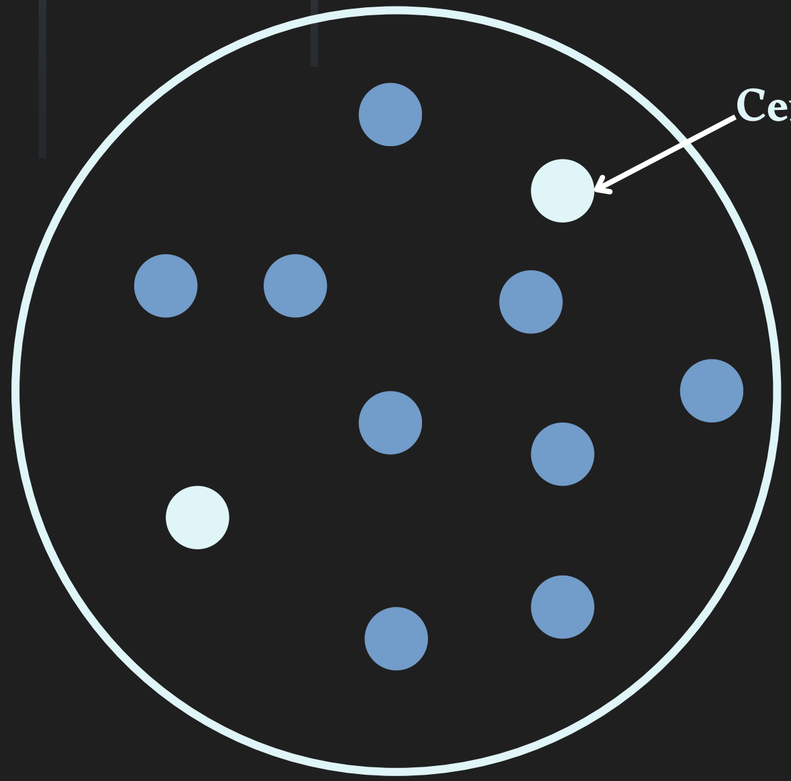
K-Means



```
n = len(lst[0]) # nombre de dimensions
clusters = [[] for x in range(k)]
centroides = [lst[randint(1, len(lst))-1] for x in range(k)]
```

Description

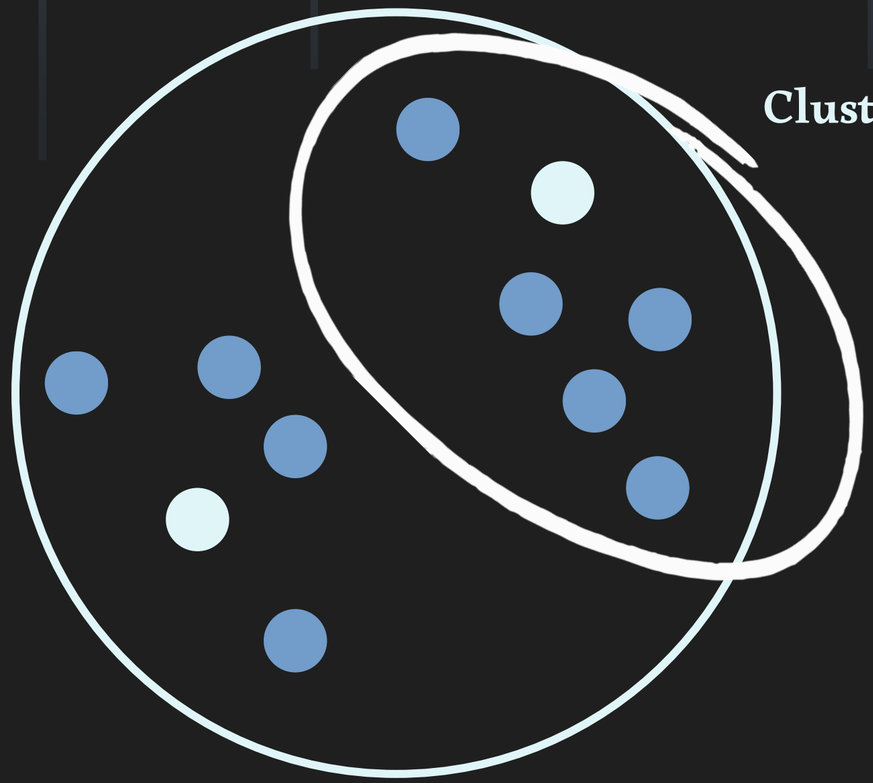
K-Means




```
n = len(lst[0]) # nombre de dimensions
clusters = [[] for x in range(k)]
centroides = [lst[randint(1, len(lst))-1] for x in range(k)]
```

Description

K-Means




```
clusters = doClusters(lst, centroides, k)
```

Description

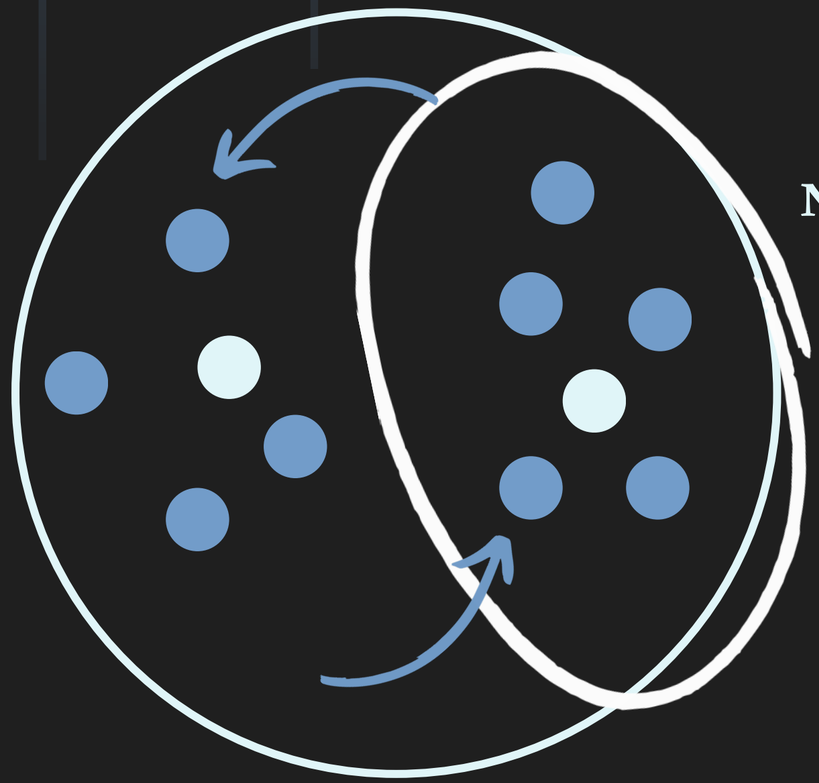
K-Means

Nouveau centroïde

```
for i in range(len(clusters[x])):
    d1 = 0
    d2 = 0
    for l in range(n):
        d1 += abs(clusters[x][i][l] - moy[l])
        d2 += abs(clusters[x][proche][l] - moy[l])
        if d1 < d2:
            proche = i
    clusters[x][proche], centroides[x] = centroides[x], clusters[x][proche]
```

Description

K-Means

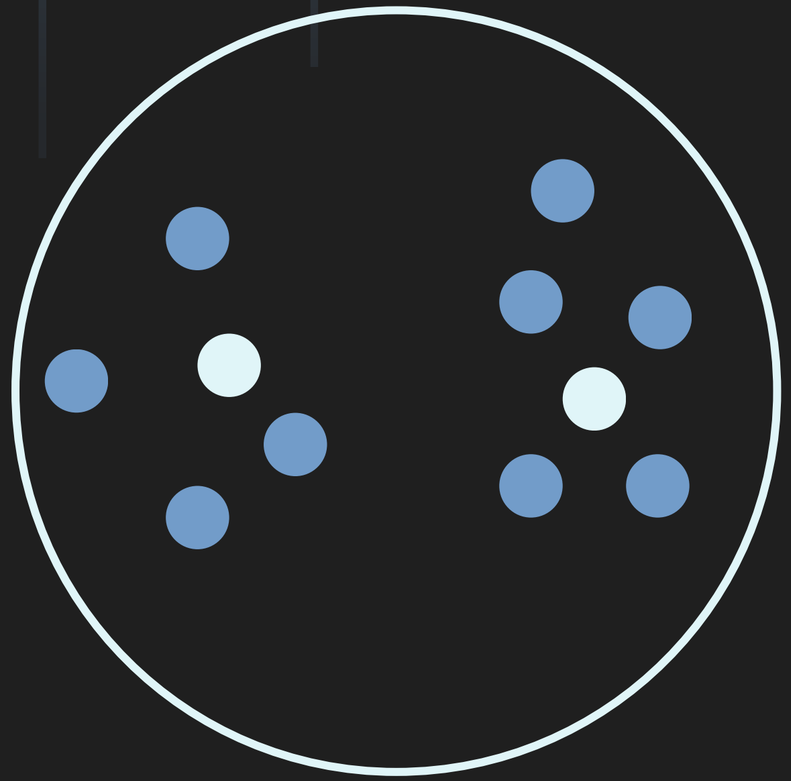


Nouveau cluster

```
clusters = doClusters(lst, centroides, k)
```

Description

K-Means



Fin de programme :

```
nb_iteration += 1
stop = convergence >= 3 or nb_iteration > 100000
```

Limites et obstacles

KNN

Termes techniques

Recherches en ligne

Intégrer au code

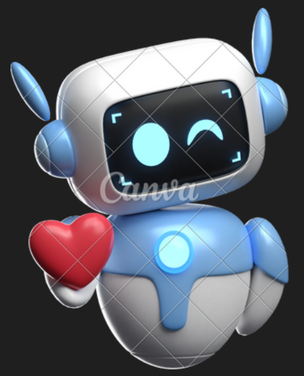
K-Means

Compréhension de l'algorithme

Reproduction en python

Termes techniques

Adaptable au nombre de dimensions et de points



Général

Connaissance en python faible

Beaucoup de recherches

Consignes peu claires

Beaucoup de questions a l'enseignant

Merci de votre attention

Regnier Mathilde - Le Beaudour Bastien - Fourre Mael