DISTANCES ET CLUSTERING

DISTANCE

Une distance est une fonction qui calcule la longueur qui sépare deux points.
Sans faire de maths, on connaît de nombreuses manières de calculer des distances:

- Distance entre 2 villes (nombre de km via une route terrestre, navale ou aérienne)
- Distance à vol d'oiseau.
- Distance sur un clavier (nombre de touches séparant deux lettres)
- Distance sur un arbre généalogique (nombre de générations, principe du frère, du cousin, ...)
- Distance entre amis (ami proche, connaissance, collègue...)

GROUPES / CLUSTERS

Grâce aux distances, on forme intuitivement des **groupes** ou des **clusters**:

- Des clusters d'amis
- Des clusters d'images
- Des clusters de pays
- Des clusters de mots
- Des clusters d'animaux...

Oui, mais: sur quelle base? Selon ce qu'on regarde et ce qui nous semble important, les clusters ne sont pas les mêmes.

CLUSTERING

Clustering :

À partir d'un jeu de données, faire du clustering revient à séparer les données en clusters de telle manière qu'au sein d'un cluster,
les données se ressemblent (distance faible) et qu'entre les clusters, les données ne se ressemblent pas (distance élevée).

Exemple:

Proposons des manières de clusteriser les personnes de la classe.

CLUSTERING

Clustering :

À partir d'un jeu de données, faire du clustering revient à séparer les données en clusters de telle manière qu'au sein d'un cluster,
les données se ressemblent (distance faible) et qu'entre les clusters, les données ne se ressemblent pas (distance élevée).

Exemple:

Proposons des manières de clusteriser les personnes de la classe.

- Par sexe?
- Par couleur de cheveux?
- Par âge?
- Par couleur de cheveux et d'yeux?
- Tout ceci en même temps?

CONCLUSIONS ANTICIPÉES

Intuitivement on remarque que cela dépend au moins :

- du nombre de clusters que l'on veut
- des attributs (variables) que l'on regarde
- de la manière de calculer la distance

Il n'y a donc pas de clustering unique.

OBJECTIFS DU CLUSTERING

Un algorithme de clustering est une méthode qui sépare les points en groupes en minimisant la distance au sein des groupes et en maximisant la distance entre les groupes.

De nombreuses méthodes existent : on ne peut pas dire avec certitude que l'une est meilleure que l'autre. Cela dépend des données, de la distance, etc.

POUR VOS PROJETS

- Distance entre ebooks
- Distance entre équipes de foot
- Distance entre photos
- Distance entre amis
- Distance entre pays
- Distance entre appartements
- Distance entre tweets
- Distance entre produits
- Distance entre parcours de course
- Distance entre films

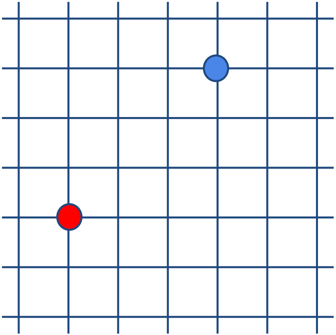
DISTANCES

PROPRIÉTÉS DES DISTANCES

Pour être mathématiquement une distance, une fonction doit vérifier les trois propriétés suivantes:

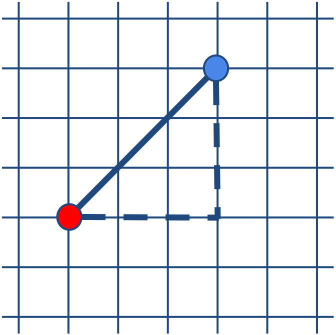
- Symétrie: d(a, b) = d(b, a)
- Inégalité triangulaire: d(a, c) <= d(a, b) + d(b, c)
- Séparation: d(a, b) = 0 équivaut à a = b

DISTANCE EUCLIDIENNE



Calculée grâce au théorème de Pythagore, le plus court chemin entre deux vecteurs.

DISTANCE EUCLIDIENNE



Calculée grâce au théorème de Pythagore, le plus court chemin entre deux vecteurs.

$$\begin{aligned} D(X,Y) &= \sqrt{(X_1 - Y_1)^2 + (X_2 - Y_2)^2 + \cdots + (X_n - Y_n)^2} \\ &= \sqrt{\sum_{i=1}^{n} (X_i - Y_i)^2} \end{aligned}$$

EXEMPLE AVEC DU TEXTE

	voici	un	premier	texte	second	ce	document	contient
d1	0.1	0.1	0.275	0	0	0	0	0
d2	0.1	0.1	0	0	0.275	0	0	0
d3	0	0	0	0	0	0.44	0.22	0.22

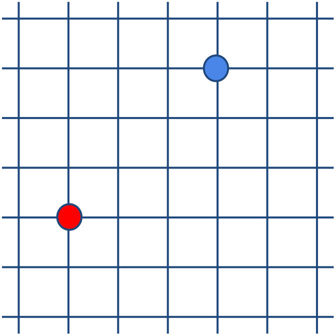
$$D(d_1, d_2) = \sqrt{0^2 + 0^2 + 0,275^2 + 0^2 + 0,275^2 + 0^2 + 0^2 + 0^2} = 0.39$$

$$D(d_1, d_3) = \sqrt{0.1^2 + 0.1^2 + 0,275^2 + 0^2 + 0^2 + 0.44^2 + 0.22^2 + 0.22^2} = 0.62$$

$$D(d_2, d_3) = \sqrt{0.1^2 + 0.1^2 + 0^2 + 0^2 + 0.275^2 + 0.44^2 + 0.22^2 + 0.22^2} = 0.62$$

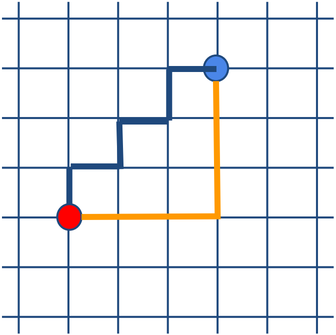
Les deux premiers documents sont proches entre eux et loin du troisième.

DISTANCE DE MANHATTAN



Vous êtes obligés d'utiliser les rues et les avenues pour vous déplacer dans Manhattan.

DISTANCE DE MANHATTAN



Vous êtes obligés d'utiliser les rues et les avenues pour vous déplacer dans Manhattan. Tous les chemins sont équivalents si vous ne faites aucun détour.

$$
\begin{aligned}
D(X,Y) &= |X_1 - Y_1| + |X_2 - Y_2| + \cdots + |X_n - Y_n| \\
&= \sum_{i=1}^{n} |X_i - Y_i|
\end{aligned}
$$

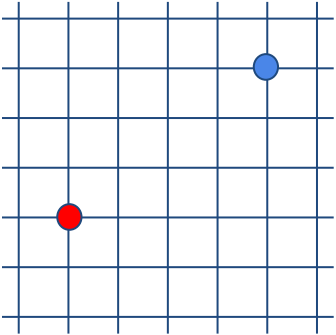
EXEMPLE AVEC DU TEXTE

	voici	un	premier	texte	second	ce	document	contient
d1	0.1	0.1	0.275	0	0	0	0	0
d2	0.1	0.1	0	0	0.275	0	0	0
d3	0	0	0	0	0	0.44	0.22	0.22

$$
\begin{aligned}
D(d_1, d_2) &= |0| + |0| + |0.275| + |0| + |0.275| + |0| + |0| + |0| = 0.555 \\
D(d_1, d_3) &= |0.1| + |0.1| + |0.275| + |0| + |0| + |0.44| + |0.22| + |0.22| = 1.365 \\
D(d_2, d_3) &= |0.1| + |0.1| + |0| + |0| + |0.275| + |0.44| + |0.22| + |0.22| = 1.365
\end{aligned}
$$

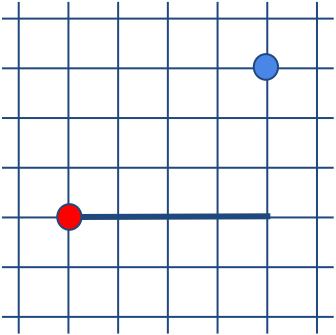
Les deux premiers documents sont toujours proches entre eux et loin du troisième mais la différence est plus accentuée car les valeurs sont < 1.

DISTANCE DE TCHEBYCHEV



On prend la pire des différences sur toutes les variables.

DISTANCE DE TCHEBYCHEV



On prend la pire des différences sur toutes les variables.

$$
\begin{aligned}
D(X,Y) &= \max(|X_1 - Y_1|, |X_2 - Y_2|, \ldots, |X_n - Y_n|) \\
&= \max_{i=1\ldots n} |X_i - Y_i|
\end{aligned}
$$

EXEMPLE AVEC DU TEXTE

	voici	un	premier	texte	second	ce	document	contient
d1	0.1	0.1	0.275	0	0	0	0	0
d2	0.1	0.1	0	0	0.275	0	0	0
d3	0	0	0	0	0	0.44	0.22	0.22

$$
\begin{aligned}
D(d_1, d_2) &= \quad |0.275 - 0| = 0.275 \\
D(d_1, d_3) &= \quad |0 - 0.44| = 0.44 \\
D(d_2, d_3) &= \quad |0 - 0.44| = 0.44
\end{aligned}
$$

Cette distance reflète le poids du mot le plus différent pour chaque paire de documents.

DISTANCE DE HAMMING

$$D(x,y) = \sum_i \delta(x_i = y_i)$$

	voici	un	premier	texte	second	ce	document	contient
d1	0.1	0.1	0.275	0	0	0	0	0
d2	0.1	0.1	0	0	0.275	0	0	0
d3	0	0	0	0	0	0.44	0.22	0.22

$$D(d_1, d_2) = \quad 0+0+1+0+1+0+0+0 = 2$$
$$D(d_1, d_3) = \quad 1+1+1+0+0+1+1+1 = 6$$
$$D(d_2, d_3) = \quad 1+1+0+0+1+1+1+1 = 6$$

Distance plus grossière. Plus appropriée pour comparer deux mots du même nombre de lettres (ou deux phrases du même nombre de mots).

DISTANCE ENTRE DES PERSONNES ?

Anne-Claire Haury


```json
{
  "id": "610174245",
  "name": "AC Haury",
  "birthday": "09/06/1984",
  "hometown": {
    "id": "110774245616525",
    "name": "Paris, France"
  },
  "location": {
    "id": "110774245616525",
    "name": "Paris, France"
  },
  "gender": "female",
  "timezone": 2,
  "locale": "en_US",
  "first_name": "AC",
  "last_name": "Haury"
}
```


Paris Hilton


```json
{
  "name": "Paris Hilton",
  "about": "Official Facebook Page for Paris Hilton",
  "id": "113208373525",
  "birthday": "02/17/1981",
  "location": {
    "city": "Beverly Hills",
    "country": "United States",
    "state": "CA",
    "zip": "90210"
  },
  "website": "www.ParisHilton.com",
  "category": "Public Figure",
  "bio": "Twitter - @ParisHilton
Instagram - @ParisHilton
Snapchat - ParisHilton",
  "talking_about_count": 50900,
  "rating_count": 0
}
```


Objectif : calculer la distance entre Paris Hilton et moi...

DISTANCE ENTRE DES PERSONNES ?

Que prendre en compte? Quels sont les attributs qui ont un sens pour ce problème?

- Localisation géographique ?
- Âge ?
- Nombre d’amis en commun ?
- Sexe ?
- Likes en commun ?
- Prénom ?

DISTANCE ENTRE DES PERSONNES ?

Puisque les données sont de types différents, il va falloir les regarder séparément. On établit donc une distance entre, par exemple :

- Les localisations (facile avec les coordonnées géographiques)
- Les likes : une fonction qui dépend des likes en commun.
- Les amis : une fonction qui dépend du nombre d'amis en commun.
- etc.

Puis on les agrège:

Distance totale = distance_géo + distance_amis + distance_likes + ...

On peut mettre des poids:

Distance totale = $\alpha_1 \times$ **distance géo +** $\alpha_2 \times$ **distance amis +**$\alpha_3 \times$**distance likes + ...**

PARTI PRIS

Il y a donc un parti à prendre sur :

- ce qu'on regarde
- quelle(s) distance(s) on choisit
- comment on les agrège

Tous les choix sont ok, il faut être capable de les justifier en fonction de votre application.

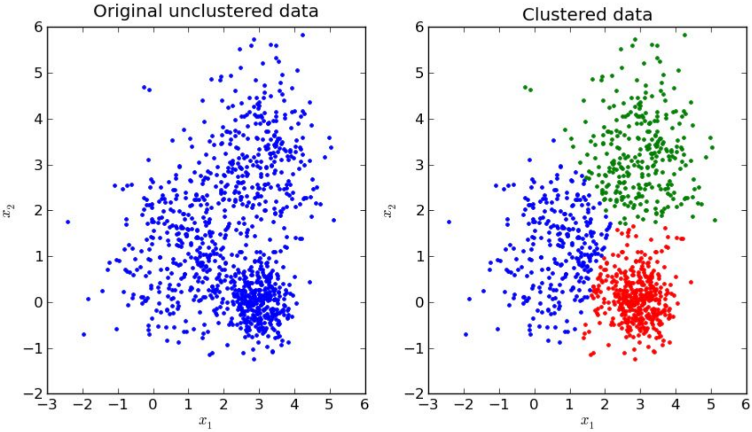
LA DISTANCE EST UN OUTIL

En soi, la distance n'est qu'un outil pour arriver à une fin, par exemple clustering ou recommandation.

- Clustering : la distance permet de grouper les données proches.
- Recommandation : la distance permet de proposer à des personnes proches des objets proches.

CLUSTERING

PRINCIPE



Source: stackoverflow.com

ALGORITHMES

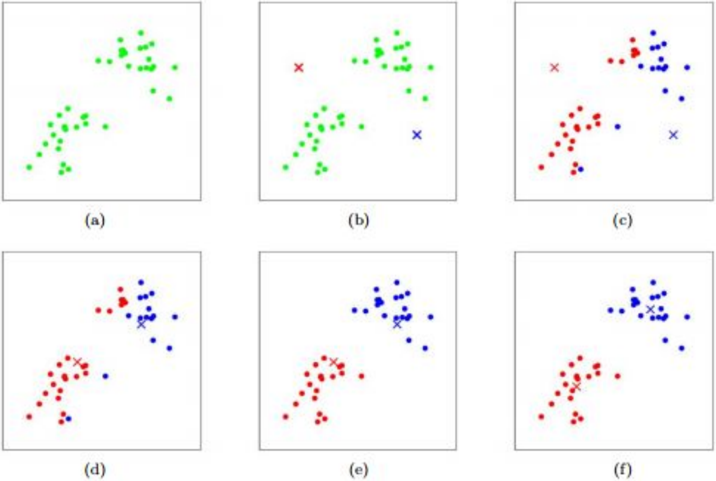
Toutes les méthodes reposent sur des distances.

Nous verrons ici:

- Une méthode probabiliste : K-Means
- Une méthode algorithmique : CAH (classification ascendante hiérarchique)

K-MEANS

FONCTIONNEMENT



Source : stanford.edu

NOTATIONS

- n points à clusteriser
- K clusters finaux

ALGORITHME

Cet algorithme **itératif** prend en entrée les données et le nombre de clusters.

K-Means

Initialisation : m_1, m_2, ...m_K : K points choisis au hasard parmi tous les points
while Les points changent de cluster **do**
    for Chaque point **do**
        Trouver le cluster dont le centre m_j est le plus proche du point;
        Assigner le point au cluster en question;
    end for
    Recalculer la moyenne m_j de chaque cluster;
end while

CENTROÏDE LE PLUS PROCHE

- Cet algorithme part du principe que les clusters suivent des **lois normales** (cf cours précédent).
- Il utilise la **distance euclidienne** (distance associée à cette loi).
- L'étape d'**assignation** d'un point à un cluster consiste donc à calculer, pour chaque point x_i de taille k et pour chaque centre m_j :

$$D(x_i, m_j) = \sqrt{\sum_{k=1}^{K}(x_{i,k} - m_{j,k})^2}$$

- On cherche alors le cluster j∗ **le plus proche**:

$$j^* = \arg\min_j D(x_i, m_j)$$

- Le point i fait désormais partie du cluster j∗.

MISE À JOUR DES MOYENNES

Une fois que les points ont tous été assignés à un cluster, **on recalcule** pour tous les j **la moyenne** m_j du cluster C_j :

$$m_j^{(t+1)} = \frac{1}{n_j^{(t)}} \sum_{i : x_i \in C_j^{(t)}} x_i$$

ARRÊT

On arrête lorsque plus aucun point ne change de cluster.

EVALUATION

Comment décider si un clustering est meilleur qu'un autre ?

On mesure la distance totale qui est aussi la **variance totale**:

$$L = \frac{1}{n} \sum_{i=1}^{n} (x_i - m_{C(x_i)})^2$$

où $c(x_i)$ est le cluster final du point x_i.

Plus L est petit, plus le clustering est bon.

INITIALISATION

- Cet algorithme est très **sensible à l'initialisation** : deux initialisations différentes peuvent produire deux clusterings différents.
- La solution la plus courante consiste à le lancer **plusieurs fois**, on obtient donc potentiellement différents clusters. On choisit le **meilleur clustering** en termes de distance (moyenne des distances de chaque point à son centre).

NOMBRE DE CLUSTERS K

- Dans la plupart des cas, on ne le connaît pas.
- Une solution consiste à essayer différentes valeur pour K . On obtient donc différents clusterings. On choisit le meilleur: celui qui minimise la moyenne des distances entre chaque point et son centre.

UTILISATION EN PYTHON

```
from sklearn.cluster import KMeans
model  =  KMeans(n_clusters  =  5)
model.fit(X)
y = model.labels_
```

Autres arguments intéressants:

- n_jobs: pour paralléliser
- n_init: nombre d’initialisations
- max_iter: nombre d’itérations

CONCLUSION SUR LE K-MEANS

Avantages :

- Un algorithme **intuitif**.
- Un algorithme **itératif**
- Un algorithme **simple à implémenter**
- Un algorithme **rapide** la plupart du temps

Inconvénients:

- **Hypothèse forte** sur la forme des clusters (loi normale). Ne donnera pas toujours des résultats attendus.
- Ne fonctionne qu'avec la **distance euclidienne**.

CLUSTERING HIÉRARCHIQUE

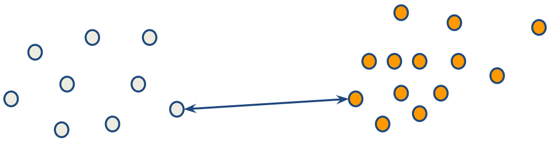
PRINCIPE

- Au départ, on a n points, chaque point est un cluster : n clusters.
- À chaque itération on groupe les deux clusters se ressemblant le plus.
- On arrête lorsqu'on a le nombre voulu de clusters.

DISTANCE ENTRE DEUX CLUSTERS

Distance minimale / Single Linkage: on calcule toutes les distances entre tous les points des deux clusters. On prend **la plus petite**:

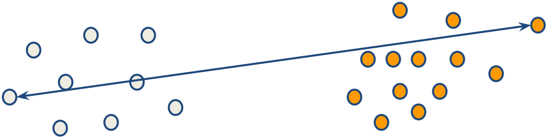
$$D(C_1, C_2) = \min_{x \in C_1, y \in C_2} d(x, y)$$



DISTANCE ENTRE DEUX CLUSTERS

Distance maximale / Complete linkage: on calcule toutes les distances entre tous les points des deux clusters. On prend **la plus grande**:

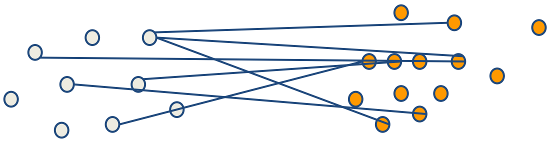
$$D(C_1, C_2) = \max_{x \in C_1, y \in C_2} d(x, y)$$



DISTANCE ENTRE DEUX CLUSTERS

Distance moyenne / Average linkage: on calcule toutes les distances entre tous les points des deux classes. On prend **la moyenne**:

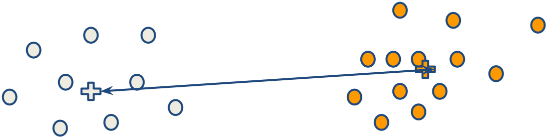
$$D(C_1, C_2) = \frac{1}{n_1 \times n_2} \sum_{x \in C_1, y \in C_2} d(x, y)$$



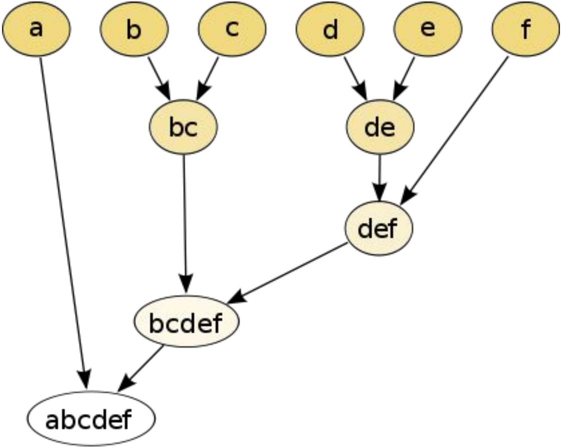
DISTANCE ENTRE DEUX CLUSTERS

Distance de Ward / Ward Linkage: distance entre les deux centres.

$$D(C_1, C_2) = \frac{n_1 \times n_2}{n_1 + n_2} d(m_1, m_2)$$



RÉSULTAT



Source: wikipedia

UTILISATION EN PYTHON

```
from sklearn.cluster import AgglomerativeClustering
model = AgglomerativeClustering(linkage = “complete”,
n_clusters = 10)
model.fit(X)
y = model.labels_
```

CONCLUSION SUR LA CAH

Avantages :

- Simple
- Fonctionne avec toutes les distances
- On voit l’évolution des clusters / Facile à analyser

Inconvénients :

- On doit choisir le nombre final de clusters
- On doit choisir le type de distance entre deux clusters.
- On peut donc obtenir des résultats très différents.

CONCLUSION

- Différents algorithmes et distances = différentes manières de voir des groupes
- Beaucoup de choix à faire qui vont influencer le résultat de manière cruciale.
- Facile à implémenter => satisfaction immédiate!