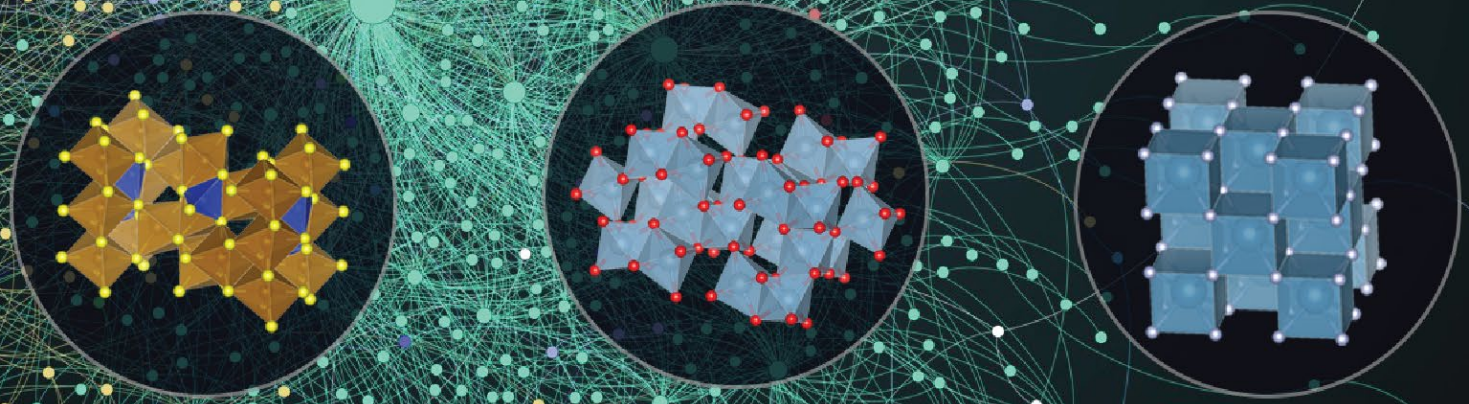
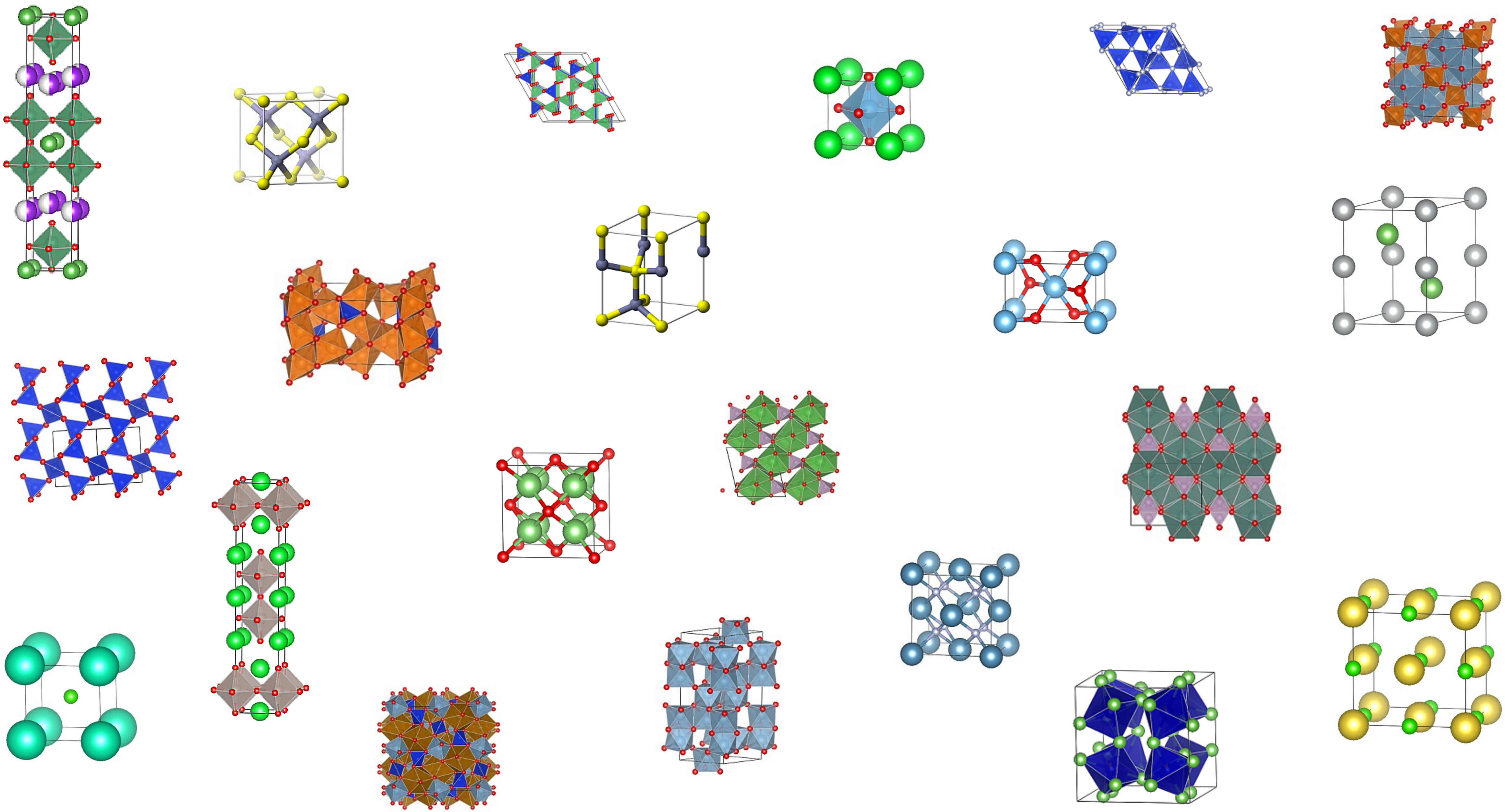
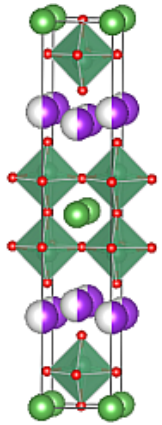
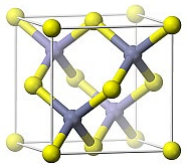
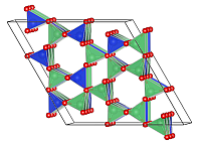
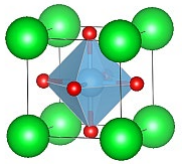
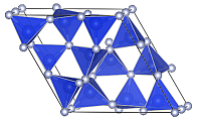
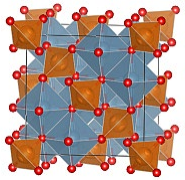
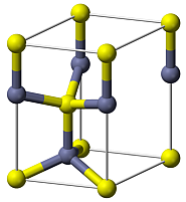
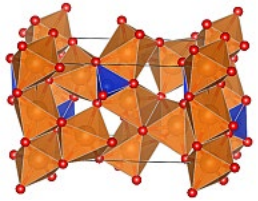
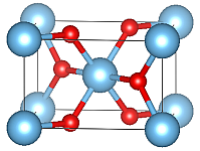
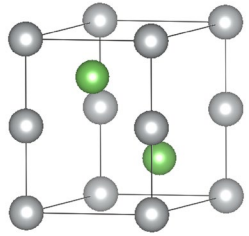
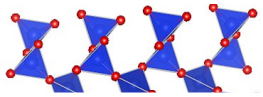
Clustering

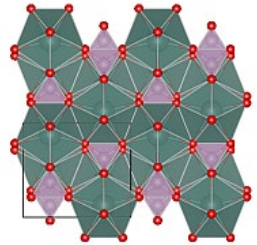
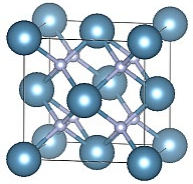
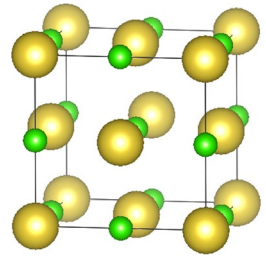
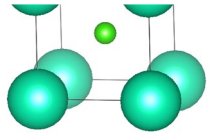
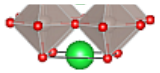
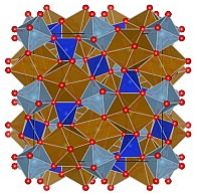
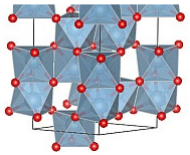
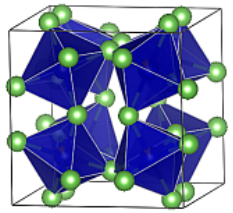


Cluster analysis is about finding (and visualizing) things that are similar



Cluster analysis is about finding (and visualizing) things that are similar

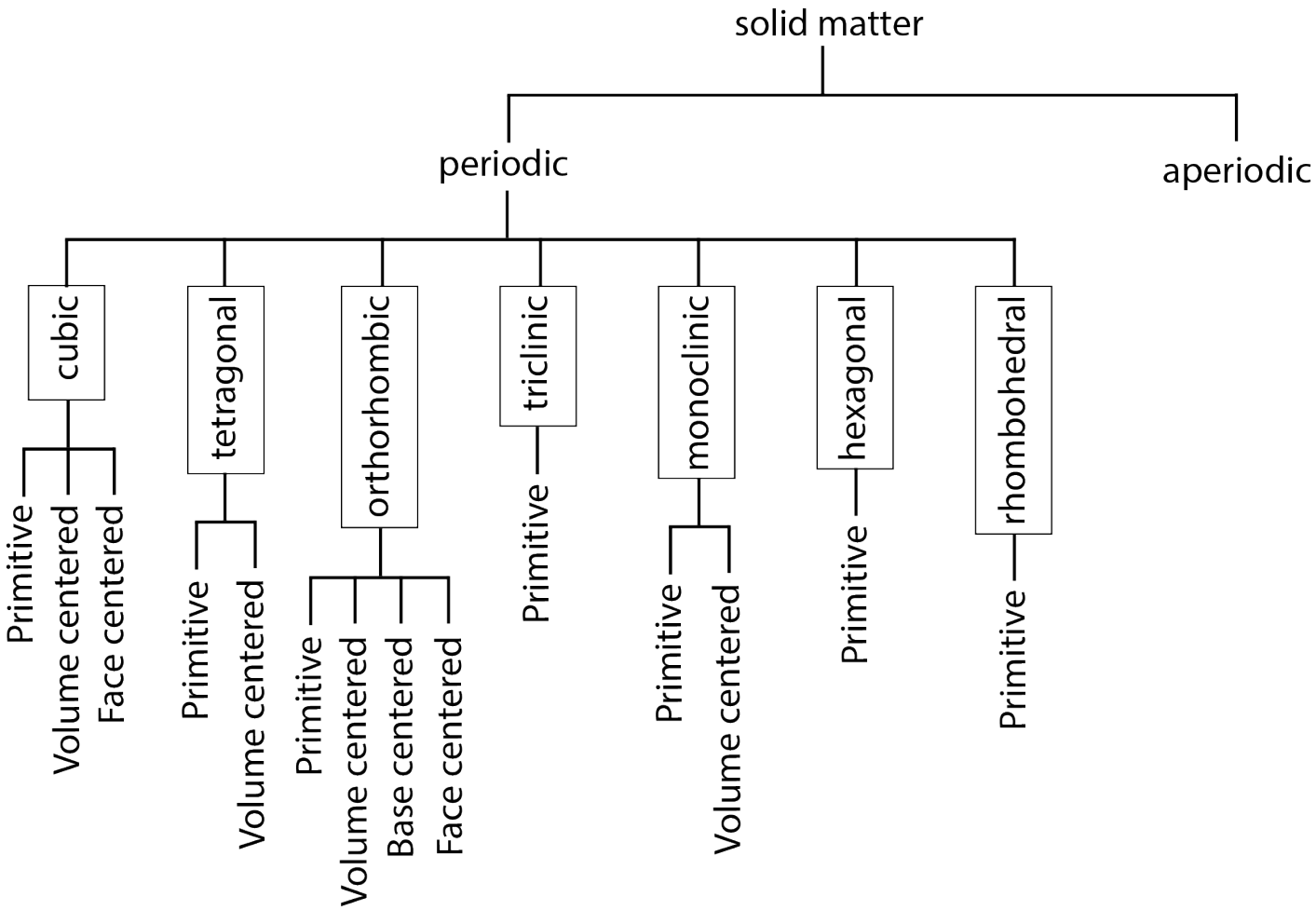









- Clustering is subjective in that it depends on your interests and/or perspective
- You use features when you cluster!
- We need a measurement of similarity

Distance or similarity metrics should follow a few basic rules

$D(A,B) = D(B,A)$	Symmetry
$D(A,A) = 0$	Constancy of Self-Similarity
$D(A,B) = 0\ iif\ A = B$	Positivity (Separation)
$D(A,B) \leq D(A,C) + D(B,C)$	Triangular Inequality

Clustering can be hierarchical or partitional



Spinels

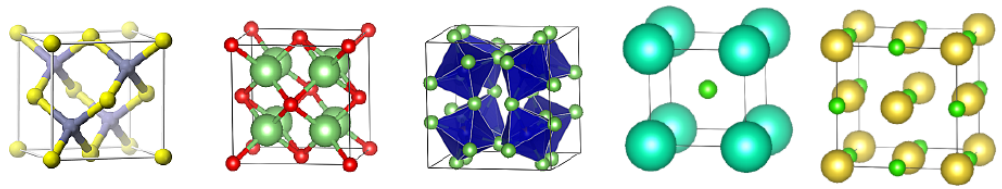
Perovskites

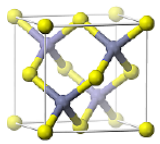
Ruddlesden-Popper

Zintl phases

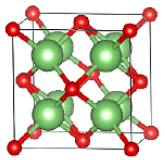
Clathrates

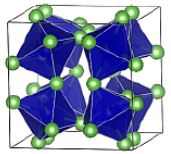
Clustering begins with distance matrix

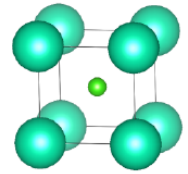


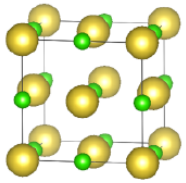


0	4	3	5	5
	0	5	3	3
		0	4	3
			0	2
				0









Structure features

Different cation coordination = 1

Different polyhedra connectivity = 1

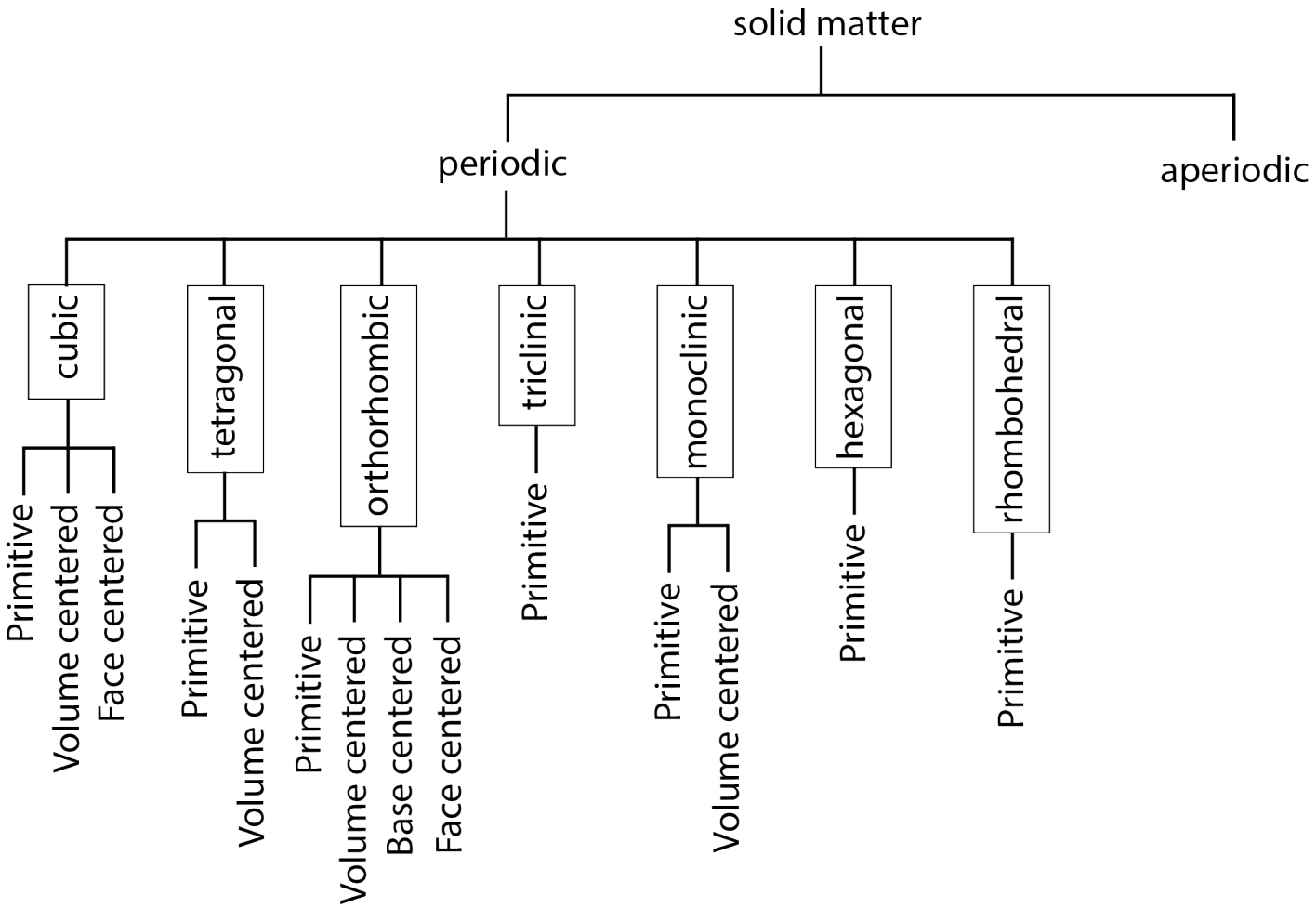
Different close/open packing = 1

Composition features

Different cation PT group= 1

Different anion PT group = 1

Distances between clusters is more challenging

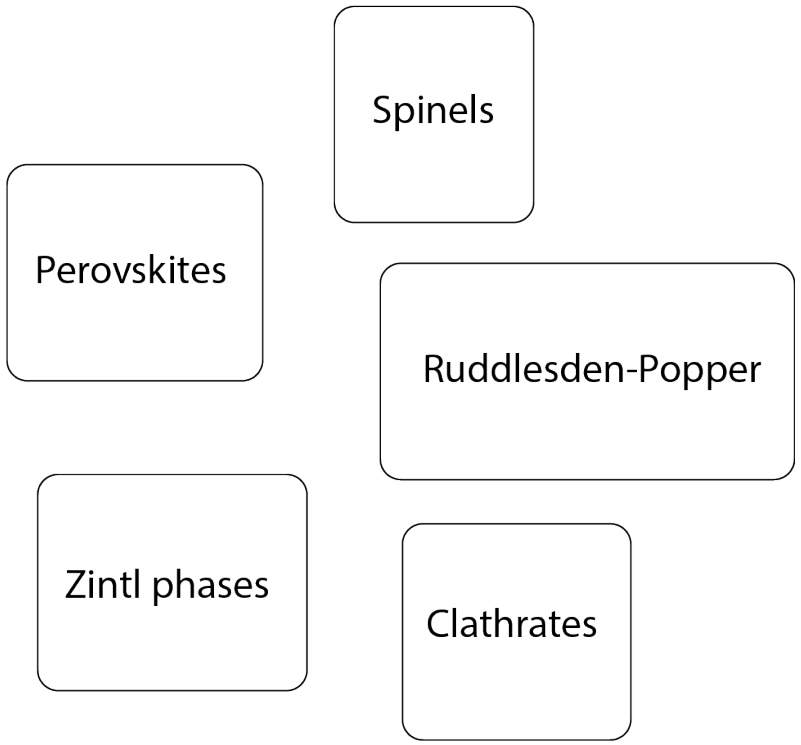


Nearest neighbor (single linkage)

Furthest neighbors (complete linkage)

Center of mass (group average)

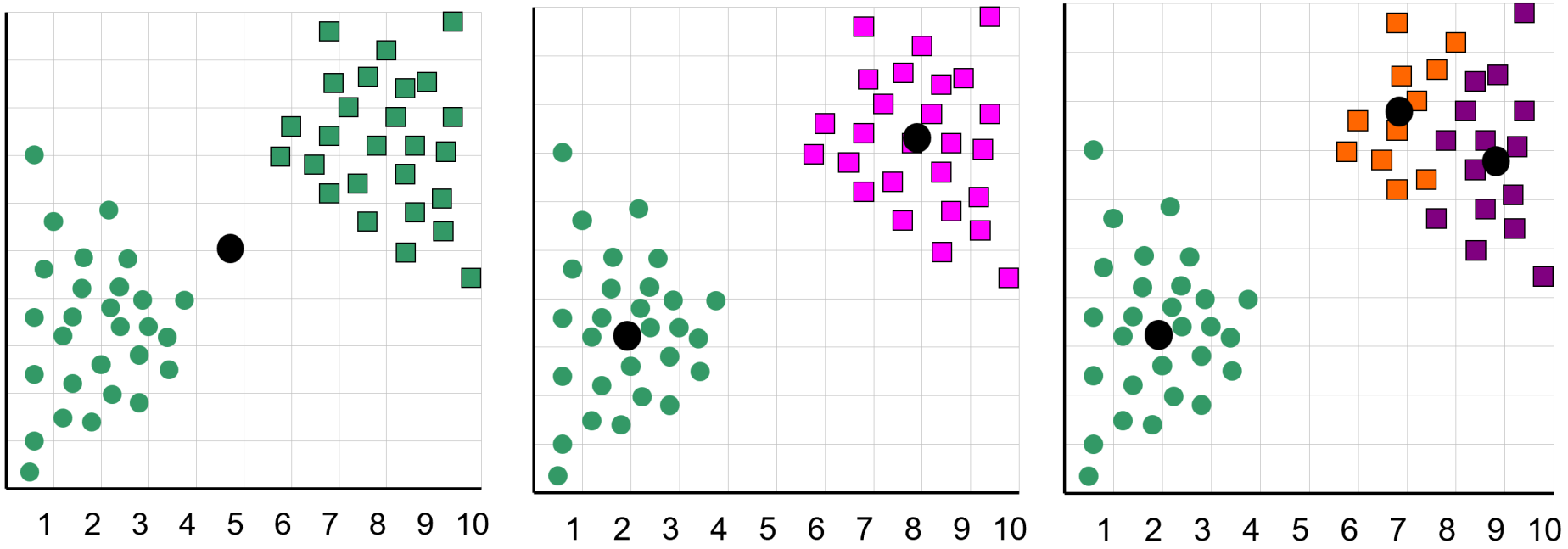
Partition clustering typically has user-selected number of clusters

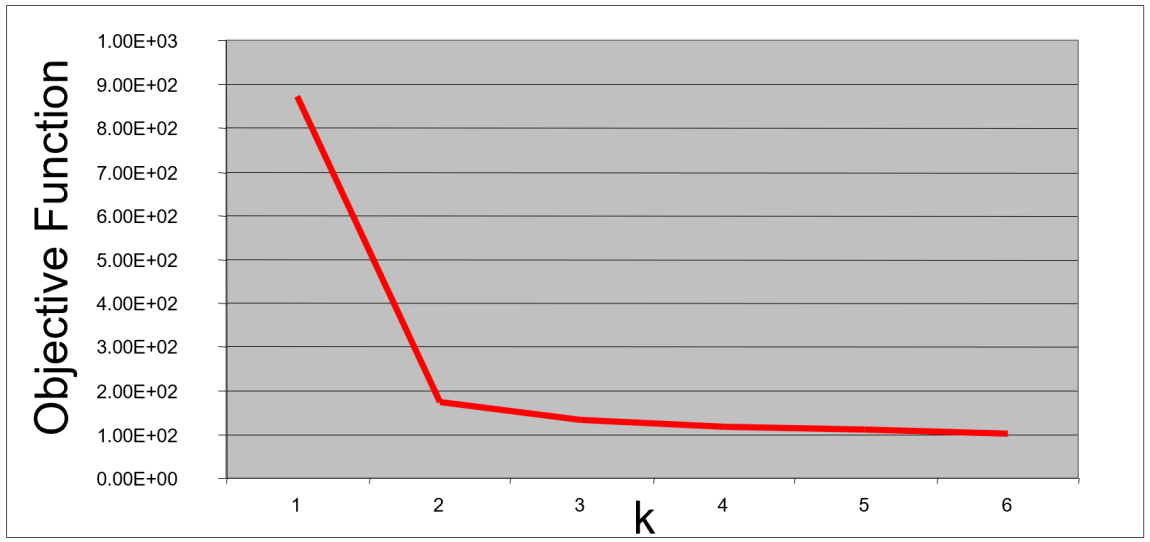


Steps:

1. Decide how many clusters, k
2. Initialize clusters centers (random if necessary)
3. Determine cluster membership by assigning to nearest cluster center
4. Re-estimate cluster centers assuming correct membership
5. If no objects change membership exit, otherwise re-assign based on new cluster centers

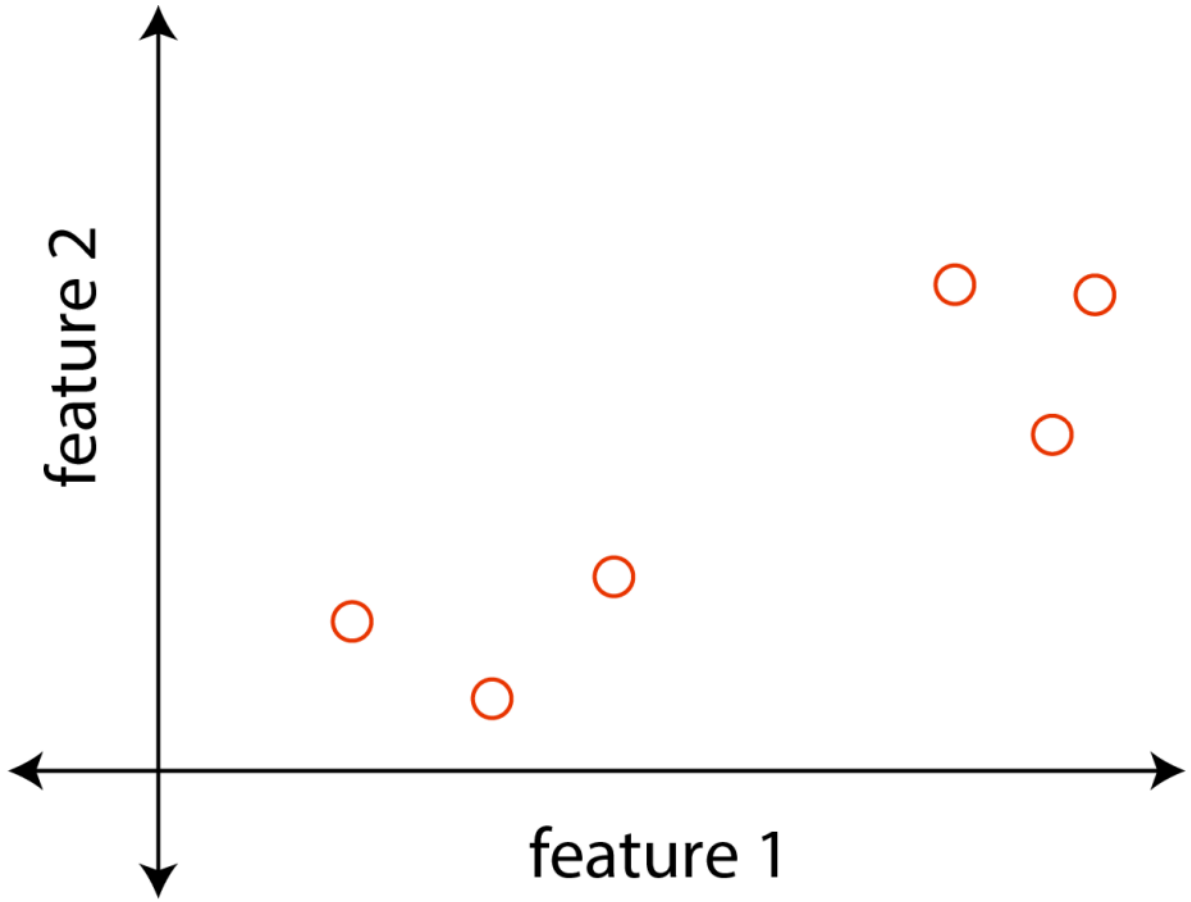
How many clusters should we pick is uncertain and a bit subjective





https://www.cs.ucr.edu/~eamonn/time_series_data_2018/

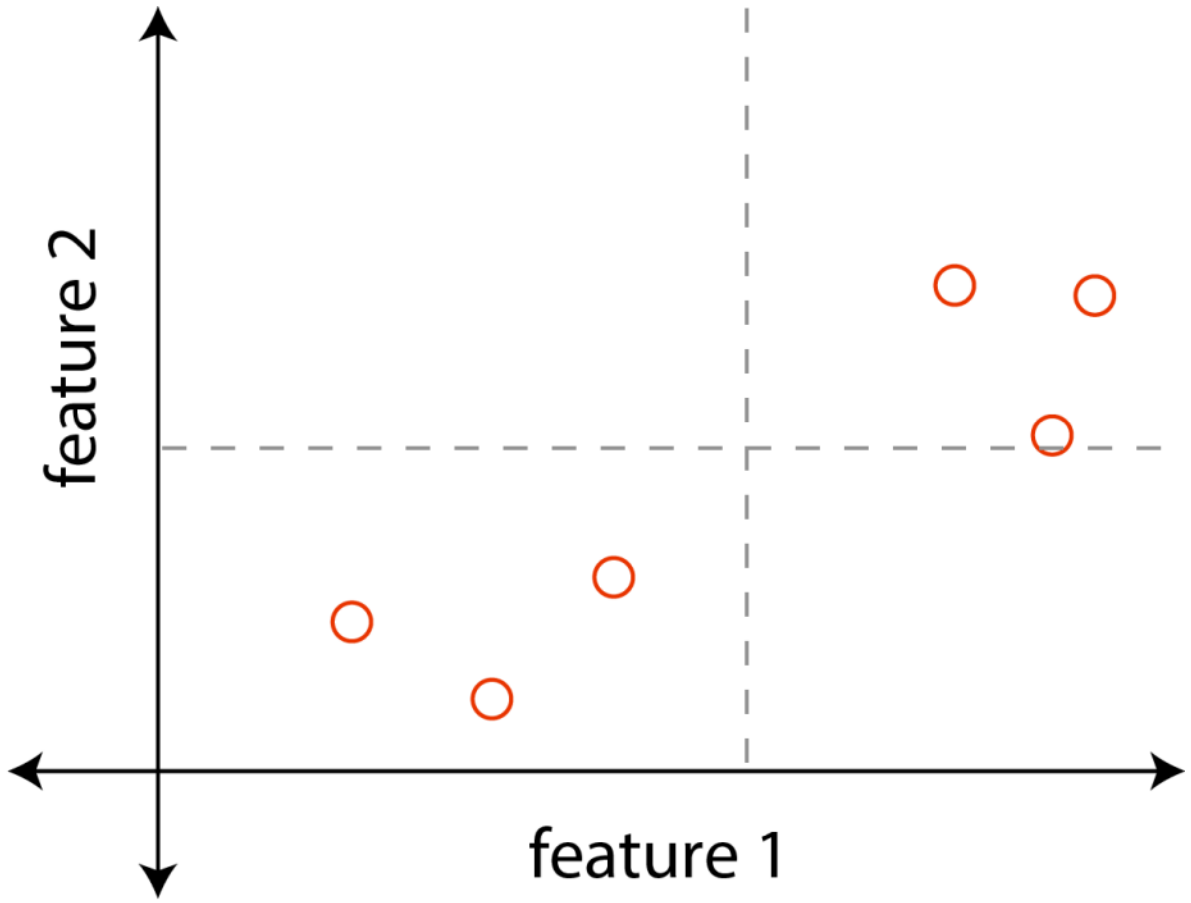
Principal component analysis is often lumped together with clustering



Steps:

1. Plot data for each variable
2. Calculate average value for each variable
3. Plot center of data
4. Move center to (0,0) origin
5. Fit line through origin and then rotate to get best fit

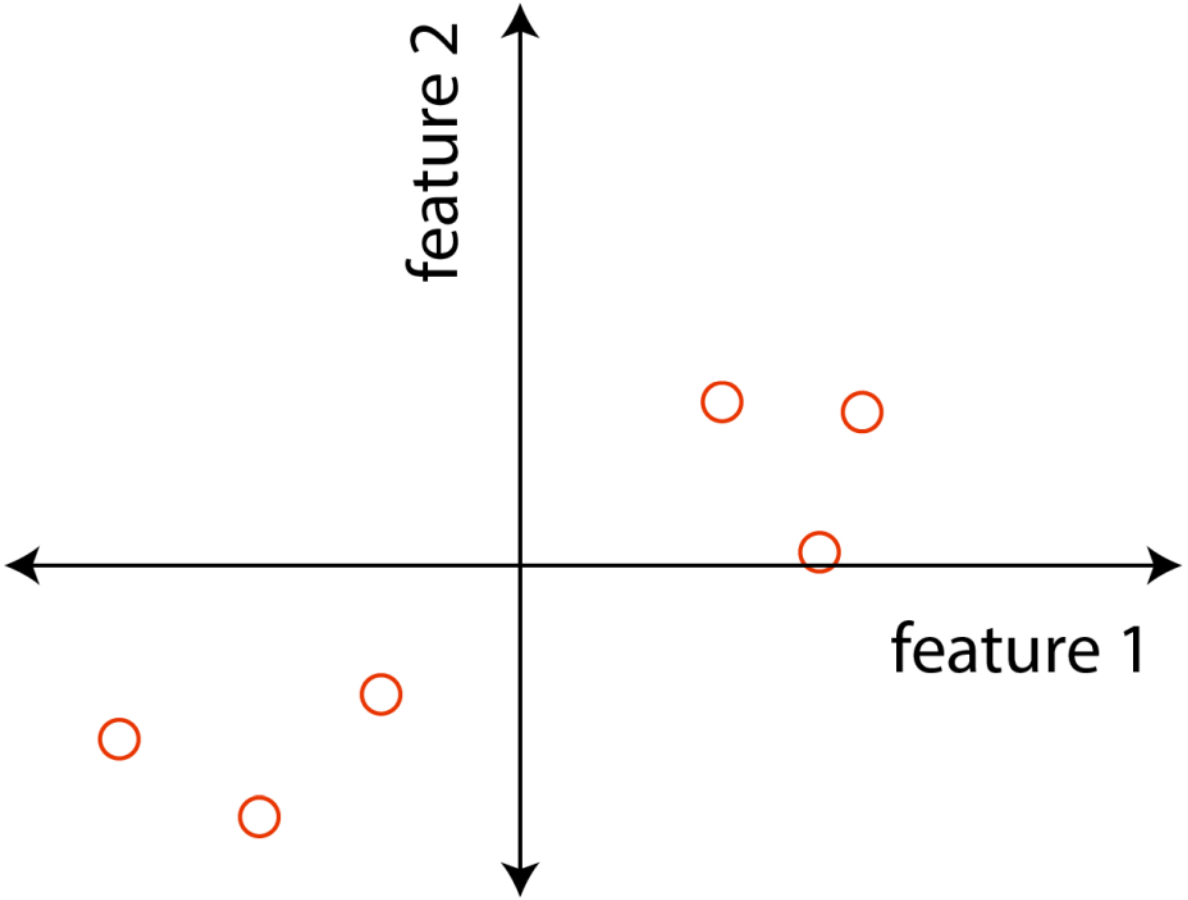
Principal component analysis is often lumped together with clustering



Steps:

1. Plot data for each variable
2. Calculate average value for each variable
3. Plot center of data
4. Move center to (0,0) origin
5. Fit line through origin and then rotate to get best fit

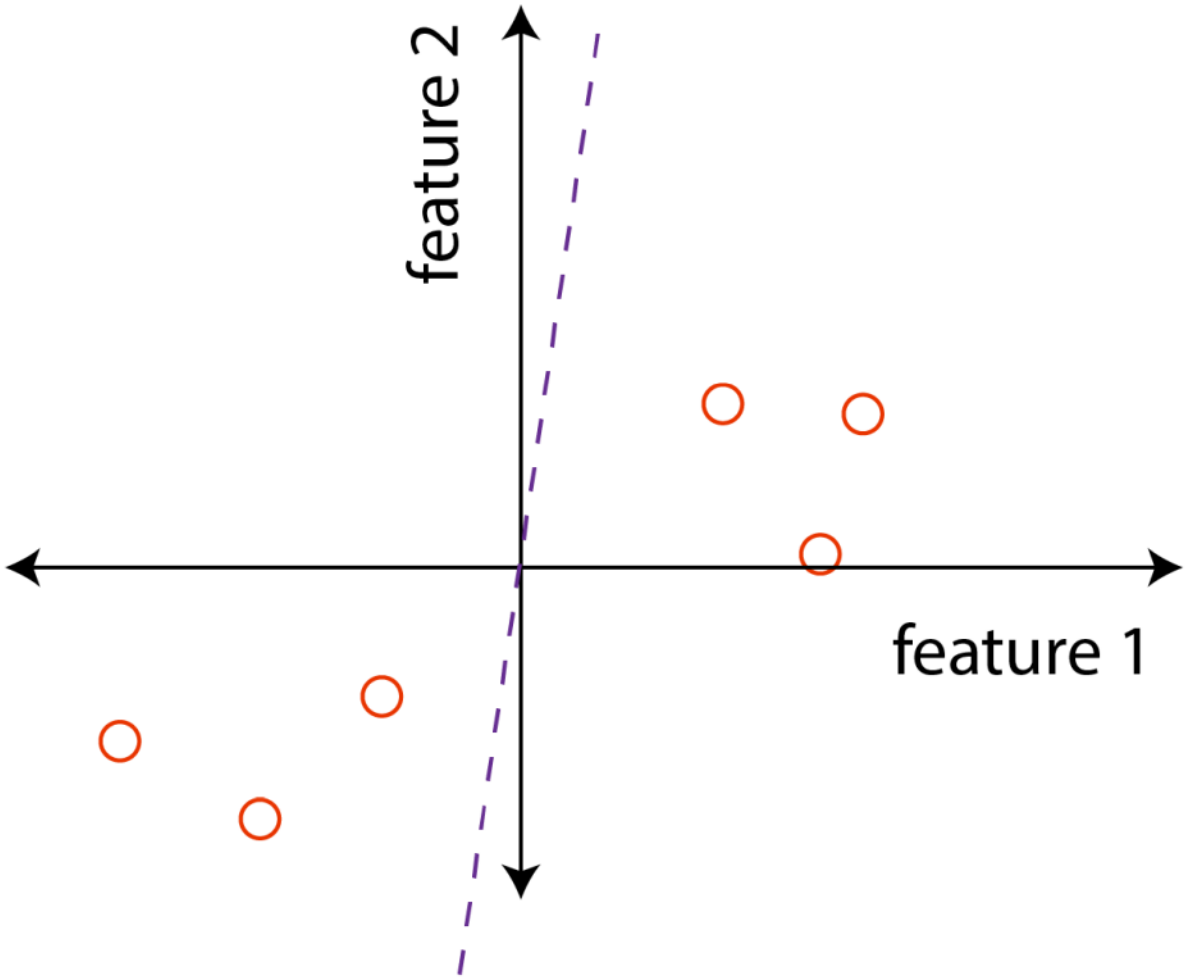
Principal component analysis is often lumped together with clustering



Steps:

1. Plot data for each variable
2. Calculate average value for each variable
3. Plot center of data
4. Move center to (0,0) origin
5. Fit line through origin and then rotate to get best fit

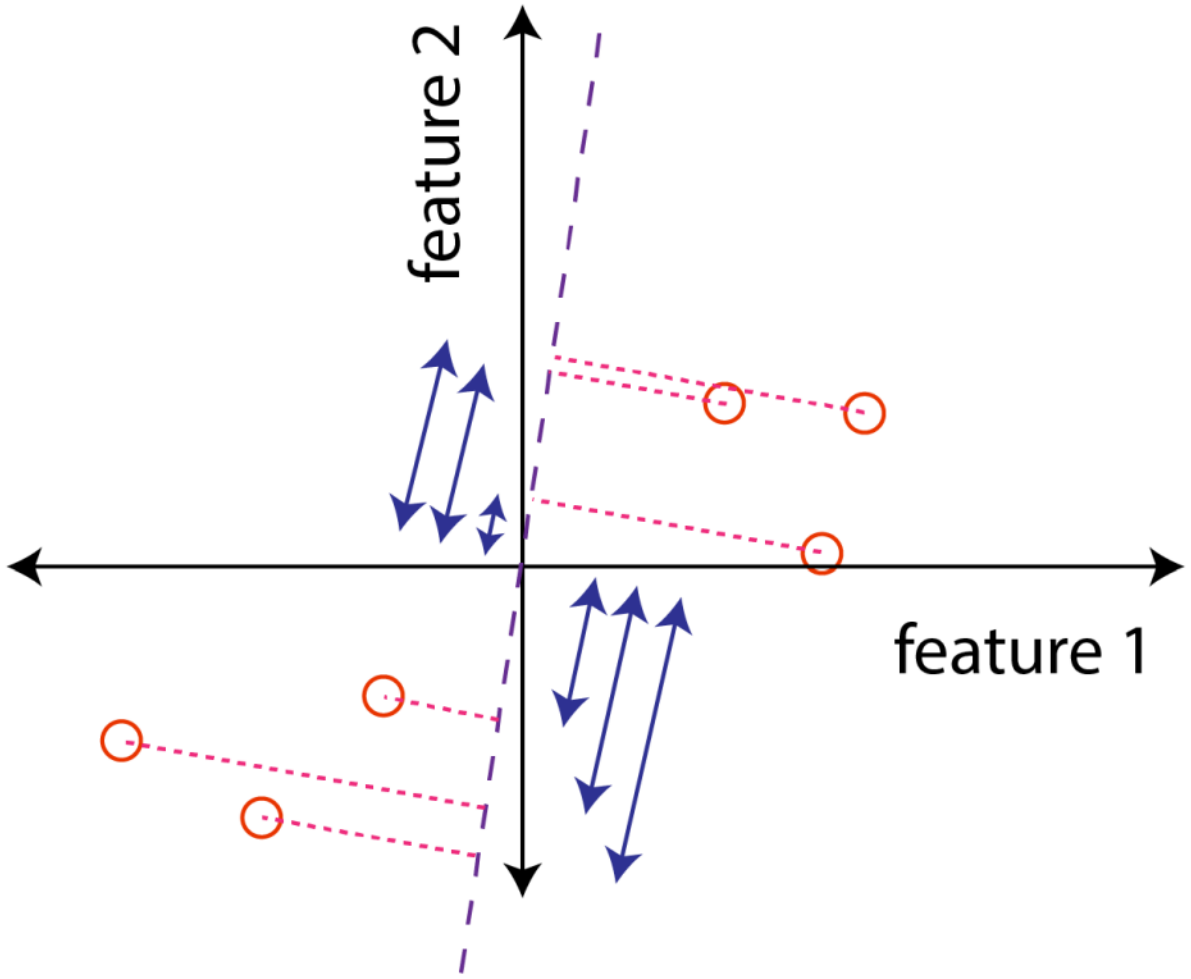
Principal component analysis is often lumped together with clustering



Steps:

1. Plot data for each variable
2. Calculate average value for each variable
3. Plot center of data
4. Move center to (0,0) origin
5. Fit line through origin and then rotate to get best fit

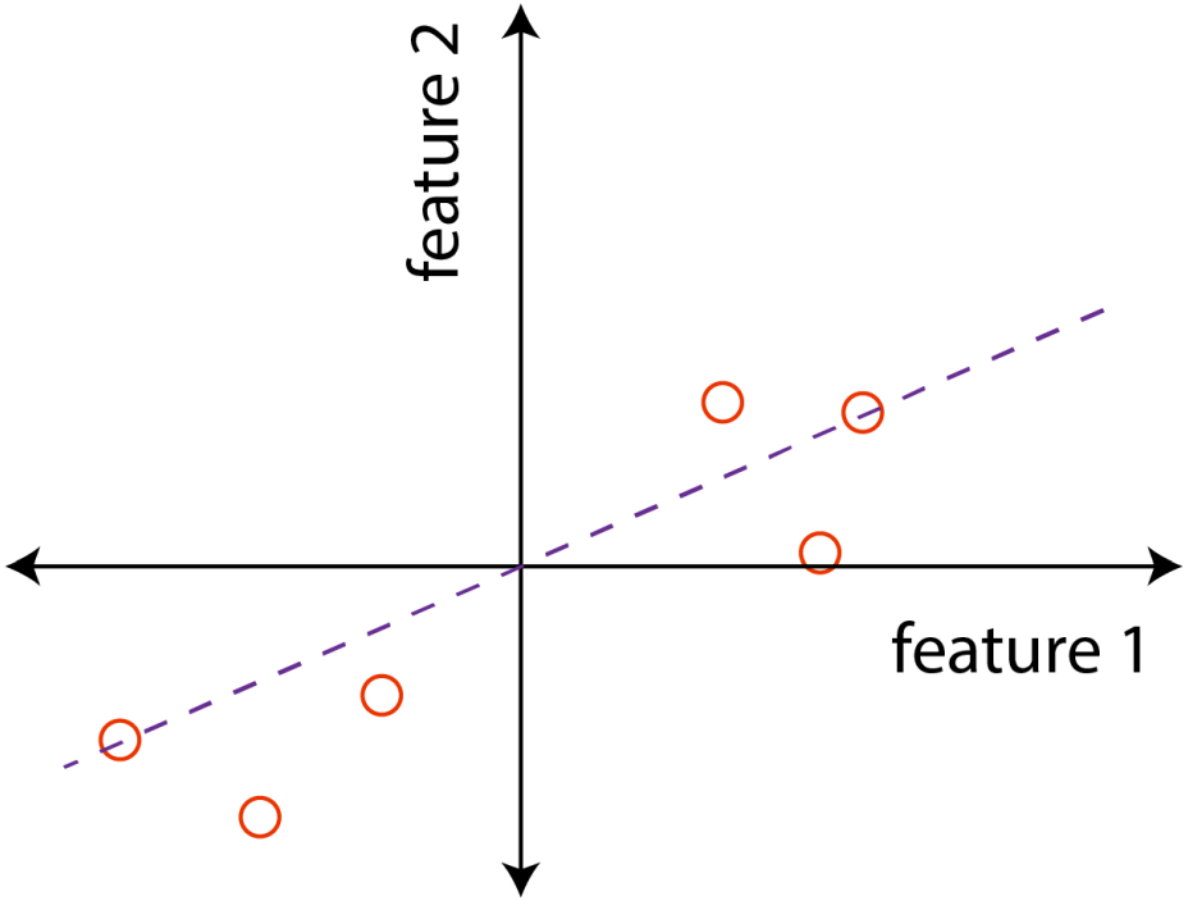
Principal component analysis is often lumped together with clustering



Steps:

1. Plot data for each variable
2. Calculate average value for each variable
3. Plot center of data
4. Move center to (0,0) origin
5. Fit line through origin and then rotate to get best fit

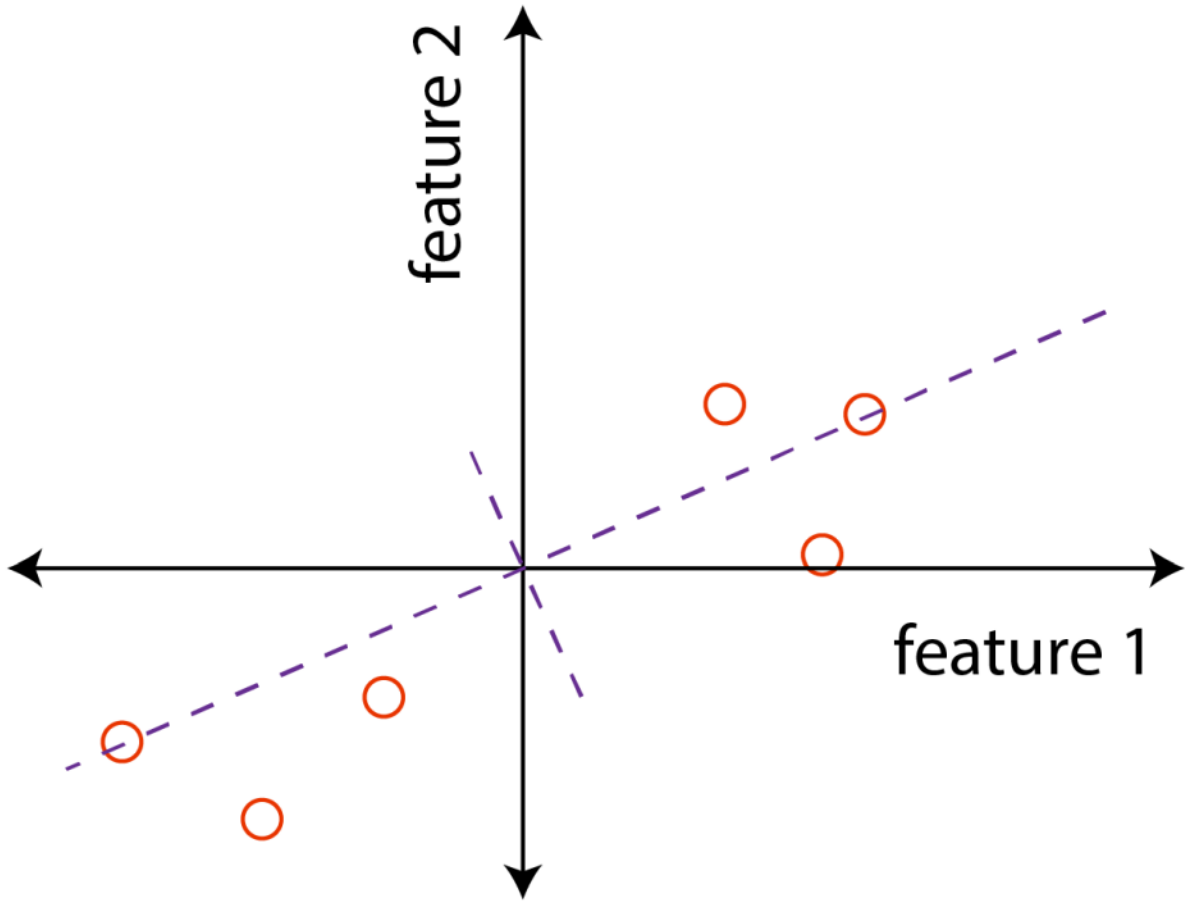
Principal component analysis is often lumped together with clustering



Steps:

1. Plot data for each variable
2. Calculate average value for each variable
3. Plot center of data
4. Move center to (0,0) origin
5. Fit line through origin and then rotate to get best fit

PCA is simply a linear combination of variables!

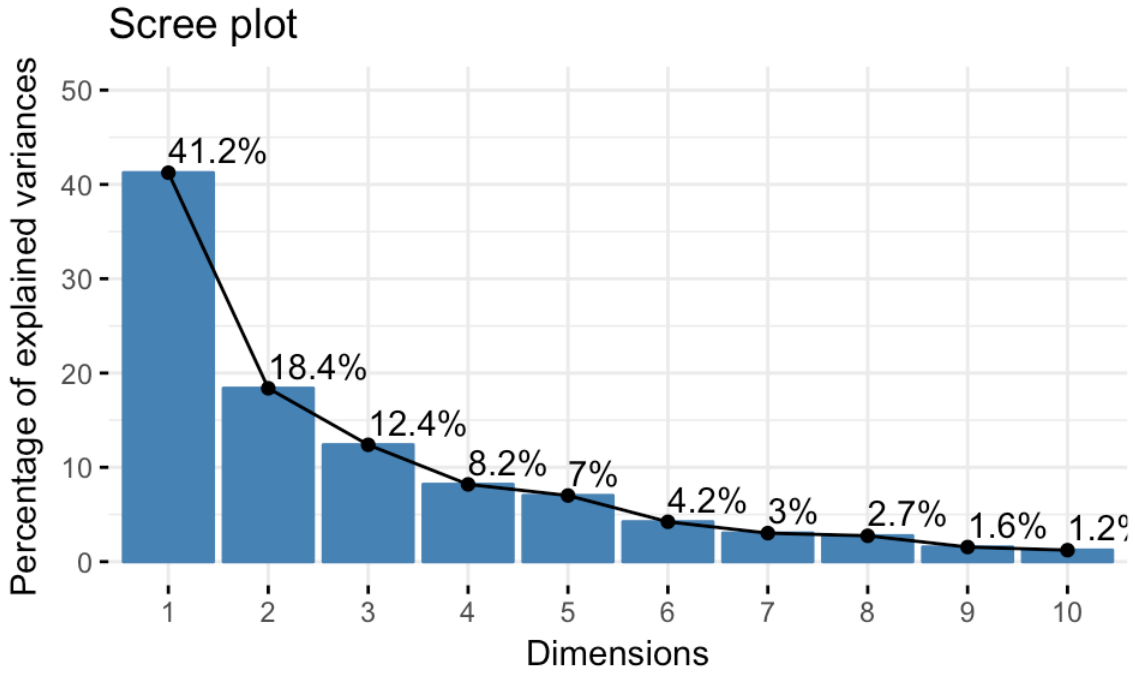


We can examine the total variation in the dataset that is due to each variable or feature

$$variation\ of\ PC1 = \frac{SS(D)\ for\ PC1}{n-1}$$

$$variation\ of\ PC2 = \frac{SS(D)\ for\ PC2}{n-1}$$

PCA can help us with feature reduction by showing us which features explain variance



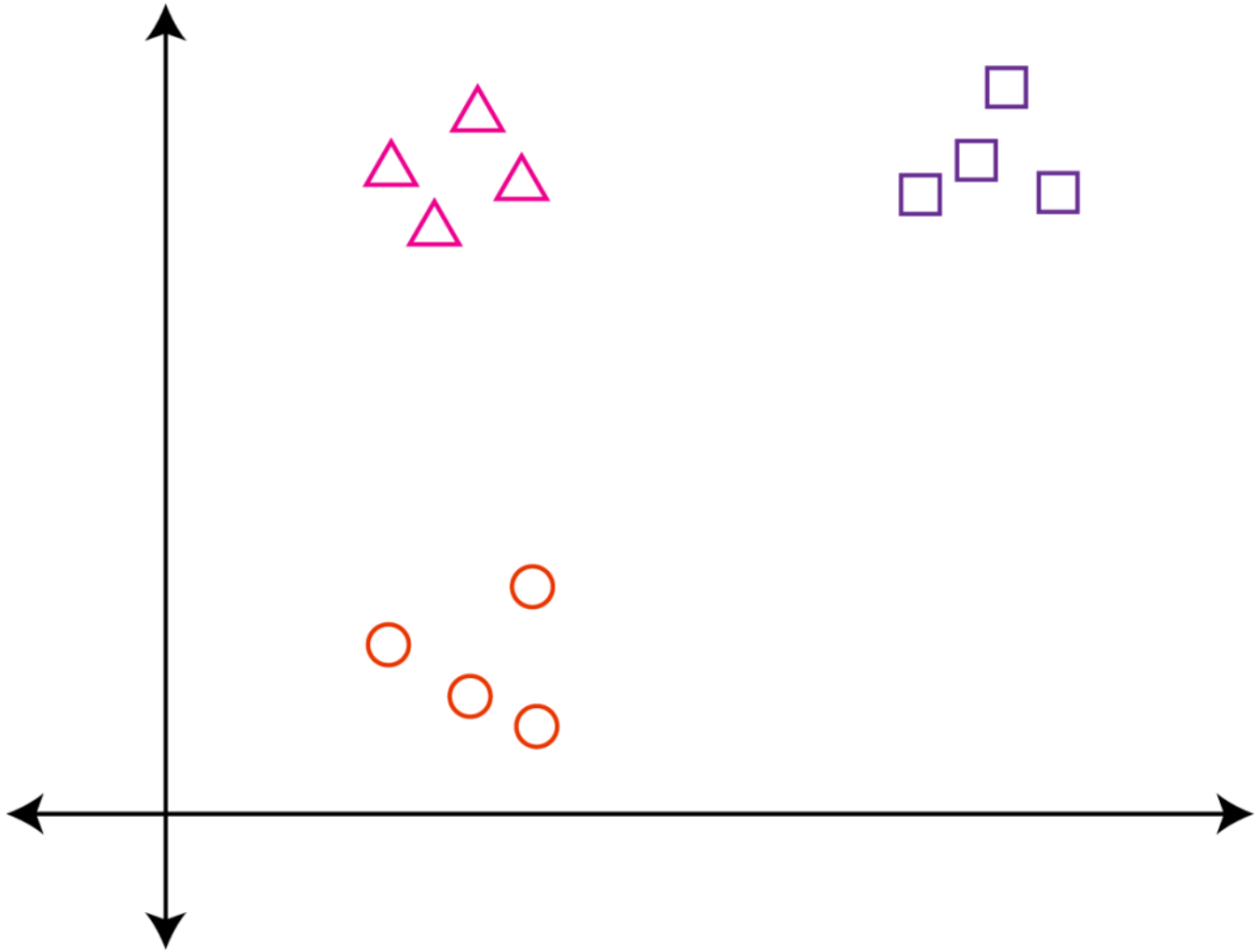
We can examine the total variation in the dataset that is due to each variable or feature

$$variation\ of\ PC1 = \frac{SS(D)\ for\ PC1}{n-1}$$

$$variation\ of\ PC2 = \frac{SS(D)\ for\ PC2}{n-1}$$

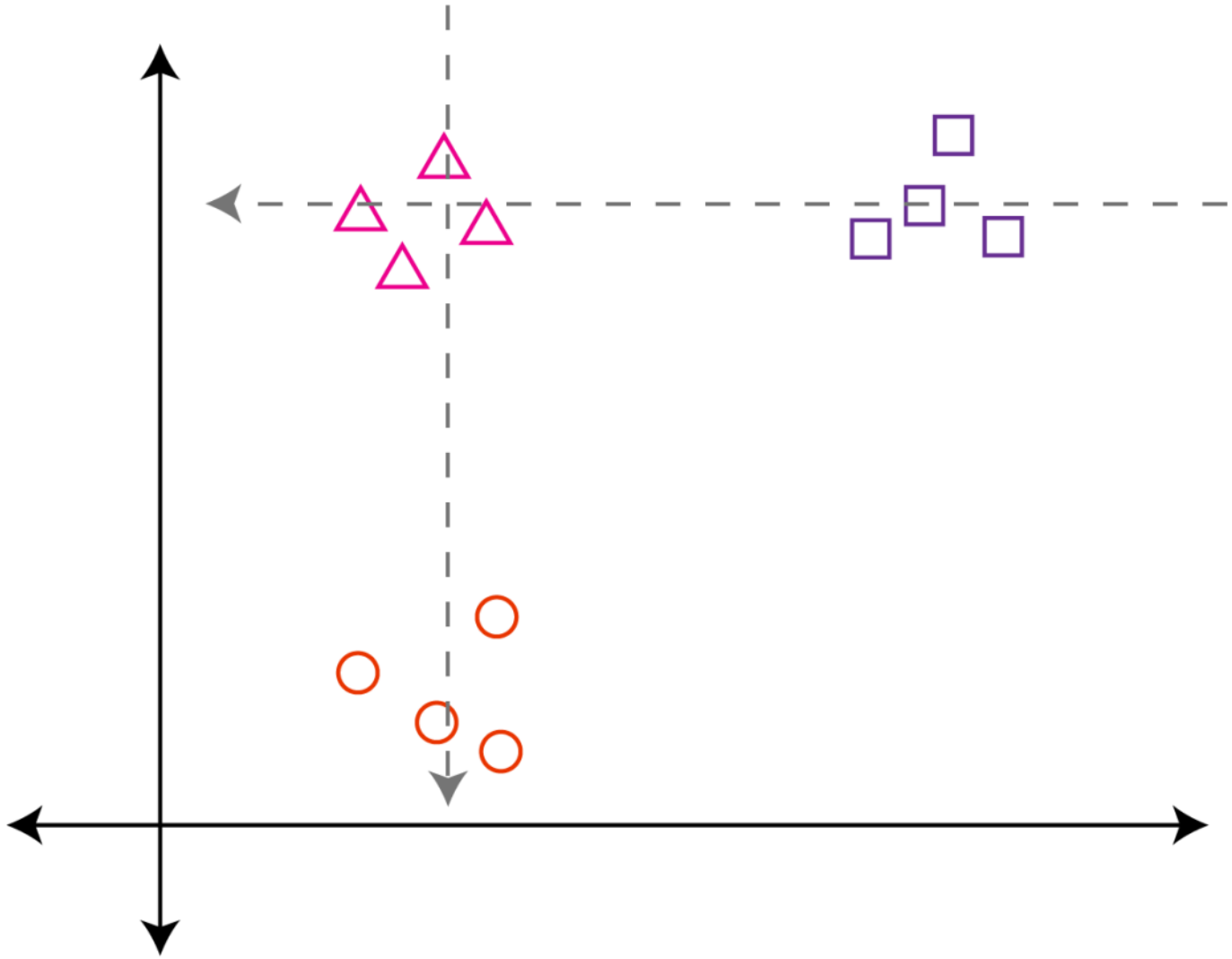
"Embedding" is taking something high dimensional and reduces it to lower dimension

tSNE is another popular clustering algorithm



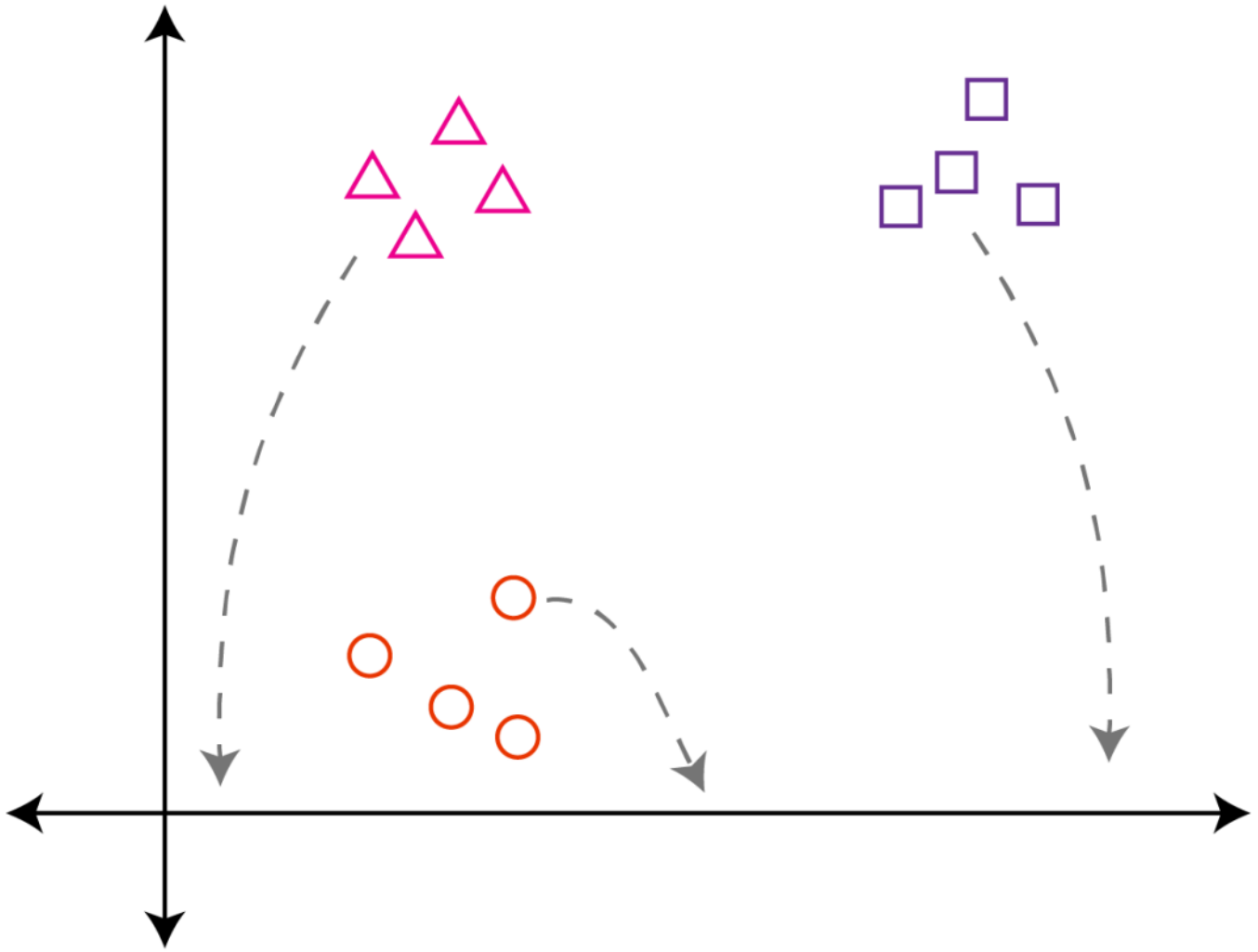
StatQuest: t-SNE, Clearly Explained, Josh Starmer, https://youtu.be/NEaUSP4YerM

Simple projection won't work to reduce dimensions because of overlap



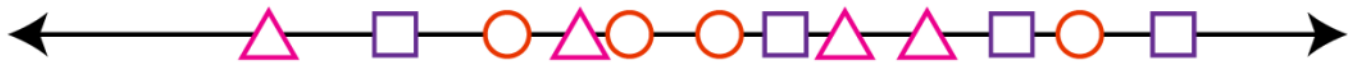
StatQuest: t-SNE, Clearly Explained, Josh Starmer, https://youtu.be/NEaUSP4YerM

We need a way to move them into regions to preserve clustering



StatQuest: t-SNE, Clearly Explained, Josh Starmer, https://youtu.be/NEaUSP4YerM

Start by moving them to random positions in a reduced dimension



#1 Incrementally move each point based on attraction to similar cluster points, and repulsion from non-cluster points

$$p_{j|i} = \frac{\exp\left(-||x_i - x_j||^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-||x_i - x_k||^2 / 2\sigma_i^2\right)}, \qquad p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N} \tag{1}$$

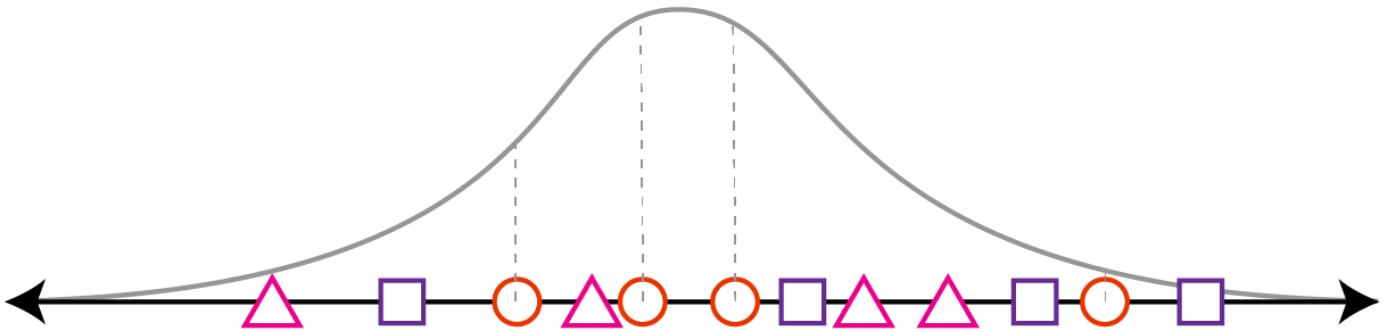
$$\text{Perplexity} = 2^{-\sum_j p_{j|i} \log_2 p_{j|i}} \tag{2}$$

$$q_{ij} = \frac{\left(1 + ||y_i - y_j||^2\right)^{-1}}{\sum_{k \neq l} \left(1 + ||y_k - y_l||^2\right)^{-1}} \tag{3}$$

$$KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}, \qquad \frac{\partial KL}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)\left(1 + ||y_i - y_j||^2\right)^{-1} \tag{4}$$

https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

Learn all relationships and then begin moving them slowly



#2 “perplexity” is introduced to preserve different cluster densities

$$p_{j|i} = \frac{\exp\left(-||x_i - x_j||^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-||x_i - x_k||^2 / 2\sigma_i^2\right)}, \qquad p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N} \tag{1}$$

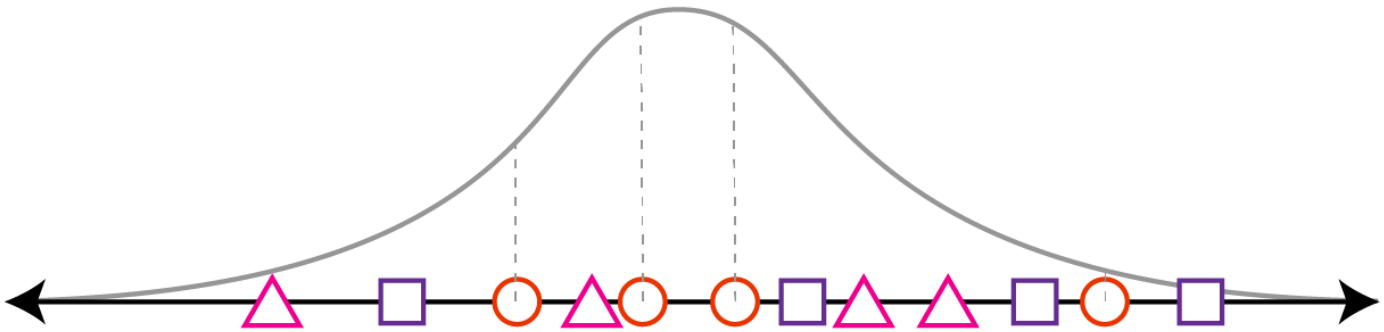
$$\text{Perplexity} = 2^{-\sum_j p_{j|i} \log_2 p_{j|i}} \tag{2}$$

$$q_{ij} = \frac{\left(1 + ||y_i - y_j||^2\right)^{-1}}{\sum_{k \neq l} \left(1 + ||y_k - y_l||^2\right)^{-1}} \tag{3}$$

$$KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}, \qquad \frac{\partial KL}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)\left(1 + ||y_i - y_j||^2\right)^{-1} \tag{4}$$

https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

Prevent center clustering by not using normal distribution and using t-distribution instead



#3 Student t-distribution is used to prevent all clusters from going to the center

$$p_{j|i} = \frac{\exp\left(-||x_i - x_j||^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-||x_i - x_k||^2 / 2\sigma_i^2\right)}, \qquad p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N} \qquad (1)$$

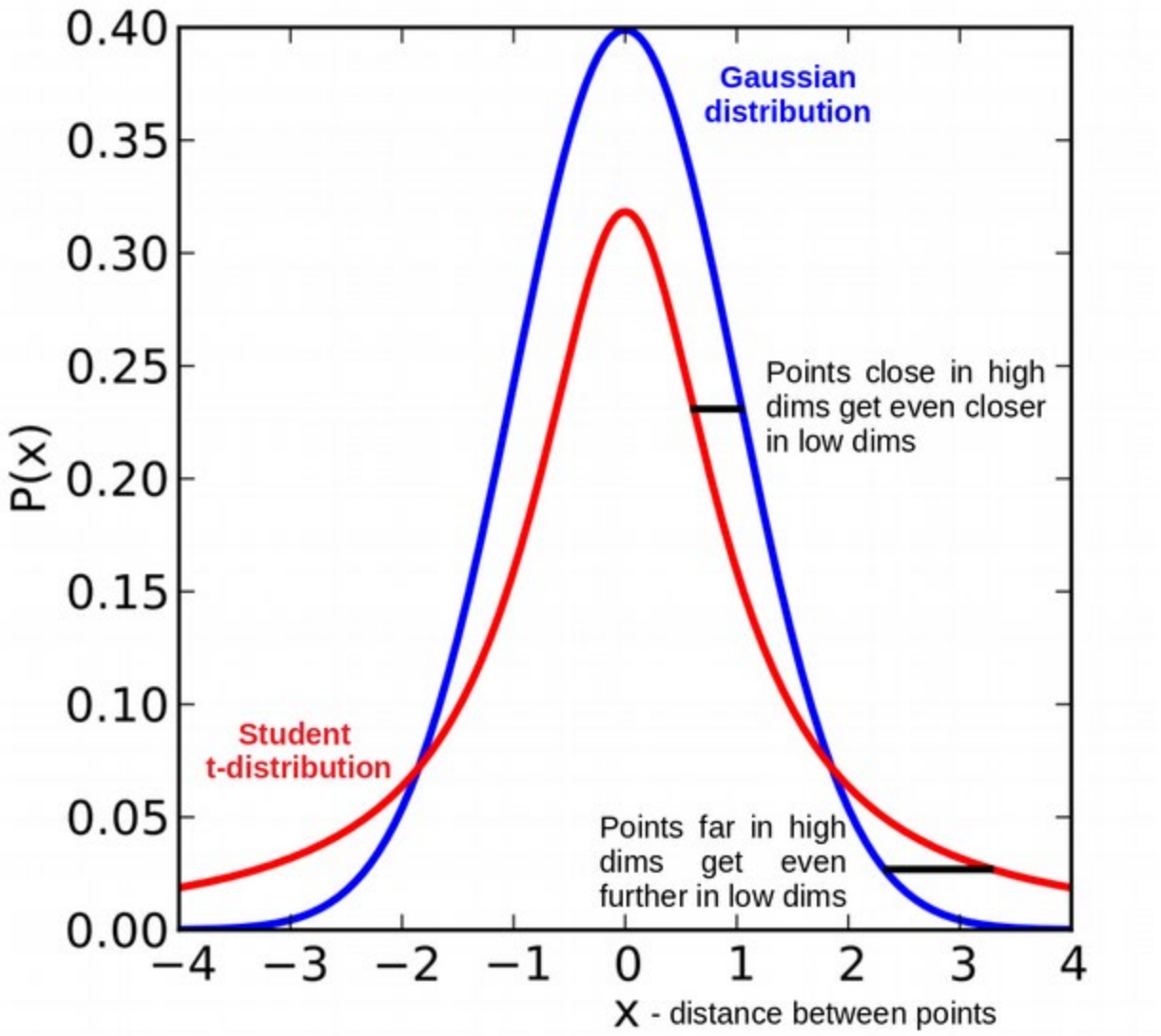
$$\text{Perplexity} = 2^{-\sum_j p_{j|i} \log_2 p_{j|i}} \qquad (2)$$

$$q_{ij} = \frac{\left(1 + ||y_i - y_j||^2\right)^{-1}}{\sum_{k \neq l} \left(1 + ||y_k - y_l||^2\right)^{-1}} \qquad (3)$$

$$KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}, \qquad \frac{\partial KL}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)\left(1 + ||y_i - y_j||^2\right)^{-1} \qquad (4)$$

https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

Prevent center clustering by not using normal distribution and using t-distribution instead



https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

Identify a loss function to utilize for gradient descent approaches

#4 Kullback-Leibler loss function used to reduce dimensionality and gradient when using gradient descent

$$p_{j|i} = \frac{\exp\left(-||x_i - x_j||^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-||x_i - x_k||^2 / 2\sigma_i^2\right)}, \qquad p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N} \tag{1}$$

$$\text{Perplexity} = 2^{-\sum_j p_{j|i} \log_2 p_{j|i}} \tag{2}$$

$$q_{ij} = \frac{\left(1 + ||y_i - y_j||^2\right)^{-1}}{\sum_{k \neq l} \left(1 + ||y_k - y_l||^2\right)^{-1}} \tag{3}$$

$$KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}, \qquad \frac{\partial KL}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)\left(1 + ||y_i - y_j||^2\right)^{-1} \tag{4}$$

https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

tSNE has some major shortcomings

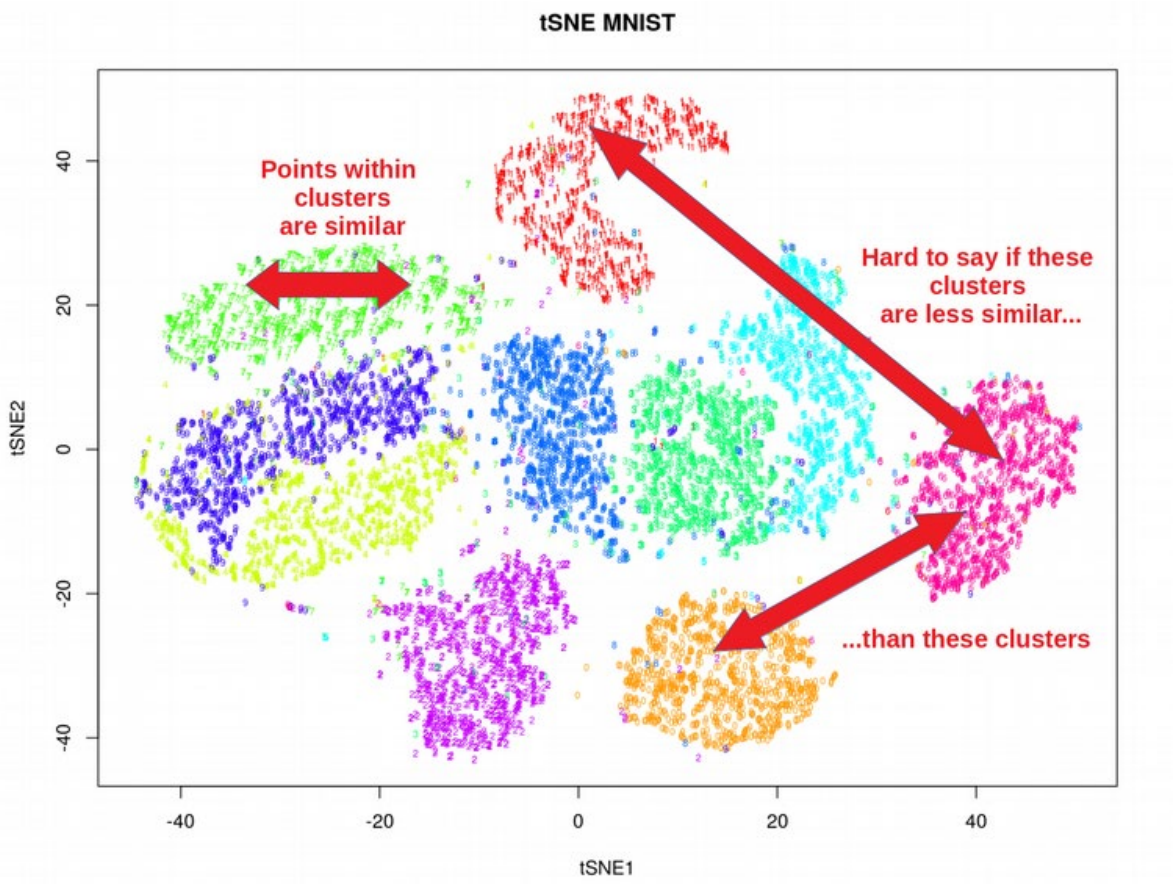
tSNE doesn't scale well

tSNE doesn't preserve global data structure

tSNE can only be embedded into 2-3 dimensions making it really for visualization, not feature reduction

tSNE performs non-parametric mapping (doesn't use feature loadings during reduction)

tSNE consumes too much memory



https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

UMAP is a more recent and improved clustering approach

UMAP starts with a high dimensional graph representation of the data,
and then optimizes a low-dimensional graph to be as similar as possible

Construct a weighted graph “fuzzy simplicial complex” where edge weights are likelihood that two points are connected

How to know if connected? Extend a radius where each radius is locally chosen based on distance of n-furthest point away

https://pair-code.github.io/understanding-umap/

Weighted graphs are created in high dimensions

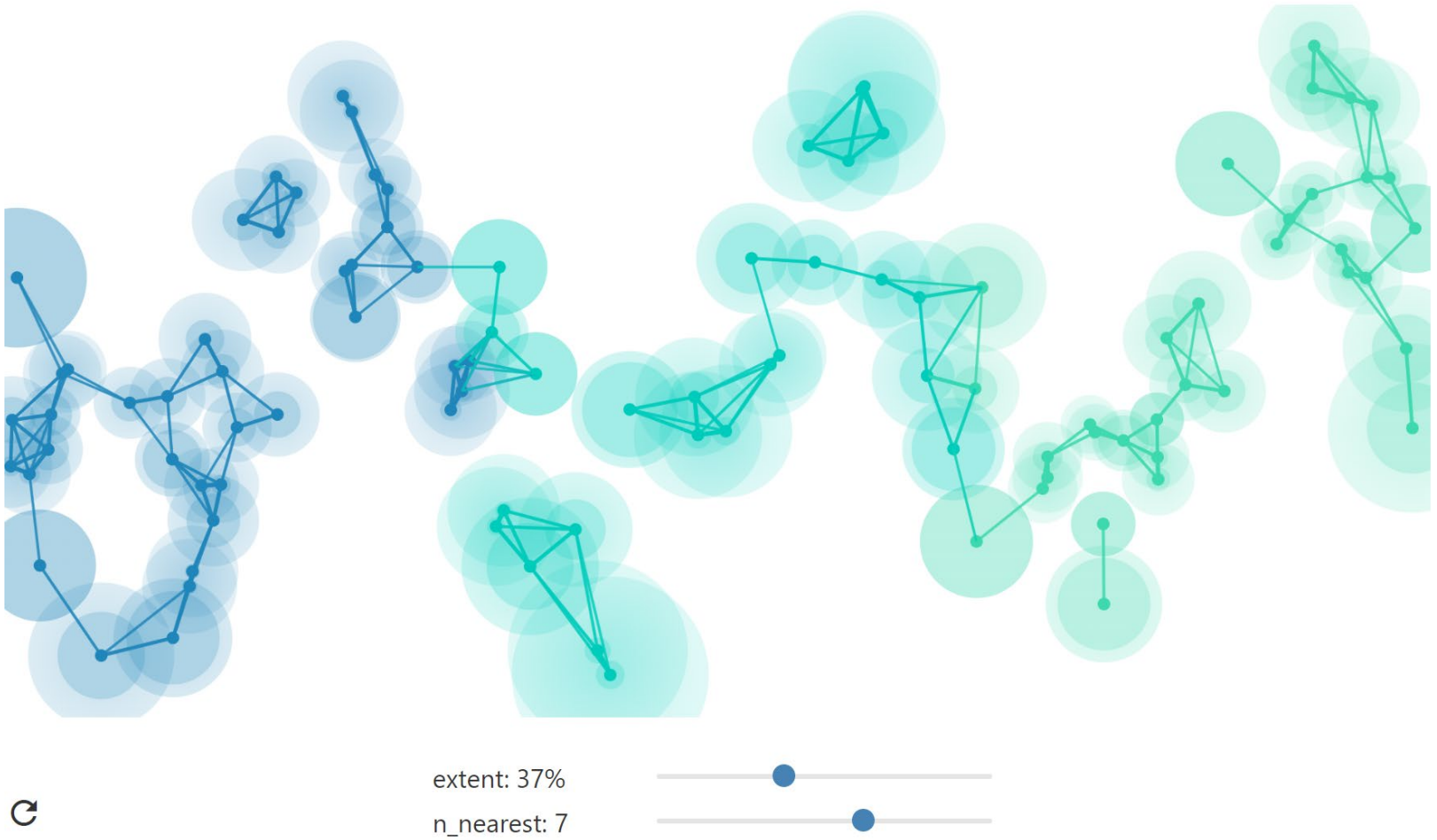


Figure 3: *Adjust the slider to extend a radius outwards from each point, computed by the distance to its nth nearest neighbor. Notice that past the intersection with the first neighbor, the radius begins to get fuzzy, with subsequent connections appearing with less weight;*

https://pair-code.github.io/understanding-umap/

Where tSNE used perplexity, UMAP uses n-nearest neighbors and minimum distance

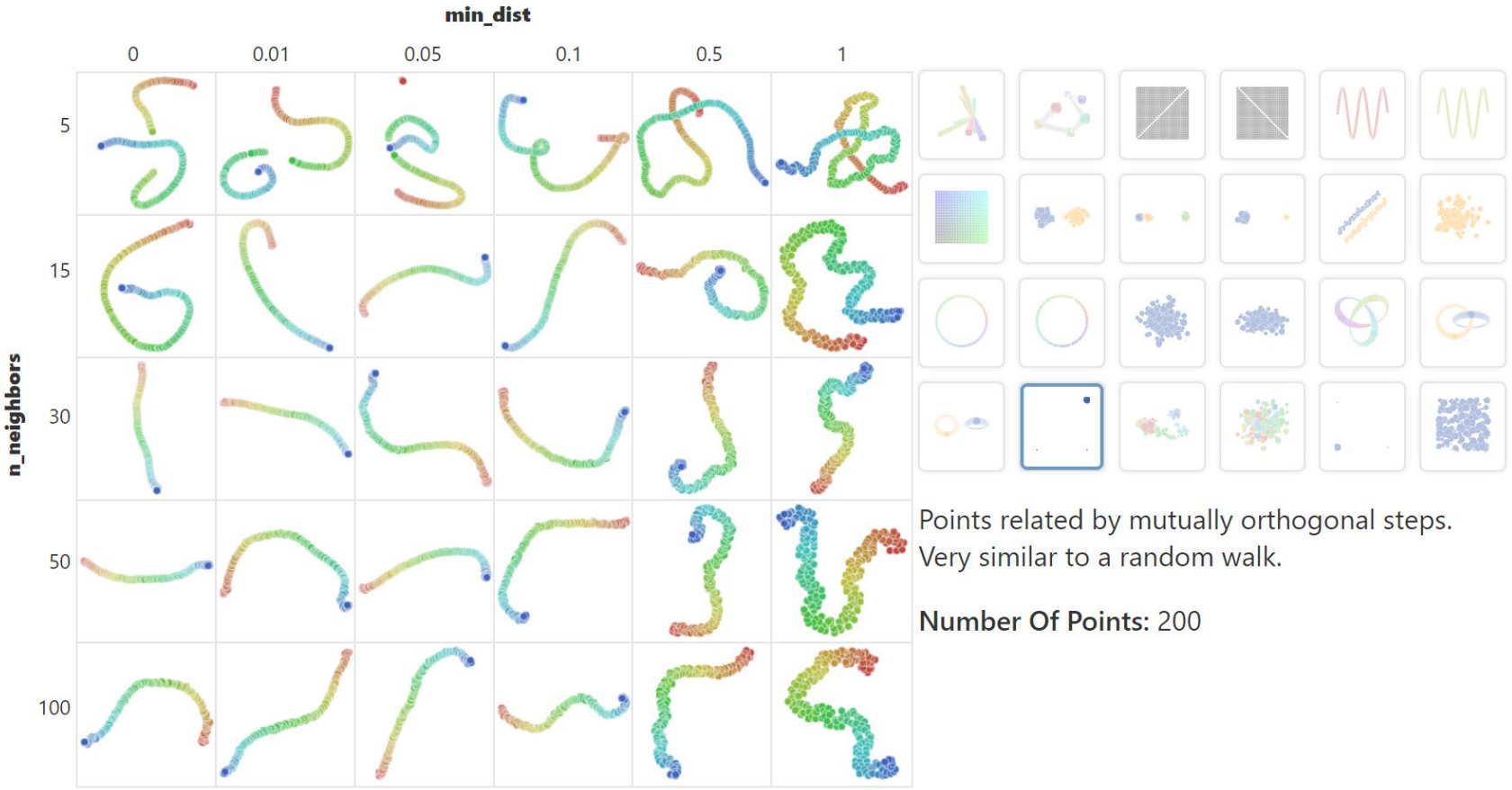


Figure 4: UMAP projection of various toy datasets with a variety of common values for the `n_neighbors` *and* `min_dist` *parameters.*

https://pair-code.github.io/understanding-umap/

Where tSNE used perplexity, UMAP uses n-nearest neighbors and minimum distance

n_neighbors controls balance between local vs global structure:
Low number focuses on local, high number focuses on global (but loses detail)

min_dist controls how tightly points get clumped together:
Low values are tightly packed embeddings, large values pack loosely to try and preserve broad topological character

“As n_neighbors increases, UMAP connects more and more neighboring points when constructing the graph representation of the high-dimensional data, which leads to a projection that more accurately reflects the global structure of the data. At very low values, any notion of global structure is almost completely lost. As the min_dist parameter increases, UMAP tends to "spread out" the projected points, leading to decreased clustering of the data and less emphasis on global structure.”

https://pair-code.github.io/understanding-umap/

There are some key takeaways to read UMAP correctly

1. Hyperparameters matter (run multiple times since it's fast)
2. Cluster size in UMAP means nothing
3. Distances between clusters might not mean anything
4. Random noise doesn't always look random
5. You may need more than one plot

https://pair-code.github.io/understanding-umap/

neural networks