# Clustering Part 1

Daniel Lawson University of Bristol

Lecture 03.2.1 (v1.0.1)

# Signposting

- We have made latent structures using SVD and PCA.
- This **dimensionality reduction** is essential for many types of analysis including clustering.
- Clustering is one of the most fundamental data analysis tools and the ideas form the cornerstone of more complex approaches.
- In Part 1 we cover:
  - How Clustering methods are organised,
  - Hierarchical clustering
- In Part 2 we cover:
  - K-means
  - Gaussian Mixture Modelling
  - Density-based model-free clustering (dbscan)

# Intended Learning Outcomes

- ILO1 Be able to **access and process cyber security data** into a format suitable for mathematical reasoning
- ILO2 Be able to **use and apply basic machine learning** tools
- ILO3 Be able to make and report appropriate inferences from the results of applying basic tools to data

# Clustering

- Clustering contains enough complexity to cover several courses by itself.
- You are likely to use clustering in several projects, sometimes as the goal and sometimes as a data processing step.
- We will talk about **computational complexity**. This is covered in full detail later in the course. Today, $O(f(N))$ means that “the algorithm run-time increases as $f(N)$, ignoring complexity” (for the worst case data).

# Clustering paradigms

- Most clustering procedures fit one or more of these paradigms:
- **Algorithmic clustering**
  - An algorithm is run which outputs a clustering of the data
  - Usually fast
  - Usually data-type specific
  - Often hard to interpret
- **Distance-based clustering**
  - Distances between all items are considered and then clustered somehow
  - Widely applicable
  - Often can be linked to a model
- **Model-based clustering**
  - Explicit objective function used
  - Can be slower - unless a convenient model is chosen
  - Can be made to solve a specific task, handle uncertainty
  - Most appropriate when you want the clusters to “mean something”

# Most important clustering methods

- **Algorithmic**:
  - graph-cutting methods, e.g. modularity
  - space partitioning, e.g. KD-trees, etc
- **Hierarchical**, distance-based:
  - single linkage
  - complete linkage
  - average linkage
- **Model-based**:
  - k-means (though was introduced as an algorithm)
  - Gaussian mixture modelling (GMM)
  - Bayesian clustering

# Algorithmic clustering

Algorithmic approaches are best when used with a goal that exploits the structure provided. We'll visit them as needed. For example:

- There are really fast **graph clustering algorithms**. The clusters are not always "best" but they are useful.
    - See for example modularity maximisation, min-cut
    - General problem: community detection
- Some really useful **data structures** in computer science resemble clustering.
    - KD-trees are a binary splitting method for $\mathbb{R}^d$
    - They partition the space using the specified points
    - See also Quadtree, R-tree, etc.
    - They solve lookup problems; for example, fast recall of approximate nearest-neighbours.

# Hierarchical clustering

This comes in two flavours:

- **Divisive clustering**: start with all objects in a single cluster and split them;
- **Agglomerative clustering**: start with all objects in a different cluster and merge them.
- In general divisive clustering is harder to "get right" so we focus on agglomerative methods. Broadly, these:
    1. start with $N$ clusters $c_i$; defined by the original points
    2. choose the closest two clusters $a$ and $b$ to **merge** based on a distance measure $d_{ab}$
    3. **update** the locations and hence the **distances** of the clusters according to some rule.

# Distances

- The choice of distance is very important for clustering. Here are some common ones:

| Model | Norm | Equation |
|---|---|---:|
| Euclidean | $\|x-y\|_2$ | $\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$ |
| Squared Euclidean | $\|x-y\|_2^2$ | $\sum_{i=1}^{n}(x_i-y_i)^2$ |
| Manhattan | $\|x-y\|_1$ | $\sum_{i=1}^{n}\lvert x_i-y_i\rvert$ |
| Maximum | $\|x-y\|_\infty$ | $\max_i\lvert x_i-y_i\rvert$ |
| Mahalanobis | $\|x-y\|_M$ | $[(\vec{x}-\vec{y})C^{-1}(\vec{x}-\vec{y})^T)]^{1/2}$ |

- Note the connection of the Mahalanobis norm to PCA[1]!
- See also: Hamming Distance (for binary variables), edit distance, etc.

[1] "The squared Mahalanobis distance is equal to the sum of squares of the scores of all non-zero standardised principal components."

# Metrics and related objects

- Distances $d : X \times X \to [0, \infty)$ are a **Metric** and satisfy:
  - $d(x, y) = d(y, x)$: symmetry
  - $d(x, y) \geq 0$: non-negativity
  - $d(x, y) = 0 \Leftrightarrow x = y$: (the distance is only zero if the elements are the same)
  - $d(x, z) \leq d(x, y) + d(y, z)$: Triangle inequality
- Some methods can work with **divergences**, which need not satisfy symmetry or the Triangle inequality.
- If instead $d(x, z) \leq \max(d(x, y), d(y, z))$ the $d$ is called **ultrametric**. This is important for certain types of tree.

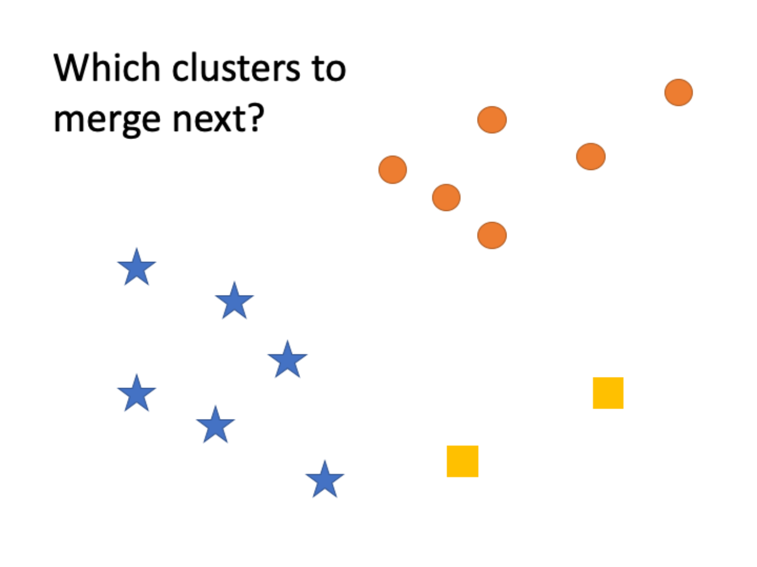
# Hierarchical clustering

- Hierarchical clustering methods report **trees** as their output.
- We select the threshold $k$ (a “tree cut”) to select the number of clusters
- Many criteria exist to do this selection in an automated way:
  - Within-vs Between cluster variation[2]
  - Gap statistic[3]
  - etc ...
  - Why not use **Cross validation**!

[2] Calinski and Harabasz (1974), “A dendrite method for cluster analysis”

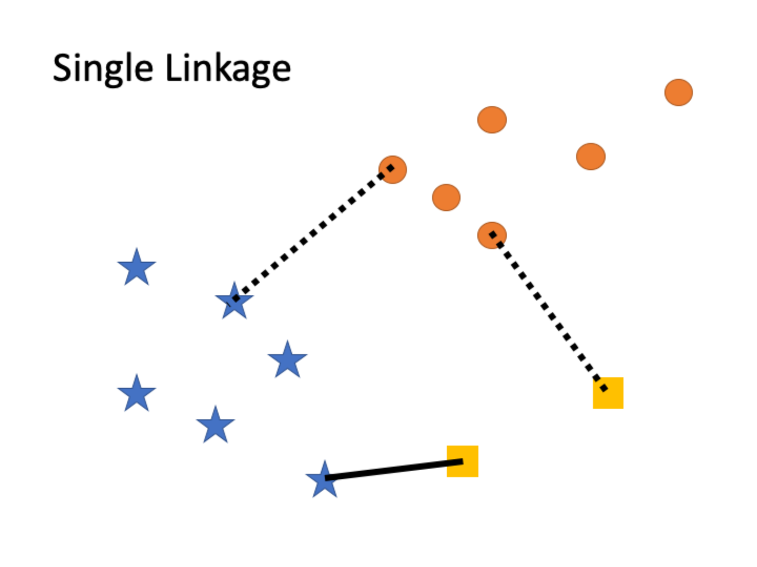
[3] Tibshirani et al. (2001), “Estimating the number of clusters in a data set via the gap statistic”

# Linkage clustering



Which clusters to merge next?

# Single linkage clustering



# Single linkage clustering

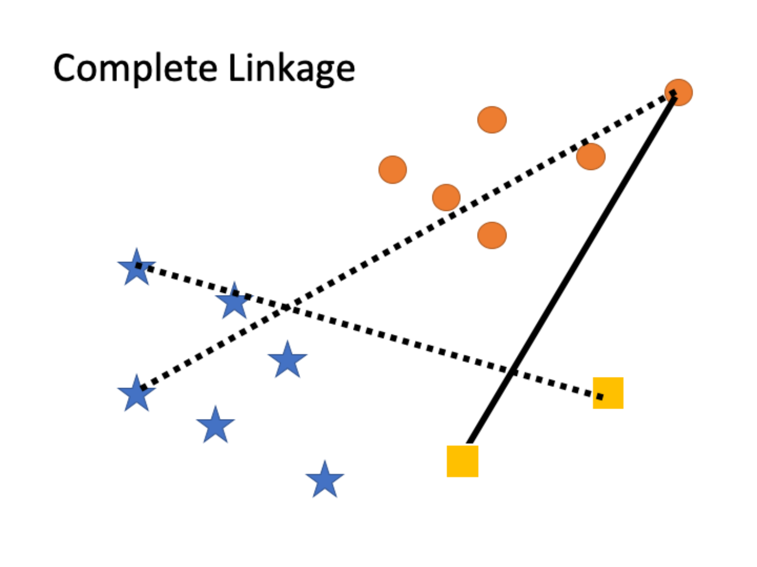
- Hierarchical clustering where we set

$$d_{a,b} = \min_{i \in a, j \in b} d_{i,j}$$

- i.e. the distance is the **closest point** in each cluster.
- The naive implementation would take $O(N^3)$.
- Good implementations are $O(N^2)$ (e.g. SLINK, 1973)[4], Kruskal's algorithm for minimum spanning trees.

[4] Sibson 1973, "SLINK: An optimally efficient algorithm for the single-link cluster method".

# Complete linkage clustering



# Complete linkage clustering

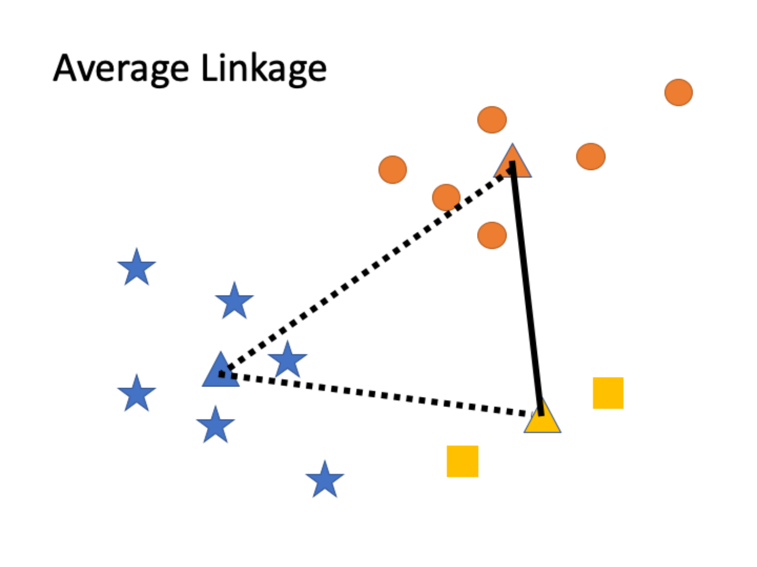
- Hierarchical clustering where we set

$$d_{a,b} = \max_{i \in a, j \in b} d_{i,j}$$

- i.e. the distance is the **furthest point** in each cluster.
- The naive implementation would take $O(N^3)$.
- Good implementations are $O(N^2)$ (CLINK, 1977)[5].

[5] Defays 1977, "An efficient algorithm for a complete link method".

# Average linkage clustering



# Average linkage clustering

- Also known as “Unweighted Pair Group Method with Arithmetic mean” (UPGMA).
- Hierarchical clustering where we set

$$d_{a,b} = \mathbb{E}_{i \in a, j \in b}(d_{i,j})$$

- i.e. the distance is the **average distance** between each cluster.
- The naive implementation would take $O(N^3)$.
- Good implementations are $O(N^2 \log(N))$.
- It can be “meaningful”:
  - the recovered tree is the “true tree” if the clusters diverged at constant rate.
  - This is plausible in evolution, for example.

# Hierarchical Clustering: See also

- **Centroid** Linkage: Define centres of each cluster, compute distance to cluster centres
- **Minimax** Clustering[6] : Minimise the maximum radius to the centre of each group
- NB: Minimax is an important concept in Machine Learning!

[6] Bien et al. (2011), "Hierarchical Clustering with Prototypes via Minimax Linkage"

# Implementations in R

```r
library("hclust") # default hierarchical clustering
library("fastcluster") # faster implementations
```

- Implementations are important for computational complexity and speed[7]

[7] http://danifold.net/fastcluster.html?section=1

# Signposting

- We'll go straight to **03.2.2 Clustering Part 2** for:
  - K-means
  - Gaussian Mixture Modelling
  - Density-based model-free clustering (dbscan)