# K MOYENNES (K-MEANS)

Jérémie Cabessa Laboratoire DAVID, UVSQ

# APPRENTISSAGE NON-SUPERVISÉ (UNSUPERVISED LEARNING)

- $\blacktriangleright$  On dispose de variables explicatives  $X_1,\dots,X_p$  mais pas de variable réponse Y
- Il s'agit donc de découvrir des structures sous-jacentes à des données non étiquetées.
- ► Techniques de clustering, méthodes de représentation, visualisation, etc.

# APPRENTISSAGE NON-SUPERVISÉ (UNSUPERVISED LEARNING)

- lackbox On dispose de variables explicatives  $X_1,\dots,X_p$  mais pas de variable réponse Y
- ► Il s'agit donc de découvrir des structures sous-jacentes à des données non étiquetées.
- ► Techniques de clustering, méthodes de représentation, visualisation, etc.

# APPRENTISSAGE NON-SUPERVISÉ (UNSUPERVISED LEARNING)

- lackbox On dispose de variables explicatives  $X_1,\dots,X_p$  mais pas de variable réponse Y
- ► Il s'agit donc de découvrir des structures sous-jacentes à des données non étiquetées.
- Techniques de clustering, méthodes de représentation, visualisation, etc.

- ► Clustering: ensemble de techniques qui visent à identifier des clusters homogènes dans un dataset.
- On partitionne les données en clusters. Les données d'un même cluster sont "similaires" entre elles.
- Exemple: clusterisation de clients selon leurs âge, sexes, habitudes de consommation, ou tout autre type de données (market segmentation)
- ▶ Mais que signifie "similaires"? Quel est le critère de "similarité"?

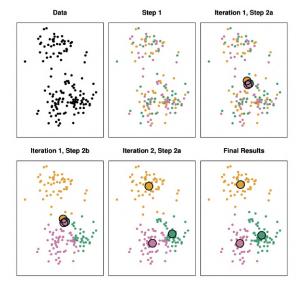
- ► Clustering: ensemble de techniques qui visent à identifier des clusters homogènes dans un dataset.
- On partitionne les données en clusters. Les données d'un même cluster sont "similaires" entre elles.
- Exemple: clusterisation de clients selon leurs âge, sexes, habitudes de consommation, ou tout autre type de données (market segmentation)
- ▶ Mais que signifie "similaires"? Quel est le critère de "similarité"?

- ► Clustering: ensemble de techniques qui visent à identifier des clusters homogènes dans un dataset.
- On partitionne les données en clusters. Les données d'un même cluster sont "similaires" entre elles.
- ► Exemple: clusterisation de clients selon leurs âge, sexes, habitudes de consommation, ou tout autre type de données (market segmentation)
- ▶ Mais que signifie "similaires"? Quel est le critère de "similarité"?

- ► Clustering: ensemble de techniques qui visent à identifier des clusters homogènes dans un dataset.
- On partitionne les données en clusters. Les données d'un même cluster sont "similaires" entre elles.
- ► Exemple: clusterisation de clients selon leurs âge, sexes, habitudes de consommation, ou tout autre type de données (market segmentation)
- ▶ Mais que signifie "similaires"? Quel est le critère de "similarité"?

- ► K-Means: le nombre de clusters est déterminé à l'avance.
- ► Hierarchical Clustering: le nombre optimal de clusters n'est pas connu à l'avance.

- ► K-Means: le nombre de clusters est déterminé à l'avance.
- ► Hierarchical Clustering: le nombre optimal de clusters n'est pas connu à l'avance.



- ▶ Soient un training set  $S_{\text{train}} = \{x_1, \dots, x_N\}$  et un nombre de clusters  $K \in \mathbb{N}$ .
- lacktriangle On cherche une bonne partition  $C_1,\ldots,C_k$  de  $S_{ ext{train}}$ , i.e.:

$$C_1 \cup \cdots \cup C_K = S_{\text{train}}$$
 et  $C_i \cap C_j = \emptyset$  pour tout  $i \neq j$ .

- ▶ Soient un training set  $S_{\text{train}} = \{x_1, \dots, x_N\}$  et un nombre de clusters  $K \in \mathbb{N}$ .
- ▶ On cherche une bonne partition  $C_1, \ldots, C_k$  de  $S_{\text{train}}$ , i.e.:

$$C_1 \cup \cdots \cup C_K = S_{\text{train}}$$
 et  $C_i \cap C_j = \emptyset$  pour tout  $i \neq j$ .

- ▶ Idée générale: on définit une notion de variation intra-cluster (within-cluster variation).
- Plus les data au sein d'un cluster  $C_k$  sont proches, plus sa variation intra-cluster  $W(C_k)$  est petite.
- La meilleure partition et celle qui minimise la variation intracluster totale (total within-cluster variation)  $\sum_{k=1}^{K} W(C_k)$ .

- ► Idée générale: on définit une notion de variation intra-cluster (within-cluster variation).
- Plus les data au sein d'un cluster  $C_k$  sont proches, plus sa variation intra-cluster  $W(C_k)$  est petite.
- La meilleure partition et celle qui minimise la variation intracluster totale (total within-cluster variation)  $\sum_{k=1}^{K} W(C_k)$ .

- ▶ Idée générale: on définit une notion de variation intra-cluster (within-cluster variation).
- Plus les data au sein d'un cluster  $C_k$  sont proches, plus sa variation intra-cluster  $W(C_k)$  est petite.
- La meilleure partition et celle qui minimise la variation intracluster totale (total within-cluster variation)  $\sum_{k=1}^{K} W(C_k)$ .

- La choix le plus simple et le plus commun de *variation intra- cluster* est basé sur la distance Euclidienne.
- ▶ La variation intra-cluster  $W(C_k)$  est la distance moyenne entre les point de  $C_k$ :

$$W(C_k) = \frac{1}{|C_k|} \sum_{x_i, x_{i'} \in C_k} ||x_i - x_{i'}||^2$$

On a alors le problème de minimisation suivant:

$$\underset{C_1,...,C_K}{\text{minimize}} \sum_{k=1}^K W(C_k) = \underset{C_1,...,C_K}{\text{minimize}} \sum_{k=1}^K \frac{1}{|C_k|} \sum_{\boldsymbol{x_i}, \boldsymbol{x_{i'}} \in C_k} \|\boldsymbol{x_i} - \boldsymbol{x_{i'}}\|^2$$

- La choix le plus simple et le plus commun de *variation intra- cluster* est basé sur la distance Euclidienne.
- ▶ La variation intra-cluster  $W(C_k)$  est la distance moyenne entre les point de  $C_k$ :

$$W(C_k) = \frac{1}{|C_k|} \sum_{x_i, x_{i'} \in C_k} ||x_i - x_{i'}||^2$$

On a alors le problème de *minimisation* suivant

$$\underset{C_1,...,C_K}{\text{minimize}} \sum_{k=1}^K W(C_k) = \underset{C_1,...,C_K}{\text{minimize}} \sum_{k=1}^K \frac{1}{|C_k|} \sum_{\boldsymbol{x_i}, \boldsymbol{x_{i'}} \in C_k} \|\boldsymbol{x_i} - \boldsymbol{x_{i'}}\|^2$$

- La choix le plus simple et le plus commun de *variation intra- cluster* est basé sur la distance Euclidienne.
- ▶ La variation intra-cluster  $W(C_k)$  est la distance moyenne entre les point de  $C_k$ :

$$W(C_k) = \frac{1}{|C_k|} \sum_{x_i, x_{i'} \in C_k} ||x_i - x_{i'}||^2$$

On a alors le problème de minimisation suivant:

$$\underset{C_1,\dots,C_K}{\text{minimize}} \ \sum_{k=1}^K W(C_k) \ = \ \underset{C_1,\dots,C_K}{\text{minimize}} \ \sum_{k=1}^K \frac{1}{|C_k|} \sum_{\boldsymbol{x_i,x_{i'}} \in C_k} \|\boldsymbol{x_i} - \boldsymbol{x_{i'}}\|^2$$

- L'approche "bruteforce" de ce problème est trop complexe.
- En effet, il y a environ K<sup>N</sup> manières de partitionner N data en K clusters.
- L'algorithme K-means est une approche simple qui converge vers un minimum local de ce problème.

- L'approche "bruteforce" de ce problème est trop complexe.
- ▶ En effet, il y a environ  $K^N$  manières de partitionner N data en K clusters.
- L'algorithme K-means est une approche simple qui converge vers un minimum local de ce problème.

- L'approche "bruteforce" de ce problème est trop complexe.
- ▶ En effet, il y a environ  $K^N$  manières de partitionner N data en K clusters.
- L'algorithme K-means est une approche simple qui converge vers un minimum local de ce problème.

- ▶ *Initialisation:* Séparer les data en *K* clusters aléatoires.
- ltération: Tant que les clusters ne se sont pas stabilisés, faire
  - $\triangleright$  Calculer les *centroïdes*  $c_1, \ldots, c_K$  des clusters  $C_1, \ldots, C_K$ .
    - le plus proche de lui.

- ▶ *Initialisation*: Séparer les data en *K* clusters aléatoires.
- ltération: Tant que les clusters ne se sont pas stabilisés, faire:
  - ightharpoonup Calculer les *centroïdes*  $c_1, \ldots, c_K$  des clusters  $C_1, \ldots, C_K$ .
  - Assigner chaque point x au cluster  $C_k$  dont le centroïde  $c_k$  est le plus proche de lui.

- ▶ *Initialisation:* Séparer les data en *K* clusters aléatoires.
- ltération: Tant que les clusters ne se sont pas stabilisés, faire:
  - ightharpoonup Calculer les *centroïdes*  $c_1, \ldots, c_K$  des clusters  $C_1, \ldots, C_K$ .
  - Assigner chaque point x au cluster  $C_k$  dont le centroïde  $c_k$  est le plus proche de lui.

- ▶ *Initialisation:* Séparer les data en *K* clusters aléatoires.
- ltération: Tant que les clusters ne se sont pas stabilisés, faire:
  - ightharpoonup Calculer les *centroïdes*  $c_1, \ldots, c_K$  des clusters  $C_1, \ldots, C_K$ .
  - Assigner chaque point x au cluster  $C_k$  dont le centroïde  $c_k$  est le plus proche de lui.

```
Algorithm 1: k_means(dataset, k, stop_dist, max_iter)
```

```
Algorithm 1: k_means(dataset, k, stop_dist, max_iter)

iter = 0

dist = [inf] * k

stop_dist = [stop_dist] * k

centroids = initialize_centroids(dataset)

while (dist > stop_dist).any() and iter \le max_iter do

clusters = assign_clusters(dataset, centroids)

dist = norm(centroids - new_centroids)

centroids = new_centroids

iter = iter + i

end

return clusters, centroids
```

```
Algorithm 1: k_means(dataset, k, stop_dist, max_iter)

iter = 0

dist = [inf] * k

stop_dist = [stop_dist] * k

centroids = initialize_centroids(dataset)

while (dist > stop_dist).any() and iter \leq max_iter do

| clusters = assign_clusters(dataset, centroids)
| dist = norm(centroids - new_centroids)
| centroids = new_centroids
| iter = iter + 1
| end
```

```
Algorithm 1: k_means(dataset, k, stop_dist, max_iter)

iter = 0

dist = [inf] * k

stop_dist = [stop_dist] * k

centroids = initialize_centroids(dataset)

while (dist > stop_dist).any() and iter \le max_iter do

clusters = assign_clusters(dataset, centroids)

dist = norm(centroids - new_centroids)

centroids = new_centroids

iter = iter + 1

end

return clusters, centroids
```

```
Algorithm 1: k_means(dataset, k, stop_dist, max_iter)

iter = 0

dist = [inf] * k

stop_dist = [stop_dist] * k

centroids = initialize_centroids(dataset)

while (dist > stop_dist).any() and iter \le max_iter do

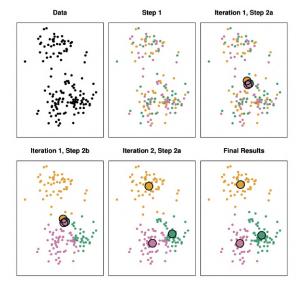
| clusters = assign_clusters(dataset, centroids)
| dist = norm(centroids - new_centroids)
| centroids = new_centroids
| iter = iter + 1

end

return clusters centroids
```

#### Algorithm 1: k means(dataset, k, stop dist, max iter)

```
iter = 0
dist = [inf] * k
stop dist = [stop dist] * k
centroids = initialize centroids(dataset)
while (dist > stop dist).any() and iter ≤ max iter do
    clusters = assign clusters(dataset, centroids)
    {\tt dist} = {\tt norm}({\tt centroids} - {\tt new \ centroids})
    {\tt centroids} = {\tt new} \ {\tt centroids}
    iter = iter + 1
end
return clusters, centroids
```



# BIBLIOGRAPHIE



James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An Introduction to Statistical Learning: with Applications in R, volume 103 of Springer Texts in Statistics. Springer, New York.