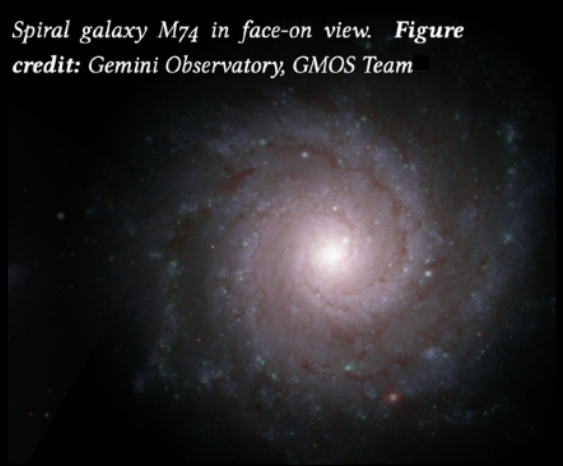
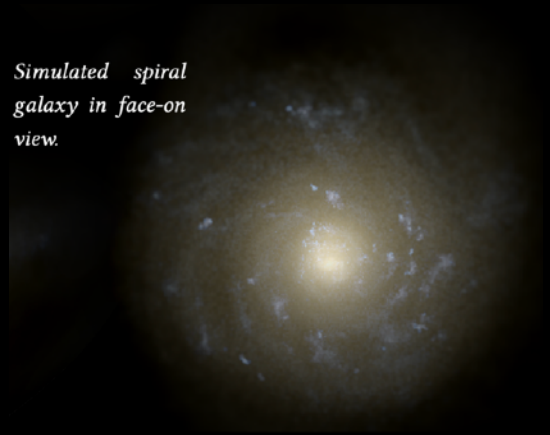
ASTR4004

COMPUTATIONAL ASTRONOMY

Week 8 https://github.com/svenbuder/astr4004_2024_week9



Spiral galaxy M74 in face-on view. ***Figure credit:*** *Gemini Observatory, GMOS Team*



Simulated spiral galaxy in face-on view.

Figure credit: *Tobias Buck*

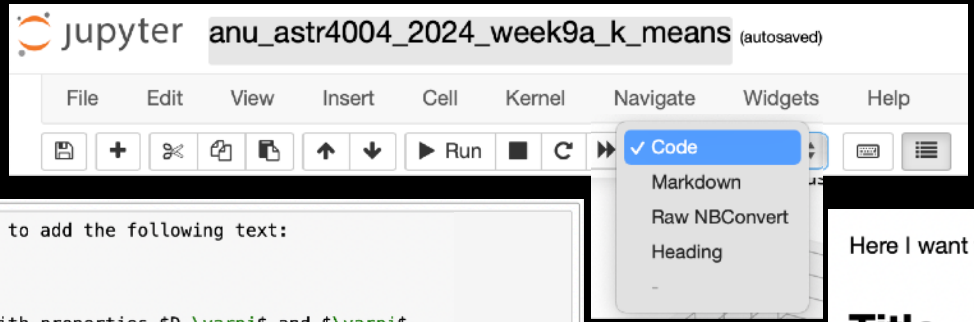
DR SVEN BUDER | RESEARCH SCHOOL OF ASTRONOMY AND ASTROPHYSICS

Australian National University

Adding description in Jupyter notebooks (.ipynb)

Markdown (.md) is an often used hybrid format between text, latex, and html.

It comes in very handy for adding text between code, e.g. for "discussion" tasks of an assignment.



Here I want to add the following text:

Title

Some text with properties D_ϖ and ϖ. I also want to show how to convert between them via:

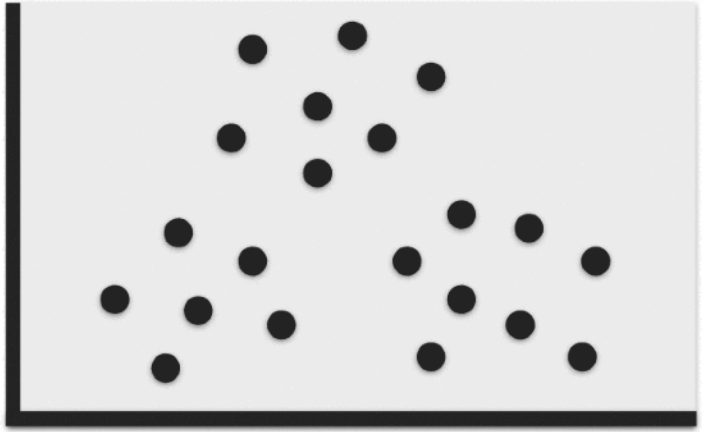
$$D_\varpi = \frac{1}{\varpi}$$

and then continue.

Most GitHub repository and code documentation (e.g. README.md) is written in markdown.

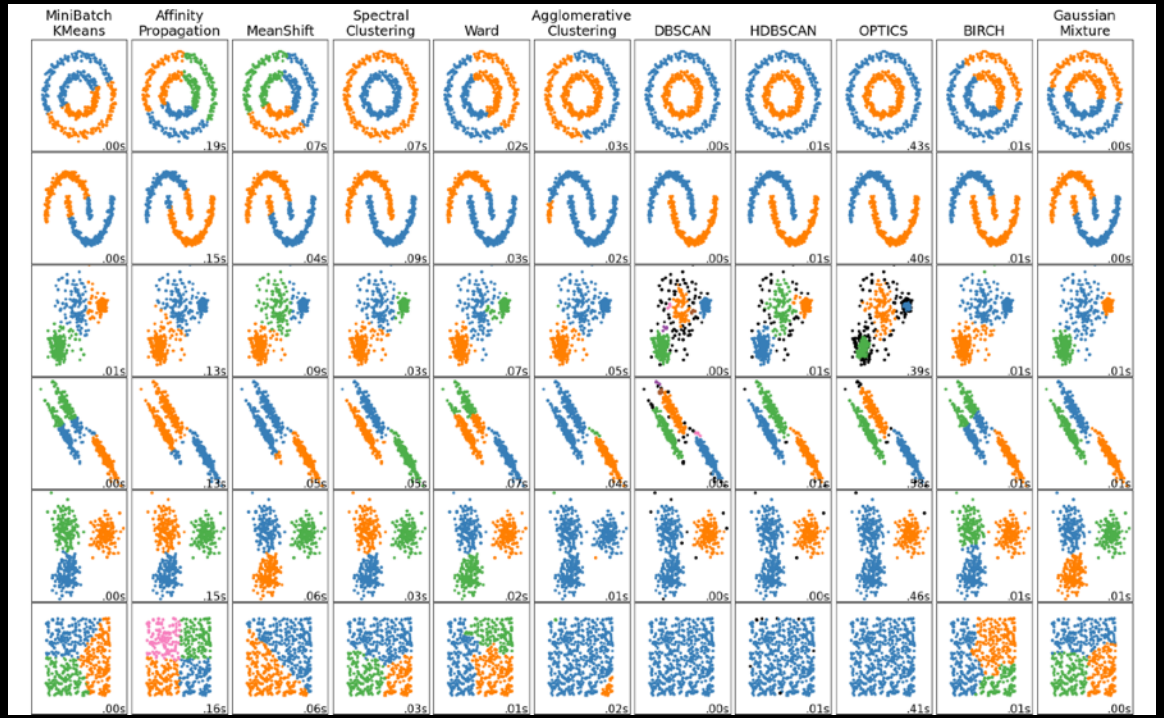
CLUSTERING:

1. Initialise random centroids
2. Until convergence:
   - Assign step
   - Update step
3. End



Clustering

You can imagine a lot of examples of easy or difficult clustering problems



Overview from https://scikit-learn.org/stable/modules/clustering.html

Clustering Checklist

- Why am I clustering these data? What do I want to learn or infer from the data?
- Since the clustering algorithm I choose is almost guaranteed not to be representative of the generative model that produced the data, what artefacts in the clustering outputs do I need to be worried about?
- How much can I believe the clustering results? What things can I cross-check to make sure they look sensible?
- Even though you may disbelieve the clustering results, are they **sufficient for my purpose**? That is to say: even if the model is wrong, does it still have utility?

K-means

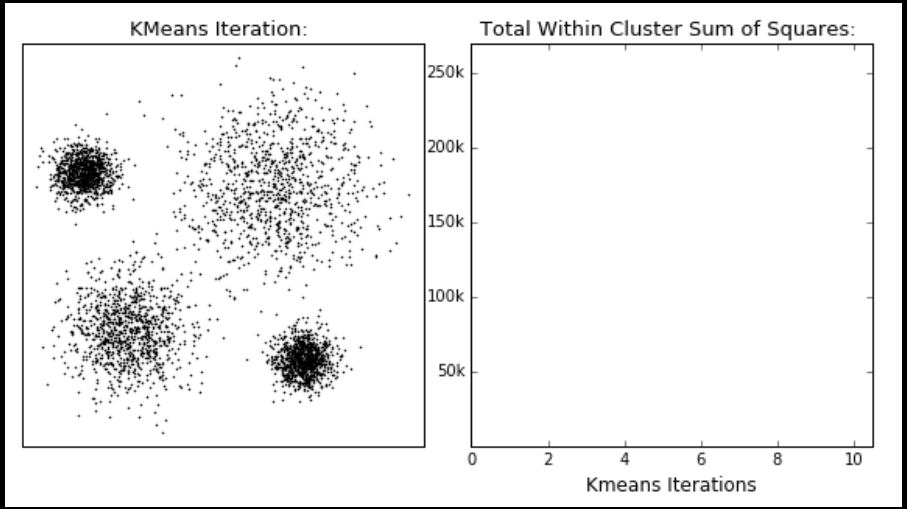
clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the *inertia (within-cluster sum-of-squares):*

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$

Initialise cluster centres μ_j

Iteratively reassign data points x_i to the closest centroid

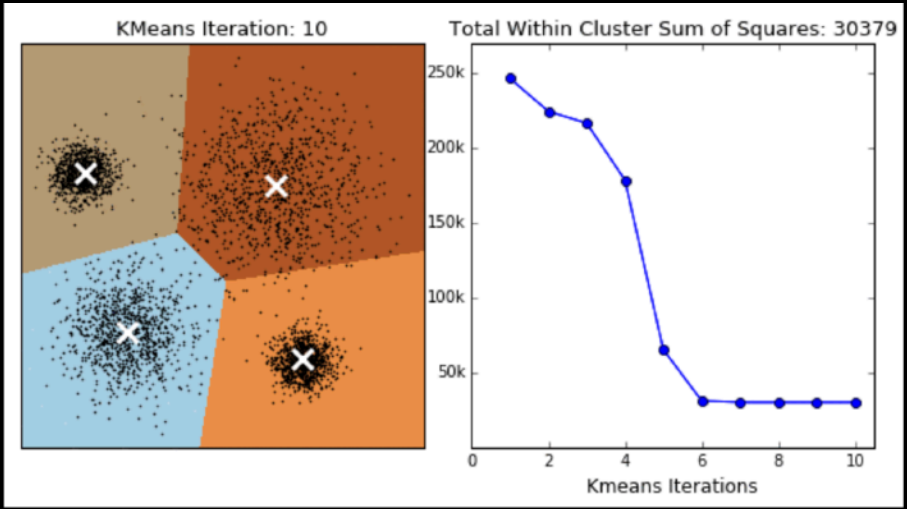
Move the centroids to the centre of their assigned points

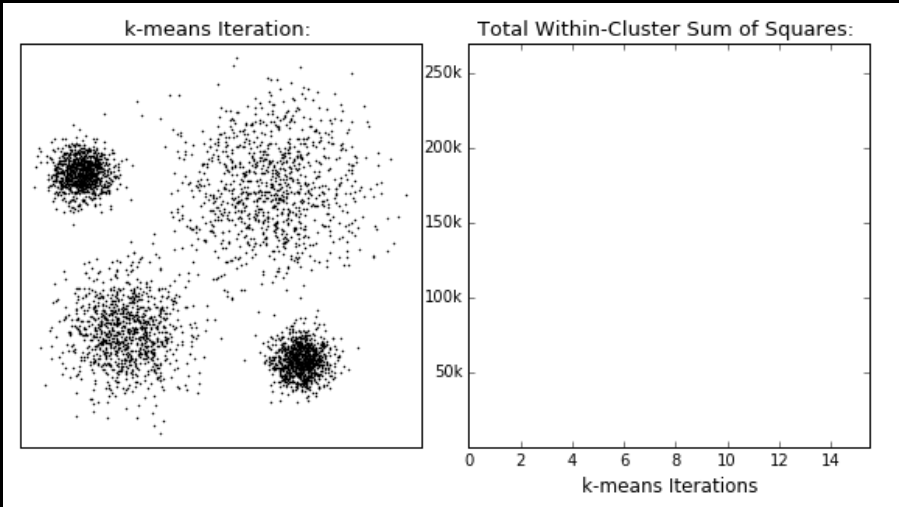


Iterate towards more compact & separated clusters until little to no improvement

K-means

k-means algorithms are sensitive to the starting position of the cluster centres, as each method converges to local optima



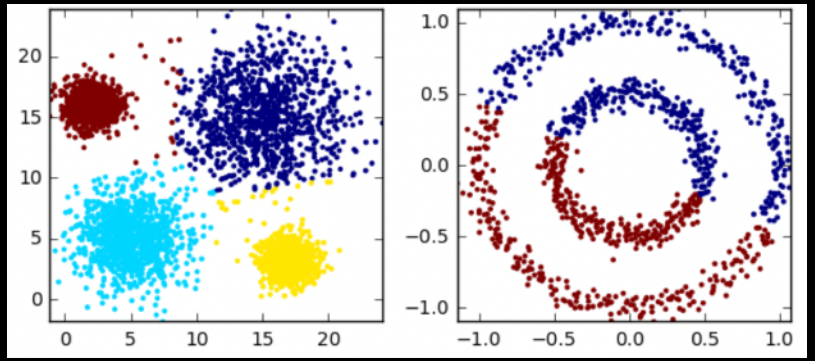


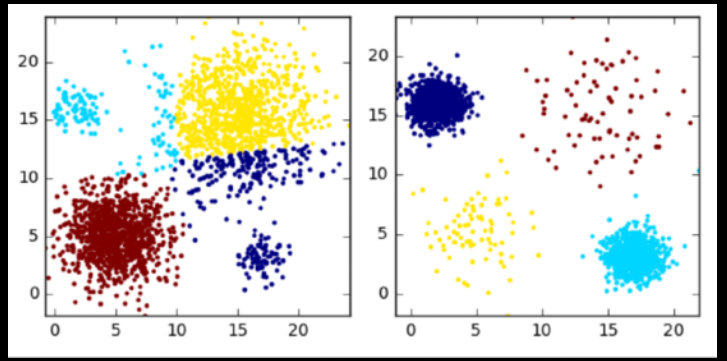
sklearn allows you to initialise "smarter" than random (init='k-means++') and multiple times (**n_init)

K-means

The algorithm seeks and identifies globular (essentially spherical) clusters.

If this assumption doesn't hold,
the model output may be inadequate (or just really bad).





k-means can also underperform with clusters of different size and density.

But: k-means is one of the least complex and thus fastest algorithms!

GAUSSIAN MIXTURE MODELS

Australian National University

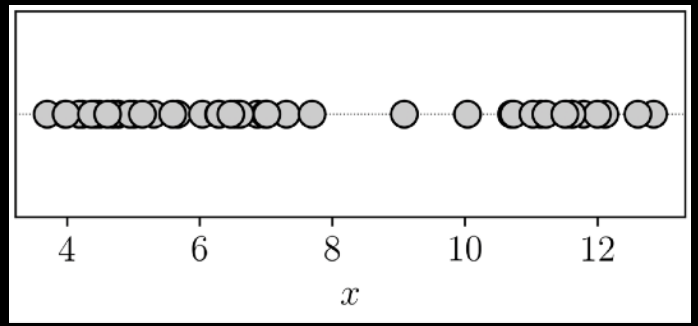
Gaussian Mixture Models

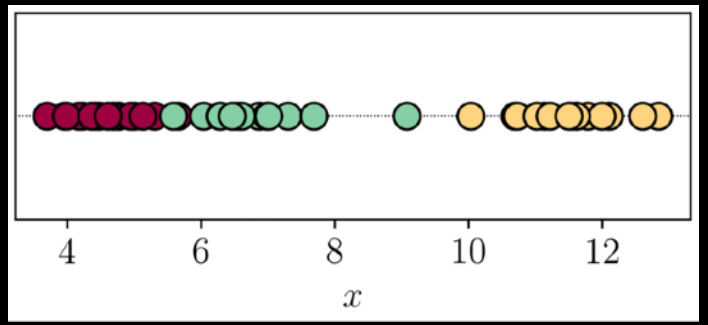
Let's consider the possibility that some data x is described by K components with model parameters Z_k:

$$p(x) = \sum_{k=1}^{K} \pi_k p(x|Z_k)$$

with weight values π_k

$$\sum_{k=1}^{K} \pi_k = 1$$





Simple case:

$Z_k = (\mu_k, \sigma_k)$

$$\mu_1 = \frac{1}{N_{\text{red}}} \sum_{i=1}^{N_{\text{red}}} x_i$$

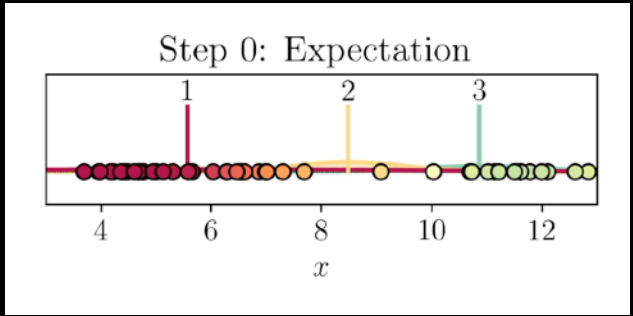
$$\sigma_1^2 = \frac{1}{N_{\text{red}}} \sum_{i=1}^{N_{\text{red}}} (x_i - \mu_1)^2$$

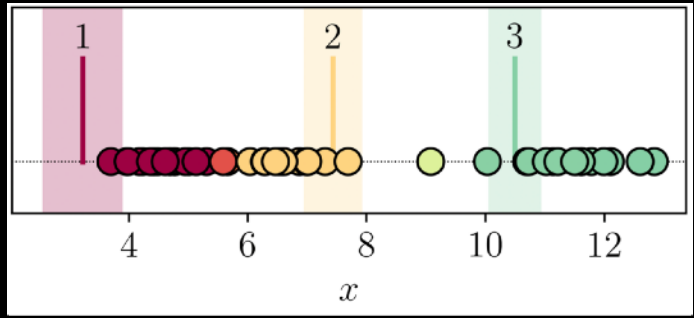
$$\pi_1 = \frac{N_{\text{red}}}{N}$$

Gaussian Mixture Models

$$p(\mathcal{M}_k|x_i) = \frac{p(x_i|\mathcal{M}_k)p(\mathcal{M}_k)}{\sum_{j=1}^{K} p(x_i|\mathcal{M}_j)p(\mathcal{M}_j)}$$

posterior probability of membership for belonging to one component, compared to all components





data points in between two components have a membership proportion that is split between the two components

Expectation Maximization (EM):
Alternate steps between expectation which component data point comes from and update of model parameter estimate

Gaussian Mixture Models (GMM)

GMM for N data points each of D dimensions, drawn from K Gaussian components

$$p(y|\theta) = \sum_{k=1}^{K} w_k p_k(y|z_k, \theta_k)$$

where $\sum^K w_k = 1$ and $\theta_k = \{\mu_k, \Sigma_k\}$

Iterative Expectation-Maximization (EM)

E-step (Expectation): Estimate which Gaussian each data point most likely belongs to.

$$w_{nk} = p(z_{nk} = 1|y_n, \theta) = \frac{w_k p_k(y_n|z_k, \theta_k)}{\sum_{m=1}^{K} w_m p_m(y_n|z_m, \theta_m)}$$

$$p_k(\mathbf{y}|\theta_k) = \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{y} - \mu_k)^\top \boldsymbol{\Sigma}_k^{-1}(\mathbf{y} - \mu_k)\right]$$

M-step (Maximization): Update the parameters (means, variances, and mixing coefficients) based on the estimated membership from the E-step.

Effective number of data points described by the k-th mixture: $N_k = \sum^N w_{nk}$ and $\sum^K w_{nk} = 1$ for every data point

New weights

New means

New covs.

$$w_k^{(\text{new})} = \frac{N_k}{N}$$

$$\mu_k^{(\text{new})} = \frac{1}{N_k} \sum_{n=1}^{N} w_{nk} \mathbf{y}_n$$

$$\boldsymbol{\Sigma}_k^{(\text{new})} = \frac{1}{N_k} \sum_{n=1}^{N} w_{nk} \left(\mathbf{y}_n - \mu_k^{(\text{new})}\right)\left(\mathbf{y}_n - \mu_k^{(\text{new})}\right)^\top$$

K-MEANS AND GAUSSIAN MIXTURE MODELS

IN PRACTICE

Australian National University

DENSITY-BASED SPATIAL CLUSTERING OF APPLICATIONS WITH NOISE (DBSCAN)

Australian National University

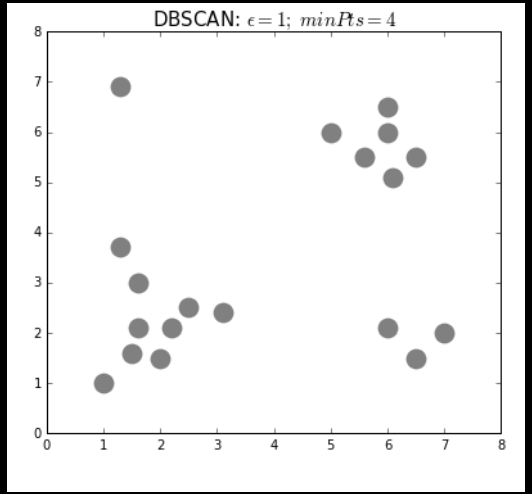
DENSITY-BASED SPATIAL CLUSTERING OF APPLICATIONS WITH NOISE (DBSCAN)

Clusters are considered zones that are sufficiently dense

Points that lack neighbours do not belong to any cluster and are thus classified as noise

Doesn't require the user to specify the number of clusters; it works that out for you

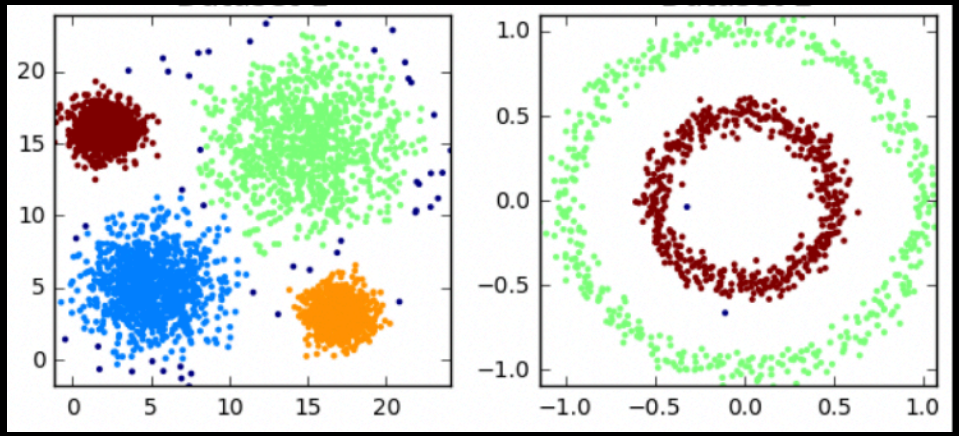
Needs minimum of points that constitutes a cluster (minPts) and the size of the neighbourhoods (epsilon)

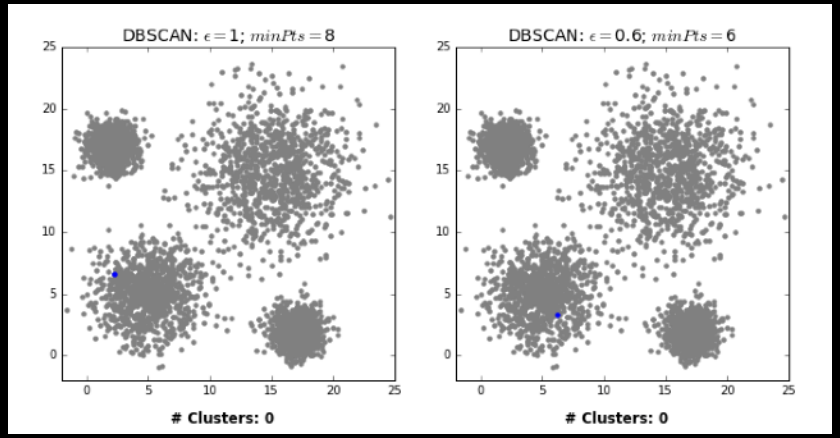


Identifies clusters and then expands clusters by scanning the neighbourhoods

Once all neighbourhoods have been exhausted, the process repeats with a new cluster, until all observations belong to a segment or have been classified as noise

DENSITY-BASED SPATIAL CLUSTERING OF APPLICATIONS WITH NOISE (DBSCAN)





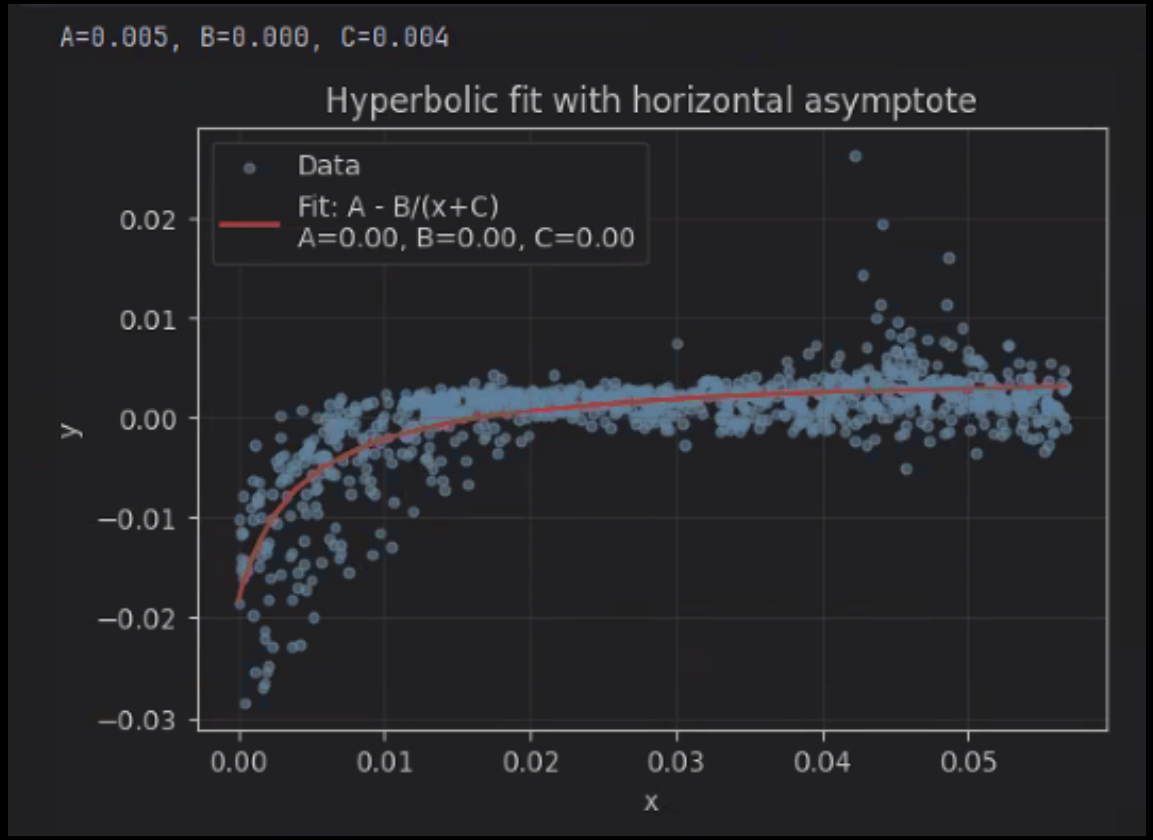
Hierarchical DBSCAN (HDBSCAN), for example, is building a hierarchy of clusters and then condensing it into the most stable ones.

It is continuously varying the density parameter epsilon to detect clusters at multiple density levels (dense points of small clusters -> less dense and larger clusters)

There are millions of other clustering algorithms!

PLOT CLINIC

Australian
National
University



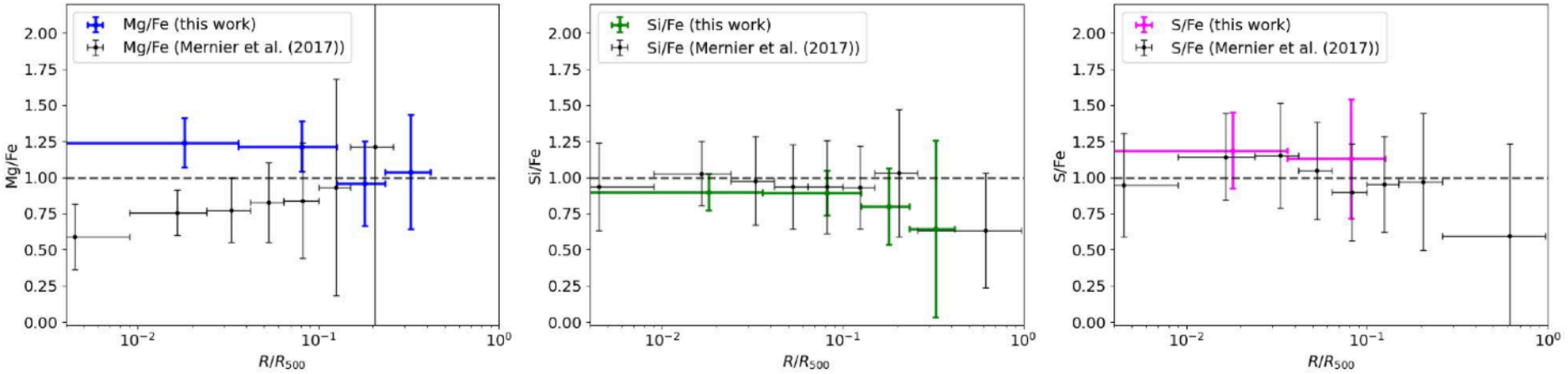


Figure 6. Radial abundance ratio profiles ([Mg/Fe], [Si/Fe] and [S/Fe]) of Mrk1216 and groups average from Mernier et al. (2017).