


TENSOR.BY

ML-course

4. Clustering

Volha Hedranovich (Data Scientist),
hiedranovich@rocketscience.ai





Outline

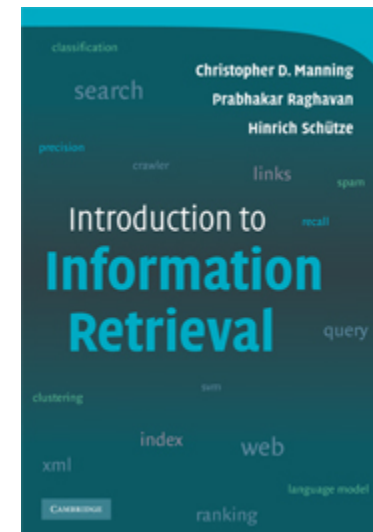
1. References
2. Overview
3. Flat clustering
4. Hierarchical clustering
5. Soft clustering

Cases

References

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, *Introduction to Information Retrieval*, Cambridge University Press. 2008.

<https://nlp.stanford.edu/IR-book/>
(available online for free)





References

jakevdp / sklearn_tutorial

Watch 55

Star 584

Fork 377

Code

Issues 1

Pull requests 3

Projects 0

Wiki

Insights

Branch: master sklearn_tutorial / notebooks /

Create new file

Upload files

Find file

History



jakevdp committed on Dec 1, 2017 Update for new library versions

Latest commit 8da069a on Dec 1, 2017

..		
fig_code	Update for new library versions	2 months ago
images	Add data layout visualization	a year ago
01-Preliminaries.ipynb	Update for new library versions	2 months ago
02.1-Machine-Learning-Intro.ipynb	Update for new library versions	2 months ago
02.2-Basic-Principles.ipynb	Update for new library versions	2 months ago
03.1-Classification-SVMs.ipynb	Update for new library versions	2 months ago
03.2-Regression-Forests.ipynb	Update for new library versions	2 months ago
04.1-Dimensionality-PCA.ipynb	Update for new library versions	2 months ago
04.2-Clustering-KMeans.ipynb	update all notebooks	2 years ago
04.3-Density-GMM.ipynb	update all notebooks	2 years ago
05-Validation.ipynb	update all notebooks	2 years ago
Index.ipynb	initial commit of tutorial materials	2 years ago
URL.ipynb	initial commit of tutorial materials	2 years ago

Jake VanderPlas, Scikit-learn Tutorial.

https://github.com/jakevdp/sklearn_tutorial

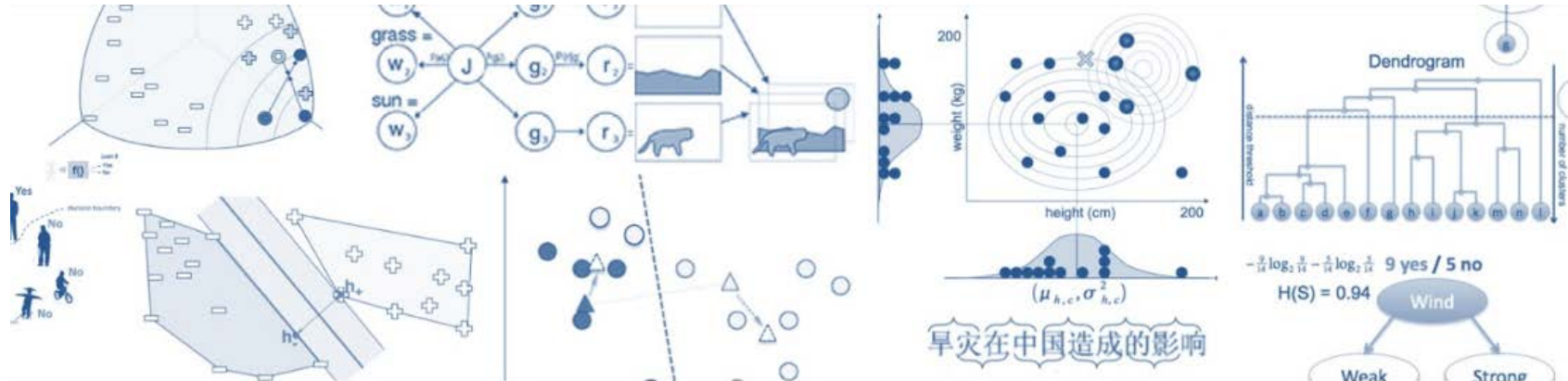
References

5



Victor Lavrenko, **ML and data analysis channel.**

<https://www.youtube.com/channel/UCs7aIOMRnxhzfKAJ4Jz7Wg>



Victor Lavrenko

23,157 subscribers

HOME

VIDEOS

PLAYLISTS

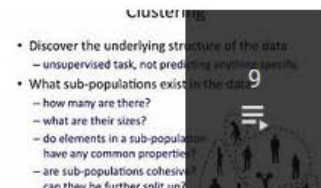
CHANNELS

DISCUSSION

ABOUT



Data Analysis ▾



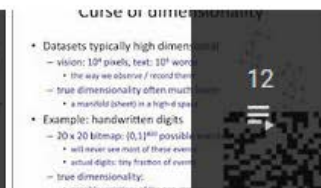
K-means Clustering

Victor Lavrenko



Mixture Models

Victor Lavrenko



Principal Component Analysis


Victor Lavrenko



Hierarchical Clustering

Victor Lavrenko

References



[Home](#)
[Installation](#)
[Documentation](#)
[Examples](#)

Previous
2.2.
Manifold...

Next
2.4.
Biclustering

Up
2.
Unsupervised

scikit-learn v0.19.1
Other versions

Please [cite us](#) if you use the software.

2.3. Clustering











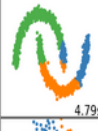








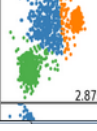

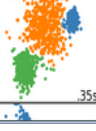
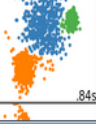
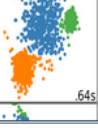
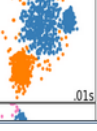
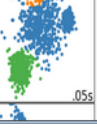
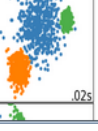
2.3.1. Overview of clustering methods

Each clustering algorithm comes in two variants: a class, that implements the `fit` method to learn the clusters on train data, and a function, that, given train data, returns an array of integer labels corresponding to the different clusters. For the class, the labels over the training data can be found in the `labels_` attribute.

Input data

One important thing to note is that the algorithms implemented in this module can take different kinds of matrix as input. All the methods accept standard data matrices of shape `[n_samples, n_features]`. These can be obtained from the classes in the `sklearn.feature_extraction` module. For `AffinityPropagation`, `SpectralClustering` and `DBSCAN` one can also input similarity matrices of shape `[n_samples, n_samples]`. These can be obtained from the functions in the `sklearn.metrics.pairwise` module.

2.3.1. Overview of clustering methods

MiniBatchKMeans	AffinityPropagation	MeanShift	SpectralClustering	Ward	AgglomerativeClustering	DBSCAN	Birch	GaussianMixture
								
.01s	4.34s	.07s	1.48s	.23s	.12s	.01s	.04s	.01s
								
.02s	4.79s	.05s	2.83s	.22s	.12s	.01s	.04s	.01s
								
.02s	2.87s	.10s	.35s	.84s	.64s	.01s	.05s	.02s

<http://scikit-learn.org/stable/modules/clustering.html>



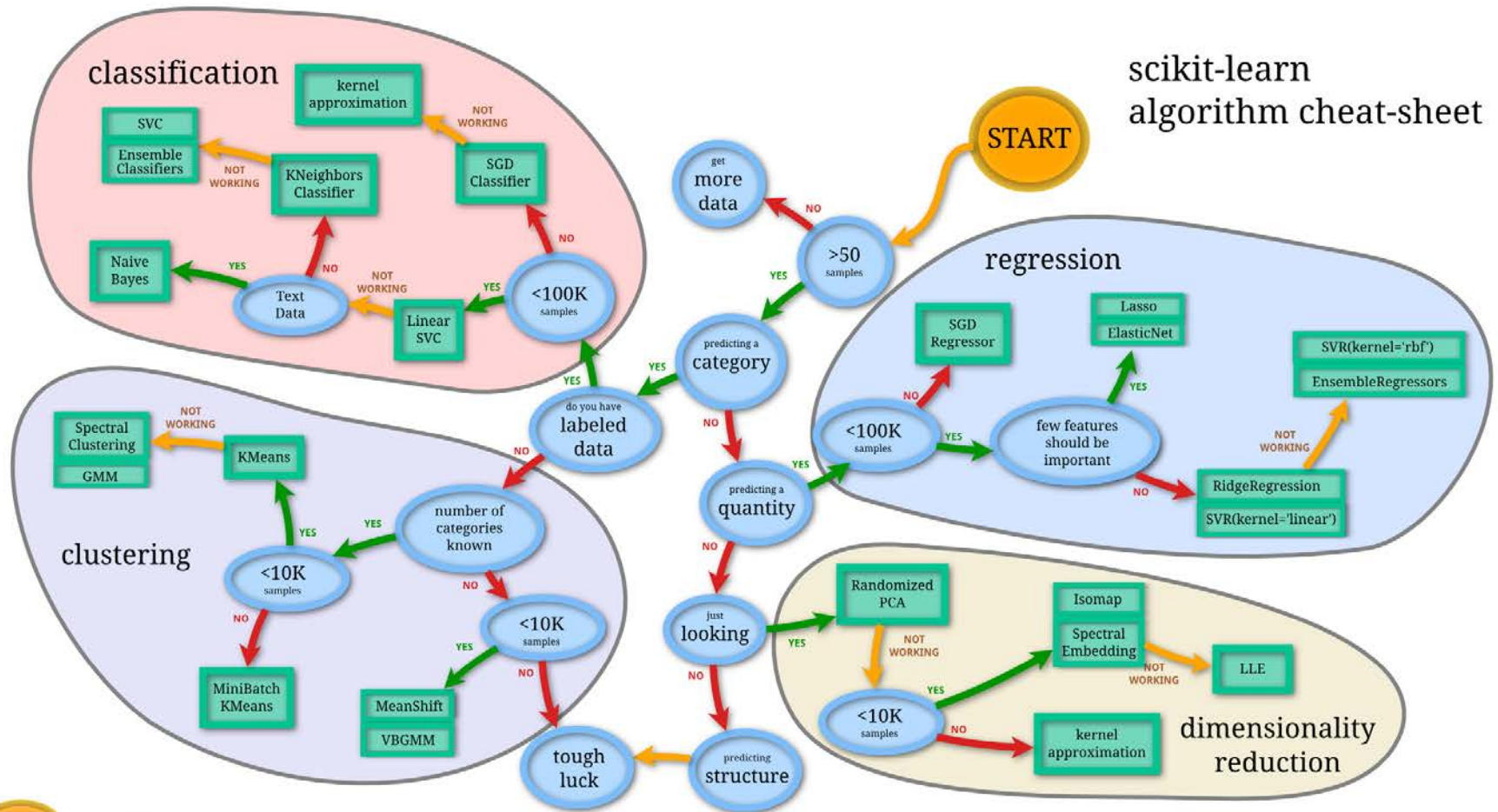
Overview

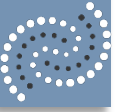
Clustering is an **unsupervised** learning task

It takes into account:

- NO labels
- NO division into train, valid and test data
- Similarity of observations

Overview

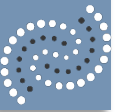




Overview

To discover / understand the underlying structure of the data

- Are there subsets in the data?
- How many?
- What sizes?
- What are common properties?
- Can the subsets be further split-up?



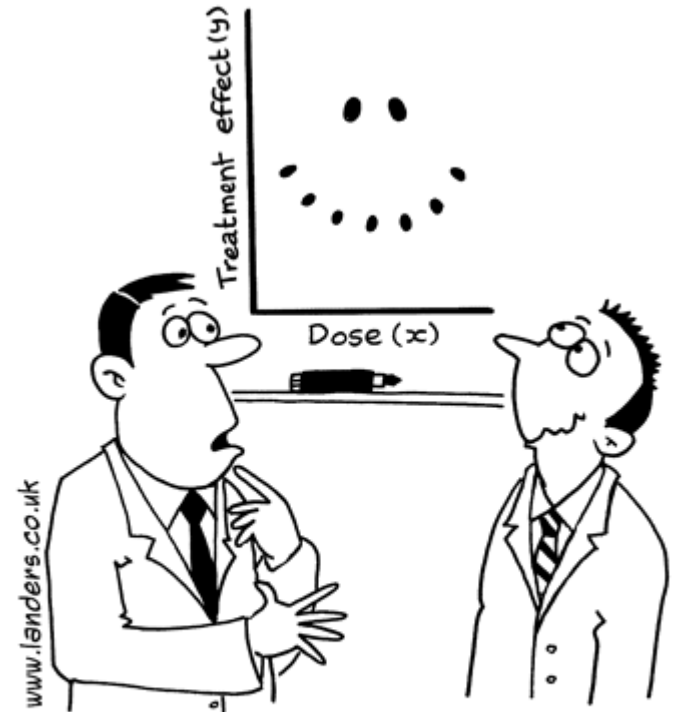
What clusters?



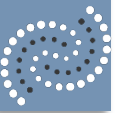


Overview: Basic tasks

- to split the data
- to understand the structure of the data
- to define outliers
- to generate features
- to reduce dimensionality
- to balance the data
- to define centroids



"It's a non-linear pattern with outliers.....but for some reason I'm very happy with the data."



Overview: Types of methods

Goal:

- **Monothetic:** cluster members have a common property
- **Polythetic:** cluster members are similar to each other

Overlap:

- **Hard:** do not overlap
- **Soft:** may overlap (a degree of attribution to a cluster)

Model-based:

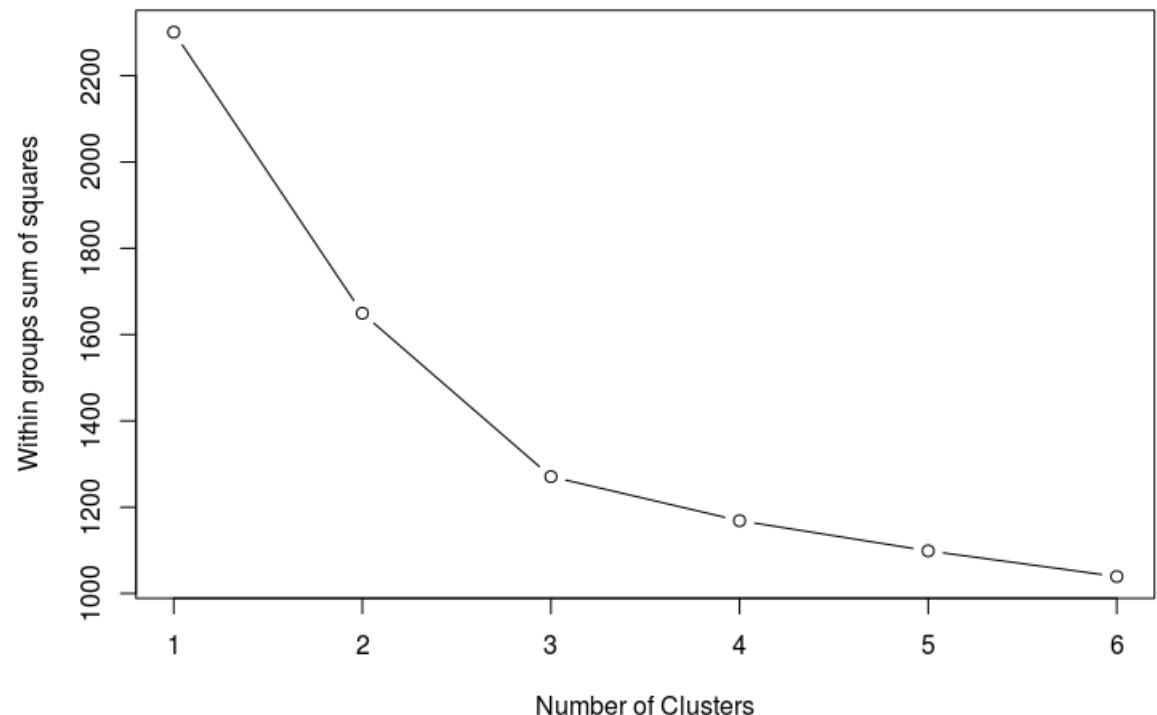
- **Flat:** set of groups (K-means)
- **Hierarchical:** taxonomy or linkage



Overview: Cardinality

The number of clusters (e.g. denoted by K).

- A good guess based on experience or domain knowledge
- Labels (if available)
- Task constraints
- Natural division
- Optimize variance — “Scree” plot



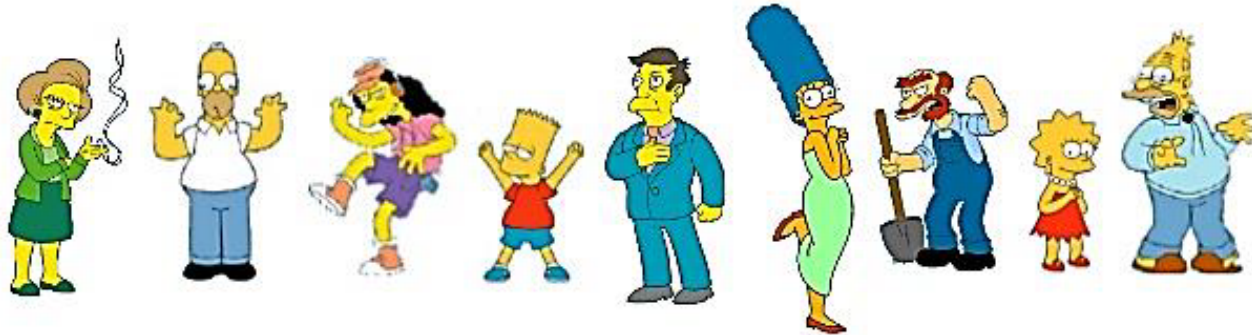


Overview: Basic methods

- K-means clustering
 - splits data into a specified number of populations
 - polythetic, hard boundaries, flat
- Agglomerative clustering
 - creates an “ontology” of nested sub-populations
 - polythetic, hard boundaries, hierarchical
- Gaussian mixtures
 - fits a mixture of K Gaussians to the data
 - polythetic, soft boundaries, flat

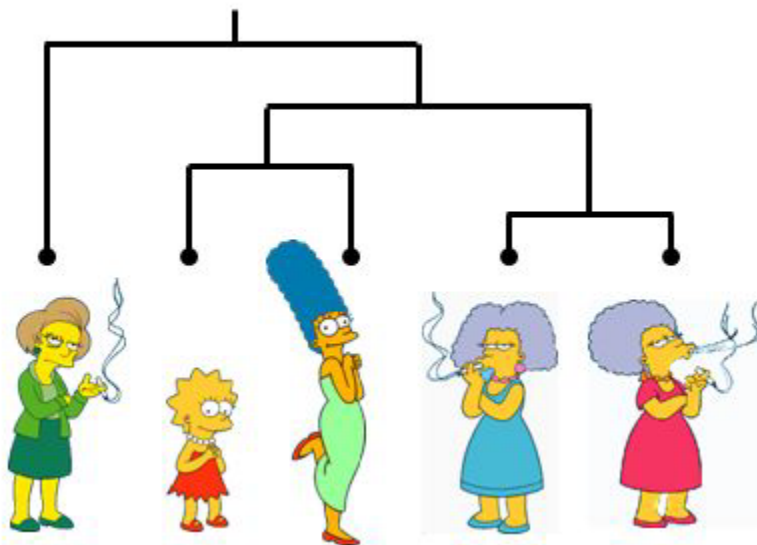
Overview

15



Hierarchical

Partitional





Overview: Is clustering satisfying?

- Does it help to solve the overall task?
 - Represent images with features
 - Get important features
 - Train models within sup-populations
 - Identify outliers
- Is it good enough by itself?
 - Helps to understand the data
 - Corresponds to classes
 - Expert evaluation



Overview: Evaluation of clustering

Set of metrics:

- Labels are present
 - Adjusted Rand index
 - Mutual Information based scores
 - Homogeneity, completeness and V-measure
 - Fowlkes-Mallows scores
- No labels → compare models:
 - Silhouette Coefficient
 - Calinski-Harabaz Index



Overview: Cluster labeling

If users interact with clusters, we must label clusters, so that users can see what a cluster is about.

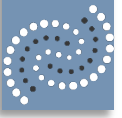
- Mutual information (MI)
- Frequency-based selection
- Centroid terms
- Title nearest to centroid, etc.

	labeling method		
# docs	centroid	mutual information	title
622	oil plant mexico production crude power 000 refinery gas bpd	plant oil production barrels crude bpd mexico dolly capacity petroleum	MEXICO: Hurricane Dolly heads for Mexico coast
1017	police security russian people military peace killed told grozny court	police killed military security peace told troops forces rebels people	RUSSIA: Russia's Lebed meets rebel chief in Chechnya
1259	00 000 tonnes traders futures wheat prices cents september tonne	delivery traders futures tonne tonnes desk wheat prices 000 00	USA: Export Business - Grain/oilseeds complex



K-means: properties

- Polythetic, hard, flat
- Data is partitioned into K sub-populations
- Points in each sub-population are similar to a centroid

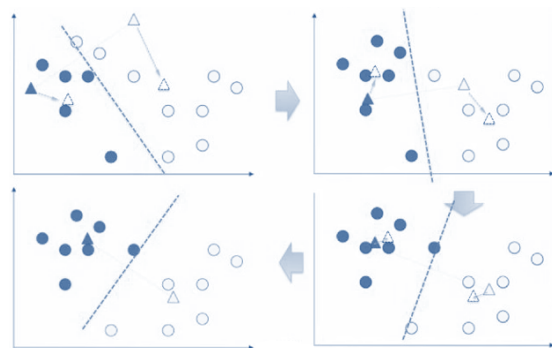
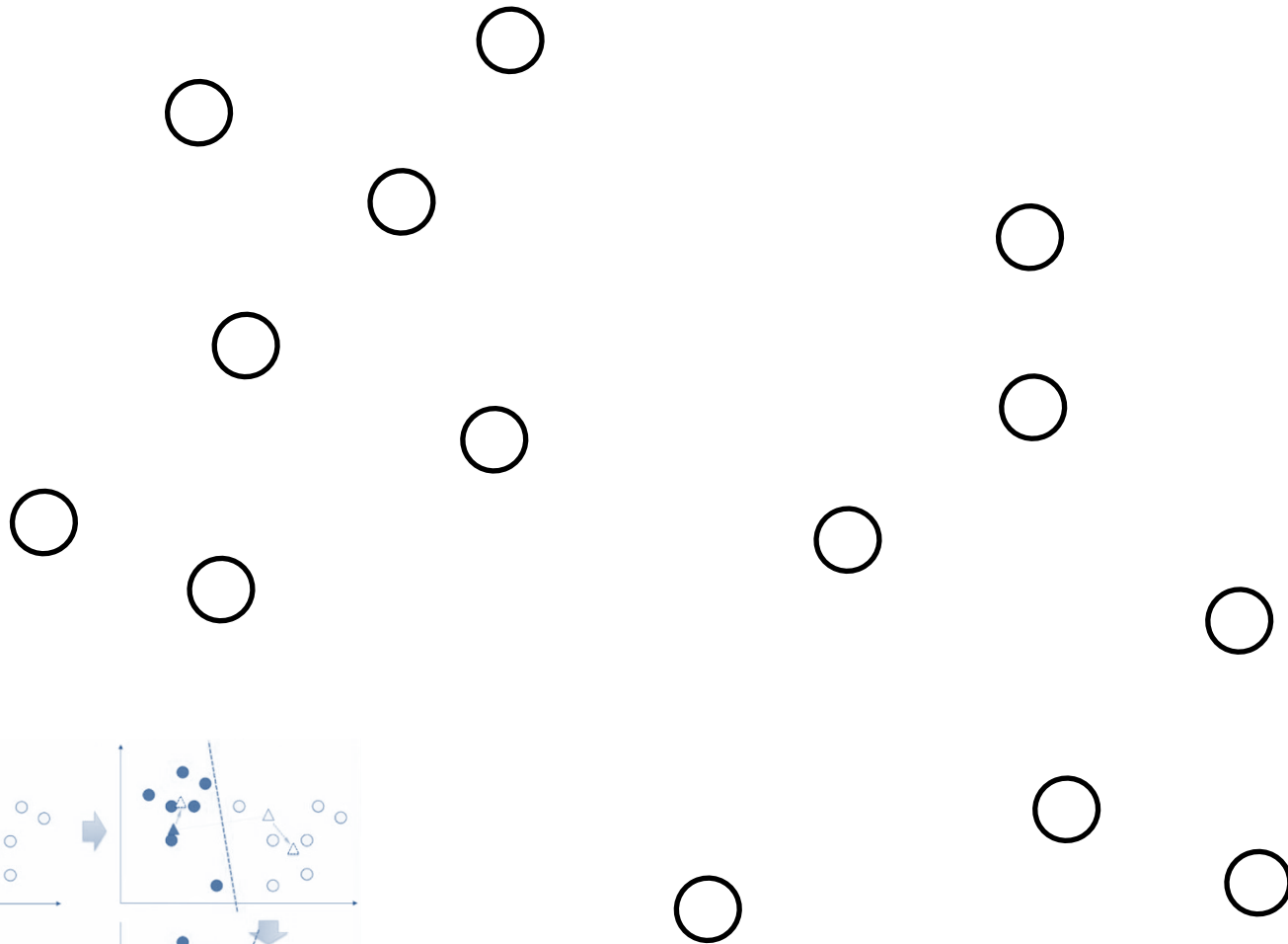


K-means: algorithm

- Inputs: K , set of points
- Places centroids at random locations
- Repeats until convergence:
 - for each point
 - find nearest centroid
 - assign the point to cluster
 - for each cluster
 - new centroid - mean of all points assigned to cluster in previous step
- Stops when assignments don't change



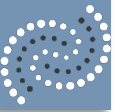
K-means



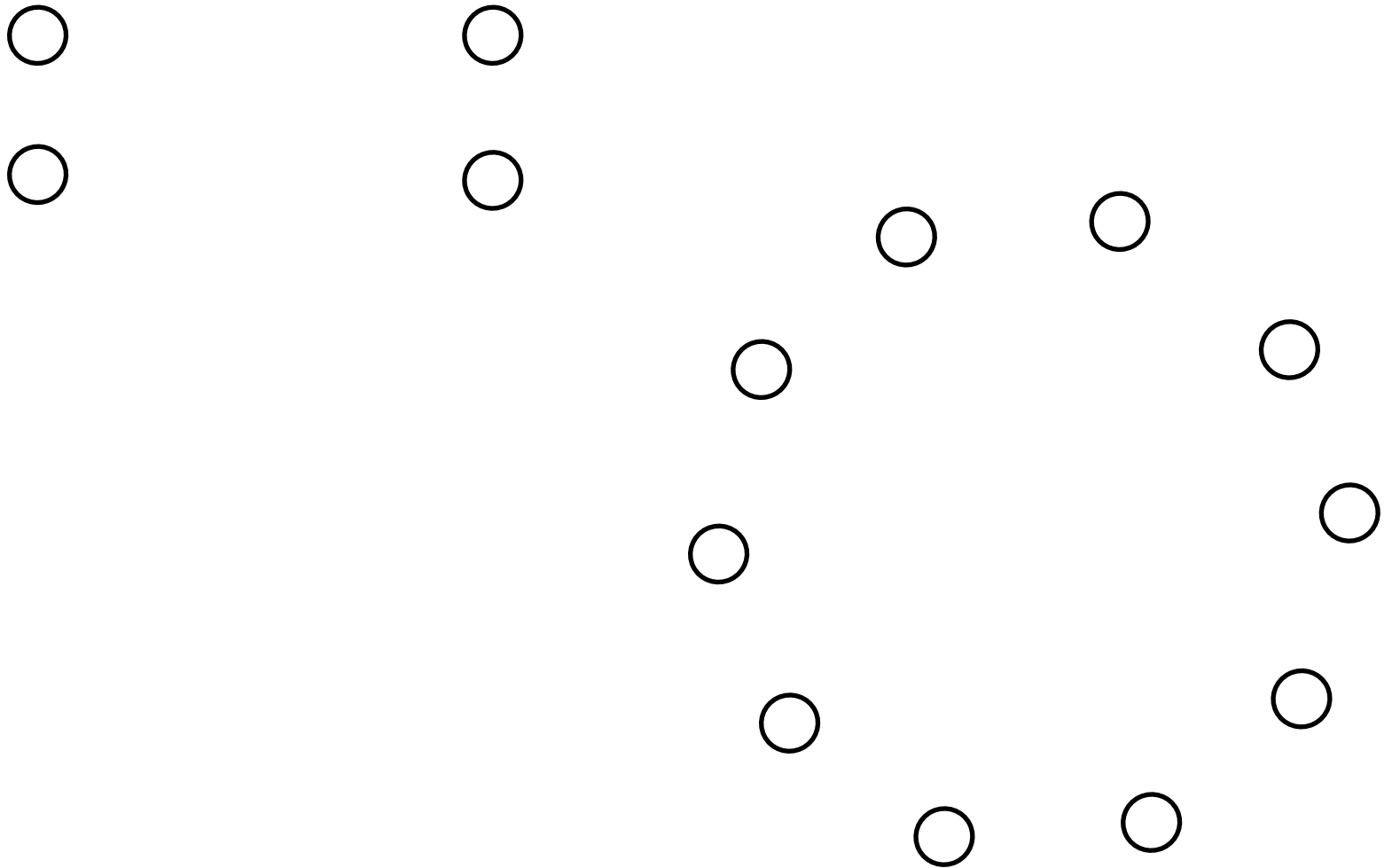


K-means

- Minimizes aggregate intra-cluster distance
 - total squared distance from point to center of its cluster
 - the same as variance if Euclidian distance is used
- Converges to a local minimum
 - different starting points – different results
 - run several times and choose model with the smallest aggregate distance
- Nearby points may not appear in the same cluster
- Centroids may appear not representative



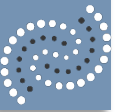
K-means





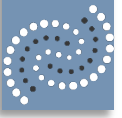
Principal Component Analysis (PCA)

- Developed to capture as much of the variation in data as possible
- Linear combinations of the original variables, uncorrelated with each other
 - Principal component #1 (PC_1) is the direction along which there is greatest variation
 - Principal component #2 (PC_2) is the direction with maximum variation left in data, orthogonal to the PC_1



Cases: Image posterization



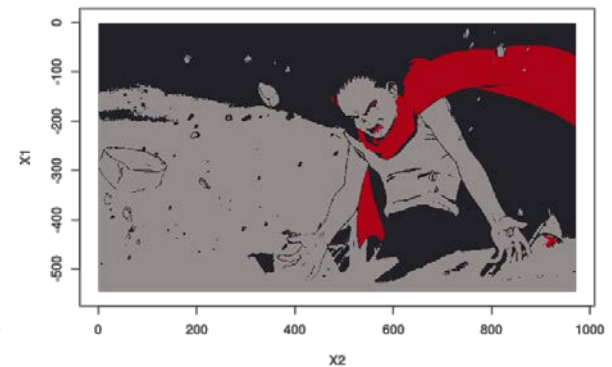
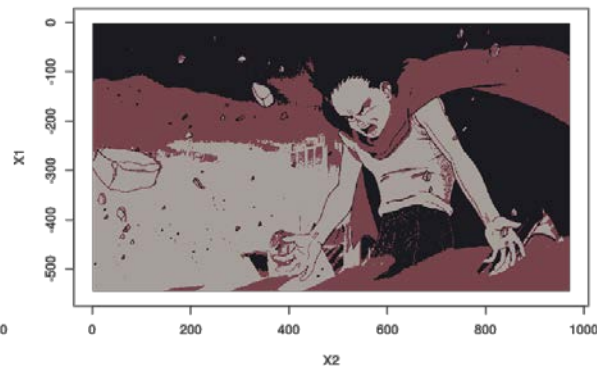
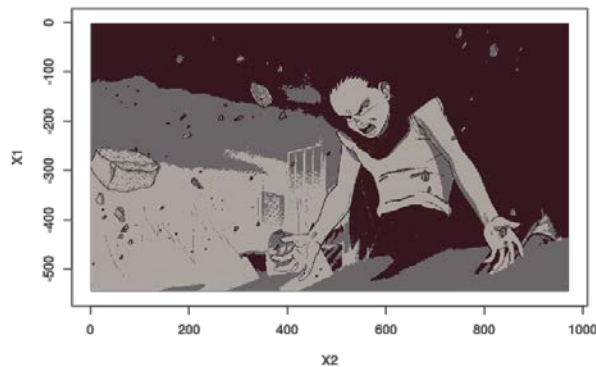
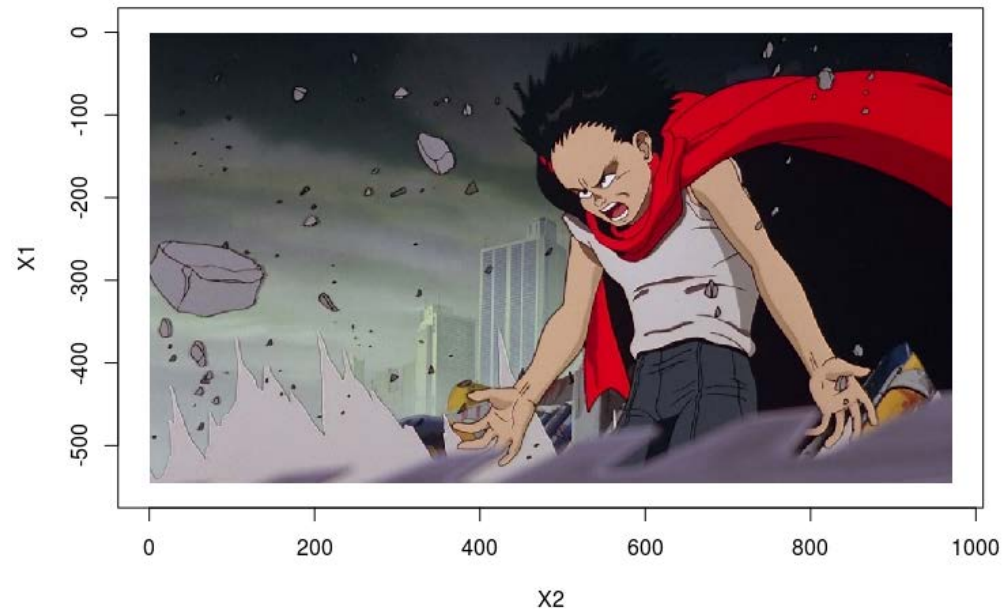


Cases: Image posterization

1. Get RGB levels of each pixel
2. K-means clustering
3. Build a plot upon pixels painted with centroids' colours



Cases: Image posterization





Hierarchical clustering

Instead of picking K — build a hierarchy of data:

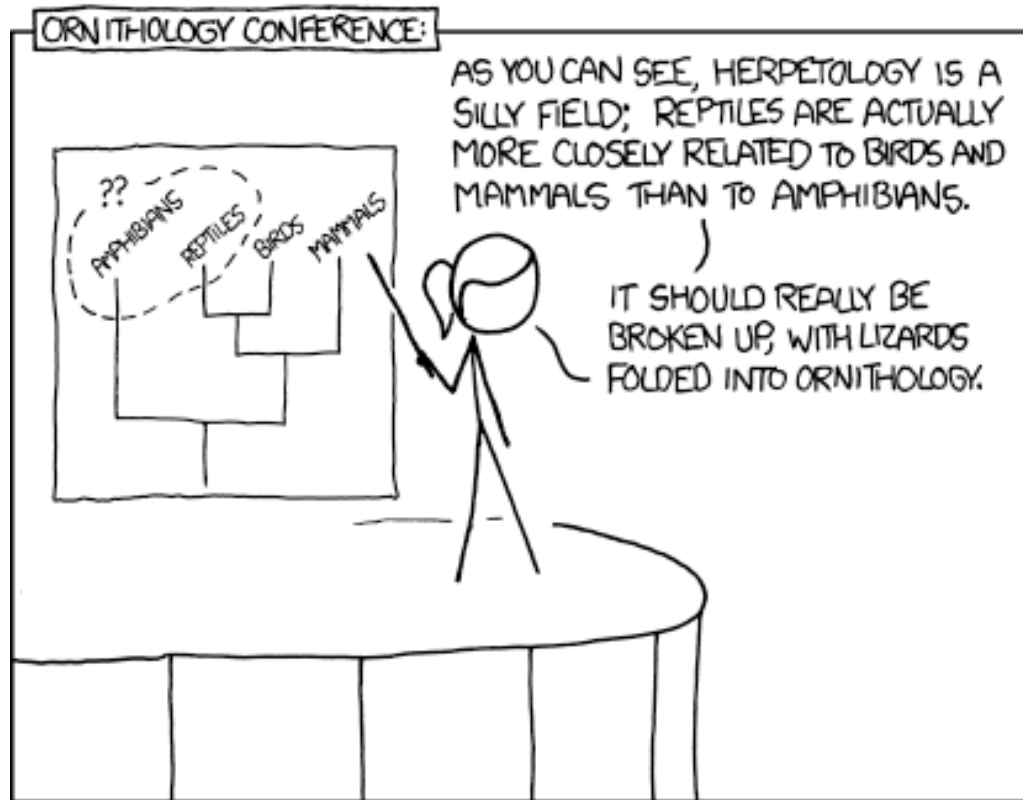
- A topmost cluster contains every point from the dataset.
- A bottom level contains singleton clusters — one per each data point.

Strategies:

- top-down (divisive)
- bottom-up (agglomerative)



Hierarchical clustering





Hierarchical: top-down

Algorithm:

- Run e.g. K-means on the overall data
- Recursively run K-means on each cluster subset

Properties:

- Relatively fast
- Nearby points can appear in different clusters
- It has the advantage of being more efficient if we do not generate a complete hierarchy all the way down to individual document leaves.
- Top-down clustering benefits from complete information about the global distribution when making top-level partitioning decisions.



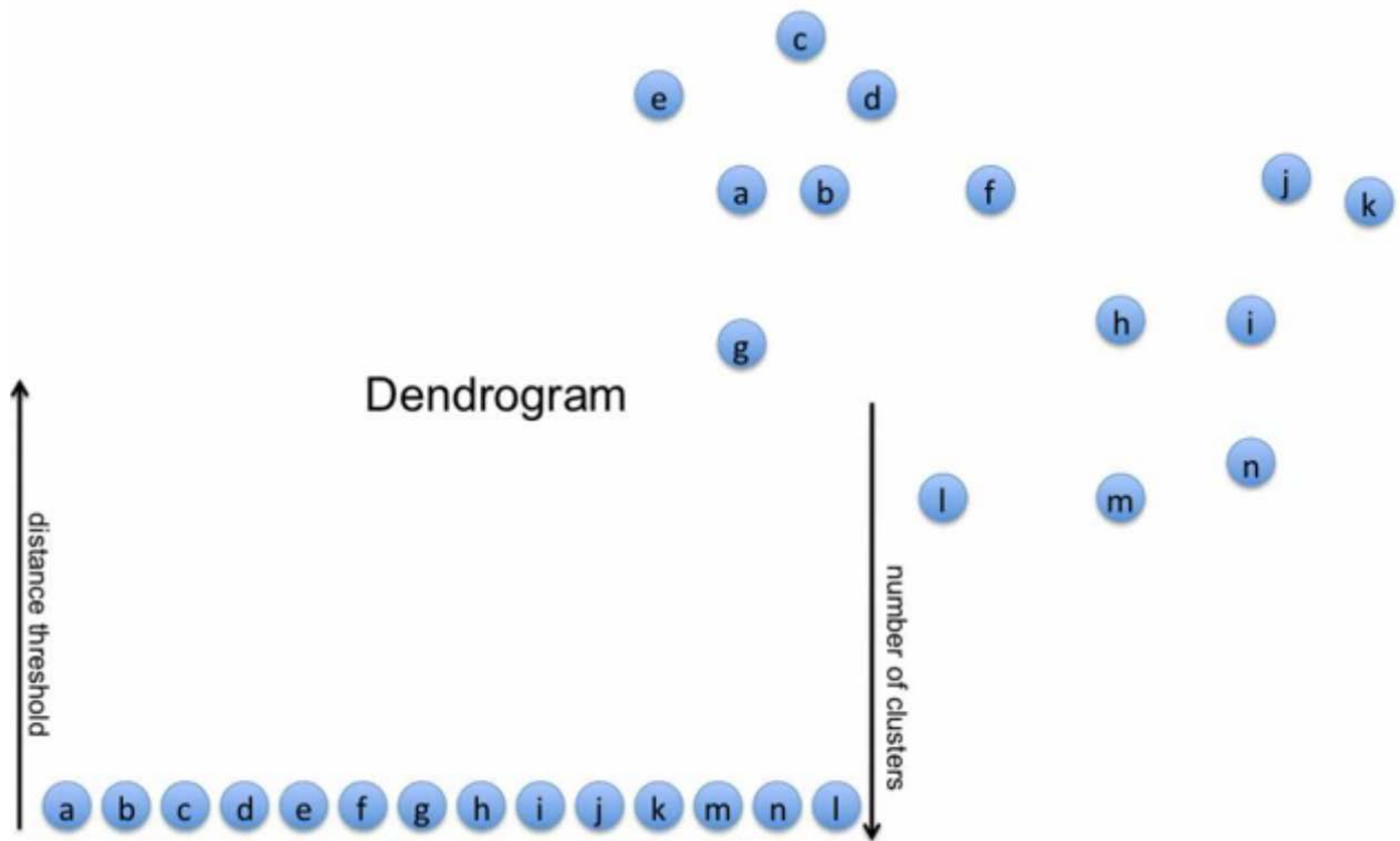
Hierarchical: bottom-up

Agglomerative clustering: nearby points must appear in the same cluster

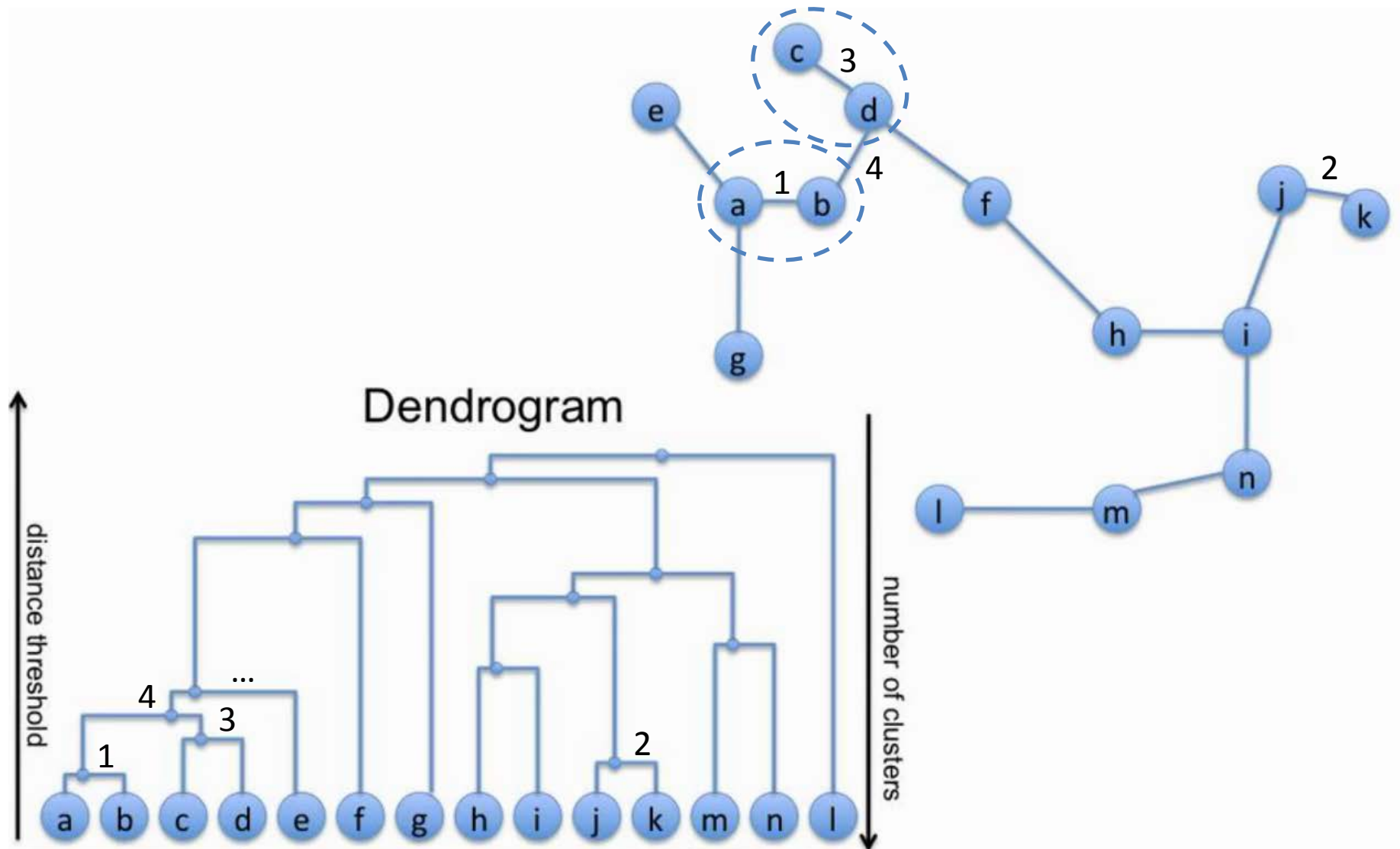
- Start with singleton points
- Repeat until one cluster is left
 - find a pair of closest clusters
 - merge it into one cluster
 - Re-estimate distances between clusters
- Relatively slow
- Reproduces a dendrogram



Hierarchical: bottom-up

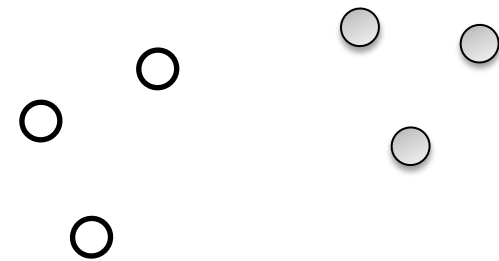


Hierarchical: bottom-up

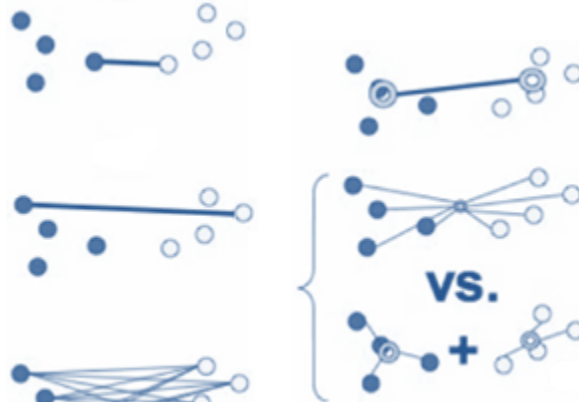




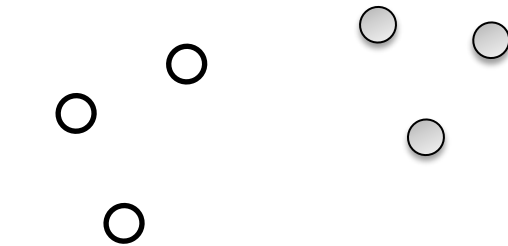
Hierarchical: Distances



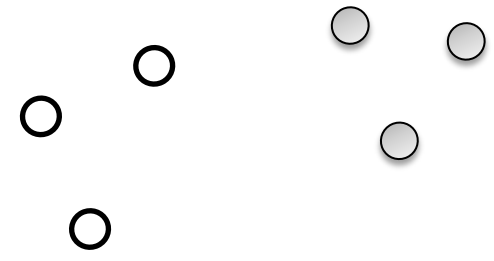
Single link: d. between closest elements in clusters (chaining effect)



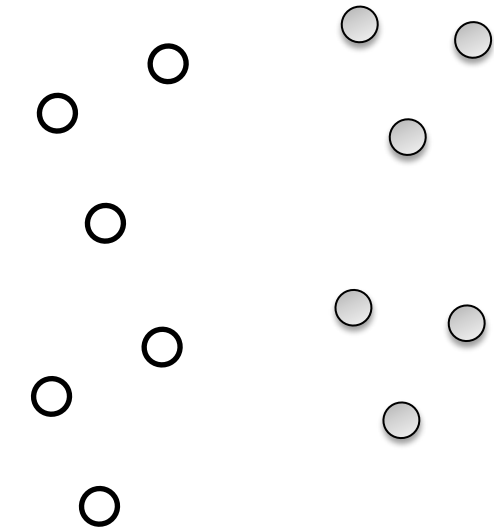
Complete link: d. between farthest elements in clusters (sensitive to outliers)



Average link: average of all pairwise distances (usually best choice)



Centroids: d. between centroids of clusters (inversions can occur)

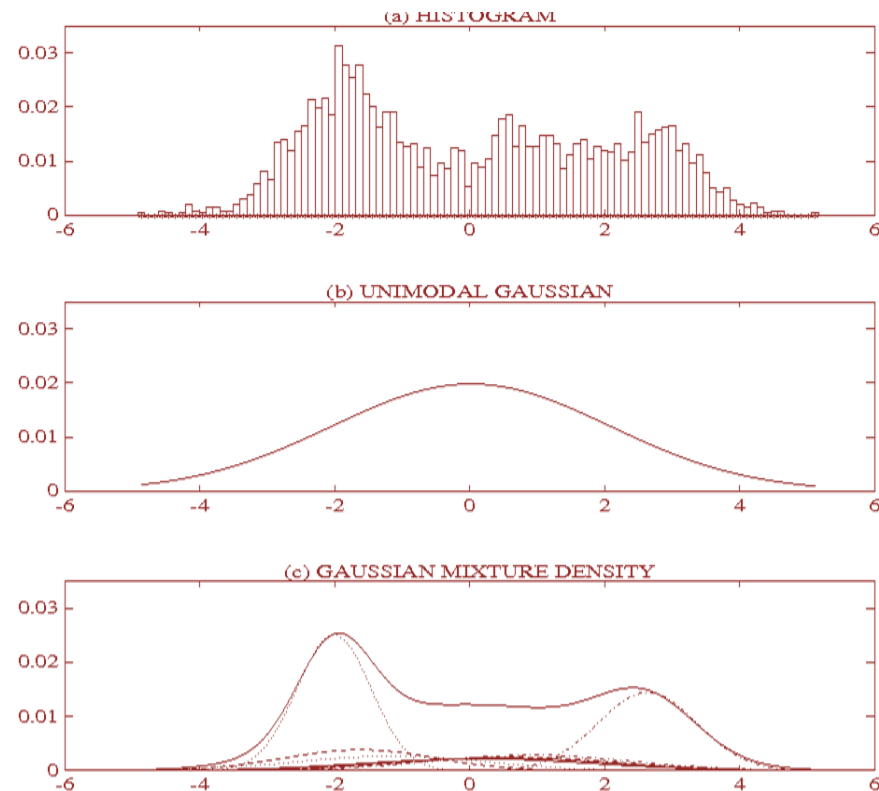


Ward's method: joining of clusters depends on how does it change the total d. from centroids (takes variance into account)



Gaussian mixtures

- Probabilistically-grounded way for performing soft clustering
- Each cluster is a generative model (e.g. Gaussian)
- Parameters (mean, covariance) are unknown





Gaussian mixtures

If we would know the sources:

- observations x_1, \dots, x_n
- $K = 2$ Gaussians with unknown μ, σ^2

$$\mu_a = \frac{x_1 + x_2 + \dots + x_{n_a}}{n_a}$$

$$\sigma_a^2 = \frac{(x_1 - \mu_a)^2 + \dots + (x_n - \mu_a)^2}{n_a}$$





Gaussian mixtures

If we wouldn't know the sources but would know parameters (μ, σ^2) :

- Possible to guess whether point is more likely to be **a** or **b**

$$P(b|x_i) = \frac{P(x_i|b) \cdot P(b)}{P(x_i|b) \cdot P(b) + P(x_i|a) \cdot P(a)}$$

$$P(x_i|b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \cdot \exp\left(-\frac{(x_i - \mu_b)^2}{2\sigma_b^2}\right)$$





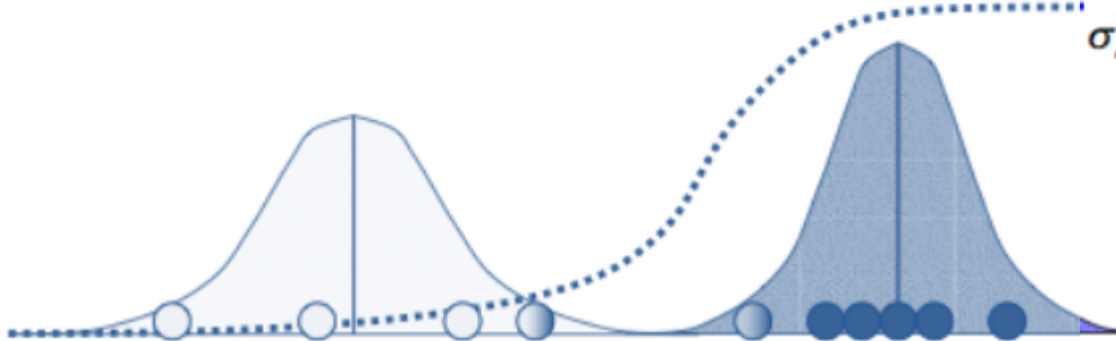
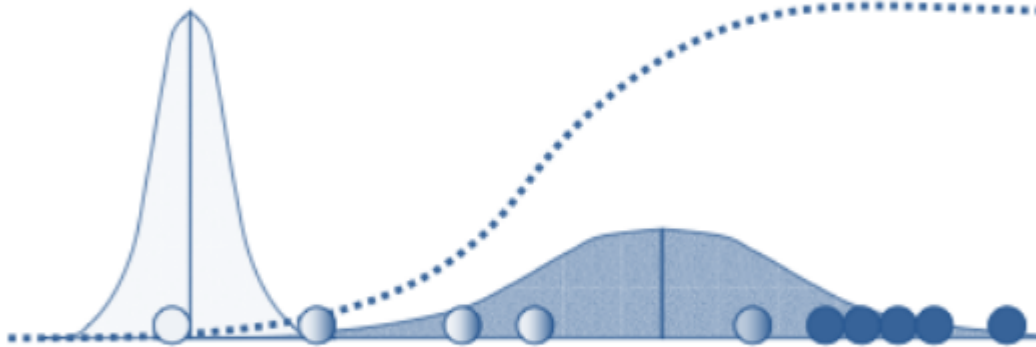
Gaussian mixtures

GMM algorithm

- starts with two randomly placed Gaussians
- for each point a probability is estimated:
did it come from **a** or **b**
- adjusts parameters to fit points assigned to them
- iterates until convergence



Gaussian mixtures



$$P(x_i | b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{(x_i - \mu_b)^2}{2\sigma_b^2}\right)$$

$$b_i = P(b | x_i) = \frac{P(x_i | b)P(b)}{P(x_i | b)P(b) + P(x_i | a)P(a)}$$

$$a_i = P(a | x_i) = 1 - b_i$$

$$\mu_b = \frac{b_1 x_1 + b_2 x_2 + \dots + b_n x_n}{b_1 + b_2 + \dots + b_n}$$

$$\sigma_b^2 = \frac{b_1 (x_1 - \mu_b)^2 + \dots + b_n (x_n - \mu_b)^2}{b_1 + b_2 + \dots + b_n}$$

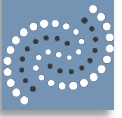
$$\mu_a = \frac{a_1 x_1 + a_2 x_2 + \dots + a_n x_n}{a_1 + a_2 + \dots + a_n}$$

$$\sigma_a^2 = \frac{a_1 (x_1 - \mu_a)^2 + \dots + a_n (x_n - \mu_a)^2}{a_1 + a_2 + \dots + a_n}$$

could also estimate priors:

$$P(b) = (b_1 + b_2 + \dots + b_n) / n$$

$$P(a) = 1 - P(b)$$



Gaussian mixtures

- Maximizes likelihood of the data (soft)
- Sensitive to starting point
- Converges to a local maximum
(until change in P is sufficiently small –
threshold or max number of iterations parameter)



Gaussian mixtures: Case

	mpg	hp			
			Fiat 128	32.4	66
Mazda RX4	21.0	110	Honda Civic	30.4	52
Mazda RX4 Wag	21.0	110	Toyota Corolla	33.9	65
Datsun 710	22.8	93	Toyota Corona	21.5	97
Hornet 4 Drive	21.4	110	Dodge Challenger	15.5	150
Hornet Sportabout	18.7	175	AMC Javelin	15.2	150
Valiant	18.1	105	Camaro Z28	13.3	245
Duster 360	14.3	245	Pontiac Firebird	19.2	175
Merc 240D	24.4	62	Fiat X1-9	27.3	66
Merc 230	22.8	95	Porsche 914-2	26.0	91
Merc 280	19.2	123	Lotus Europa	30.4	113
Merc 280C	17.8	123	Ford Pantera L	15.8	264
Merc 450SE	16.4	180	Ferrari Dino	19.7	175
Merc 450SL	17.3	180	Maserati Bora	15.0	335
Merc 450SLC	15.2	180	Volvo 142E	21.4	109
Cadillac Fleetwood	10.4	205			
Lincoln Continental	10.4	215			
Chrysler Imperial	14.7	230			



Titanic case

Let's generate a feature for Titanic data set with clustering:

- Build K-means ($K \leq 10$, -survived)
- Predict Titanic with the K-means model
- Combine a new data frame
- Split into train and valid subsets
- Build GBM on initial data
- Build GBM on new data
- Compare

Q & A

Thank you!

Volha Hedranovich (Data Scientist),
hiedranovich@rocketscience.ai

