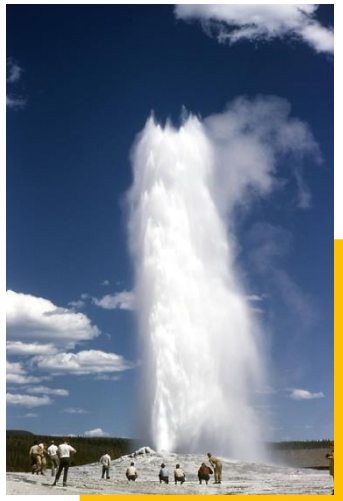


Data Analysis & Machine Learning
Unsupervised learning: clustering

Daniel Oberski

Department of Methodology, Statistics & Data Science

Utrecht University

Today

- **Part A – unsupervised learning: what changes without y**
- **Part B – distance-based clustering**
  - distances, hierarchical clustering, k-means
- **Part C – model-based clustering**
  - Gaussian mixtures, EM, BIC, mclust
- **Part D – is the clustering any good?**
  - five ways to evaluate a clustering
- **Clustering text (Assignment 2) and a checklist**

Reading

James, Witten, Hastie & Tibshirani, An Introduction to Statistical Learning (ISLR), §12.1 and §12.4

Oberski (2016), Mixture models: latent profile and latent class analysis, §1–2

Optional

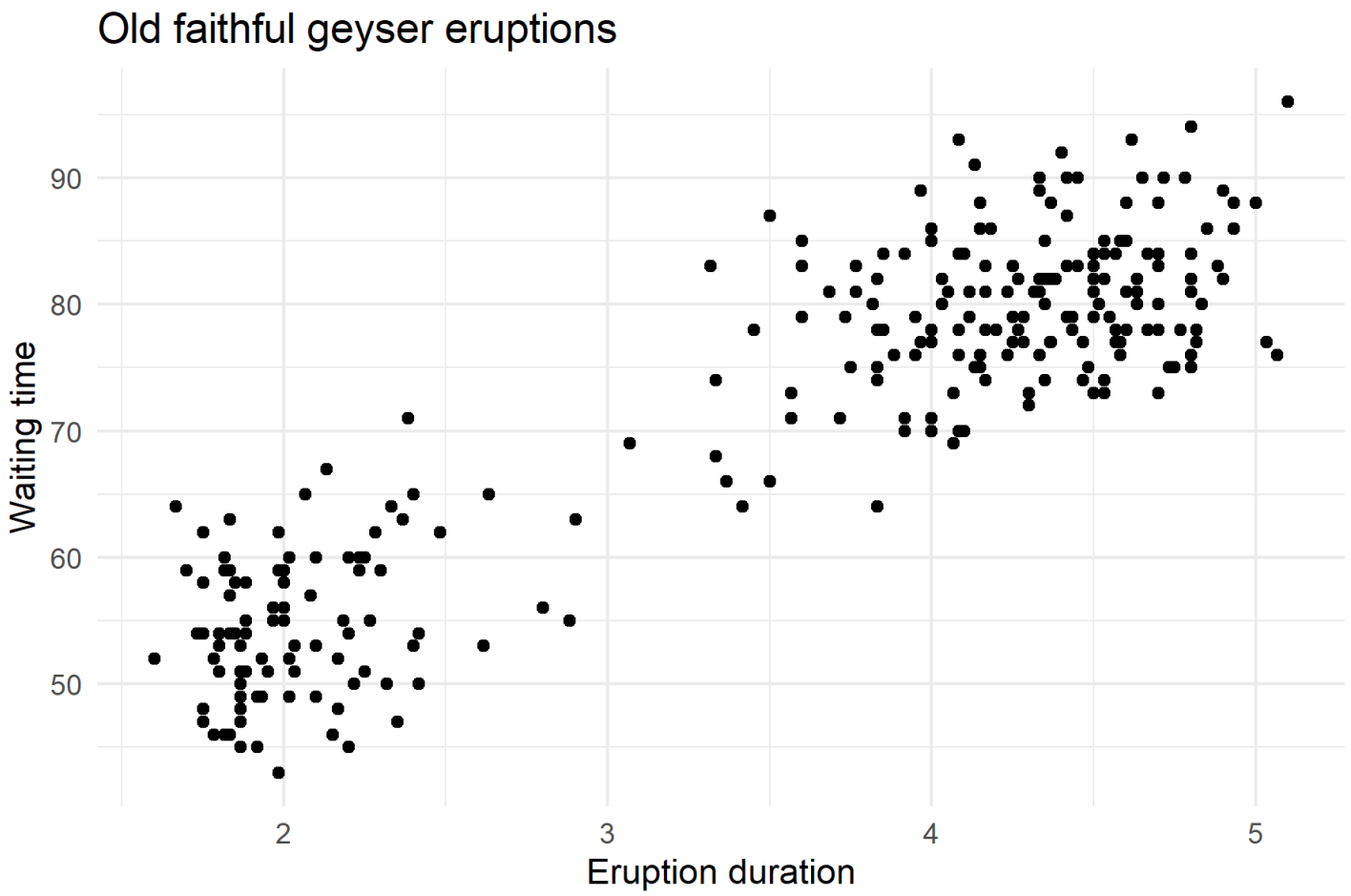
Hennig (2016), Clustering strategy and method selection, Handbook of Cluster Analysis ch. 31 (arXiv 1503.02059)

Bouveyron, Celeux, Murphy & Raftery (2019), Model-Based Clustering and Classification for Data Science

Lab Thursday: hierarchical clustering, k-means and mclust (R)

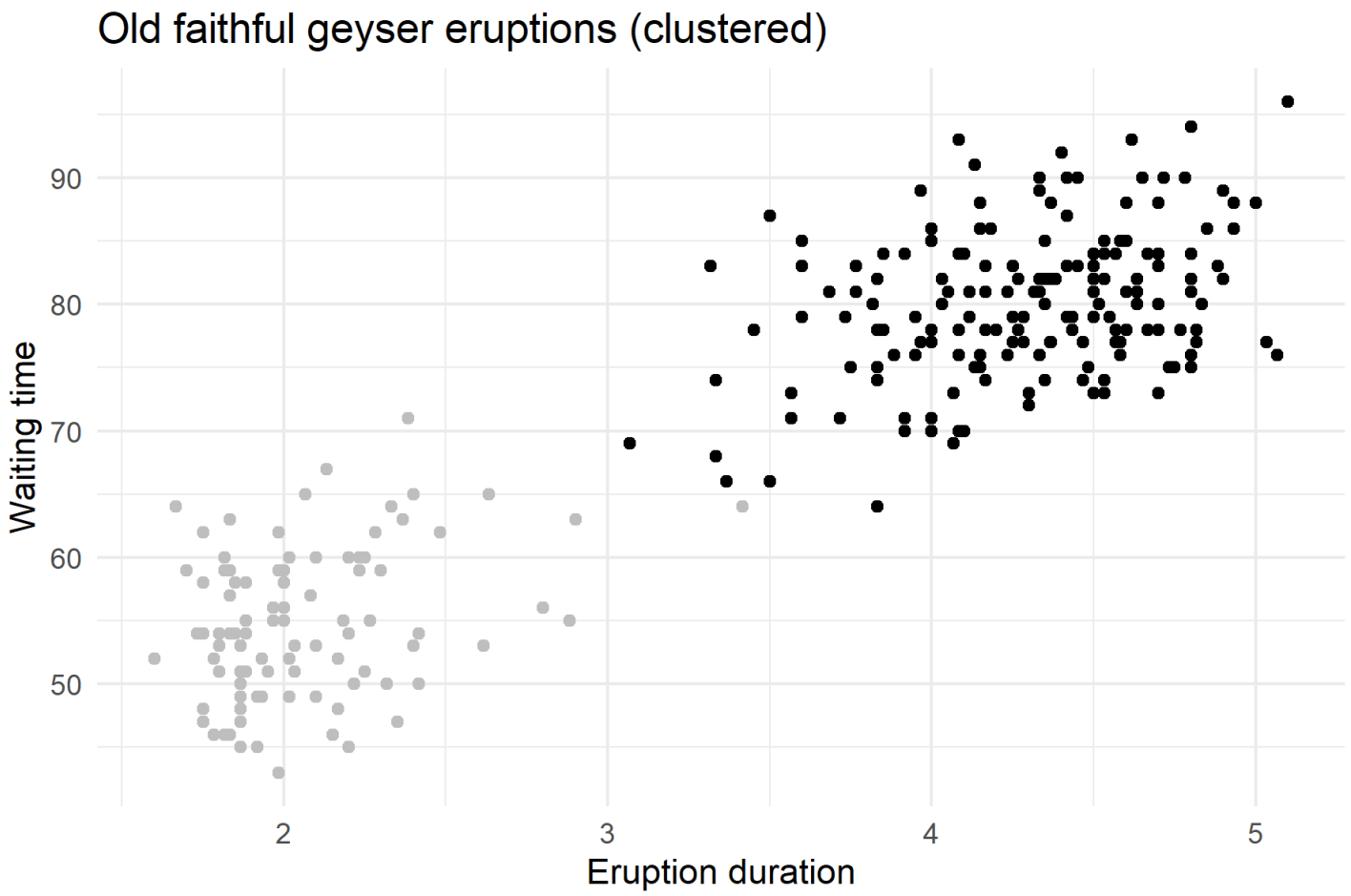


Old Faithful: two types of eruption?



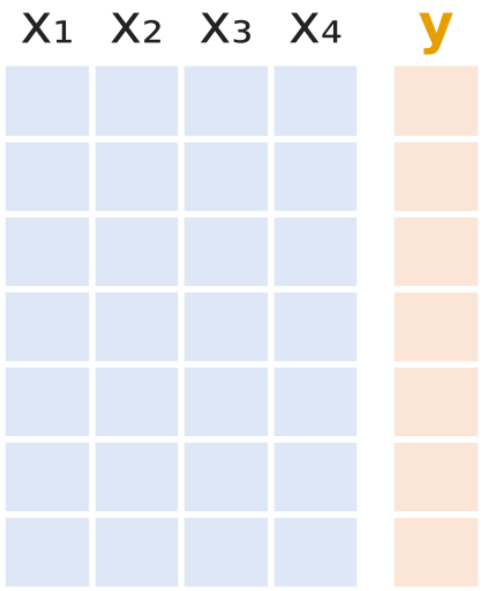


Old Faithful: two types of eruption?



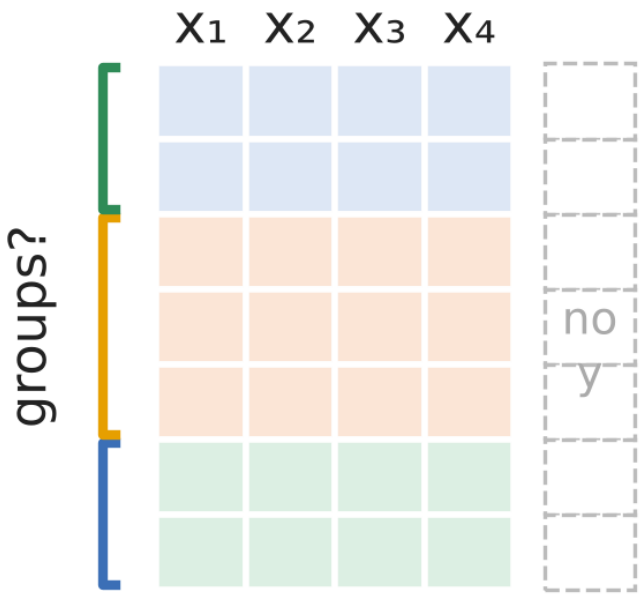
So far: learn y from X. Today: only X

Supervised (weeks 1–4)



learn f : X → y, then predict y for new rows

Unsupervised (today)



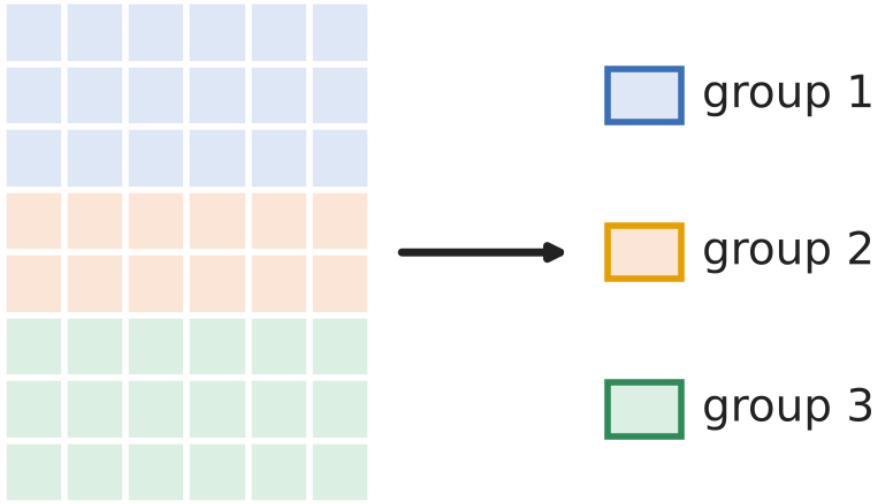
only X: find structure in the data itself

Same kind of distances as kNN (week 2) — but no labels to learn from

Two ways to simplify a data matrix

Cluster the rows

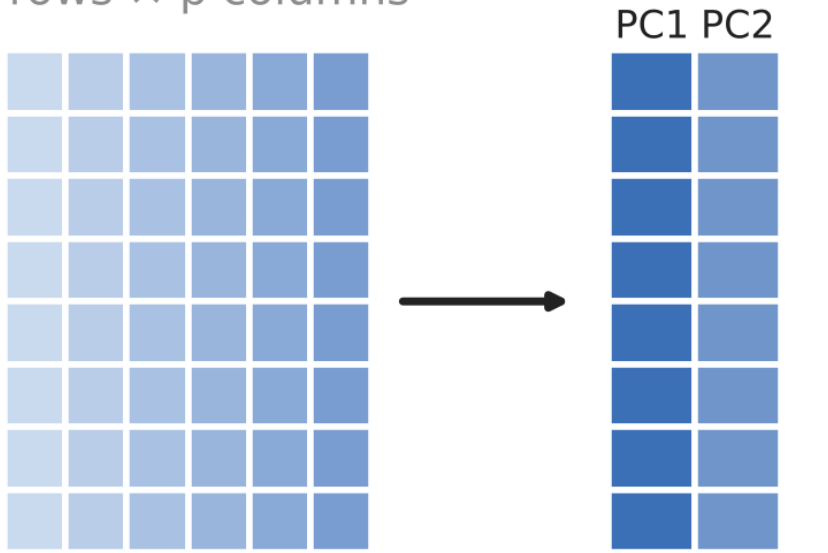
n rows × p columns



similar observations → K groups
(a new categorical variable)

Compress the columns

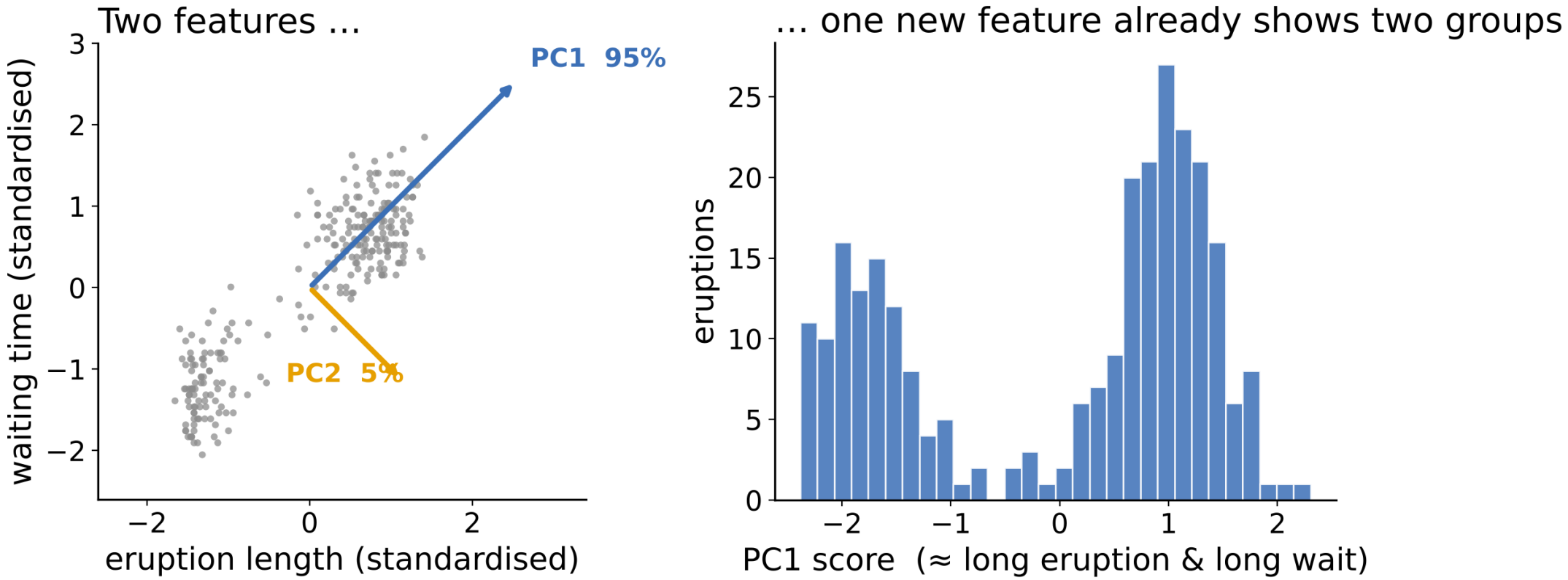
n rows × p columns



many features → a few new ones
(PCA; later in the course: embeddings)

Also unsupervised: density estimation and anomaly detection — a Gaussian mixture (Part C) does both

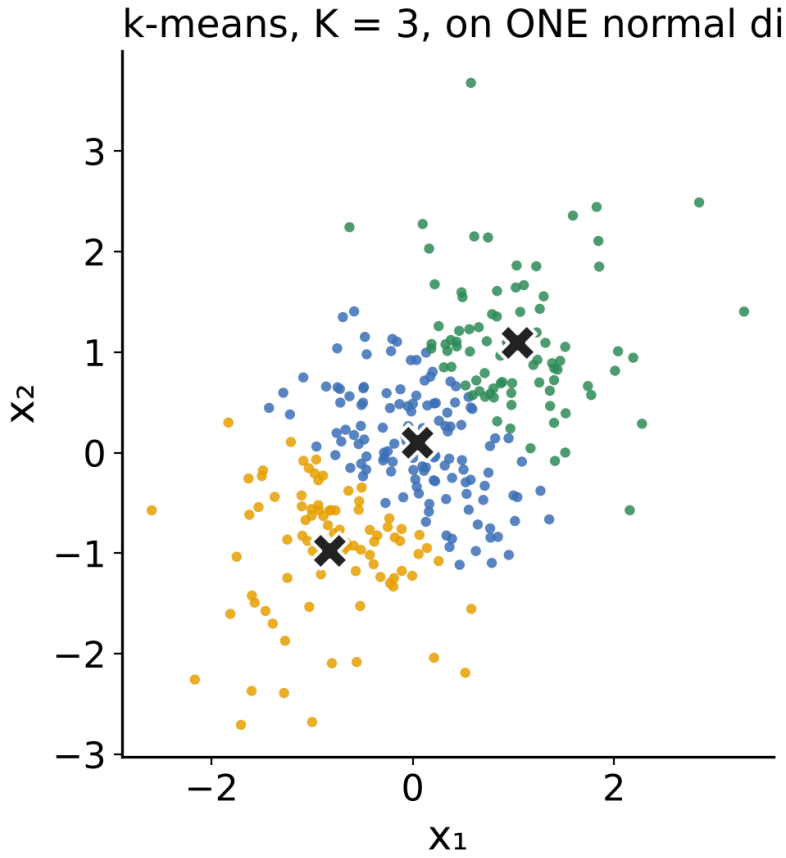
Dimension reduction in one picture



- PCA finds a few new features (principal components) that keep most of the variation
- Today: look at clusters in 2-D (Part D) · reduce text before clustering (Assignment 2)

Why unsupervised learning is harder

- No y: no test error, no cross-validation in the usual sense
- What counts as a “good” clustering depends on your aim
- Every algorithm returns clusters, even when there are none (**overfitting**) →
- Week 3: “EDA can produce chance findings” — so can clustering
- **This is why evaluating unsupervised solutions / clustering solutions is an important topic**



three 'clusters' — but there are no groups
(average silhouette width 0.32: weak)

Clustering: what and why

Find subgroups (clusters) of similar observations

Why?

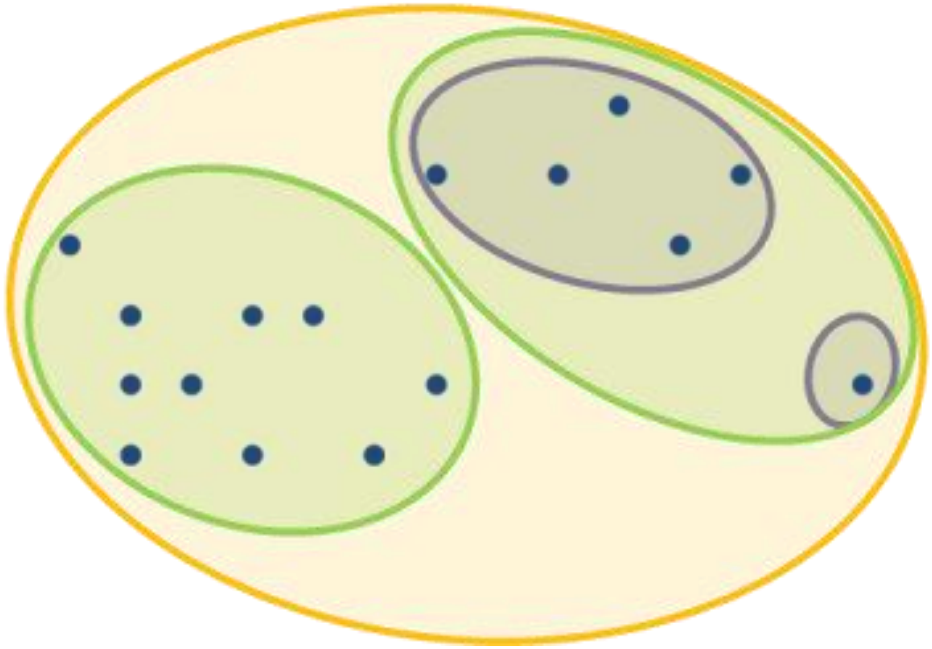
- discover groups we could not measure: cancer subtypes, types of eruption
- summarise many features into one categorical feature: customer segments

Some applications

customer segmentation
recommendation (collaborative filtering)
data summarisation and compression
trend detection
image and text analysis (Assignment 2)
biology: subtypes, gene expression
social networks: communities
anomaly detection

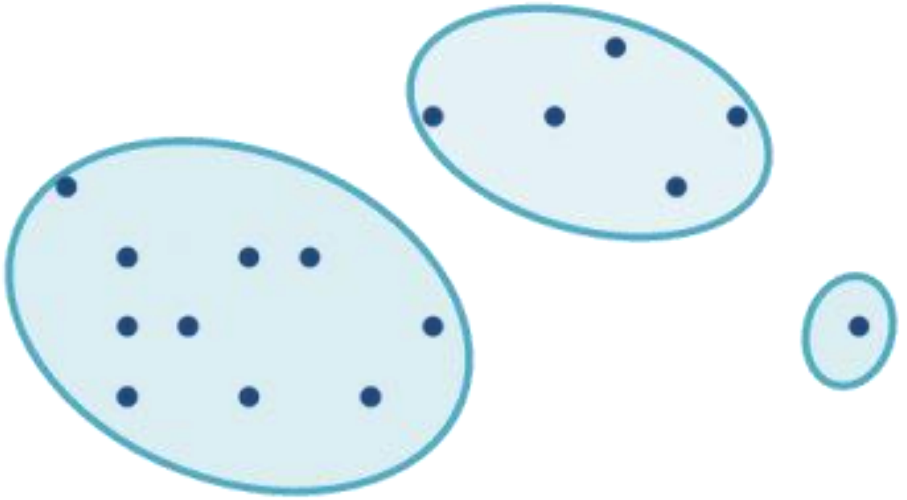
Two families of clustering algorithms

Hierarchical Clustering



Hierarchical: nested clusters → a tree;
choose the number of clusters afterwards

Partitional Clustering

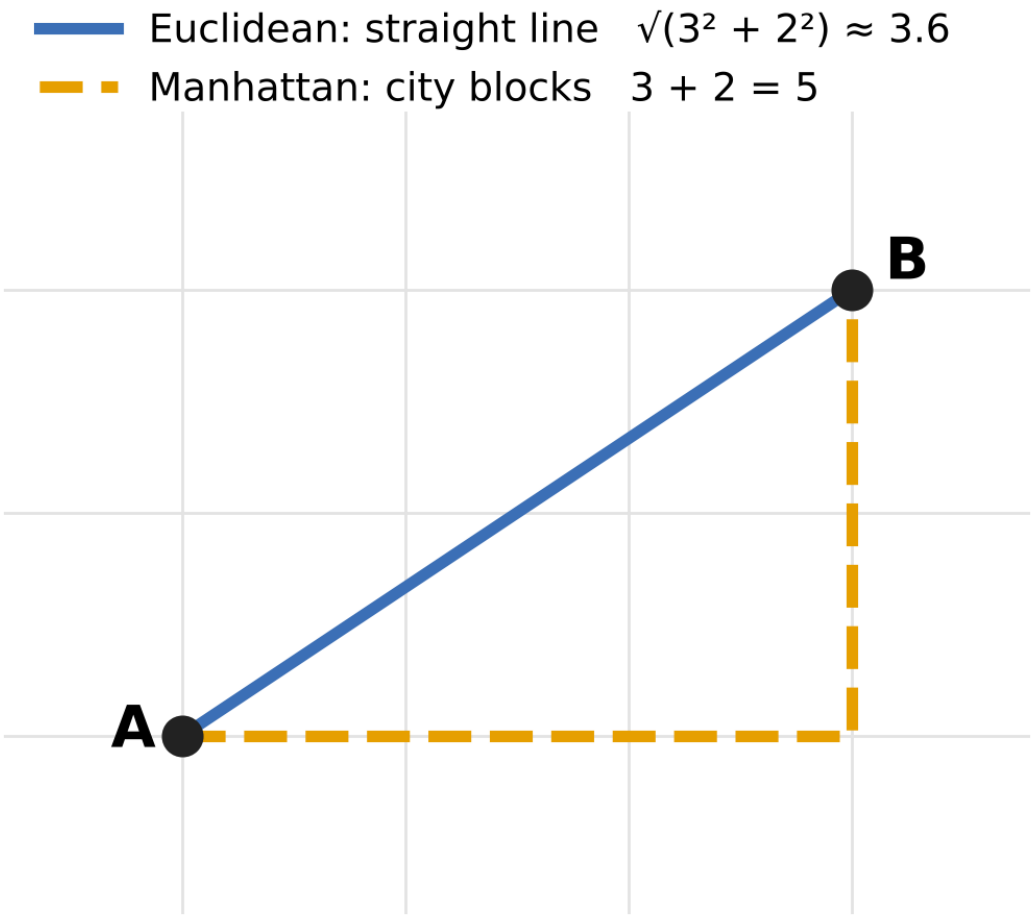


Partitional: one split into K groups
(k-means)

… and a third family, model-based clustering, in Part C

Figures: T. Fuertes, quantdare.com/hierarchical-clustering

It all starts with a distance



- Euclidean: the straight line — the usual default
- Manhattan: the sum of absolute differences (city blocks)
- **Scale first! Height in cm would outweigh width in m → standardise each feature (z-score)**
- k-medians: k-means with medians as centres — its Manhattan cousin, less sensitive to outliers

“**Distance**” can mean lots of things

- **Continuous**: Euclidean, maximum, Manhattan, Minkowski, …
- Time series: ”Dynamic time warping”, Fréchet, cross-corr., wavelets, …
- Networks: Modularity, Shortest path, …
- Text/DNA: cosine (TF-IDF, embeddings), edit distance, Hamming, …
- Images: “Structural similarity”, GAN loss…
- Mixed: Gower distance (numeric + categorical features)

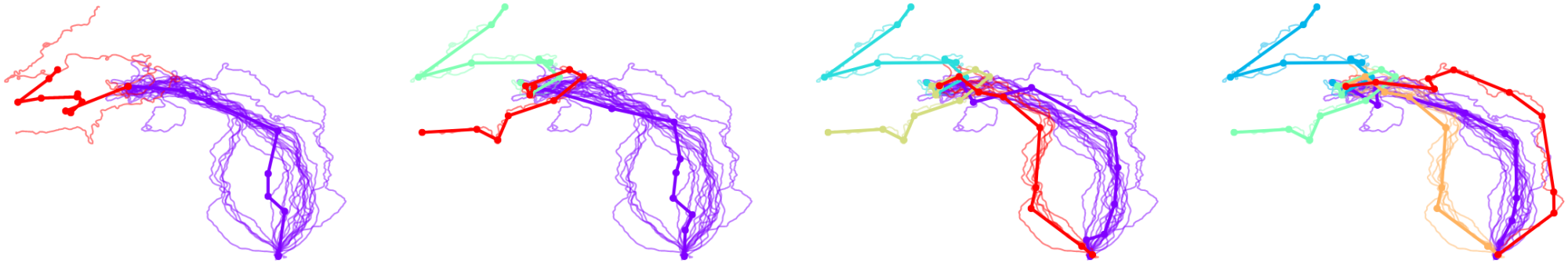


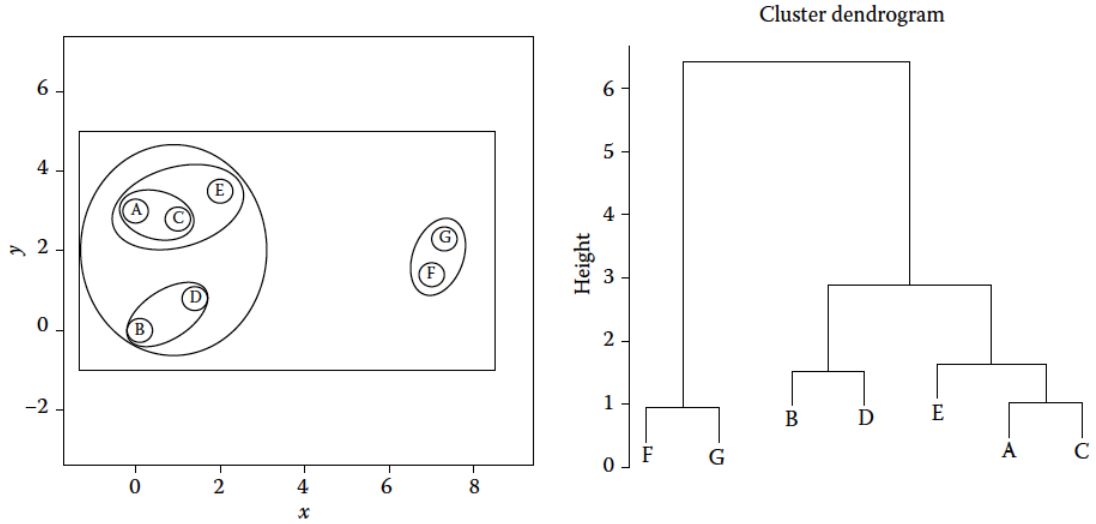
Figure 1: Example of a (k, ℓ)-clustering for the flight paths of a pigeon with the number of clusters k increasing from 2 (left) until 5 (right) and the complexity of the clusters being $\ell = 10$. Trajectories belonging to the same cluster are shown in the same color. For each cluster, a center trajectory generated by the algorithm is shown using thick lines of the same color.

Buchin et al. 2019

Hierarchical clustering: merge the closest, repeat

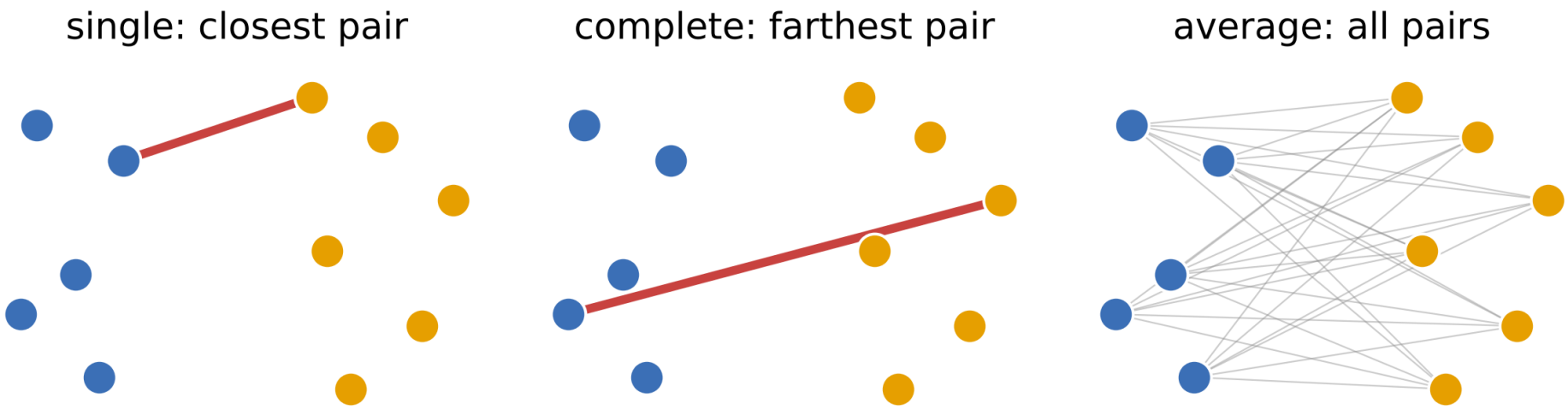
Bottom-up agglomerative clustering

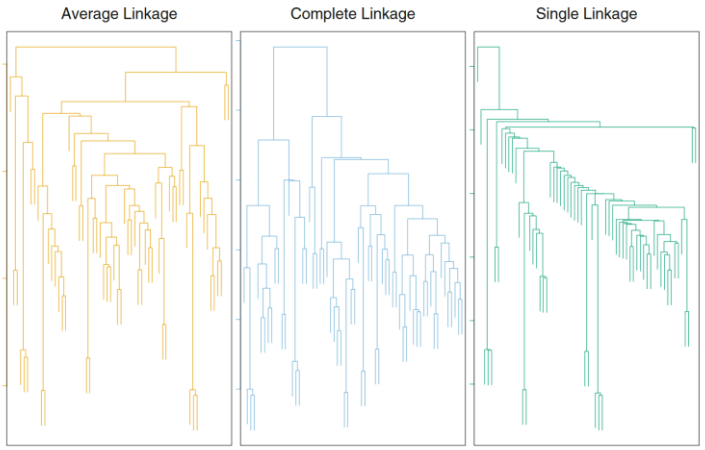
- For each observation, compute the *distance* to all other observations
- Assign all examples to their individual cluster
- Combine *most similar* clusters
- Keep combining clusters until there is only one cluster left
- Select number of clusters for the final solution



*(Divisive: start with **one** cluster and keep splitting most **different**)*

Linkage: distance between clusters

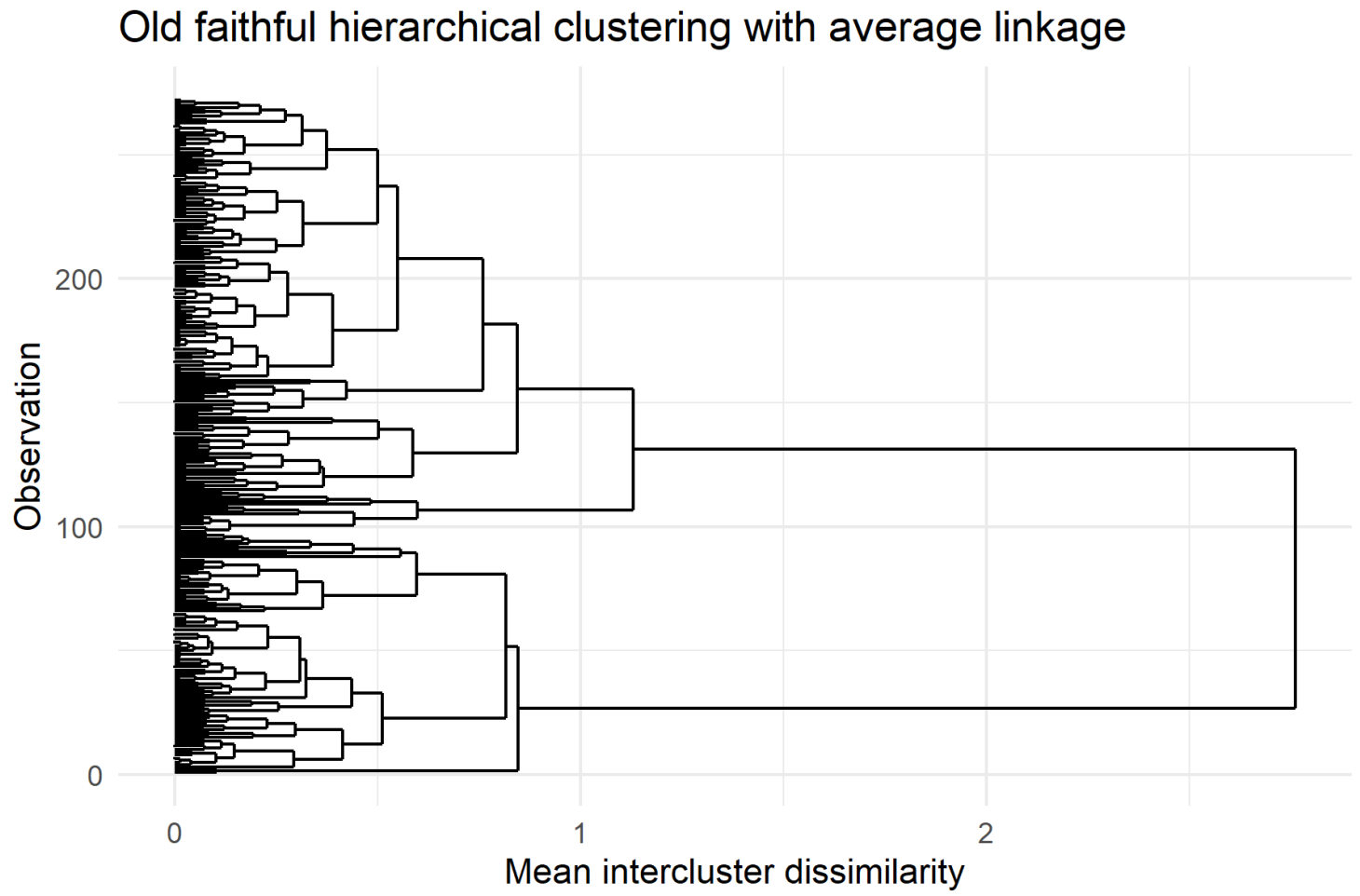




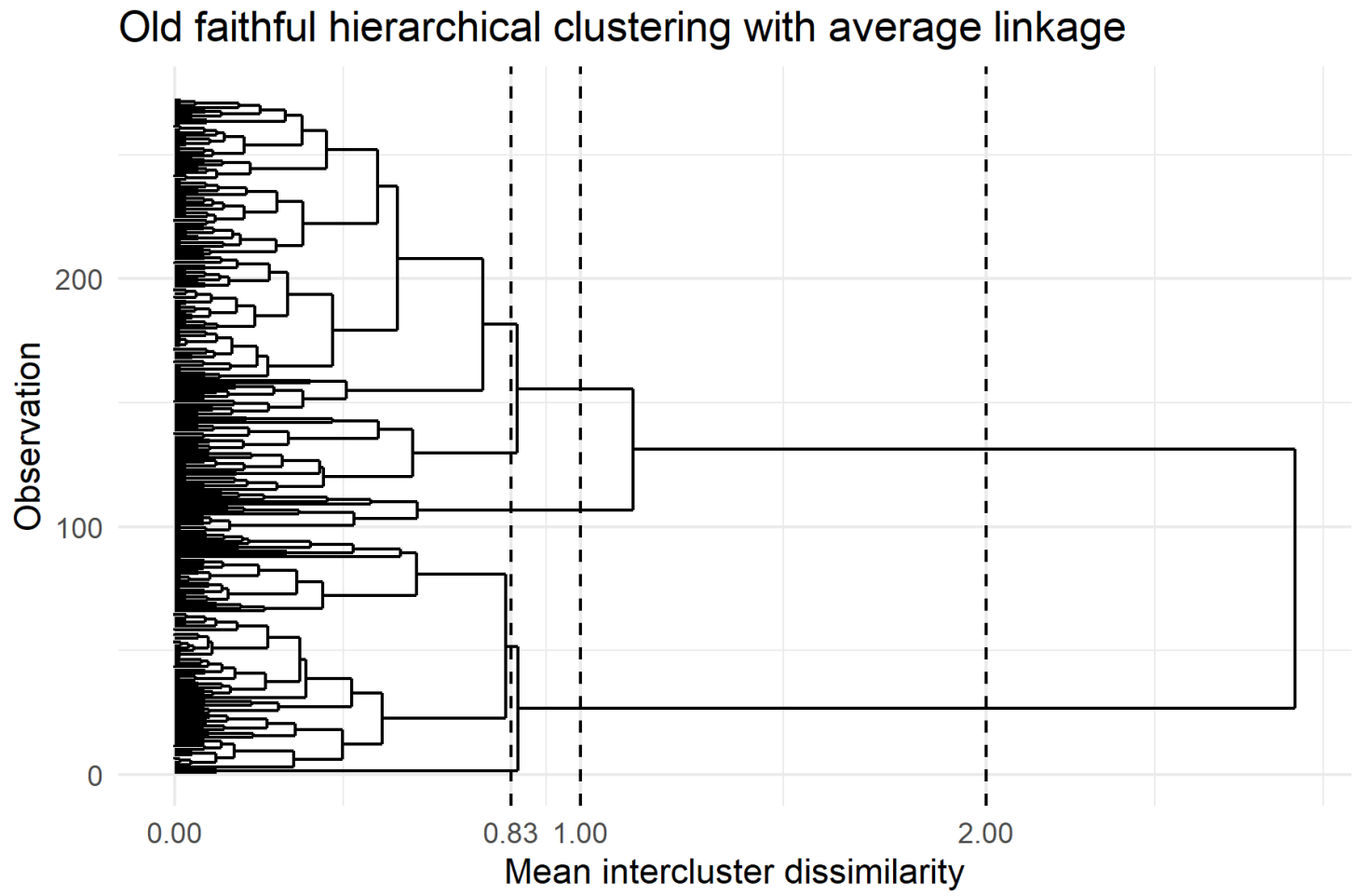
- Same data, three linkages → three different trees
- Complete: compact clusters (used in the lab)
- Single: can chain into long, thin clusters

Dendrograms: ISLR Fig. 12.14 (average, complete, single); see ISLR Table 12.3

Where to cut the tree?

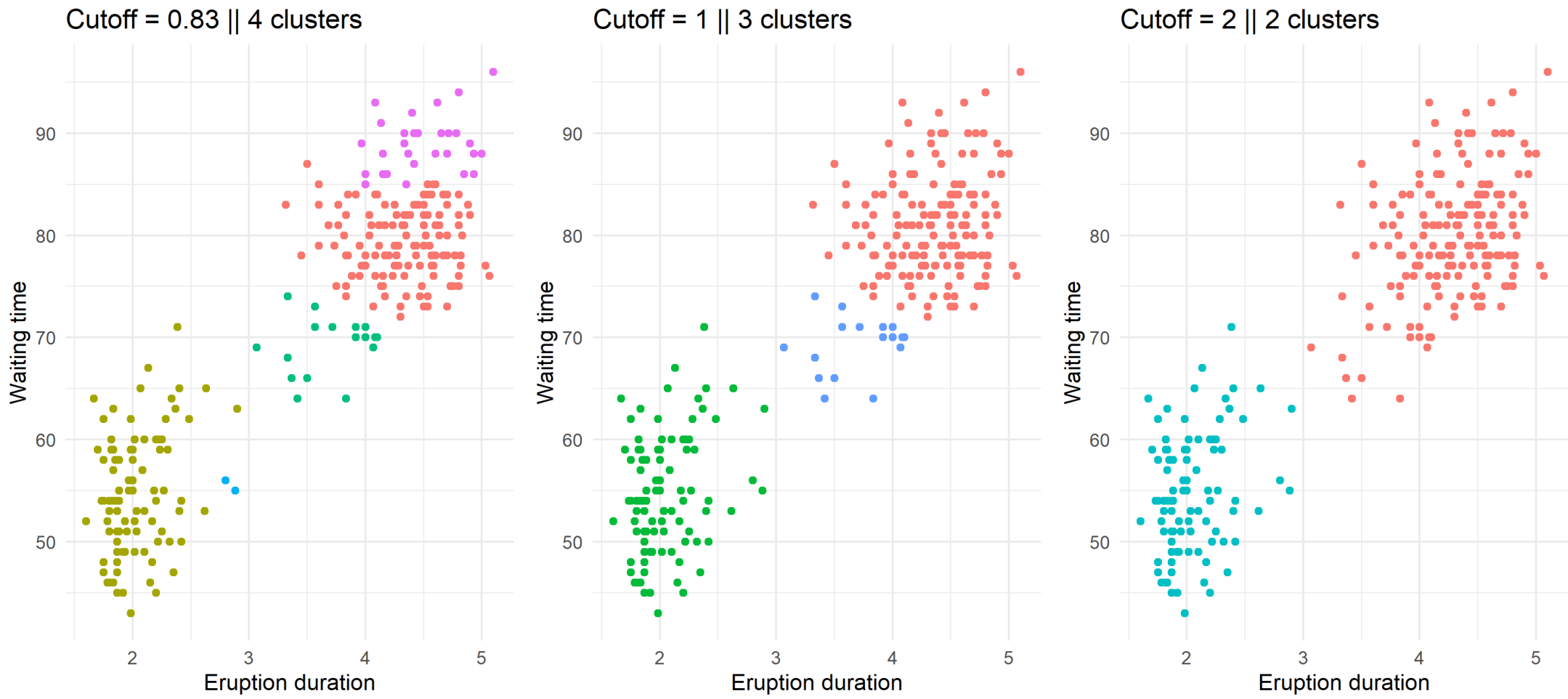


Where to cut the tree?



Question: which cut would you choose — and why?

Different cuts, different clusterings



Hierarchical clustering in R

In R:

```r
distances <- dist(faithful,    method = "euclidean")
result    <- hclust(distances, method = "average")
```

Then we can plot the dendrogram with `plot()` or `ggdendrogram`

```r
library(ggdendro)
ggdendrogram(result)
```

Then, select the number of clusters using a cutoff point

```r
cutree(result, h = 2)
```

```
# Python: scipy linkage(X, method='complete') + fcluster(), or sklearn AgglomerativeClustering
```

k-means in four steps

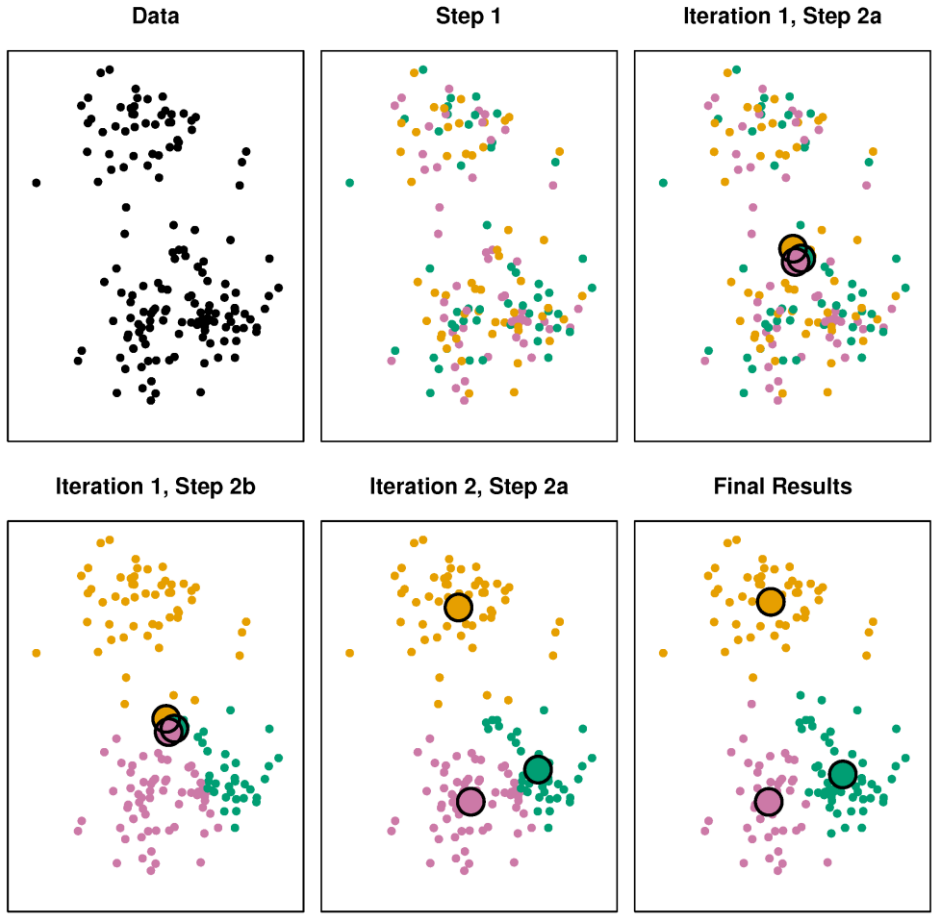
1. Randomly assign every observation to one of K clusters

2a. Compute each cluster's centroid (the per-feature mean)

2b. Move every observation to its closest centroid

3. Repeat 2a–2b until nothing changes

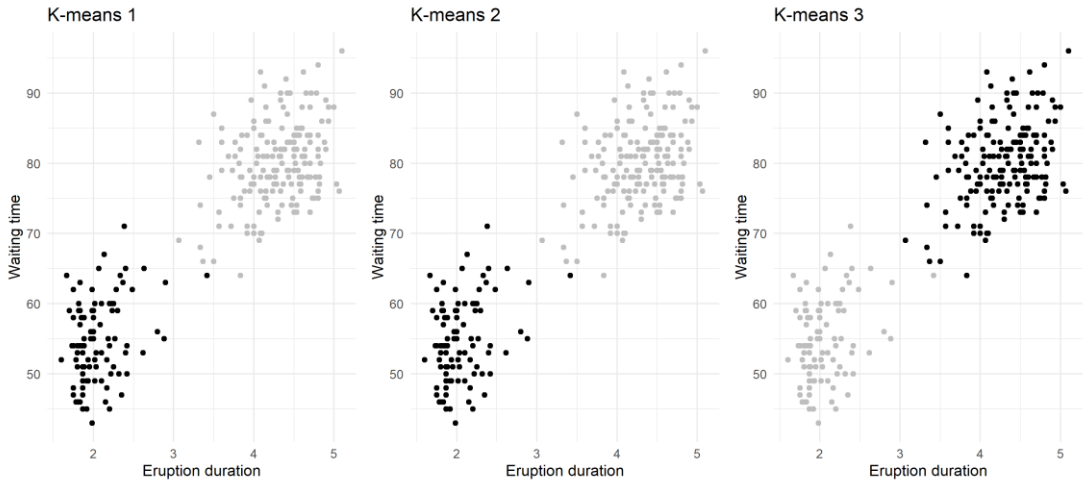
Every step lowers the within-cluster sum of squares → it always stops, but at a local optimum



ISLR Fig. 12.8

k-means in practice

- K is a hyperparameter: choose it from knowledge or your goal, or compare several K (Part D)
- Random start → random result: label switching, local optima
- **Use many random starts and keep the best; set a seed**



R

```r
set.seed(1)
km <- kmeans(x, centers = 3, nstart = 25)
km$cluster          # cluster of each row
km$centers          # centroids
km$tot.withinss     # within-cluster SS
```

Python

```python
from sklearn.cluster import KMeans
km = KMeans(n_clusters=3, n_init=25,
            random_state=1).fit(X)
km.labels_, km.cluster_centers_
```

Original

671 kB

k-means on pixel colours (“vector quantisation”)

$K = 2$

18 kB





k-means on pixel colours (“vector quantisation”)

Original

671 kB



$K = 10$

126 kB

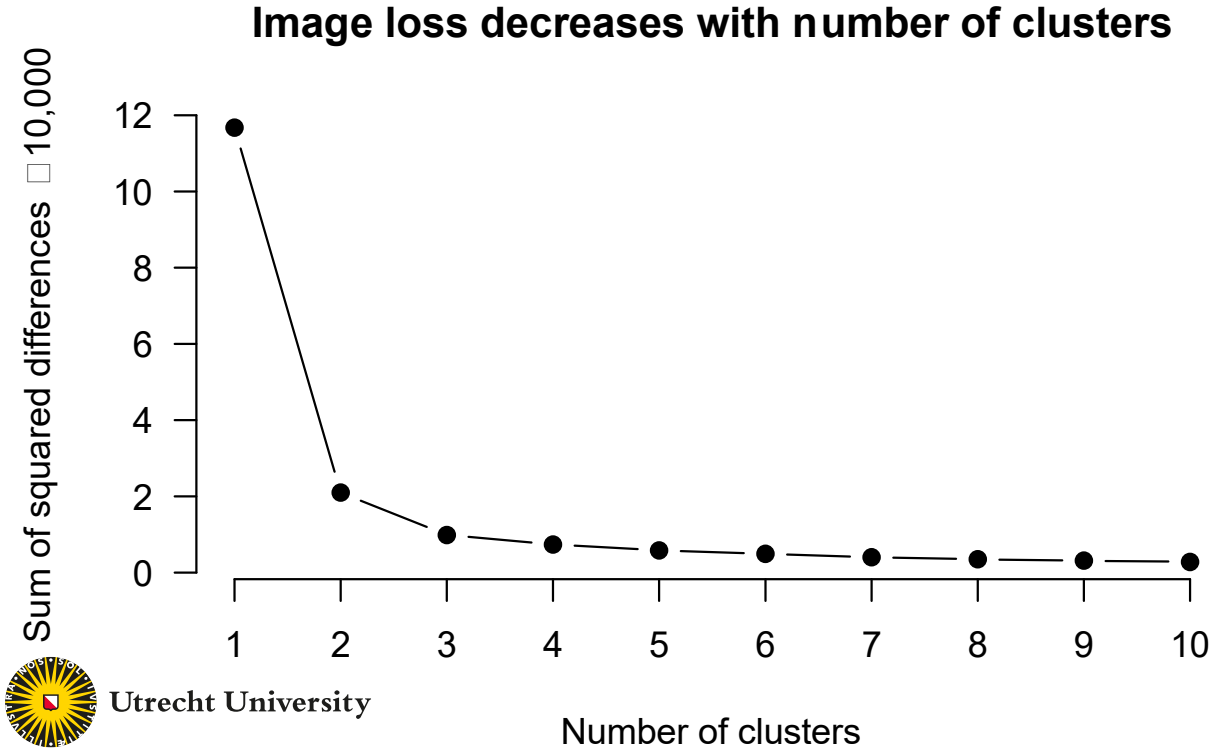


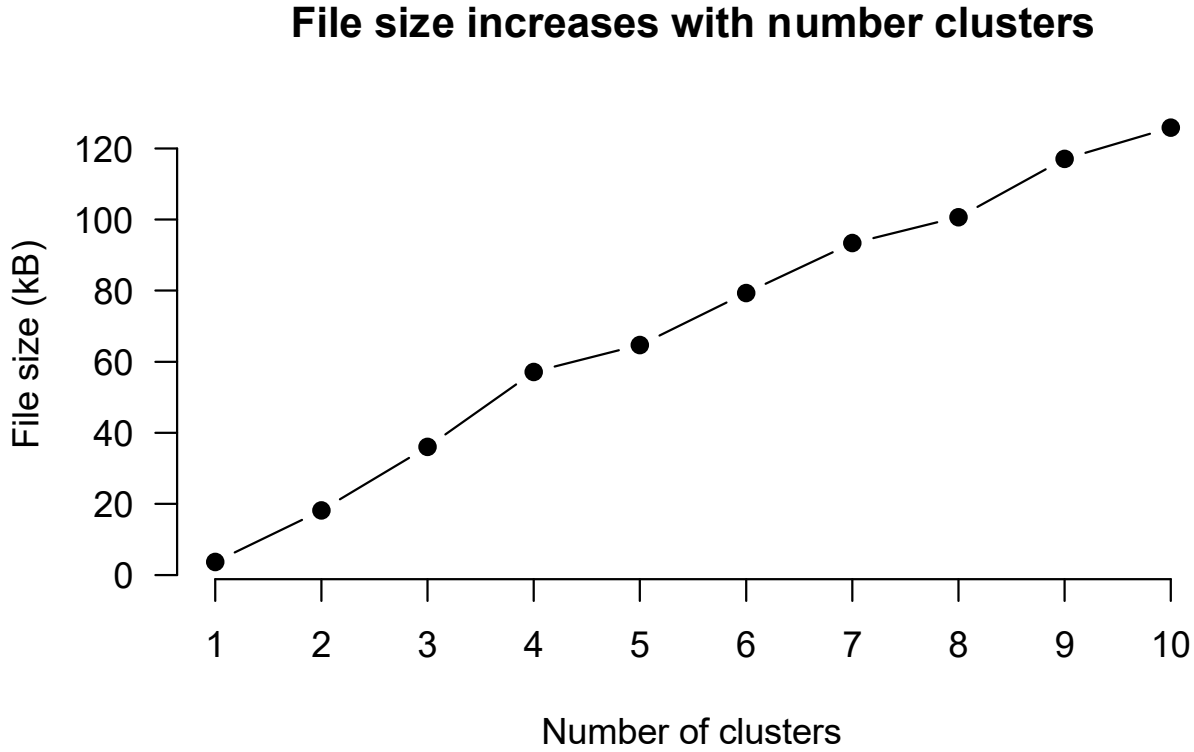
Question for you

Which of these is the best and why?

K= 1 5 10 100 640,000







The problem with learning only algorithms

- K-means is based on a **rule**
- Why this rule and not some other rule?
- What kind of data does the rule work well for?
- In what situations would the rule fail?
- What happens if we want to change the rule?

All **difficult to answer by staring at the algorithm.**

The model-based idea

Steps:

1. Pretend we believe in some *statistical model* that describes data as belonging to unobserved ("latent") groups;
2. Estimate ("train") this model using the data.

- **The rule follows from the model!**
- Instead of worrying about *algorithm*, we worry about *model*.
- Questions are easy to answer.
- Assumptions about the clusters are explicit, not implicit.
- We know how to test the model on holdout data

Heights: two hidden groups?

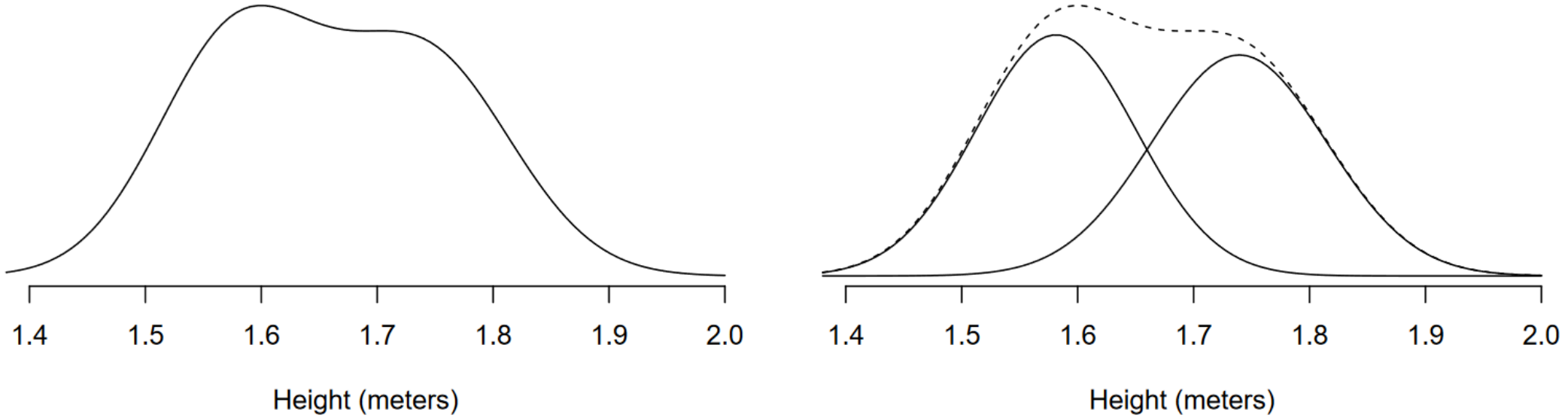


Fig. 1 Peoples' height. Left: observed distribution. Right: men and women separate, with the total shown as a dotted line.

Heights: a mixture of two normals

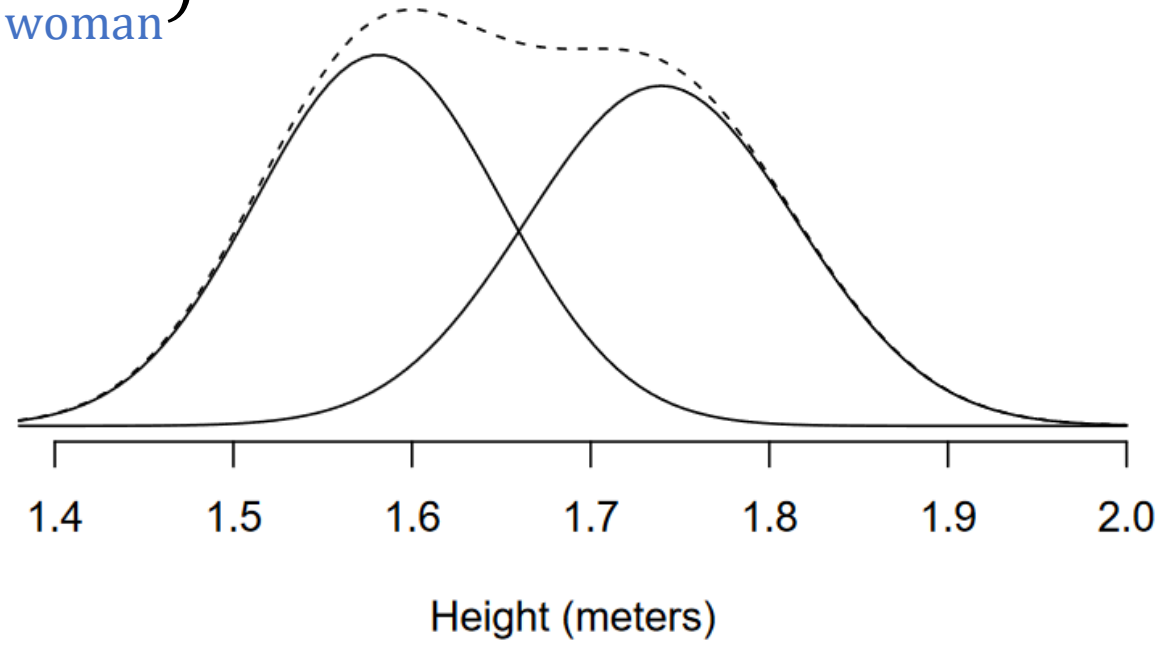
Maximum likelihood: choose the parameters that make the observed heights most probable

Likelihood (*density*) for height data:

$$p(\text{height} \mid \theta) =$$
$$\Pr(\text{man}) \cdot \text{Normal}(\mu_{\text{man}}, \sigma_{\text{man}}) +$$
$$\Pr(\text{woman}) \cdot \text{Normal}(\mu_{\text{woman}}, \sigma_{\text{woman}})$$

Or, more concise notation:

$$p(\text{height} \mid \theta) =$$
$$\pi_1^X \text{Normal}(\mu_1, \sigma_1) +$$
$$(1 - \pi_1^X) \text{Normal}(\mu_2, \sigma_2)$$



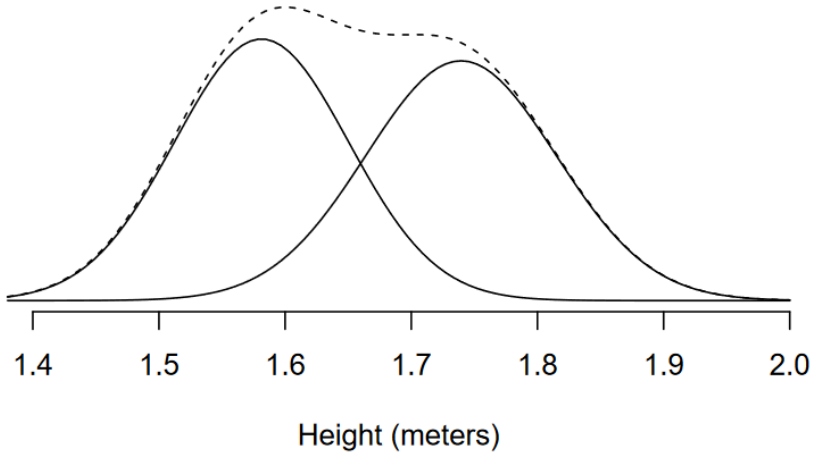
Five unknown parameters

Gaussian mixture model parameters:

- π_1 — relative cluster sizes: the share of observations in cluster 1 (cluster 2 gets $1 - \pi_1$)
- μ_1 and μ_2 — where the clusters are, like centroids in k-means
- σ_1 and σ_2 — volume: how spread out each cluster is

Together, these 5 unknown parameters describe how the data were generated

Equal variances (mclust “E”): 4 parameters; unequal (“V”): 5

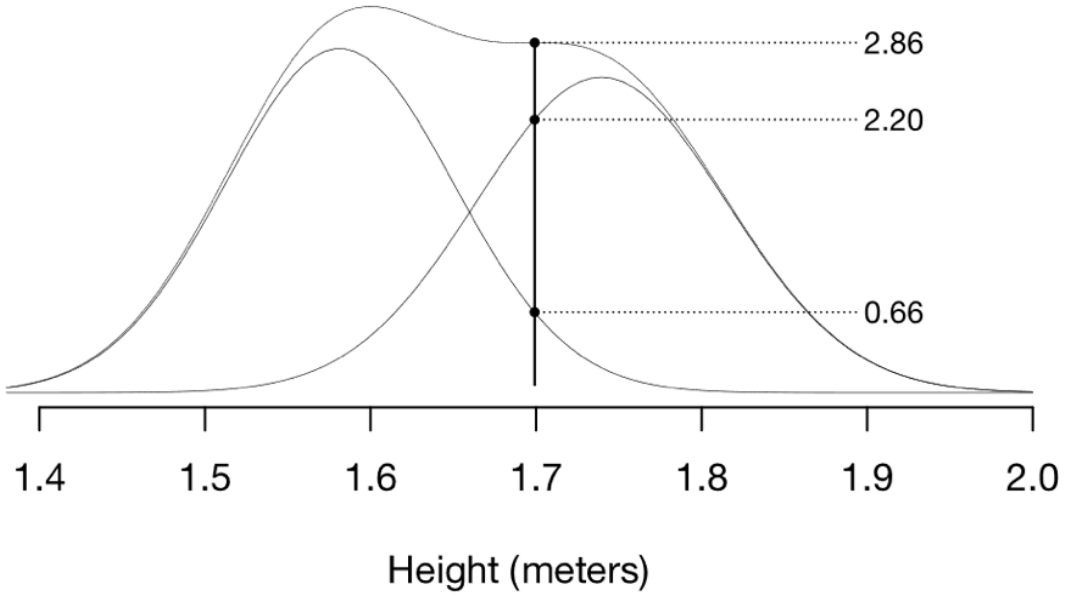


Who is a man? Posterior probabilities

- Suppose we knew the two normal curves
- How likely is it that someone of 1.70 m is a man?
- Bayes' rule: the men's curve as a share of the total at 1.70 m

P(man | 1.70 m) = 2.20 / 2.86 ≈ 0.77

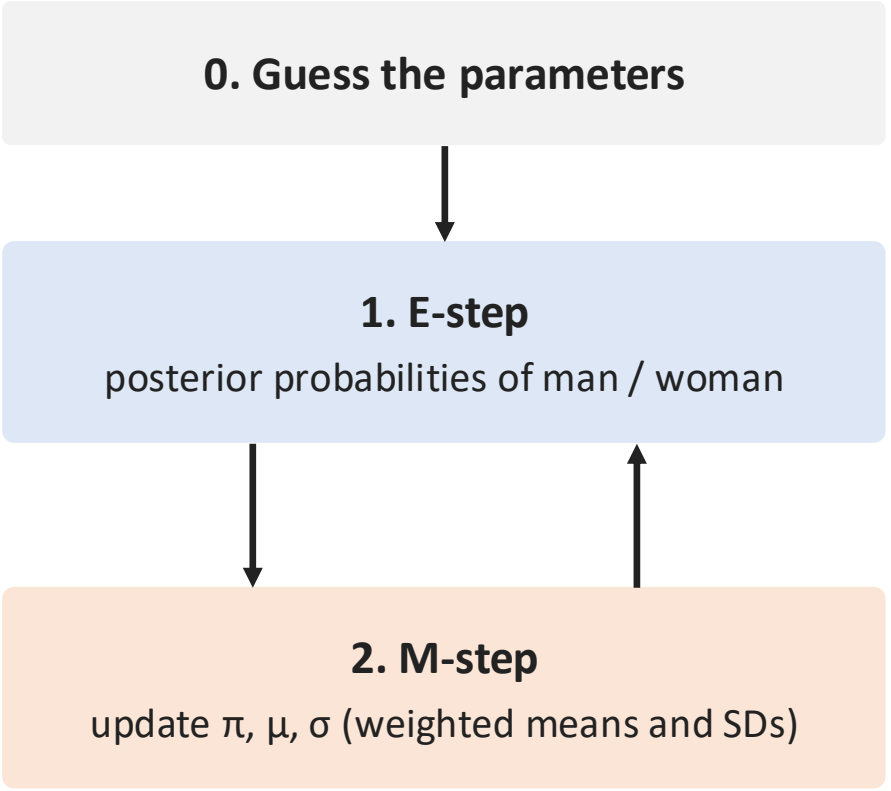
- Every observation gets a probability for every cluster: a soft assignment that shows uncertainty
- Hard assignment: the most probable cluster



Oberski (2016), Fig. 3

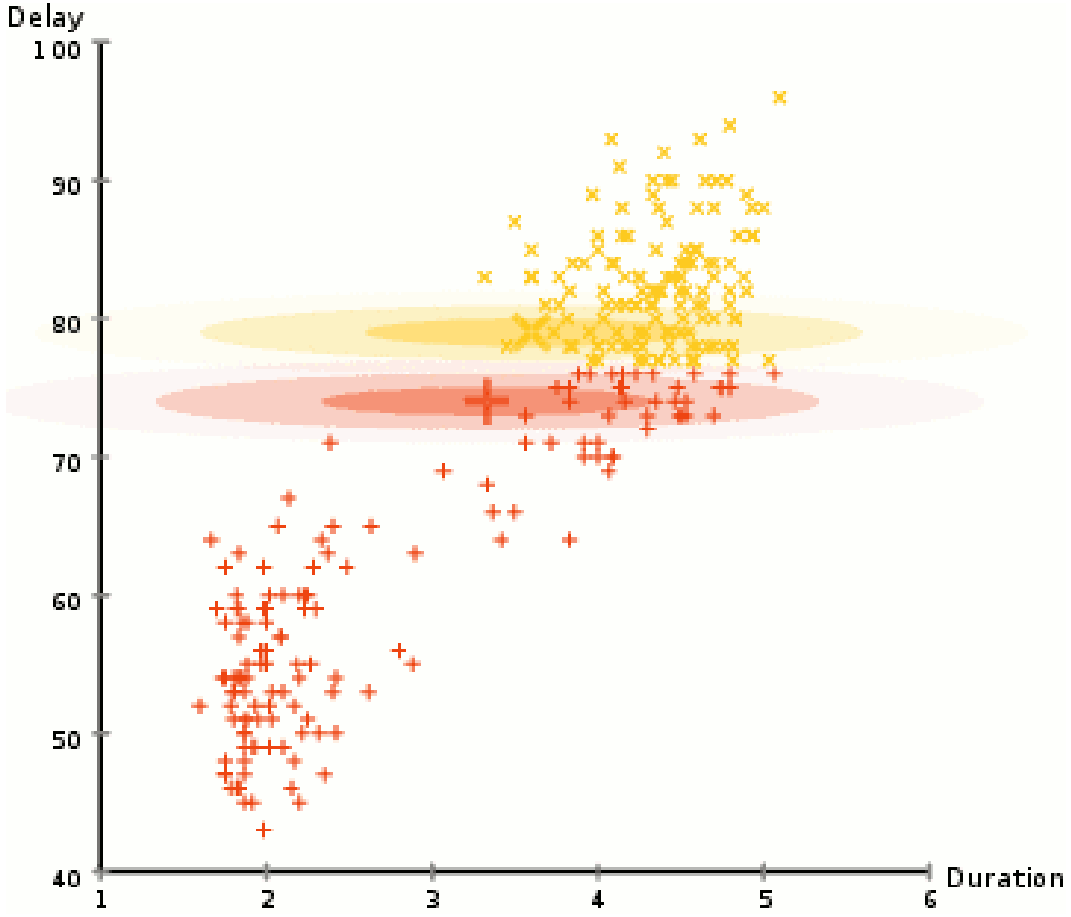
EM: guess, fill in the labels, update

- If we knew who is a man → μ and σ are just group means and SDs
- If we knew μ and σ → posterior probabilities (previous slide)
- We know neither → guess, then alternate until nothing changes
- **Remind you of something? k-means does the same, with hard assignments**
- Last week: the cluster label is a variable missing for everyone — EM was invented for incomplete data (Dempster, Laird & Rubin 1977)



stop when the parameters stop changing

EM in action (Old Faithful)

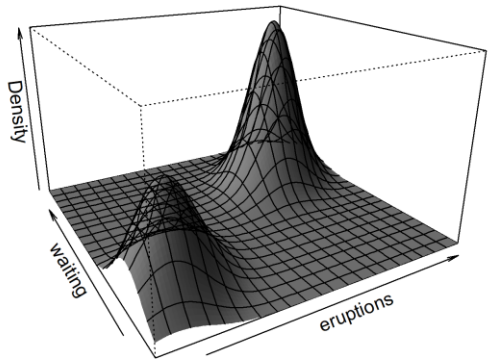


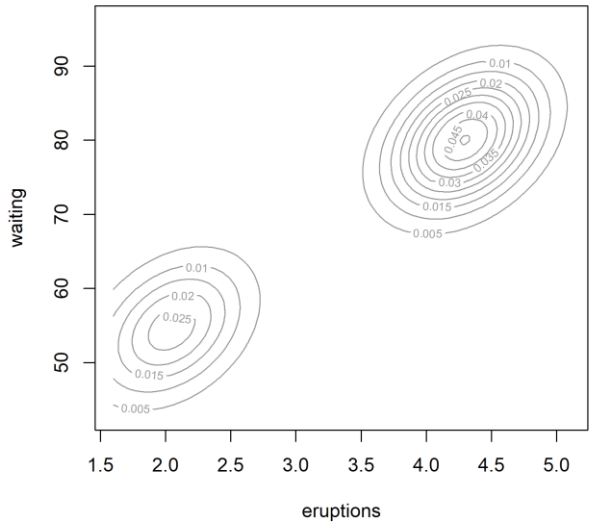
By Chire - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=20494862

Several features → ellipses

- With 2 features, each cluster has
  - a mean vector: 2 means
  - a 2 × 2 covariance matrix: 2 variances + 1 covariance
- Seen from above, each cluster's density is an ellipse
- The covariance matrix sets its size, shape and orientation

$$p(y \mid \theta) = \pi_1 \cdot MVN(\mu_1, \Sigma_1) + (1 - \pi_1) \cdot MVN(\mu_2, \Sigma_2)$$





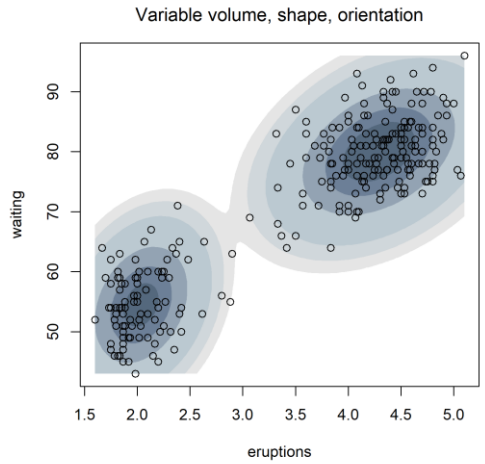
Volume, shape and orientation

- Each cluster's ellipse has a volume (size), a shape (round or elongated) and an orientation (angle)
- mclust: one letter for each — E = equal across clusters, V = variable, I = identity (round / axis-aligned)
- EEE: all the same ellipse · VVV: each its own · VVE: same orientation only
- **Equal = simpler; variable = more flexible → bias–variance**
- Latent profile analysis (reading): features unrelated within a cluster = the diagonal models (EEI, VVI, …)

EEE



VVV



k-means: a special Gaussian mixture

k-means = a Gaussian mixture in which

- all cluster proportions are 1/K;
- **`EII`** model: equal volume, only circles;
- all posteriors are either 0 or 1 (“classification likelihood”).
- **So k-means assumes round clusters of similar size and spread: that is when it works, and when it fails.**

BIC: fit + a penalty for complexity

$$\mathbf{BIC = -2 \cdot \log L + m \cdot \log(n)}$$

training-data fit (deviance) model capacity (penalty)

- L: likelihood, p(data | θ) — how well the model fits
- m: number of free parameters; n: number of observations
- Lower is better here — think bias–variance
- Heights: equal variances (E) m = 4, unequal (V) m = 5 (mclust: fit$df)

Careful with the sign!

mclust reports $2 \cdot \log L - m \cdot \log(n)$

→ **higher is better (its BIC plots go up)**

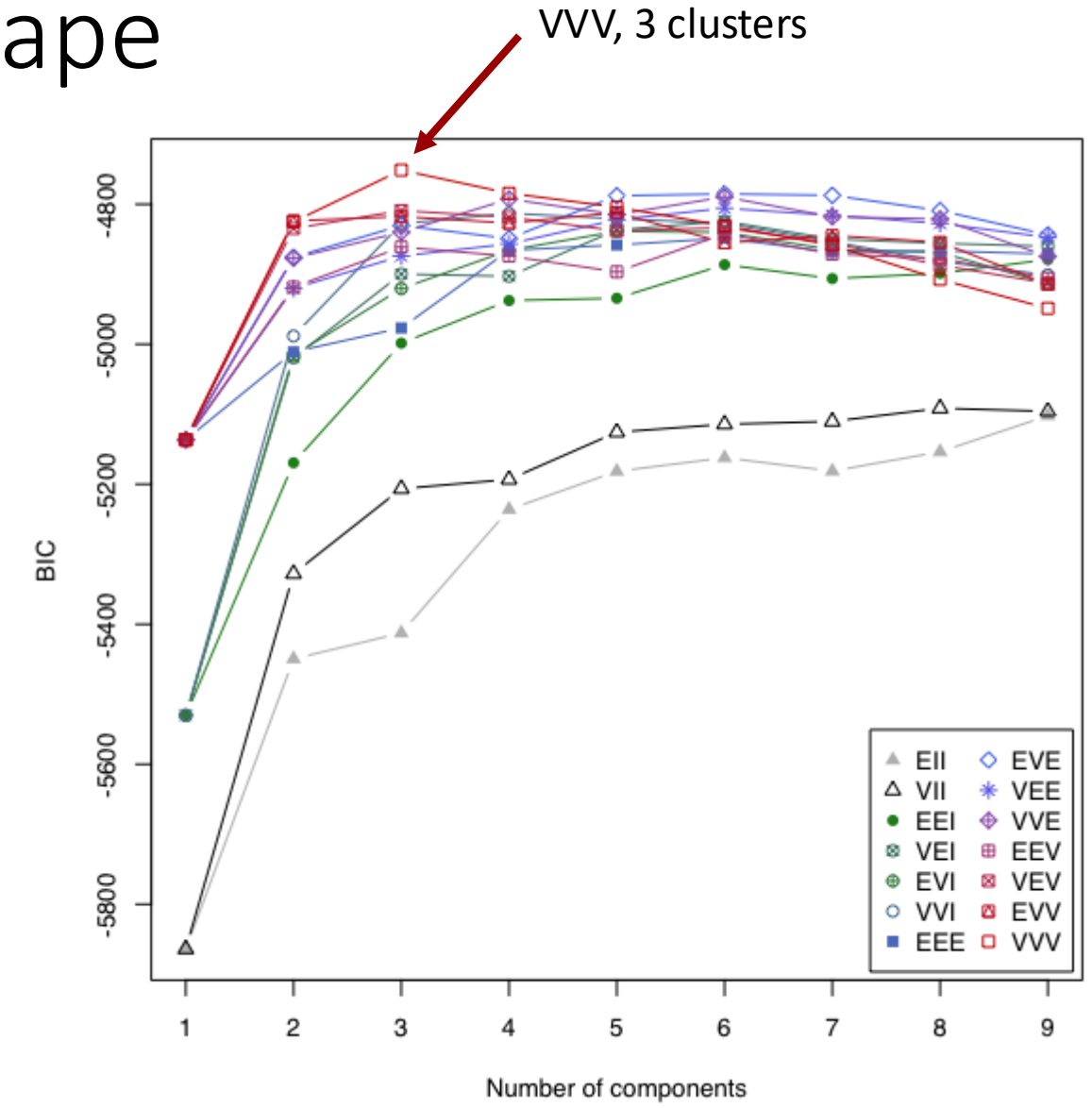
sklearn GaussianMixture.bic(X)

→ **lower is better**

BIC chooses K and the shape

- How `mclust` optimizes hyperparameters:
  - Fit all the models with up to 9 clusters (or more, your choice!)
  - Compute the BIC (or ICL) of each model
  - Choose the model with the highest BIC (mclust's sign)

- G = 1 is included: is there any cluster structure at all? (Part D)
- Lab Thursday: mclust on the Swiss banknote data



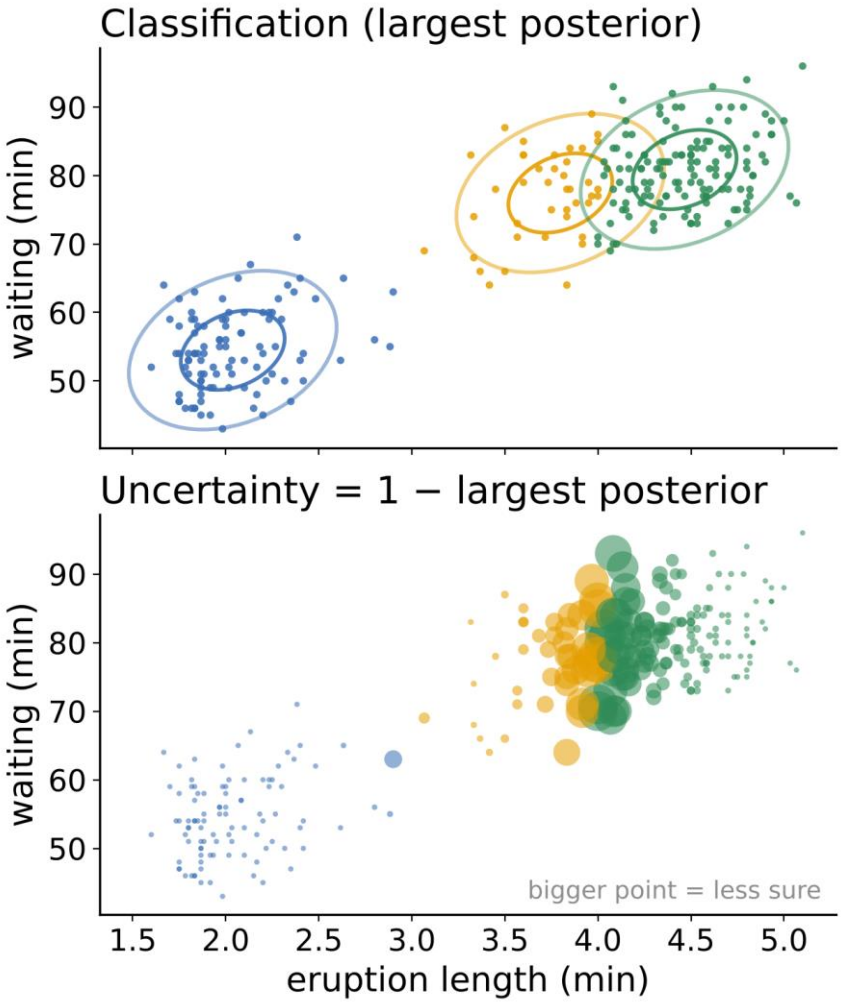
mclust in R (and Python)

R

```
library(mclust)
fit <- Mclust(x, G = 1:9)    # 1-9 clusters
summary(fit, parameters = TRUE)
plot(fit, what = "BIC")
plot(fit, what = "classification")
plot(fit, what = "uncertainty")
fit$z              # posterior probabilities
fit$uncertainty    # 1 - largest posterior
```

Python

```
from sklearn.mixture import GaussianMixture
gm = GaussianMixture(n_components=3,
      covariance_type="tied").fit(X)  # ~EEE
gm.predict_proba(X)    # posteriors
gm.bic(X)              # lower = better!
```



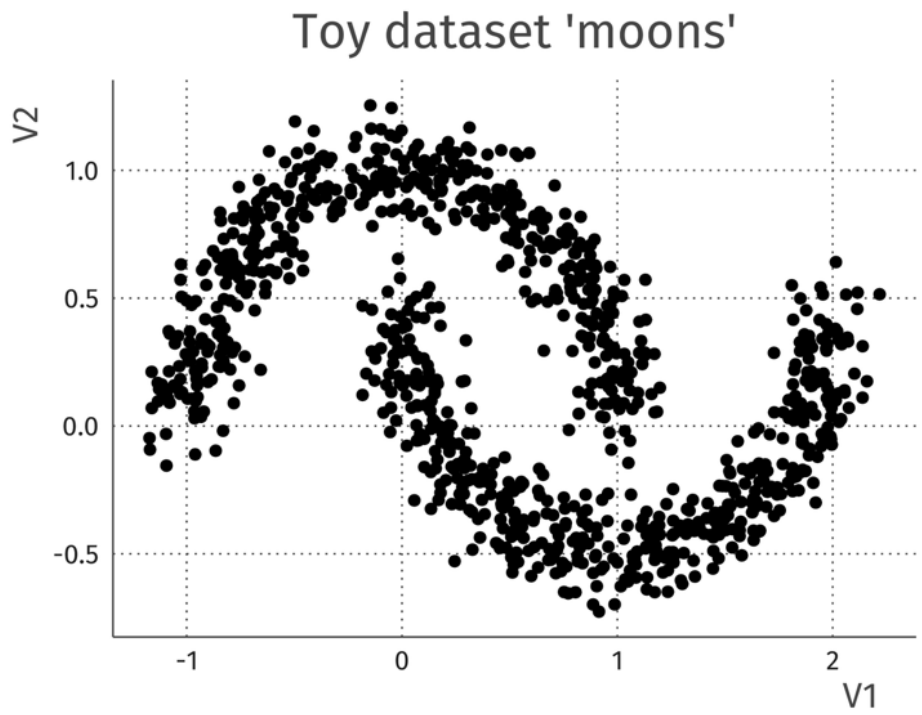
Non-elliptical clusters: merging

- GMM obviously has trouble with clusters that are not ellipses
- Secret weapon: **merging**

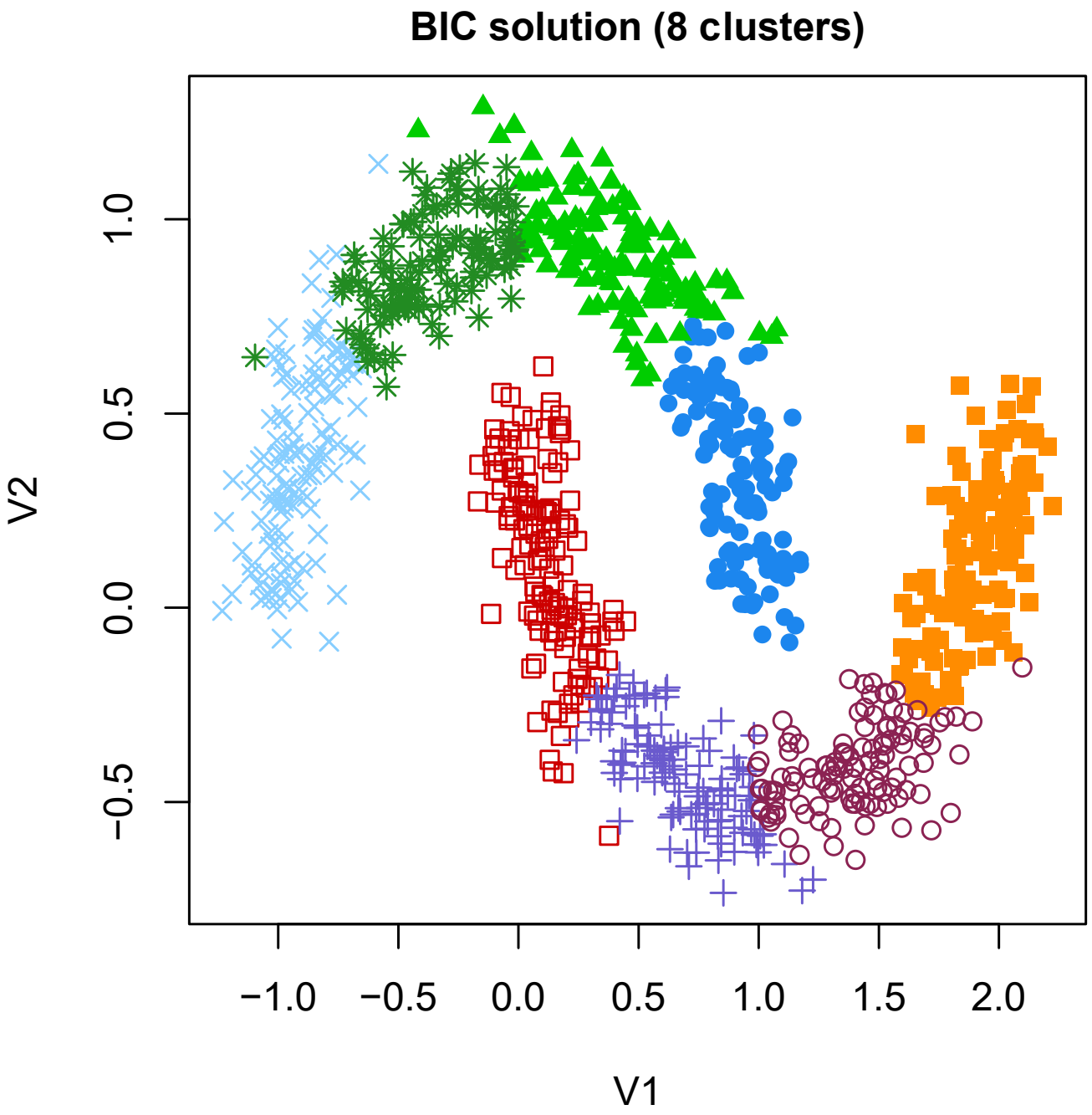
Powerful idea:

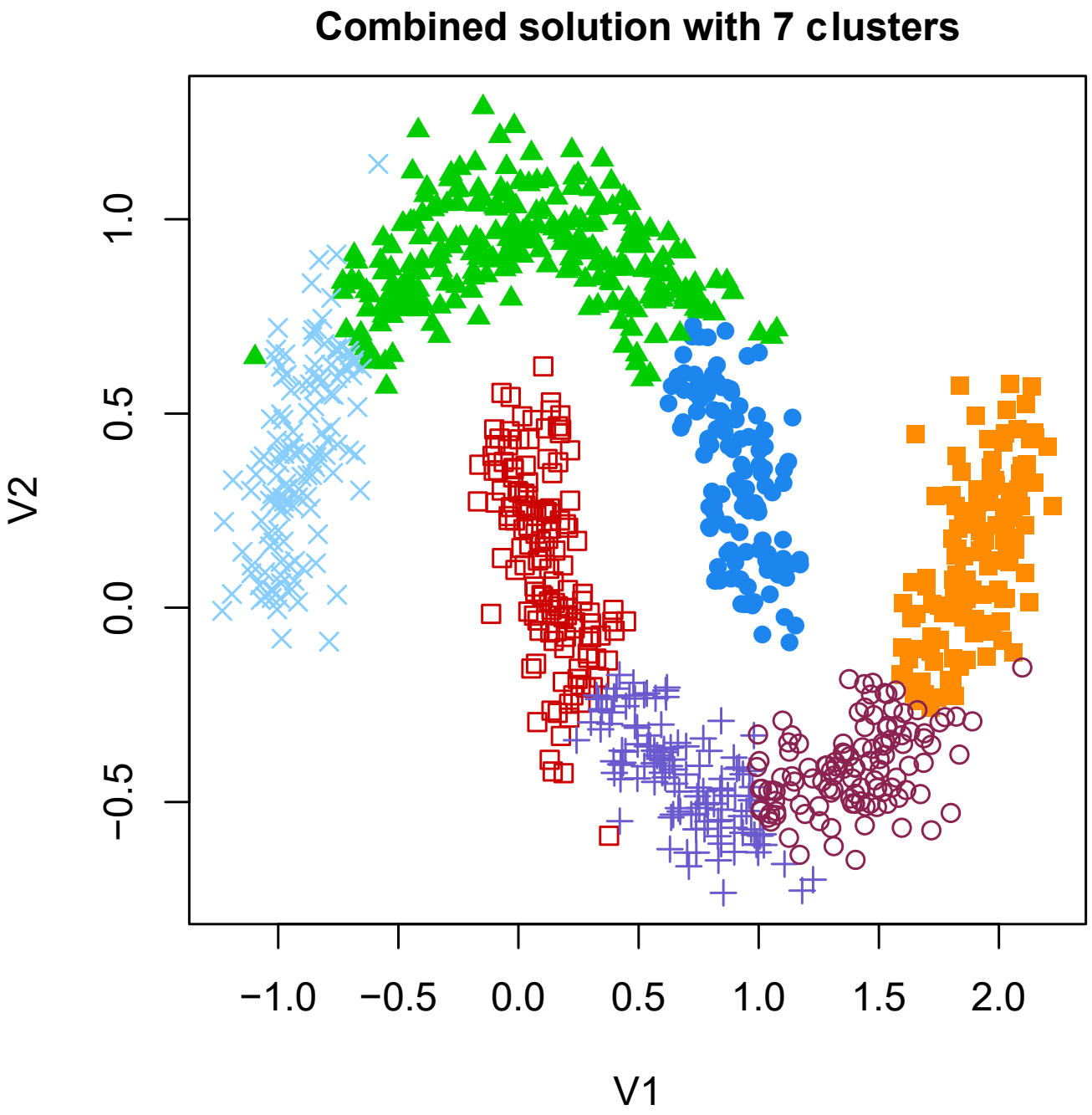
- Start out with the usual Gaussian mixture solution;
- **merge** "similar" *components* to create non-Gaussian *clusters.*

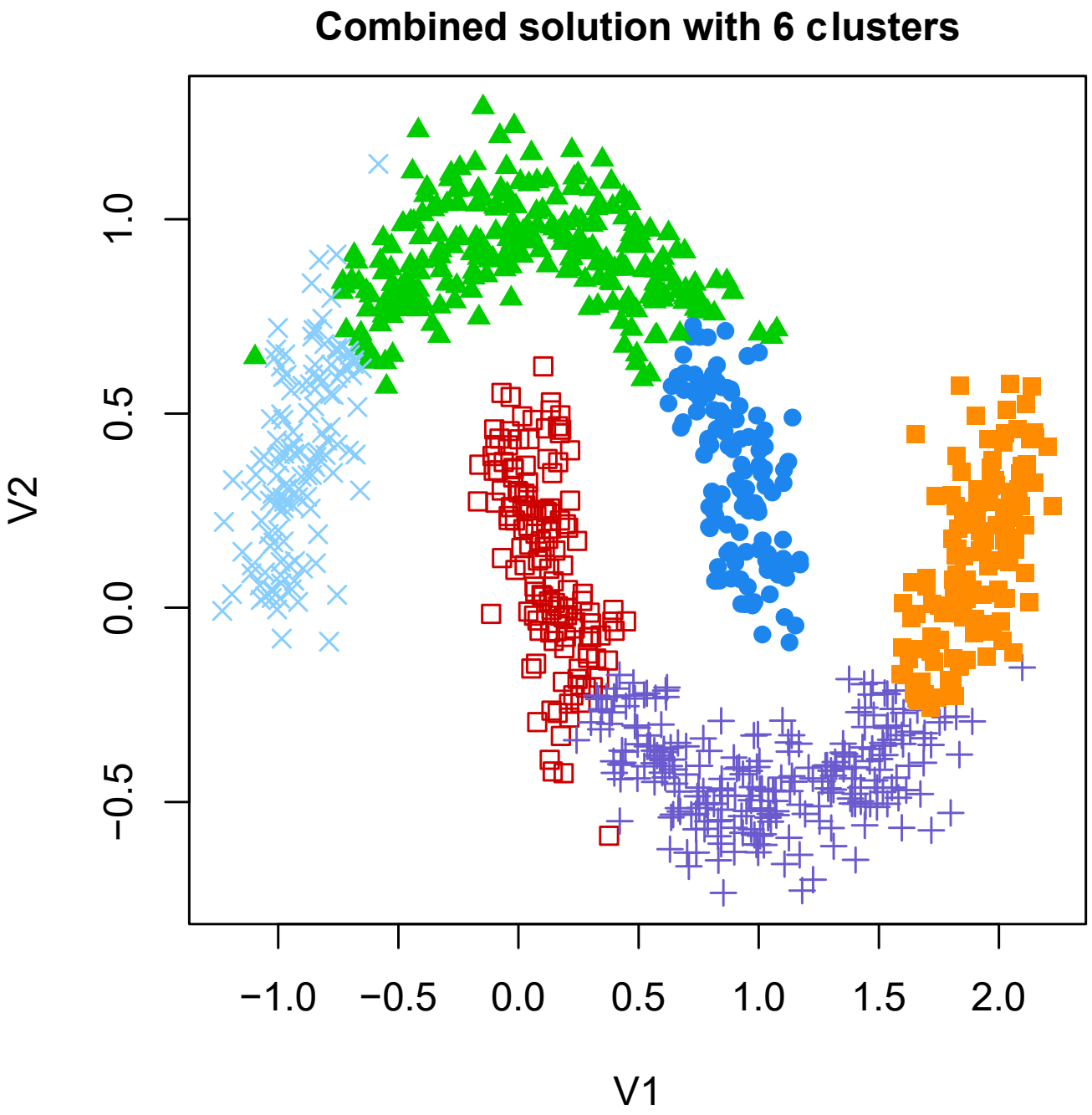
Note: we're distinguishing "components" from "clusters" now.

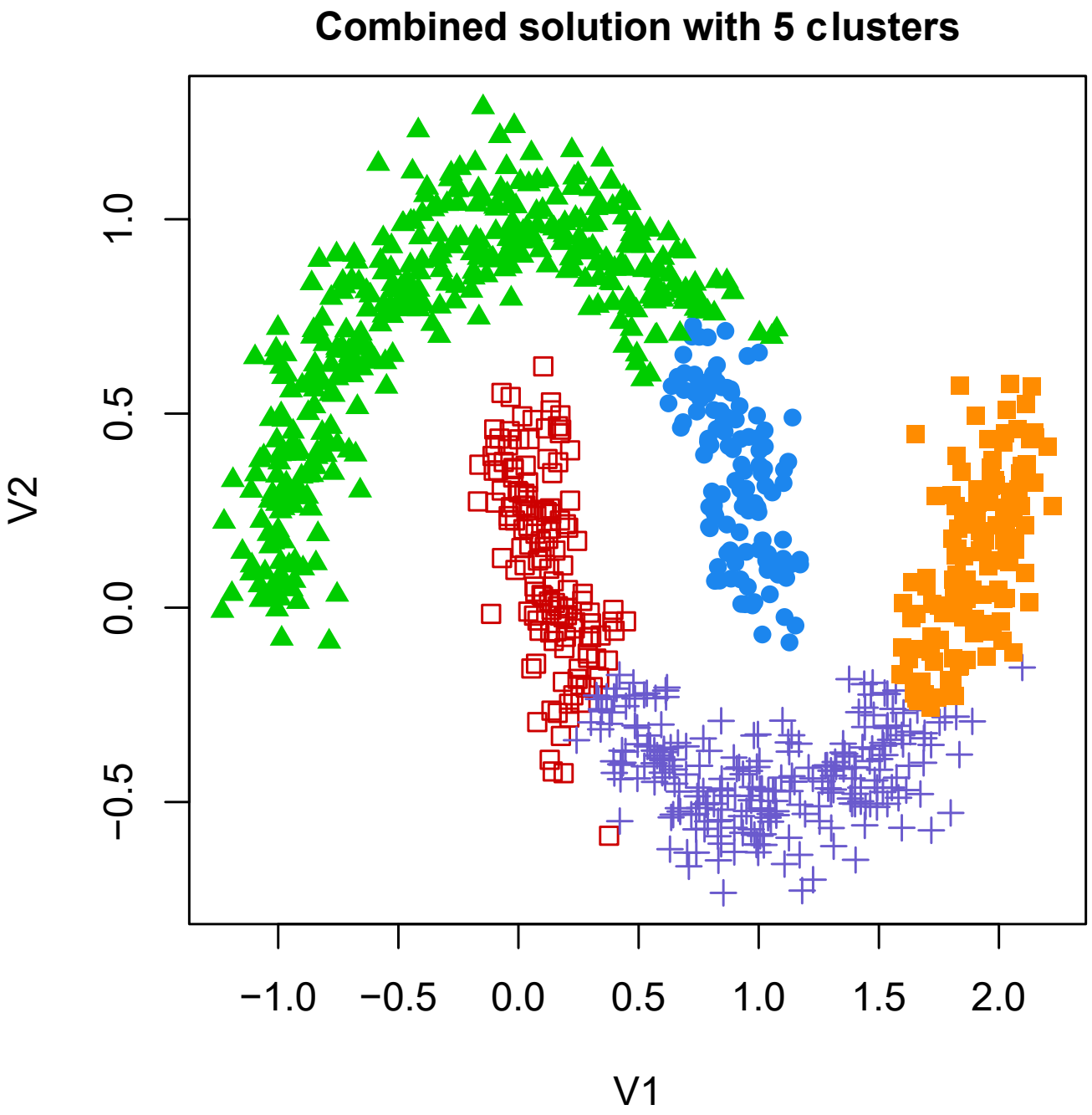


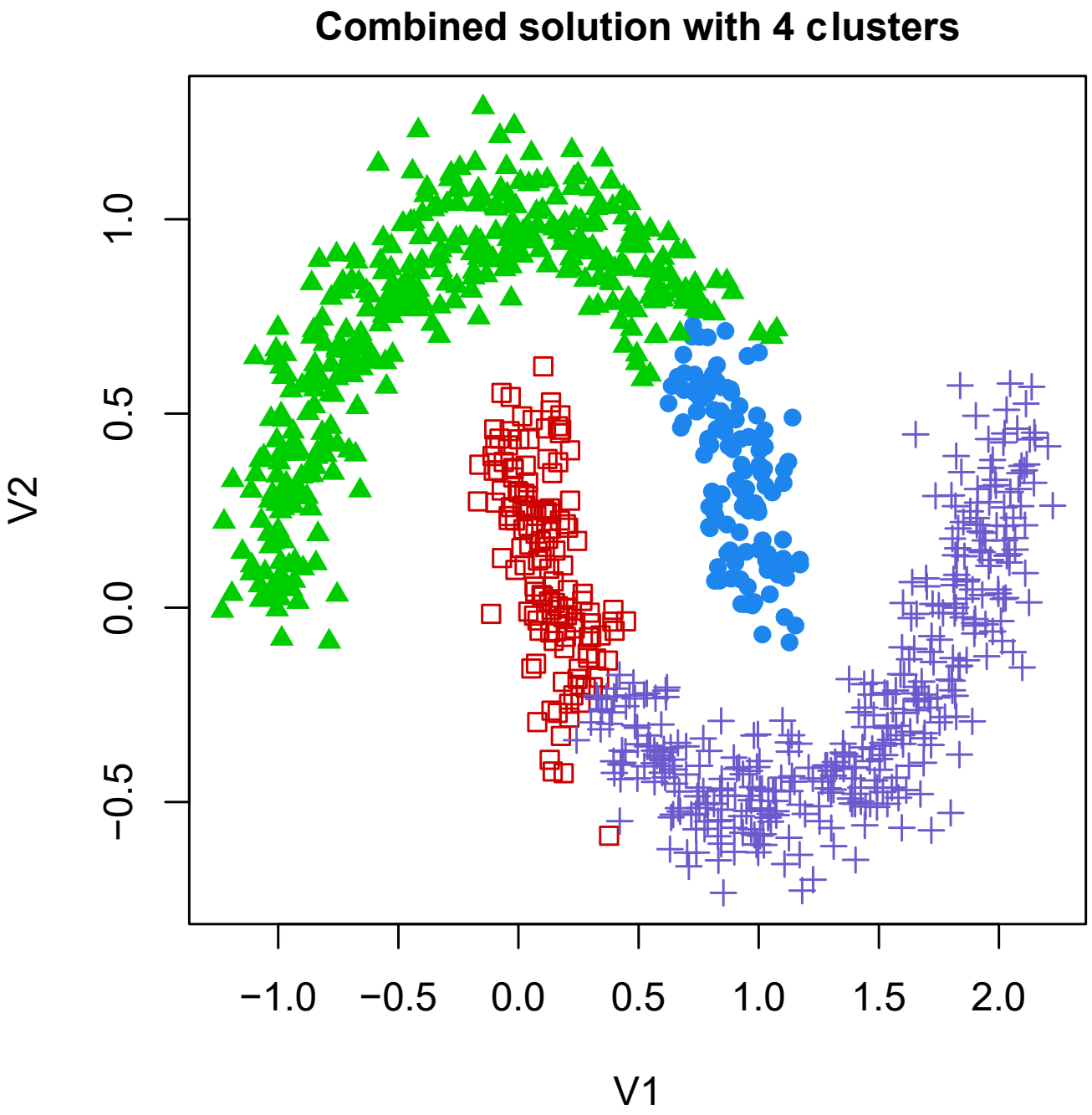

```
library(mclust)
output <- clustCombi(data = x)
plot(output)
```

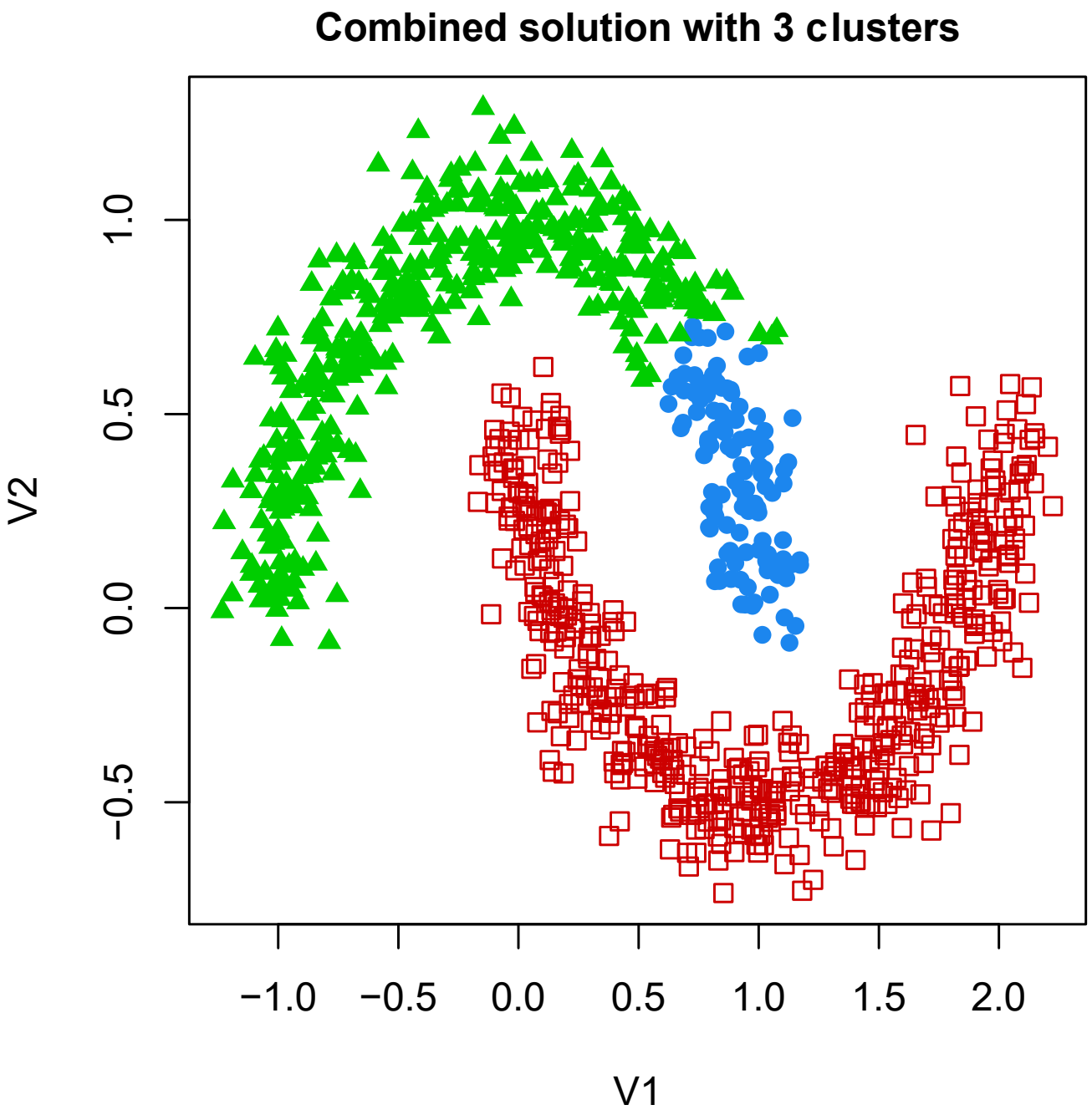

BIC solution (8 clusters)
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1

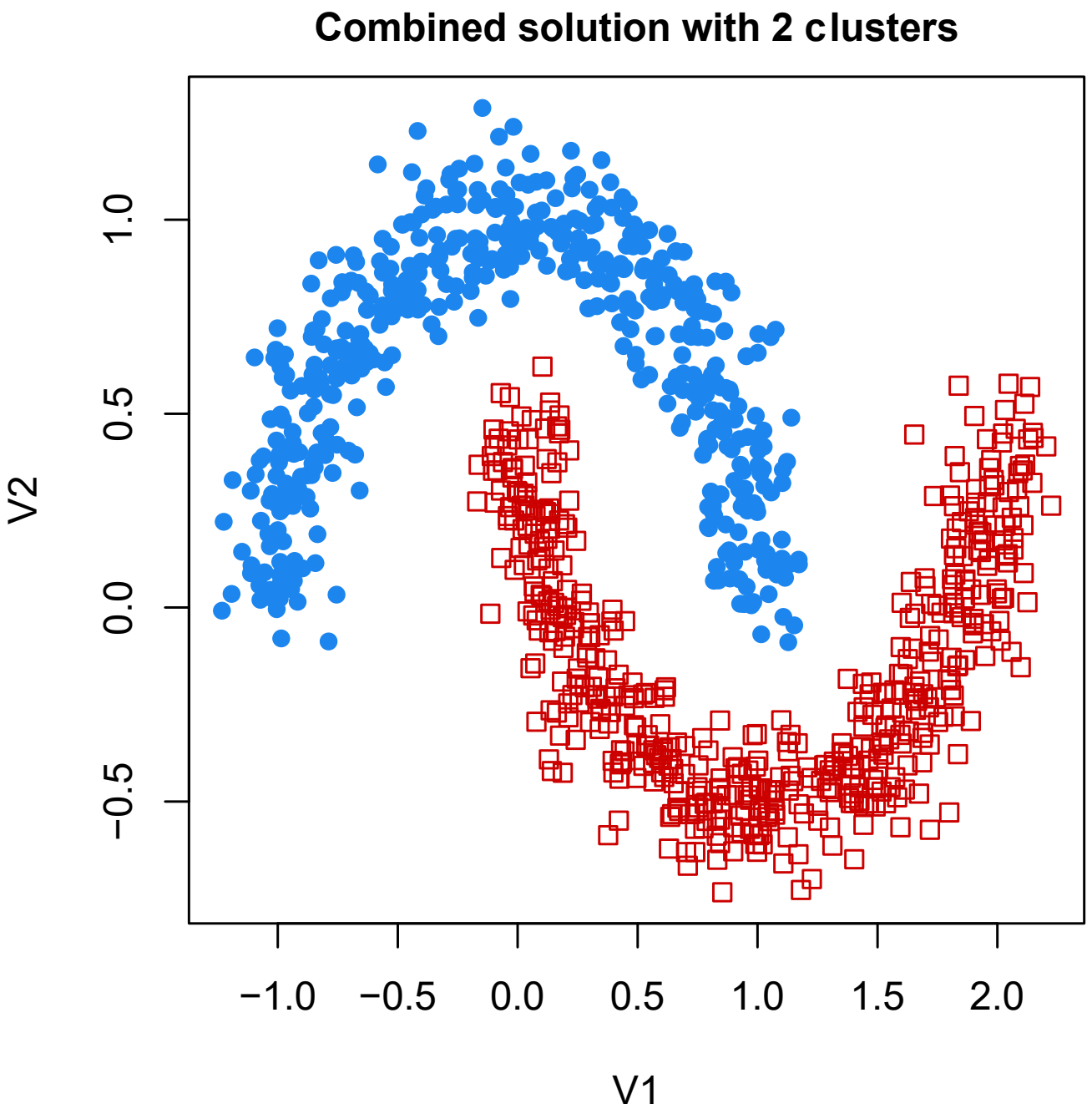

Combined solution with 7 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1

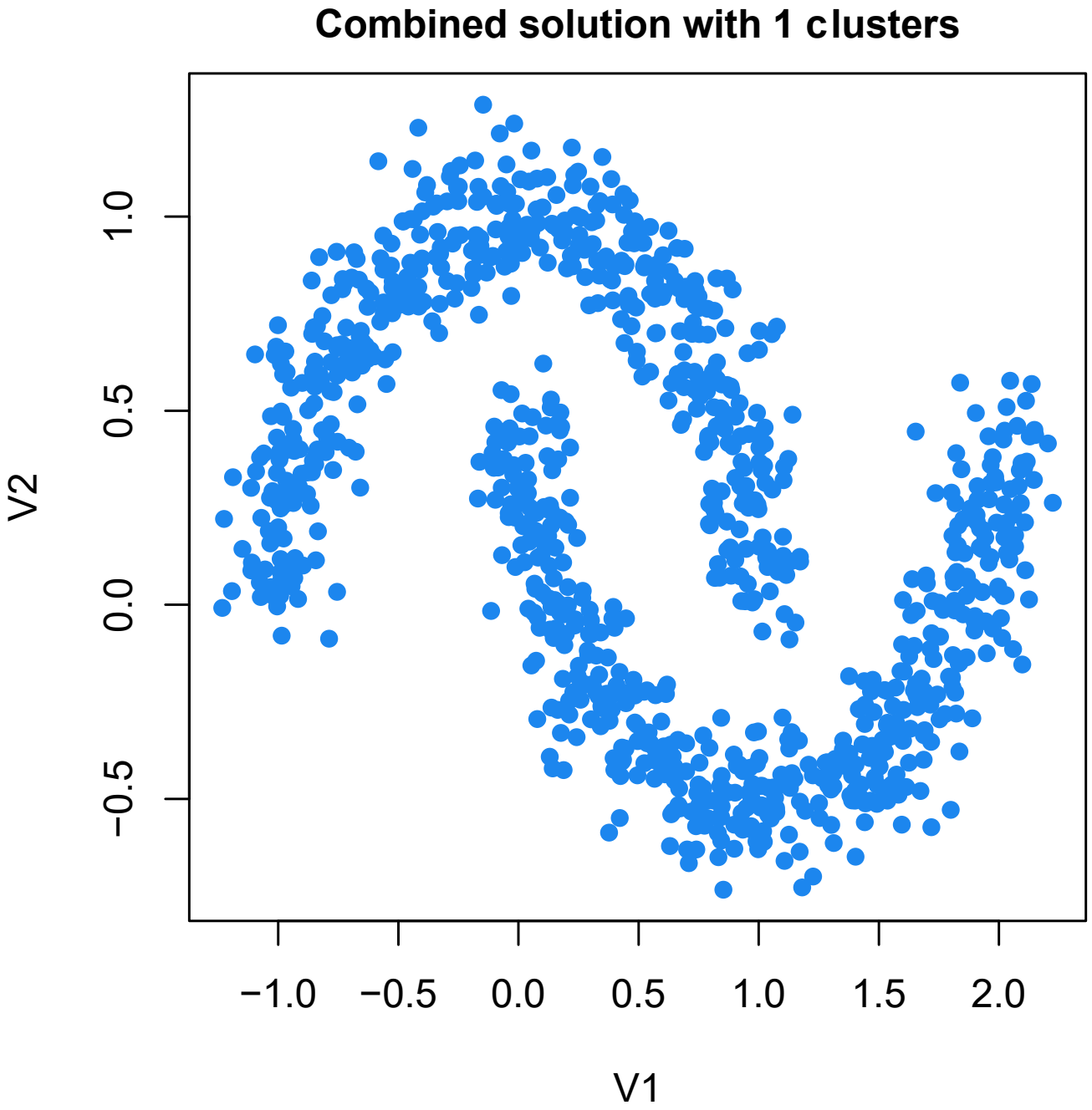

Combined solution with 6 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1


Combined solution with 5 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1


Combined solution with 4 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1


Combined solution with 3 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1


Combined solution with 2 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1


Combined solution with 1 clusters
V2
1.0
0.5
0.0
−0.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
V1

How to **evaluate** clustering results

1. Use of external information
2. Visual exploration
3. Stability assessment / sensitivity analysis
4. Internal validation indexes
5. Testing for clustering structure

Much more info & helpful advice: Clustering strategy & method selection (ch 31 of Handbook of clustering), https://arxiv.org/pdf/1503.02059.pdf

1. External validation

- **Are the clusters associated with a feature Y the clustering did not use?**

"Making unsupervised supervised"

- Customer segments vs demographics; eruption types vs water pressure; can you recognise the person in the compressed picture?
- Known labels? Cross-tabulate, and compute the adjusted Rand index (ARI): 0 = chance, 1 = identical
- **Careful: the label need not be what the clusters capture (Assignment 2: sentiment vs genre)**

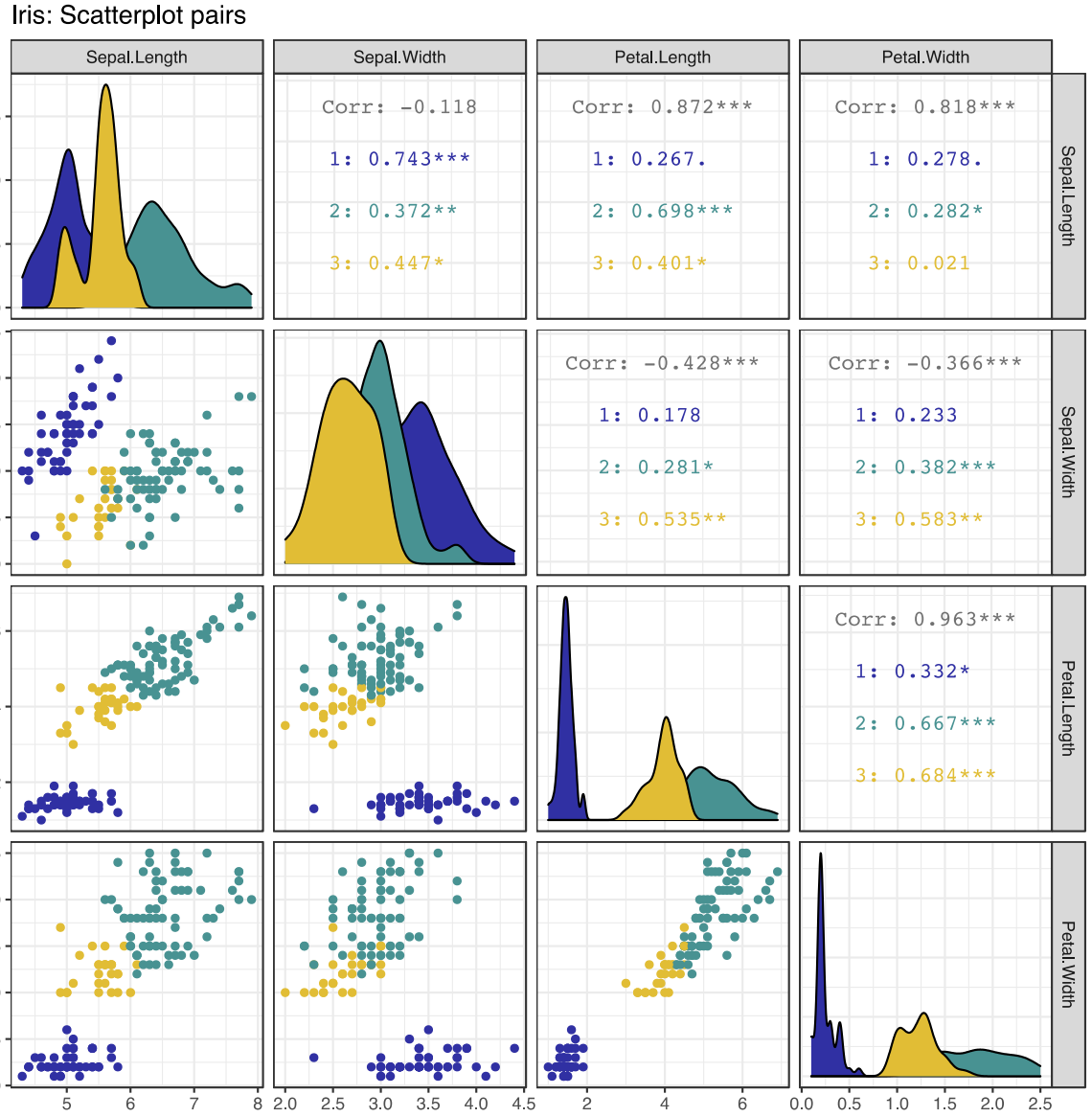
k-means cluster	setosa	versicolor	virginica
1	50	0	0
2	0	48	14
3	0	2	36

Iris, k-means (K = 3) vs species: ARI = 0.73

Gaussian mixture (VVV): ARI = 0.90

```
R:  mclust::adjustedRandIndex(cl, y)
Python:  adjusted_rand_score(y, labels)
```

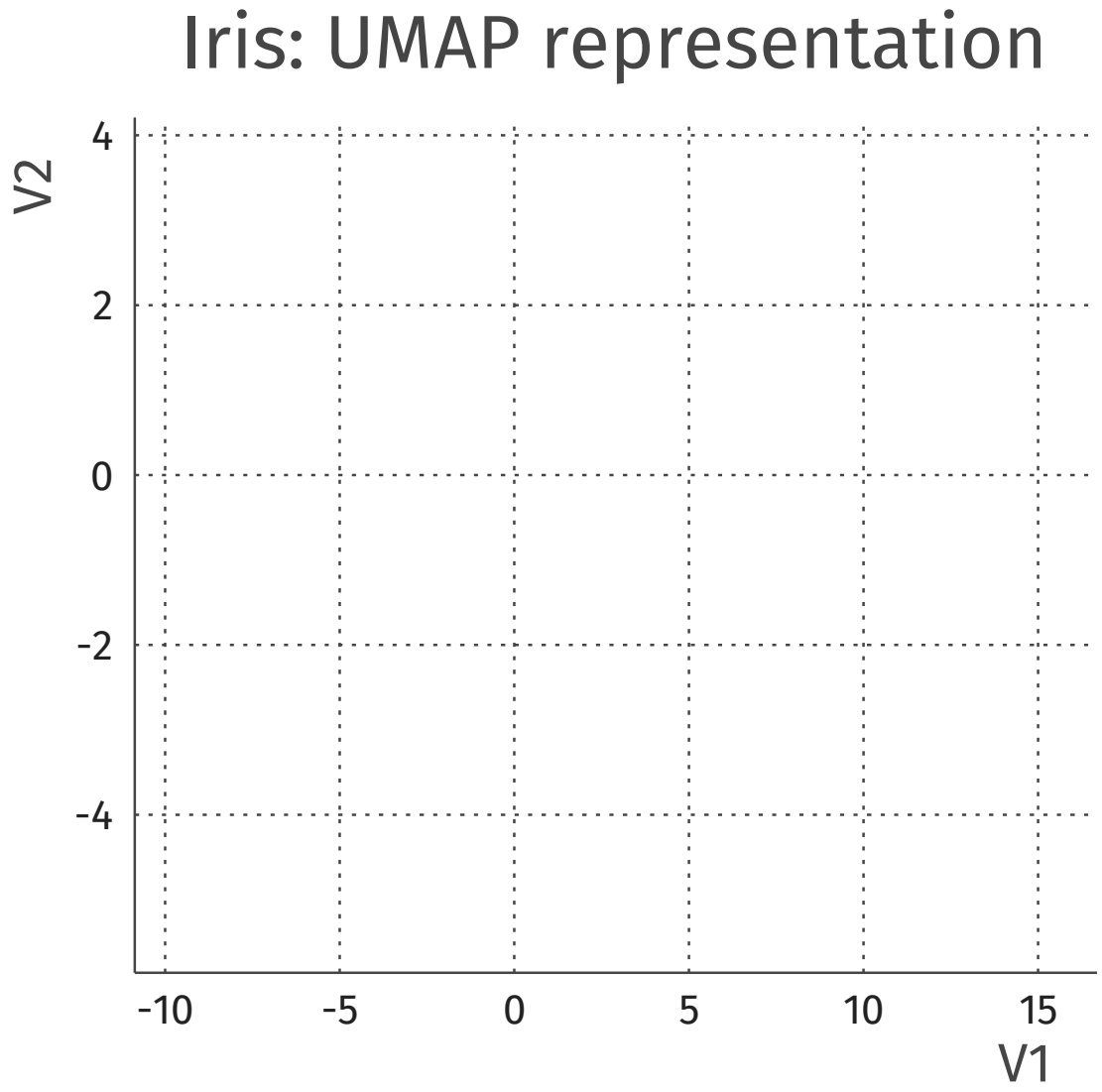
2. Visual exploration



- **Problem**: Kind of hard to see already…
- Wait till you get 1000 variables!

- **New idea**: Reduce variables into 2D “manifold” for visualization
- Popular techniques: UMAP, t-SNE, MDS, Discriminant Coordinates, PCA (Part A)

2. Visual exploration (using “manifold”)

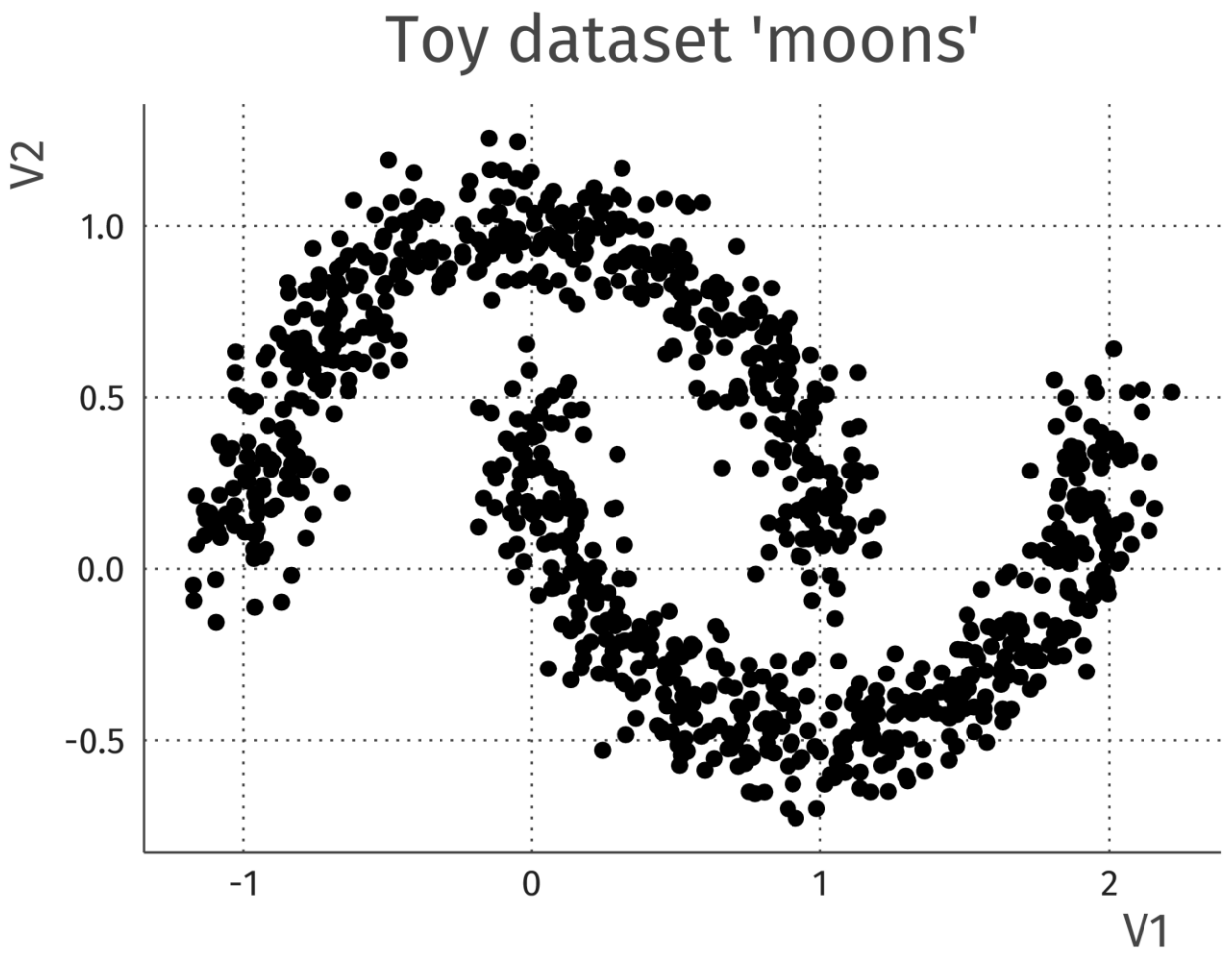


Careful

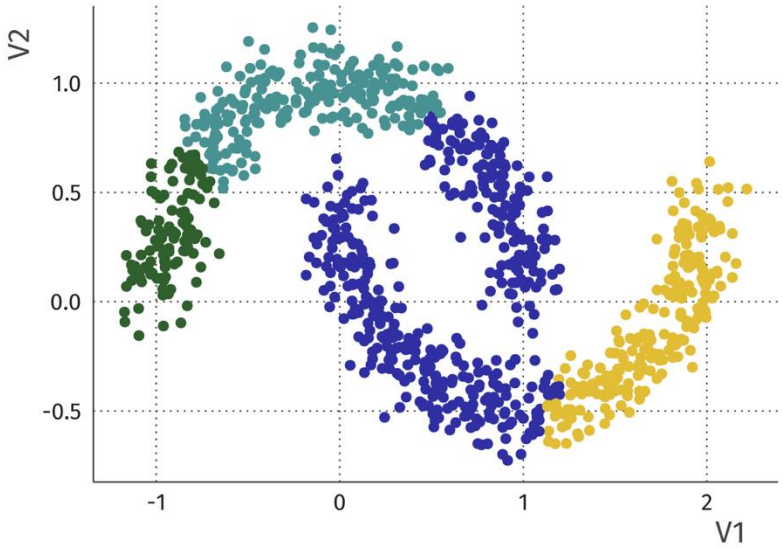
t-SNE / UMAP maps can suggest clusters that aren’t there, and distances between clusters mean little

3. Stability assessment

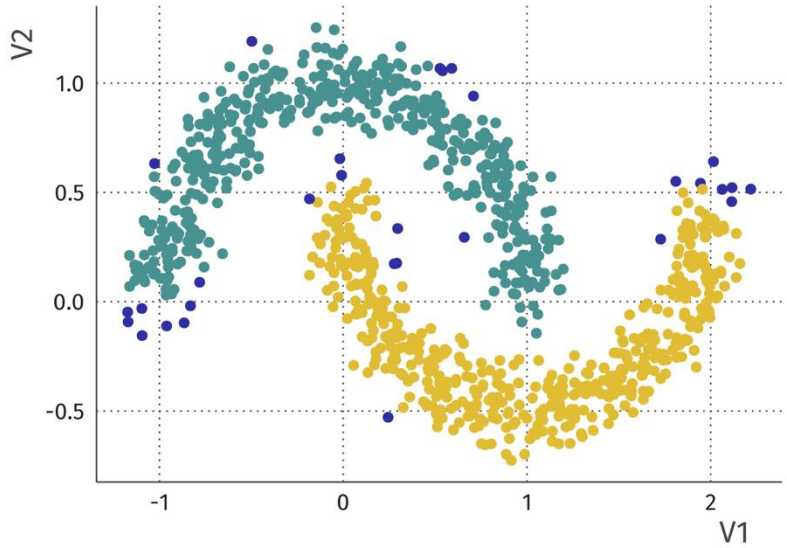
A.k.a.: Clustering can be fiddly



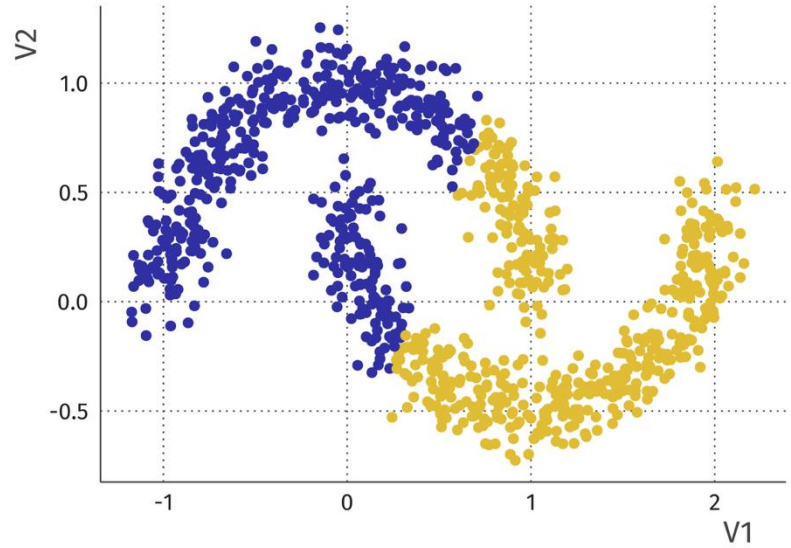
hclust [default settings, 2 cluster cutoff]



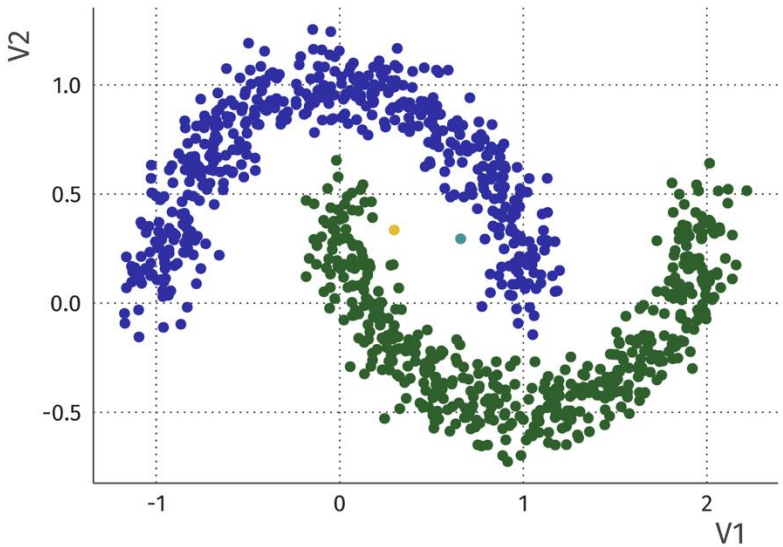
dbscan [noise set to true level, 0.1]



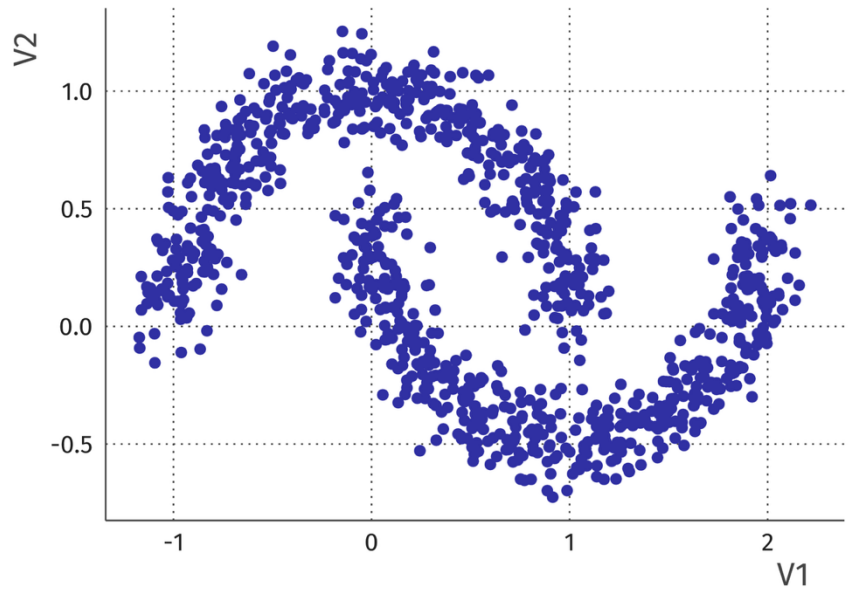
kmeans [K = 2]



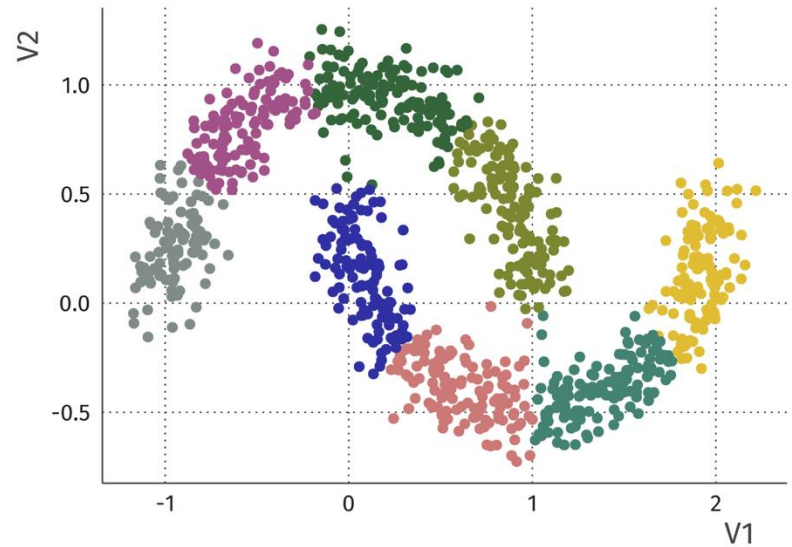
hclust [single linkage, 4 cluster cutoff]



dbscan [noise set to 0.15]



kmeans [K = 8]



3. Three kinds of stability

How much does the clustering change when you change…

- the hyperparameters (distance, linkage, K, …)
- the observations (bootstrap resamples; Hennig 2007)
- the features
- Compare the clusterings with the Jaccard or (adjusted) Rand index

```R
library(fpc)
clusterboot(iris[, 1:4],
            clustermethod = hclustCBI,
            method = "complete", k = 3)
```

```
Clusterwise Jaccard bootstrap mean:
[1] 0.891 0.459 0.719
```

Reading the Jaccard means (Hennig)

≥ 0.85 highly stable · ≥ 0.75 valid

0.6–0.75 a pattern, membership doubtful · < 0.6 don't trust it

4. Internal validation indices

Only the data and the clustering — no labels

- Silhouette: a = mean distance to own cluster, b = to the nearest other; s = (b − a) / max(a, b)
  - +1 well inside · 0 on the border · < 0 probably misassigned
  - average silhouette width (ASW): > 0.7 strong, 0.5–0.7 reasonable, 0.25–0.5 weak, < 0.25 none
- Davies–Bouldin: each cluster's spread relative to its distance from the most similar other cluster, averaged — lower is better

Both reward compact, round, well-separated clusters (may not be your aim)

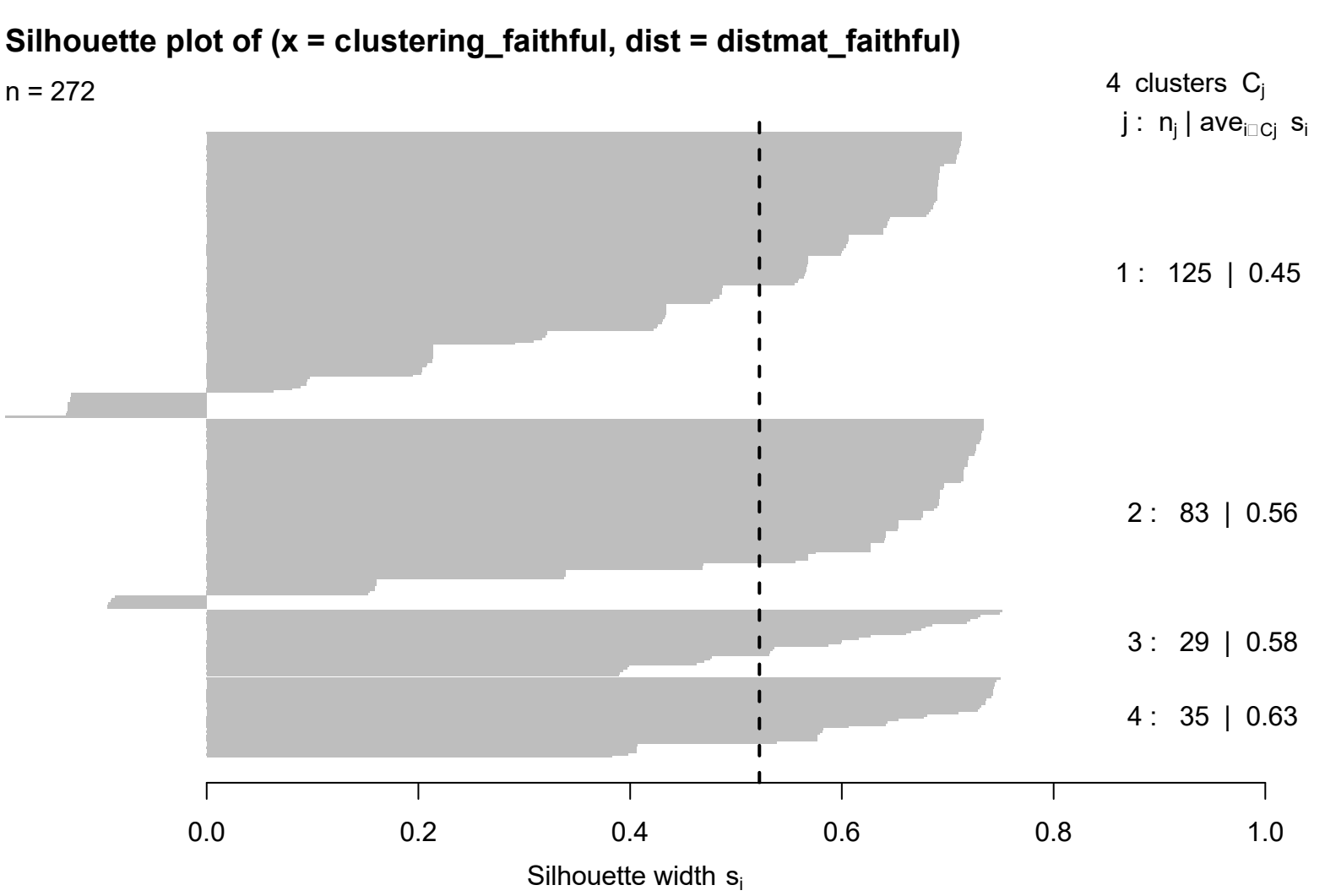
R

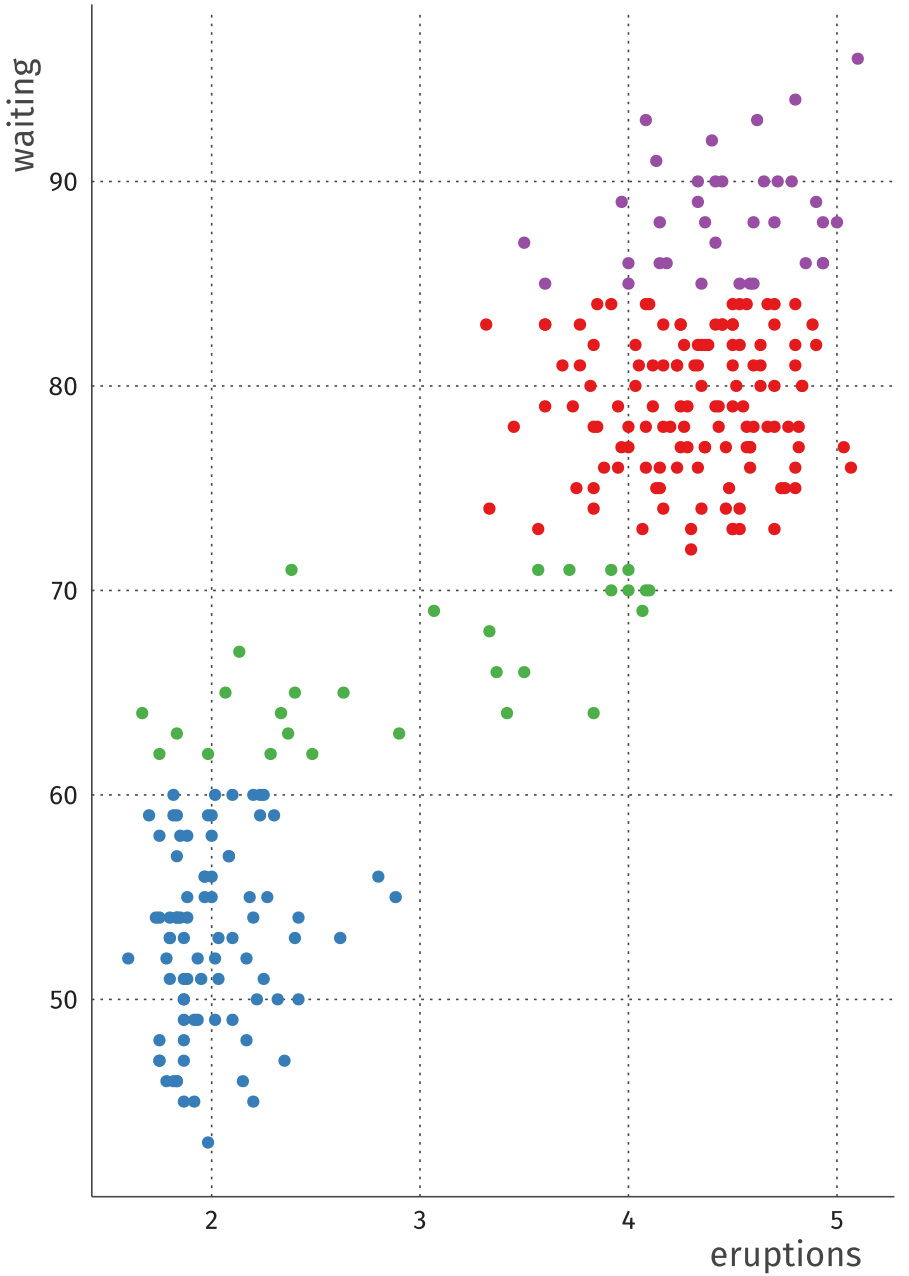
```r
library(cluster)
sil <- silhouette(cl, dist(x))
summary(sil)$avg.width
plot(sil)

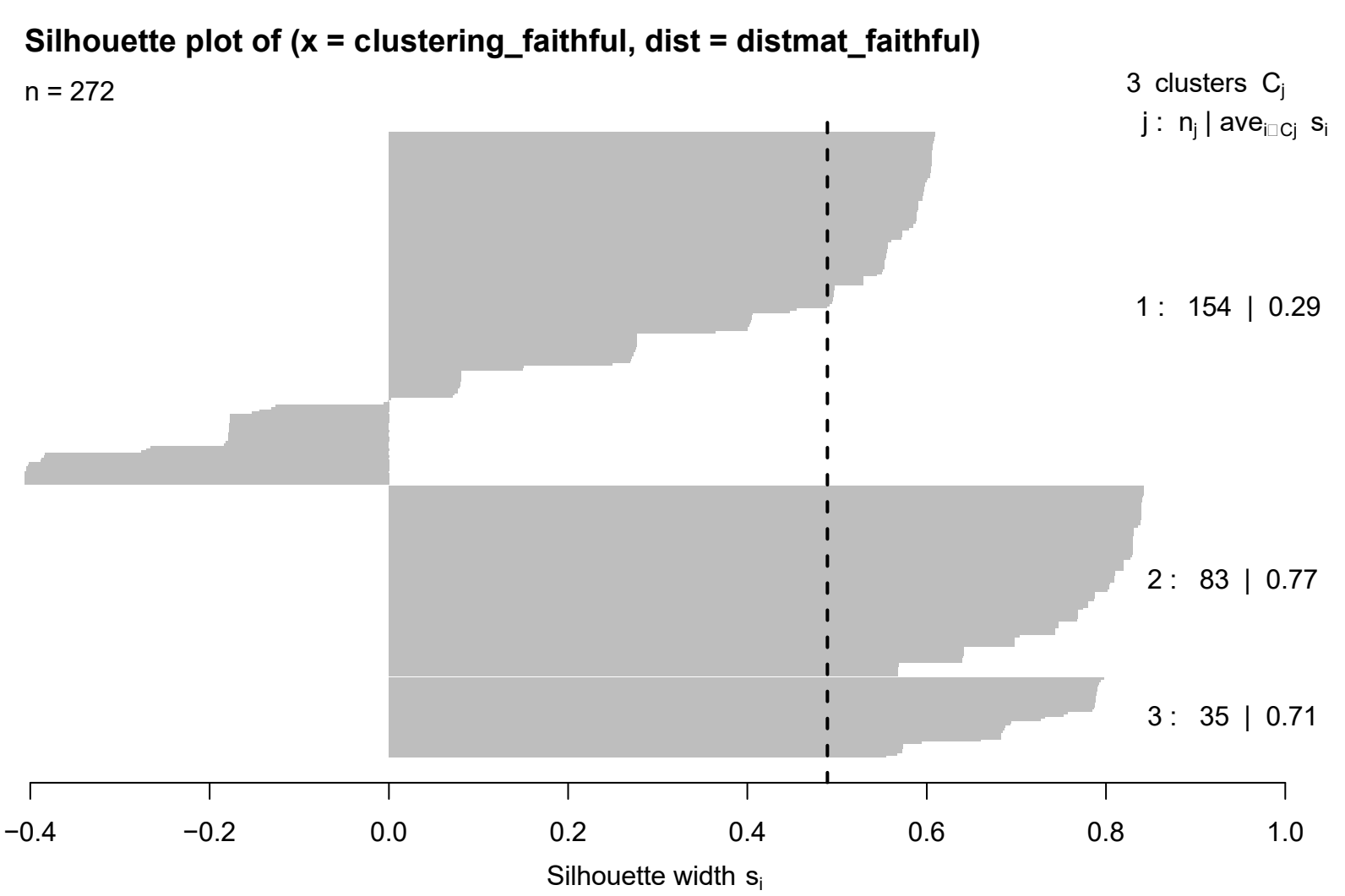
clusterCrit::intCriteria(as.matrix(x),
   cl, "Davies_Bouldin")
```

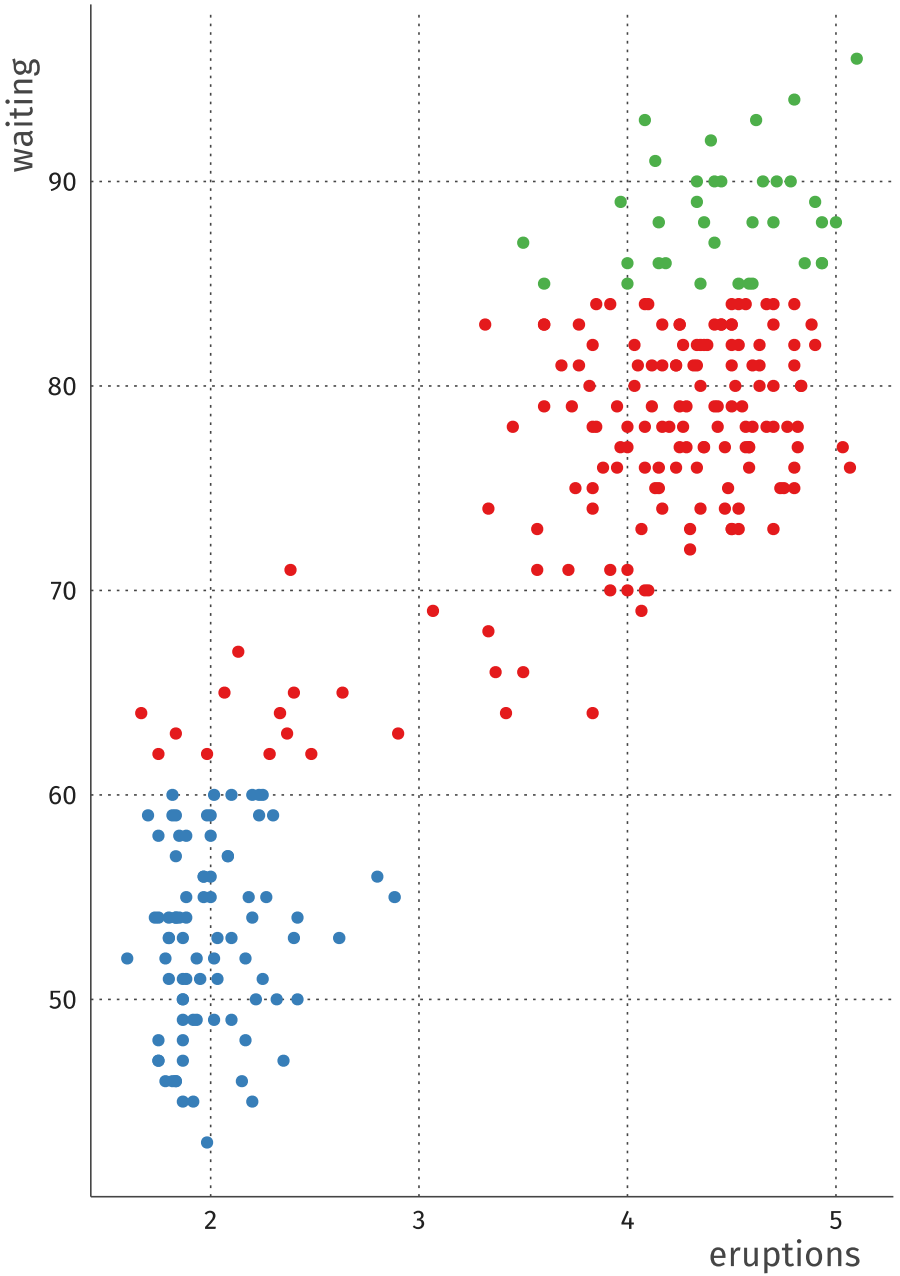
Python

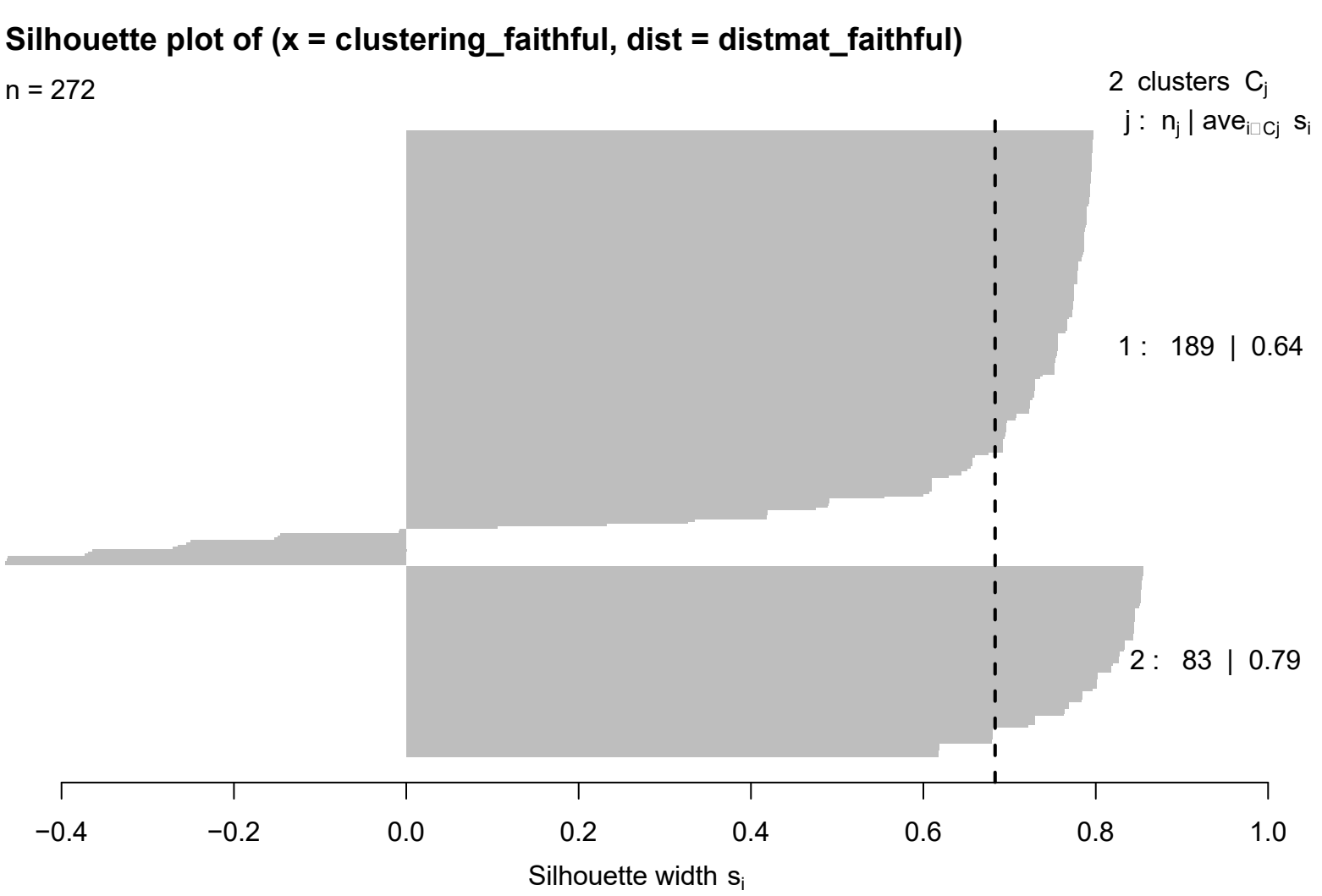
```python
from sklearn.metrics import (
    silhouette_score, davies_bouldin_score)
silhouette_score(X, labels)
davies_bouldin_score(X, labels)
```

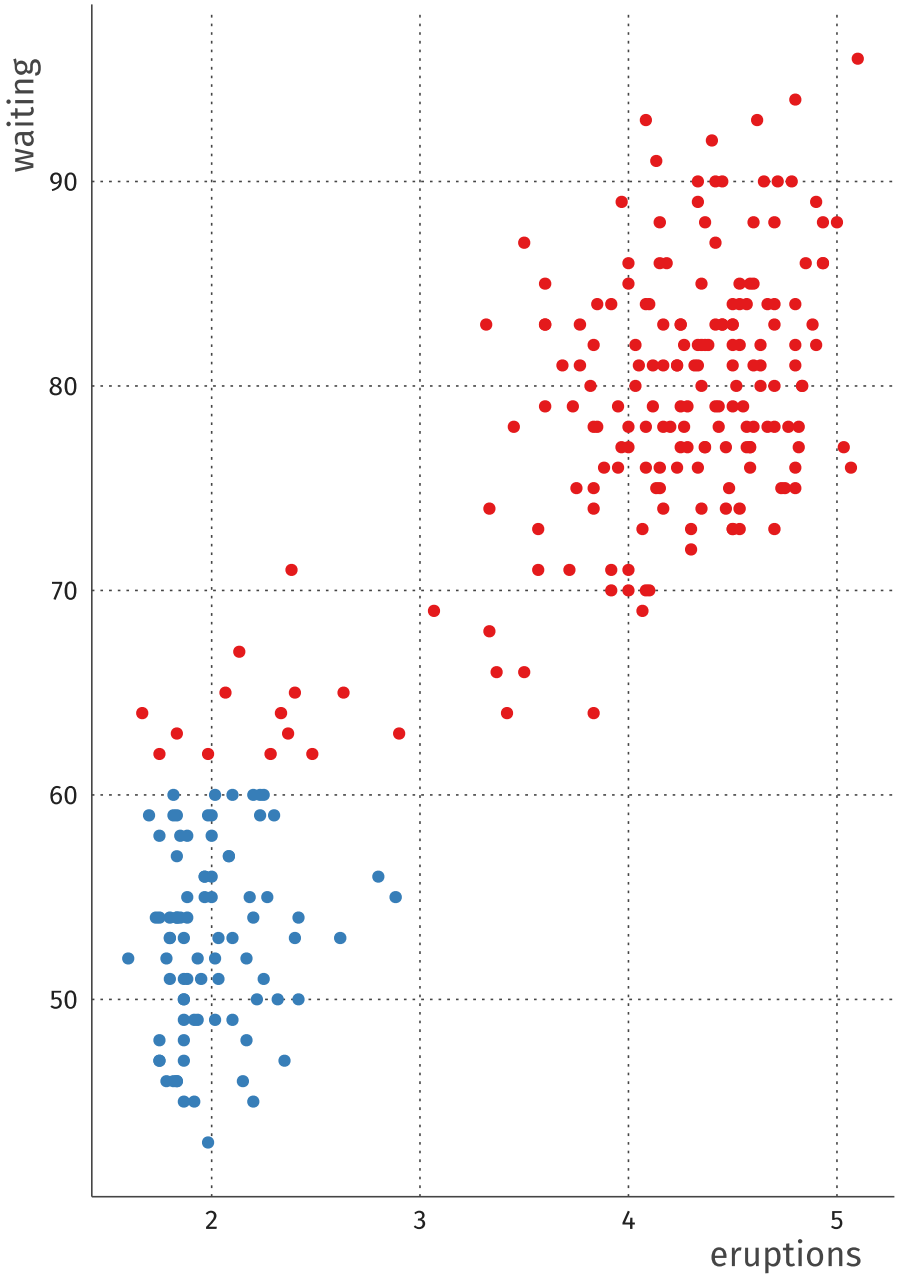

Silhouette plot of (x = clustering_faithful, dist = distmat_faithful)
n = 272
4 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 125 | 0.45
2 : 83 | 0.56
3 : 29 | 0.58
4 : 35 | 0.63
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.52


waiting
90
80
70
60
50
2
3
4
5
eruptions

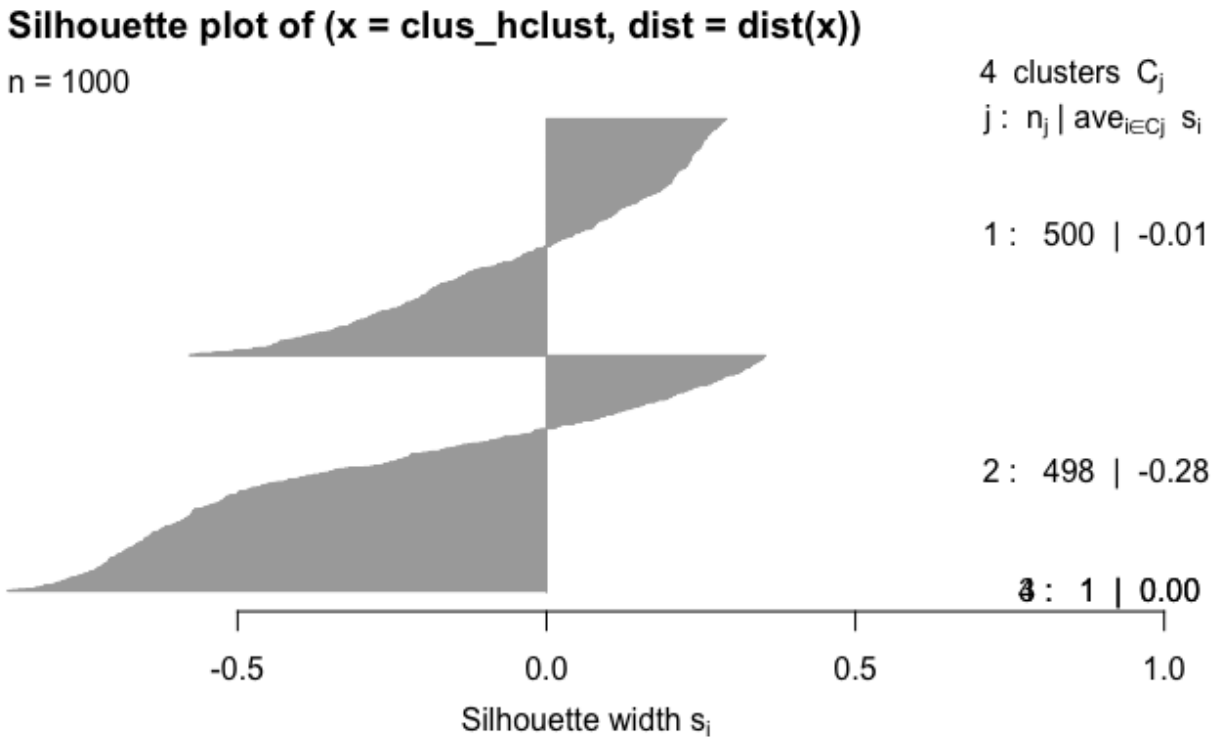

Silhouette plot of (x = clustering_faithful, dist = distmat_faithful)
n = 272
3 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 154 | 0.29
2 : 83 | 0.77
3 : 35 | 0.71
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.49

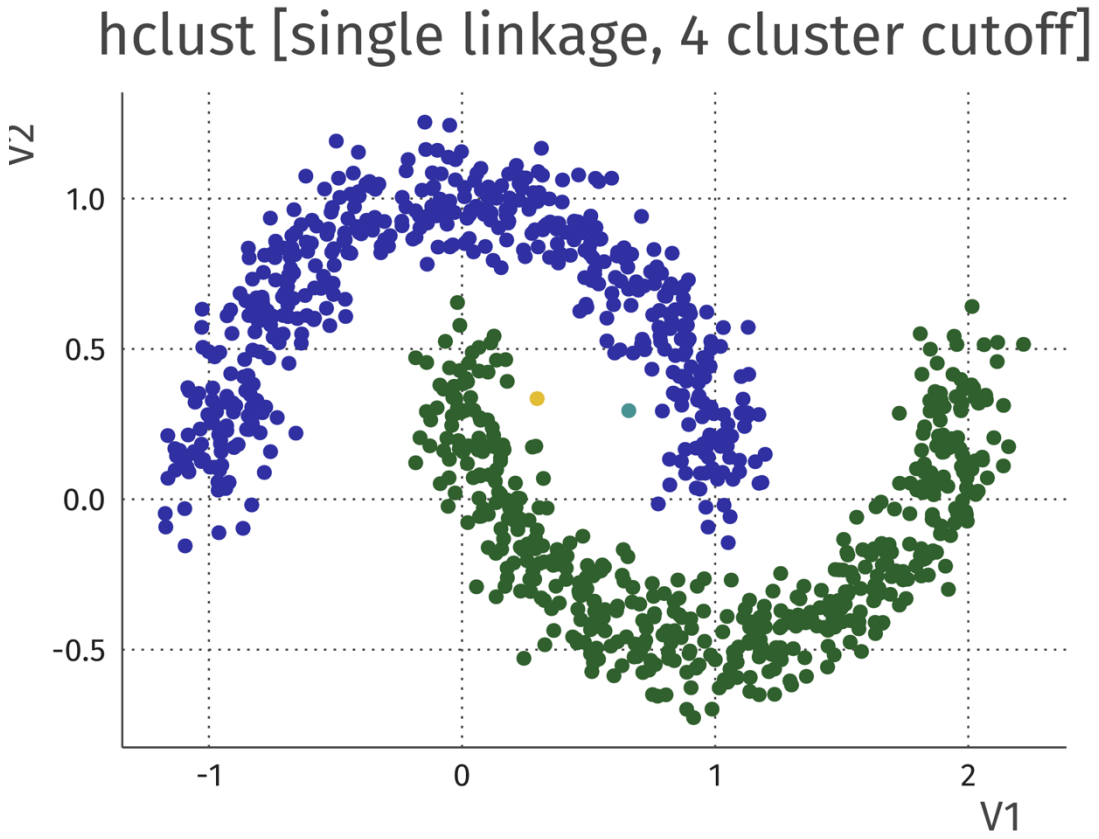

waiting
90
80
70
60
50
2
3
4
5
eruptions


Silhouette plot of (x = clustering_faithful, dist = distmat_faithful)
n = 272
2 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 189 | 0.64
2 : 83 | 0.79
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.68


waiting
90
80
70
60
50
2
3
4
5
eruptions

What do you think the average silhouette width (ASW) of this solution will approximately be?



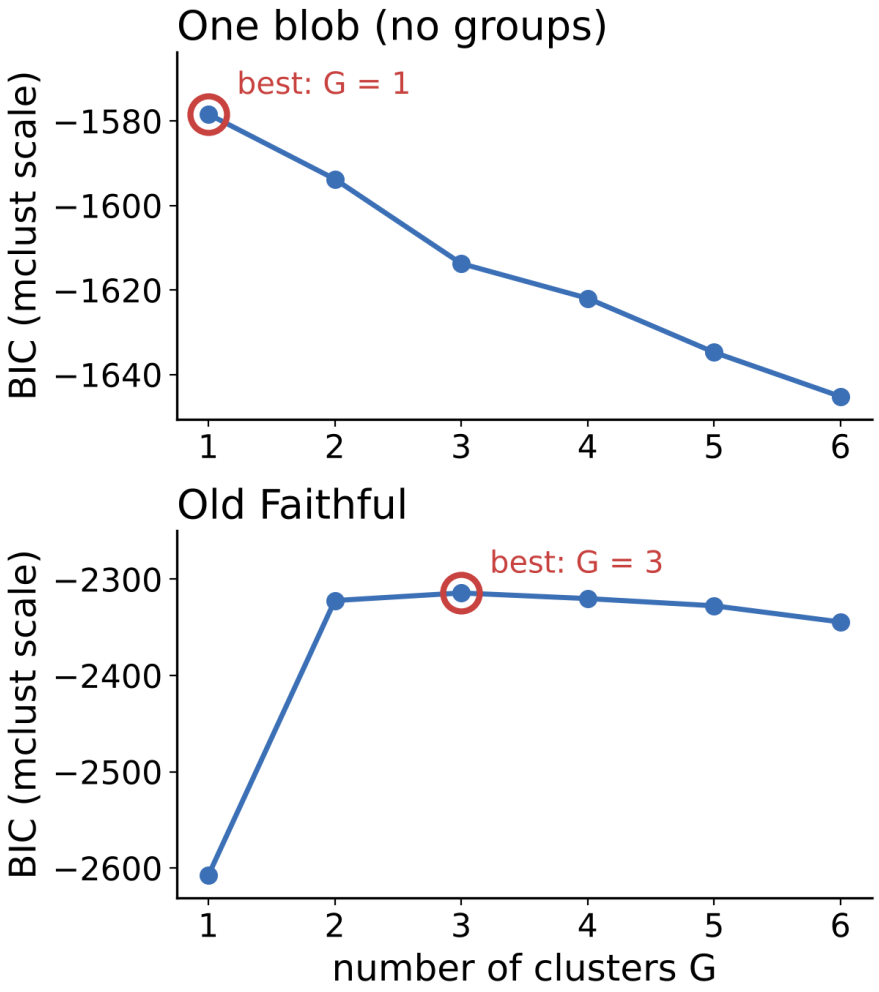


Question for you

5. Testing for clustering structure

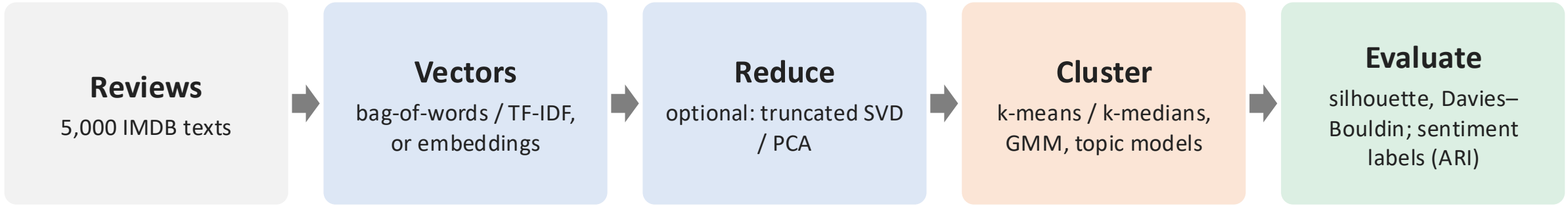
- Every algorithm returns clusters, even without any (Part A)
- **So first ask: is there cluster structure at all?**
- Model-based: include G = 1 and let BIC decide (Part C)
- Distance-based: the gap statistic compares within-cluster spread with data simulated without clusters

```R
Mclust(x, G = 1:9)
cluster::clusGap(x, kmeans, K.max = 8)
```



Gap statistic: Tibshirani, Walther & Hastie (2001)

Clustering text (Assignment 2)



Thousands of sparse dimensions

use cosine distance: the angle between documents, not their length

No full-covariance GMM

1,000 words → 500,500 covariance parameters per cluster, for 5,000 reviews: use spherical or diagonal covariances, or reduce first

Labels ≠ clusters

reviews may group by genre or topic rather than sentiment: a low ARI is not automatically bad

Due 26 Oct · the NLP lecture (27 Oct) covers text representations in depth

Checklist: clustering responsibly

Before

- choose the features deliberately — they decide which clusters you can find
- handle missing values (week 4)
- standardise, or choose a distance that fits the data
- pick a method whose idea of a cluster fits your aim

While clustering

- try at least two methods and several K
- many random starts; set a seed
- mixtures: compare shapes and K with BIC, G = 1 included

After

- validate with external information
- check stability
- look at internal indices and fit
- profile and name the clusters; report the uncertainty

Conclusion

- Unsupervised learning: no y (or everything is a y)
- Distance-based: hierarchical clustering (distance, linkage, dendrogram) and k-means (centroids, K, random starts)
- Model-based: a mixture of distributions → posterior probabilities, EM, BIC; k-means is a special case
- Evaluation: external information, visual exploration, stability, internal indices, tests for structure
- **No way to get rid of critical thought**

You can now …

explain what unsupervised learning is and why evaluating it is hard

run and interpret hierarchical clustering and k-means

explain model-based clustering (mixtures, posteriors, EM) and choose a model with BIC

evaluate a clustering in five ways

choose a sensible approach for a new clustering problem, including text

References

Aggarwal, C. C., & Reddy, C. K. (Eds.) (2014). Data Clustering: Algorithms and Applications. CRC Press.

Baudry, J.-P., Raftery, A. E., Celeux, G., Lo, K., & Gottardo, R. (2010). Combining mixture components for clustering. J. Comput. Graph. Stat., 19(2).

Biernacki, C., Celeux, G., & Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. IEEE TPAMI, 22(7).

Bouveyron, C., Celeux, G., Murphy, T. B., & Raftery, A. E. (2019). Model-Based Clustering and Classification for Data Science. Cambridge University Press.

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. IEEE TPAMI, 1(2).

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. JRSS B, 39(1).

Fraley, C., & Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. JASA, 97(458).

Hennig, C. (2007). Cluster-wise assessment of cluster stability. Comput. Stat. Data Anal., 52(1).

Hennig, C. (2016). Clustering strategy and method selection. In Hennig, Meila, Murtagh & Rocci (Eds.), Handbook of Cluster Analysis. CRC. arXiv:1503.02059.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An Introduction to Statistical Learning (2nd ed.), ch. 12. Springer.

Karrer, B., & Newman, M. E. J. (2011). Stochastic blockmodels and community structure in networks. Phys. Rev. E, 83, 016107.

Kaufman, L., & Rousseeuw, P. J. (1990). Finding Groups in Data. Wiley. • Rousseeuw, P. J. (1987). Silhouettes. J. Comput. Appl. Math., 20.

Oberski, D. L. (2016). Mixture models: latent profile and latent class analysis. In Robertson & Kaptein (Eds.), Modern Statistical Methods for HCI. Springer.

Tibshirani, R., Walther, G., & Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. JRSSB, 63(2).

Zachary, W. W. (1977). An information flow model for conflict and fission in small groups. J. Anthropol. Res., 33(4).