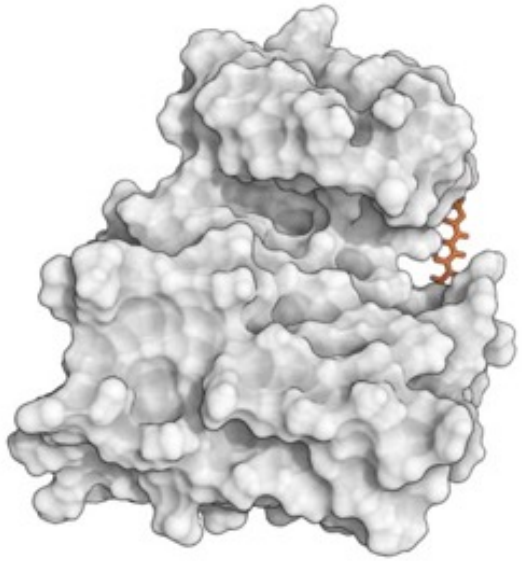
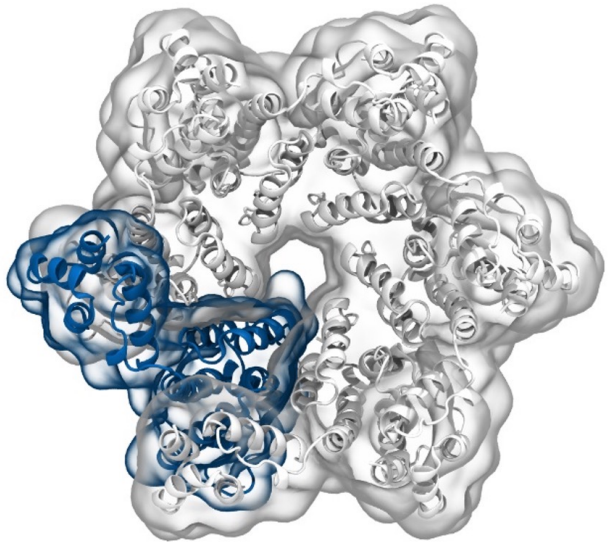
Simulation of Biomolecules



Simulation Analysis

EastBio Sneaky Peak



Dr Matteo Degiacomi
Durham University
matteo.t.degiacomi@durham.ac.uk

Dr Antonia Mey
University of Edinburgh
antonia.mey@ed.ac.uk

Large ecosystem of Python-based tools data analysis

PyTorch

scikit learn

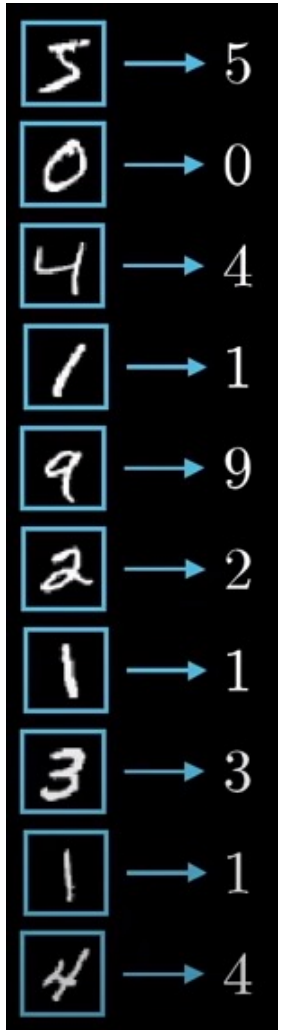
TensorFlow

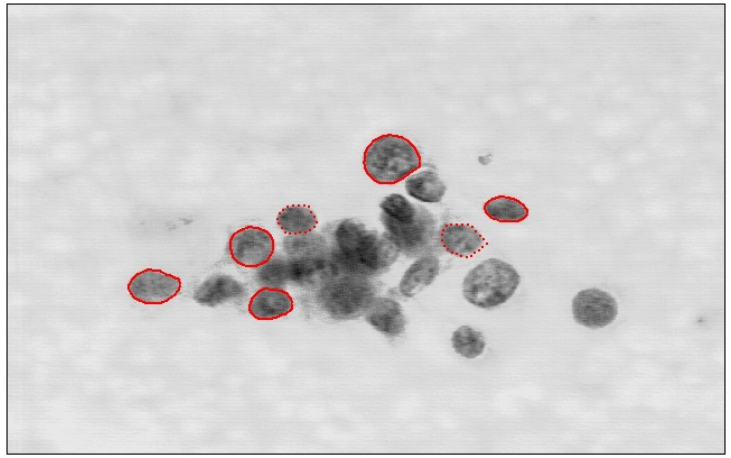
RDKit

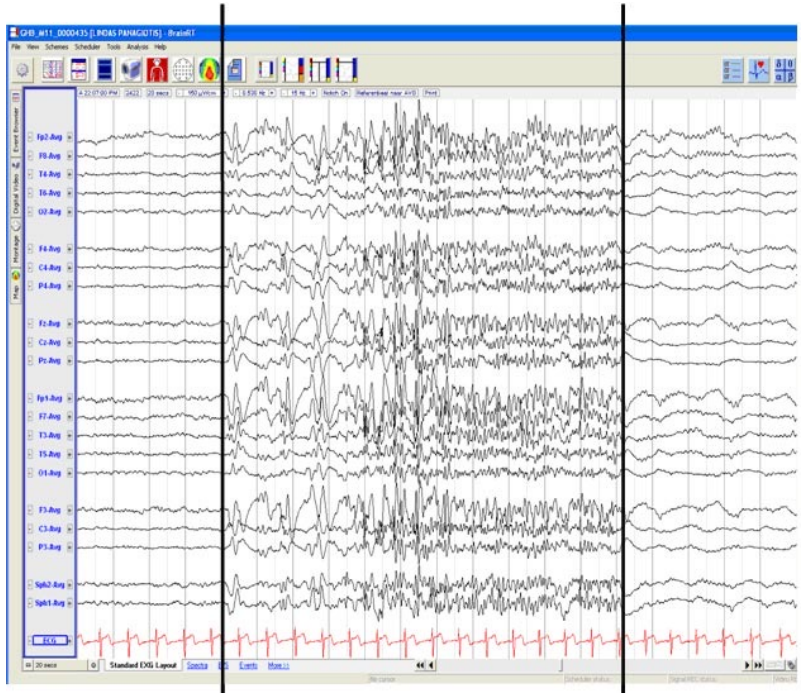
Open-Source Cheminformatics and Machine Learning

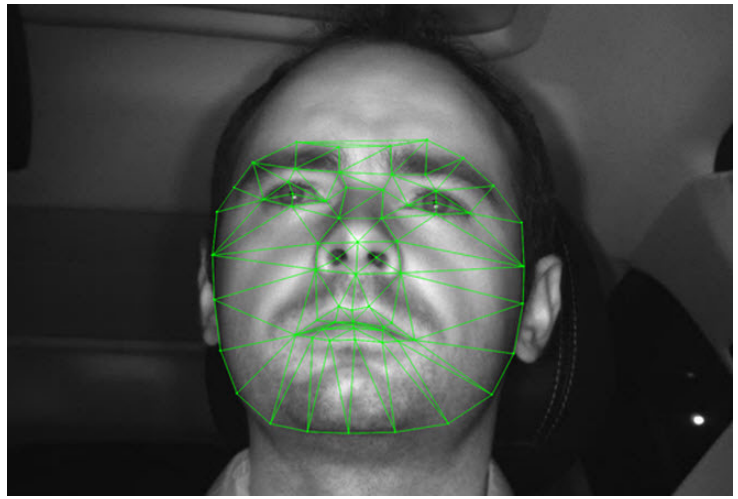
And many more…

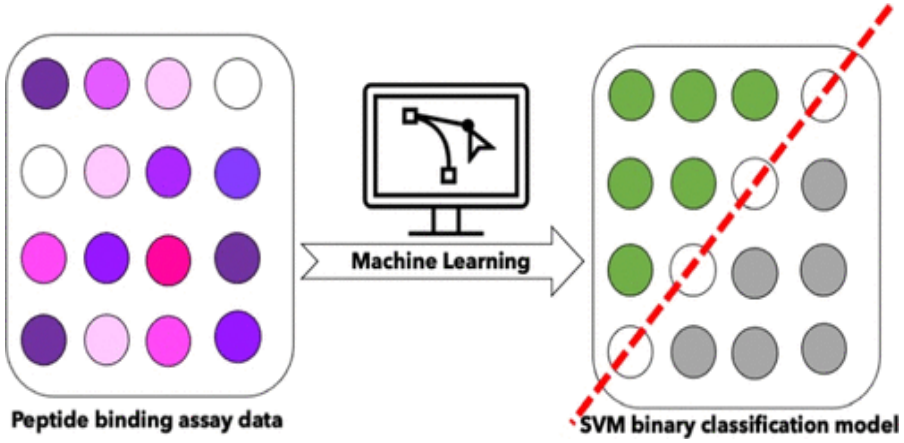
MD simulations produce complex, high-dimensional data







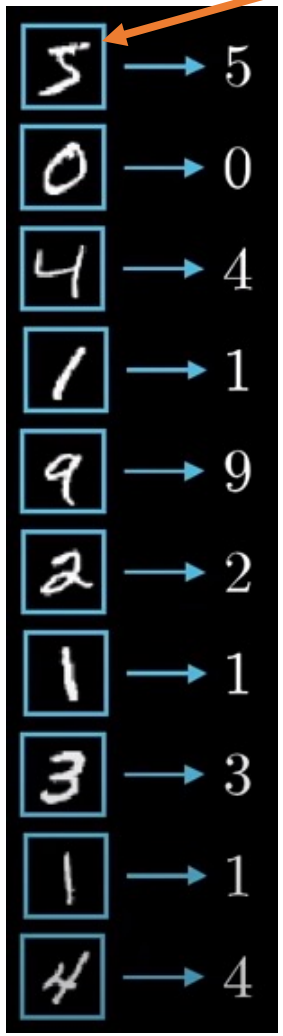


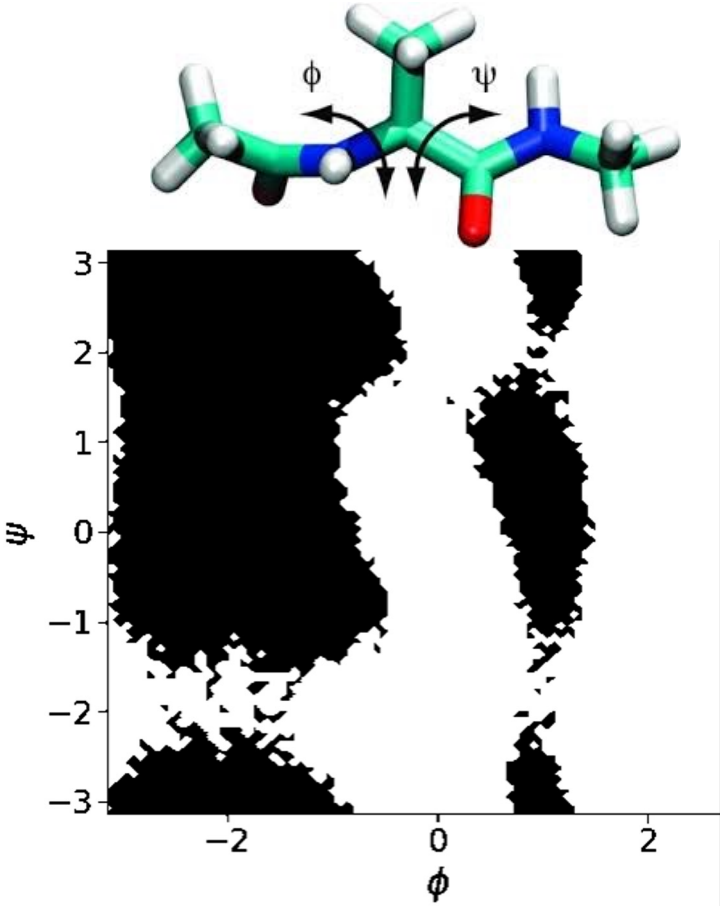


ACS Omega 2022, 7, 16, 14069–14073

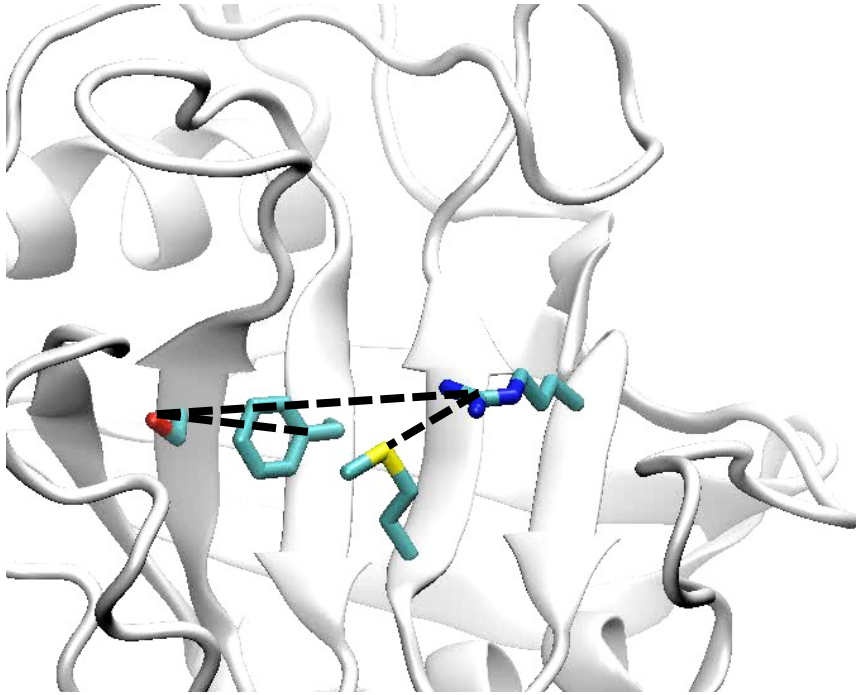
Features are possible ways to represent data

Pixels colour



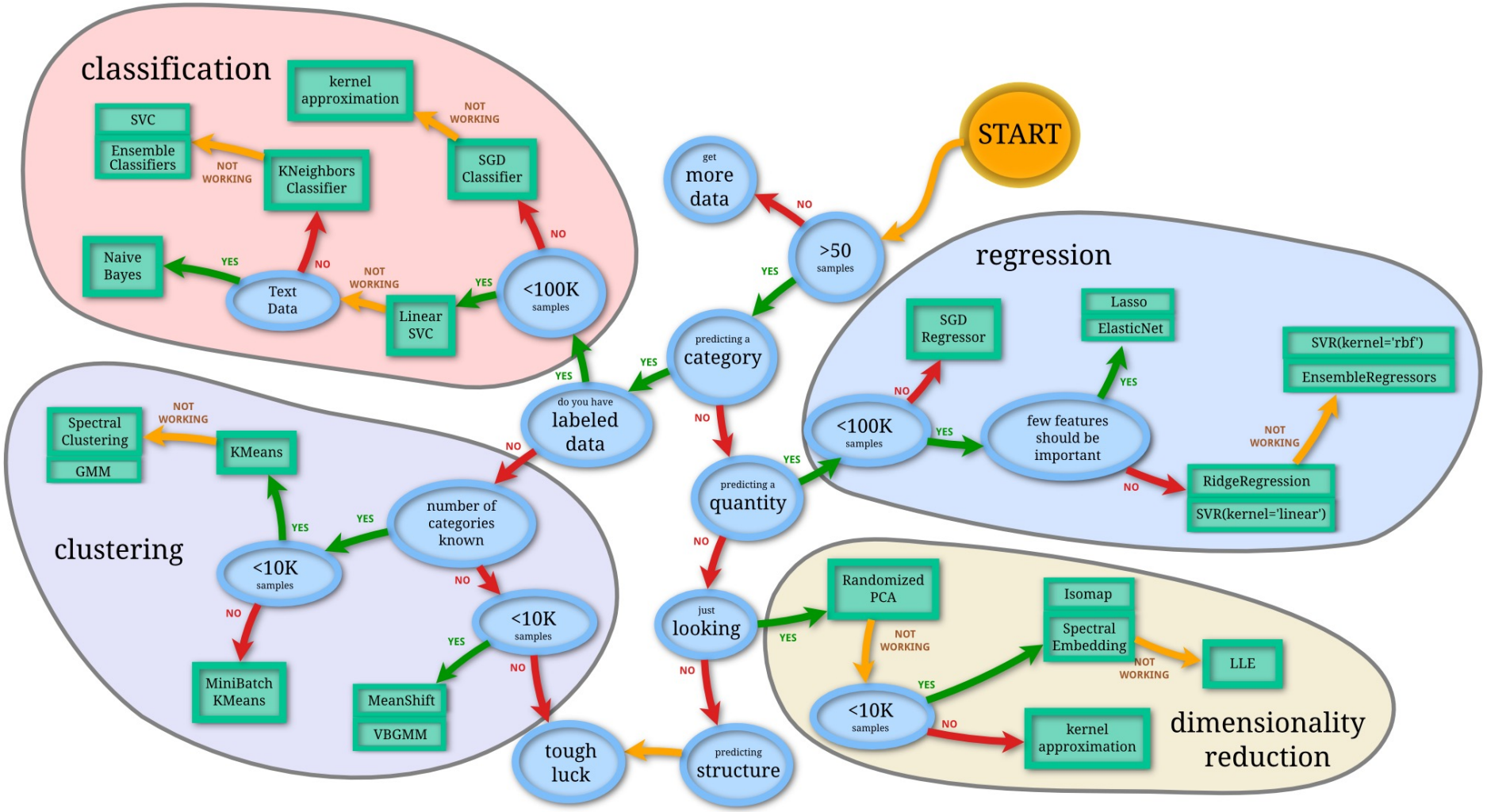


Torsional angles



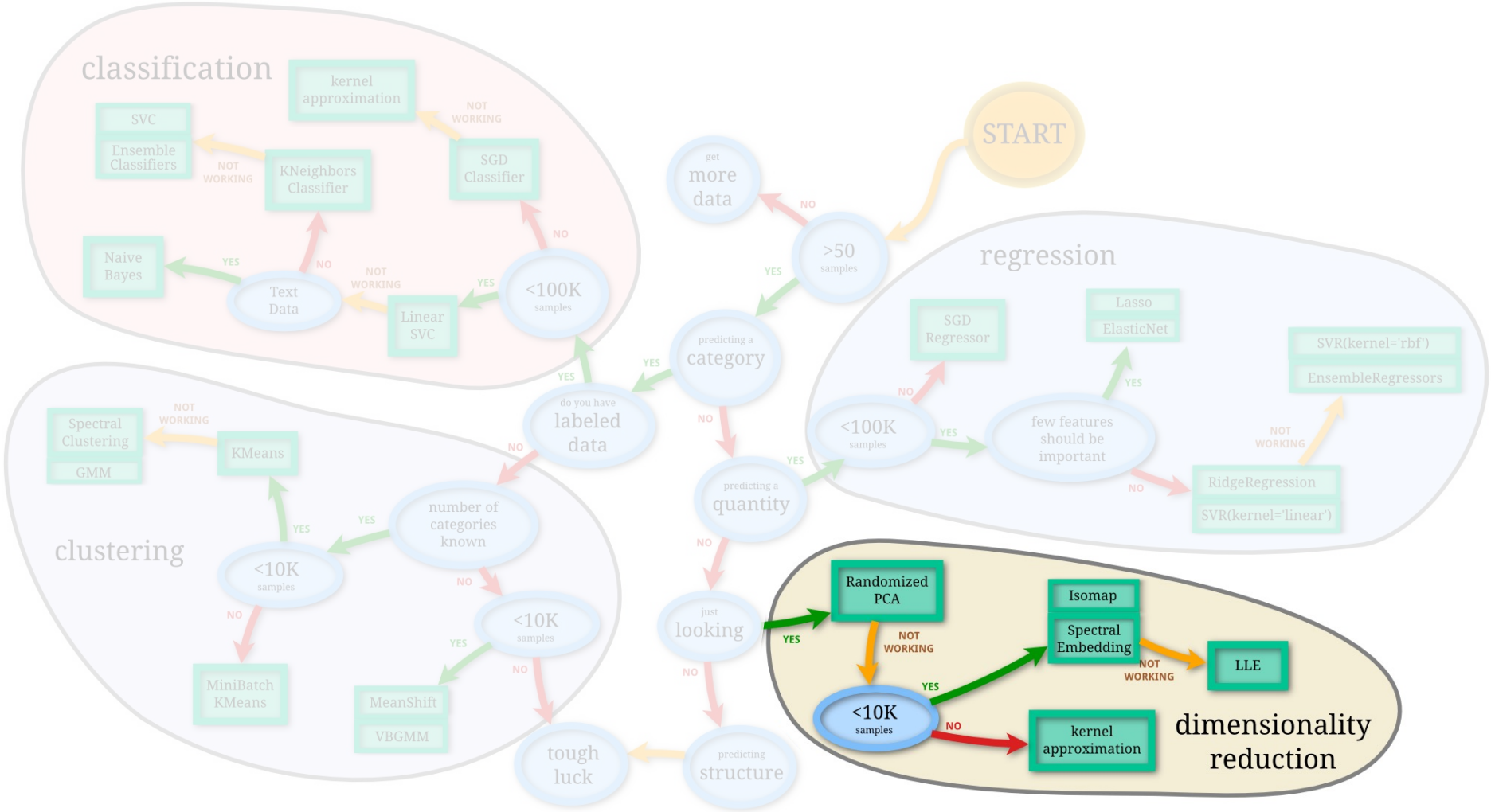
Interactomic distances

The Data Mining world



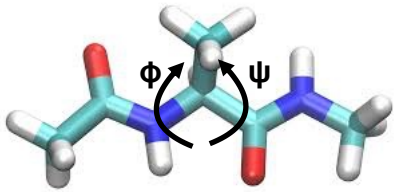
From **scikit-learn.org**

The Data Mining world

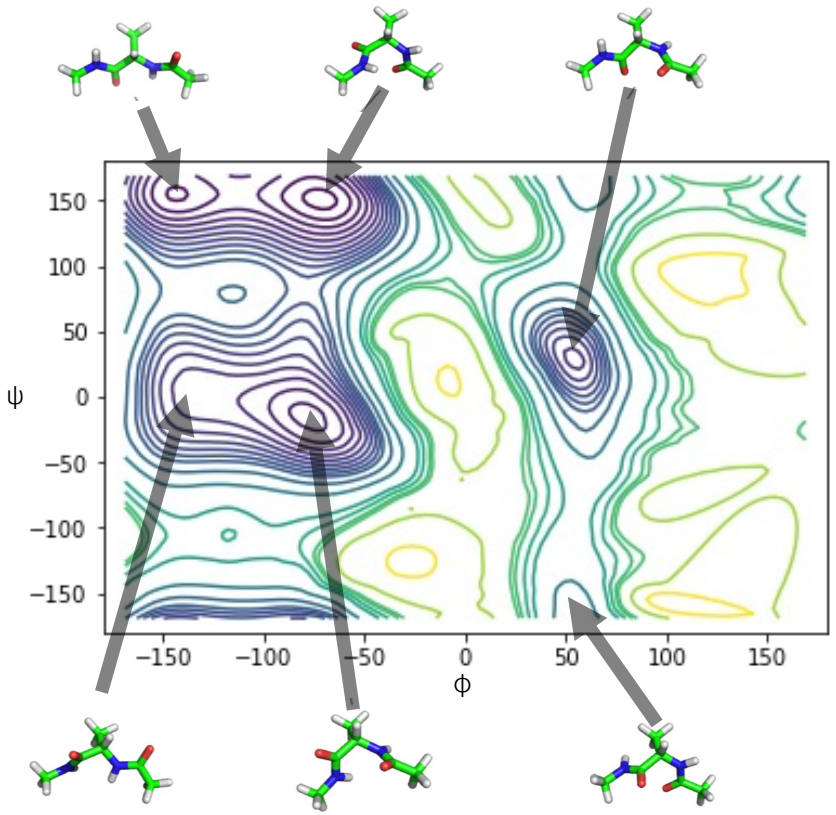


From **scikit-learn.org**

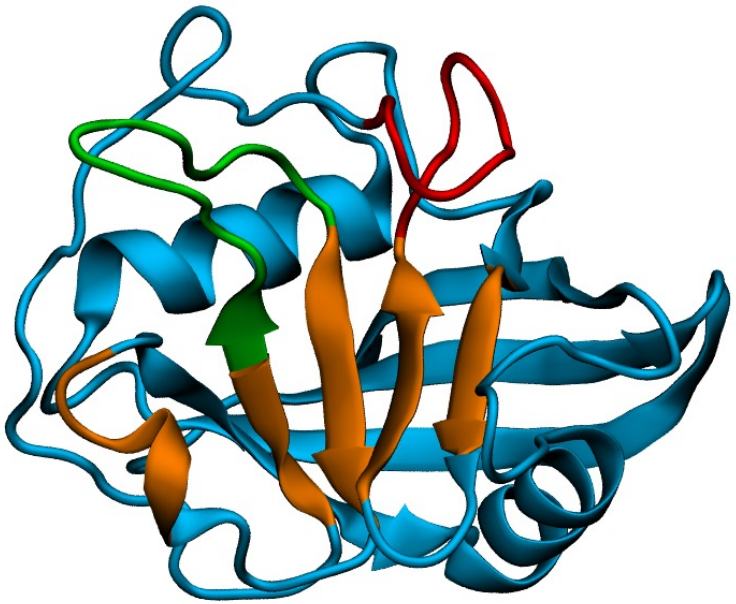
Dimensionality reduction for MD simulations



Alanine dipeptide

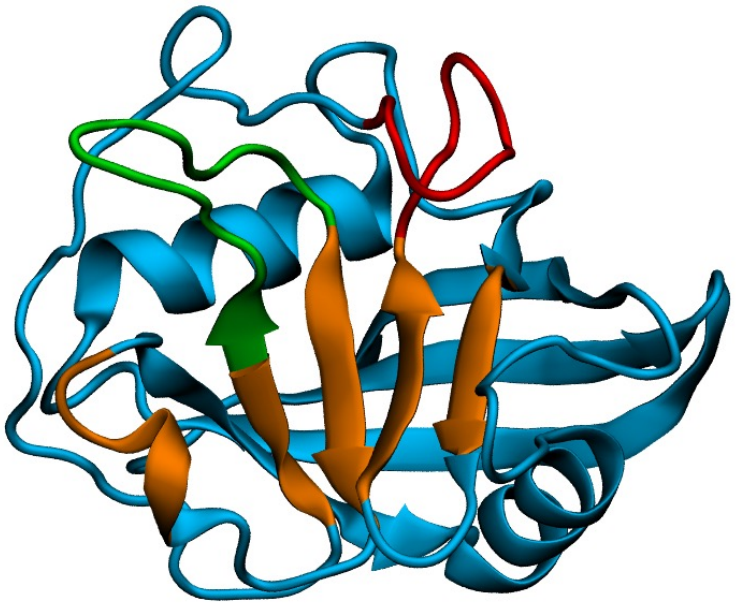


Dimensionality reduction for MD simulations

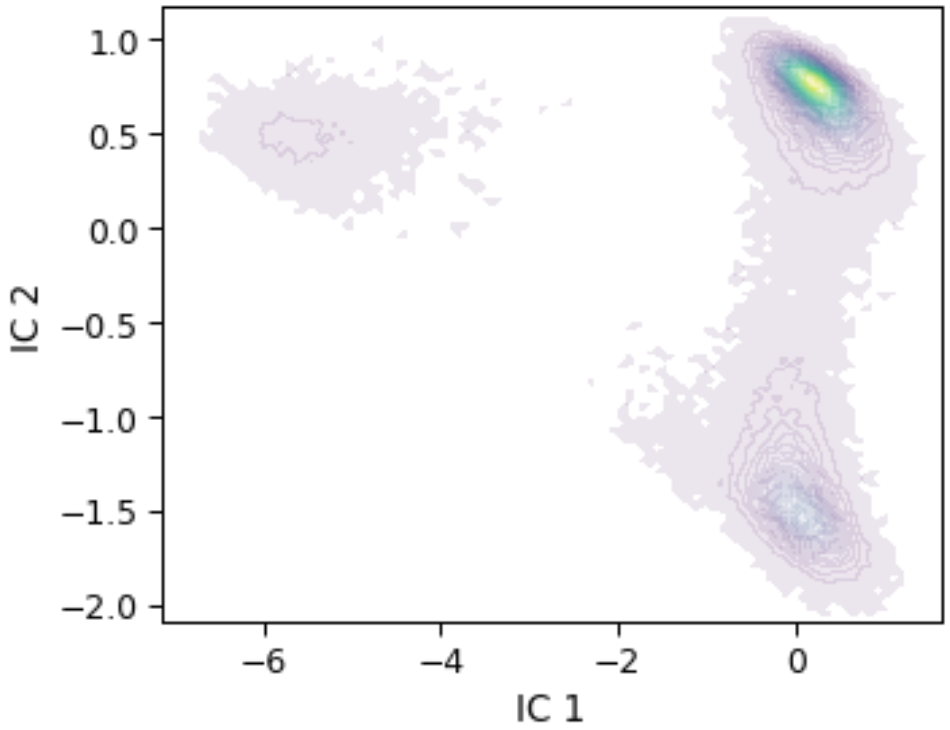


```
['ATOM:ACE 1 CH3 1 x',
 'ATOM:ACE 1 CH3 1 y',
 'ATOM:ACE 1 CH3 1 z',
 'ATOM:ACE 1 C 4 x',
 'ATOM:ACE 1 C 4 y',
 'ATOM:ACE 1 C 4 z',
 'ATOM:ACE 1 O 5 x',
 'ATOM:ACE 1 O 5 y',
 'ATOM:ACE 1 O 5 z',
 'ATOM:ALA 2 N 6 x',
 'ATOM:ALA 2 N 6 y',
 'ATOM:ALA 2 N 6 z',
 'ATOM:ALA 2 CA 8 x',
 'ATOM:ALA 2 CA 8 y',
 'ATOM:ALA 2 CA 8 z',
 'ATOM:ALA 2 CB 10 x',
 'ATOM:ALA 2 CB 10 y',
 'ATOM:ALA 2 CB 10 z',
 'ATOM:ALA 2 C 14 x',
```

Dimensionality reduction — TICA, PCA, VAMP



Project features onto low dimensional subspace



PCA: Linear combination of input features maximising the variance

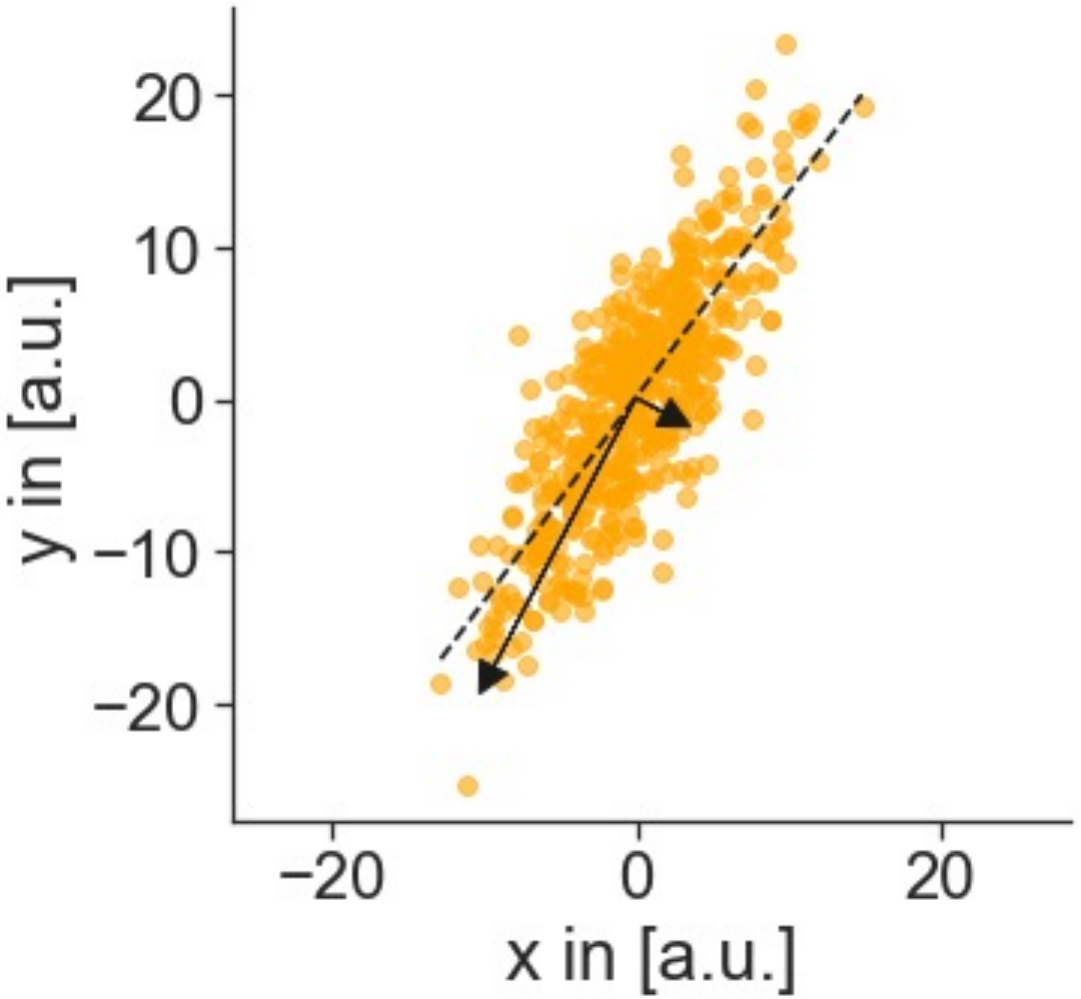
TICA: Linear combination of input features maximising time autocorrelation

VAMP: Variational approach for Markov Process, true for non-equilibrium data

VAMP: W. Chen et al., ***J. Chem. Phys.***, 2019

Principal Components Analysis (PCA)

- Finds the reference system where each dimension maximises data variance.
- *"Data is mostly spread along the first dimension, then the second, …"*
- Lowest dimensions describe noise
- First principal component does *not* align with line of best fit



PCA, formally

Let $\mathbf{C}(0)$ the covariance matrix of some data **X**

Let the generalised eigenvalue problem:

$$\mathbf{C}(0)\mathbf{W} = \mathbf{W}\boldsymbol{\Sigma}$$

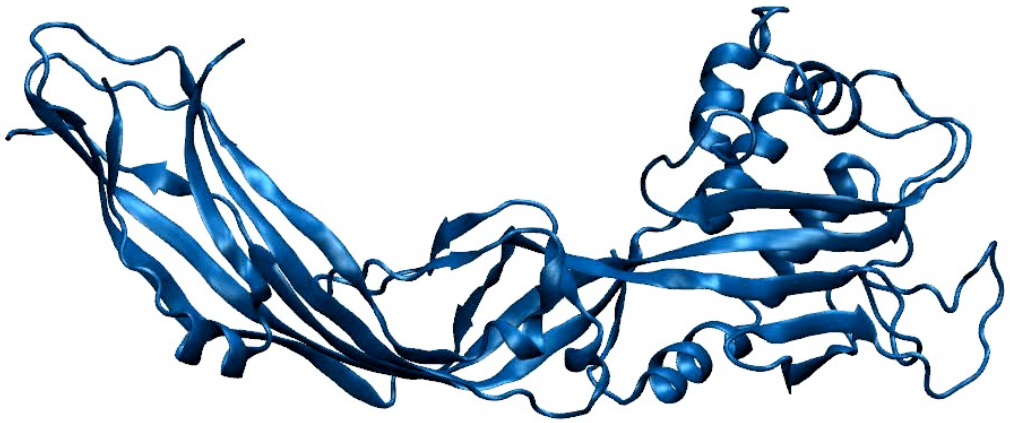
Gives an eigenvector matrix **W** that will allow the transform of the original data **X** onto a new basis **T** that maximises the variance.

$$\mathbf{T} = \mathbf{X}\mathbf{W}$$

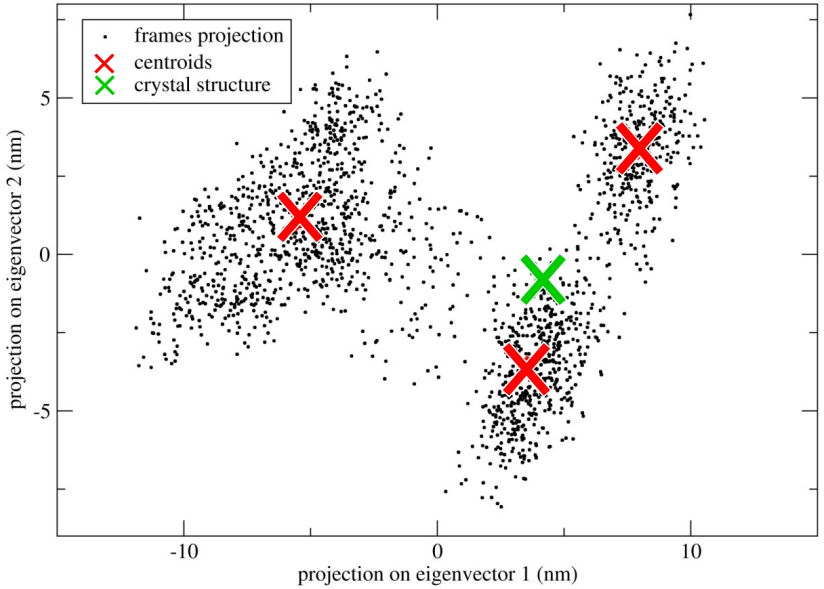
It is possible to choose m eigenvectors to project onto by only using the first m-columns of **W**.

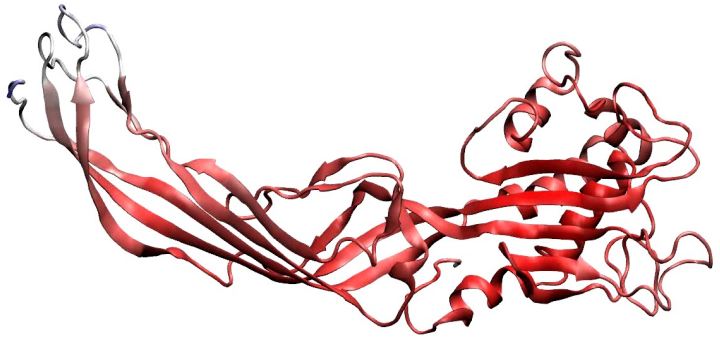
Identifying dominant motions in proteins

Protein MD simulation



Project MD in eigenspace





- Simulations are complex and noisy
- select only first few PC (eigenvectors) to separate signal from noise

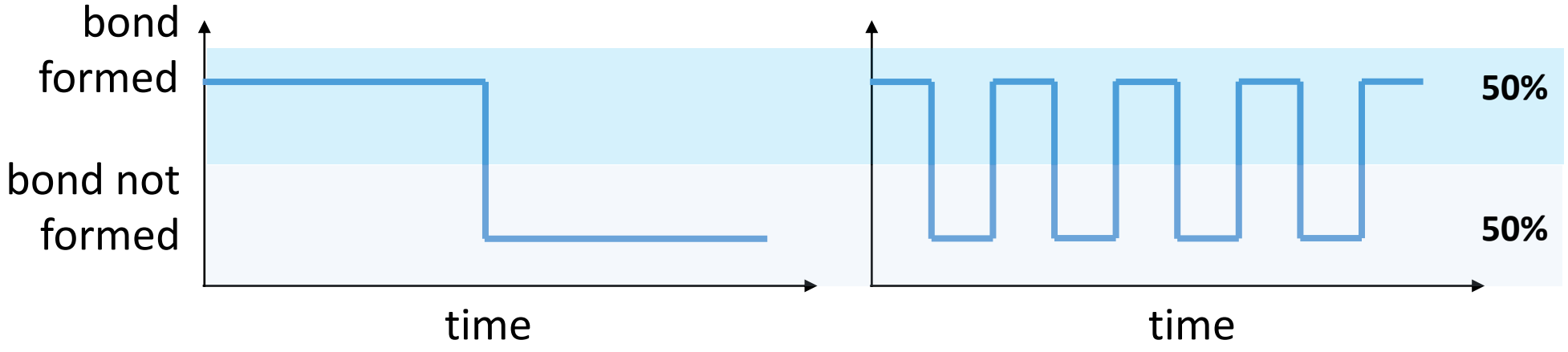
Degiacomi, M.T. & Dal Peraro, M.. *Macromolecular symmetric assembly prediction using swarm intelligence dynamic modeling*. ***Structure***, 2013

Time-lagged independent component analysis (tICA)

tICA is a **linear transform** similar to PCA

The transform is chosen such that, amongst all linear transforms, tICA **maximizes the autocorrelation** of transformed coordinates.

Thought experiment: typically hydrogen bond is considered established if donor-acceptor distance <2.5 Å, and donor-acceptor-hydrogen angle <20°.



reporting % time a bond is established in simulation can be misleading!

Time-lagged independent component analysis (tICA)

tICA is a **linear transform** similar to PCA

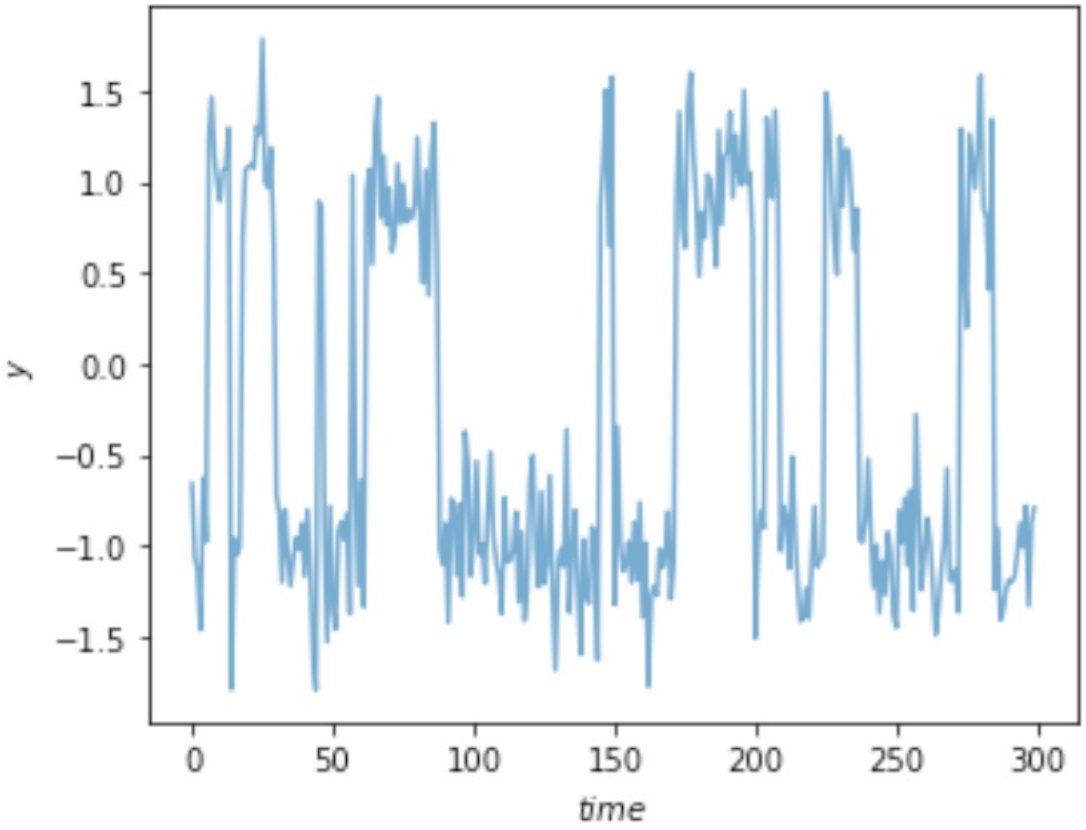
The transform is chosen such that, amongst all linear transforms, tICA **maximizes the autocorrelation** of transformed coordinates.

$\mathbf{r}(t) = (r_i(t))_{i=1,\ldots,D}$
D-dimensional input data vector that is mean free, i.e., $\mathbf{r}(t) = \mathbf{r}(t) - \langle \mathbf{r}(t) \rangle_t$

Computing the covariance of the data at $t = 0$ and $t = \tau$ which is the lag-time chosen

$c_{ij}(\tau) = \langle r_i(t) r_j(t+\tau) \rangle_t$

enables computing two covariance matrices: $\mathbf{C}(0)$ and $\mathbf{C}(\tau)$



F. Nüske, […], A. S. J. S. Mey and F. Noé: *JCTC*, 2014 G. Perez-Hernandez, et al., ***JCTC***, 2013 C. R. Schwantes et al., ***JCTC***, 2013

Time-lagged independent component analysis (tICA)

tICA is a **linear transform** similar to PCA

The transform is chosen such that, amongst all linear transforms, tICA **maximizes the autocorrelation** of transformed coordinates.

Entries of the covariance matrix can be computed as:

$$c_{ij}(\tau) = \frac{1}{N-\tau-1} \sum_{t=1}^{N-\tau} r_i(t) r_j(t+\tau)$$

$\mathbf{C}(0)$ will be a symmetric matrix. The symmetry of $\mathbf{C}(\tau)$ will need to be enforced with:

$$\mathbf{C}(\tau) = \frac{1}{2}(\mathbf{C}_d(\tau) + \mathbf{C}_d^{\mathsf{T}}(\tau))$$

We can now solve the generalised eigenvalue problem:

$$\mathbf{C}(\tau)\mathbf{U} = \mathbf{C}(0)\mathbf{U}\mathbf{\Lambda} \longrightarrow \mathbf{z}^{\mathsf{T}}(t) = \mathbf{r}^{\mathsf{T}}(t)\mathbf{U}$$

Eigenvector matrix containing ICs

Diagonal matrix with eigenvalues

M columns of full rank U for DR

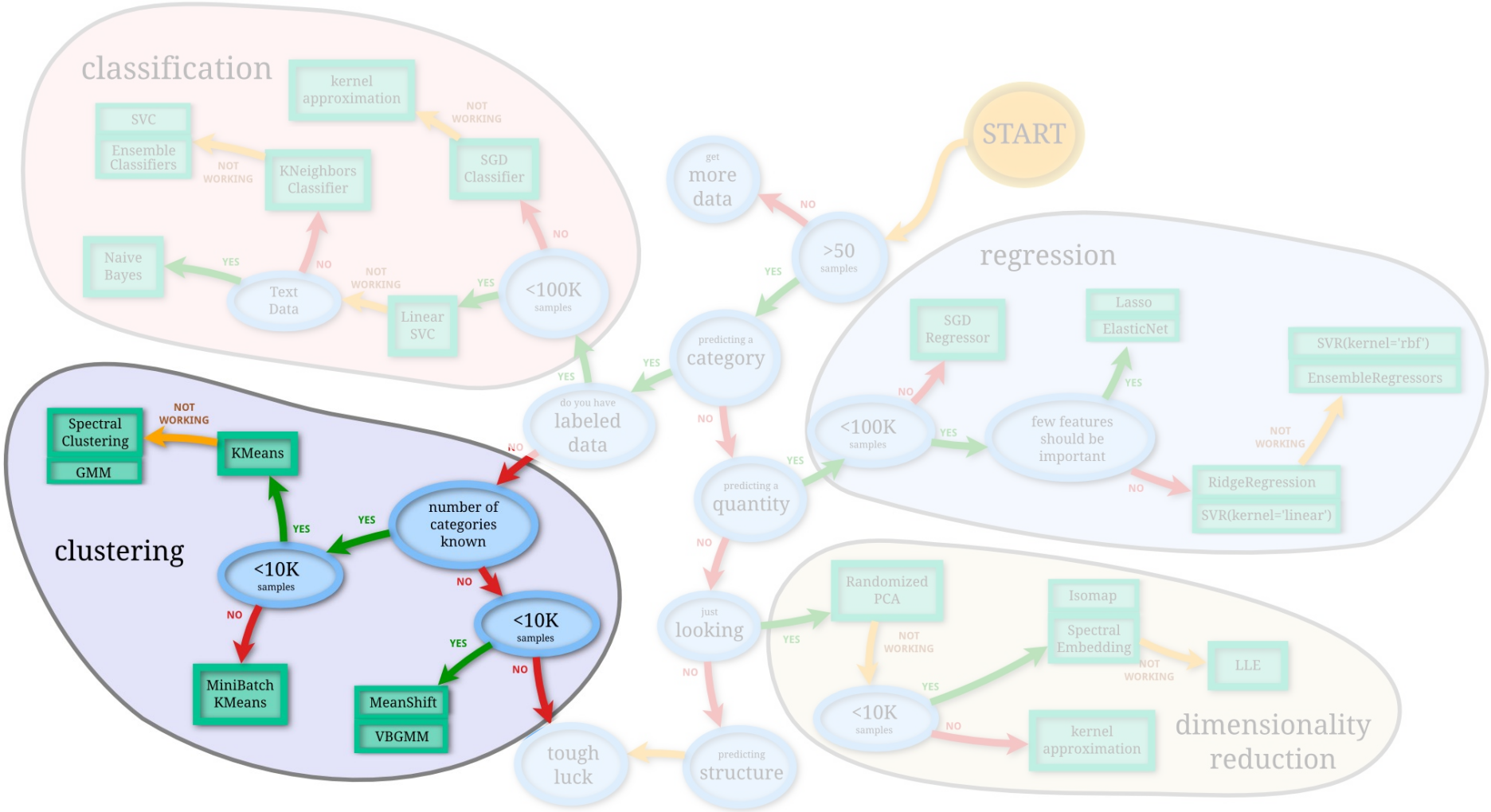
F. Nüske, […], A. S. J. S. Mey and F. Noé: *JCTC*, 2014

G. Perez-Hernandez, et al., ***JCTC***, 2013

C. R. Schwantes et al., ***JCTC***, 2013

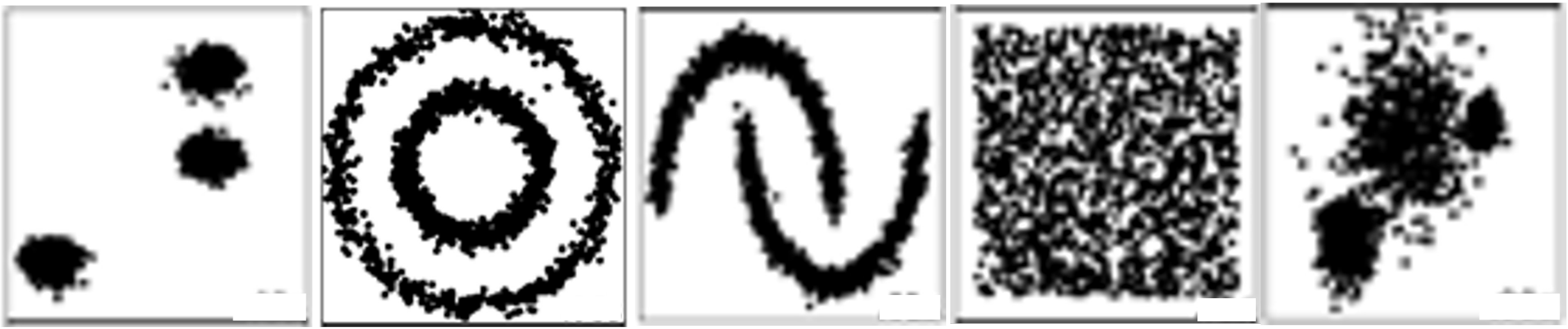

Break Time!

The Data Mining world



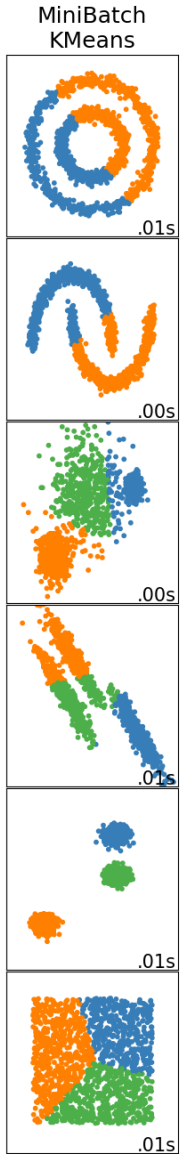
From **scikit-learn.org**

Clustering (i.e., unsupervised learning)



Known number of clusters? Flat geometry?
Even cluster size? Outliers? Centroids needed?

Clustering algorithms



scikit-learn.org/stable/modules/clustering.html

How does k-means work?

Input: K, set of points $x_1 \ldots x_n$ (can be in N-dimensional)

Place centroids, $c_1 \ldots, c_n$ at random locations

Repeat until convergence:

For each point x_i:

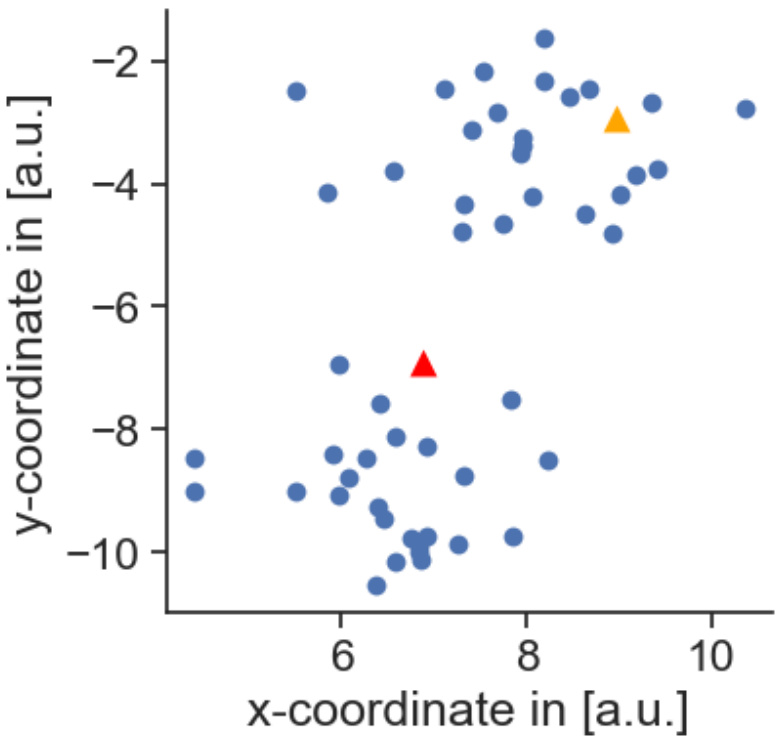
Find nearest centroid $c_j = \arg\min_j D(x_i, c_j)$

Assign the point x_i to cluster j

For each cluster j = 1…K:

Compute the centroid mean for all points in one cluster and update the centroid

$$c_j(a) = \frac{1}{n_j} \sum_{x_i \to c_j} x_i(a)$$



How does k-means work?

Input: K, set of points $x_1 \ldots x_n$ (can be in N-dimensional)

Place centroids, $c_1 \ldots, c_n$ at random locations

Repeat until convergence:

For each point x_i:

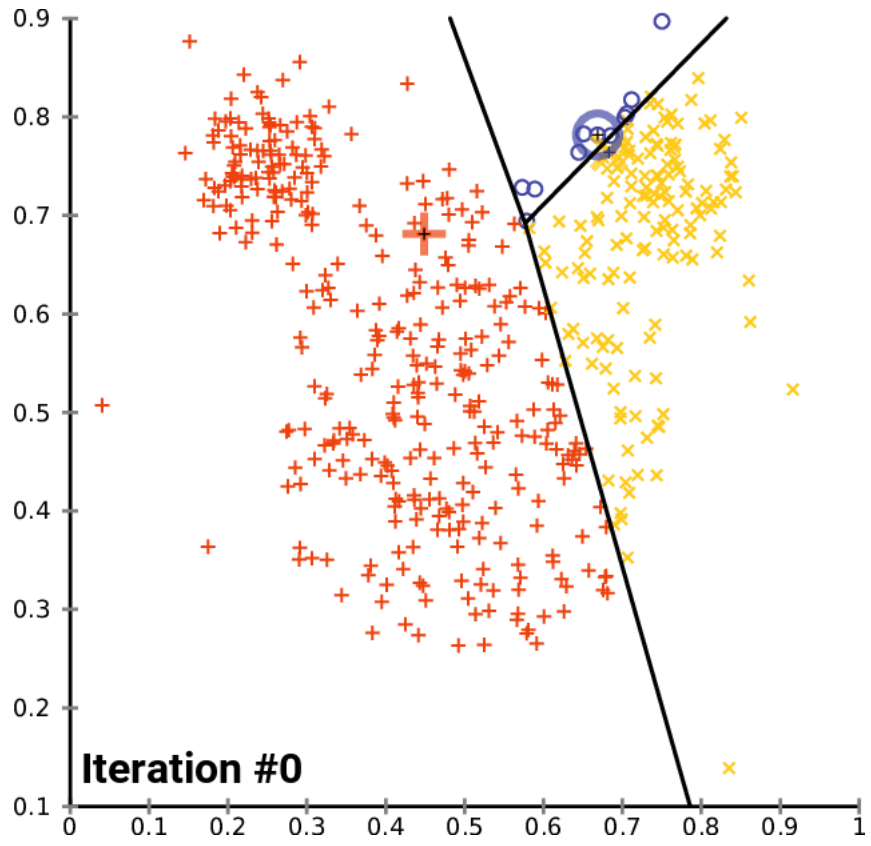
Find nearest centroid $c_j = \arg\min_j D(x_i, c_j)$

Assign the point x_i to cluster j

For each cluster j = 1…K:

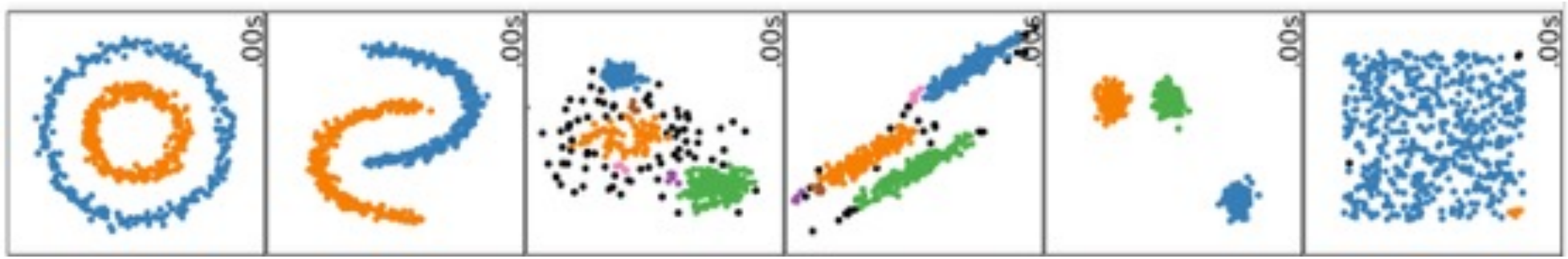
Compute the centroid mean for all points in one cluster and update the centroid

$$c_j(a) = \frac{1}{n_j} \sum_{x_i \to c_j} x_i(a)$$

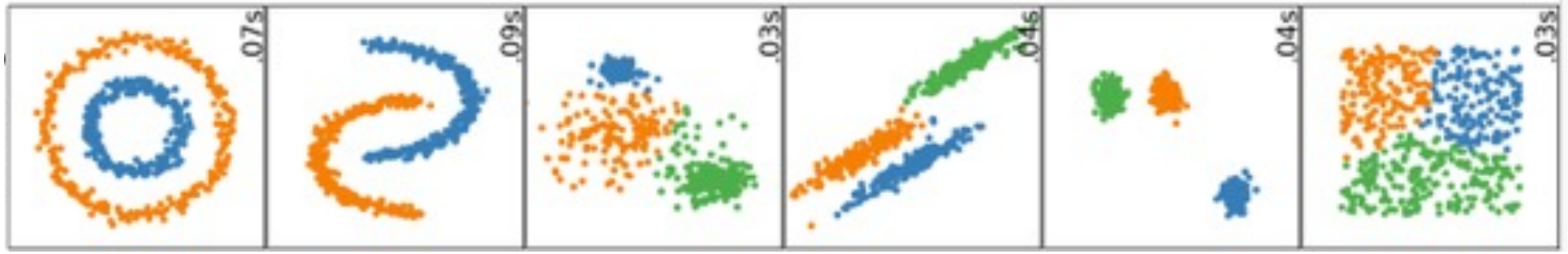


Density-based and spectral clustering

DBSCAN

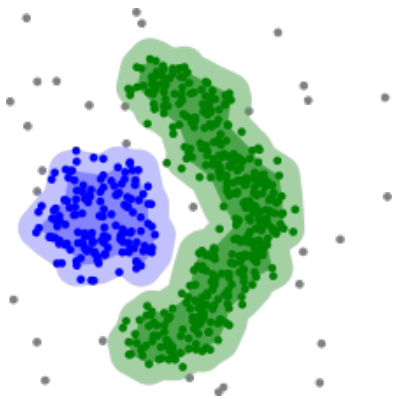


Spectral Clustering



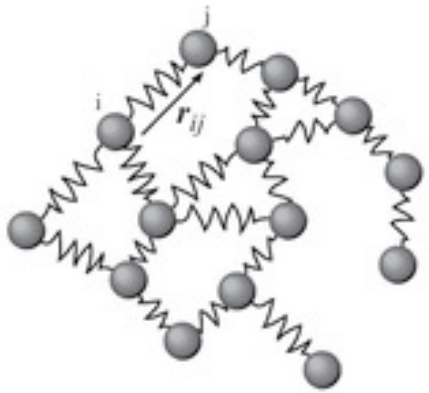
Density-based and spectral clustering

DBSCAN



- Find the points in the *ε* neighbourhood of every point, identify core points with more than *n* neighbours.
- Find the connected components of core points on the neighbour graph, ignoring all non-core points.
- Assign each non-core point to a nearby cluster if the cluster is an ε neighbour, otherwise assign to noise otherwise.

Spectral Clustering



- Calculate the Laplacian
- Calculate the first k eigenvectors
- Consider the matrix formed by the first k-eigenvectors
- Cluster the graph nodes based on these features (e.g. k-means)

Post-its