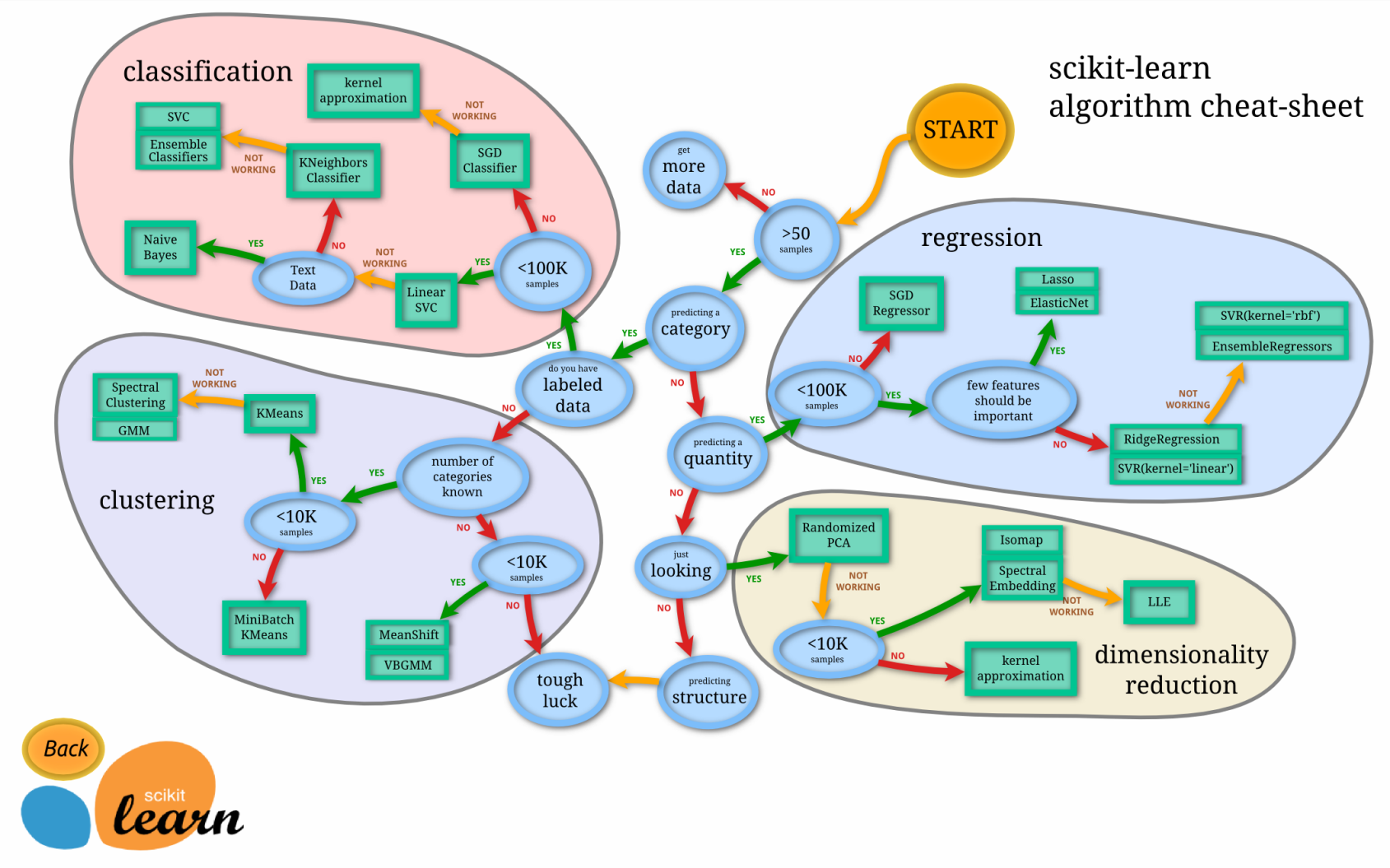
A concise overview of classification and clustering methods

Maxime Sangnier

GDR IAMAT - May 31, 2022

Sorbonne Université, CNRS, LPSM, Paris, France

A scikit-learn map



Source: `https://scikit-learn.org/`

Mathematical setting

Numerical formulation

Vectors and matrices

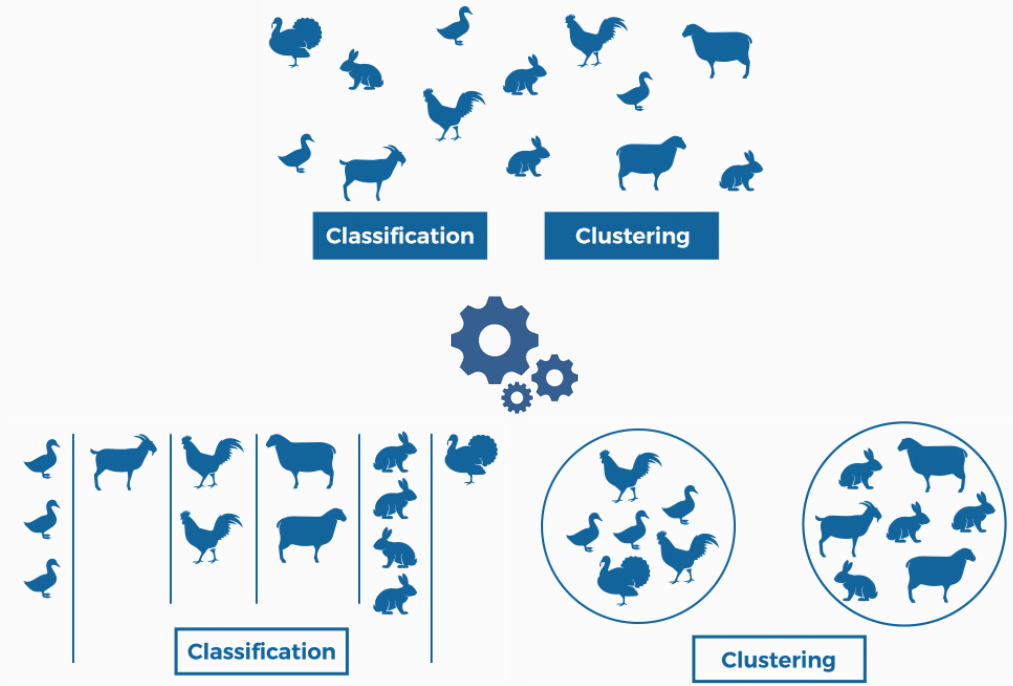
- Each experiment is defined by initial conditions $\rightarrow x$ and a result $\rightarrow y$.
- Each observation is defined by an individual $\rightarrow x$ and a feature $\rightarrow y$.
- For computational purposes, $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$.
- Many repetitions of the experiment/observation provide $(x_1, y_1), \ldots, (x_n, y_n)$.
- Data matrix $\mathbf{X} = \begin{pmatrix} x_1 \text{ in raw} \\ \vdots \\ x_n \text{ in raw} \end{pmatrix}$ (size $n \times d$) and vector of outputs $\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$.

Numerical formulation

Goal

1. **Classification/Regression** Given x_{new}, "predict" *i.e.* approximate y_{new}: $y_{new} \approx f(x_{new})$.
2. **Clustering** When y_i's are *not* observed, gather x_i's that are *similar*: $\{x_1, x_3, x_4, \ldots\} - \{x_2, x_7, x_{11}, \ldots\} - \{x_5, x_{13}, x_{22}, \ldots\}$.

Classification and clustering are roughly similar except that no information concerning the group is available for clustering.



Source: `https://blog.bismart.com`

Mathematical formulation

Randomness

- (x_i, y_i) is a realization of a random pair (X_i, Y_i).
- Why randomness? Because y_i cannot be computed only based on x_i:
  - y_i is noisy and does not reflect actually our desire.
  - x_i is incomplete and a perfect description is out of reach.

Statistics

- All (X_i, Y_i)'s are independent and have the same distribution.
- We are interested in the expected behavior: overall true now and in the future.
  1. **Classification/Regression**

$$\mathbb{P}\left(f(X_{new}) \text{ is close to } Y_{new}\right) = \mathbb{E}\left[\text{similarity}\left(f(X_{new}), Y_{new}\right)\right].$$

  2. **Clustering**

$$\begin{aligned}&\mathbb{P}\left(X_{new} \text{ is similar to } X_{other} \text{ when } X_{new} \text{ and } X_{other} \text{ fall in the same group}\right)\\ &= \mathbb{E}\left[\text{similarity}\left(X_{new}, X_{other}\right) \text{ when } X_{new} \text{ and } X_{other} \text{ fall in the same group}\right].\end{aligned}$$

- The distribution of (X_i, Y_i)'s is unknown but...
- By the law of large numbers,
  $\frac{1}{n}\sum_{i=1}^{n} \text{similarity}\left(f(X_i), Y_i\right) \xrightarrow[n\to\infty]{a.s.} \mathbb{E}\left[\text{similarity}\left(f(X_{new}), Y_{new}\right)\right]$.

Supervised learning

Supervised learning

Two methodologies

- We observe inputs and outputs: $(X_1, Y_1), \ldots, (X_n, Y_n)$.
- Let ℓ be a loss (*i.e.* a dissimilarity) function.
- **Goal** Find f such that $\mathbb{E}\left[\ell\left(f(X_{new}), Y_{new}\right)\right]$ is minimal.
  - **Classification** $\ell\left(f(X_{new}), Y_{new}\right) = \begin{cases} 1 & \text{if } f(X_{new}) \neq Y_{new} \\ 0 & \text{if } f(X_{new}) = Y_{new} \end{cases}$.
  - **(L^2) Regression** $\ell\left(f(X_{new}), Y_{new}\right) = \left(Y_{new} - f(X_{new})\right)^2$.

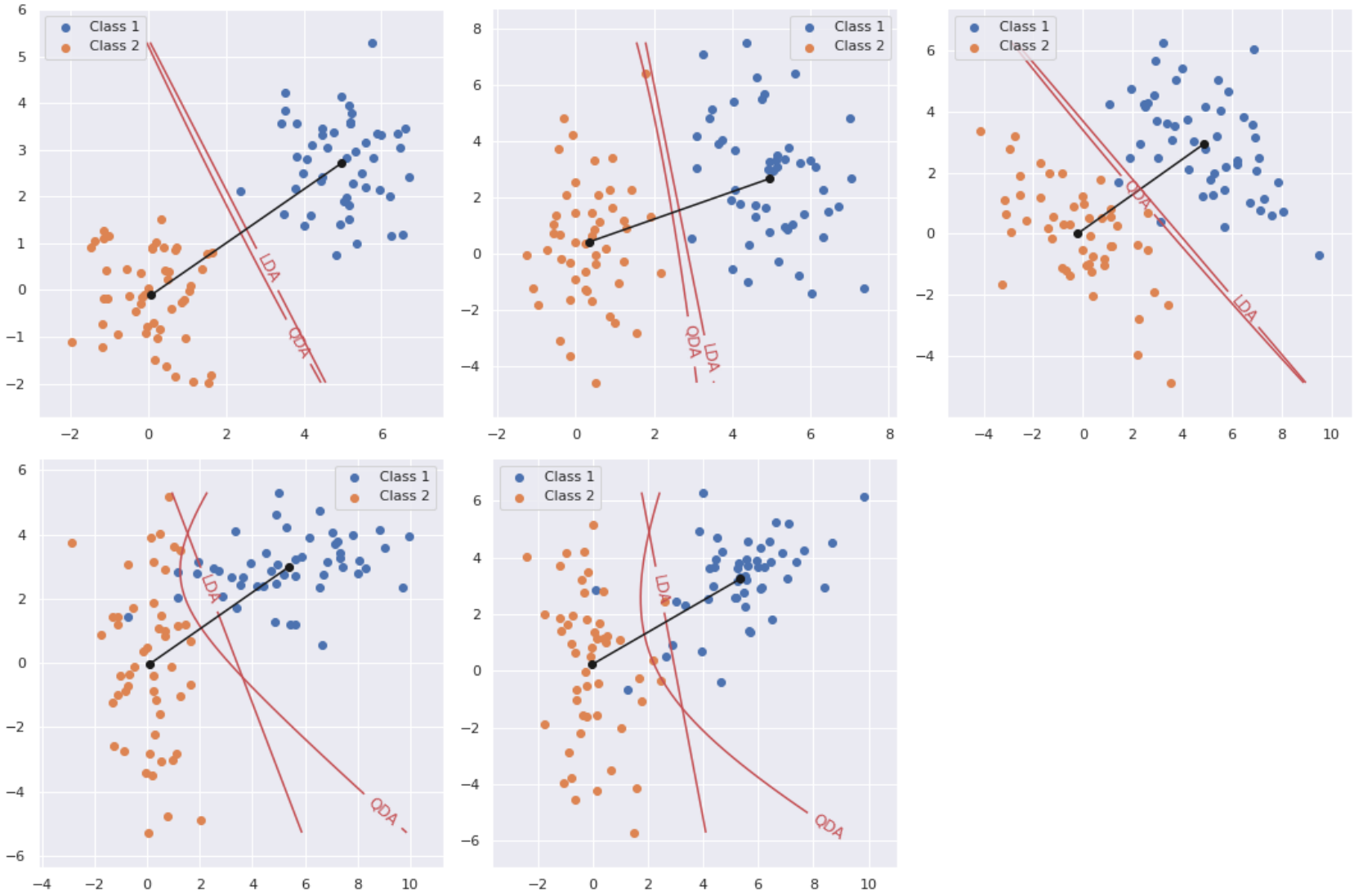
Plug-in estimator

- Bayes minimizer: $f^\star(x) = L\left(\mathbb{E}\left[Y_{new} \,\middle|\, X_{new} = x\right]\right)$, where L is known.
- Estimator: $\hat{f}(x) = L\left(\hat{\mathbb{E}}\left[Y_{new} \,\middle|\, X_{new} = x\right]\right)$.
- Needs statistics.
- Often:
  - We have to model the data distribution.
  - We resort to simple and non-robust estimators.
  - We are quite limited (the Bayes estimator is not always explicit).
- **Examples** Parametric models (LDA, QDA, Logistic Regression, Linear Regression), kernel methods (k-Nearest Neighbors, Trees, Random Forests).

Empirical Risk Minimization

- By the law of large numbers: $\frac{1}{n}\sum_{i=1}^{n}\ell\left(f(X_i), Y_i\right) \approx \mathbb{E}\left[\ell\left(f(X_{new}), Y_{new}\right)\right]$.
- Estimator: $\hat{f}$ such that $\frac{1}{n}\sum_{i=1}^{n}\ell\left(\hat{f}(X_i), Y_i\right)$ is minimal.
- Needs optimization tools.
- Sometimes:
  - We have to design efficient optimization algorithms (convex optimization).
  - We have to settle for a good algorithm applied in an inadequate setting (non-convex optimization).
- **Examples** Boosting, Support Vector Machines, Neural Networks.

Discriminant analysis



The method

- For binary or multiclass classification only.
- Assumption on data:

$$\begin{cases} X_{new} \mid Y_{new} = \bullet \sim \mathcal{N}(\mu, \Sigma), & \mathbb{P}(Y_{new} = \bullet) = \pi, \\ X_{new} \mid Y_{new} = \bullet \sim \mathcal{N}(\mu, \Sigma), & \mathbb{P}(Y_{new} = \bullet) = \pi, \end{cases}$$

  for unknown $\mu, \mu, \Sigma, \Sigma, \pi, \pi$.

- The Bayes classifier draws a parabolic frontier:

$$f^\star(x) = \begin{cases} \bullet & \text{if } \frac{1}{2} x^\top \left(\Sigma^{-1} - \Sigma^{-1}\right) x + \left(\Sigma^{-1}\mu - \Sigma^{-1}\mu\right)^\top x + \ldots \\ & + \log\left(\frac{\pi}{\pi}\right) \geq 0 \\ \bullet & \text{otherwise.} \end{cases}.$$

- Plug-in estimation via Maximum Likelihood: closed-form expressions for $\hat{\mu}, \hat{\mu}, \hat{\Sigma}, \hat{\Sigma}, \hat{\pi}, \hat{\pi}$.

Discriminant analysis

What if $\Sigma = \Sigma$?

- The Bayes classifier draws a hyperplane frontier:

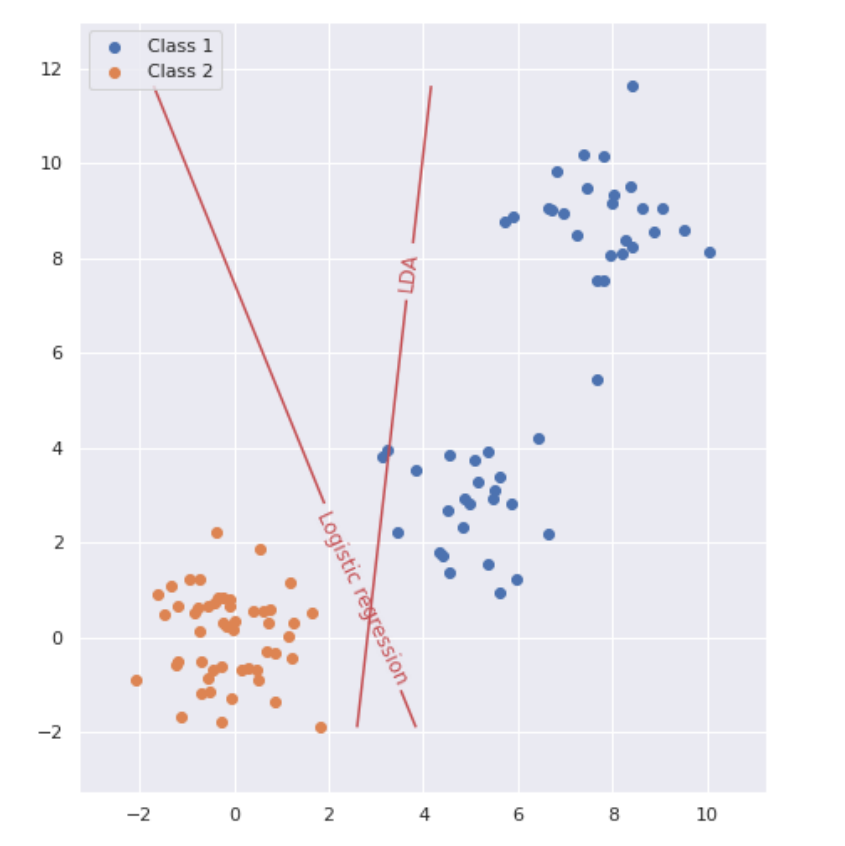
$$f^\star(x) = \begin{cases} \bullet & \text{if } (\mu - \mu)^\top \Sigma^{-1} x + \cdots + \log\left(\frac{\pi}{\pi}\right) \geq 0 \\ \bullet & \text{otherwise.} \end{cases}$$

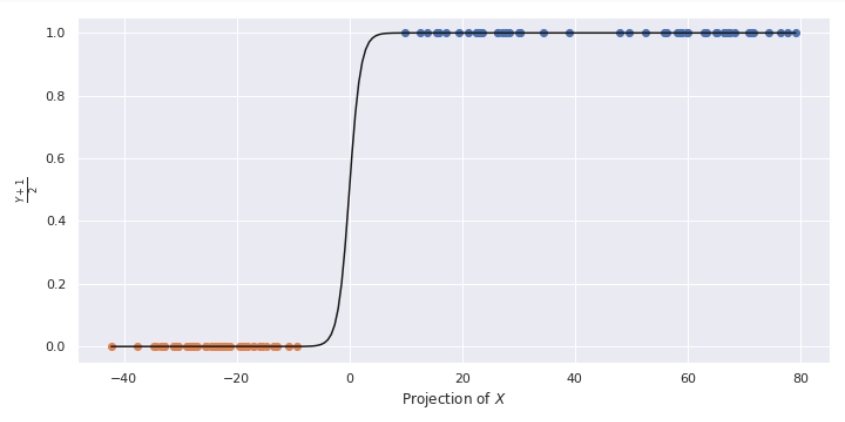
- $\frac{\pi}{\pi}$ translates the hyperplane frontier.
- If $\pi = \pi$, the Bayes classifier is a minimum-Mahalanobis-distance-to-center classifier.

Pros and cons

- It is computationally tractable (OK in high dimension and with large amount of data).
- It is very restrictive (categorical data...).
- It is not robust.
- It is not very expressive.

Logistic Regression





Logistic Regression

The method

- For binary or multiclass classification only.
- Assumption on data:

$$\log\left(\frac{\mathbb{P}\left(Y_{new} = \bullet \mid X_{new}\right)}{\mathbb{P}\left(Y_{new} = \bullet \mid X_{new}\right)}\right) = w^\top X_{new} + b,$$

  for unknown w, b.
- The Bayes classifier draws a hyperplane frontier:

$$f^\star(x) = \begin{cases} \bullet & \text{if } w^\top x + b \geq 0 \\ \bullet & \text{if } w^\top x + b < 0 \end{cases}.$$

- Plug-in estimation via Regularized Maximum Likelihood: $(\hat{w}, \hat{b})$ such that

$$\frac{1}{n}\sum_{i=1}^{n} \log\left(1 + \mathrm{e}^{-Y_i(\hat{w}^\top X_i + \hat{b})}\right) + \frac{\lambda}{2}\|\hat{w}\|^2$$

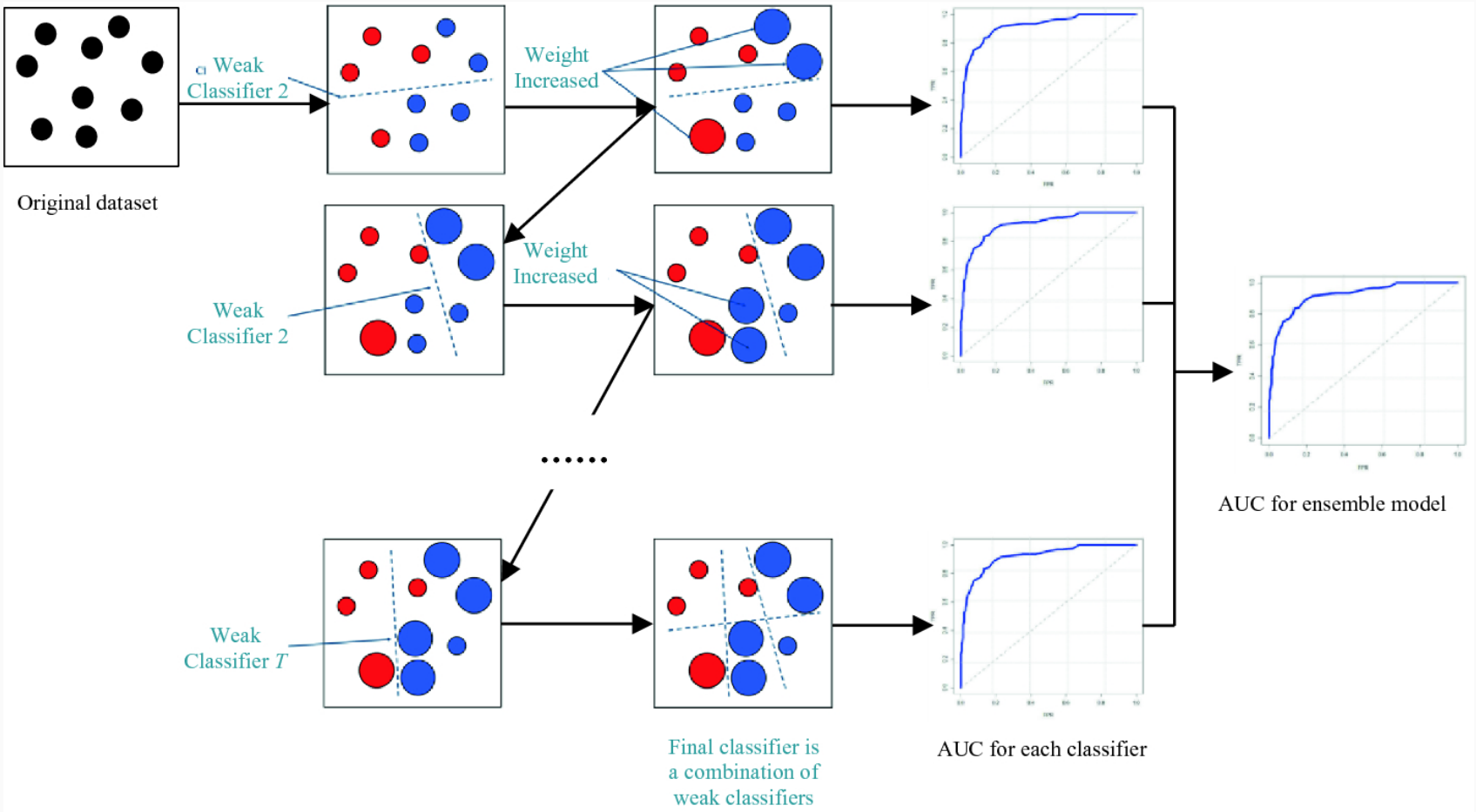
  is minimal.
- Parameter: regularization coefficient $\lambda > 0$.

Logistic Regression

Pros and cons

- It is robust.
- It is computationally tractable (OK in high dimension and with large amount of data).
- It is not very expressive.

Gradient boosting



Source: `https://datascience.eu`

Gradient boosting

The method

- For binary classification and regression.
- No assumption on data.
- Combination of simple estimators (weak learners):

$$\hat{f}(x) = \hat{f}_1(x) + \cdots + \hat{f}_T(x) \qquad \text{such that} \qquad \frac{1}{n}\sum_{i=1}^{n} \ell\left(\hat{f}(X_i), Y_i\right)$$

  is almost minimal.

- Iterative procedure: $\hat{f}_{t+1}$ is such that

$$\frac{1}{n}\sum_{i=1}^{n} \ell\left(\hat{f}_1(X_i) + \cdots + \hat{f}_t(X_i) + \hat{f}_{t+1}(X_i), Y_i\right)$$

  is almost minimal.

- Similar to gradient descent:

$$\hat{f}_{t+1}(X_i) \approx -\frac{\eta}{\text{normalization}} \frac{\partial \ell}{\partial x}(\hat{f}_1(X_i) + \cdots + \hat{f}_t(X_i), Y_i).$$

Gradient boosting

The method

- The case of L^2 regression: $\hat{f}_{t+1}(X_i) \approx Y_i - (\hat{f}_1(X_i) + \cdots + \hat{f}_t(X_i))$.
- Parameters: number of simple estimators T, shrinkage coefficient $\eta \in]0, 1]$.

Pros and cons

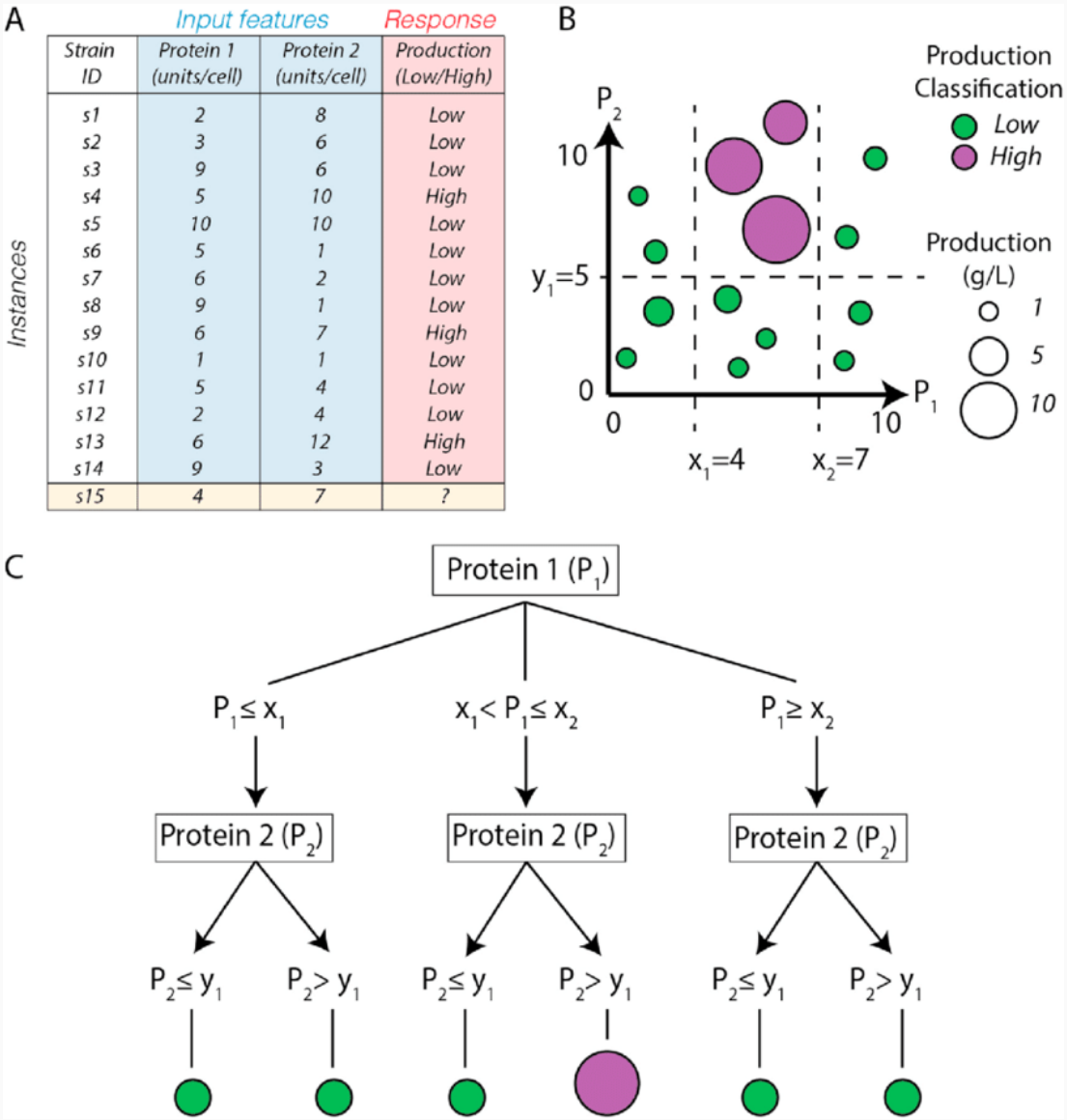
- It is robust and efficient.
- It is fairly computationally tractable.
- It is very expressive.
- In practice, it has more than two parameters.

Decision tree

A

	Input features		Response
Strain ID	Protein 1 (units/cell)	Protein 2 (units/cell)	Production (Low/High)
s1	2	8	Low
s2	3	6	Low
s3	9	6	Low
s4	5	10	High
s5	10	10	Low
s6	5	1	Low
s7	6	2	Low
s8	9	1	Low
s9	6	7	High
s10	1	1	Low
s11	5	4	Low
s12	2	4	Low
s13	6	12	High
s14	9	3	Low
s15	4	7	?

Instances



Source: `https://www.researchgate.net`

Decision tree

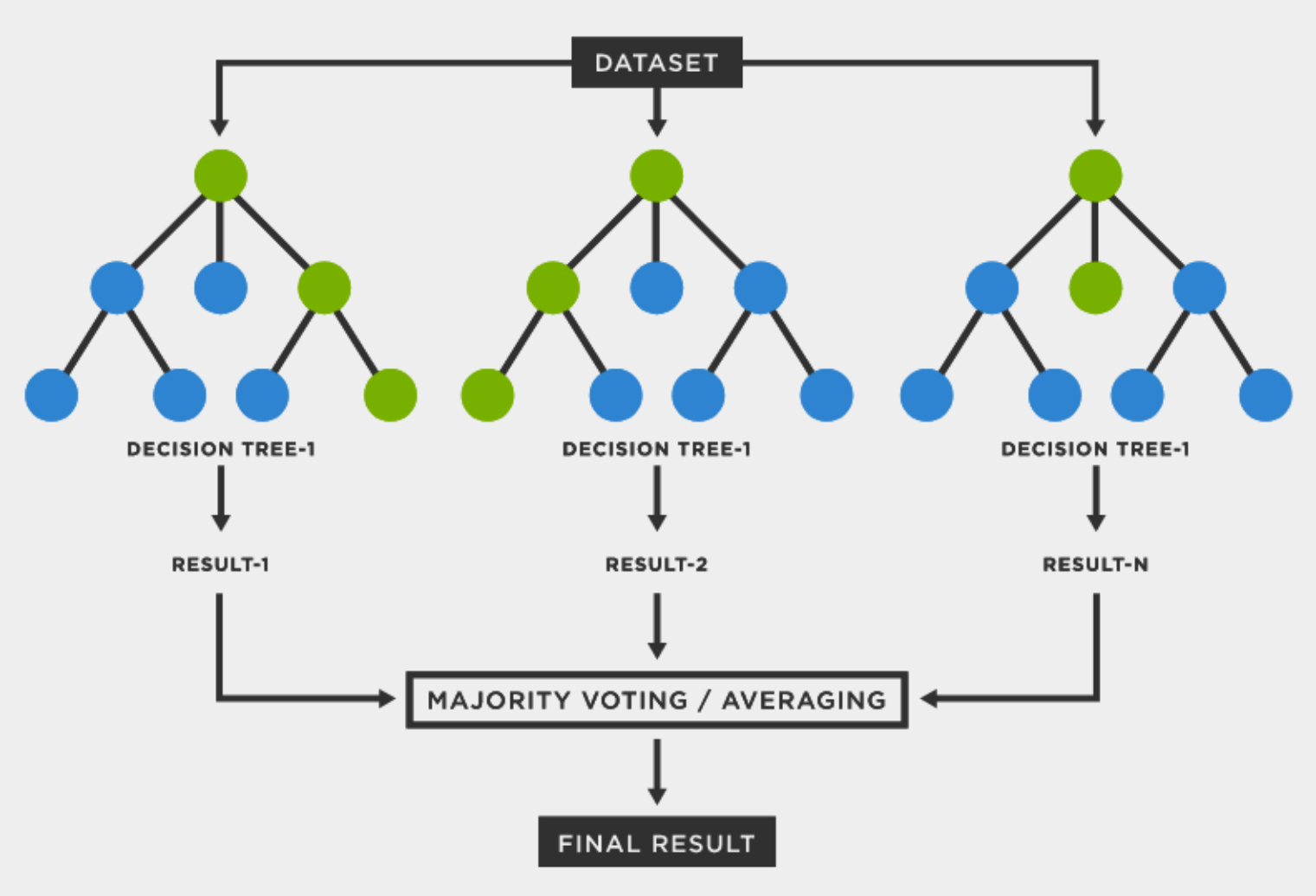
The method

- For multiclass classification and regression.
- No assumption on data.
- Data dependent partitioning of the input space: $x \in \text{Cell}_1$ or $\text{Cell}_2 \ldots$.
- Partition implemented as a binary tree.
- Piece-wise constant estimator: $\hat{f}(x) = \text{Value}_k$ for $x \in \text{Cell}_k$.
- Cells and values are determined in order to maximize the output *homogeneity*.
- Parameters: size of the tree (*i.e.* number of cells).

Pros and cons

- It is computationally tractable.
- It is very expressive.
- It is prone to overfitting.
- It is used in gradient boosting with few cells.

Random forest



Source: `https://www.tibco.com`

Random forest

The method

- For multiclass classification and regression.
- No assumption on data.
- Combination of decision trees:

$$\hat{f}(x) = \frac{\text{tree}_1(x) + \cdots + \text{tree}_T(x)}{T},$$

  such that trees are roughly independent.
  - Trees are learned on bootstrap samples.
  - Construction of partitions is perturbed by noise.
- Parameters: number of trees T, size of trees, level of noise in partitions building.

Random forest

Pros and cons

- It is robust and efficient.
- It is fairly computationally tractable.
- It is very expressive.
- Construction is parallelizable.
- In practice, it has many parameters.

To go further

Feature importance and selection

- It is mainly model-dependent.
- In-model importance and post-selection:
  - Feature weights for linear models.
  - Contribution to the score for decision trees.
- Iterative procedure: score improvement when adding a feature.
- In-model importance and in-selection: sparse regularization for linear models.

Multiclass problems

- Natively handled by some methods: k-Nearest neighbors, trees, random forests, neural networks.
- Others are mainly for binary classification: linear models, boosting, Support Vector Machines.
- Four strategies:
  - Binary encoding of the class number + $\lceil \log_2(\#\text{ classes}) \rceil$ classifiers.
  - One versus One + $\binom{\#\text{ classes}}{2}$ classifiers.
  - One versus Rest + $\#$ classes classifiers.
  - Hierarchical One versus Rest + $\#$ classes $-$ 1 classifiers.

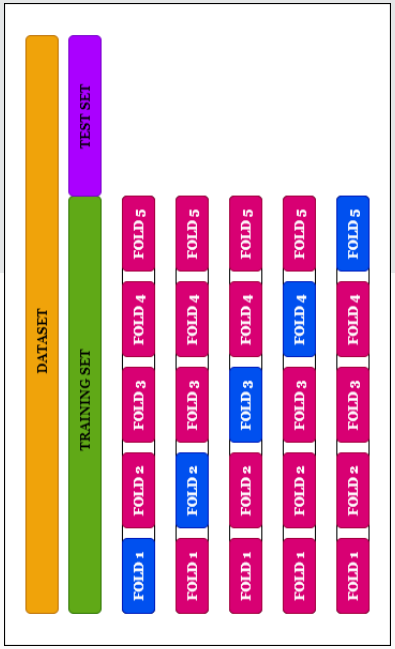
Evaluation and selection

Metrics

- It depends on what is important for you but is not necessarilly reflected in the loss ℓ.
- Accuracy, balanced accuracy, top-k accuracy.
- Area under the ROC curve, F1 score.
- Mean squared error, R^2 score.

Generalization and model selection

- Cross-validation.
- Grid or random search.
- Regularization path.



Source: `https://medium.com`

Clustering

Three methodologies

- We observe only inputs: $X_1, \ldots, X_n$.
- **Goal** Find a partition of the space $\{A, A\}$ such that

$$\begin{cases} X_{new} \in A, X_{other} \in A \iff X_{new} \text{ and } X_{other} \text{ are similar;} \\ X_{new} \in A, X_{other} \in A \iff X_{new} \text{ and } X_{other} \text{ are similar.} \end{cases}$$

The latent variable model

- Partial observation in the classification setting: $(X_1, Y_1), \ldots, (X_n, Y_n)$, $Y_1 = \bullet$ or $\bullet$.
- Expected clustering ($\approx$ Bayes classifier):

$$A = \left\{x : \mathbb{P}\left(Y_{new} = \bullet \,\middle|\, X_{new} = x\right) \geq \mathbb{P}\left(Y_{new} = \bullet \,\middle|\, X_{new} = x\right)\right\},$$

  A is the rest.

- Estimator:

$$\hat{A} = \left\{x : \hat{\mathbb{P}}\left(Y_{new} = \bullet \,\middle|\, X_{new} = x\right) \geq \hat{\mathbb{P}}\left(Y_{new} = \bullet \,\middle|\, X_{new} = x\right)\right\}.$$

The latent variable model

- Needs statistics.
- In practice:
  - We have to model the data distribution.
  - We resort to simple and non-robust estimators.
  - We are very limited.
- **Examples** Soft k-Means.

Empirical Risk Minimization

- For a dissimilarity function ℓ, find $\{A, A\}$ such that

$$\mathbb{E}\left[\ell(X_{new}, X_{other})\mathbb{1}_{X_{new}\in A, X_{other}\in A \text{ or } X_{new}\in A, X_{other}\in A}\right]$$

  is minimal.

- By the law of large numbers:

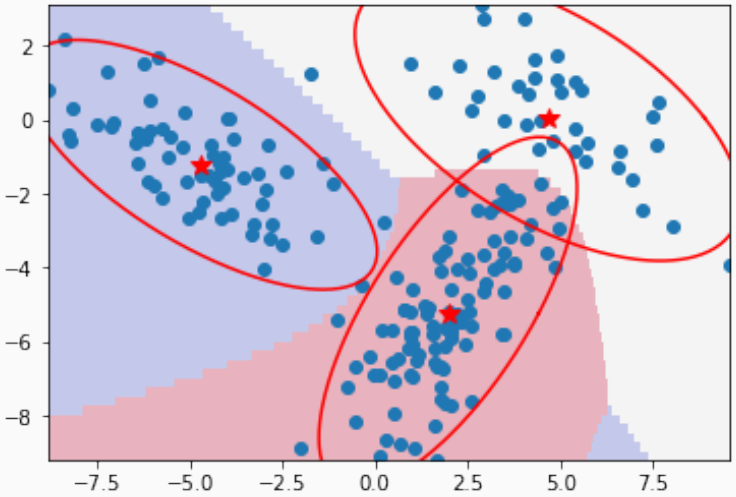
$$\frac{1}{2}\sum_{1\leq i,j\leq n}\ell(X_i, X_j)\left[\frac{\mathbb{1}_{X_i\in A, X_j\in A}}{\#X_\ell \in A} + \frac{\mathbb{1}_{X_i\in A, X_j\in A}}{\#X_\ell \in A}\right] \approx \mathbb{E}\left[\ell(X_{new}, X_{other})\mathbb{1}_{\dots}\right].$$

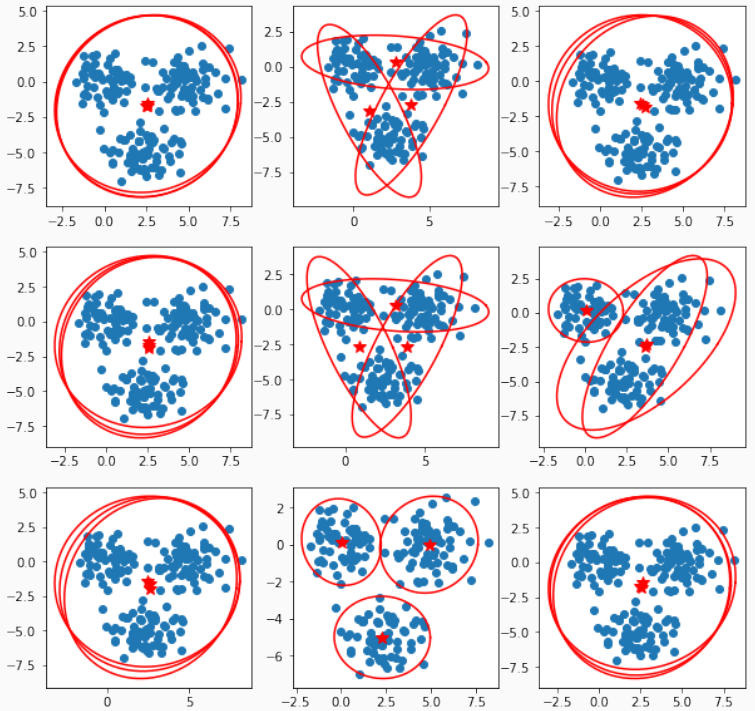
- Estimator: $\{\hat{A}, \hat{A}\}$ such that $\frac{1}{n}\sum_{i=1}^{n}\ell(X_{new}, X_{other})\mathbb{1}_{\dots}$ is minimal.
- Needs optimization tricks.
- In practice:
  - We have to change the optimization problem because it is not tractable (NP-hard).
  - We have to design a simple and non-optimal iterative procedure because the optimization problem it is not tractable (NP-hard).
- **Examples** k-means, spectral clustering, hierarchical clustering.

Density-based approaches

- Implicitly estimate the density of X_{new}.
- Detect the modes and set automatically the number of groups.
- Needs intuition and geometry.
- In practice:
  - We have only an intuitive understanding.
  - We have to set a *density* parameter (neighborhood size, bandwith).
- **Examples** DBSCAN, OPTICS, Mean shift.

Soft k-means





Soft k-means

The method

- Partial observation in the classification setting: $(X_1, Y_1), \ldots, (X_n, Y_n)$.
- Assumption on data:

$$\begin{cases} X_{new} \mid Y_{new} = \bullet \sim \mathcal{N}(\mu, \Sigma), & \mathbb{P}(Y_{new} = \bullet) = \pi, \\ X_{new} \mid Y_{new} = \bullet \sim \mathcal{N}(\mu, \Sigma), & \mathbb{P}(Y_{new} = \bullet) = \pi, \end{cases}$$

  for unknown $\mu, \mu, \Sigma, \Sigma, \pi, \pi$.

- A posteriori distribution:

$$\mathbb{P}(Y_{new} = \bullet | X_{new} = x) = \frac{\pi\varphi(x)}{\pi\varphi(x) + \pi\varphi(x)} = p(x).$$

- Plug-in estimation via Maximum Likelihood: $\hat{\pi}, \hat{\pi}, \hat{\varphi}, \hat{\varphi}$ such that

$$\sum_{i=1}^{n} \log\left(\hat{\pi}\hat{\varphi}(X_i) + \hat{\pi}\hat{\varphi}(X_i)\right)$$

  is maximal.

Soft k-means

The method

- Proxy maximization:

$$\mathbb{E}\left[\sum_{i=1}^{n} \log\left(\text{density}_{(X_1, Z_{t,1})}(X_i, Z_{t,i})\right) \mid X_1, \ldots, X_n\right]$$

$$= \sum_{i=1}^{n} \left[\hat{p}_t(X_i) \log\left(\hat{\pi}_{t+1}\hat{\varphi}_{t+1}(X_i)\right) + (1 - \hat{p}_t(X_i)) \log\left(\hat{\pi}_{t+1}\hat{\varphi}_{t+1}(X_i)\right)\right],$$

  where $Z_{t,i} \mid X_i = x$ has the distribution of $Y_{new} \mid X_{new} = x$ estimated with $\hat{\pi}_t, \hat{\pi}_t, \hat{\varphi}_t, \hat{\varphi}_t$.

- Estimator: $\hat{A} = \left\{x : \hat{p}_t(x) \geq \frac{1}{2}\right\}$.

Soft k-means

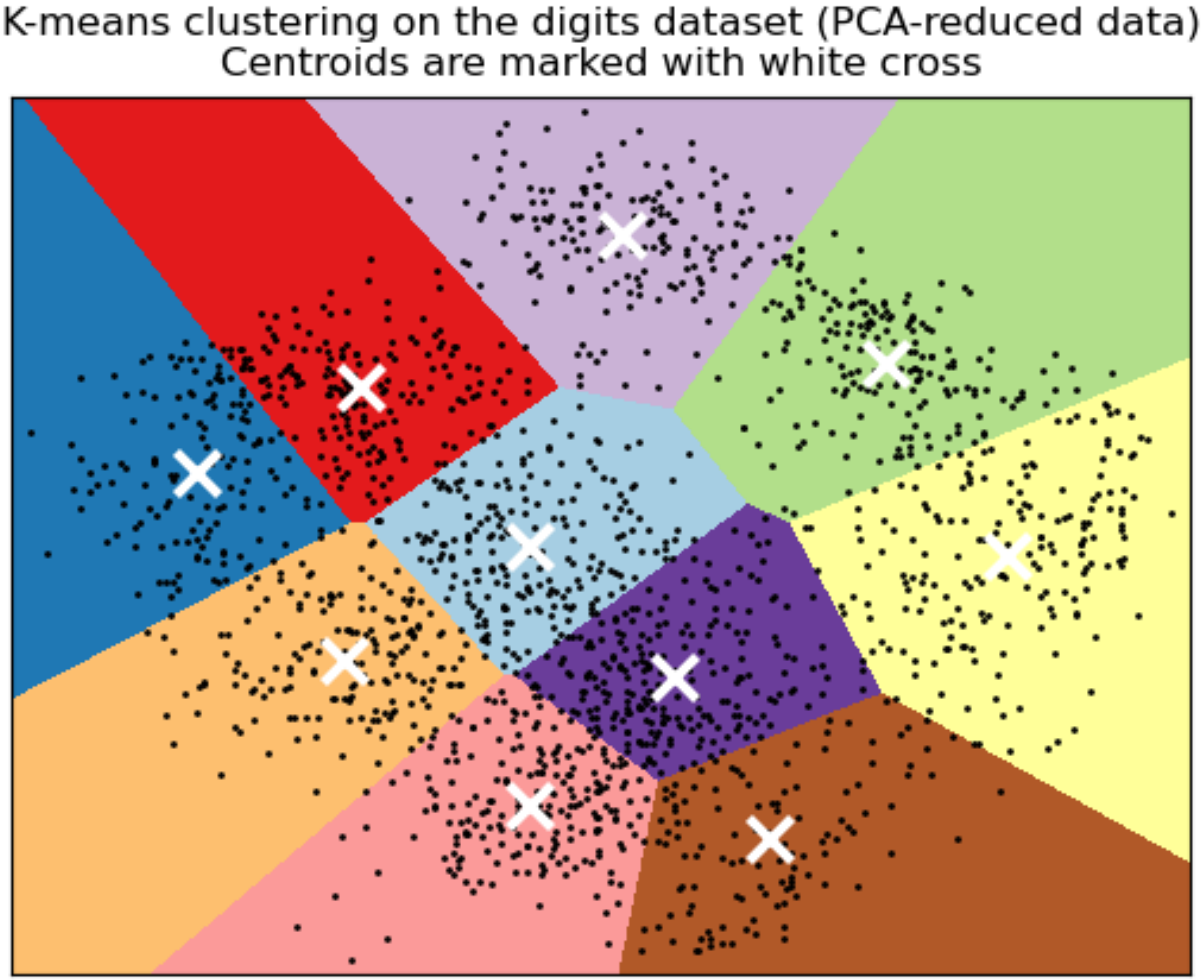
Iterative algorithm

1. Compute $\hat{p}_t(X_1), \dots, \hat{p}_t(X_n)$ with $\hat{\pi}_t, \hat{\pi}_t, \hat{\varphi}_t, \hat{\varphi}_t$.
2. $\hat{\pi}_{t+1} = \frac{1}{n} \sum_{i=1}^{n} \hat{p}_t(X_i)$.
3. $\hat{\mu}_{t+1} =$ empirical mean weighted by $\hat{p}_t(X_1), \dots, \hat{p}_t(X_n)$.
4. $\hat{\Sigma}_{t+1} =$ empirical covariance centered at $\hat{\mu}_{t+1}$ and weighted by $\hat{p}_t(X_1), \dots, \hat{p}_t(X_n)$.
5. Same for $\hat{\pi}_{t+1}, \hat{\mu}_{t+1}, \hat{\Sigma}_{t+1}$ with $(1 - \hat{p}_t(X_1)), \dots, (1 - \hat{p}_t(X_n))$.

Pros and cons

- It is a very simple and cheap iterative algorithm.
- It is suboptimal (does not necessarily return the MLEs).
- It is very sensitive to initialization $\hat{\mu}_0$ and $\hat{\mu}_0$ (in practice, it is initialized with k-means++ output).

k-means



Source: `https://scikit-learn.org`

The method

- No assumption on data.
- ERM with dissimilarity $\ell(x, x') = \|x - x'\|_2^2$: minimize

$$\frac{1}{2} \sum_{1 \leq i,j \leq n} \|X_i - X_j\|_2^2 \left[\frac{\mathbb{1}_{X_i \in \hat{A}, X_j \in \hat{A}}}{\#X_\ell \in \hat{A}} + \frac{\mathbb{1}_{X_i \in \hat{A}, \hat{X}_j \in \hat{A}}}{\#X_\ell \in \hat{A}} \right]$$

$$= \sum_{i=1}^{n} \left[\|X_i - \hat{\mu}\|_2^2 \mathbb{1}_{X_i \in \hat{A}} + \|X_i - \hat{\mu}\|_2^2 \mathbb{1}_{X_i \in \hat{A}} \right],$$

  with $\hat{\mu} = \frac{1}{\#X_\ell \in \hat{A}} \sum_{i=1}^{n} X_i \mathbb{1}_{X_i \in \hat{A}}$.
- Alternating procedure (Lloyd's algorithm):
  1. Find $\{\hat{A}, \hat{A}\}$ with $\hat{\mu}$ and $\hat{\mu}$ fixed: Voronoi partitioning.
  2. Compute $\hat{\mu}$ and $\hat{\mu}$ with fixed partition $\{\hat{A}, \hat{A}\}$.
- Estimator: $\hat{A} = \{x : \|x - \hat{\mu}\| \leq \|x - \hat{\mu}\|\}$.

k-means

Iterative algorithm

1. $\hat{p}_t(X_i) = \mathbb{1}_{\|X_i - \hat{\mu}_t\| \leq \|X_i - \hat{\mu}_t\|}$.
2. $\hat{\mu}_{t+1} =$ empirical mean weighted by $\hat{p}_t(X_1), \ldots, \hat{p}_t(X_n)$.
3. Same for $\hat{\mu}_{t+1}$ with $(1 - \hat{p}_t(X_1)), \ldots, (1 - \hat{p}_t(X_n))$.

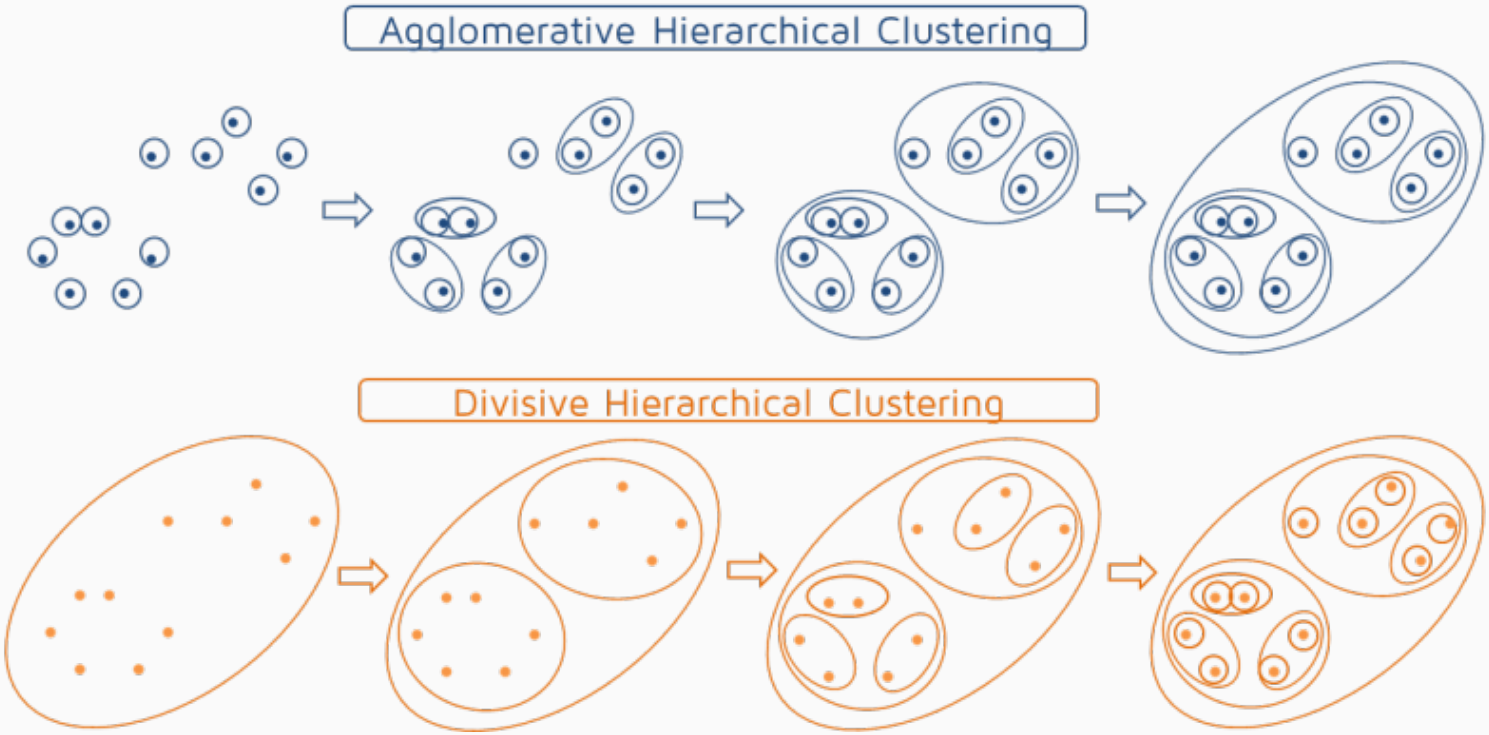
Connection with soft k-means

- Hard assignment: $\hat{p}_t(x) = \mathbb{1}_{\ldots}$ instead of $\hat{\mathbb{P}}(Y_{new} = \bullet \mid X_{new} = x)$.
- No a priori in partitioning (or $\hat{\pi}_{t+1} = \frac{1}{2}$).
- No variance (or $\hat{\Sigma}_{t+1} \to 0$).

Pros and cons

- It is a simple and cheap iterative algorithm.
- It is suboptimal (does not necessarily return the optimal partition).
- It is very sensitive to initialization $\hat{\mu}_0$ and $\hat{\mu}_0$: k-means++.
- Groups are convex, not hierarchically structured.

Agglomerative clustering



Source: `https://quantdare.com`

Agglomerative clustering

The method

- No assumption on data.
- Iterative procedure:
  1. Start with n groups: a point = a group.
  2. Iteratively merge *nearest* groups.
- Famous *distances* between groups A and A:
  - Single linkage:

$$d(A, A) = \min_{x \in A, x \in A} \|x - x\|.$$

  - Ward's criterion:

$$d(A, A) = \text{Inertia}(A \cup A) - \text{Inertia}(A) - \text{Inertia}(A) = \frac{\text{Size}(A)\text{Size}(A)}{\text{Size}(A) + \text{Size}(A)} \|\mu - \mu\|^2.$$

Connection with k-means

- With Ward's criterion: tend to minimize the total inertia

$$\sum_{i=1}^{n} \left[\|X_i - \hat{\mu}\|_2^2 \mathbb{1}_{X_i \in \hat{A}} + \|X_i - \hat{\mu}\|_2^2 \mathbb{1}_{X_i \in \hat{A}} \right]$$

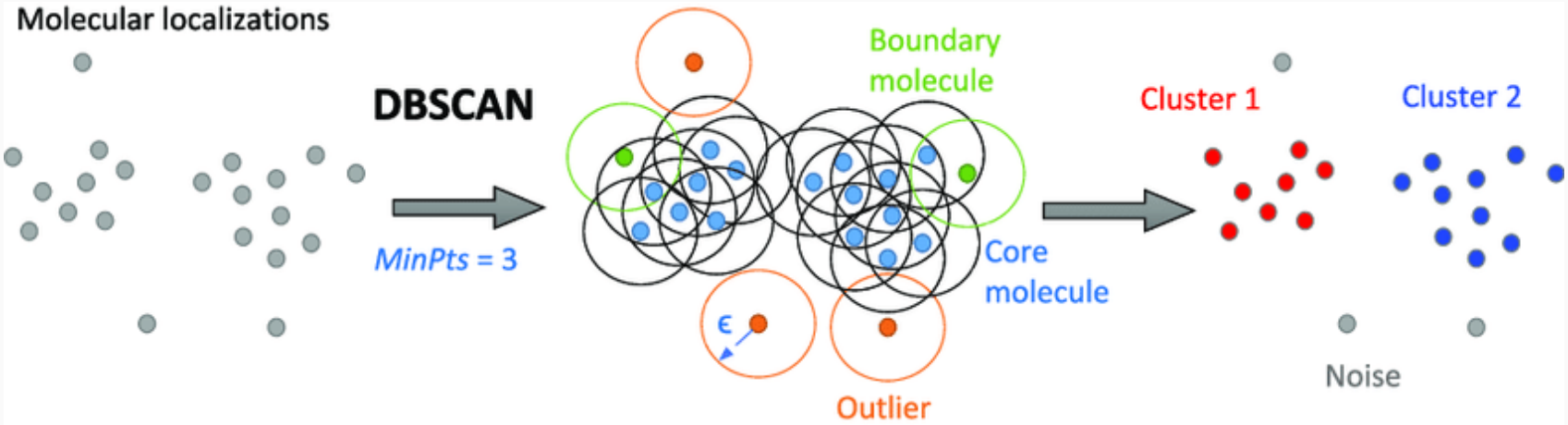
  with a hierarchical procedure.

Agglomerative clustering

Pros and cons

- It is a simple iterative algorithm.
- It is suboptimal (does not necessarily return the optimal partition) but provides a hierarchical structure of groups.
- It is deterministic given $X_1, \ldots, X_n$.
- Groups may be non-convex.

DBSCAN



Source: `https://www.researchgate.net`

DBSCAN

The method

- *Density Based Spatial Clustering and Applications with Noise*.
- No assumption on data.
- Iterative growing and birth of groups.
- Three types of points:
  1. Core points (at least m neighbors with a distance ϵ).
  2. Reachable points (non-core points in the ϵ-neighborhood of a core point).
  3. Outliers.
- Parameters: number of neighbors m and radius ϵ.
- Movie.

Connection with agglomerative clustering

- With $m = 2$, DBSCAN is similar to single linkage with a dendrogram cut at ϵ.

DBSCAN

Pros and cons

- It is a simple iterative algorithm.
- No need to specify the number of groups.
- Groups may be non-convex.
- It is barely sensitive to initialization (for reachable points).
- It cannot detect groups with different densities: OPTICS (*Ordering Points To Identify the Clustering Structure*).

Evaluation and selection

Metrics

- Global: inertia

$$I = \sum_{i=1}^{n} \left[\|X_i - \hat{\mu}\|_2^2 \mathbb{1}_{X_i \in \hat{A}} + \|X_i - \hat{\mu}\|_2^2 \mathbb{1}_{X_i \in \hat{A}} \right].$$

- Individual and global: silhouette coefficient

$$S = \frac{1}{n} \sum_{i=1}^{n} \frac{b_i - a_i}{\max(a_i, b_i)},$$

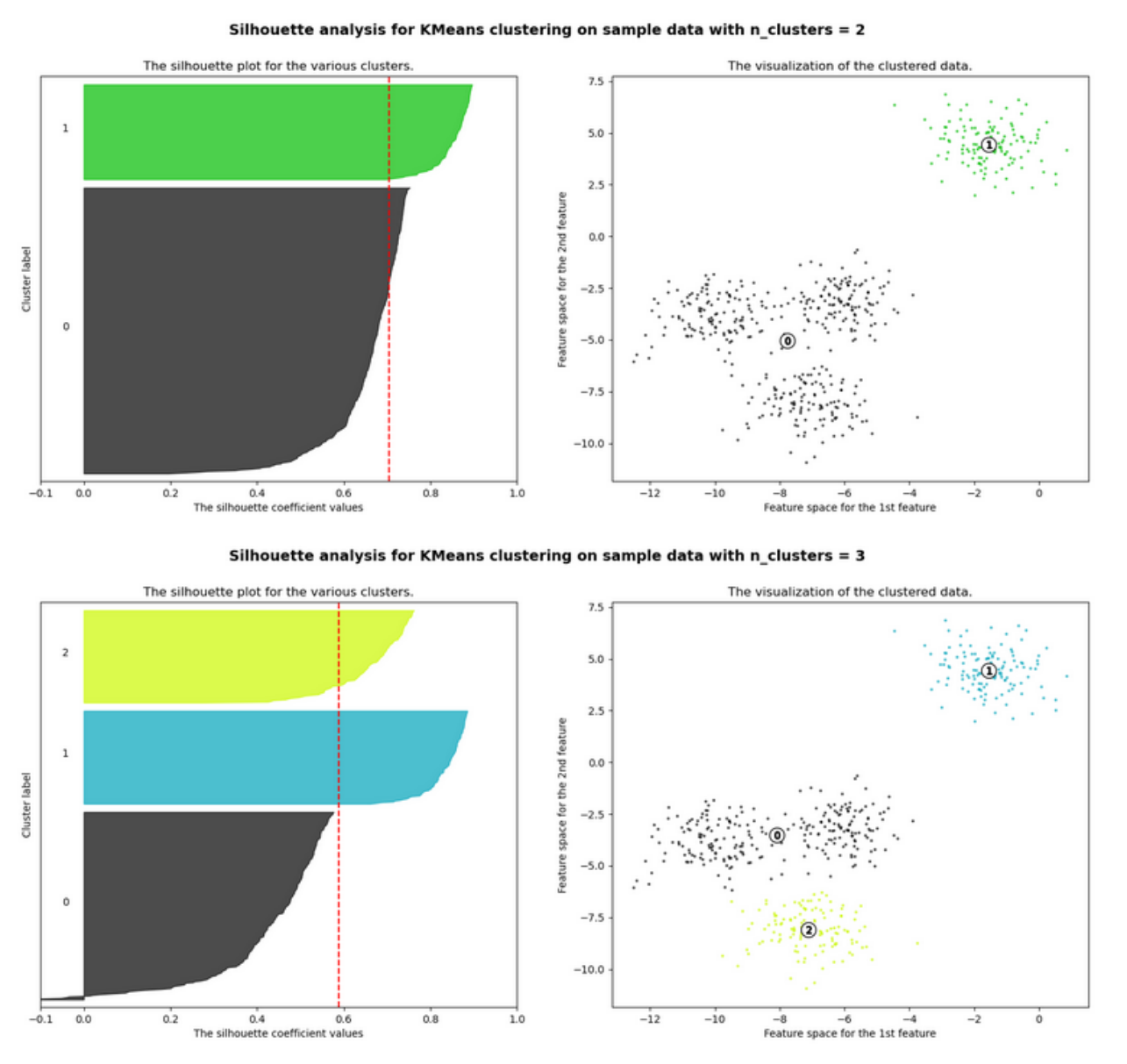
  where a_i = average distance of X_i to its group and b_i = average distance of X_i to the nearest group.

- Make sens for convex groups.

Model selection

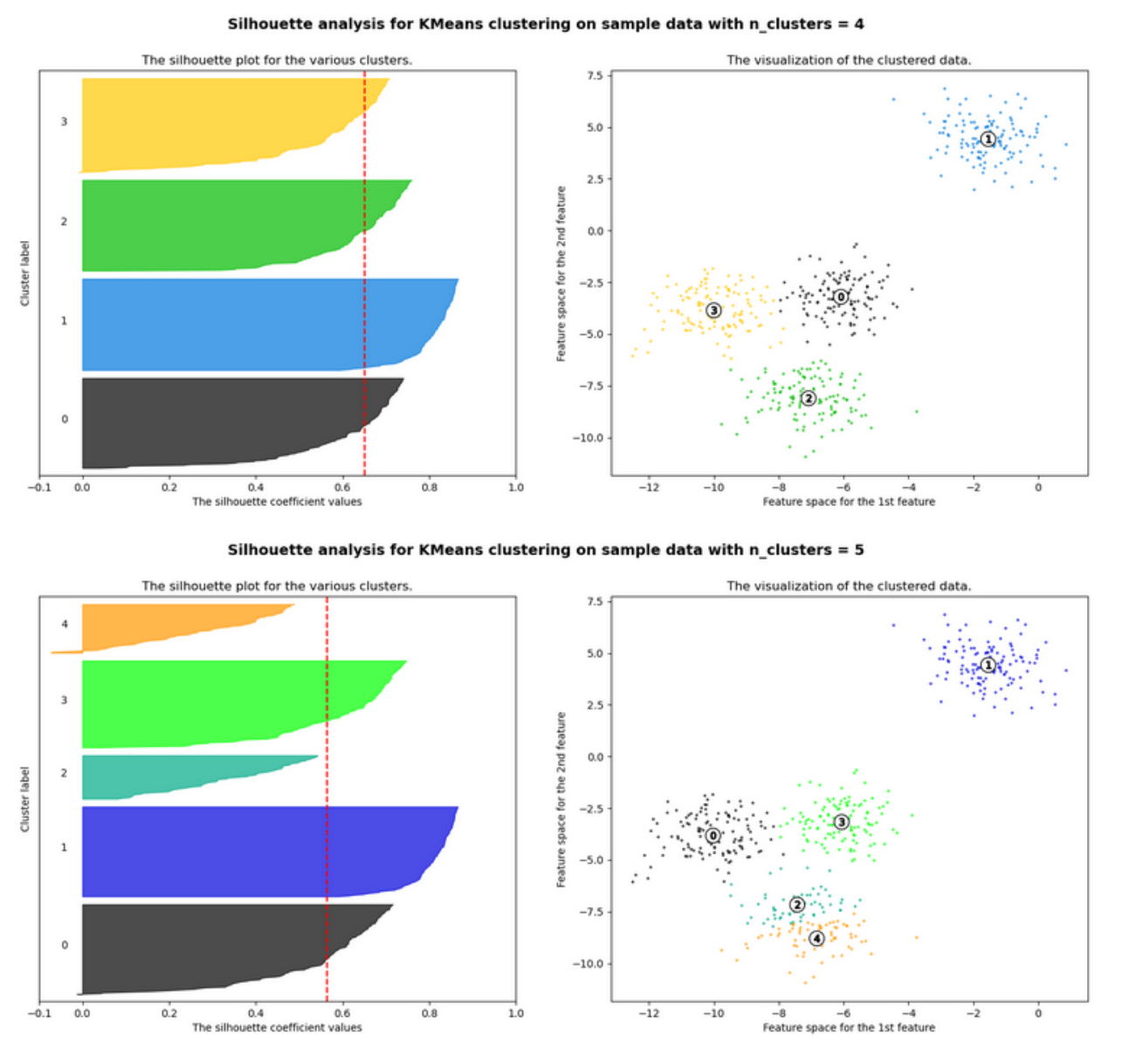
- The elbow method on inertia.
- Analyzing the silhouette coefficient: example.

Evaluation and selection



Source: `https://scikit-learn.org`

Evaluation and selection



Source: `https://scikit-learn.org`

What’s next?

What's next?

Other learning domains

- Neural networks.
- Dimensionality reduction.
- Data preprocessing.
- Time series.
- Reinforcement learning.
- Data generation.
- Active learning.
- Domain adaptation.
- Image and natural language processing.
- Causality.
- Visualization.
- . . .