M2 Data Science – M2 Intelligence Artificielle

Data Mining

Ordonner et regrouper
ses données

Problème – comment comparer ?

# ID	Name	Type 1	Type 2	# Total	# HP	# Attack	# Defense	# SP Atk	# SP Def	# Speed
1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45
2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60
3	Venusaur	Grass	Poison	525	80	82	83	100	100	80
3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80
4	Charmander	Fire		309	39	52	43	60	50	65
5	Charmeleon	Fire		405	58	64	58	80	65	80
6	Charizard	Fire	Flying	534	78	84	78	109	85	100
6	CharizardMega Charizard X	Fire	Dragon	634	78	130	111	130	85	100
6	CharizardMega Charizard Y	Fire	Flying	634	78	104	78	159	115	100
7	Squirtle	Water		314	44	48	65	50	64	43
8	Wartortle	Water		405	59	63	80	65	80	58
9	Blastoise	Water		530	79	83	100	85	105	78
9	BlastoiseMega Blastoise	Water		630	79	103	120	135	115	78
10	Caterpie	Bug		195	45	30	35	20	20	45
11	Metapod	Bug		205	50	20	55	25	25	30
12	Butterfree	Bug	Flying	395	60	45	50	90	80	70
13	Weedle	Bug	Poison	195	40	35	30	20	20	50
14	Kakuna	Bug	Poison	205	45	25	50	25	25	35
15	Beedrill	Bug	Poison	395	65	90	40	45	80	75

Comment gère-t-on les données nominales non ordonnées ?

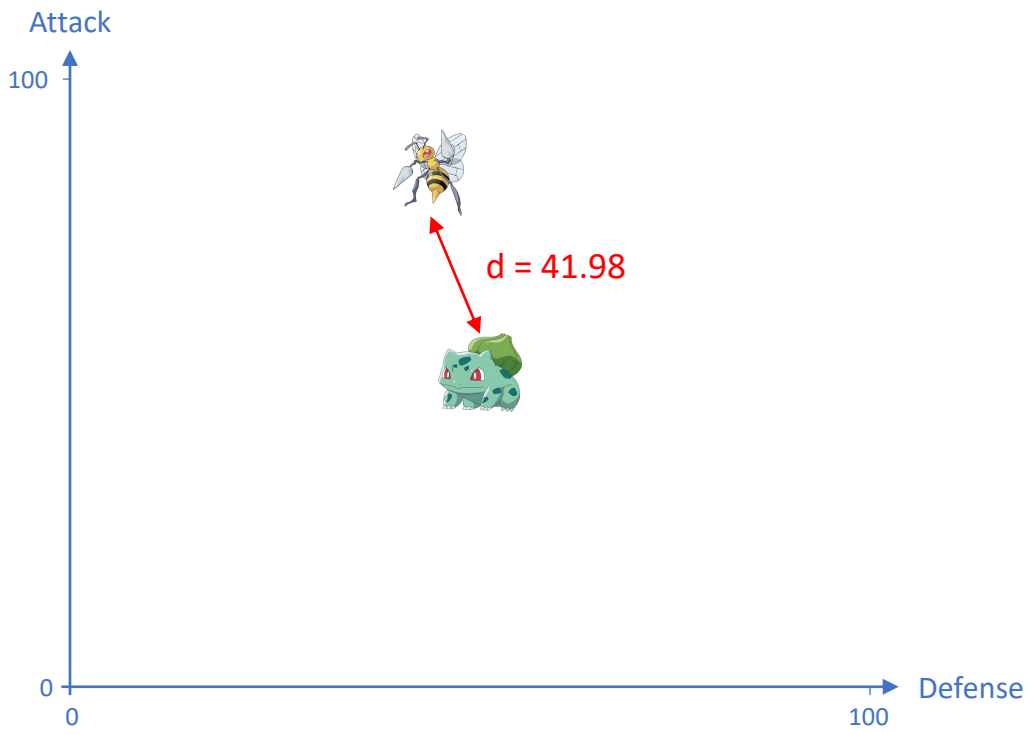
« Proches » ou « éloignés » ?

Comparer ses données

Place à la pratique !

- Sur kaggle, téléchargez l'un des jeux de données suivants:
  1. The Complete Pokemon Dataset
  2. Used Cars Dataset
  3. League of Legends champions
  4. NBA Player Statistics
  5. Magic: The Gathering
  6. Countries of the World
  7. Digimon Database
- Vérifiez et nettoyez vos données !
- Discutez sur comment comparer vos données entre elles
- Faites des tests et discutez de vos résultats

Définir une métrique de (dis)similarité
Distance euclidienne



Limites:

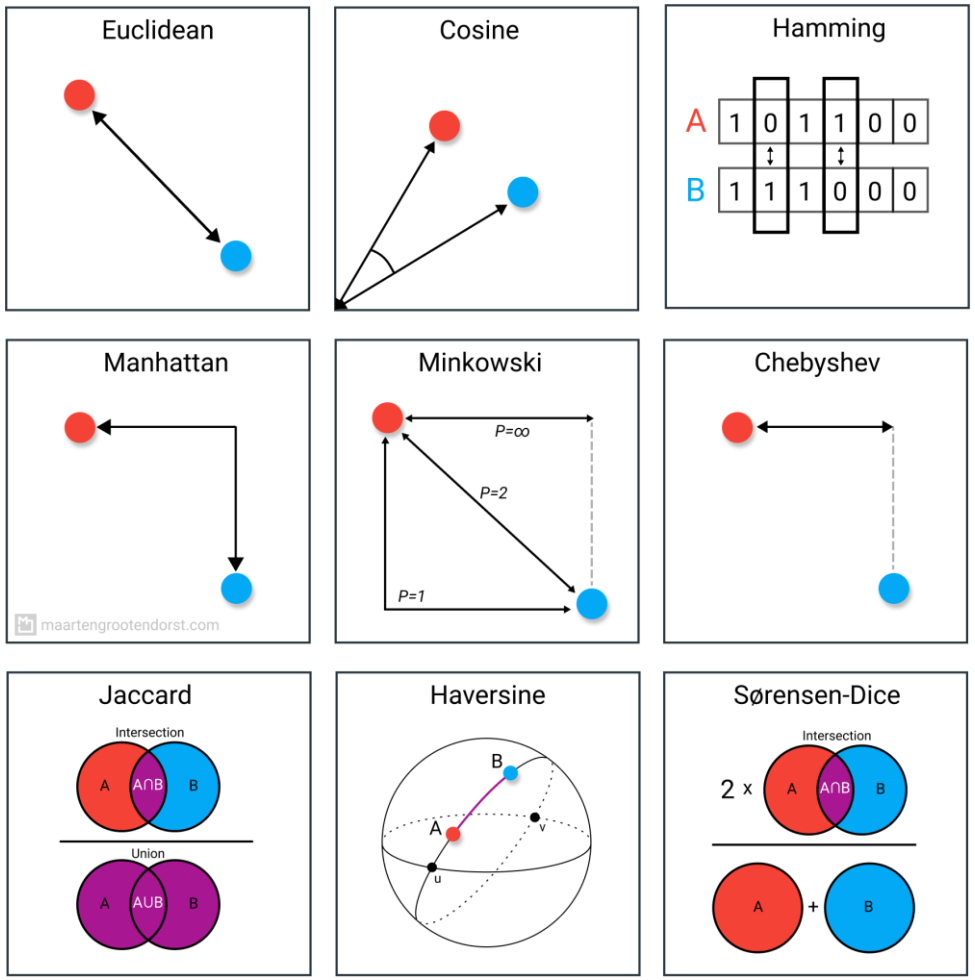
- Peu efficace sur plus de 3-4 dimensions
- Ne fonctionne que sur des données numériques

Bonnes pratiques:

1. Bien identifier les variables qui vont servir au calcul de similarité
2. Évaluer si une distance euclidienne est adaptée

=> Il y a aussi d'autres métriques possibles

Définir une métrique de (dis)similarité
Autres métriques

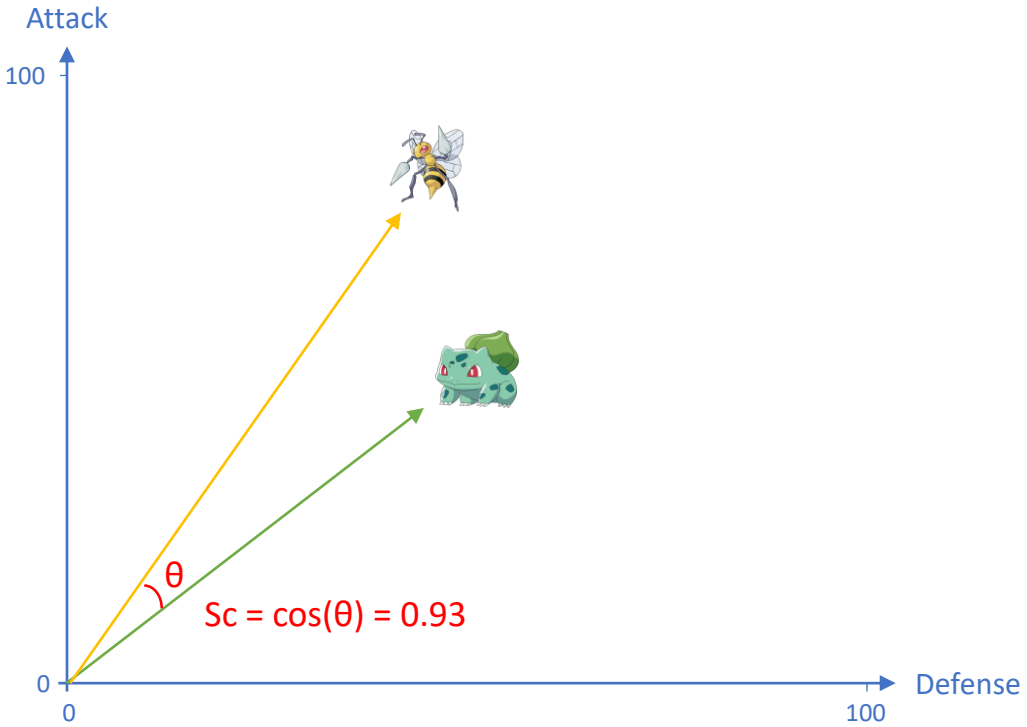


=> Chacune a ses avantages et ses inconvénients

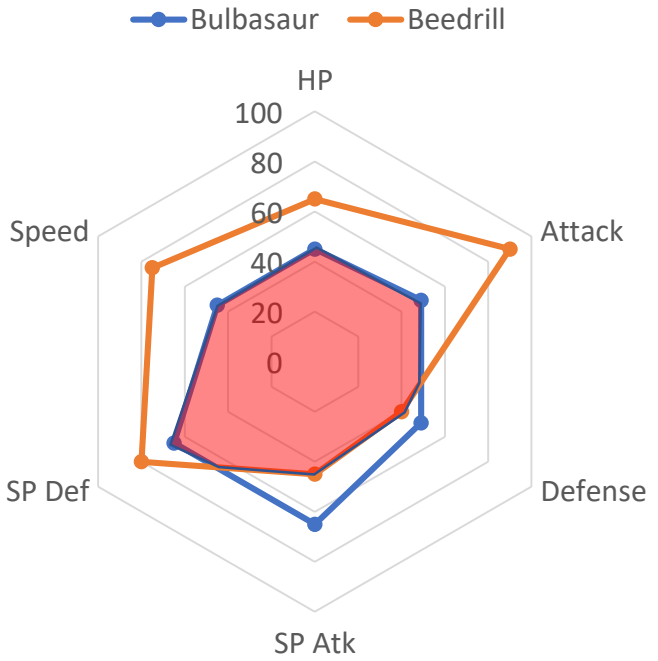
⇒ À vous d'identifier la métrique la plus adaptée et de justifier votre choix

Définir une métrique de (dis)similarité
Métriques pour données en grandes dimensions

Similarité cosinusoïdale

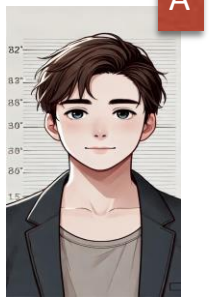


Recouvrement
Jacquart ou Sorensen-Dice



Définir une métrique de (dis)similarité
Attention aux échelles de vos variables

A



Age: 20 year
Taille: 1.82 m
Poids: 80 000 g

B

Age: 20 year
Taille: 1.82 m
Poids: 81 000 g

C

Age: 90 year
Taille: 1.50 m
Poids: 80 020 g

Distances euclidiennes:

- D(A, B) = 100.005
- D(A, C) = 72.8

Similarités cosinusoïdales:

- Sc(A, B) = 1
- Sc(A, C) = 1

=> La variable « poids », de part son échelle, impacte beaucoup plus le calcul de distance que les autres variables

Définir une métrique de (dis)similarité
Gérer les différences d'échelles

Normaliser :

$x' = \frac{x - min(x)}{max(x) - min(x)}$: [0..1]

Mean normalisation:

$x' = \frac{x - average(x)}{max(x) - min(x)}$: 0 = moyenne

Standardiser:

$x' = \frac{x - \overline{x}}{\sigma}$: 0 = moyenne, -1/+1 écart type

Attention ! Tout ne peux pas être normalisé ou standardisé !

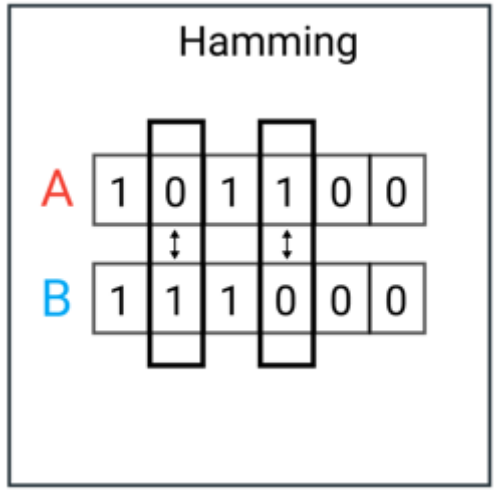
- Variables avec une « information absolue »
  - Nombre d'observation, durée, positions, etc.
- Pourcentages, notes, scores entre 0 et 1
- Variables binaire et nominales

$\Rightarrow$ À vous d'identifier l'approche la plus adaptée et de justifier votre choix

Définir une métrique de (dis)similarité
Gérer les variables nominales

Bonnes pratiques:

- Si variable nominale ordonnée:
  - Peut être gérée comme une variable numérique
- Si variable nominale non-ordonnée:
  - One-hot (ou dummy) encoding
  - Utilisez une distance de hamming

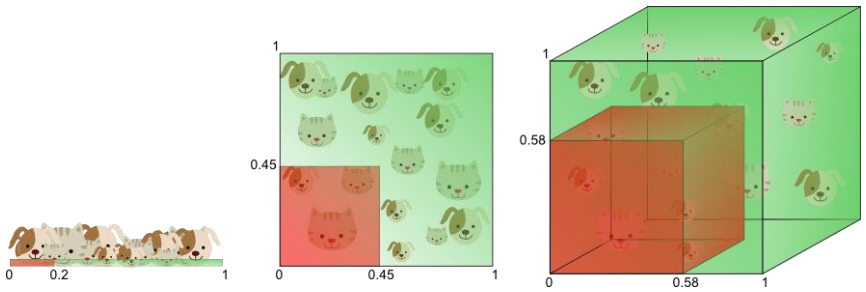


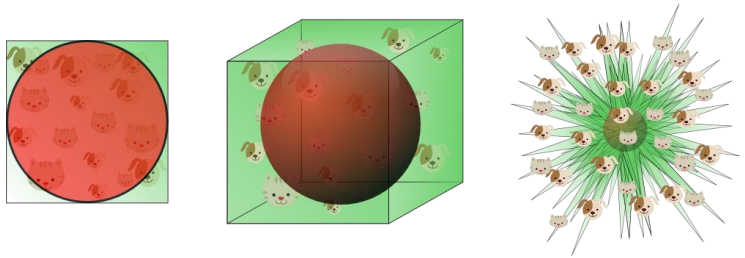
⇒ Il est parfois nécessaire de mélanger les approches

⇒ Il est possible de donner plus de poids à certaines variables

⇒ À vous d'identifier l'approche la plus adaptée et de justifier votre choix

Définir une métrique de (dis)similarité
La malédiction des grandes dimensions





Plus vous avez de dimensions, moins les données vont sembler similaires.

Supposons que vous cherchez les données avec 80% des valeurs les plus fréquente pour chaque variable:

- 1 variable: couvre 80% des données
- 2 variables: couvre 64% des données
- 3 variables: 51%
- …
- 10 variables: 10%
- …
- 100 variables: 0.00000002%

⇒ Parfois, il peut être plus adaptée de réduire le nombre de dimension plutôt que de vouloir tout prendre en compte

Réduire le nombre de dimensions de ses données (ordonnancement)

Place à la pratique !

- Reprenez le jeu de données téléchargé précédemment :
  1. The Complete Pokemon Dataset
  2. Used Cars Dataset
  3. League of Legends champions
  4. NBA Player Statistics
  5. Magic: The Gathering
  6. Countries of the World
  7. Digimon Database

- Discutez de l'intérêt et de l'importance de chaque variables
  - Appliquez normalisation et standardisation si vous pensez cela pertinent

- Discutez sur comment réduire le nombre de dimensions de vos données
  - Et sur comment les ordonner entre elles sur une ou deux dimensions

- Faites des tests et discutez de vos résultats

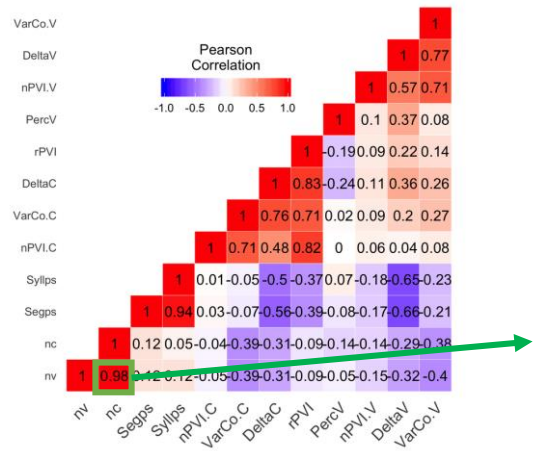
Réduire le nombre de dimensions
Gérer les variables fortement liées

Informations redondantes

ID	Nom	Prenom	Nom_Complet	Date_Naissance
006	Trevelyan	Alec	Alec Trevelyan	12/03/1928
007	Bond	James	James Bond	11/11/1920

Variables calculées

Taille	Poids	IMC
185	78	22.8
167	82	29.4

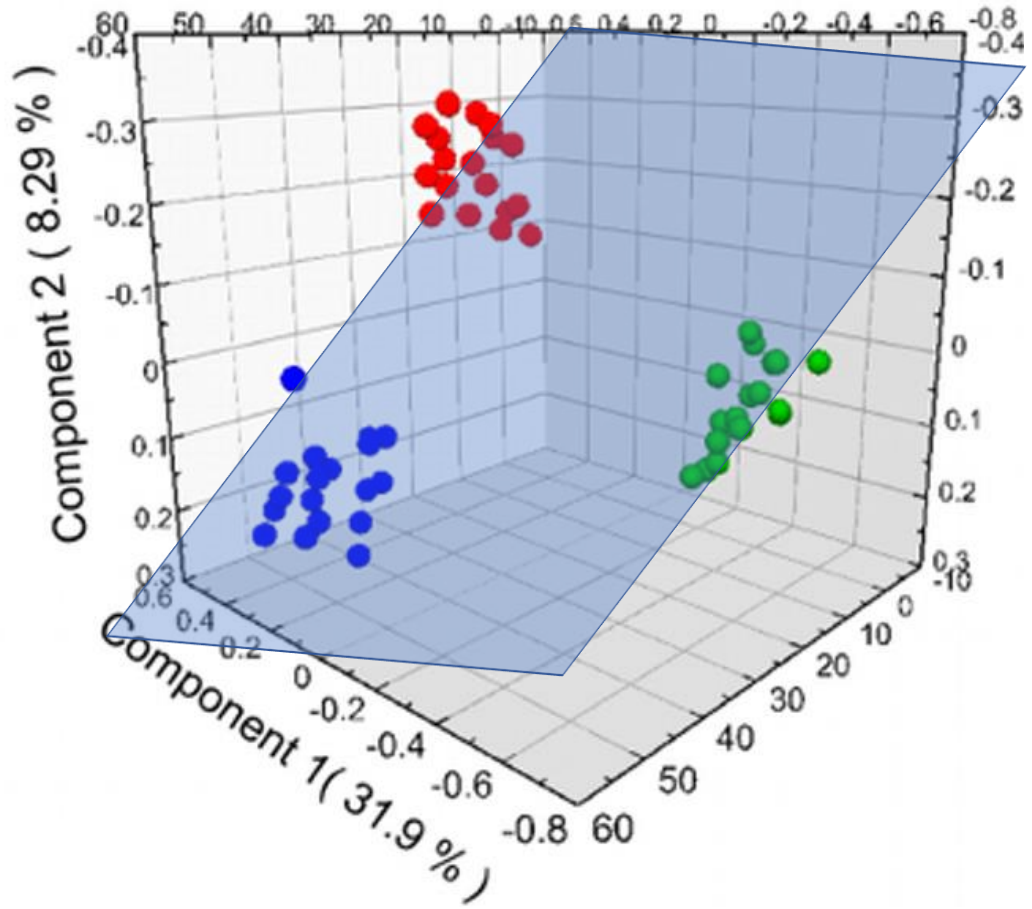


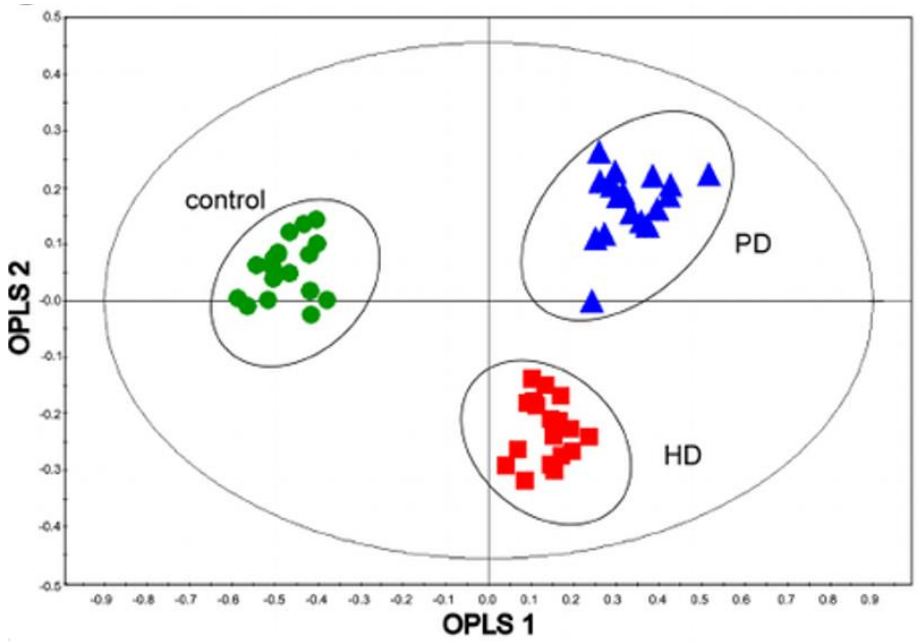
Très fortement corrélées
⇒ informations similaires ?

Bonnes pratiques (dans un objectif de réduction de dimensions):

- Gardez les variables en agrégeant d'autres
  - ex. Nom_Complet et IMC
- Vérifiez les variables fortement corrélées
  - Si informations contenues similaire, voir identique, gardez l'un des deux

Réduire le nombre de dimensions
Objectif: projeter sur un plan 2D





Réduire le nombre de dimensions
Pricipal Components Analysis (PCA)

Algorithme:

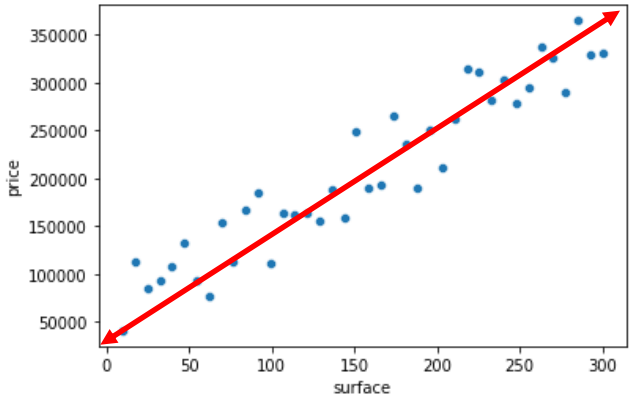
1. Trouver un « axe », qui minimise la distance entre les points et cet axe

2. Pour d dans [2:nb_dimensions]:
    1. Trouver un autre axe qui répond aux deux conditions suivantes:
        1. Orthogonal aux précédents axes
        2. Minimise la variance

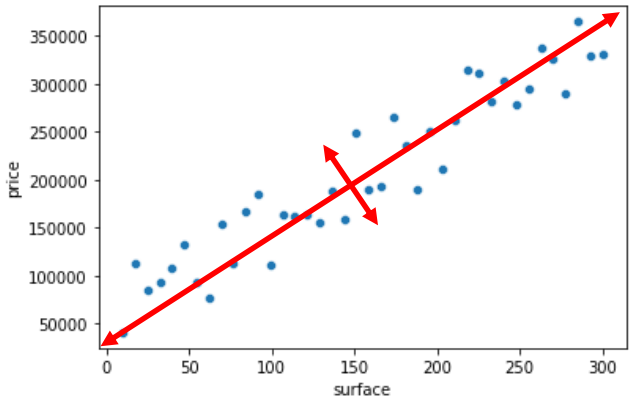
3. À la fin, garder les premières k dimensions

⇒ Simple à comprendre et à mettre en place
⇒ Mais, vite limité

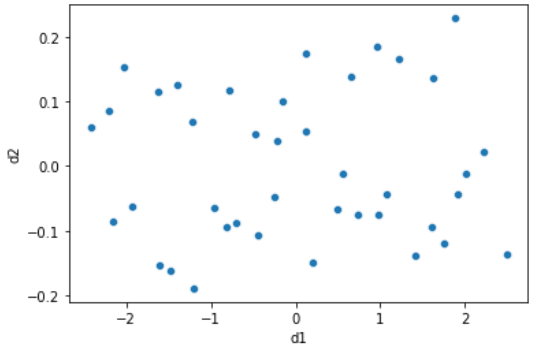
1



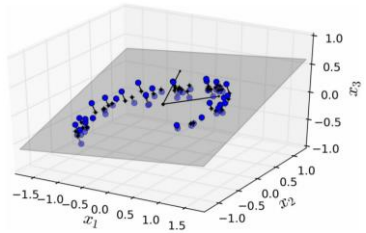
2



3



Exemple 3D => 2D



Réduire le nombre de dimensions
t-distributed Stochastic Neighbor Embedding (t-SNE)

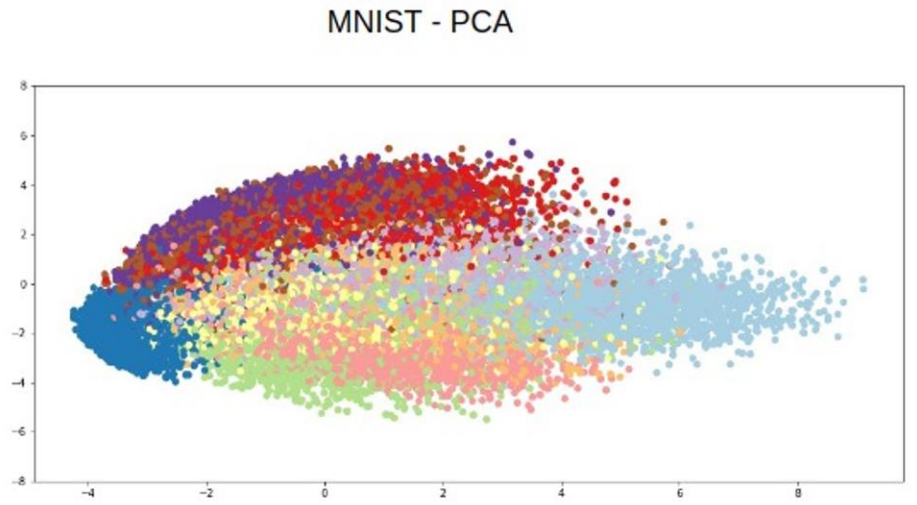
Principe général:

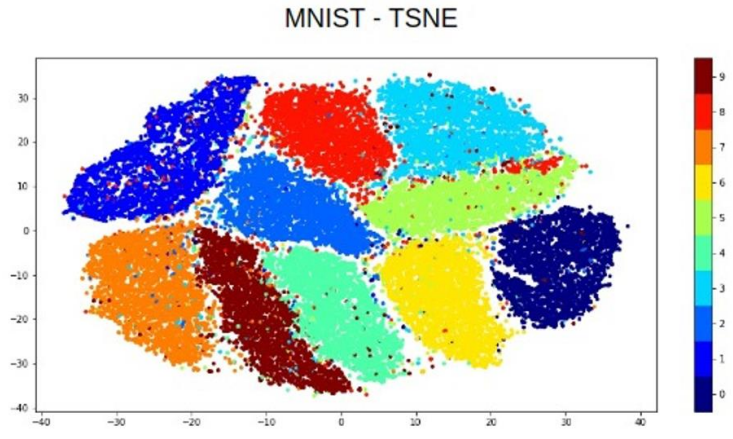
1. Définit une similarité $p_{j|i}$ dans l'espace source P
   - Basée sur une distribution normale

2. Définit une similarité $q_{j|i}$ dans l'espace objectif Q
   - Basée sur une t-distribution (a tendance à « exagérer » les différences

3. Pour chaque coordonnées x_i dans P, trouver une nouvelle coordonnée y_i dans Q telle qu'elle minimise la différence entre P et Q
   - $\forall_{i,j} p_{j|i} \approx q_{j|i}$

4. Se base sur un paramètre de « perplexité » σ pour gérer les distances / le voisinage
   - σ petit préserve les distances proches
   - σ grand donne plus d'importance aux longues distances (c'est aussi plus coûteux)





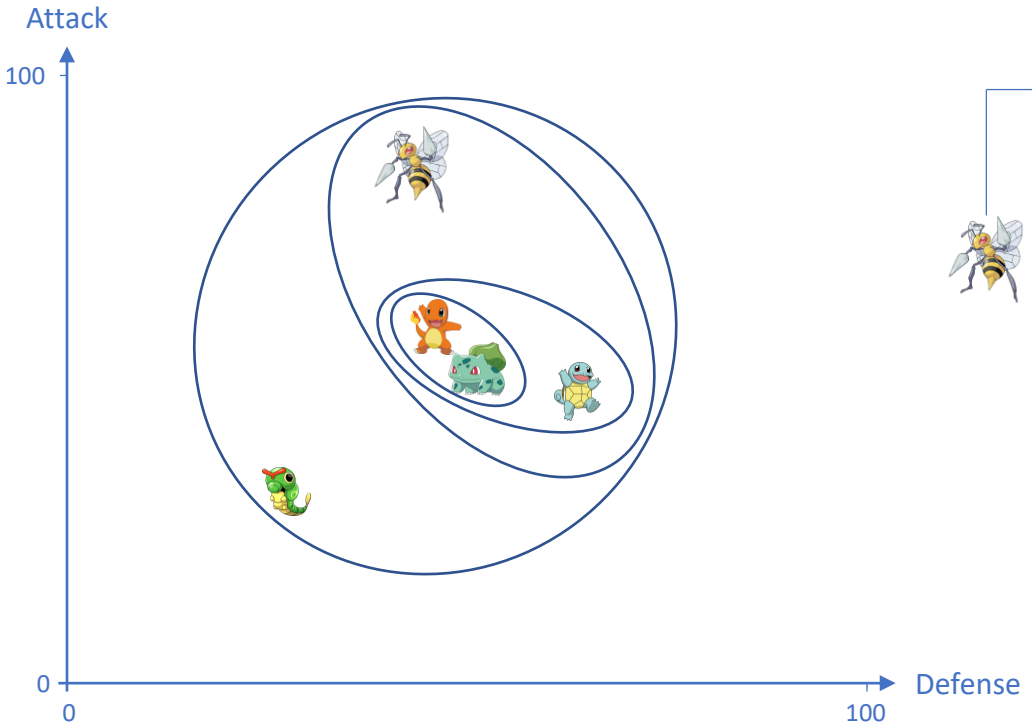
⇒ T-SNE est généralement préféré à PCA

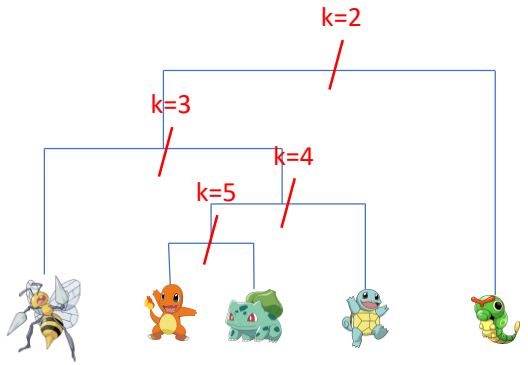
Regrouper ses données (clustering)

Place à la pratique !

- Reprenez le jeu de données téléchargé précédemment :
  1. The Complete Pokemon Dataset
  2. Used Cars Dataset
  3. League of Legends champions
  4. NBA Player Statistics
  5. Magic: The Gathering
  6. Countries of the World
  7. Digimon Database
- Testez les algorithmes PCA et t-SNE
- Discutez sur comment créer des groupes de données « similaires »
  - Vous pouvez vous baser sur la réduction de dimension ou non
- Faites des tests et discutez de vos résultats

Regrouper ses données
Approche agglomérative





Dendogramme

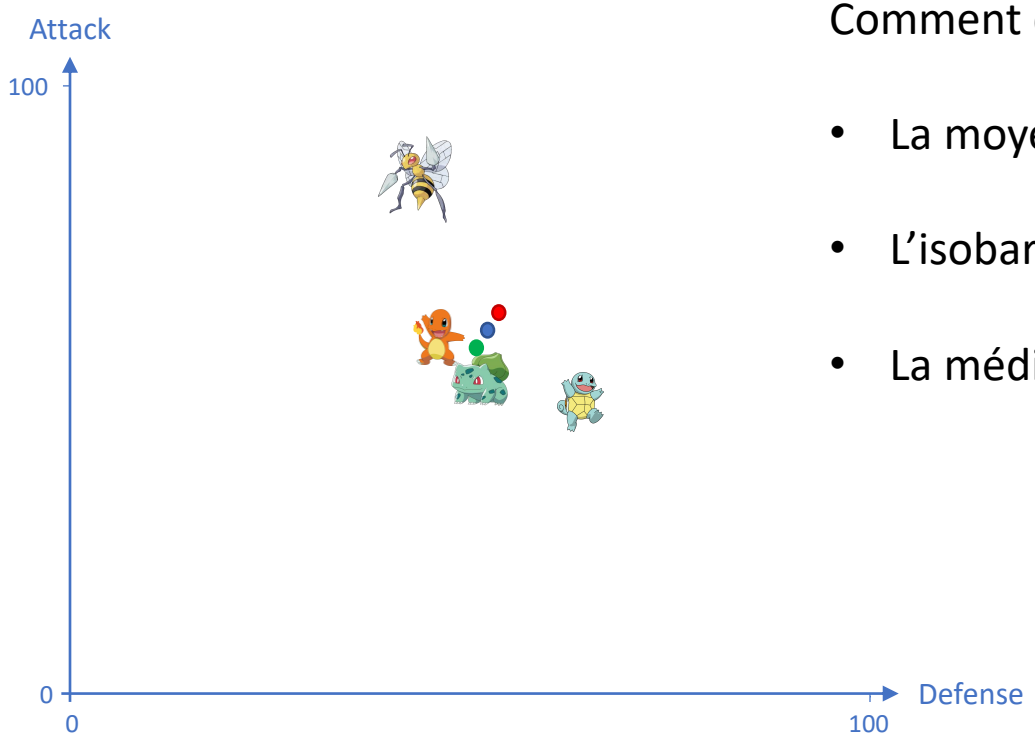
Pour créer *k* groupes:

- On coupe les branches en partant du haut
- On s'arrête quand on a le nombre de groupes souhaité

⇒ Simple à comprendre et à mettre en place
⇒ Mais, très coûteux et vite limité

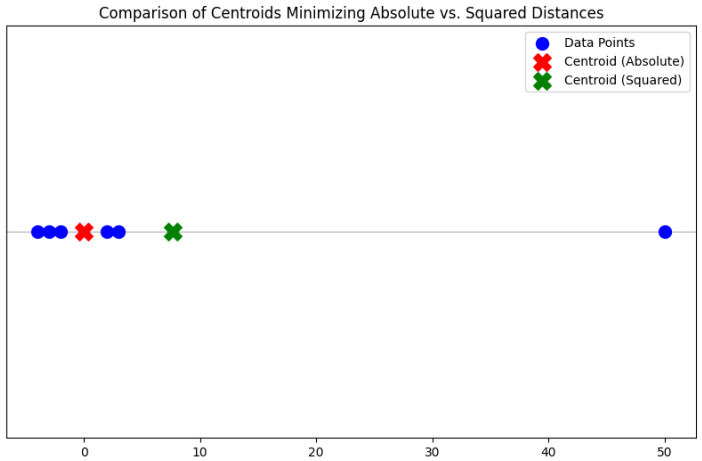
⇒ Nécessite de bien définir:
⇒ La métrique de (dis)similarité
⇒ La méthode d'aggrégation

Regrouper ses données
Méthodes d'aggrégation



Comment définir le « centre » d'un groupe ?

- La moyenne des positions ?
- L'isobarycentre ?
- La médiane des positions ?

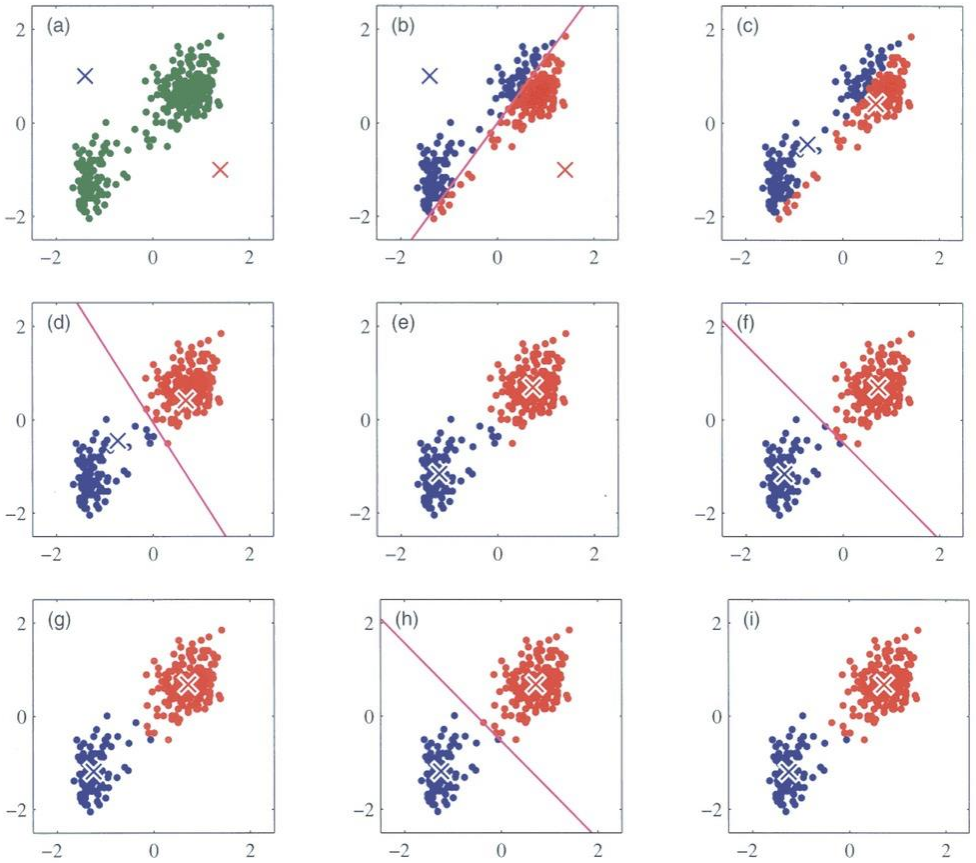


⇒ La moyenne minimise la distance quadratique (squared distance)

⇒ La médiane minimise la distance absolue

⇒ À vous d'identifier l'approche la plus adaptée et de justifier votre choix

Regrouper ses données
K-means (et K-medoids)



Algorithme K-means:

1. Initialisé *k* groupes avec 1 élément aléatoire dans chaque qui serait le centroid initial

2. Pour chaque donnée *d* non groupée:
   1. Calculer la distance entre *d* et chaque centroid
   2. Affecter *d* au groupe avec le centroid le plus proche
   3. Recalculer la position du centroid pour ce groupe (moyenne de la position de tous les éléments du groupe)

Si l'on utilise la médiane au lieu de la moyenne, on parle de medoids

⇒ Simple à comprendre et à mettre en place
⇒ Mais:
  ⇒ Nécessite de présupposer *k* en avance
  ⇒ Très sensible à l'initialisation, ainsi qu'aux calculs des distances et d'aggrégation
⇒ Version améliorée: k-means++

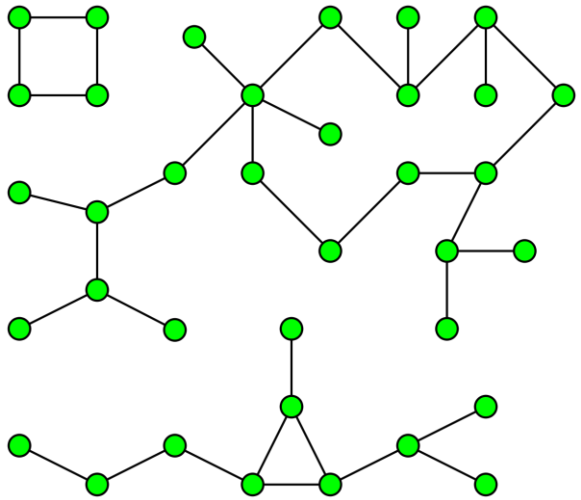
Regrouper ses données
DB-SCAN

Algorithme:

1. Construire un graphe tel que:
   - Chaque coordonnée est un nœud
   - Un lien existe entre deux nœuds si leur distance est inférieur à ϵ

2. Détecter les sous-graphes connectés
   - 2 nœuds appartiennent au même groupe si il existe un chemin entre eux

3. Pour chaque nœud non relié
   - les considérés comme du bruit



⇒ Fourni, généralement, de meilleur résultats que k-means

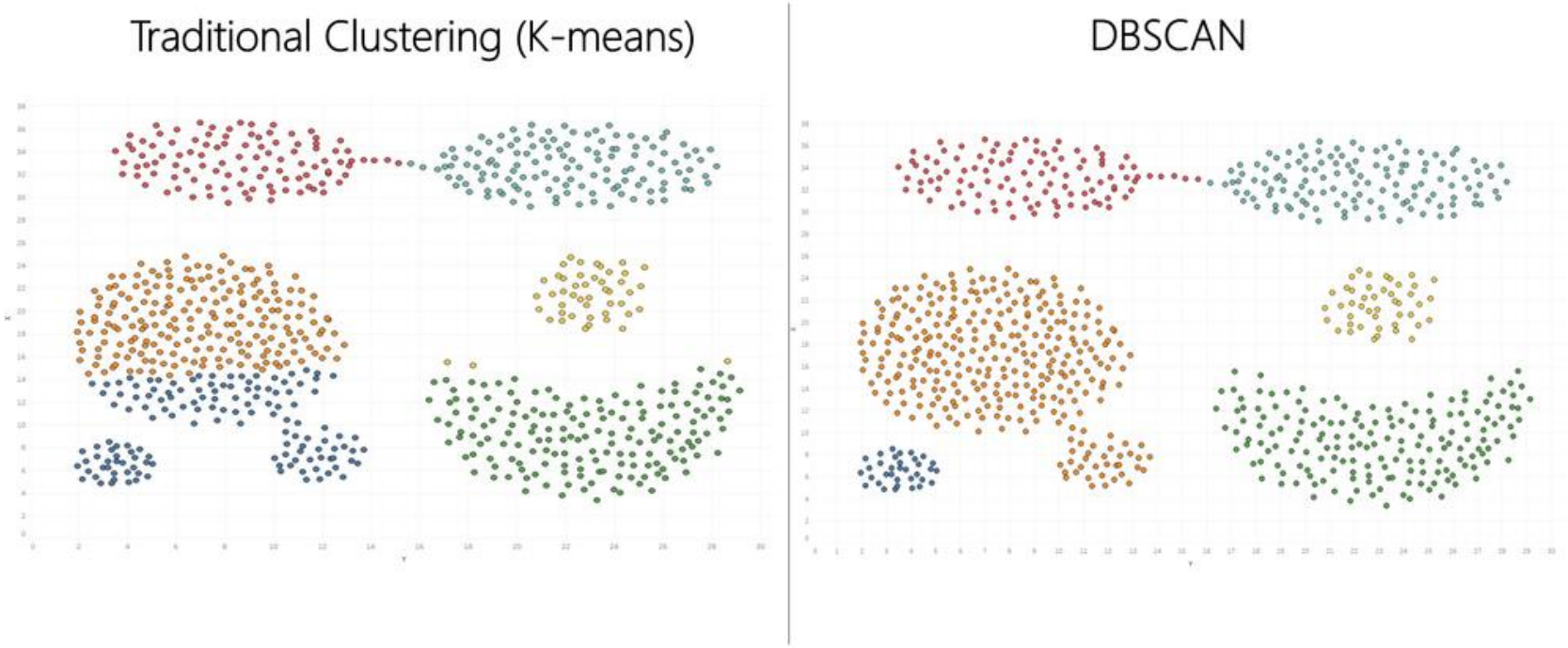
⇒ Mais,

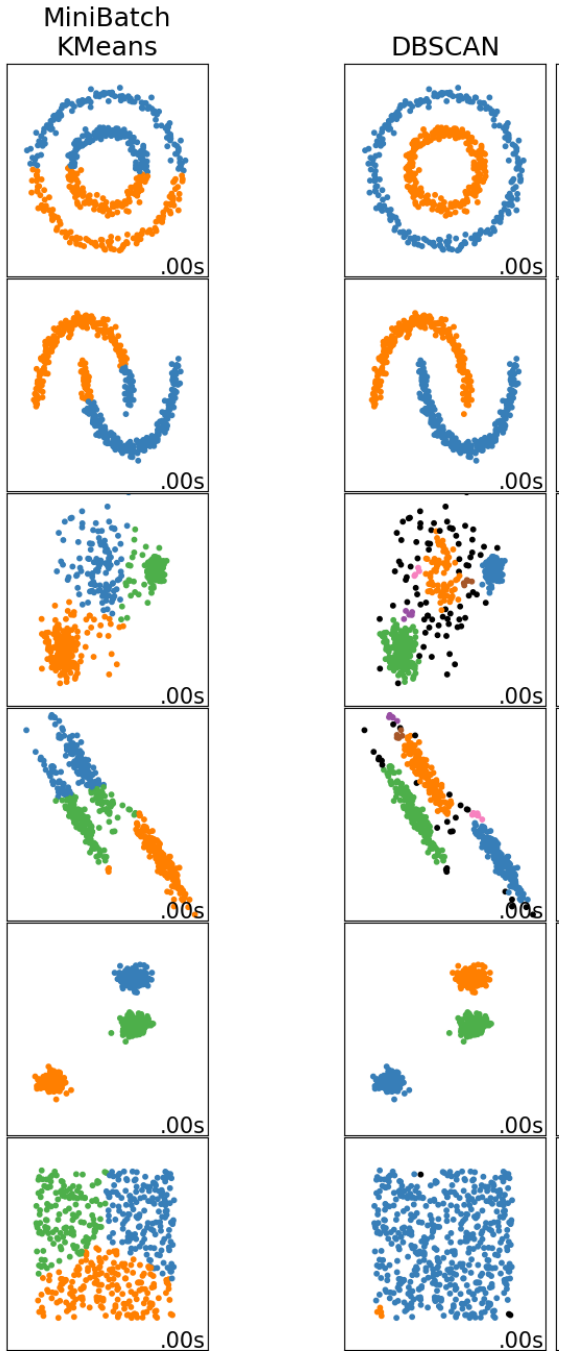
⇒ ⇒ Plus coûteux

⇒ ⇒ Ne groupe pas toutes les données entre elles (bruit)

⇒ Néanmoins, peut être utile pour détecter des anomalies (Cours suivant)

Regrouper ses données
Comparaison K-Means et DB-SCAN





Place à l'action !

Votre Mission:

- Reprendre votre code là vous l'aviez laissé
  - Lien initial: https://github.com/a-t-richard/20262027-UE-DataMining-Project
- Utiliser les nouvelles notions et méthodes vues pendant ce cours
- Trouver ce qu'il se cache dans ces données !