Data Science for Actuaries (ACT6100)

Arthur Charpentier

Non-Supervisé # 3 (Clustering)

automne 2Q20

https://github.com/freakonometrics/ACT6100/

Clustering with R & References

- Husson, Lê & Pagès (2017) Exploratory Multivariate Analysis by Example Using R, chapter 4,
- Husson, Lê & Pagès (2017) Exploratory Multivariate Analysis by Example Using R, chapter 4,

```
> FactoMineR::HCPC()
> cluster::agnes()
> stats::hclust()
> stats::kmeans()
```

Objectif

- On a n observations dans un espace de dimension p:

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \cdots, x_{ip}) \in \mathbb{R}^p, \quad i = 1, \ldots, n$$

  et aucune variable réponse.
- Chacune des observations est caractérisée par un poids p_i, $i = 1, \ldots, n$ tels que $\displaystyle\sum_{i=1}^{n} p_i = 1$.
- Généralement, on a $p_i = 1/n$, $i = 1, \ldots, n$.
- On veut déterminer K groupes $C_1, \ldots, C_K$.

Indice de similarité

Si on note N l'ensemble des observations $\{1, \ldots, n\}$, alors l'indice de similarité est une application $s : N \times N \to \mathbb{R}^+$ telle que

$$\begin{aligned} s(i,j) &= s(j,i), & \forall (i,j) \in N \times N \\ s(i,i) &= \overline{s} > 0, & \forall i \in N \\ s(i,j) &\leq \overline{s}, & \forall (i,j) \in N \times N. \end{aligned}$$

On peut également définir une version normée de cet indice:

$$s^*(i,j) = \frac{1}{\overline{s}} s(i,j), \qquad \forall (i,j) \in N \times N.$$

Indice de dissimilarité

Si on note N l'ensemble des observations, alors l'indice de dissimilarité est une application $d : N \times N \to \mathbb{R}^+$ telle que

$$\begin{aligned} d(i,j) &= d(j,i), \qquad \forall (i,j) \in N \times N \\ d(i,j) &= 0 \Leftrightarrow i = j. \end{aligned}$$

On peut également définir une version normée de cet indice:

$$d^*(i,j) = \frac{1}{\bar{d}} d(i,j), \qquad \forall (i,j) \in N \times N.$$

On a alors

$$d^*(i,j) = 1 - s^*(i,j) \text{ et } s^*(i,j) = 1 - d^*(i,j).$$

Distance

Une distance est un indice de dissimilarité qui vérifie, en plus, la propriété suivante (inégalité triangulaire):

$$d(i,j) \leq d(i,k) + d(k,j), \qquad \forall (i,j,k) \in N \times N \times N.$$

Pour $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$, on peut définir plusieurs mesures de distance (présentées dans $\mathbb{R}^2$)

- la mesure euclidienne (ℓ_2):

$$d^2(x,y) = (x_1 - y_1)^2 + (x_2 - y_2)^2;$$

- la mesure normalisée de Mahalanobis: $Q = D_{s^2}^{-1} = \text{diag}(s_1^{-1}, \ldots, s_p^{-1})$, où s_j^2 est la variance de la $j$ième composante de x; et la mesure euclidienne généralisée avec mesure $\mathbf{Q} > 0$ par

$$d^2(x,y) = (\mathbf{x} - \mathbf{y})^T \mathbf{Q} (\mathbf{x} - \mathbf{y}).$$

Distance

- la mesure Manhattan (ℓ_1):

$$d(x,y) = |x_1 - y_1| + |x_2 - y_2|;$$

- la mesure euclidienne (ℓ_2):

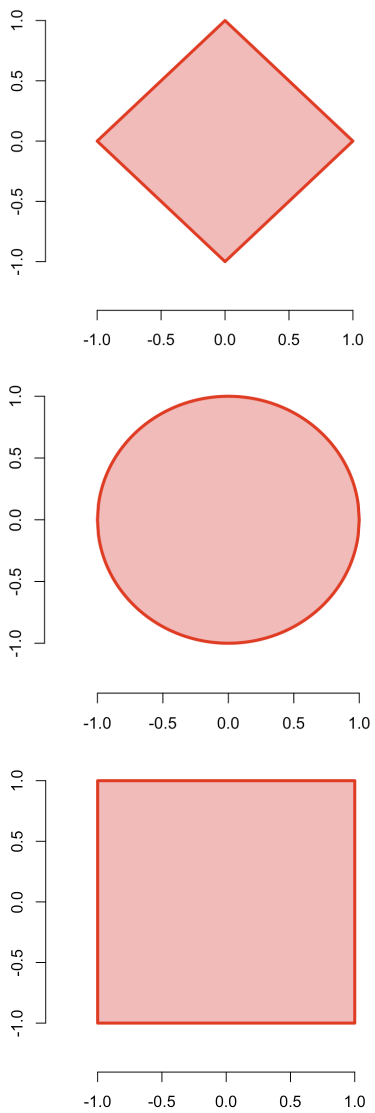
$$d^2(x,y) = (x_1 - y_1)^2 + (x_2 - y_2)^2;$$

- la mesure Minkowski d'ordre p (ℓ_p):

$$d(x,y) = ((x_1 - y_1)^p + (x_2 - y_2)^p)^{1/p};$$

- la mesure maximale (ℓ_∞):

$$d(x,y) = (\max(|x_1 - y_1|, |x_2 - y_2|));$$



Distance

- On utilise généralement la distance euclidienne si toutes les variables sont mesurées sur la même échelle.
- Si toutes les variables ne sont pas mesurées sur la même échelle, on utilise plutôt la mesure normalisée (Mahalanobis).
- De façon équivalente, on peut également centrer et réduire les données avant d'utiliser la distance euclidienne.

Euclidean Distance

Define the distance $d^2(\boldsymbol{u},\boldsymbol{v}) = (\boldsymbol{u}-\boldsymbol{v})^\top(\boldsymbol{u}-\boldsymbol{v}) = \|\boldsymbol{u}-\boldsymbol{v}\|^2$

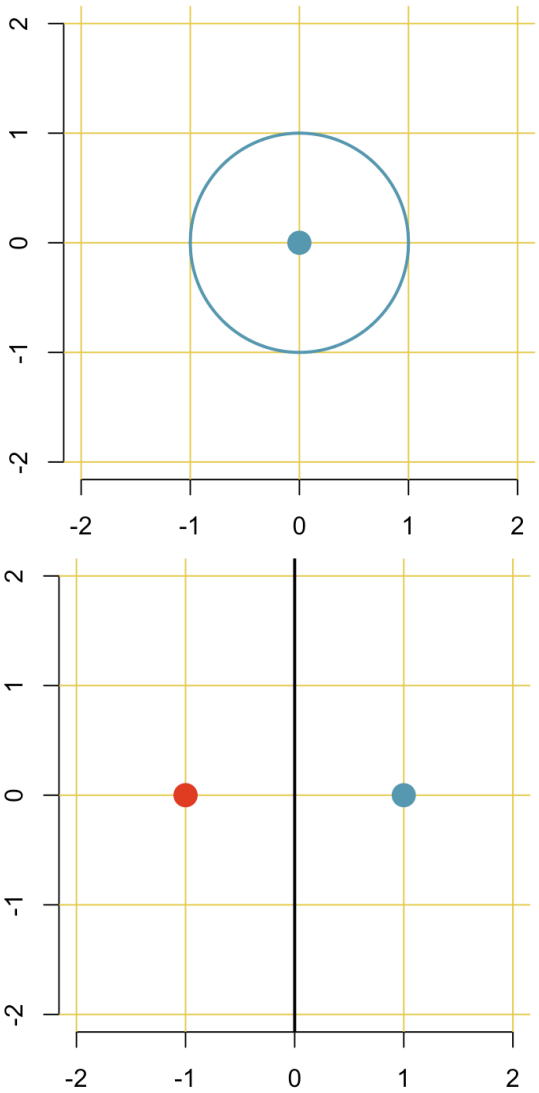
Set of points $\boldsymbol{u}$ such that $d(\mathbf{0},\boldsymbol{v}) = \|\boldsymbol{u}\| = 1$

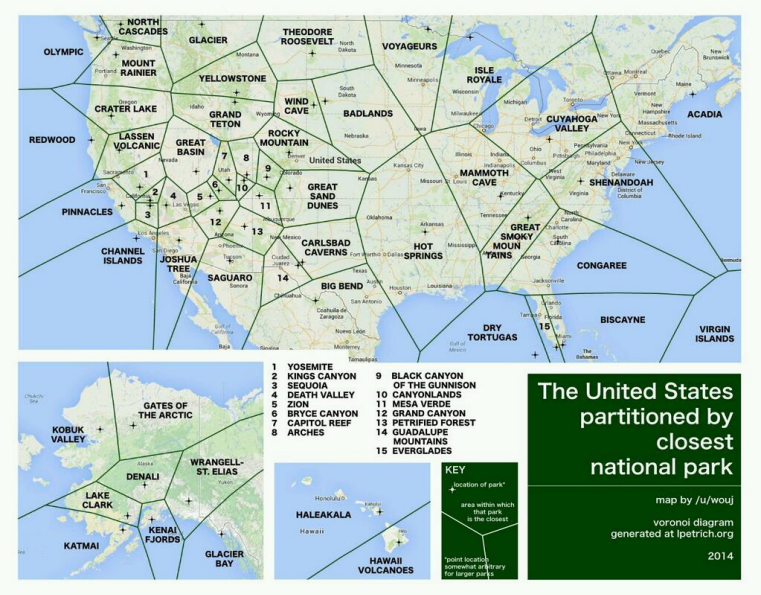
- circle (or sphere in higher dimension)

Set of points $\boldsymbol{x}$ such that $d(\boldsymbol{x},\boldsymbol{u}) = d(\boldsymbol{x},\boldsymbol{v})$?

- straight line (or plane in higher dimension)

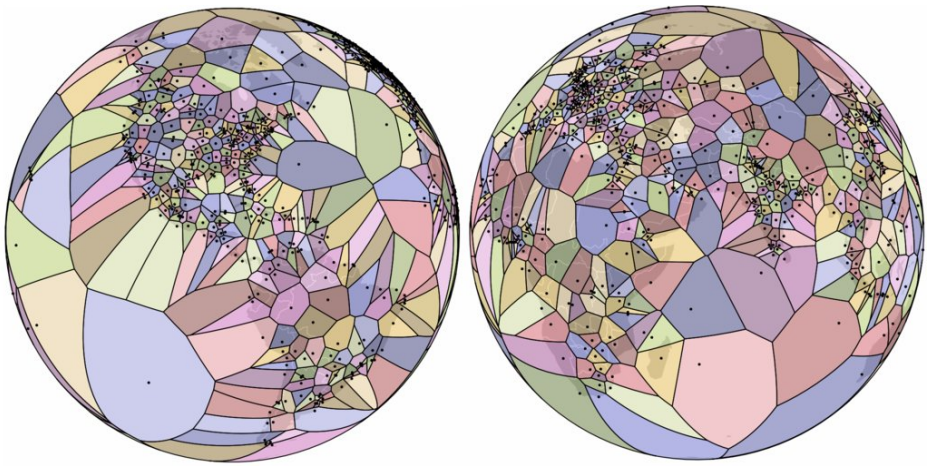
(orthogonal to $\boldsymbol{u}-\boldsymbol{v}$)



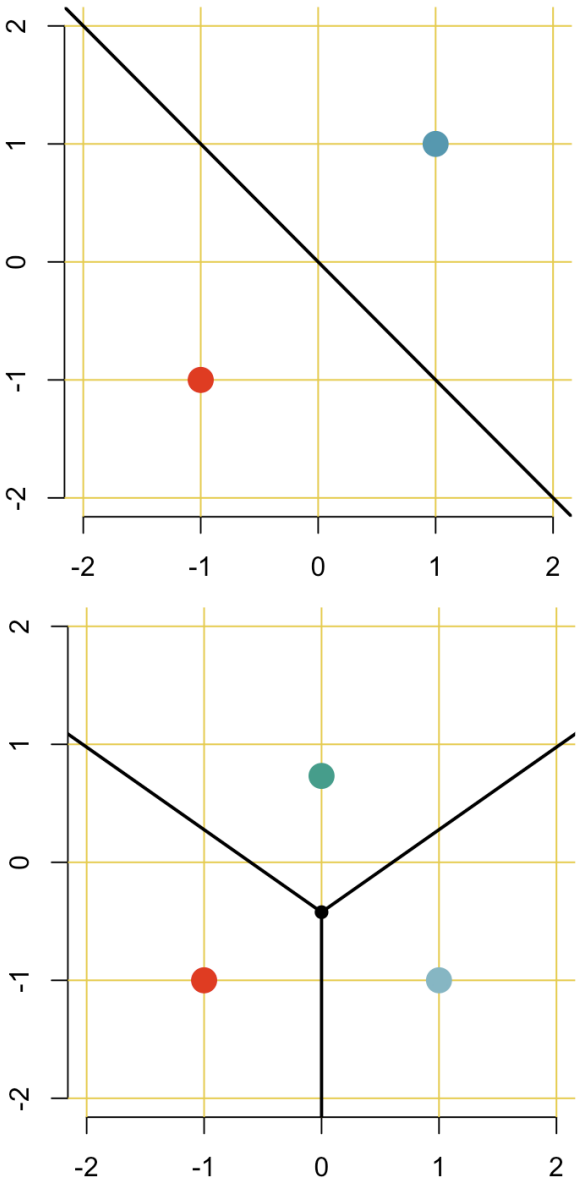


Euclidean Distance & Voronoi Sets

Given a set of points $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_n$
Set of points $\boldsymbol{x}$ the closest to $\boldsymbol{u}_i$?
(see also Delaunay triangulation)



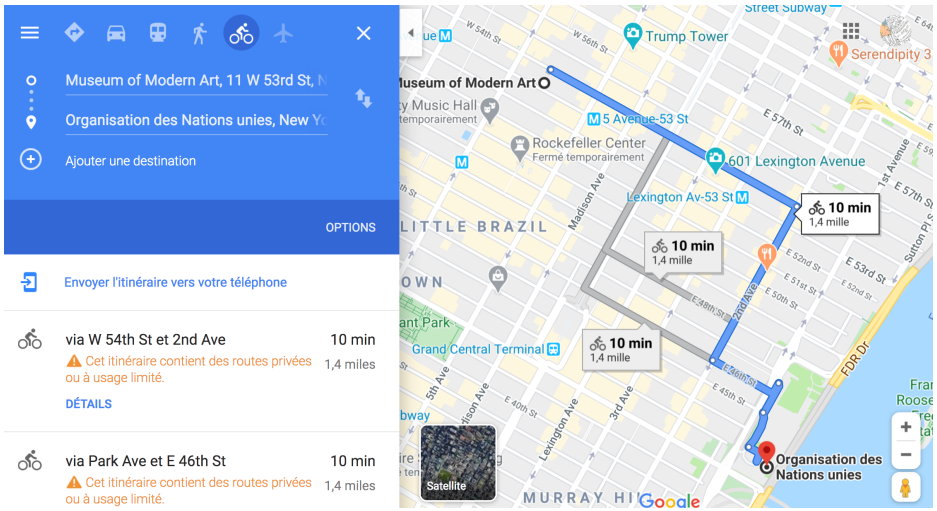
(intersections are straight linears or planes)



Manhattan (ℓ_1) Distance

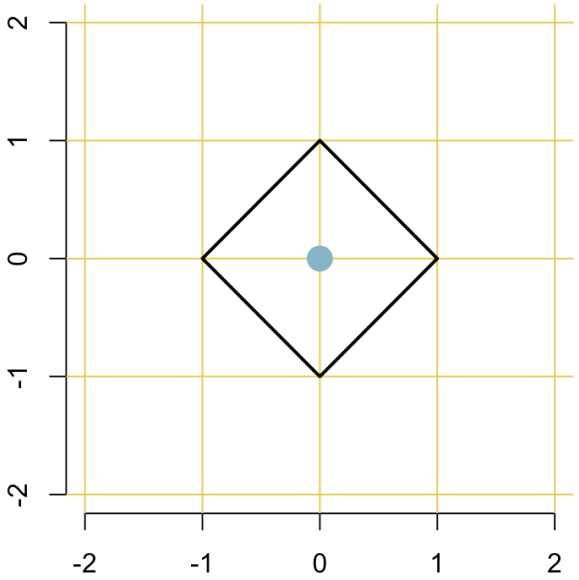
$\|\boldsymbol{u}\| = |u_1| + |u_2|$ is also a norm (Manhattan)
Set of points $\boldsymbol{u}$ such that $d(\mathbf{0}, \boldsymbol{v}) = \|\boldsymbol{u}\| = 1$

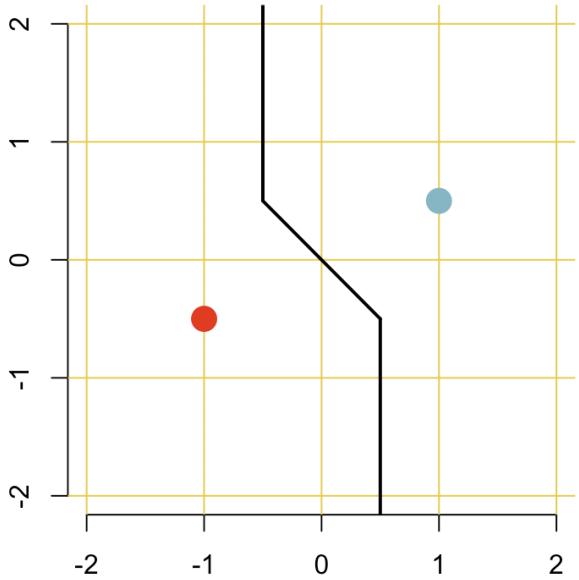
- square (or cube in higher dimension)



Set of points $\boldsymbol{x}$ such that $d(\boldsymbol{x}, \boldsymbol{u}) = d(\boldsymbol{x}, \boldsymbol{v})$?

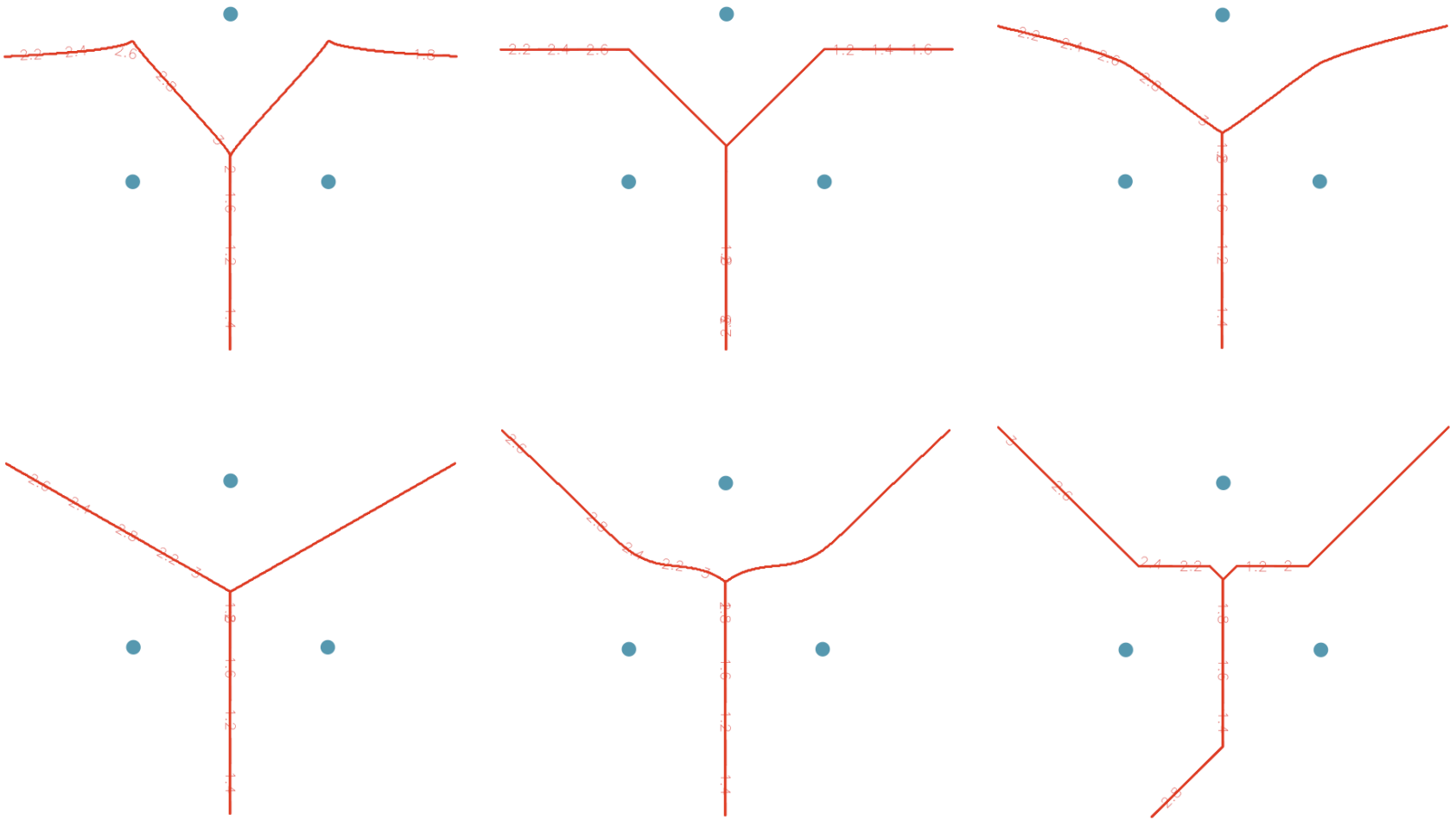
- portions of straight lines (or planes)





ℓ_p Distance & Voronoi Sets

$$p \in \{0.8, 1, 1.5, 2, 3, \infty\},\ d(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{i-1}^{d} |x_i - y_i|^p \right)^{1/p}$$



Partition

- Une partition $\mathcal{P}_K$ de N en K groupes est un ensemble $\{C_1, \ldots, C_K\}$ de groupes non-vides, d'intersections nulles et dont l'union permet de retrouver N, c'est-à-dire
  - $C_k \neq \emptyset$, $\forall k \in \{1, \ldots, K\}$;
  - $C_k \cap C_j = \emptyset$, $\forall k \neq j$; et
  - $C_1 \cup \cdots \cup C_K = N$.

Example: On considère $N = \{1, 2, 3, 4, 5, 6, 7\}$ et

$$\mathcal{P}_3 = \{\{7\}, \{5, 4, 6\}, \{1, 2, 3\}\}.$$

Il s'agit d'une partition.

k-means

Given n points $\boldsymbol{x}_i$ in $\mathbb{R}^d$, we want to find $\boldsymbol{\mu}_1^\star, \cdots, \boldsymbol{\mu}_k^\star$, solutions of

$$V_{k-\text{means}} = \min \left\{ \sum_{j=1}^{k} \sum_{i \in S_j} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2 \right\}$$

that we can write

$$V_{k-\text{means}} = \min \{ \sum_{i=1}^{n} \min_{j=1,\cdots,k} \underbrace{\left\|\boldsymbol{x}_i - \boldsymbol{\mu}_j\right\|^2}_{\text{within inertia}} \}$$

In k-means, clusters are defined as Voronoi sets obtained from means $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_k$

k-means

Finding the optimal solution to the k-means clustering problem for n observations in d dimensions can be (exactly) solved in time $O(n^{dk+1}\log n)$... which is very long.

Given an initial set of k-means, $\boldsymbol{\mu}_1,\cdots,\boldsymbol{\mu}_k$, alternate between the two following steps (also called Lloyd's algorithm, from Lloyd (1957)

- Assignment step: we partition observations according to the Voronoi diagram generated by $\boldsymbol{\mu}_1,\cdots,\boldsymbol{\mu}_k$'s,

$$S_i^{(t)}=\left\{\boldsymbol{x}_j:\left\|\boldsymbol{x}_j-\boldsymbol{\mu}_i^{(t)}\right\|^2\leq\left\|\boldsymbol{x}_j-\boldsymbol{\mu}_{i'}^{(t)}\right\|^2\ \forall i'\in\{1,\cdots,k\}\right\},$$

  where each $\boldsymbol{x}_j$ is assigned to exactly one class $S_i^{(t)}$, at step t.
- Update step: we compute the new centroids of the clusters,

$$\boldsymbol{\mu}_i^{(t+1)}=\frac{1}{|S_i^{(t)}|}\sum_{\boldsymbol{x}_j\in S_i^{(t)}}\boldsymbol{x}_j$$

k-means

Key why this might works :

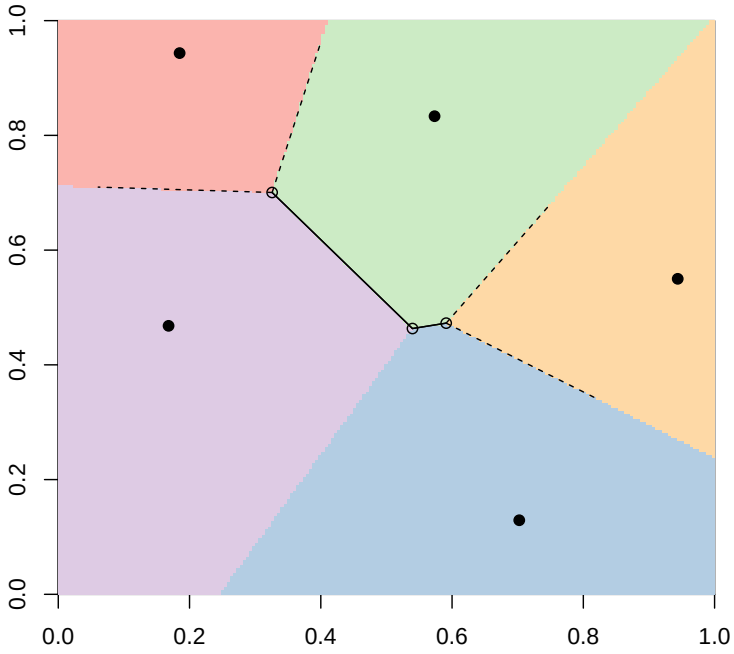
$$\frac{1}{\#I_j}\sum_{i,i'\in I}\sum_{j=1}^{k}[x_{il}-x_{i'l}]^2 = 2\sum_{i\in I}\sum_{j=1}^{k}[x_{il}-\overline{x}_{jl}]^2 \text{ where } \overline{x}_{jl} = \frac{1}{\#I_j}\sum_{i\in I_j} x_{i,l}$$

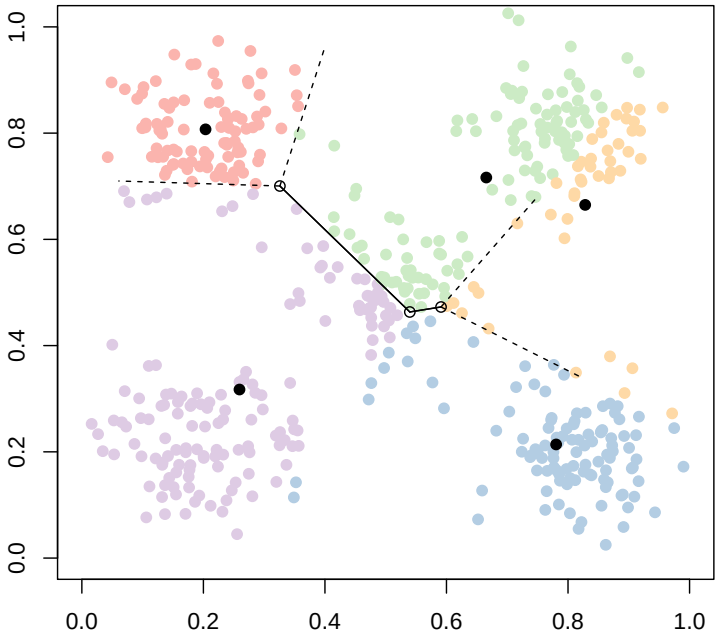
The value of

$$\min_{I_1,\cdots,I_K}\left\{\sum_{j=1}^{K}\frac{1}{\#I_j}\sum_{i,i'\in I_j}\sum_{j=1}^{k}[x_{il}-x_{i'l}]^2\right\}$$

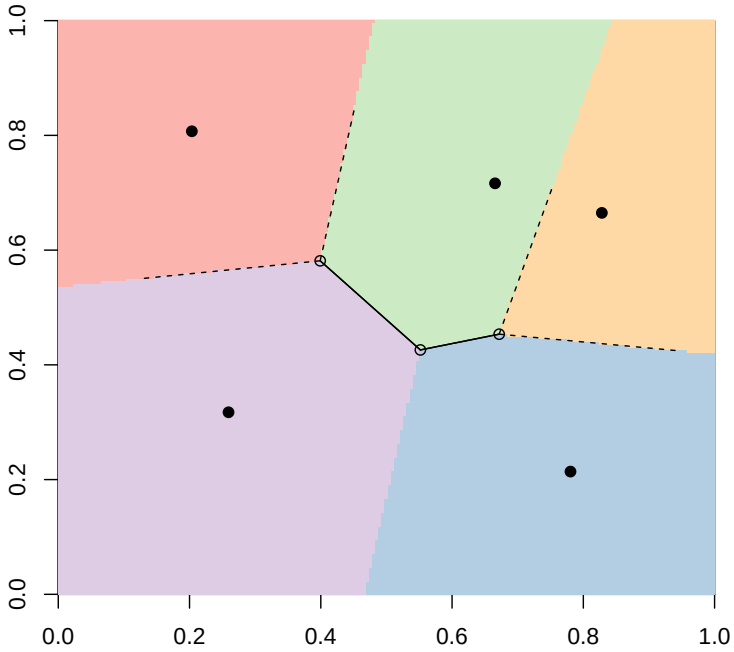
will necessarily decrease (but no guarantee to reach the minimum, since the function is not convex).

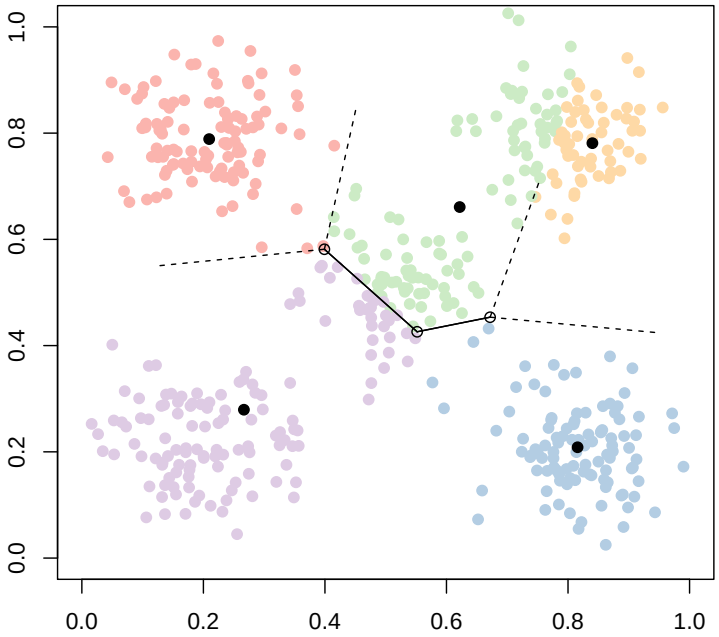
k-means



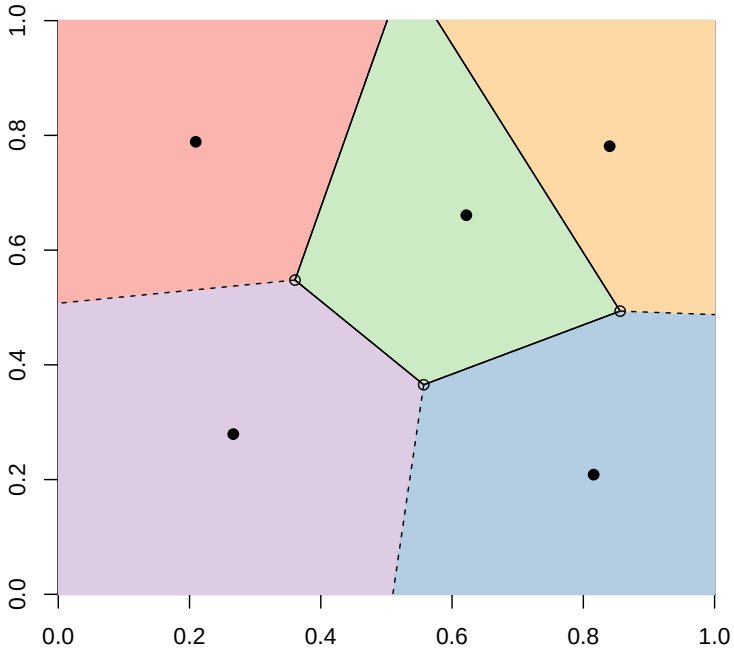


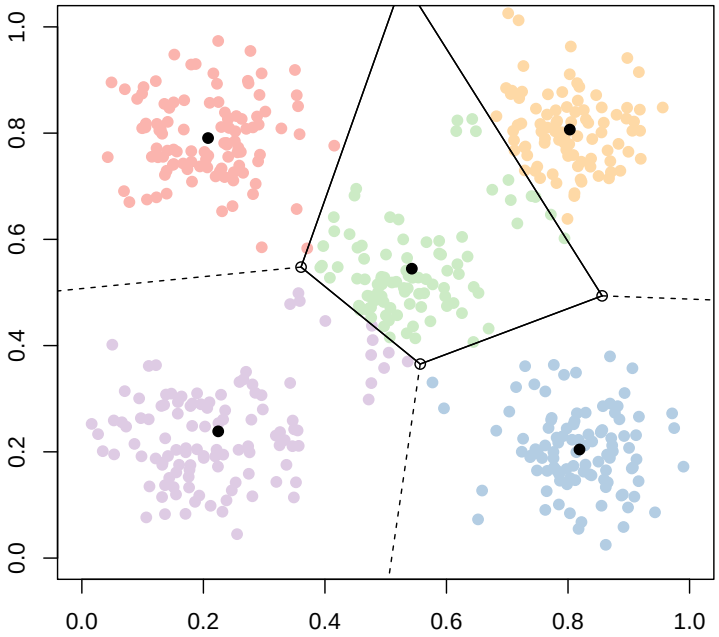
k-means



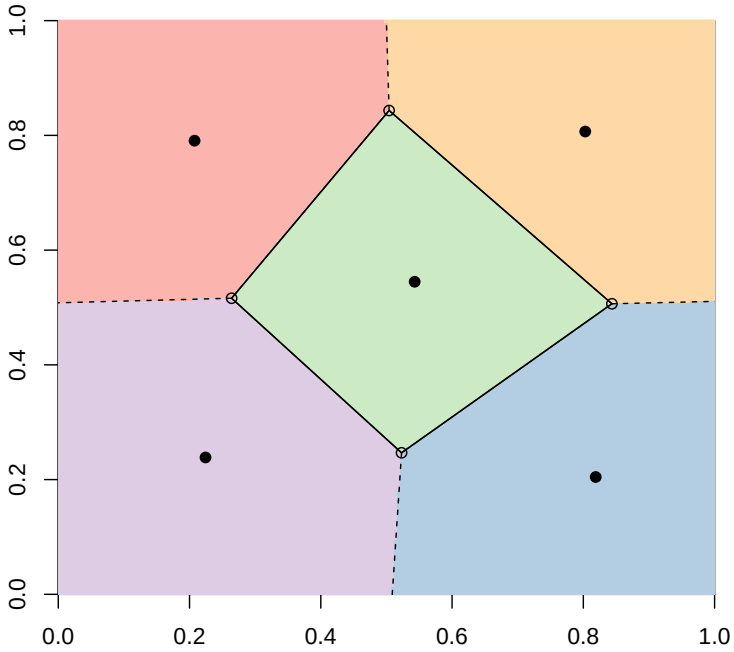


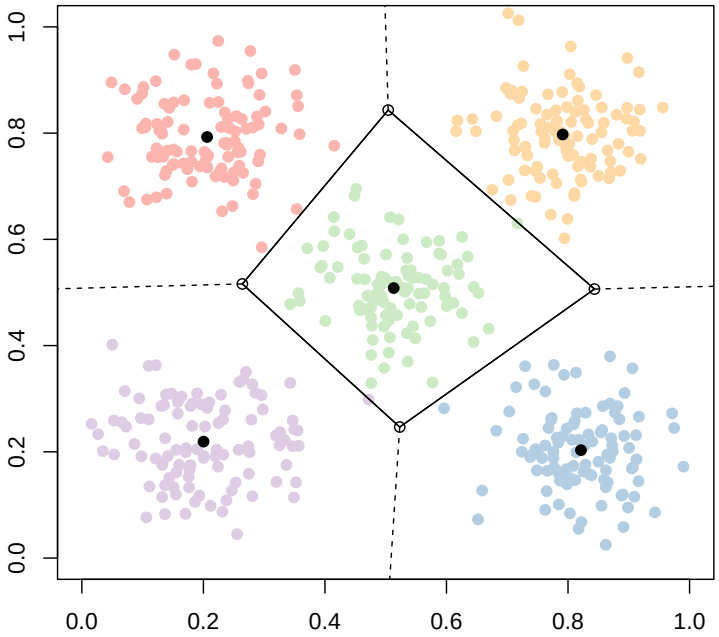
k-means



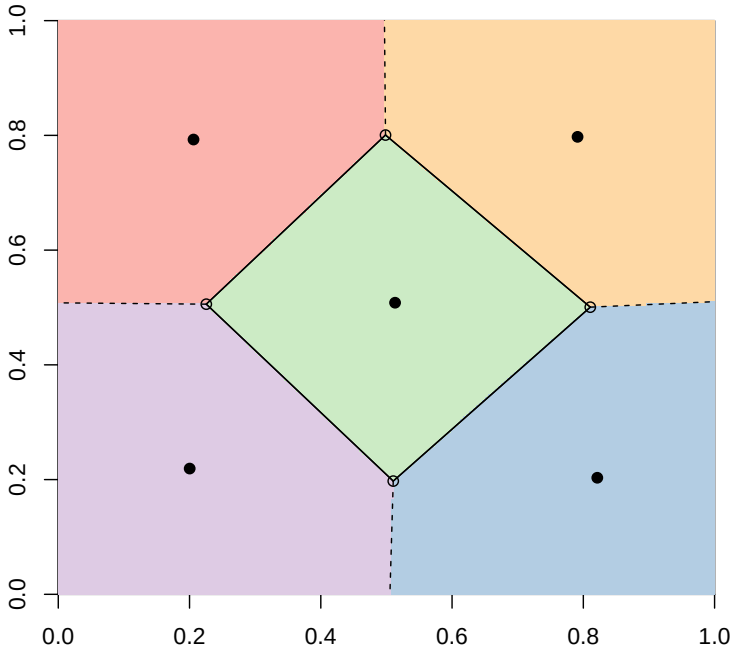


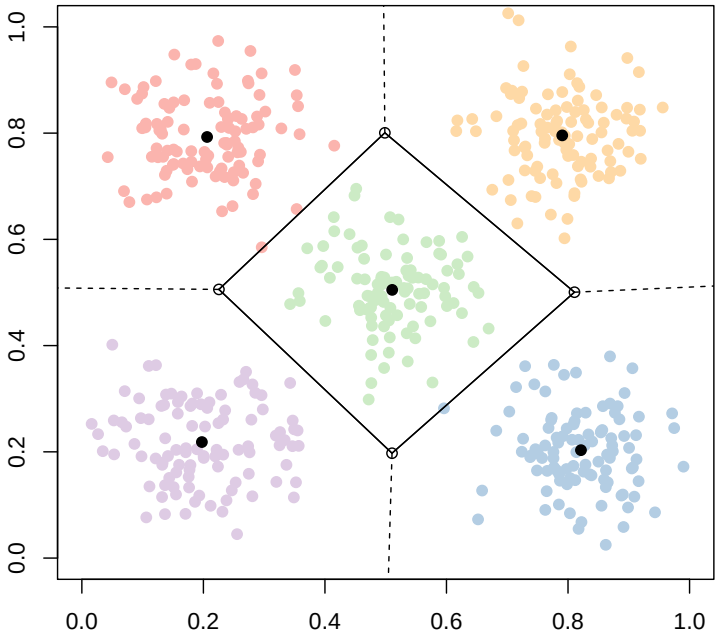
k-means



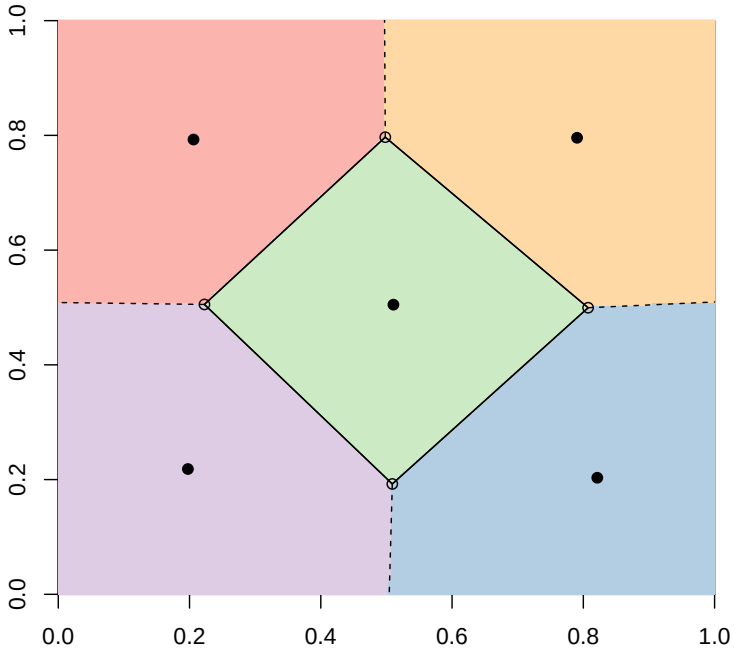


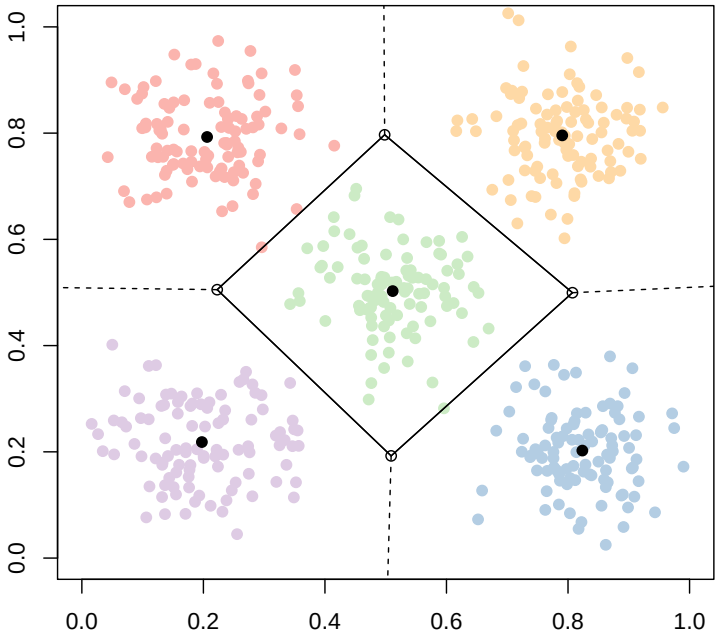
k-means



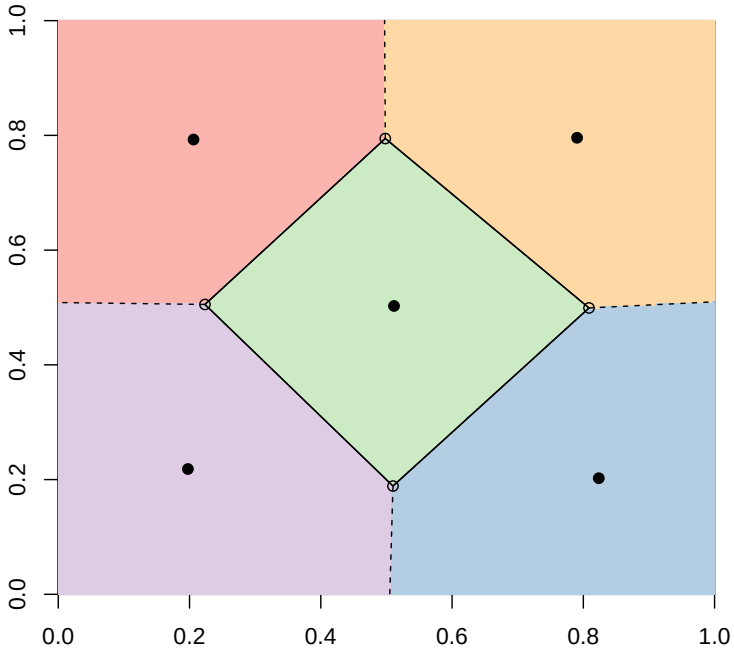


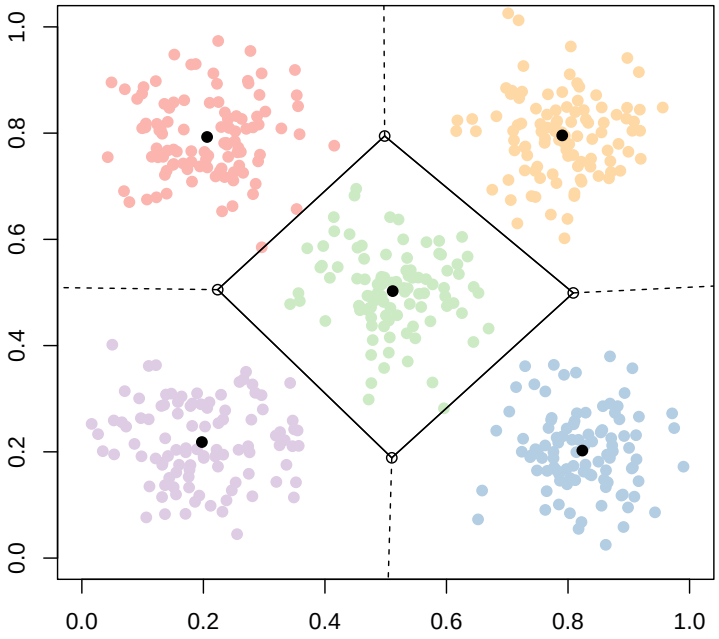
k-means



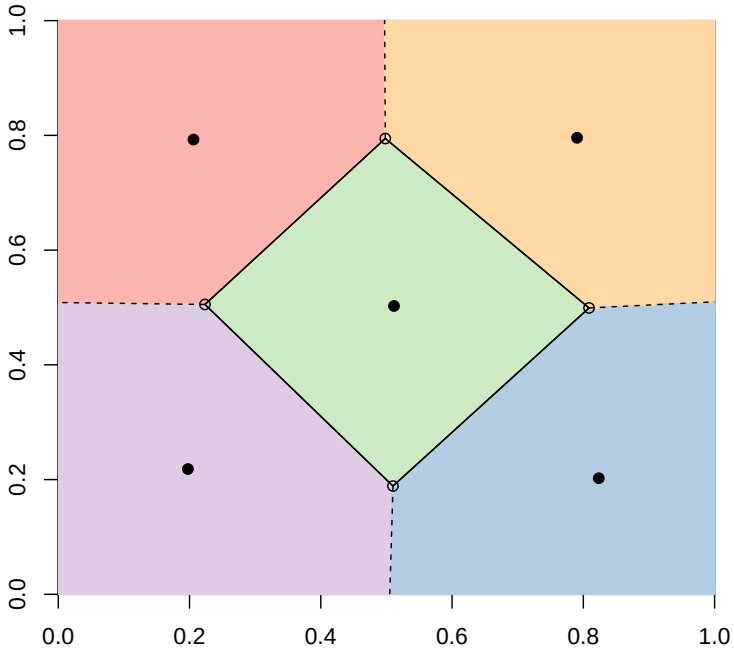


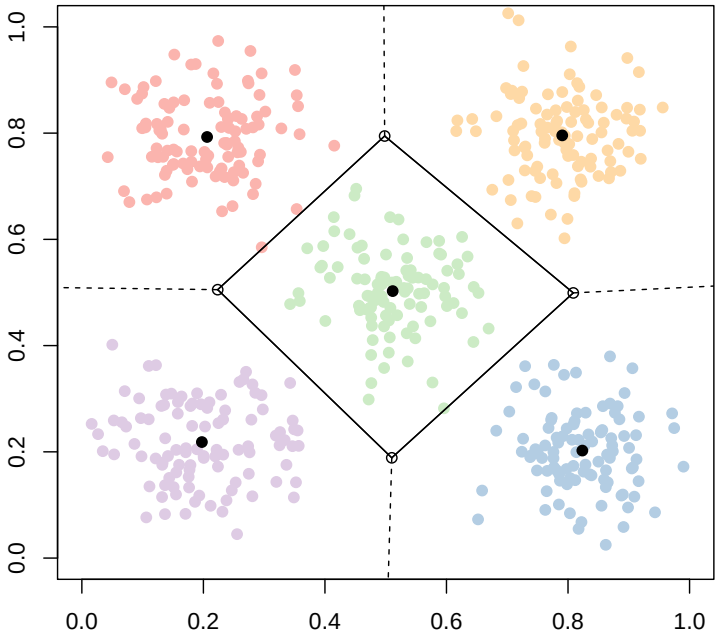
k-means



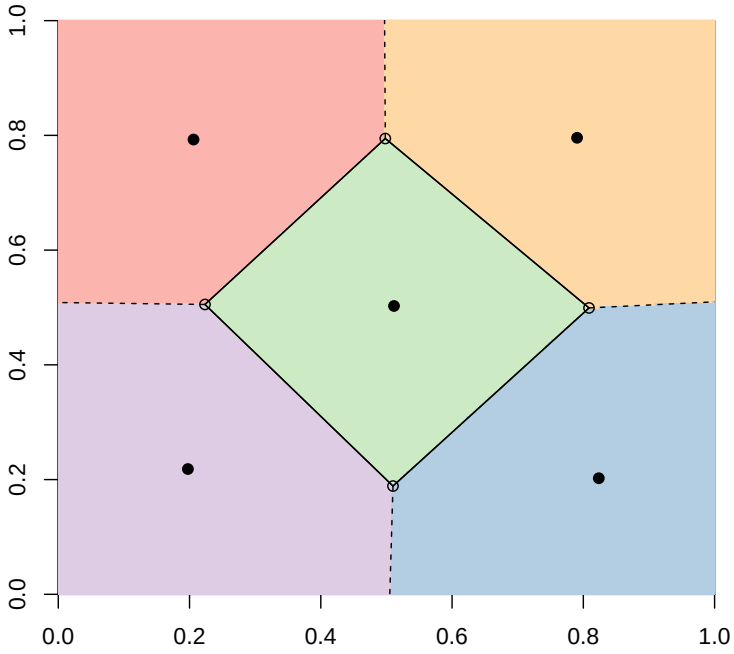


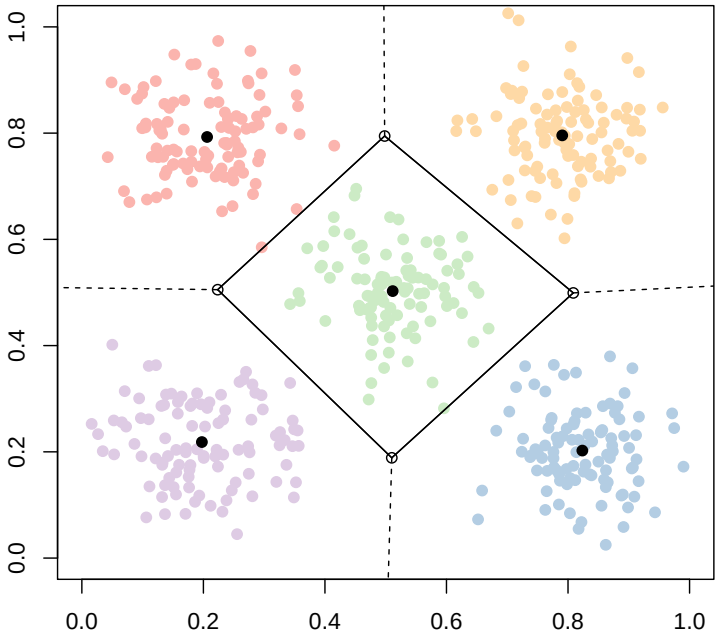
k-means



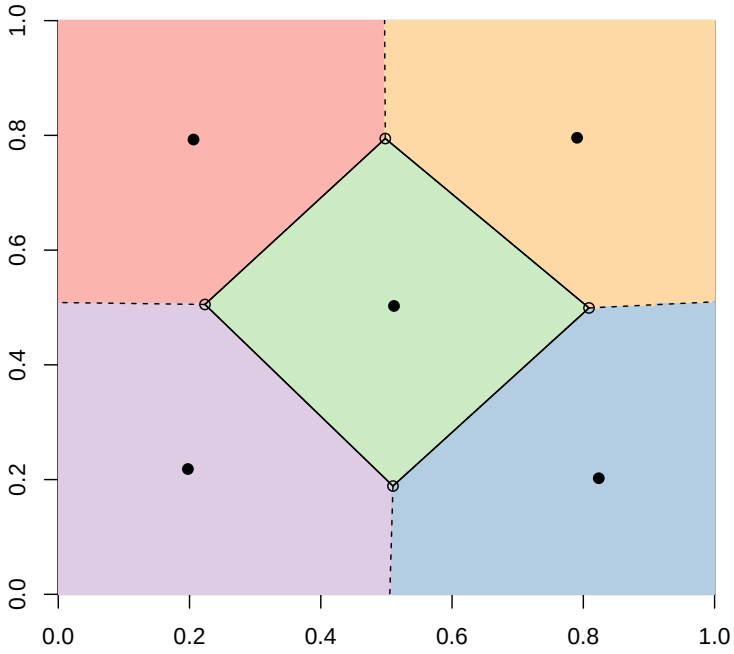


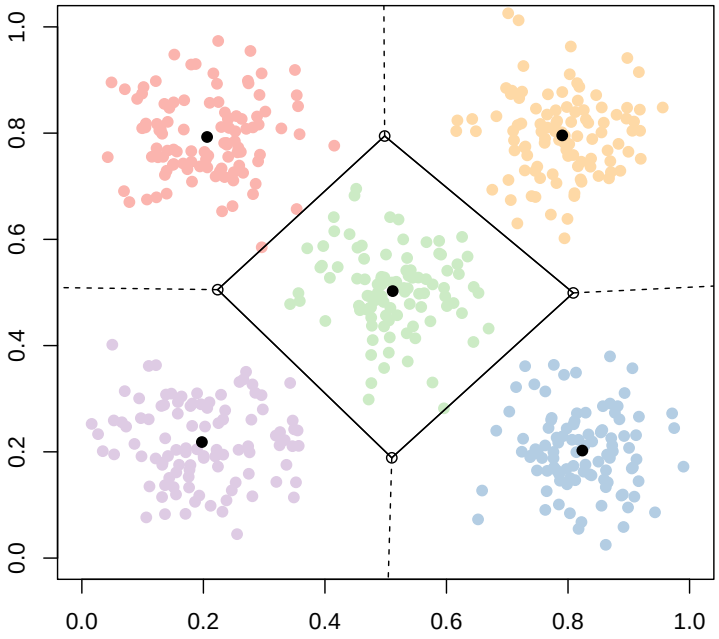
k-means





k-means





k-means

On peut visualiser des clusters d'individus sur la sortie d'une ACP (par exemple les secteurs d'activité en Europe)

```
> jobs = read.csv(header=TRUE,
"http://user.math.uzh.ch/furrer/download/sta121/
    europejobs.csv"
)
> res = PCA(jobs)
> ind = get_pca_ind(res)
> res.km = kmeans(ind$coord, centers = 3, nstart = 15)
> grp = as.factor(res.km$cluster)
> fviz_pca_ind(res, col.ind = grp)
```

En faisant une ACP, on a des observations dans $\mathbb{R}^5$ (par défaut)

k-means

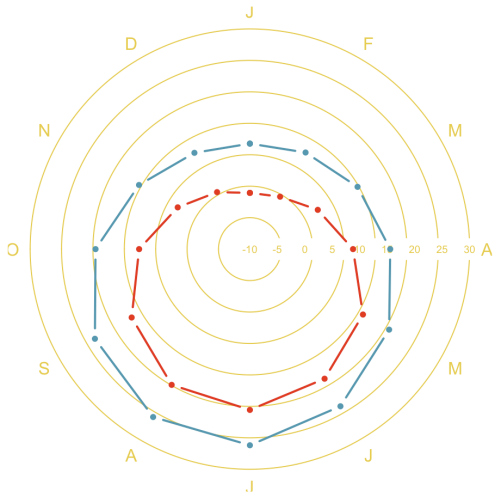
Individuals − PCA
Dim1 (38.7%)
Dim2 (23.7%)
Groups
1
2
3
Individuals − PCA
Dim1 (38.7%)
Dim2 (23.7%)
Groups
1
2
3
Individuals − PCA
Dim1 (38.7%)
Dim2 (23.7%)
Groups
1
2
3
Individuals − PCA
Dim1 (38.7%)
Dim2 (23.7%)
Groups
1
2
3
Individuals - PCA
Dim1 (38.7%)
Dim2 (23.7%)
Col.
1
2
3
Individuals - PCA
Dim1 (38.7%)
Dim2 (23.7%)
Col.
1
2
3

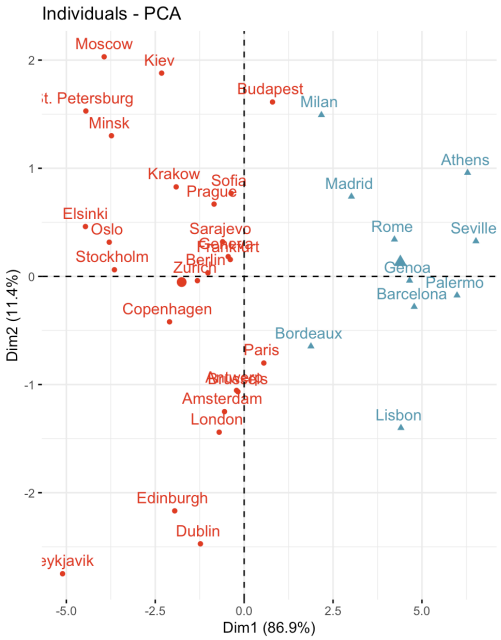
k-means Interpretation

As it typically converges to a local minimum, it is optimal to run k-means algorithm several times so as to empirically determine good clusters.
k-means work well when (true) clusters are compact and convex subsets (linear decision boundaries). If not, we cannot expect to converge.
Remark: This is not what we do with hierarchical clustering.

Clusters Interpretation

```
> res = PCA(temperature[,1:12])
> ind = get_pca_ind(res)
> res.km = kmeans(ind$coord, centers = 2, nstart = 15)
> grp = as.factor(res.km$cluster)
> df = data.frame(grp,temperature[,1:12])
> dsc = aggregate(df[,2:13],by=list(df$grp),mean)
> dsc
   Jan  Feb  Mar  Apr  May  Jun
1 -1.2 -0.4  2.8  7.3 12.3 15.8
2  7.7  8.8 11.2 14.1 17.8 21.5
   Jul  Aug  Sep  Oct  Nov  Dec
1 17.8 17.1 13.4  8.7  3.7  0.5
2 24.2 23.8 21.1 16.7 11.9  8.8
```





Hiérarchie

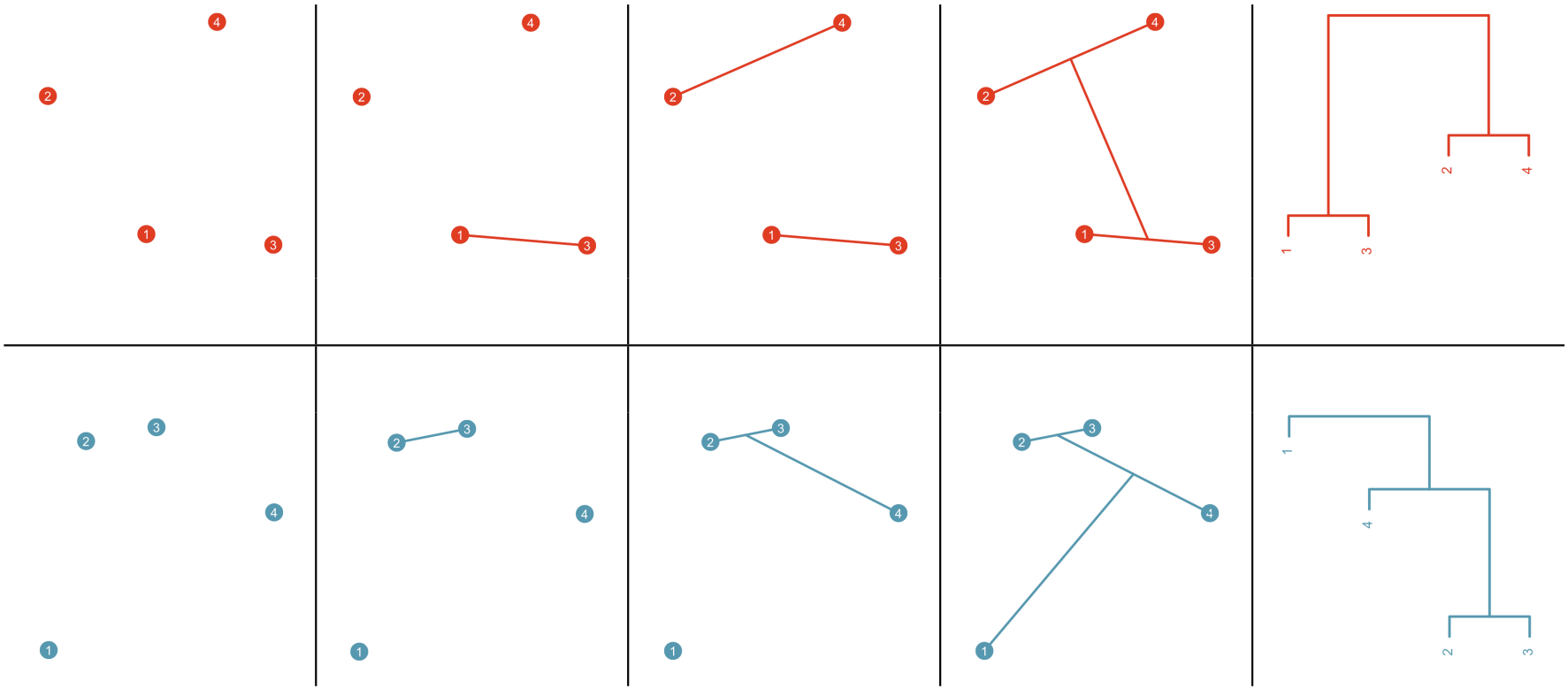
- Une hiérarchie $\mathcal{H}$ de N est un ensemble $\{C_1, \ldots, C_K\}$ de groupes non-vides tels que
  - $N \in \mathcal{H}$;
  - $\forall i \in N$, $\{i\} \in \mathcal{H}$; et
  - $\forall A, B \in N$, $A \cap B \in \{A, B, \emptyset\}$.

Example: On considère $N = \{1, 2, 3, 4, 5, 6, 7\}$ et

$$\mathcal{H} = \{\{1\}, \ldots, \{7\}, \{4, 5\}, \{2, 3\}, \{4, 5, 6\}, \{1, 2, 3\}, \{4, 5, 6, 7\}, N\}.$$

Vérifier qu'il s'agit d'une hiérarchie.

Regroupement Hiérarchique



Inertie

- L'inertie permet de décrire à l'aide d'un seul nombre l'hétérogénéité d'un groupe de points.
- Soit $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_p \end{bmatrix} \in \mathbb{R}^p$ un nuage dont chacun des points est caractérisé par un poids p_i, $i = 1, \ldots, n$ tels que $\sum_{i=1}^{n} p_i = 1$.
- Généralement, on a $p_i = 1/n$, $i = 1, \ldots, n$.

Inertie

- Le centre de gravité d'un nuage de points est défini par

$$g = \sum_{i=1}^{n} p_i \mathbf{x}_i \ = \ \overline{x}.$$

- L'inertie totale d'un nuage de point est donnée par

$$I_T = \sum_{i=1}^{n} p_i d^2(\mathbf{x}_i, g),$$

où $d()$ est la distance euclidienne.

Inertie

- Dans une procédure de classification, le nuage de points sera partitionné en K groupes $C_1, \ldots, C_K$ dont les poids respectifs sont donnés par P_j tel que

$$P_j = \sum_{i:\mathbf{x}_i \in C_j} p_i.$$

- Le centre de gravité de chacun des groupes est donné par

$$g_j = \frac{1}{P_j} \sum_{i:\mathbf{x}_i \in C_j} p_i \mathbf{x}_i.$$

Inertie

- L'inertie entre les groupes (between clusters inertia) est donnée par

$$I_B = \sum_{j=1}^{K} P_j d^2(g_j, g).$$

- L'inertie interne des groupes (within clusters inertia) est donnée par

$$I_W = \sum_{j=1}^{K} P_j I_{C_j}, \qquad I_{C_j} = \frac{1}{P_j} \sum_{i:\mathbf{x}_i \in C_j} p_i d^2(\mathbf{x}_i, g_j).$$

- Le théorème de König-Huygens indique que

$$I_T = I_B + I_W,$$

c'est-à-dire que l'inertie total d'un jeu de données est constante.

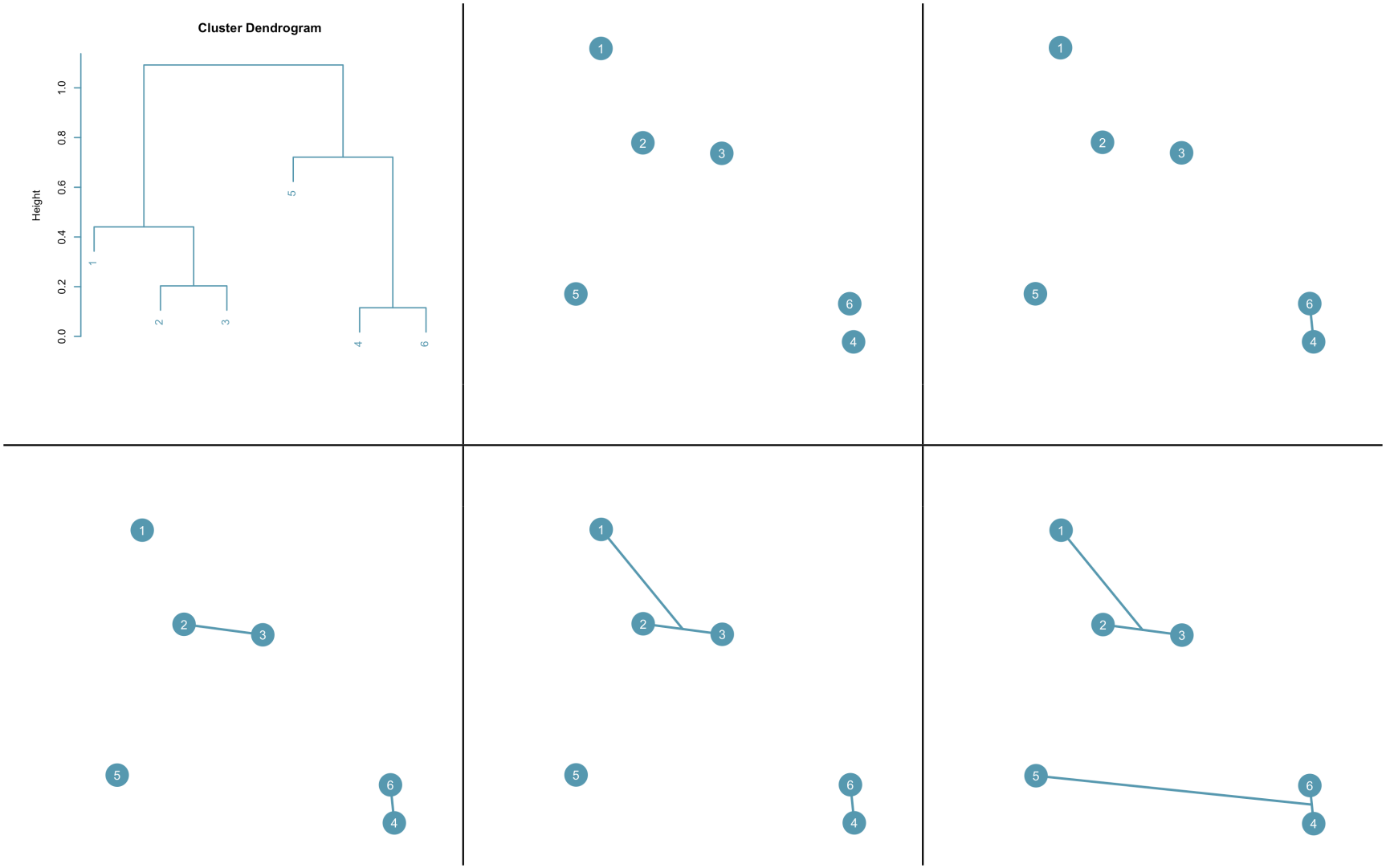
Inertie

- Une “bonne” classification aura ainsi une inertie inter-groupe élevée et une inertie intra-groupe faible.
- Minimiser l’intertie intra-groupe est équivalent à maximiser l’inertie inter-groupe (puisque l’inertie totale est constante).
- La proportion de l’inertie totale expliquée par la partition $\mathcal{P}_K$ est

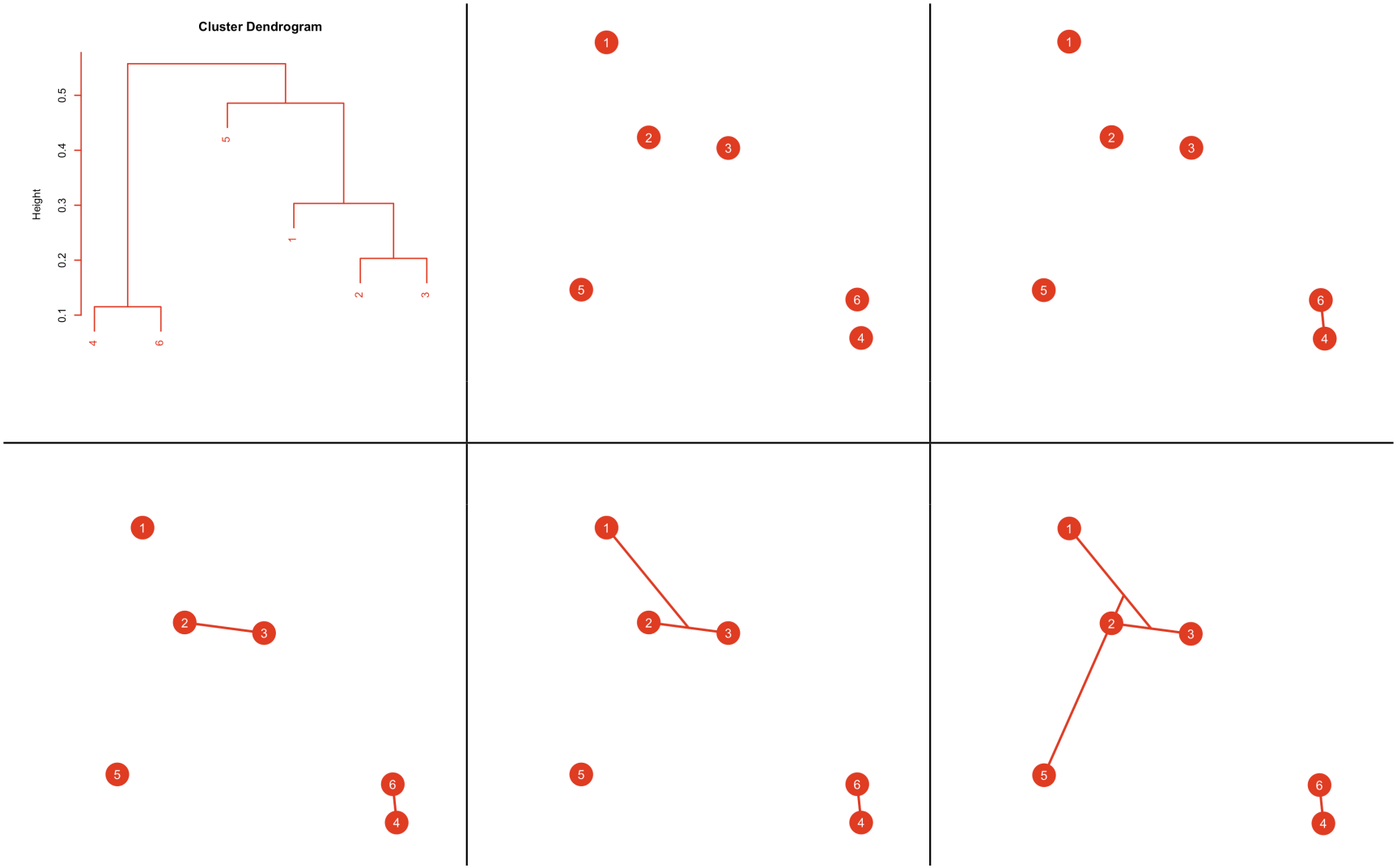
$$\frac{I_B}{I_T} = \left(1 - \frac{I_W}{I_T}\right).$$

- Cette valeur augmente lorsque le nombre de groupe(s) K augmente: elle ne peut être utilisée que pour comparer des partitions avec le même nombre de groupes.

Exemple (complete)



Exemple (single)



Algorithmes hiérarchiques ascendants

- La procédure hiérarchique ascendante implique de débuter l'analyse avec le plus grand nombre possible de groupes (cluster) (généralement en prenant les données individuellement).
- À chacune des étapes, on fusionne les deux groupes les plus semblables (les plus proches).
- On poursuite la procédure jusqu'à terminer l'analyse avec le moins de groupes possibles (généralement un seul groupe contenant toutes les données).
- On analyse l'arbre de classification (ou dendrogramme) ainsi obtenu et on choisit le nombre de classes K à conserver.

$\rightarrow$ Demande de prendre une décision supplémentaire: comment définit-on la similitude entre deux groupes?

Algorithmes hiérarchiques ascendants

La similitude entre deux groupes peut être déterminée de plusieurs façons.

- Lien simple (simple linkage): la proximité entre deux groupes (A et B) est donnée par

$$\delta(A, B) = \min_{a_i \in A, b_j \in B} (d(a_i, b_j)),$$

  c'est-à-dire qu'on utilise le minimum de la distance entre chacun des point du groupe A et chacun des points du groupe B.

Algorithmes hiérarchiques ascendants

- Lien complet (complete linkage): la proximité entre deux groupes (A et B) est donnée par

$$\delta(A, B) = \max_{a_i \in A, b_j \in B} (d(a_i, b_j)),$$

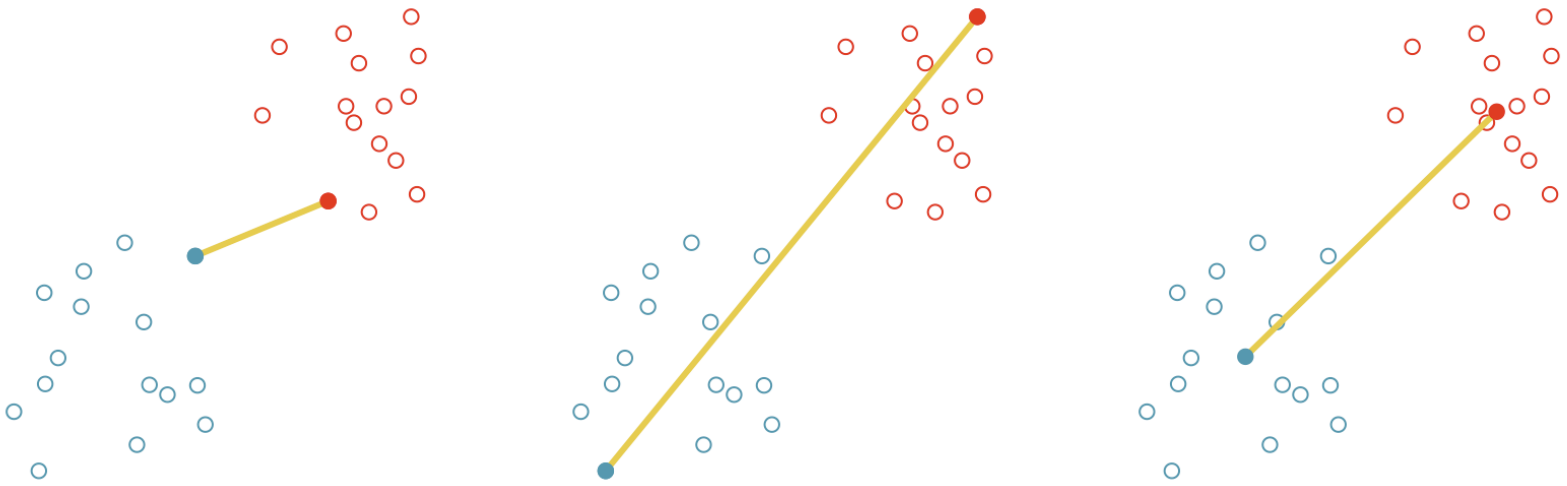
c'est-à-dire qu'on utilise le maximum de la distance entre chacun des point du groupe A et chacun des points du groupe B.

Algorithmes hiérarchiques ascendants

- Lien moyen (average linkage): la proximité entre deux groupes (A et B) est donnée par

$$\delta(A,B) = \frac{1}{\text{card}(A)\text{card}(B)} \sum_{a_i \in A} \sum_{b_j \in B} (d(a_i, b_j)).$$

Cette fois, on utilise la moyenne des distances entre chacun des points du groupe A et chacun des points du groupe B.



Algorithmes hiérarchiques ascendants

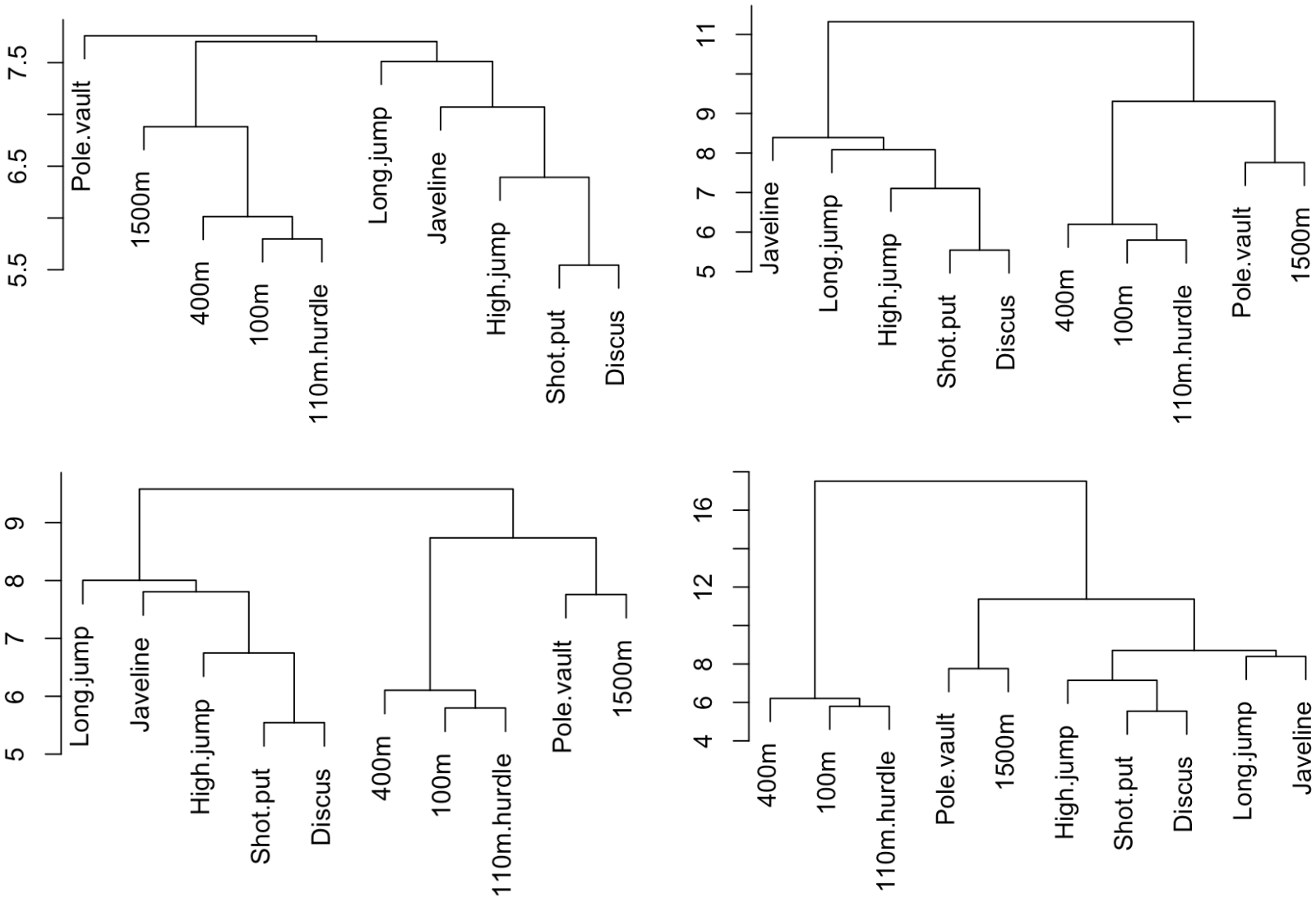
- Algorithme de Ward: on décompose l'inertie du nuage de points et minimise la perte d'information (perte d'inertie entre les groupes I_B) à chaque étape: on optimise donc le même critère que pour un algorithme de classification par ré-allocation dynamique.
- À chacune des étapes, la perte d'information due au regroupement des groupes A et B est donnée par

$$\delta(A,B) = \frac{P_A P_B}{P_A + P_B} d^2(g_A, g_B).$$

Également, on définit la perte d'information due au regroupement des groupes $C = A \cup B$ et D par

$$\delta(C,D) = \frac{(P_A + P_D)\delta(A,D) + (P_B + P_D)\delta(B,D) - P_D\delta(A,B)}{P_A + P_B + P_D}.$$

Algorithmes hiérarchiques ascendants



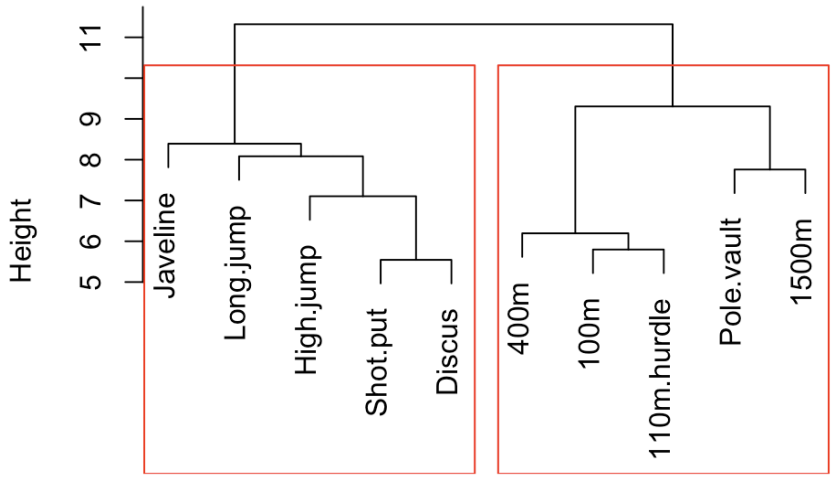
Algorithmes hiérarchiques ascendants

Ainsi, avant de pouvoir utiliser un algorithme hiérarchique ascendant, il faut répondre aux deux questions suivantes:

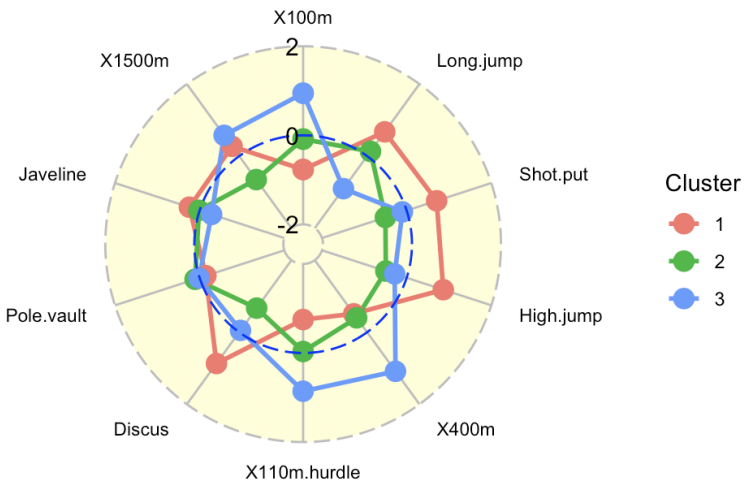
- Quelle règle de ressemblance, de proximité, entre les observations choisir? $\rightarrow$ Choix de la mesure de distance
- Comment définit-on la similitude entre deux groupes? $\rightarrow$ Choix de la méthode d'aggrégation.

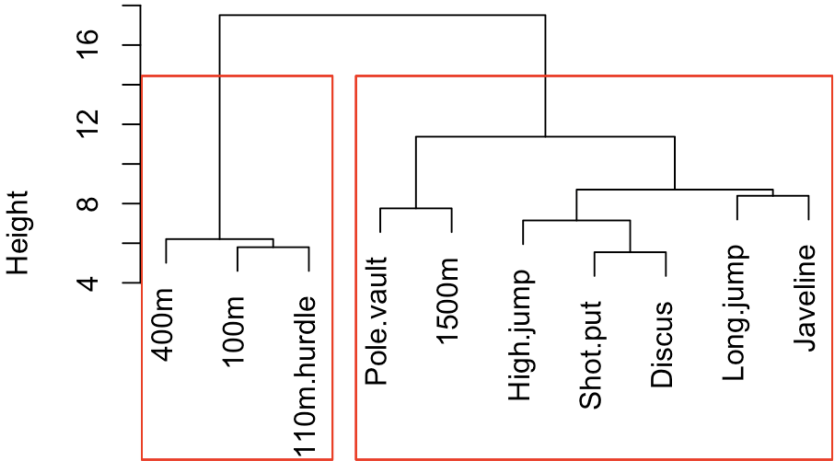
Decathlon

One can look at clusters of sports (Ward and complete)



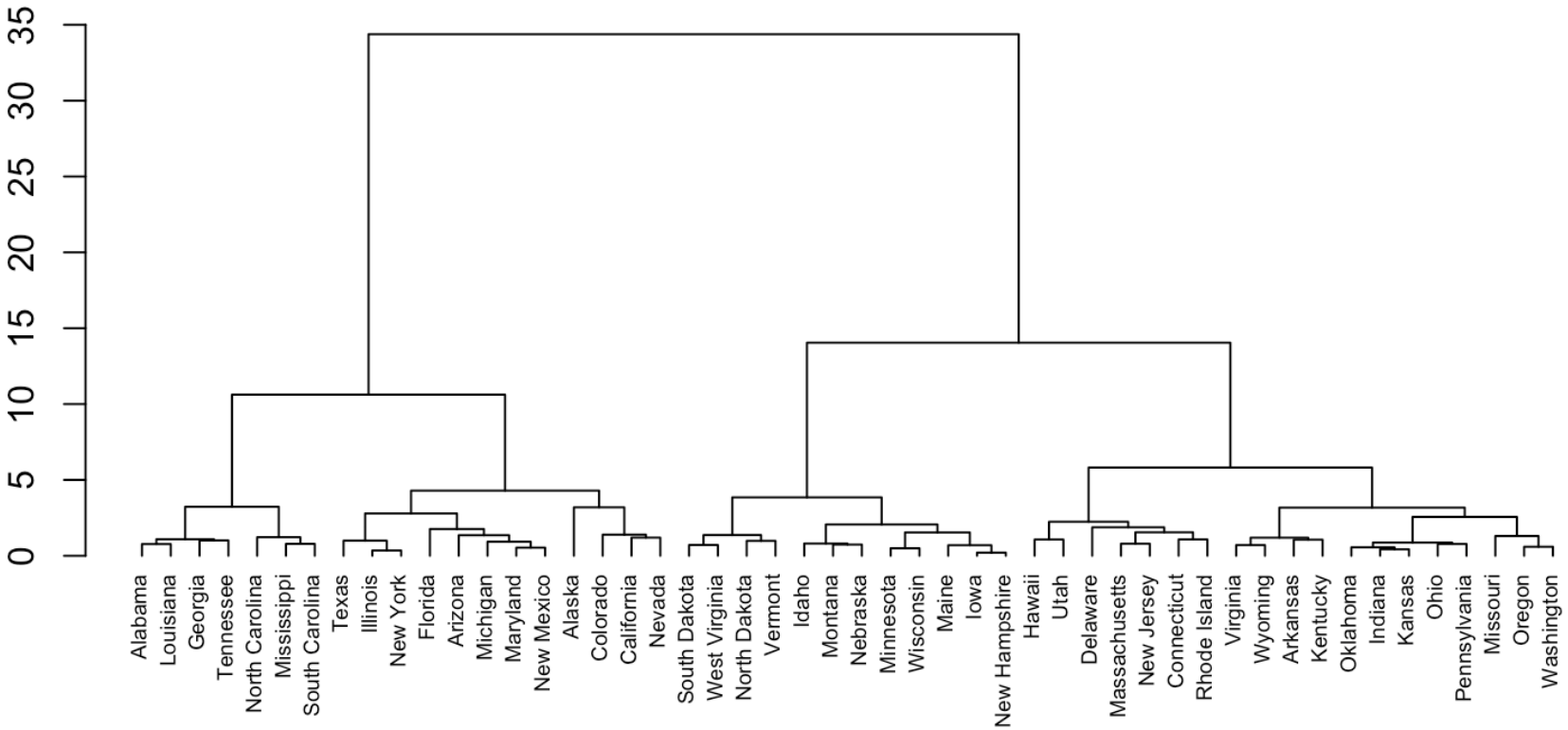
or clusters of sportmen





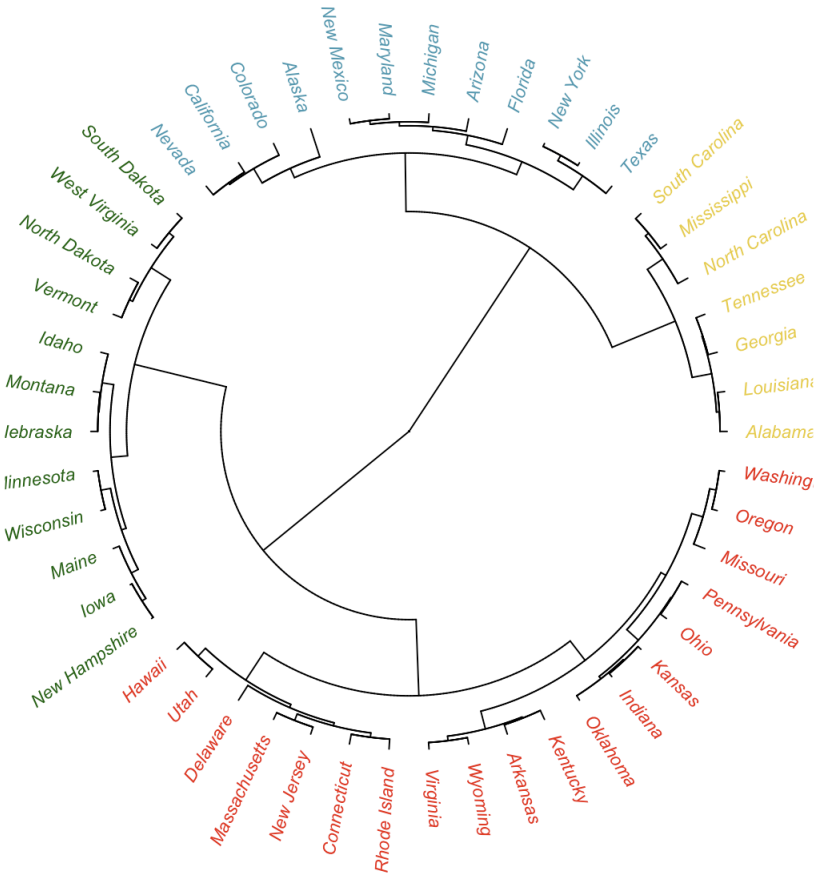
Comparing US States

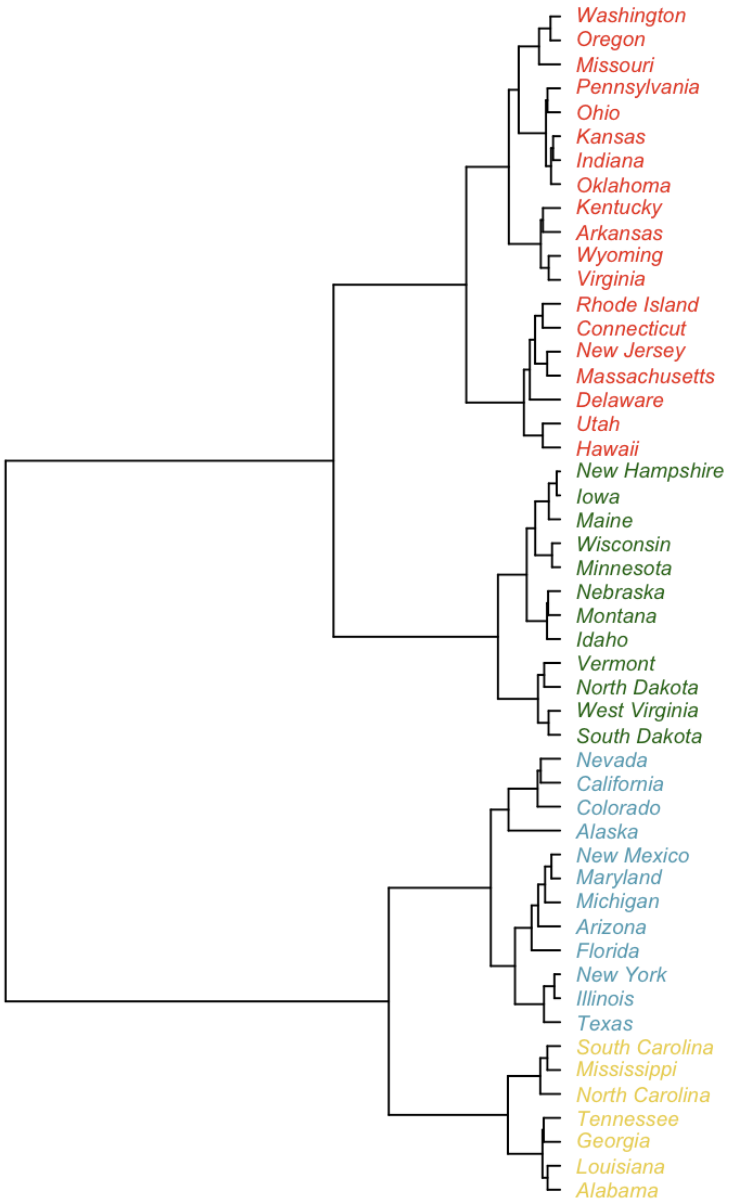
Using Ward's method,



Comparing US States

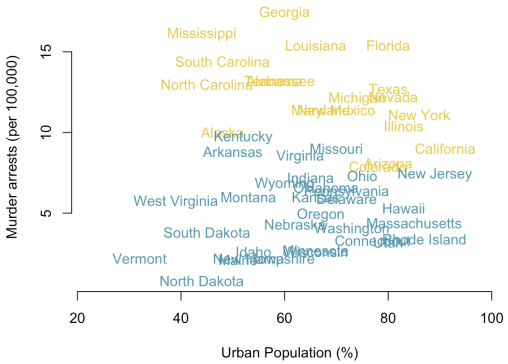
One can use a phylogenetic tree

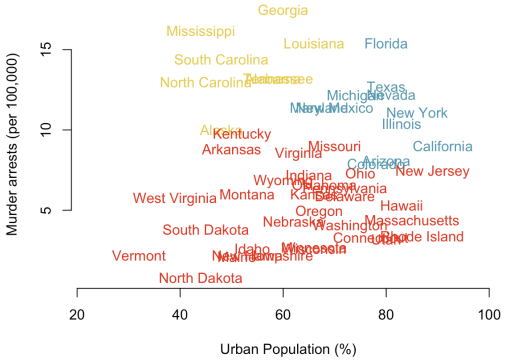


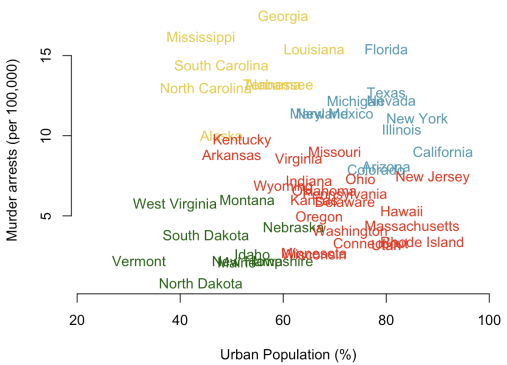


Comparing US States

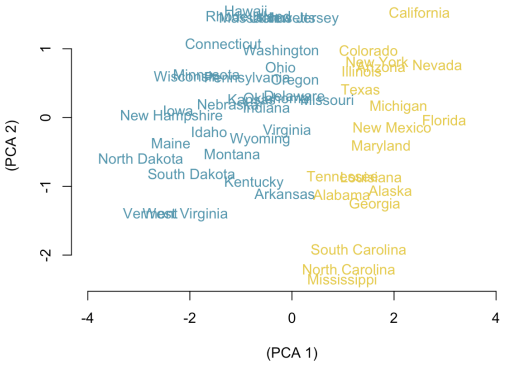
On the two dimensional representation (urban population, murder rate), when obtain, as optimal 2, 3 or 4 classes

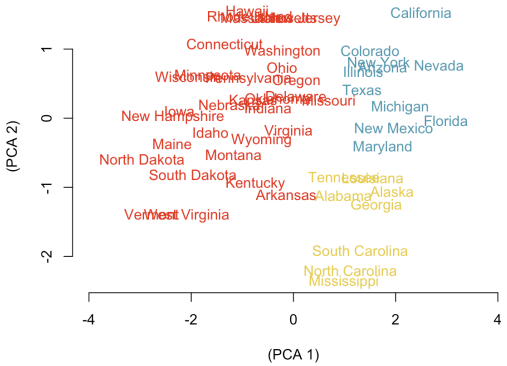


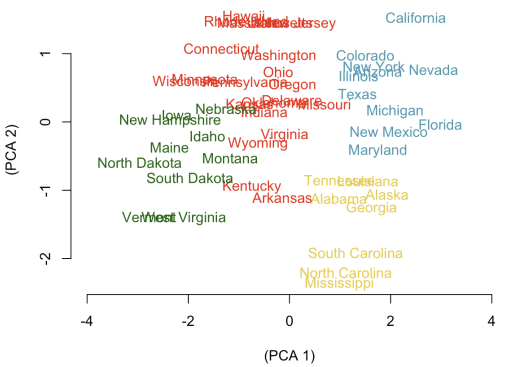




or on the first 2 principal components (PCA projection)

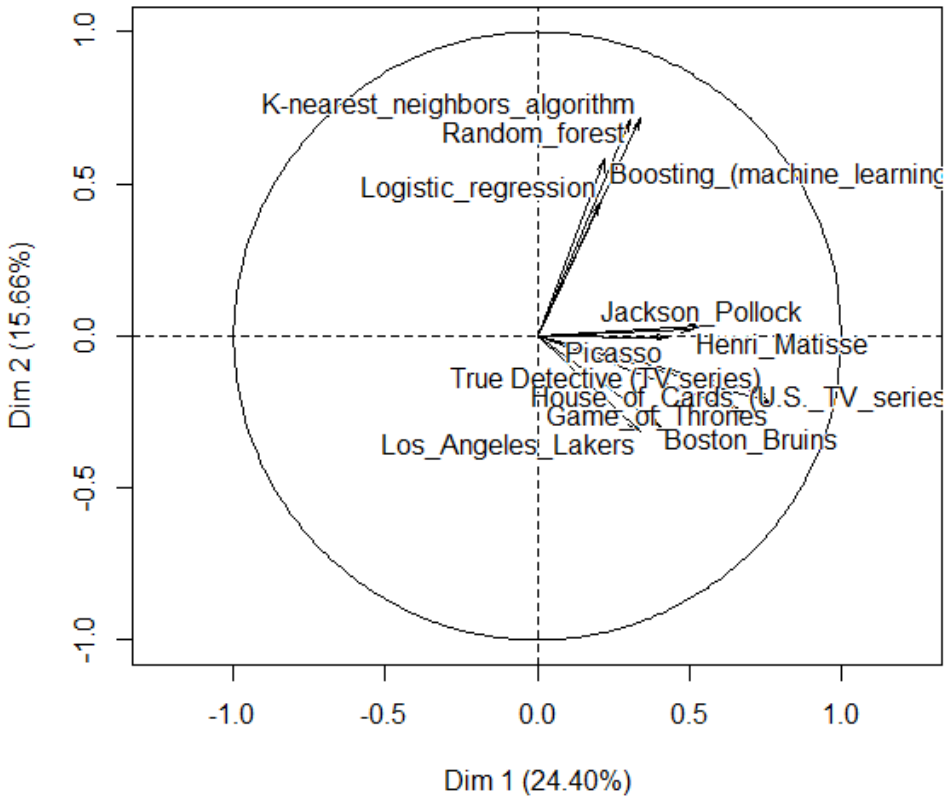


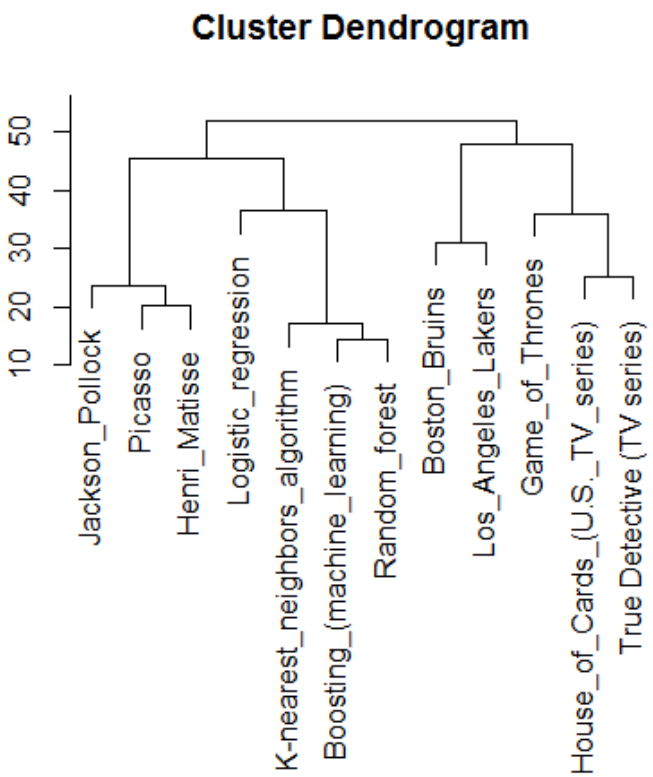




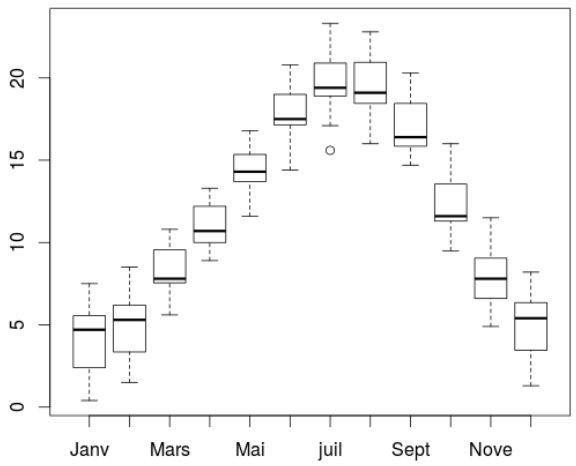
Comparing Wikipedia Pages

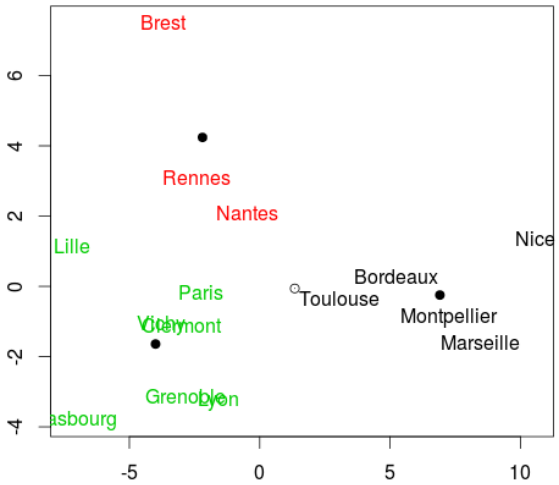
Bag of works from some pages Boosting_(machine_learning), Random_forest, K-nearest_neighbors, Logistic_regression, Boston_Bruins, Los_Angeles_Lakers, Game_of_Thrones, House_of_Cards, True_Detective, Picasso, Henri_Matisse, Jackson_Pollock.

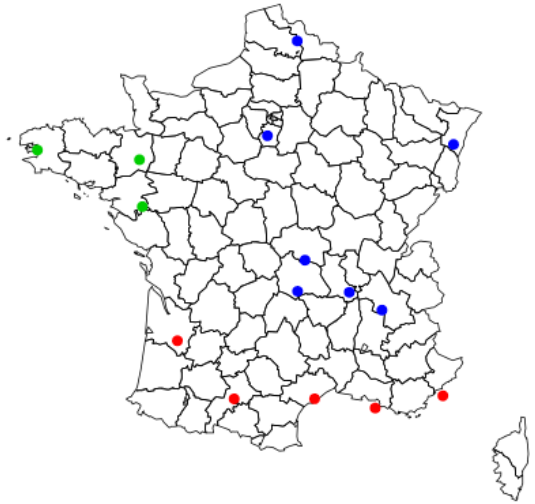


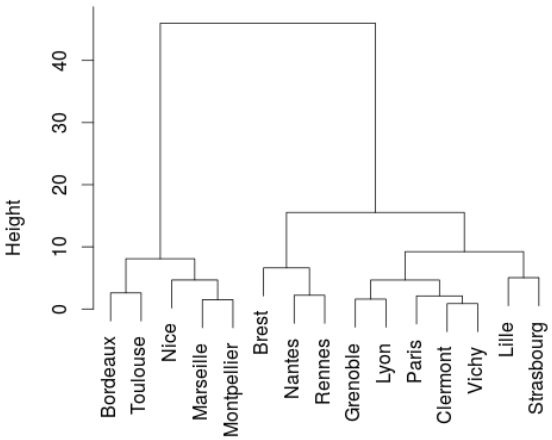


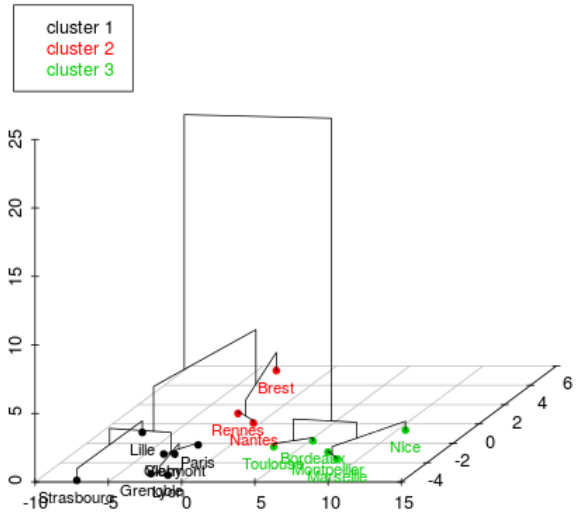
Comparing 15 Cities Temperatures (in France)

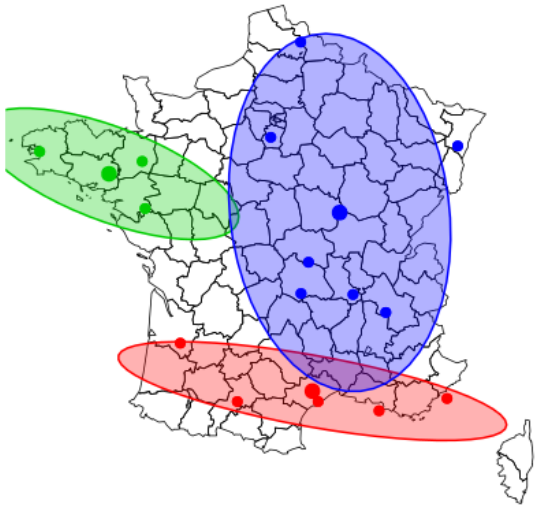






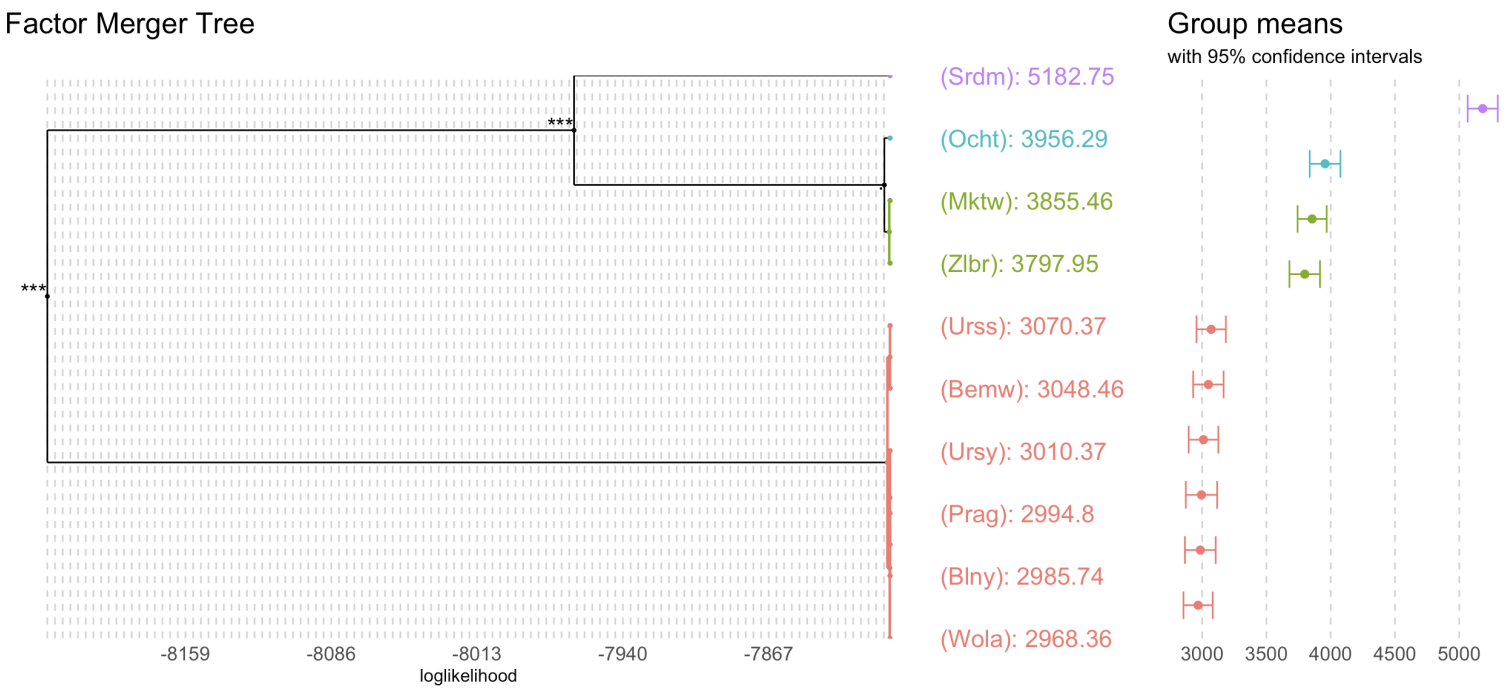






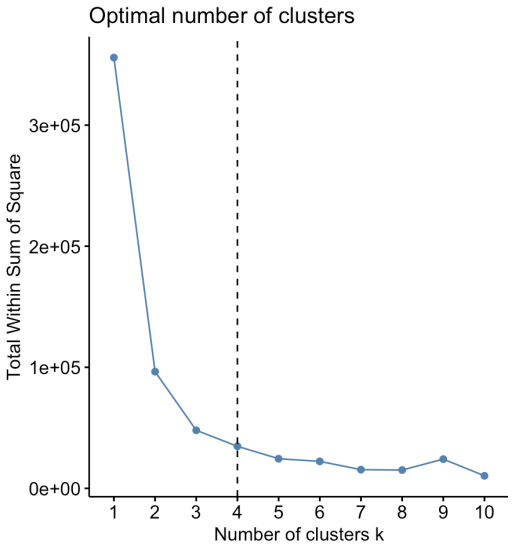
Clusters of Factors

```
> library(factorMerger)
> library(DALEX)
> data(apartments)
> MF = mergeFactors(response = apartments$m2.price,
+ factor = apartments$district,family = "gaussian")
> plot(MF)
```

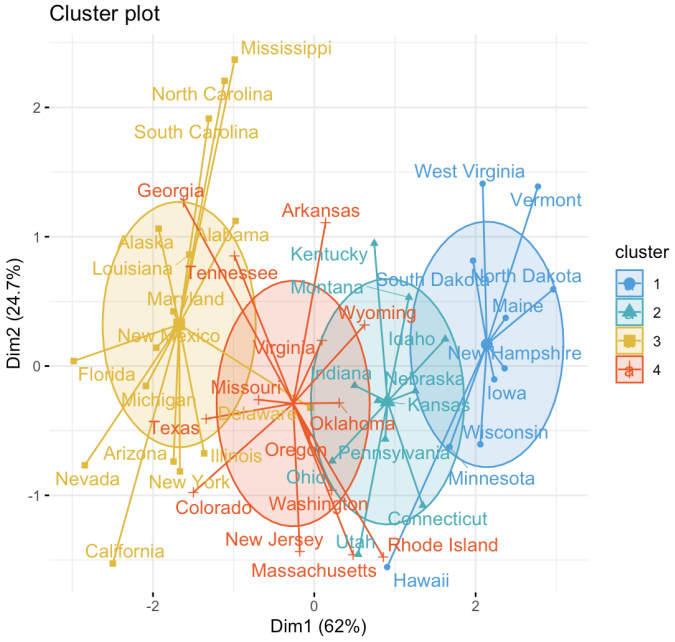


Clusters with R

```
> library(factoextra)
> fviz_nbclust(USArrests, kmeans, method = "wss") +
  geom_vline(xintercept = 4, linetype = 2)
```

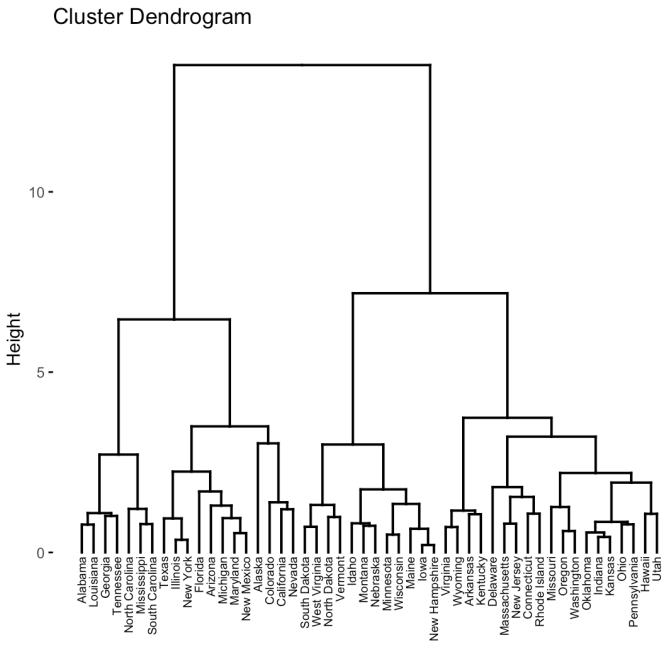



```
> km.res = kmeans(df, 4, nstart = 25)
> fviz_cluster(km.res, data = USArrests, ellipse.type = "euclid", star.plot = TRUE, repel = TRUE, ggtheme = theme_minimal())
```

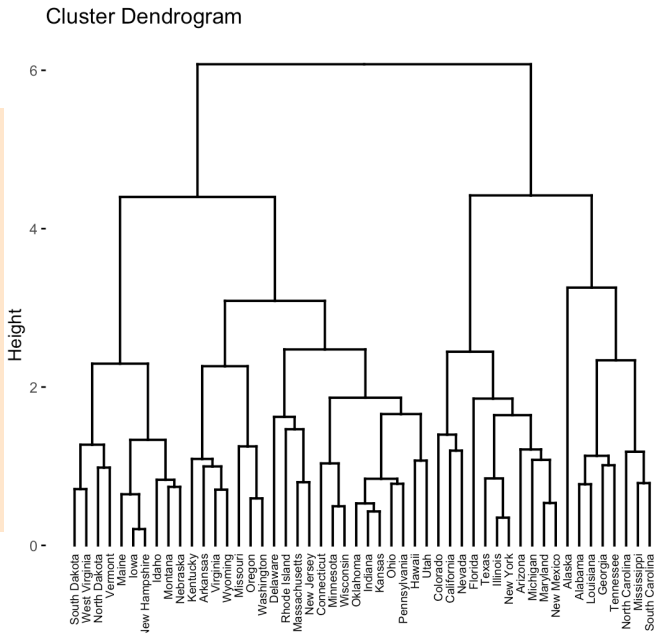


Clusters with R

```
> df = scale(USArrests)
> res.dist = dist(df, method = "euclidean")
> res.hc = hclust(d = res.dist, method = "ward.D2")
> library("factoextra")
> fviz_dend(res.hc, cex = 0.5)
```

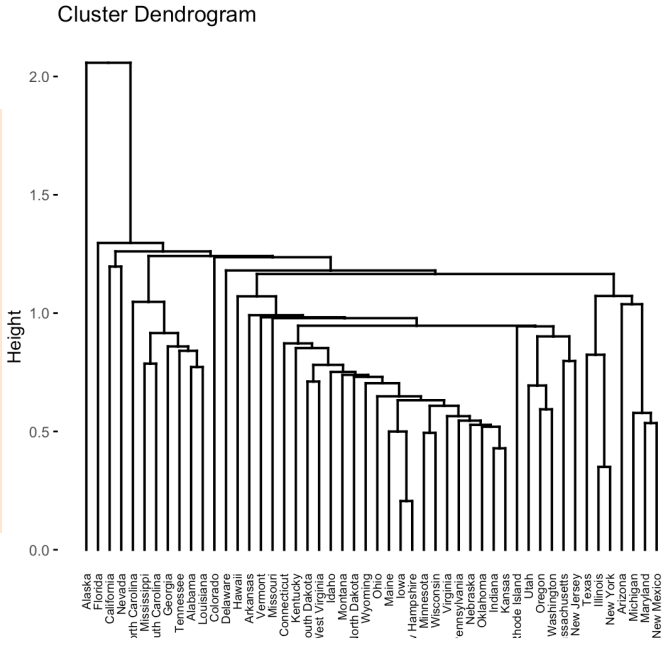



```
> df = scale(USArrests)
> res.dist = dist(df, method = "euclidean")
> res.hc = hclust(d = res.dist, method = "complete")
> library("factoextra")
> fviz_dend(res.hc, cex = 0.5)
```

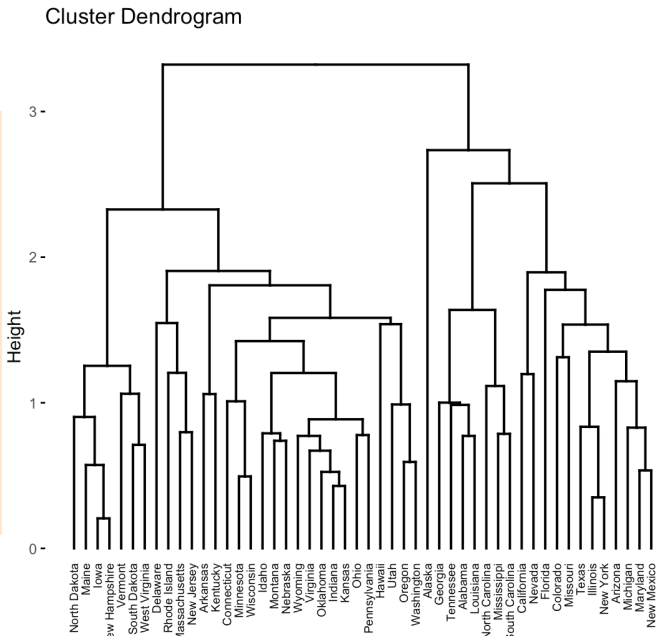


Clusters with R

```
> df = scale(USArrests)
> res.dist = dist(df, method = "euclidean")
> res.hc = hclust(d = res.dist, method = "single")
> library("factoextra")
> fviz_dend(res.hc, cex = 0.5)
```

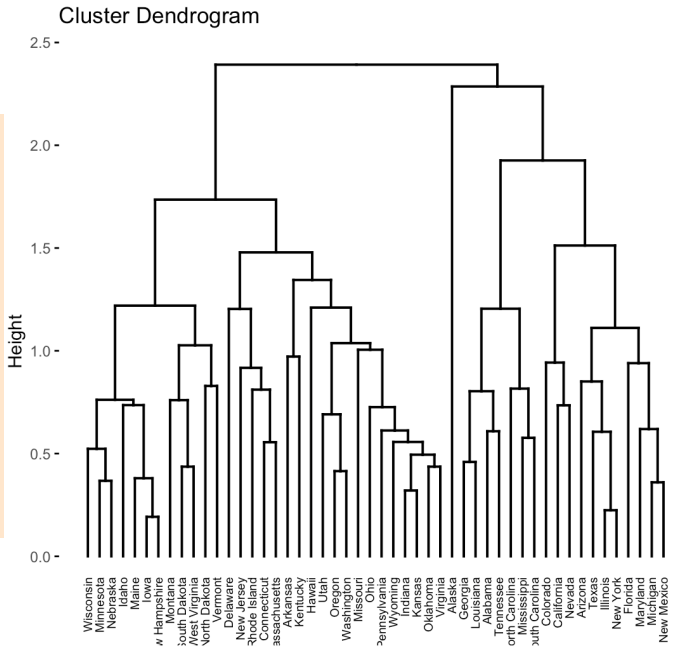



```
> df = scale(USArrests)
> res.dist = dist(df, method = "euclidean")
> res.hc = hclust(d = res.dist, method = "average")
> library("factoextra")
> fviz_dend(res.hc, cex = 0.5)
```

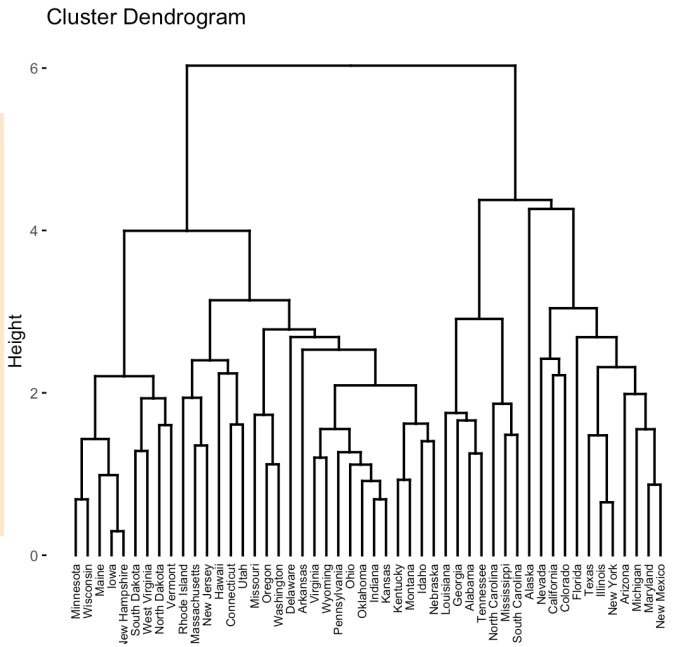


Clusters with R

```
> df = scale(USArrests)
> res.dist = dist(df, method = "
    manhattan")
> res.hc = hclust(d = res.dist,
    method = "average")
> library("factoextra")
> fviz_dend(res.hc, cex = 0.5)
```

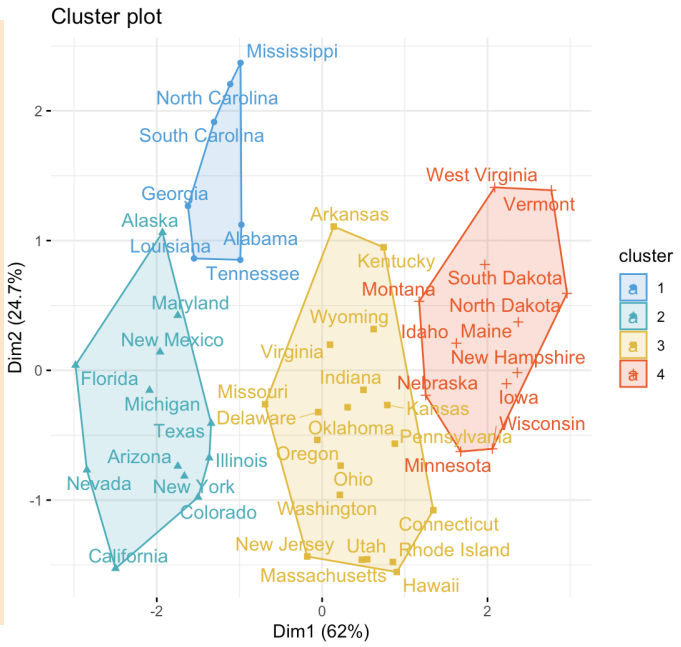



```
> df = scale(USArrests)
> res.dist = dist(df, method = "
    maximum")
> res.hc = hclust(d = res.dist,
    method = "average")
> library("factoextra")
> fviz_dend(res.hc, cex = 0.5)
```

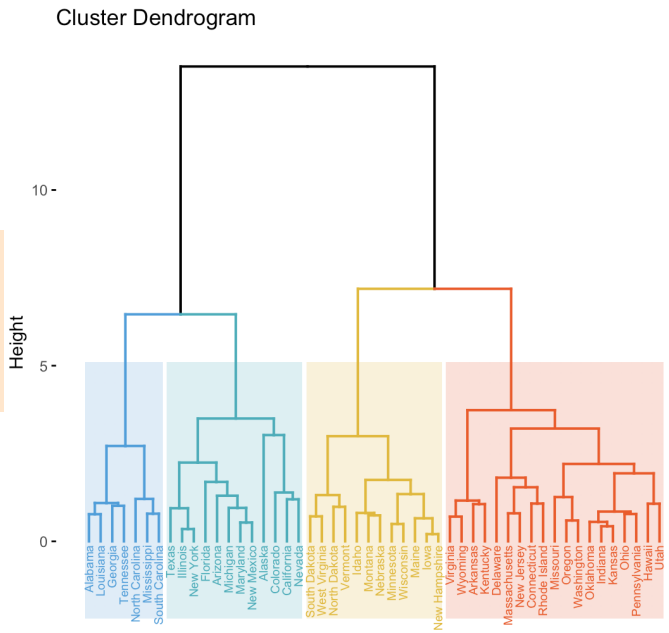


Clusters with R

```
> res.dist = dist(df, method = "euclidean")
> res.hc = hclust(d = res.dist, method = "ward.D2")
> grp = cutree(res.hc, k = 4)
> fviz_cluster(list(data = df, cluster = grp), ellipse.type = "convex", show.clust.cent = FALSE, ggtheme = theme_minimal())
```

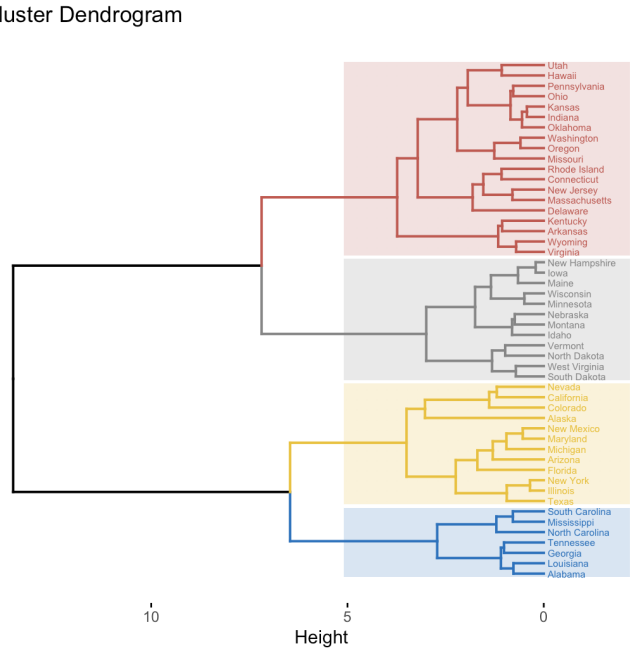



```
> fviz_dend(res.hc, k = 4, cex = 0.5, color_labels_by_k = TRUE, rect = TRUE, rect_fill = TRUE)
```

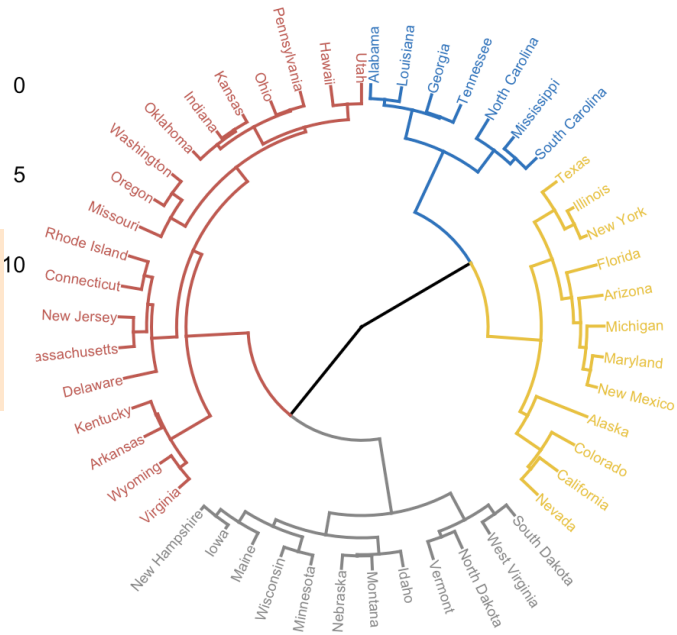


Clusters with R

```
> fviz_dend(res.hc, k = 4, cex =
    0.4, horiz = TRUE, k_colors = "
    jco", rect = TRUE, rect_border
    = "jco", rect_fill = TRUE)
```

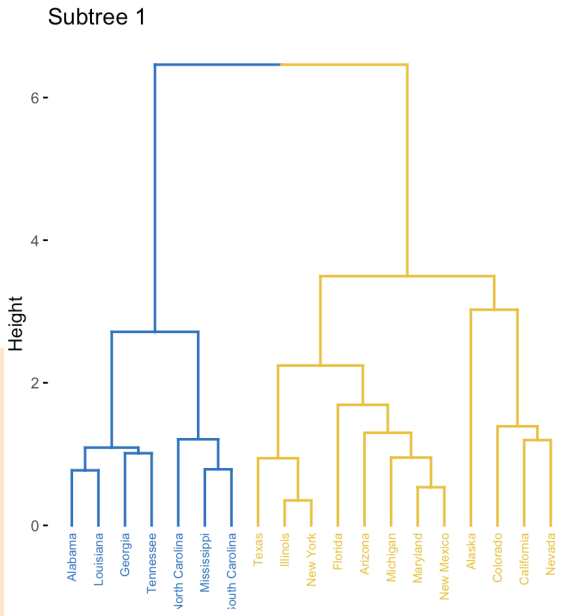


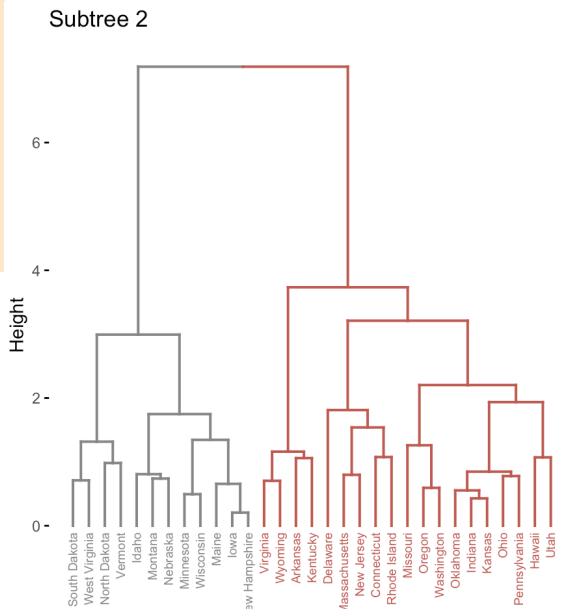

```
> fviz_dend(hc, cex = 0.5, k = 4,
    k_colors = "jco", type = "circular
    ")
```

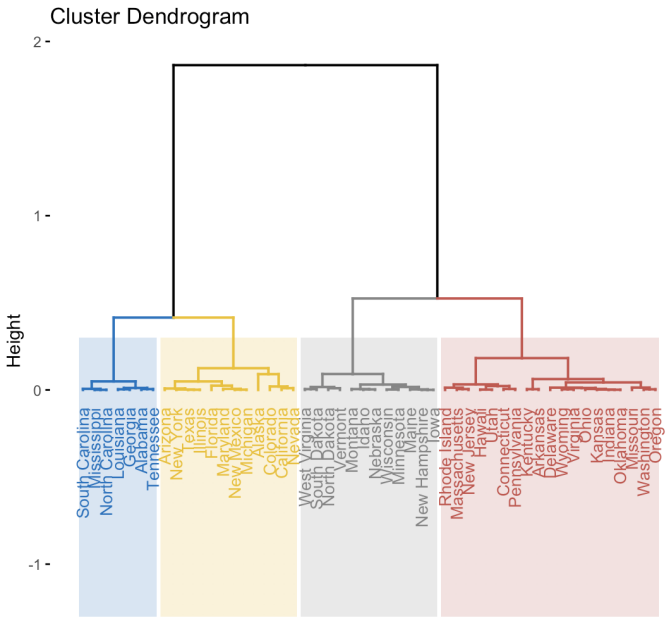


Clusters with R

```
> dend_plot = fviz_dend(hc, k = 4, cex = 0.5, k_colors = "jco")
> dend_data = attr(dend_plot, "dendrogram")
> dend_cuts = cut(dend_data, h = 10)
> fviz_dend(dend_cuts$lower[[1]], main = "Subtree 1")
> fviz_dend(dend_cuts$lower[[2]], main = "Subtree 2")
```





Clusters with R

```
> res.hcpc = HCPC(res.pca, graph =
    FALSE)
> fviz_dend(res.hcpc, cex = 0.7,
    palette = "jco", rect = TRUE,
    rect_fill = TRUE, rect_border
    = "jco", labels_track_height =
    0.8)
```