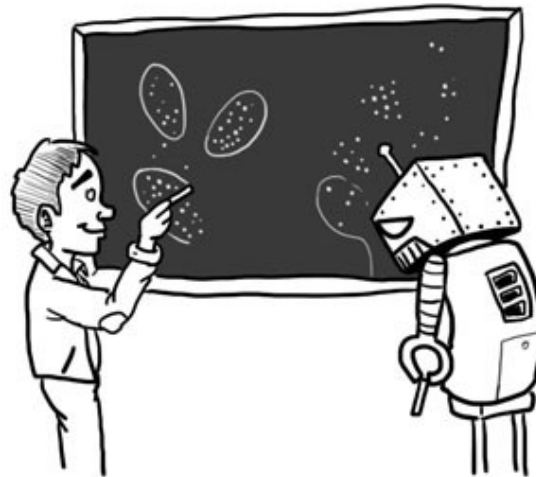


# Data-Science 1

## clusteranalyse



---

# Inhoud

- n-dimensionale ruimten
- afstanden
- clusters zoeken
  - k-means
  - hiërarchisch
- combinatie met beslissingsbomen

---

# n-dimensionale ruimten

# Voorbeeld

- gegeven: tabel met punten van studenten

vak1	vak2	vak3	vak4	vak5	vak6	vak7	vak8	vak9	vak10
10	14	11	16	15	13	9	18	14	13
16	15	18	19	16	16	15	14	17	18
...	...	...	...	...	...	...	...	...	...
8	6	9	10	14	5	0	5	1	0

- gevraagd: kunnen we de studenten indelen in bepaalde "types" (clusters)
- dit is een voorbeeld van "unsupervised learning"
- opmerking: welk meetniveau hebben de kolommen?

# n-dimensionale ruimten

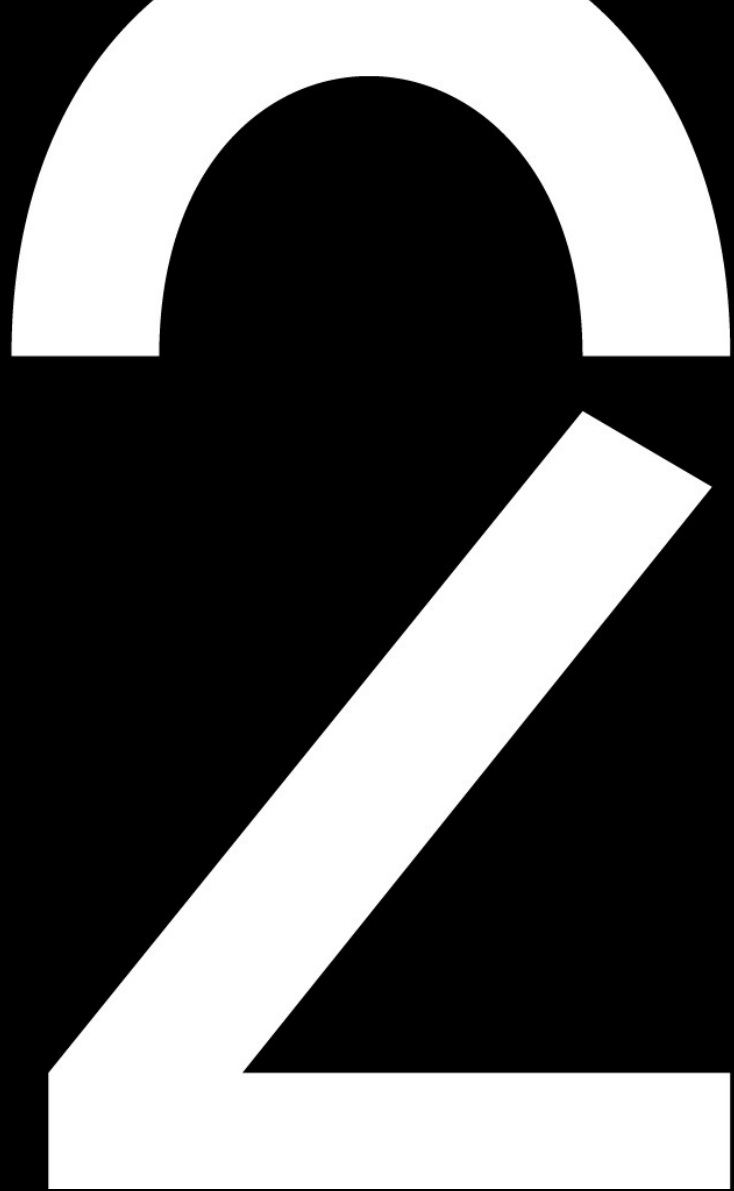
- wat als er maar 2 kolommen waren?
  - eerste kolom hernoem je "x"
  - tweede kolom hernoem je "y"
  - beide kolommen hebben minstens interval meetniveau
  - wat is iedere rij dan? (zie ook correlatie/regressie)
- wat als er 3 kolommen zijn?
- wat als er n kolommen zijn?

# Clusters

- werkt enkel als alle variabelen minstens interval meetniveau hebben
- rij = punt in n-dimensionale ruimte
- een cluster is een aantal rijen van een tabel die "bij elkaar horen" of "gelijkaardig zijn"
- 2 rijen zijn gelijkaardig als de punten dicht bij elkaar liggen
  - de "afstand" moet klein zijn

---

Afstanden



# Hoe meet je de afstand?

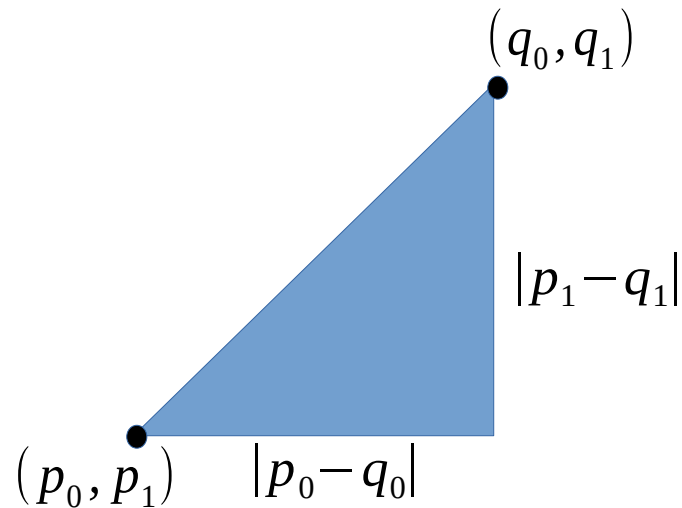
- gebruik een "metriek":
  - Euclidisch
  - Manhattan (taxi)
  - Chebychev
  - Minkowski
  - Mahalanobis
  - ...
- notatie: ieder punt heeft coördinaten

$$p = (p_0, p_1, p_2, \dots, p_{n-1})$$



# Euclidische afstand

- stelling van Pythagoras



- 2D:  $d(p, q) = \sqrt{(p_0 - q_0)^2 + (p_1 - q_1)^2}$
- 3D:  $d(p, q) = \sqrt{(p_0 - q_0)^2 + (p_1 - q_1)^2 + (p_2 - q_2)^2}$
- n-D:  $d(p, q) = \sqrt{\sum_{i=0}^{n-1} (p_i - q_i)^2}$

# Euclidische afstand

- voorbeeld: 2 rijen uit een tabel:

10	12	15	13	9
18	14	13	15	17

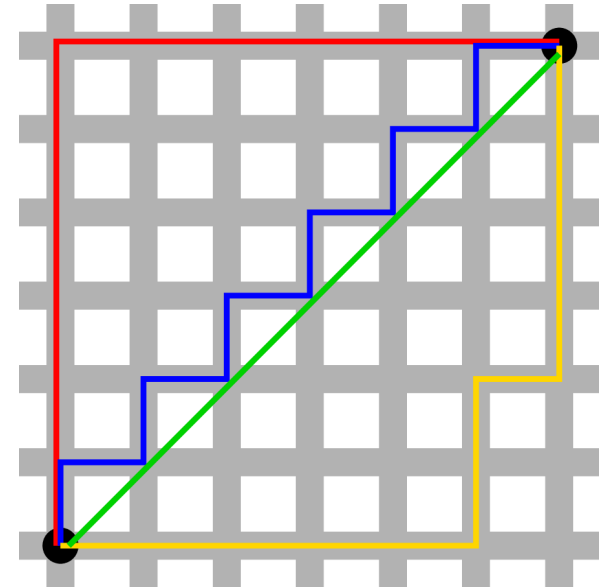
- hoeveel dimensies?
- wat is de afstand?

$$\begin{aligned} d(p, q) &= \sqrt{(10-18)^2 + (12-14)^2 + (15-13)^2 + (13-15)^2 + (9-17)^2} \\ &= \sqrt{8^2 + 2^2 + 2^2 + 2^2 + 8^2} = \sqrt{140} = 11,832 \end{aligned}$$

# Manhattan (taxi) afstand

- je afstanden ook anders meten
- voorbeeld: kortste afstand in Manhattan

- $$d(p, q) = \sum_{i=1}^n |p_i - q_i|$$



# Manhattan (taxi) afstand

- voorbeeld: 2 rijen uit een tabel:

10	12	15	13	9
18	14	13	15	17

- wat is de afstand?

$$\begin{aligned}d(p, q) &= |10 - 18| + |12 - 14| + |15 - 13| + |13 - 15| + |9 - 17| \\ &= 8 + 2 + 2 + 2 + 8 = 22\end{aligned}$$

# Gestandaardiseerde afstand

- probleem: waarden in kolommen hebben soms compleet andere grootorde
- voorbeeld: kolommen "leeftijd" en "km per jaar gereden"
  - "km per jaar" zal een grotere invloed hebben
- oplossing: schalen
  - zet iedere kolom eerst om naar Z-scores

# Metrieken algemeen

- een metriek moet volgende eigenschappen hebben:
  - afstand van een punt tot zichzelf is 0
  - afstand van  $x$  naar  $y$  is gelijk aan de afstand van  $y$  naar  $x$
  - een omweg mag niet korter zijn dan rechtstreeks

# Meetniveau's

- normaal: minstens interval meetniveau nodig
- wat als dit niet is?
  - ordinaal
    - gebruik de volgnummers (niet helemaal correct, maar kan bij clustering wel ok zijn omdat het gaat over “verder” en “meer dichtbij” wat wel bestaat op een ordinale schaal) (gebruik “Continueze”)
    - als er (disjuncte) klassen zijn: vervang ze door de klassenmiddens

# Meetniveau's

- nominaal
  - verwijderen...
  - binaire variabele: gebruik 1 en 2 of -1 en 1
  - woorden: word2vec
  - adressen: longitude/latitude
  - producten: plaats in winkel (rij, rek, hoogte)
  - kleur: R,G,B
  - ...



# Orange



	1	N	2	N	3	N	4	5
1	naam	haarlengte	gewicht	leeftijd	geslacht			
2	Homer	0	250	36	M			
3	Marge	10	150	34	V			
4	Bart	2	90	10	M			
5	Lisa	6	78	8	V			
6	Maggie	4	20	1	V			
7	Abe	1	170	70	M			
8	Selma	8	160	41	V			
9	Otto	10	180	38	M			
10	Krusty	6	200	45	M			

# Orange



Compare

☒ Rows ☐ Columns

Distance Metric

<input type="radio"/> Euclidean (normalized)	<input type="radio"/> Cosine
<input checked="" type="radio"/> Euclidean	<input type="radio"/> Pearson
<input type="radio"/> Manhattan (normalized)	<input type="radio"/> Pearson (absolute)
<input type="radio"/> Manhattan	<input type="radio"/> Spearman
<input type="radio"/> Mahalanobis	<input type="radio"/> Spearman (absolute)
<input type="radio"/> Hamming	<input type="radio"/> Jaccard

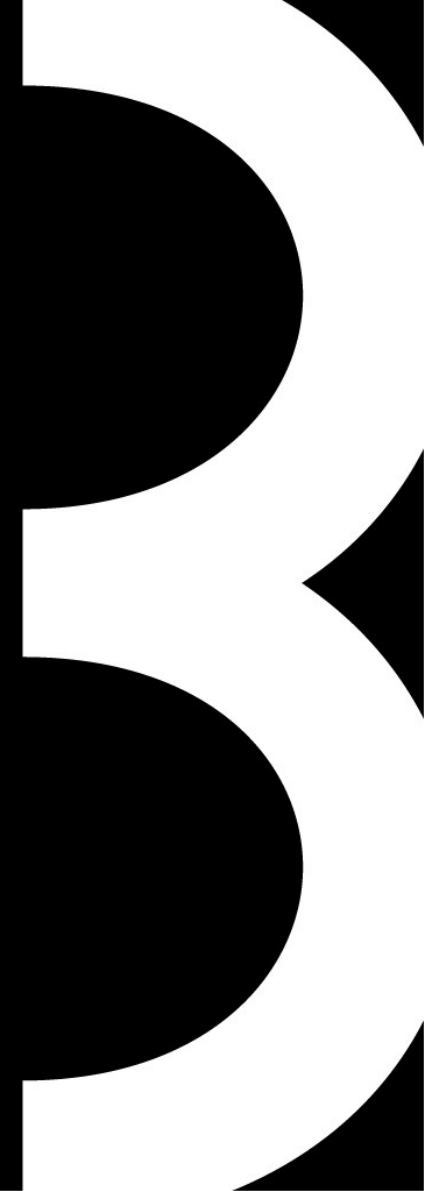
# Orange



	1	2	3	4	5	6	7	8	9
1		100,519	162,111	174,367	232,682	86,931	90,493	70,739	51,157
2	100,519		65,115	76,655	134,257	42,154	12,369	30,265	51,352
3	162,111	65,115		12,806	70,605	100,005	76,792	94,594	115,503
4	174,367	76,655	12,806		58,455	111,054	88,414	106,395	127,487
5	232,682	134,257	70,605	58,455		165,136	145,657	164,332	185,311
6	86,931	42,154	100,005	111,054	165,136		31,464	34,713	39,370
7	90,493	12,369	76,792	88,414	145,657	31,464		20,322	40,249
8	70,739	30,265	94,594	106,395	164,332	34,713	20,322		21,564
9	51,157	51,352	115,503	127,487	185,311	39,370	40,249	21,564	

---

# Clusters zoeken: k-means



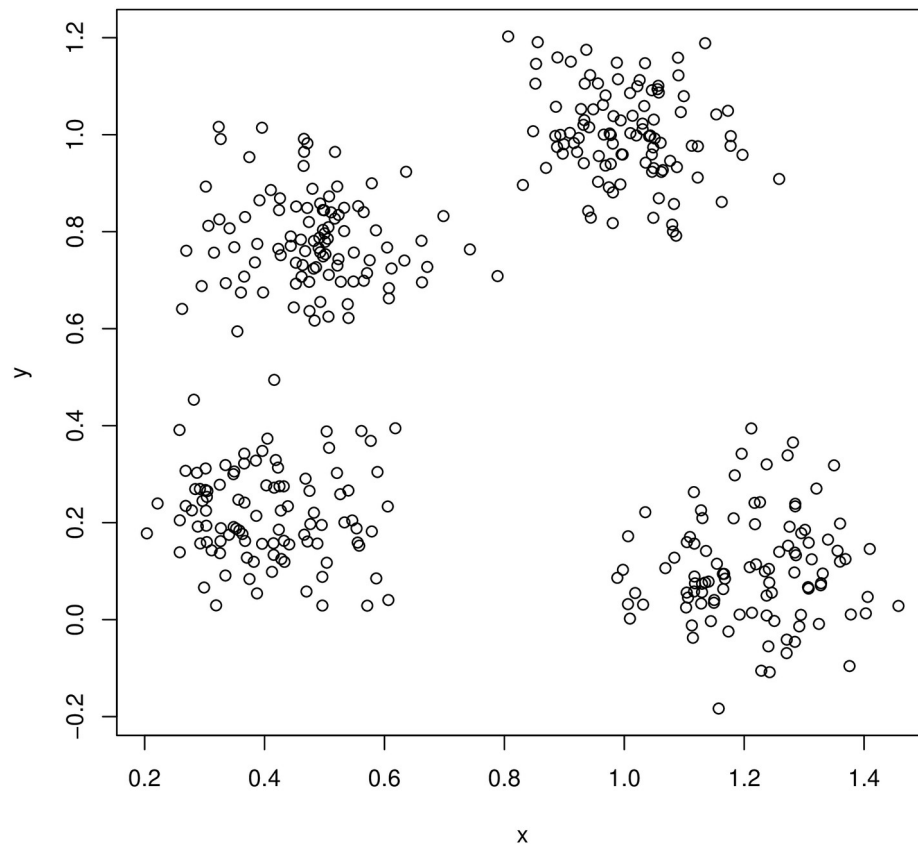
# K-means

- we zoeken  $n$  clusters ( $n$  is gegeven)
- meestal (gestandaardiseerde) euclidische afstand
- later op verder gewerkt door Kohonen (Self Organizing Maps - 1982)
  - werkt op basis van neurale netwerk
  - geïnspireerd door de werking van onze hersenen

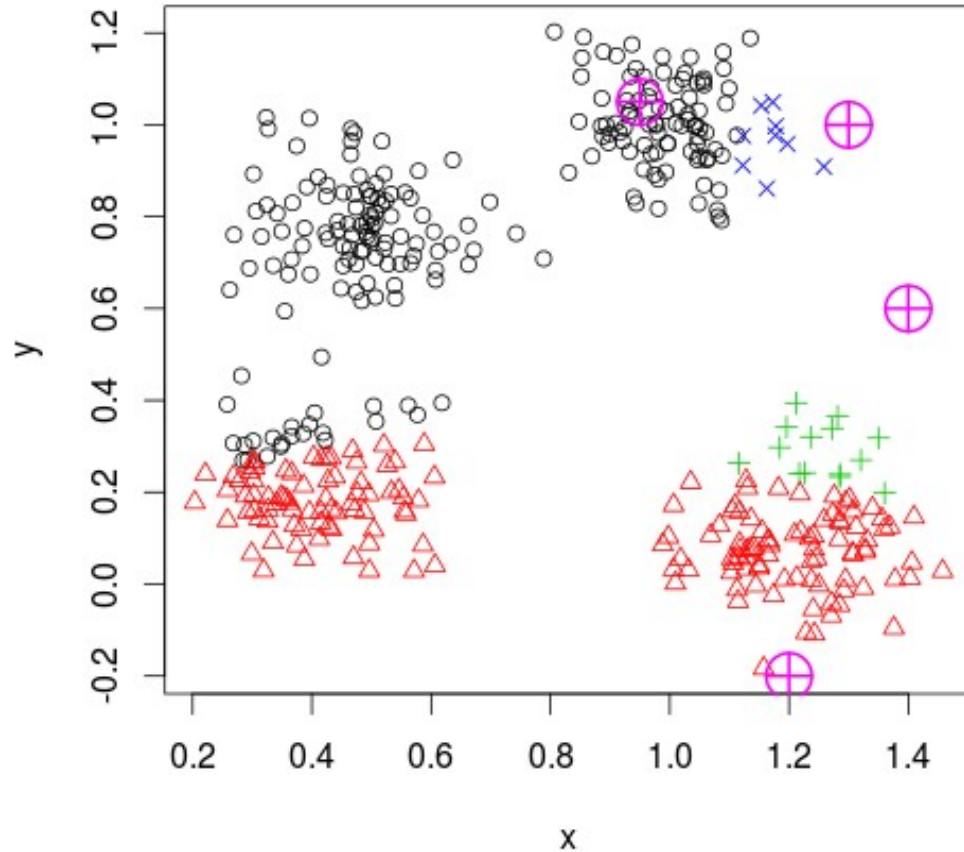
# Algoritme

- selecteer n willekeurige punten ("centroids")
- herhaal
  - associeer ieder punt van de dataset met de centroid die het dichtste bij ligt (zo maak je n clusters)
  - bereken per cluster het "midden" en vervang de centroid door deze nieuwe waarde
  - totdat de centroids niet meer veranderen

# Voorbeeld k-means

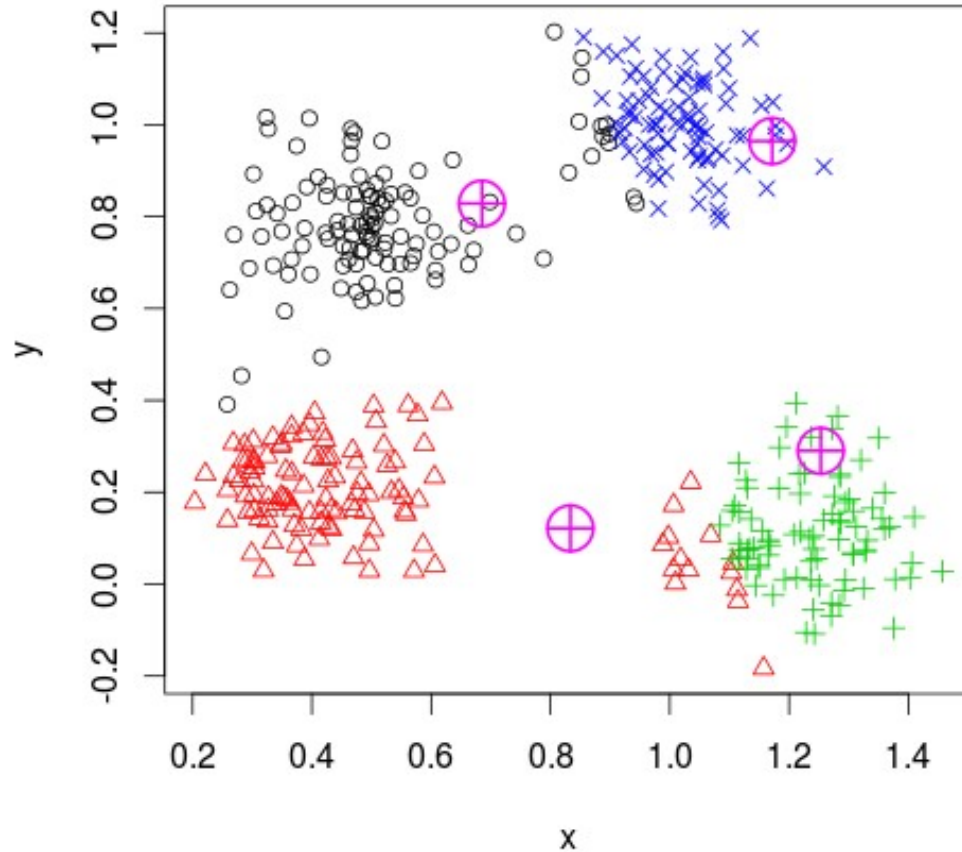


# Voorbeeld k-means

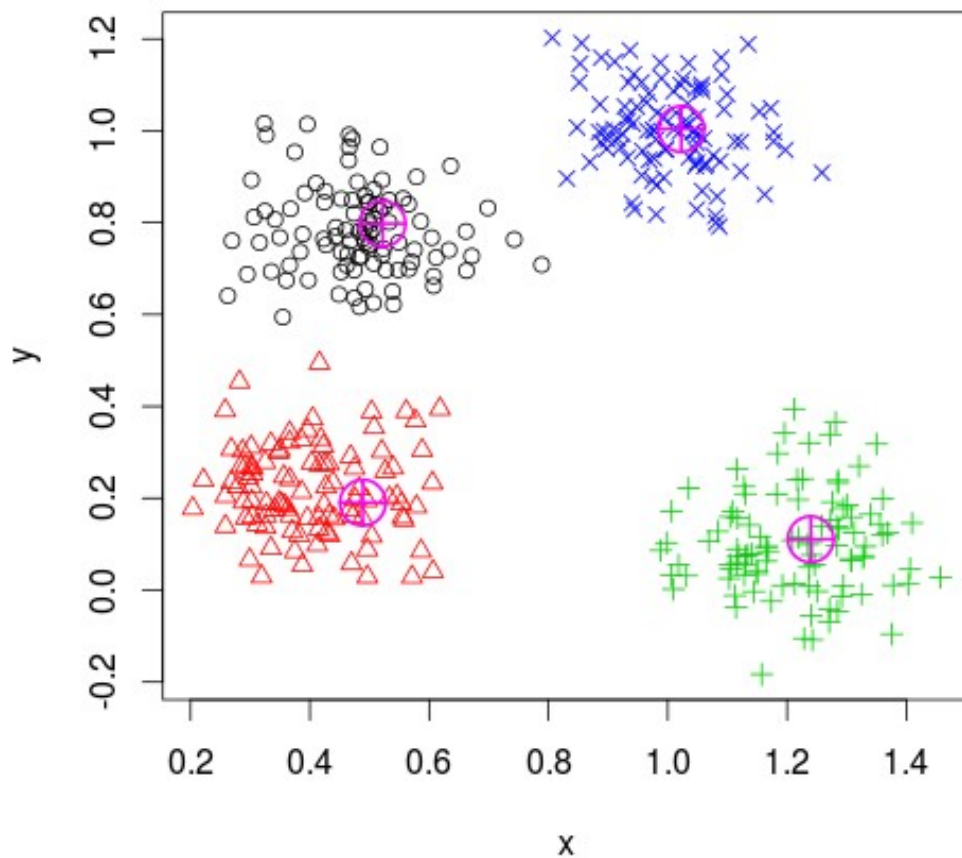




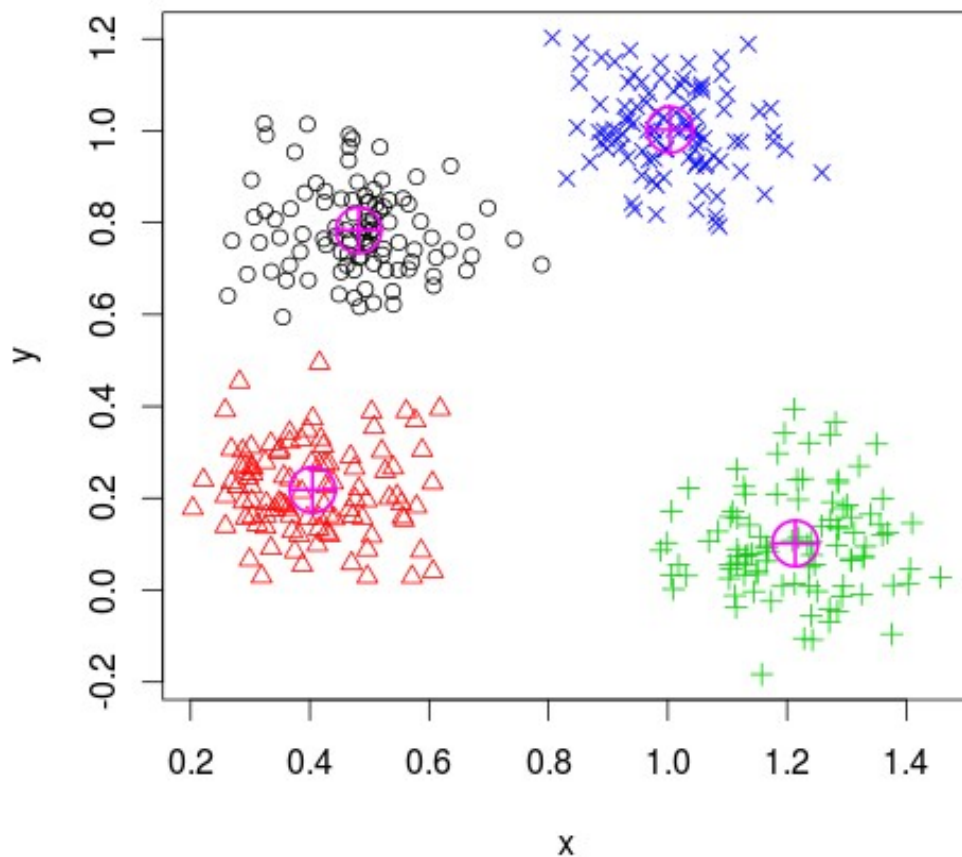
# Voorbeeld k-means



# Voorbeeld k-means



# Voorbeeld k-means



# Orange

Number of Clusters

☒ Fixed:

☐ From  to

Preprocessing

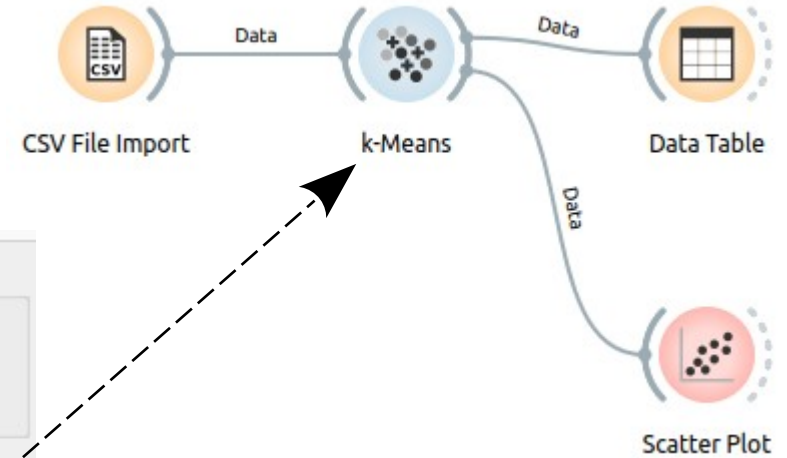
☐ Normalize columns

Initialization

Random initialization

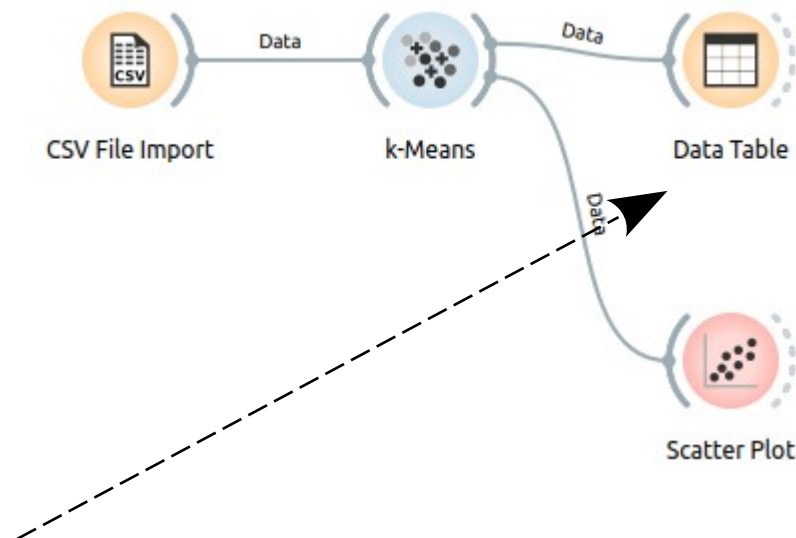
Re-runs:

Maximum iterations:

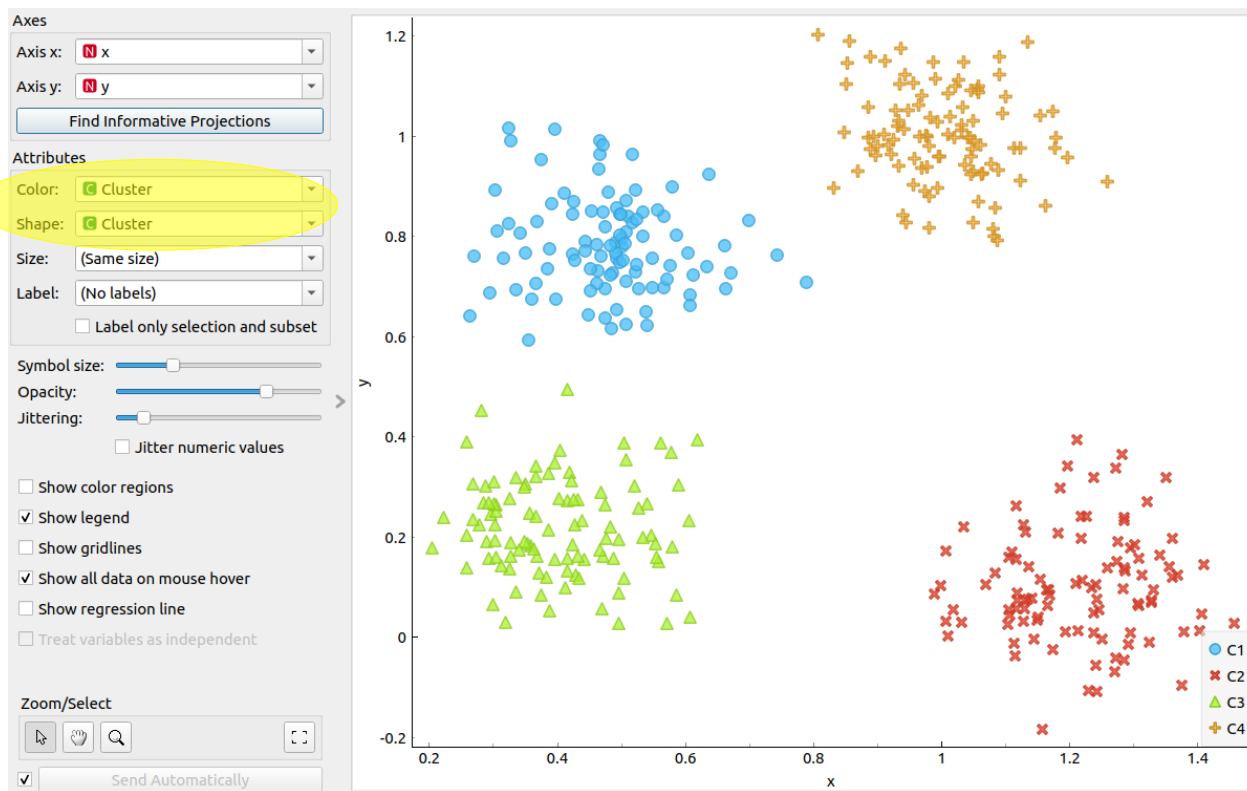
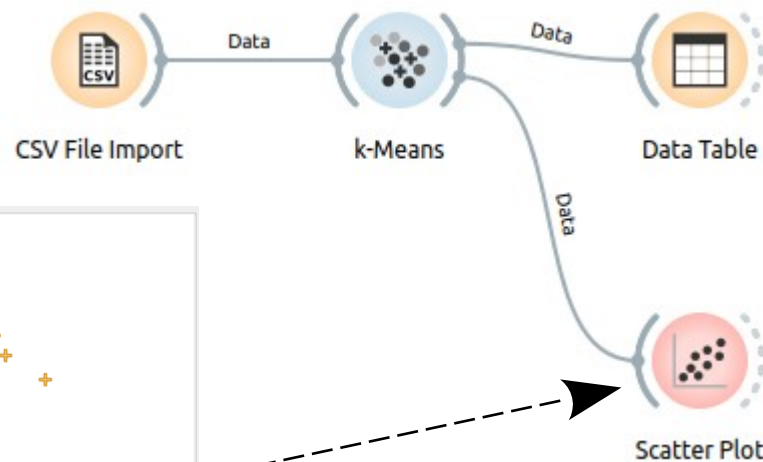


# Orange

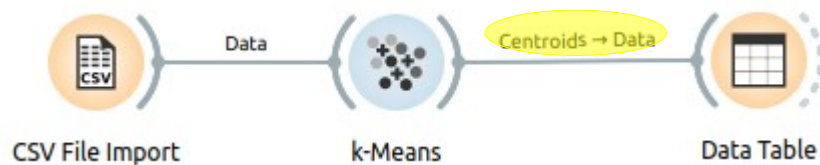
	Cluster	Silhouette	x	y
1	C1	0.664926	0.465681	0.991396
2	C1	0.69464	0.575295	0.741192
3	C1	0.705343	0.463535	0.731243
4	C1	0.710427	0.474027	0.819973
5	C1	0.701047	0.387885	0.77488
6	C1	0.702809	0.49277	0.858169
7	C1	0.681547	0.37465	0.95382
8	C1	0.71059	0.467764	0.75999
9	C1	0.705337	0.452288	0.735995
10	C1	0.712075	0.490255	0.766273
11	C1	0.700935	0.507172	0.711021
12	C1	0.652381	0.66262	0.695583
13	C1	0.660661	0.506605	0.624898
14	C1	0.7097	0.49776	0.748978
15	C1	0.663579	0.395756	1.01441
16	C1	0.702645	0.540043	0.756055



# Orange

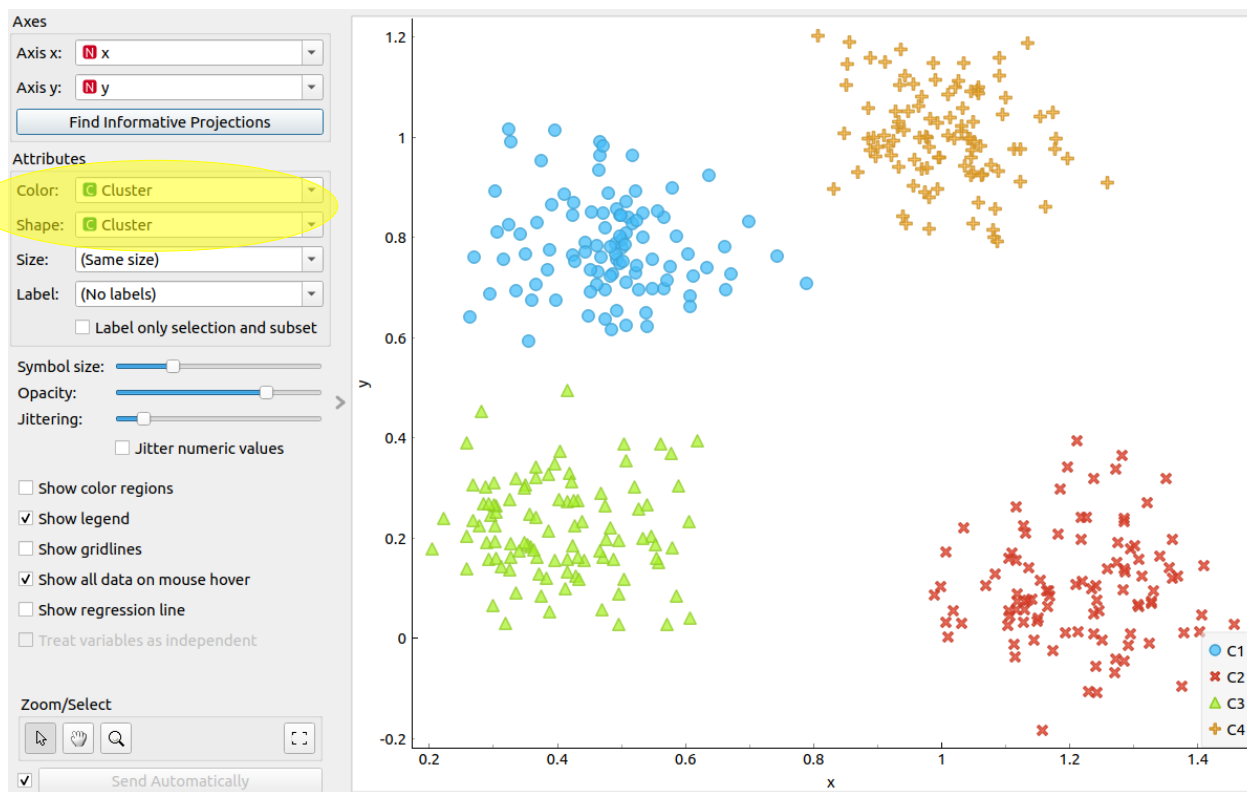
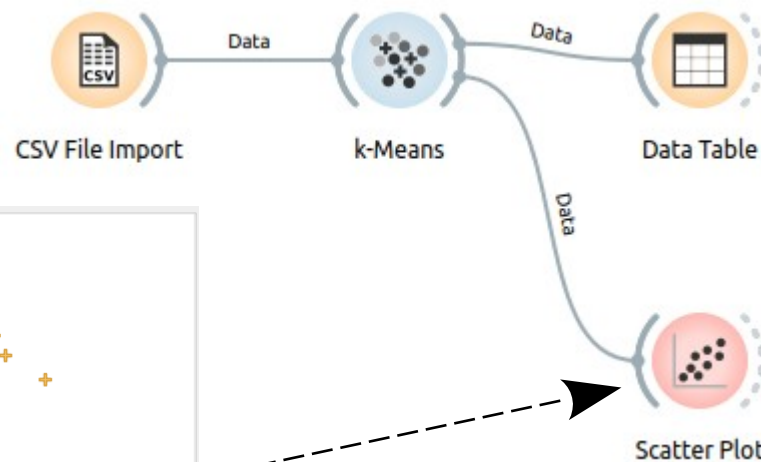


# Orange



	Cluster	Silhouette	x	y
1	C1	0.684528	0.481826	0.783411
2	C2	0.705961	1.21375	0.101757
3	C3	0.689991	0.404596	0.218268
4	C4	0.697121	1.00386	1.00228

# Orange

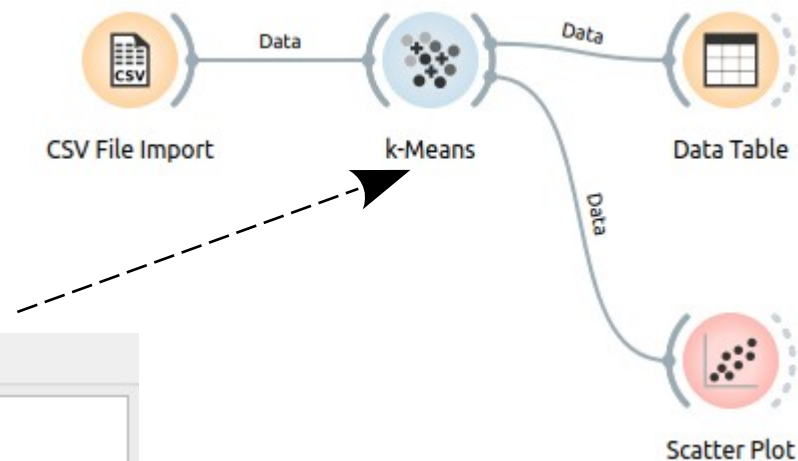




# K-means

- je moet op voorhand weten hoeveel clusters je zoekt
- je kan dit oplossen door verschillend aantal te zoeken en na te gaan welk aantal het beste scoort

# Orange



**Number of Clusters**

☐ Fixed: 4

☒ From 2 to 8

**Preprocessing**

☐ Normalize columns

**Initialization**

Random initialization

Re-runs: 10

Maximum iterations: 300

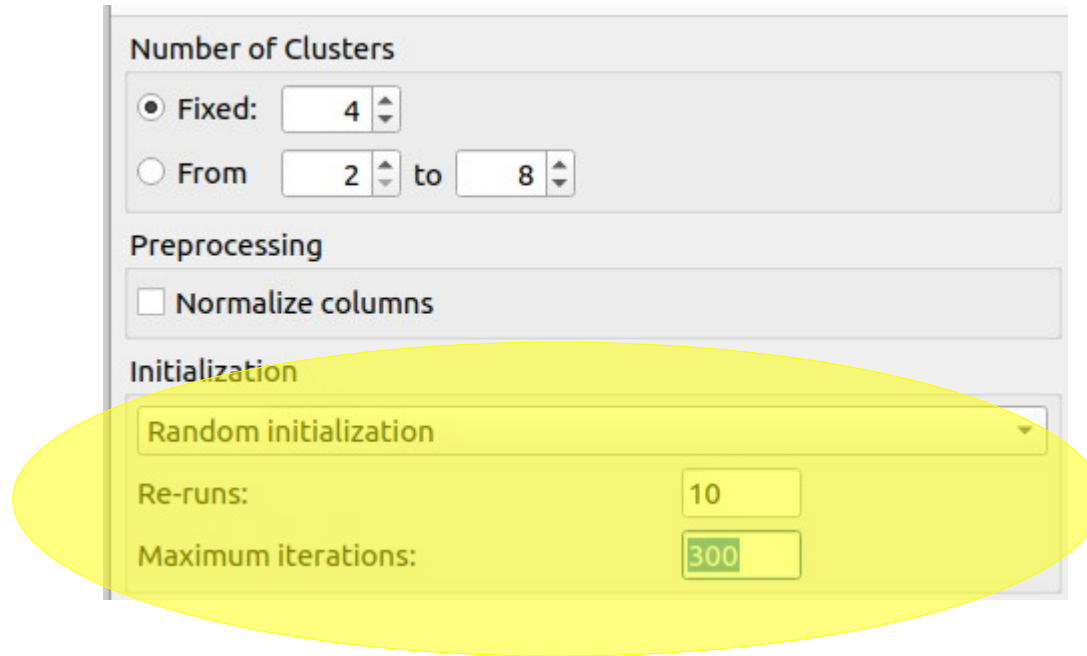
☒ Apply Automatically

**Silhouette Scores**

2	0.489
3	0.610
4	0.705
5	0.605
6	0.506
7	0.410
8	0.327

# K-means

- plaats van initiële centroids kan belangrijk zijn (Orange doet dit op een 'intelligente' manier)



Number of Clusters

☒ Fixed: 4

☐ From 2 to 8

Preprocessing

☐ Normalize columns

Initialization

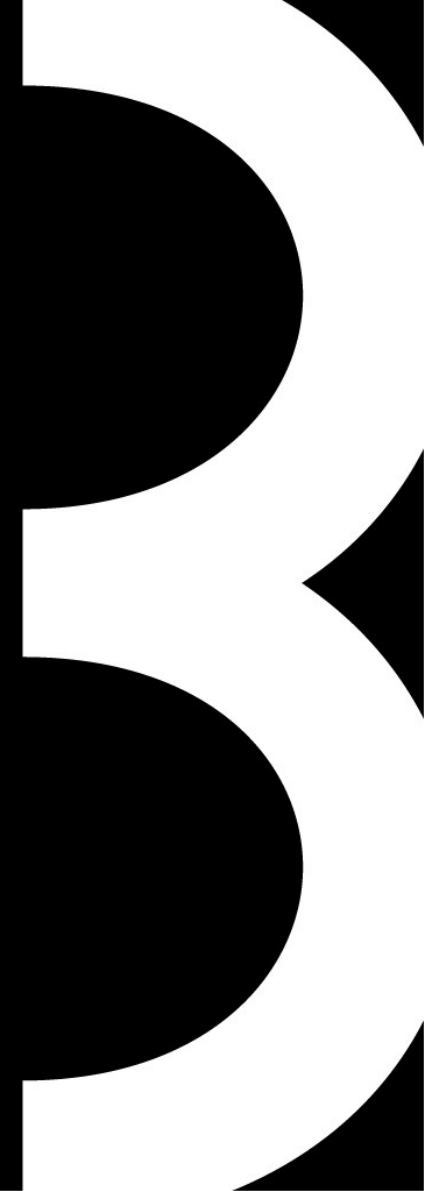
Random initialization

Re-runs: 10

Maximum iterations: 300

---

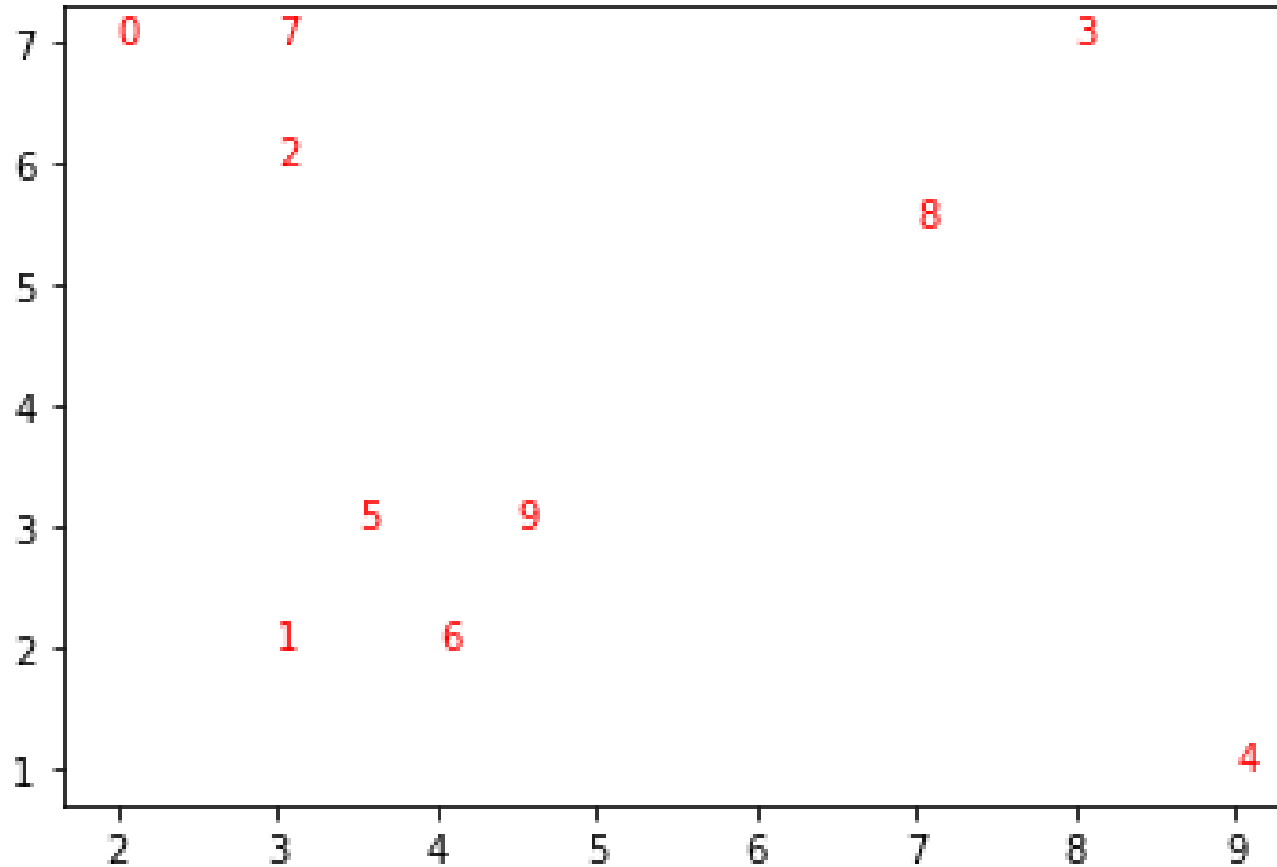
Clusters zoeken:  
hiërarchisch



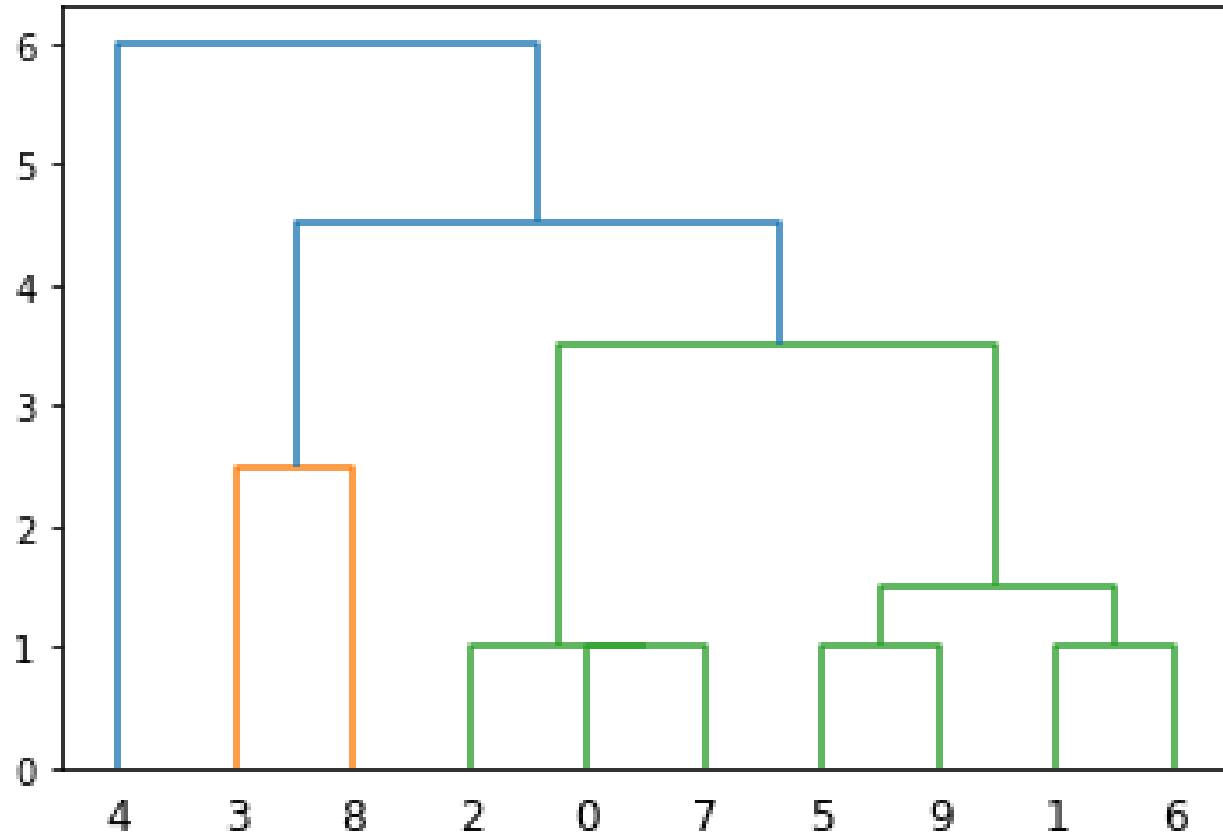
# Hiërarchische clustering

- als je op voorhand niet weet hoeveel clusters je zoekt
- deterministisch: levert steeds hetzelfde resultaat
- maakt een boomstructuur: "dendrogram"

# Voorbeeld



# Dendrogram



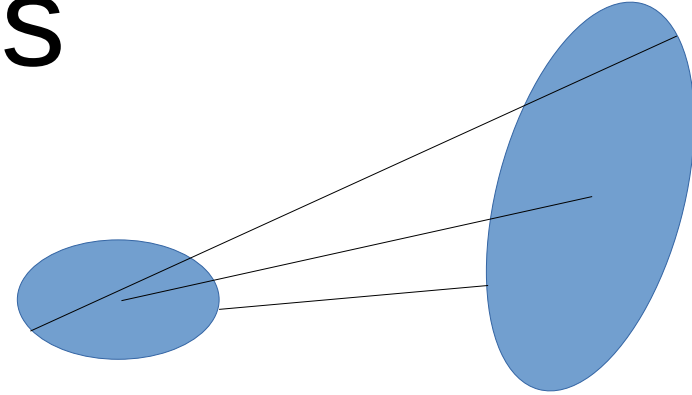
# Algoritme

- begin "onderaan": ieder punt is een cluster met 1 element erin
- herhaal
  - zoek de 2 clusters die het dichtst bij elkaar liggen
  - voeg deze clusters bij elkaar in een hoger niveau
  - tot er maar 1 cluster overblijft

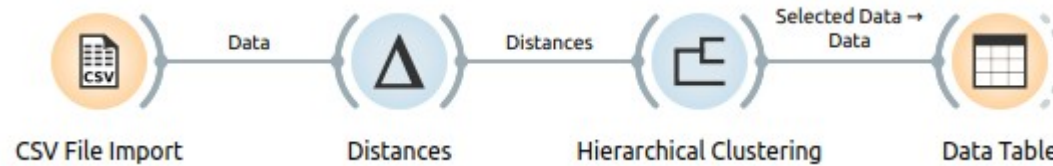


# Afstand tussen clusters

- je moet nu de afstand tussen 2 clusters kunnen berekenen
  - afstand tussen middelpunten (centroid linkage)
  - min afstand tussen de punten (single linkage)
  - max afstand tussen de punten (complete linkage)
  - gemiddelde afstand tussen de punten (average linkage)
  - mediaan afstand tussen de punten (median linkage)
  - ...



# Orange



	N	1	N	2
1		x		y
2		2		7
3		3		2
4		3		6
5		8		7
6		9		1
7		3.5		3
8		4		2
9		3		7
10		7		5.5
11		4.5		3

# Orange



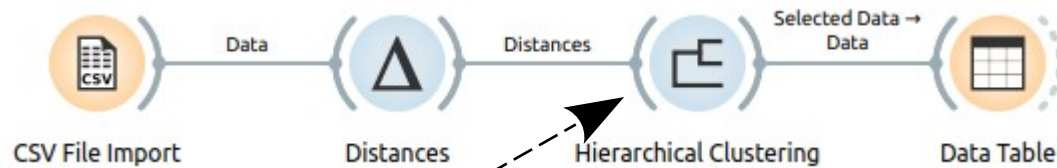
## Compare

☒ Rows☐ Columns

## Distance Metric

☐ Euclidean (normalized)☐ Cosine☒ Euclidean☐ Pearson☐ Manhattan (normalized)☐ Pearson (absolute)☐ Manhattan☐ Spearman☐ Mahalanobis☐ Spearman (absolute)☐ Hamming☐ Jaccard

# Orange



Linkage

Single

Annotations

N x

☐ Show labels only for subset

Color by: None

Pruning

☒ None

☐ Max depth:

10

Selection

☐ Manual

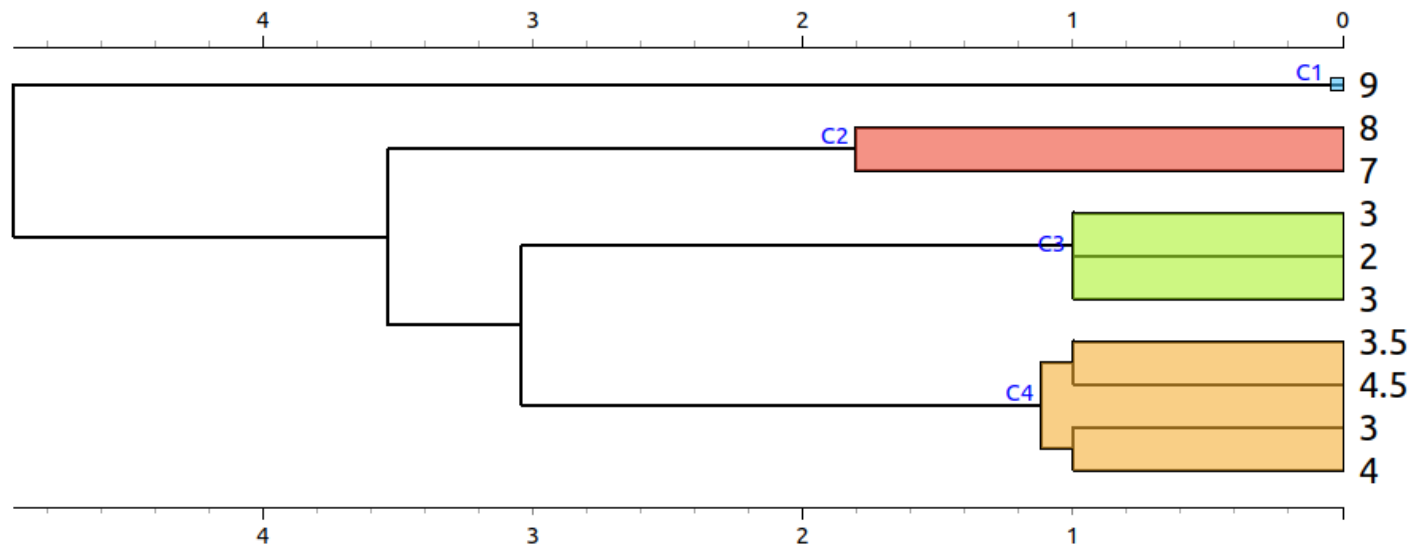
☐ Height ratio:

100,0 %

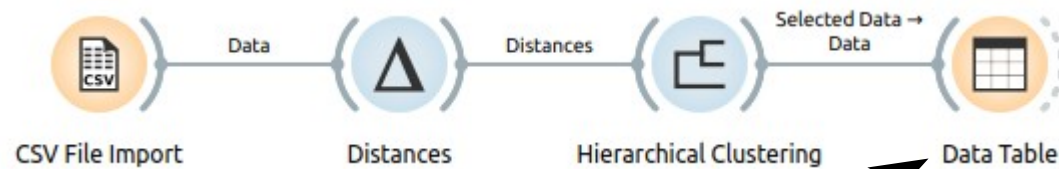
☒ Top N:

4

Zoom



# Orange



	Cluster	x	y
1	C3	2	7
2	C4	3	2
3	C3	3	6
4	C2	8	7
5	C1	9	1
6	C4	3.5	3
7	C4	4	2
8	C3	3	7
9	C2	7	5.5
10	C4	4.5	3

---

Combinatie met  
beslissingsbomen



# Clustering en beslissingsbomen

- clustering geeft een "label" aan iedere rij
- voeg dit label als kolom toe
- een beslissingsboom kan nu bepalen waarom een rij in een cluster hoort!

# Voorbeeld



## Tree

7 nodes, 4 leaves

## Display

Zoom:

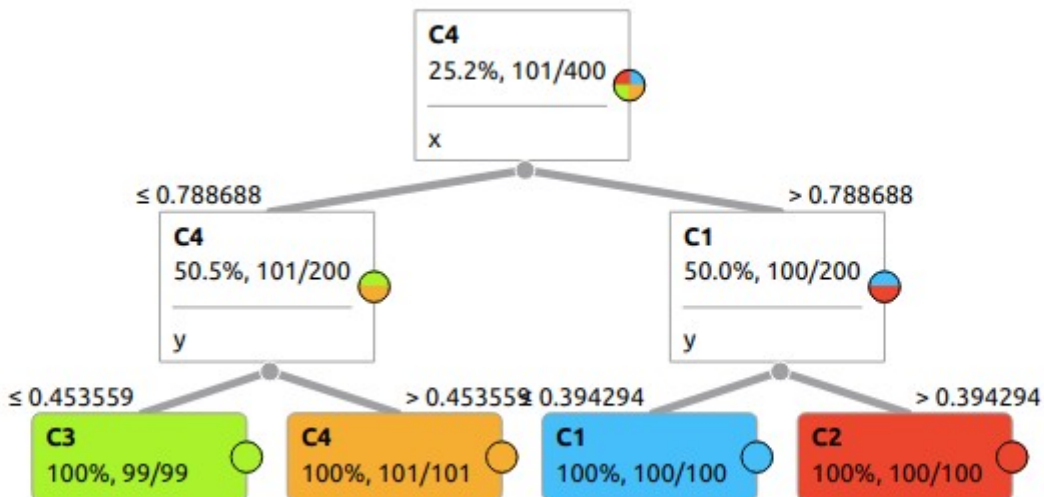
Width:

Depth: Unlimited

Edge width: Relative to parent

Target class: None

☒ Show details in non-leaves





---

Oefeningen

# Oefeningen

- Simpsons revisited
- Studenten
- Extraterrestrial life