Article presentation:

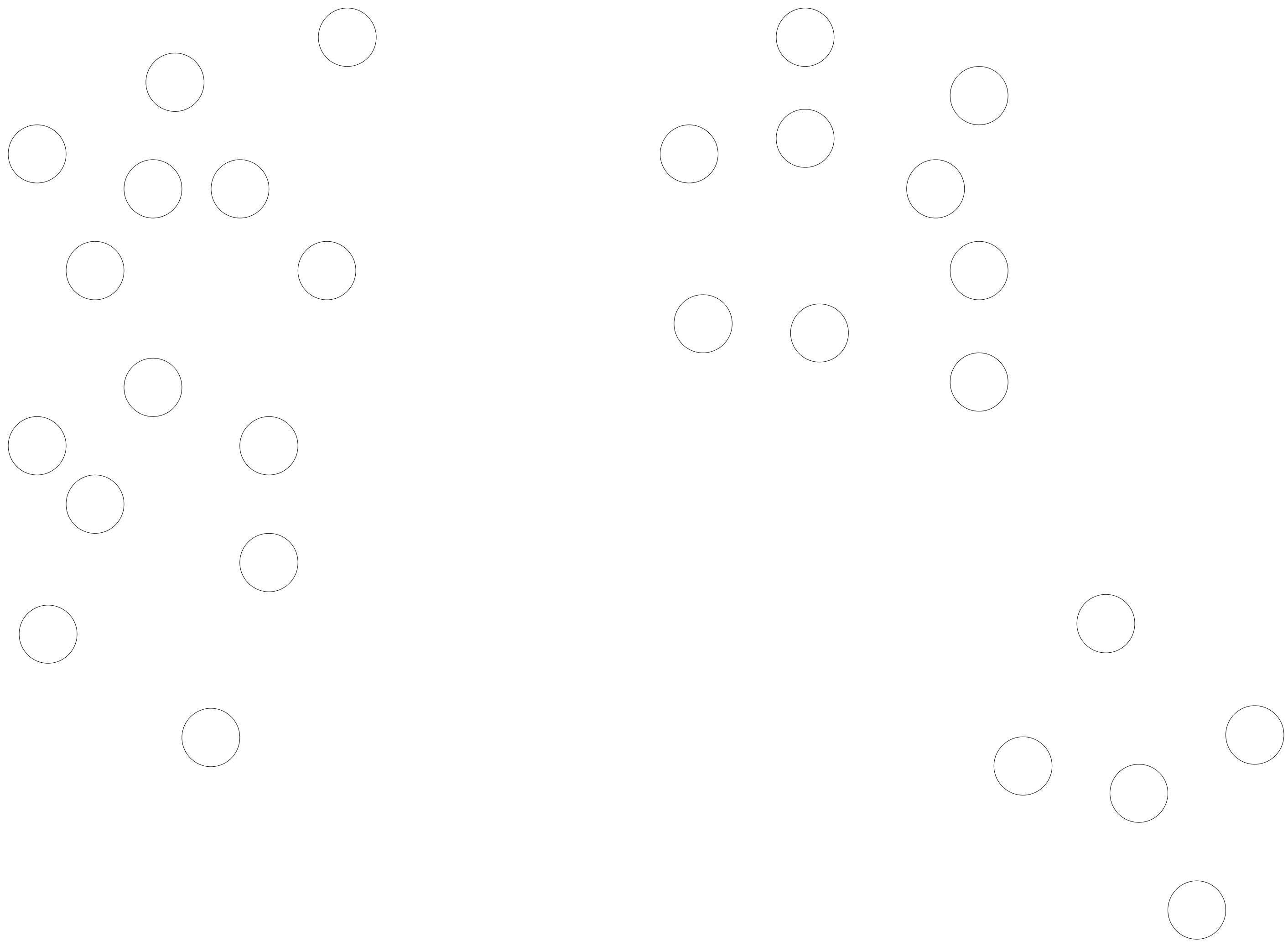
„A SOBER LOOK AT CLUSTERING STABILITY“ (Ben-David et al)

presented by Sander Sõnajalg

What is **clustering**?

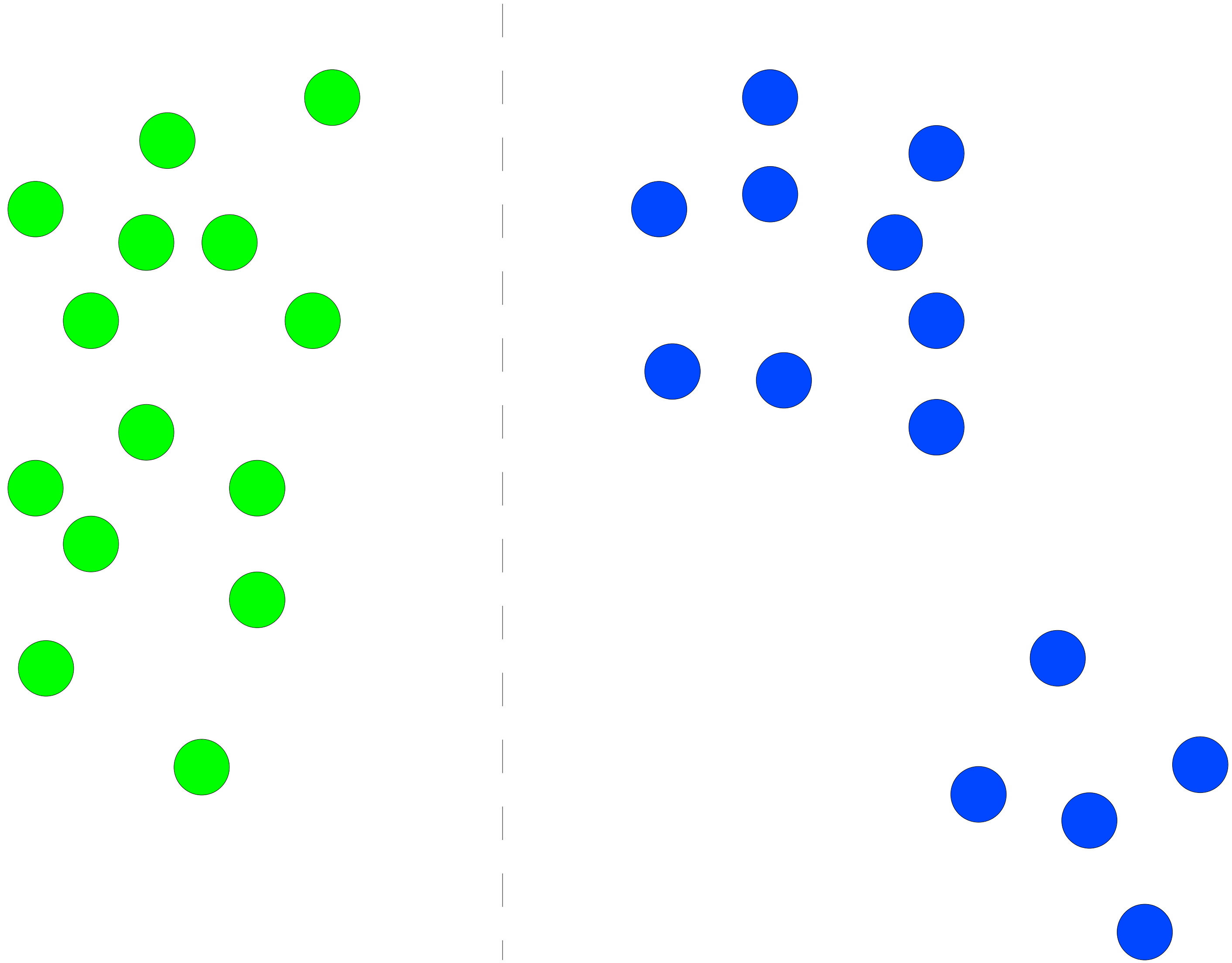
- **Problem**: How to divide a given set of datapoints into *k* meaningful groups.
- **Motivation:** Learn something new about my data, spot any patterns
- **The means:** In the given article, the focus is on the **cost-based** clustering algorithms. They all share the same idea: minimize or maximize some kind of an objective function.

What is **clustering**? (2)



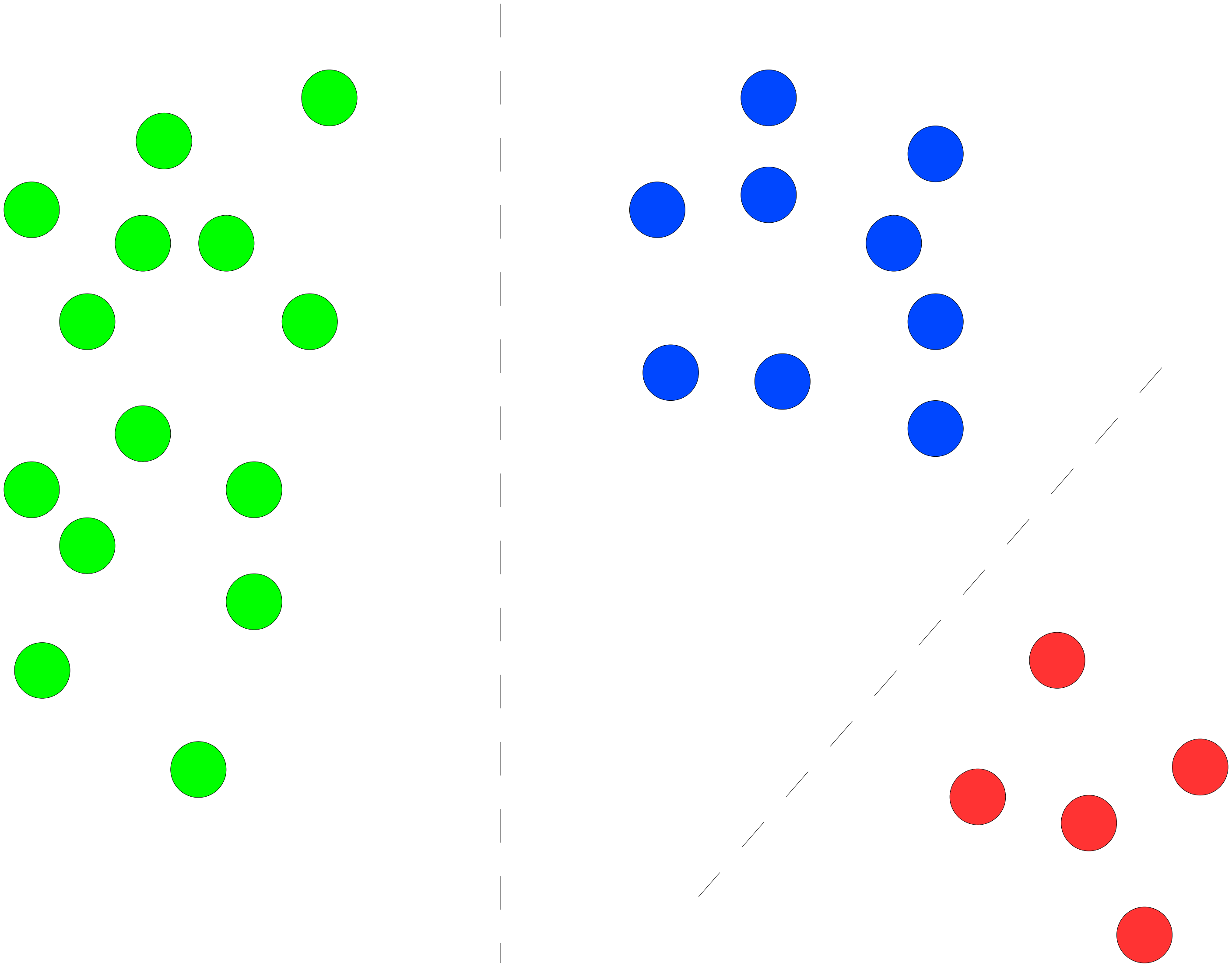
What is **clustering**? (2)

<u>k = 2</u>



What is **clustering**? (2)

k = 3

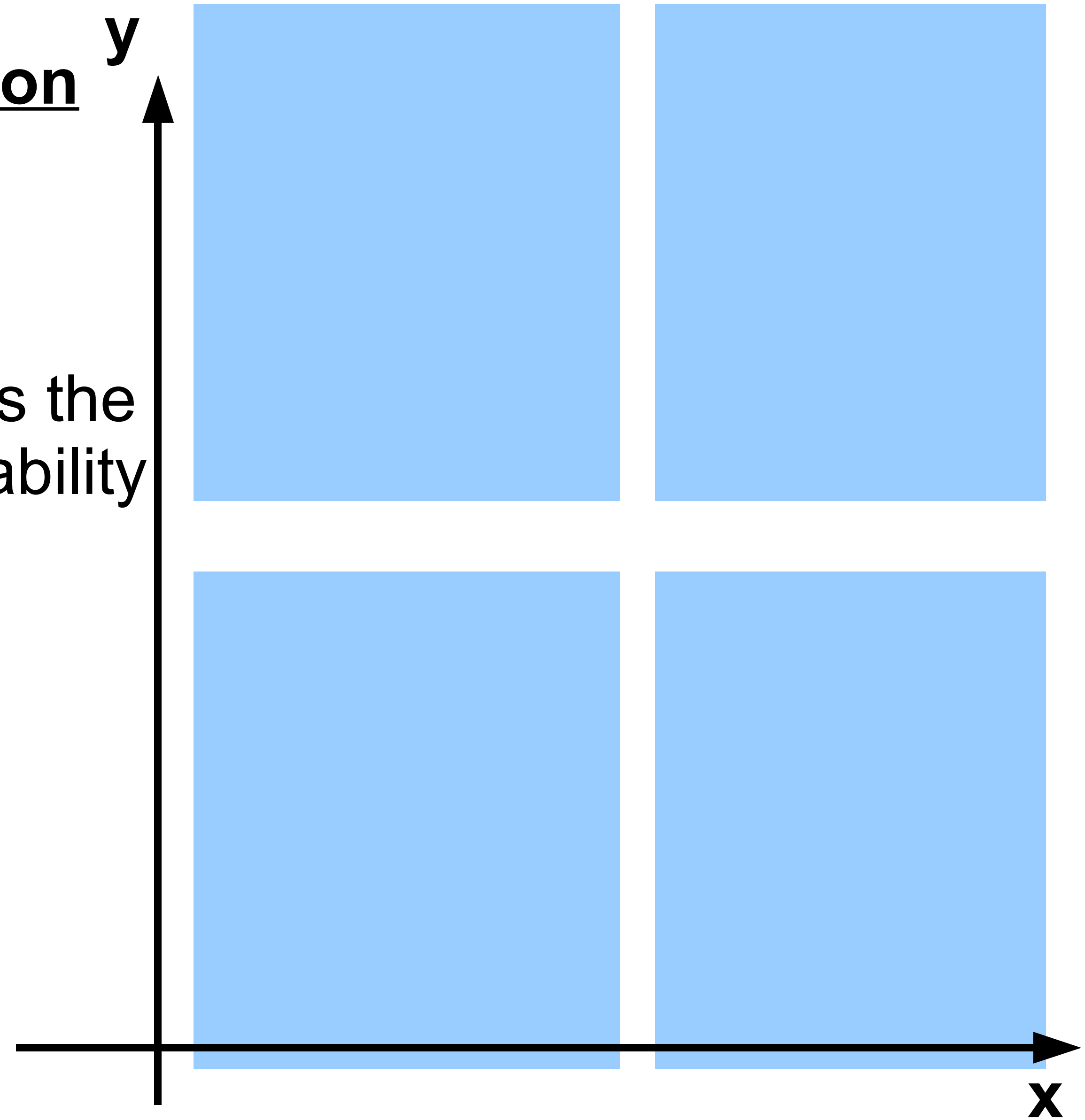


Clustering stability

The probability distribution

- Distribution is fixed, but unknown.
- (Here: blue colour marks the area for which the probability of finding a data point is higher than 0).

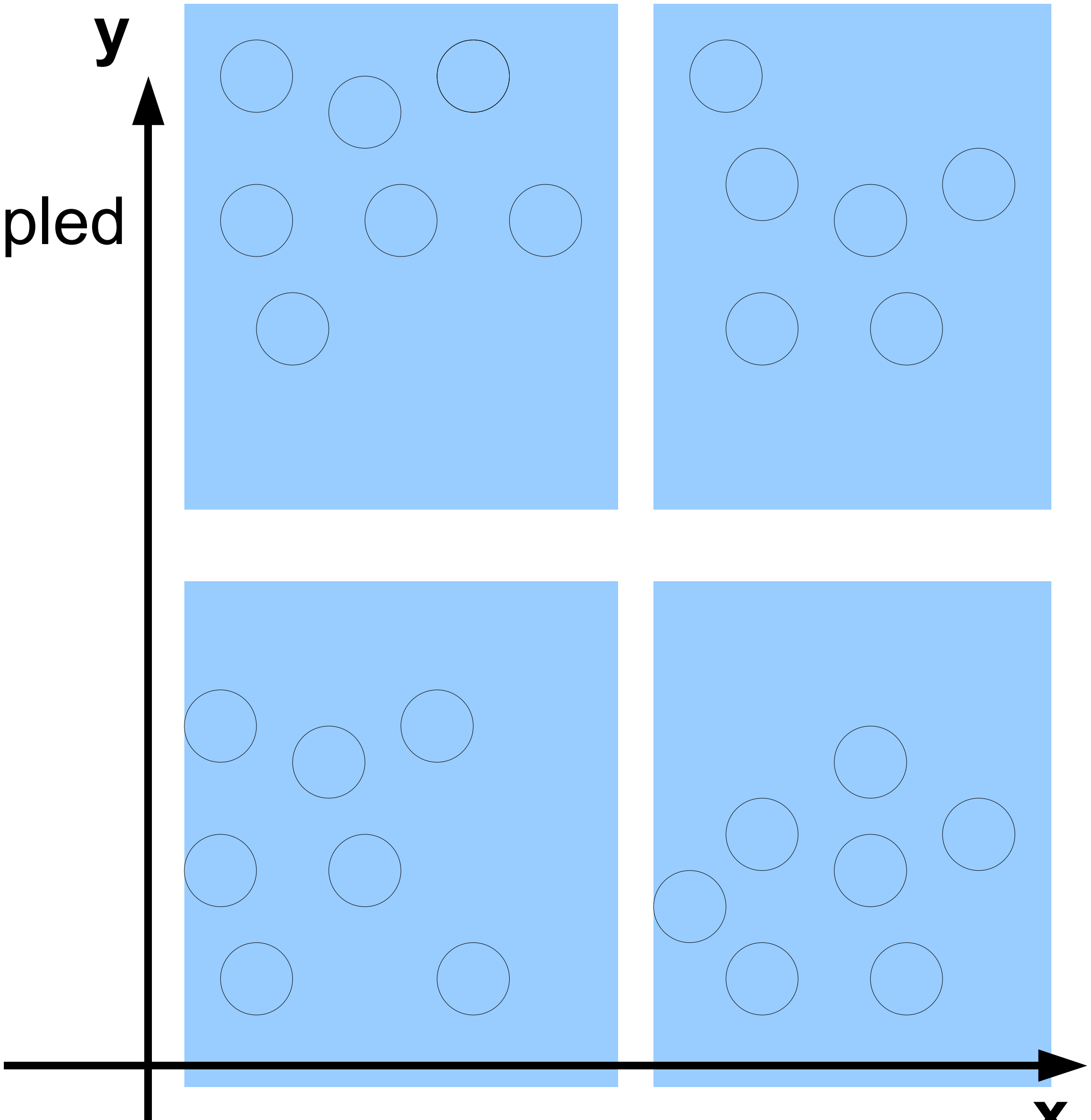
k – the number of clusters



Clustering stability

Sample

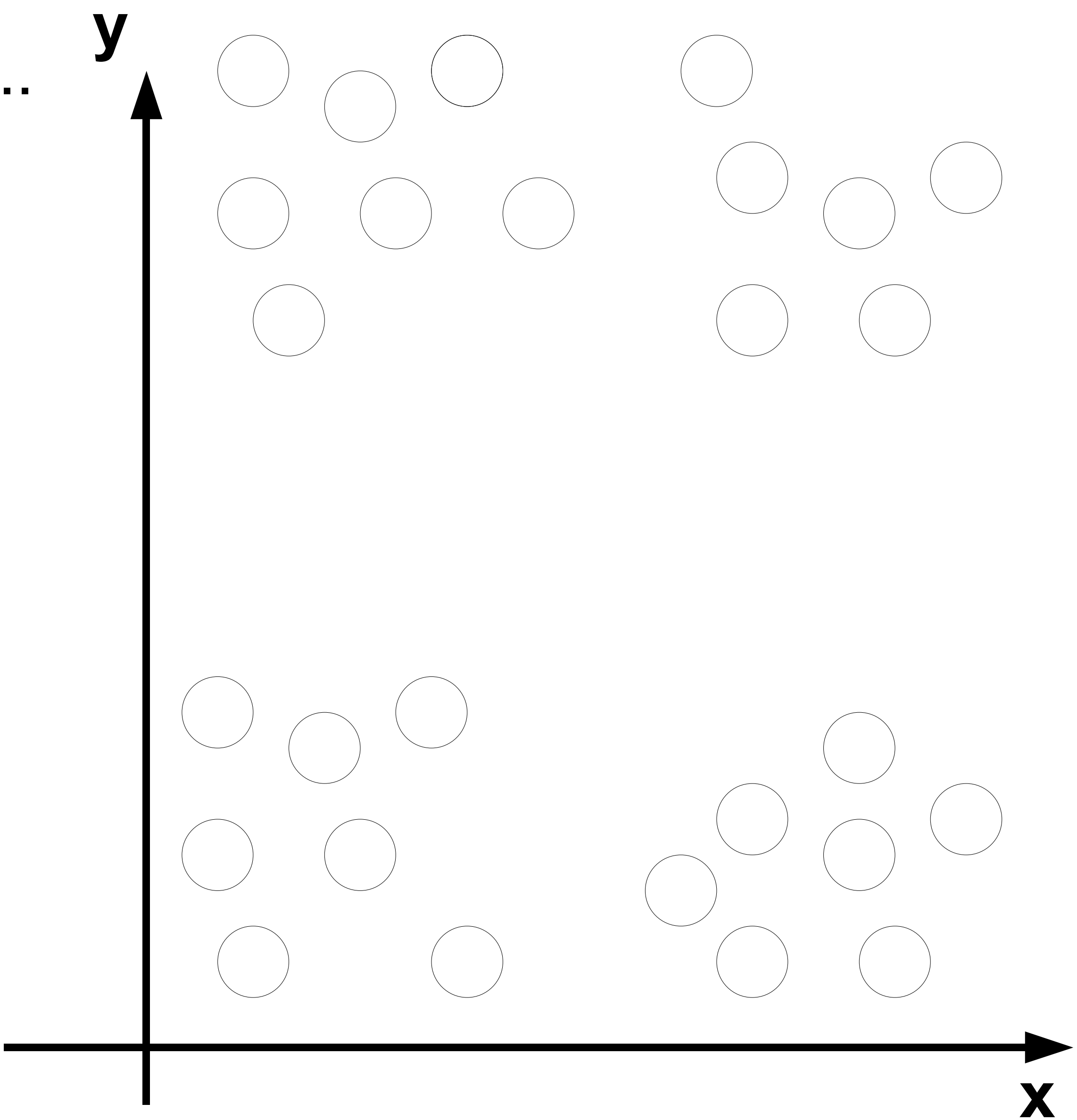
- A set of data points sampled from the underlying probability distribution.



k – the number of clusters

Clustering stability

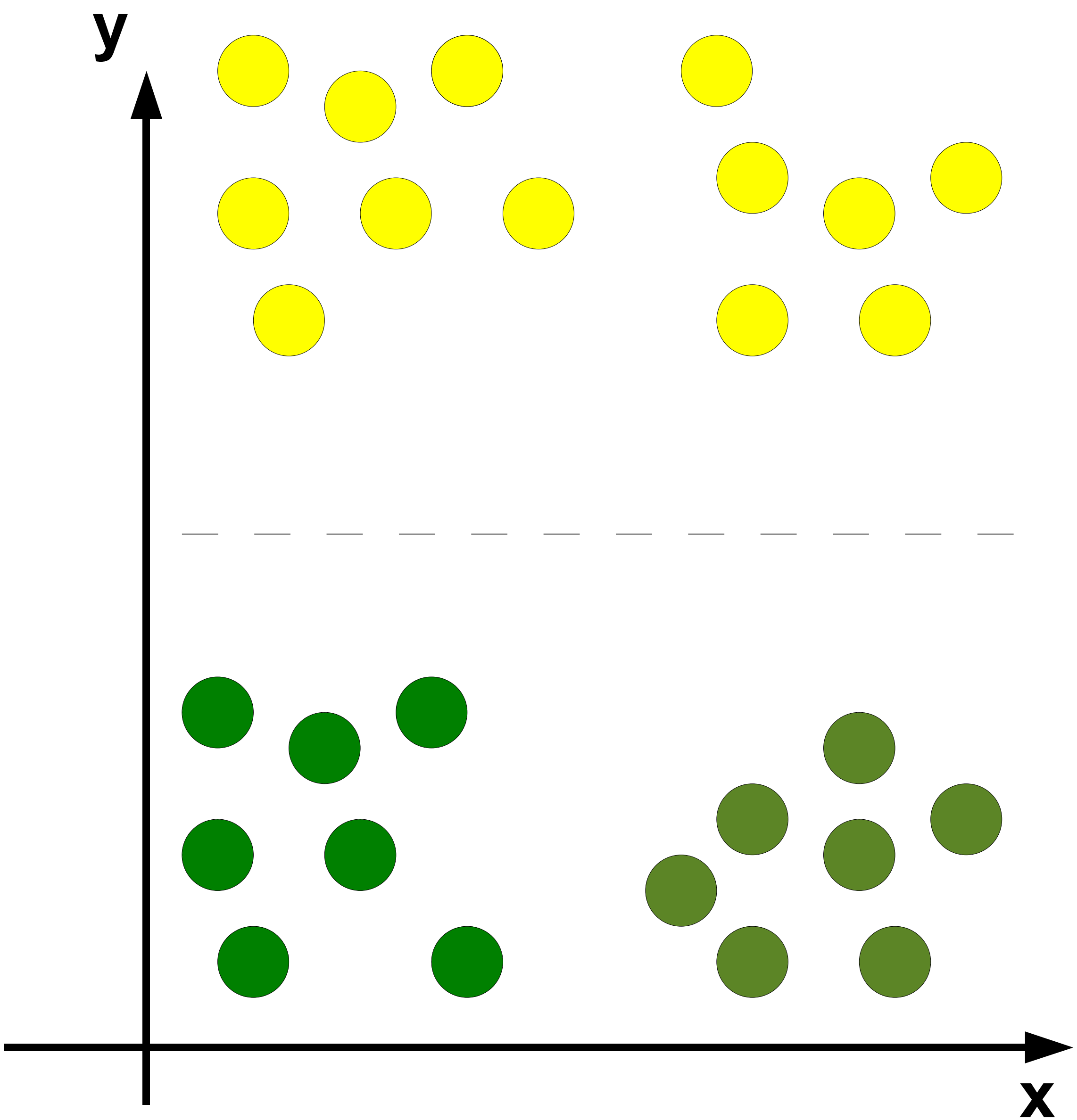
- We only see the sample.. nothing is known about the probability distribution itself.



k – the number of clusters

Clustering stability

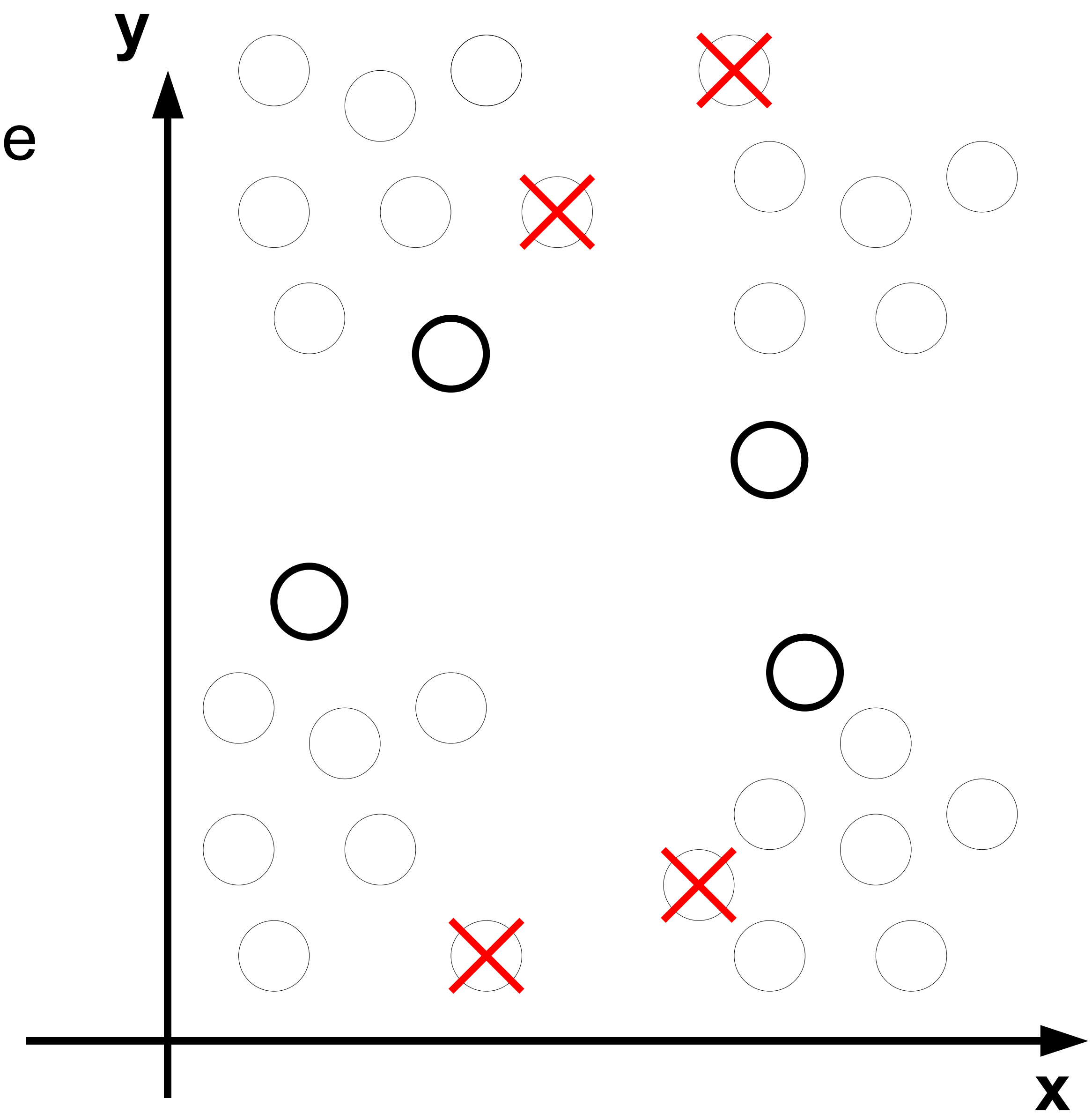
- $k = 2$



k – the number of clusters

Clustering stability

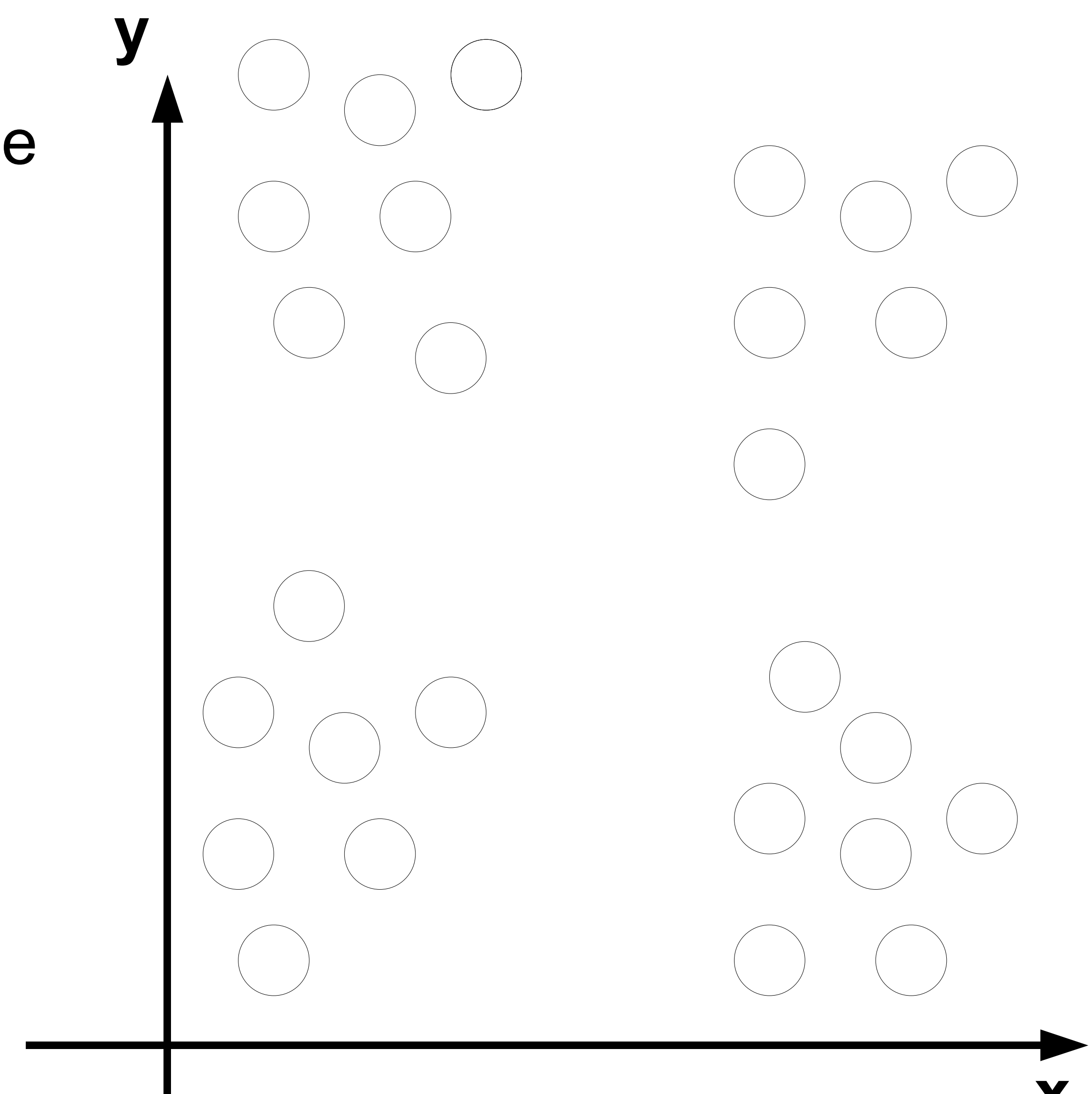
- $k = 2$
- Another sample from the same probability distribution..



k – the number of clusters

Clustering stability

- $k = 2$
- Another sample from the same probability distribution..

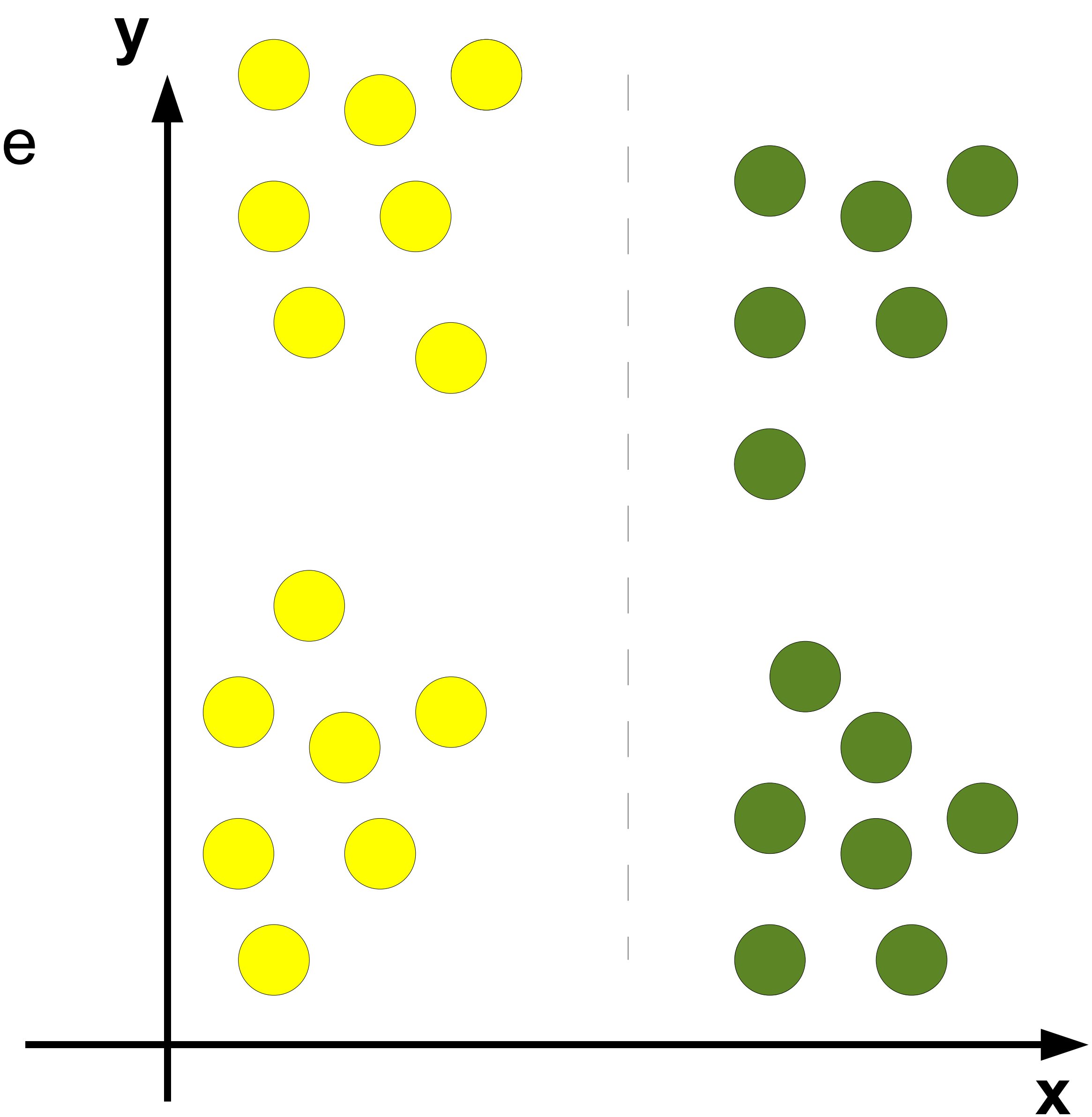


k – the number of clusters

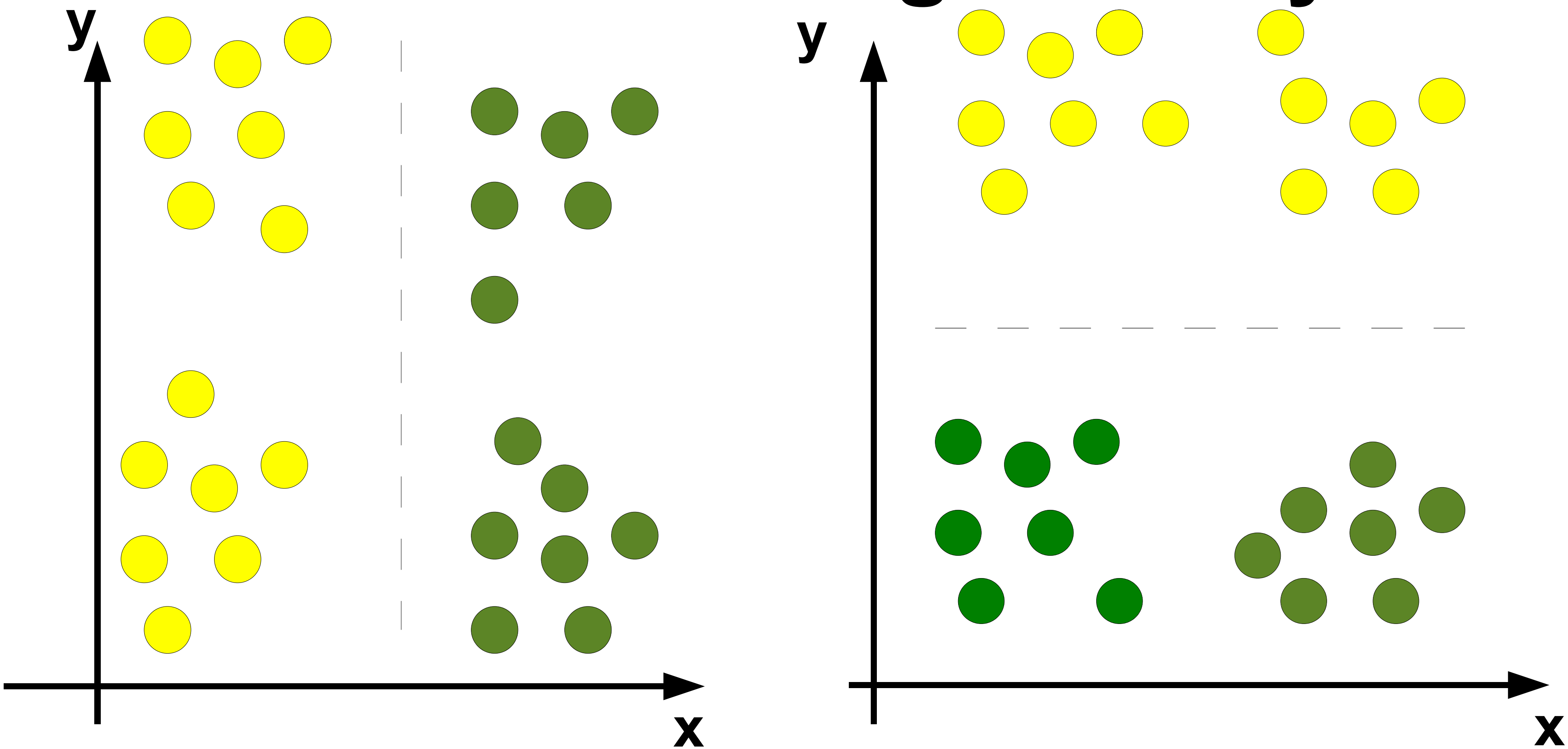
Clustering stability

- *k = 2*
- Another sample from the same probability distribution..
- **.. could result in a completely different clustering.**
- This is what we call clustering **instability**.

k – the number of clusters



What is **clustering stability**?



- Clustering algorithm is **stable** when – from one sample to another – the algorithm ends up with similar clusterings.
- Very different clusterings for different samples = **instability**.

Importance of clustering stability

- Provides a good metric for evaluating and comparing different clustering algorithms.
- Has been extensively used to guess a good number of clusters k.
- The latter, as the article suggests, is actually a mistake.

Why should stability help finding a good number of clusters *k*?

- When *k* is too large, the algorithm has to „randomly“ split several true clusters.
- When k is too small, the algorithm has to „randomly“ merge several true clusters.
- So... for bad *k* value... performance is unstable?
- **This is actually not true in general!**
- The reason: splitting or merging is not really random.

k – the number of clusters

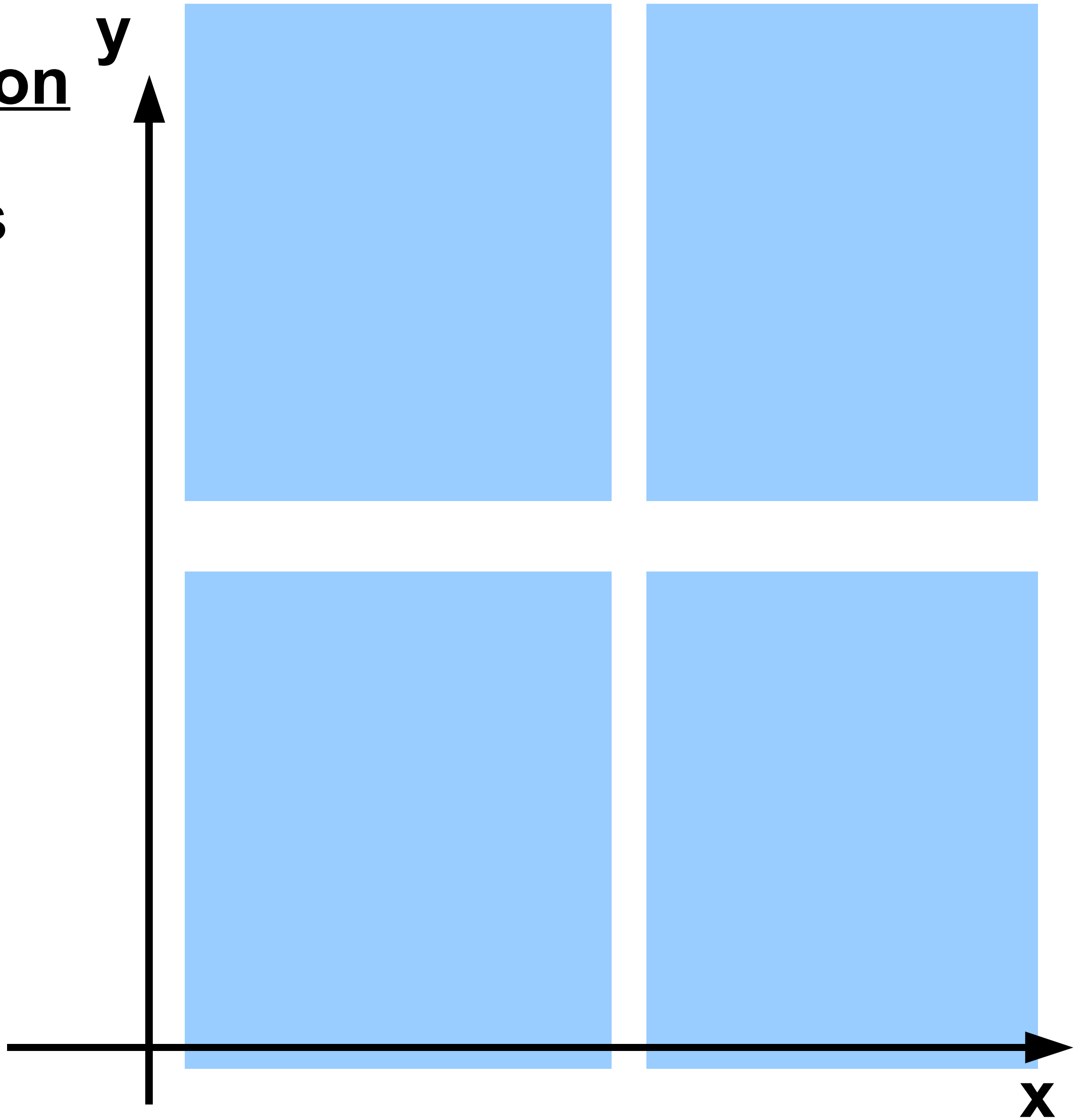
Using stability to find **k**:

instability from symmetry (the **desired** case)

The probability distribution

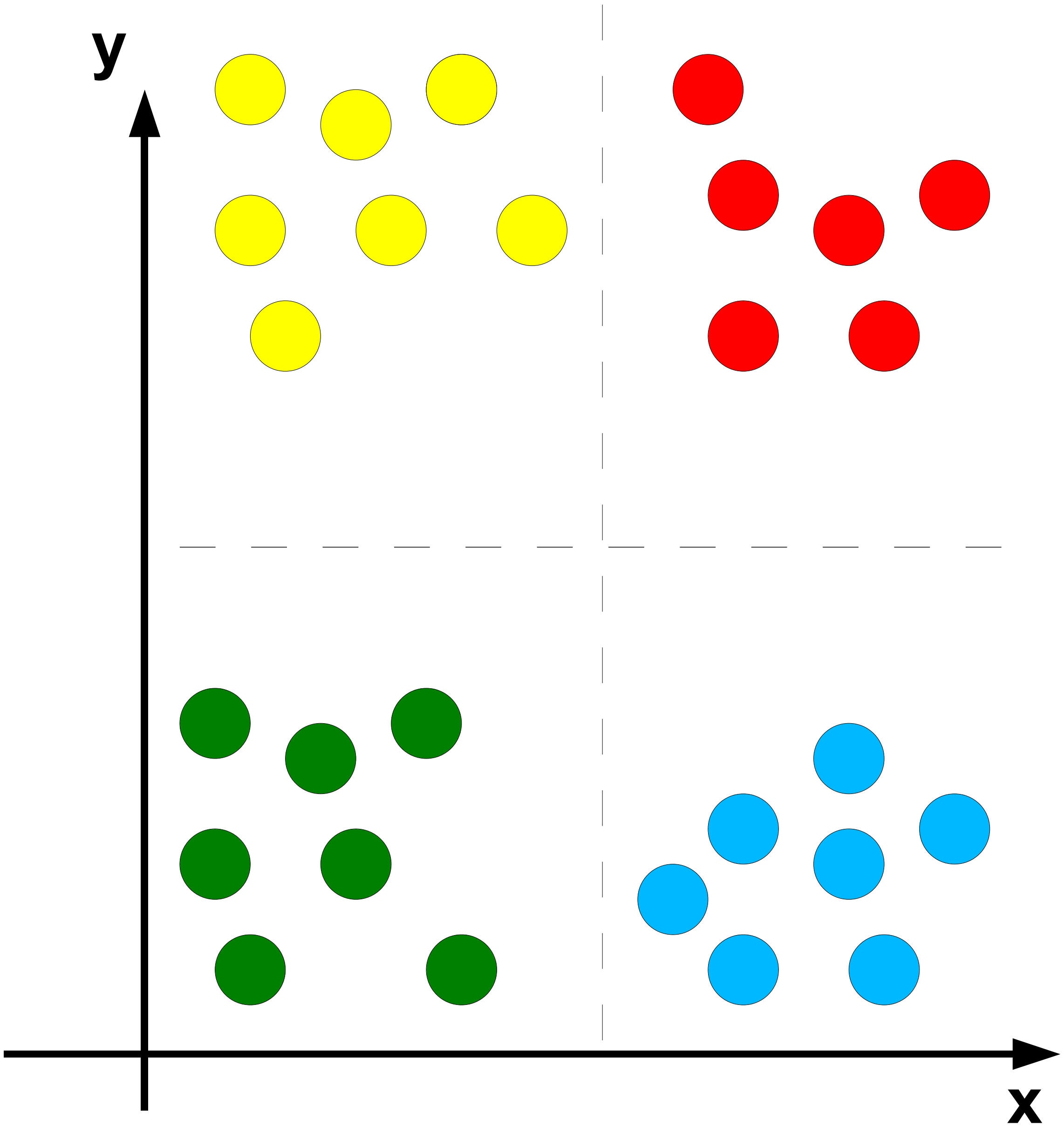
- Probability distribution is **symmetric**.

k – the number of clusters



Using stability to find **k:**
instability from symmetry (the **desired** case)

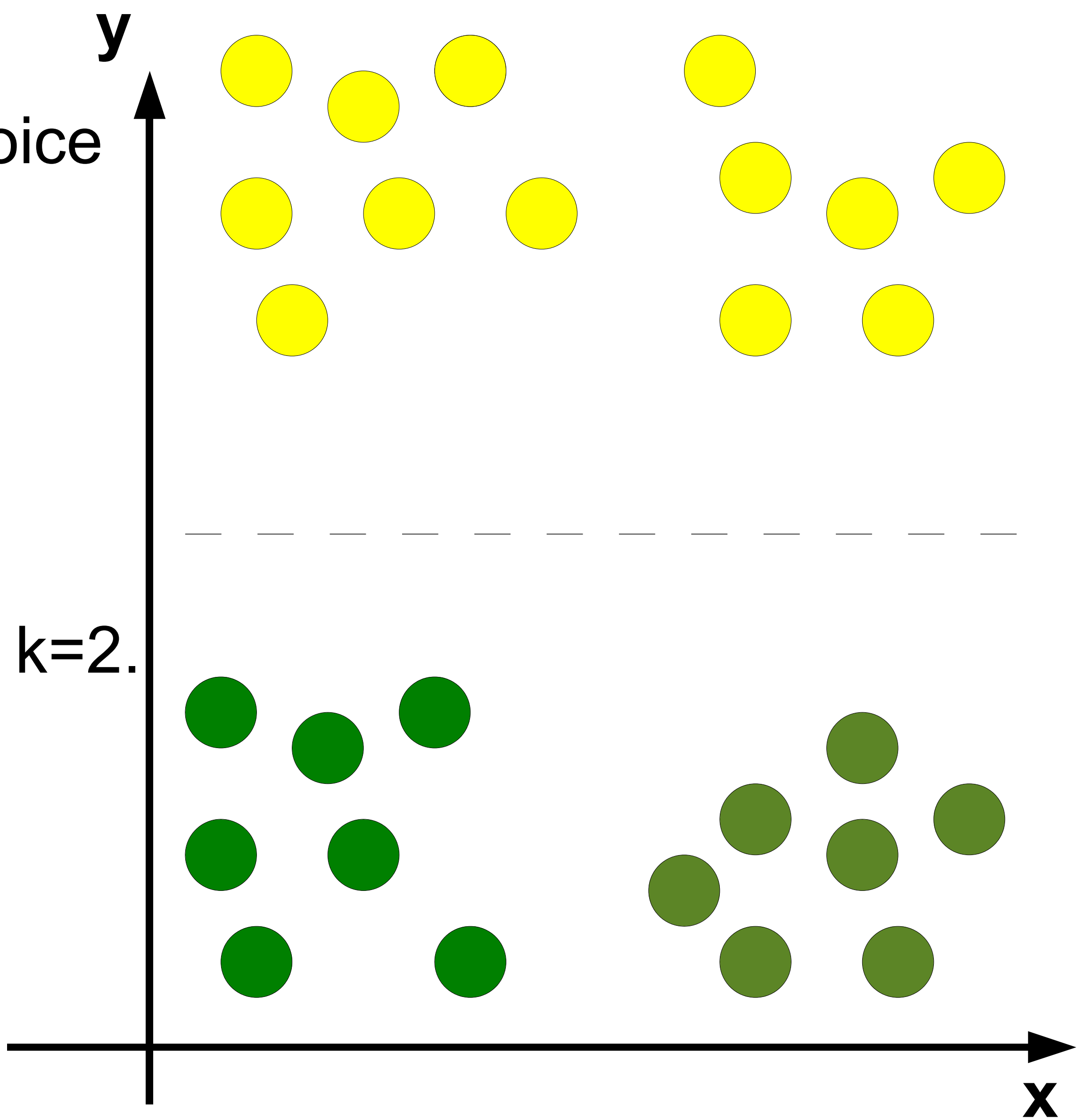
- Which *k* to use?
- Intuitively, k=4.



k – the number of clusters

Using stability to find **k:**
instability from symmetry (the **desired** case)

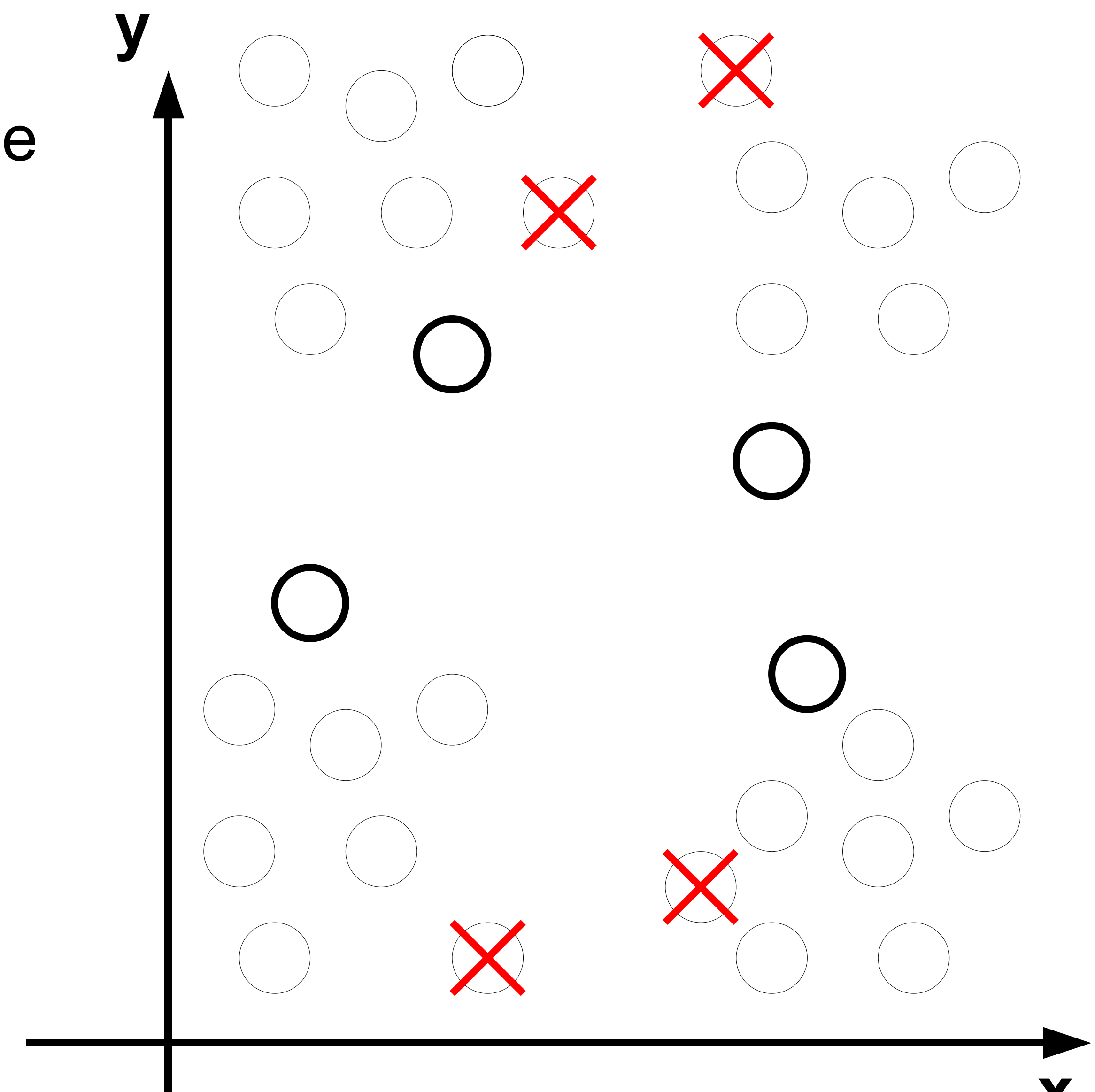
- Which *k* to use?
- **Assumption**: wrong choice of *k* results in unstable algorithm performance.
- => Trying to cluster with wrong k should result in instability.
- Example: instability with k=2.



k – the number of clusters

Using stability to find **k**:
instability from symmetry (the **desired** case)

- Which *k* to use?
- Another sample from the same probability distribution..

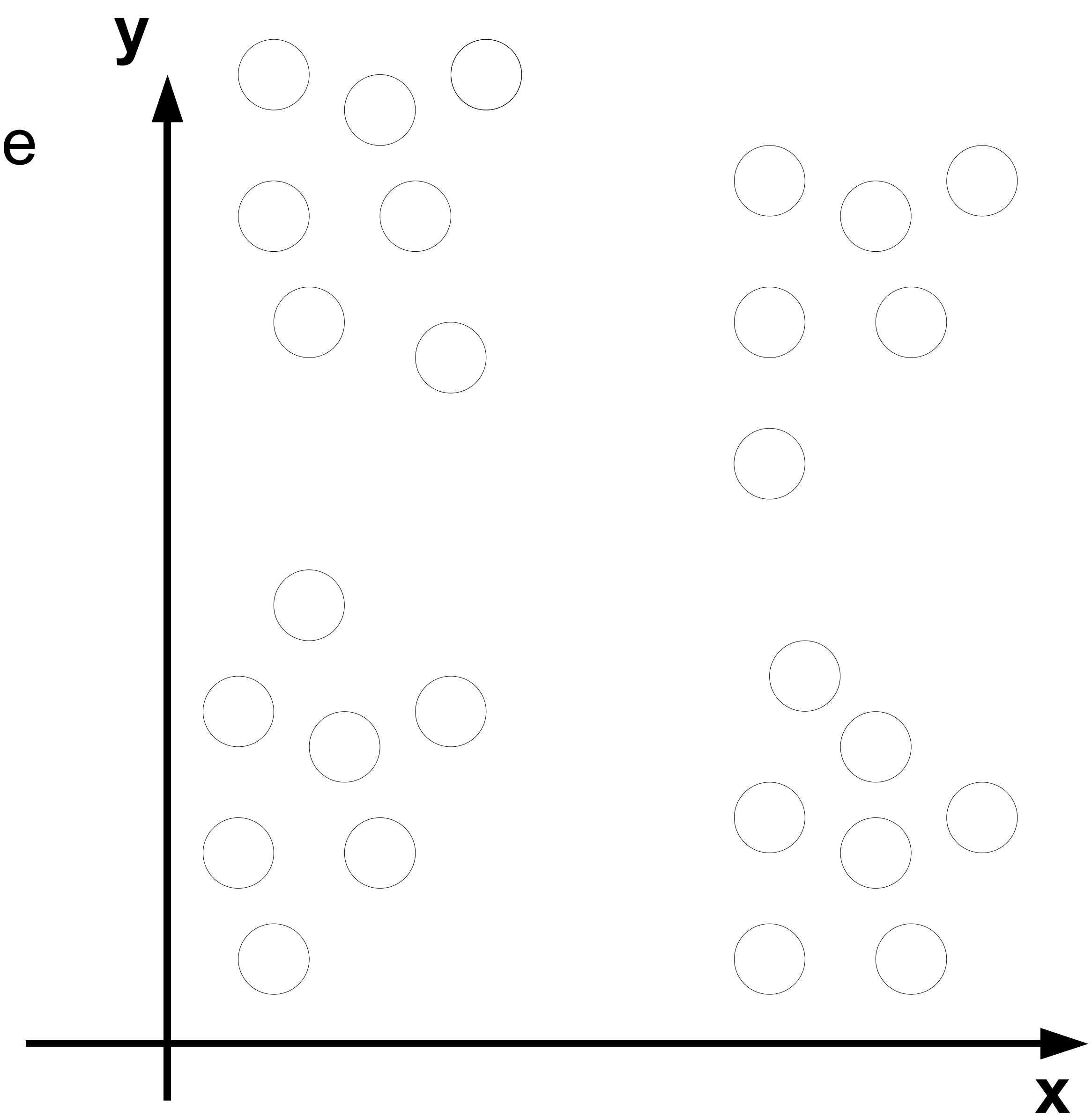


k – the number of clusters

Using stability to find **k:**

instability from symmetry (the **desired** case)

- Which *k* to use?
- Another sample from the same probability distribution..

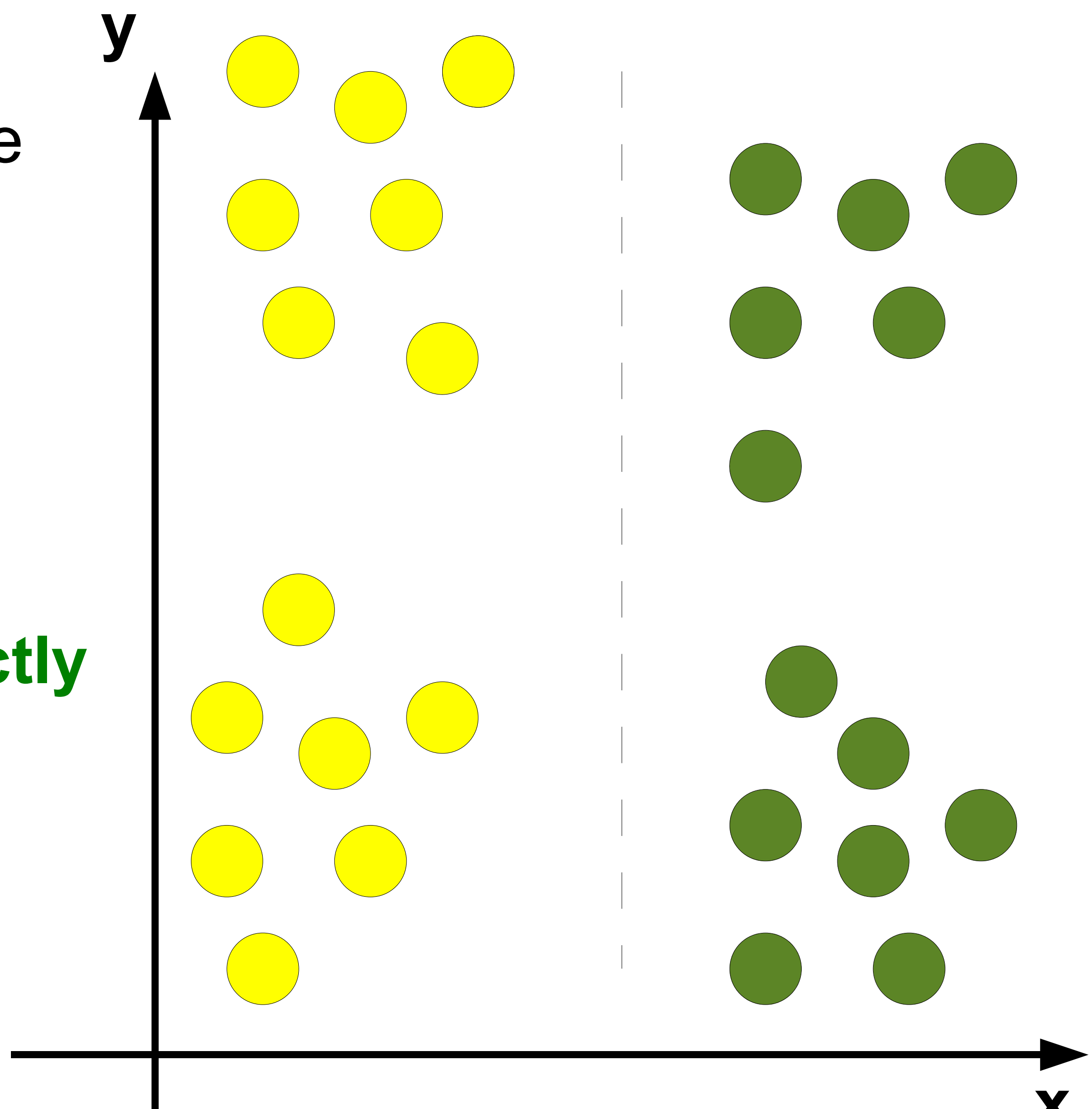


k – the number of clusters

Using stability to find **k**:

instability from symmetry (the **desired** case)

- Which *k* to use?
- Another sample from the same probability distribution..
- **.. could result in a completely different clustering.**
- **Here, instability correctly indicates that k=2 is a wrong choice.**



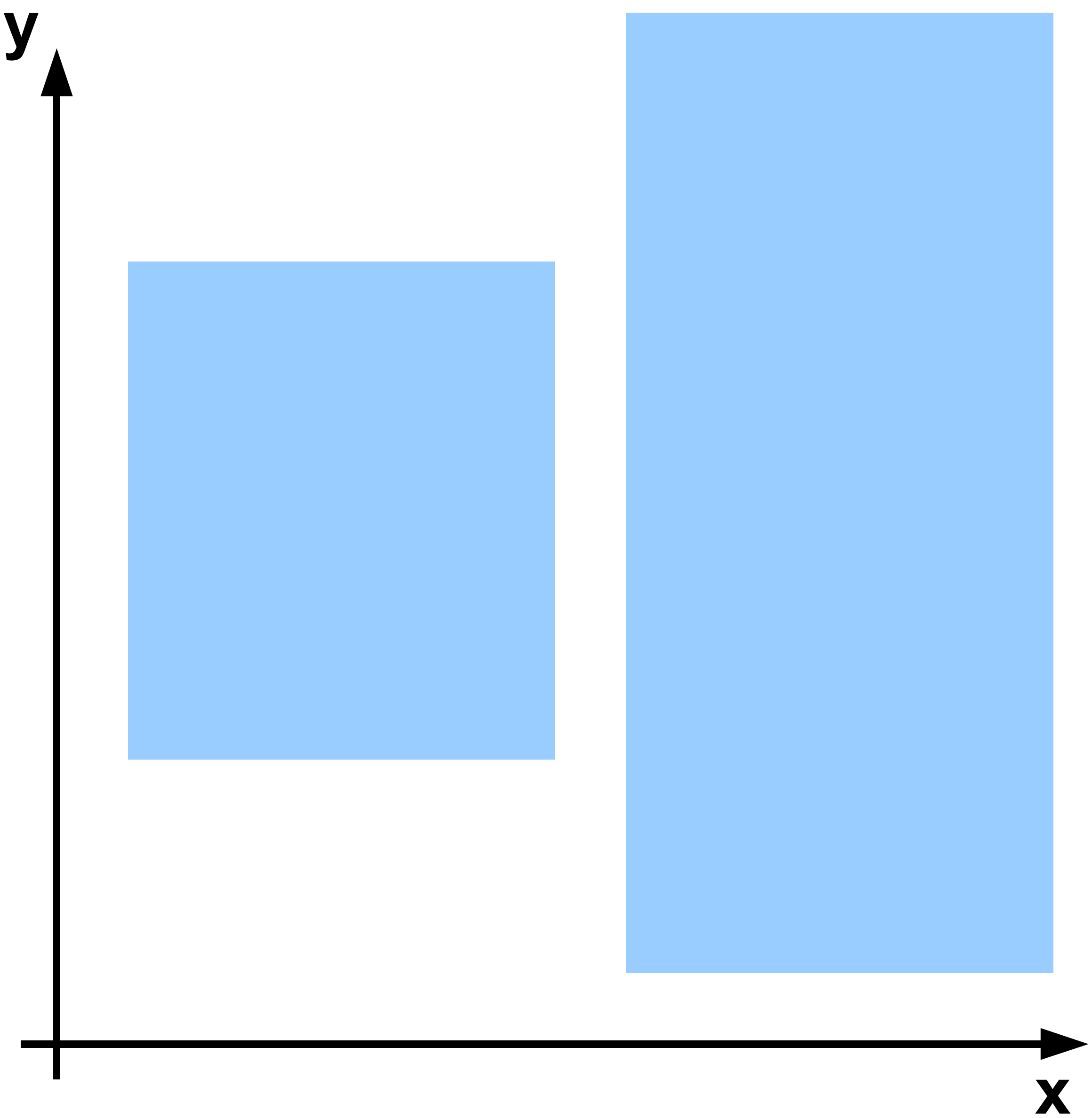
k – the number of clusters

Using stability to find **k:**

the **undesired** case: stable performance on bad *k*

The probability distribution

- non-symmetric probability distribution case.

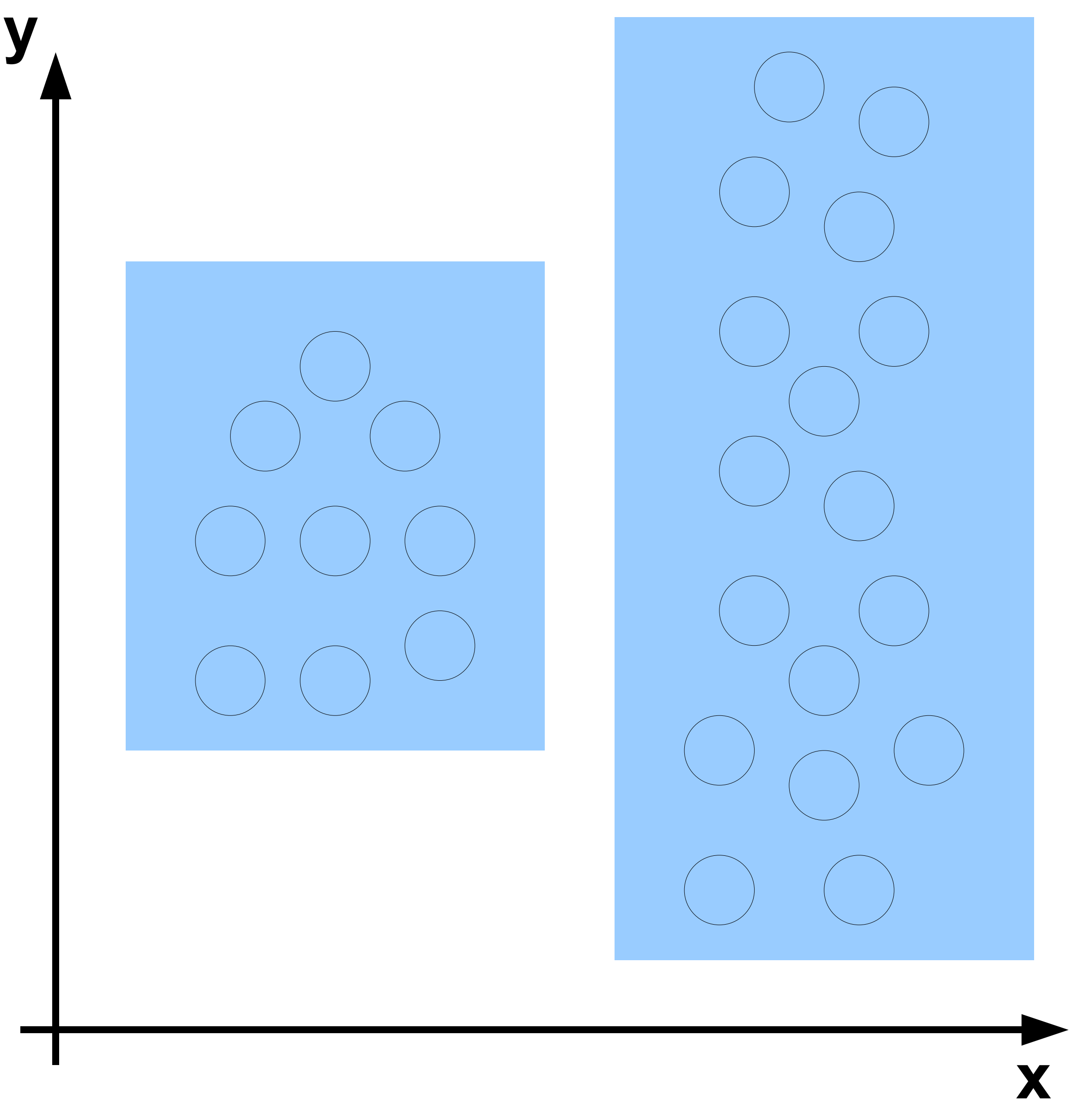


k – the number of clusters

Using stability to find **k:**

the <u>**undesired**</u> case: stable performance on bad *k*

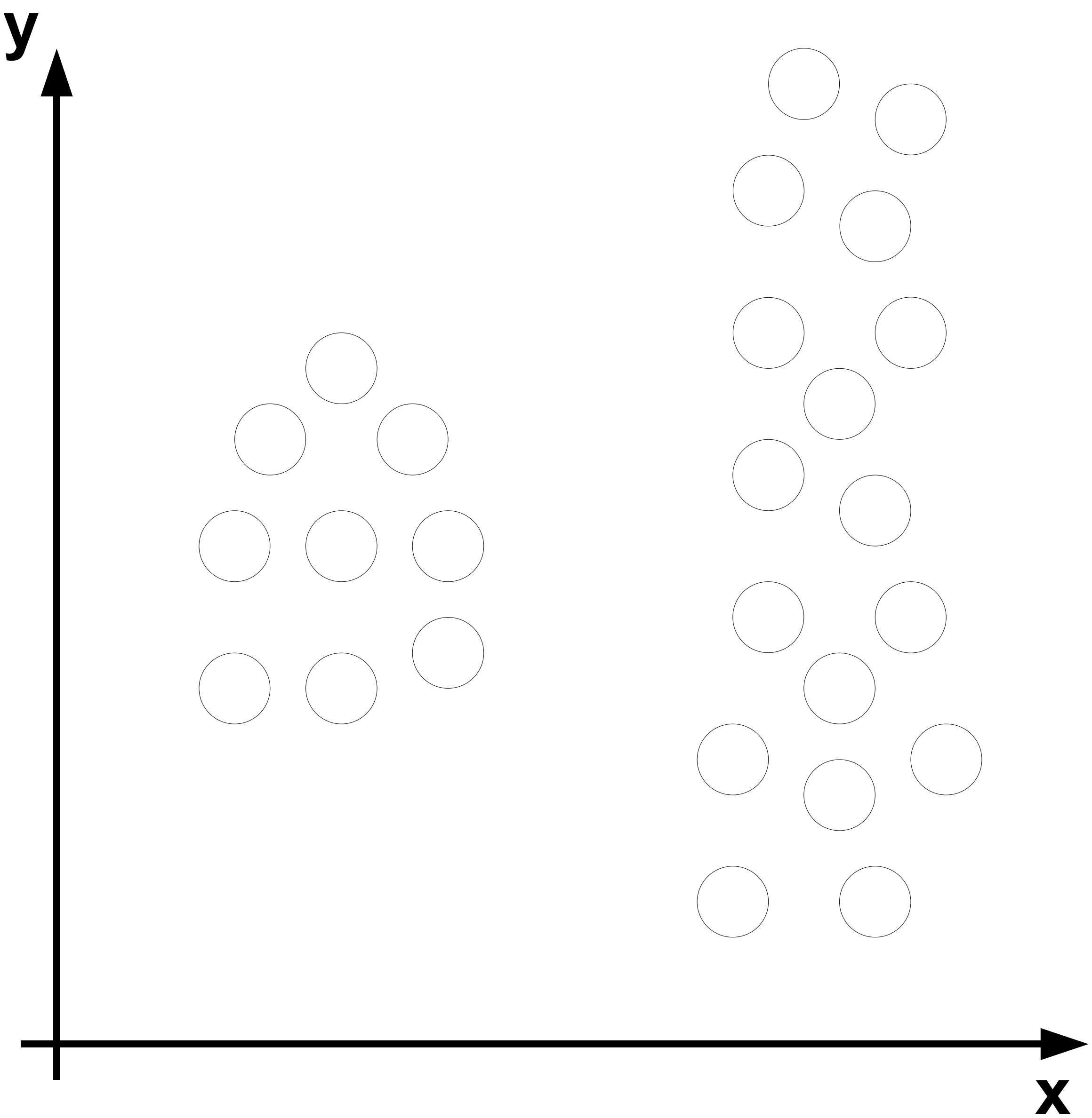
<u>The sample</u>



k – the number of clusters

Using stability to find **k:**

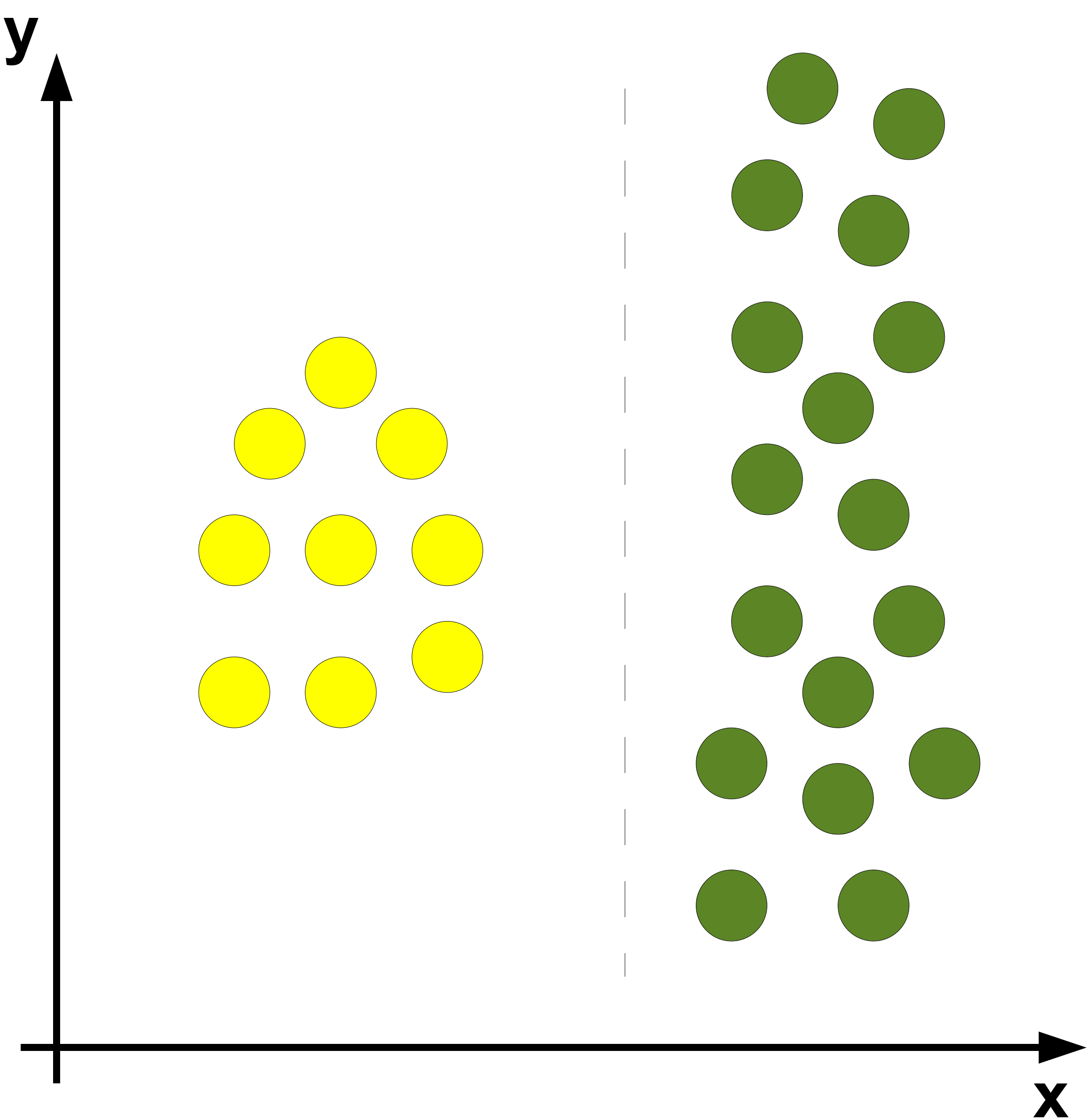
the **undesired** case: stable performance on bad *k*



k – the number of clusters

Using stability to find **k:**
the **undesired** case: stable performance on bad *k*

- The correct choice seems to be k=2.
- Empirically, k=3 doesn't seem to be a very good choice.
- If stability was to be a good indicator for detecting a good value for *k*, we would expect instability for k=3.

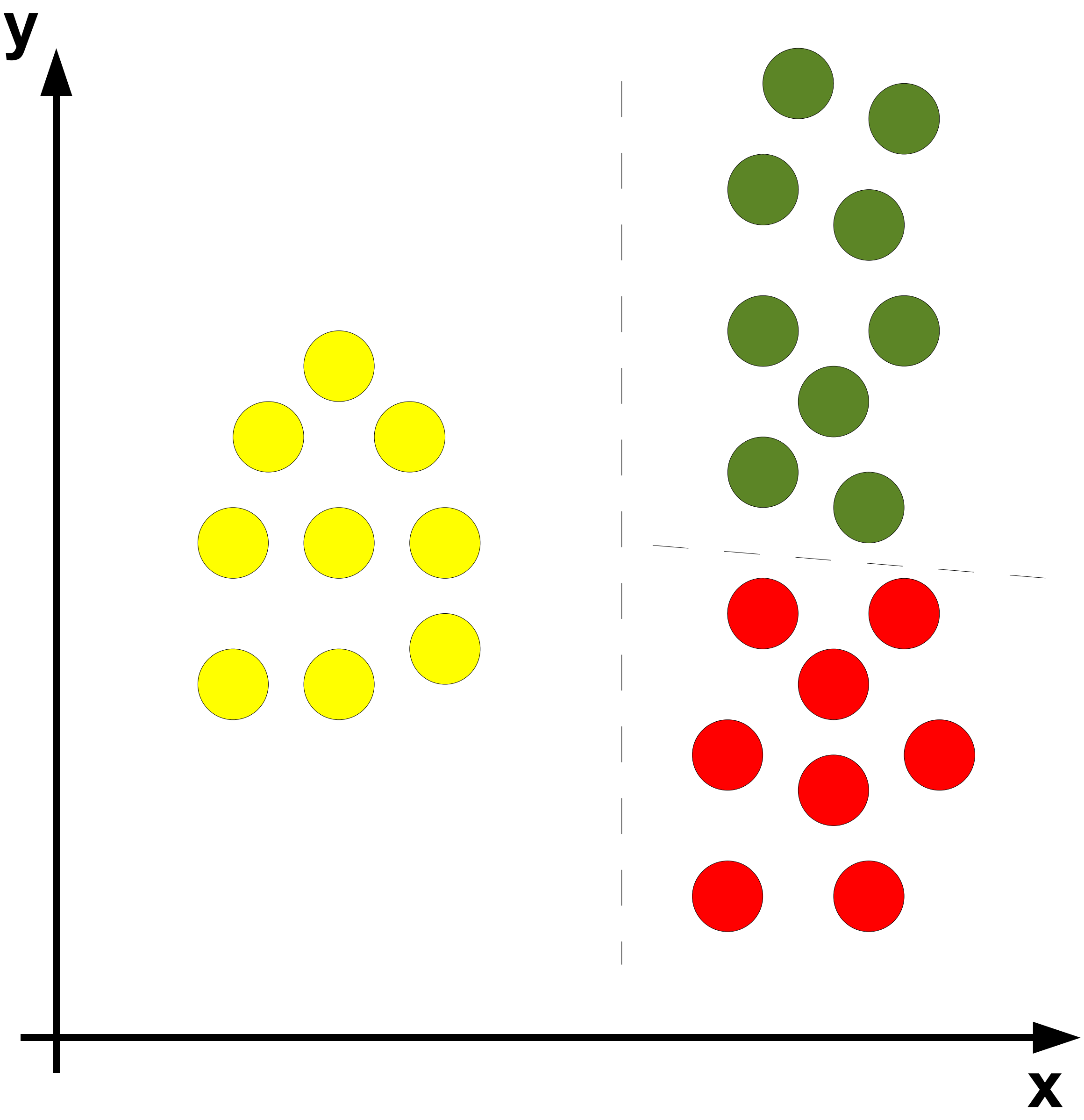


k – the number of clusters

Using stability to find **k:**

the **undesired** case: stable performance on bad *k*

Clustering with k=3



k – the number of clusters

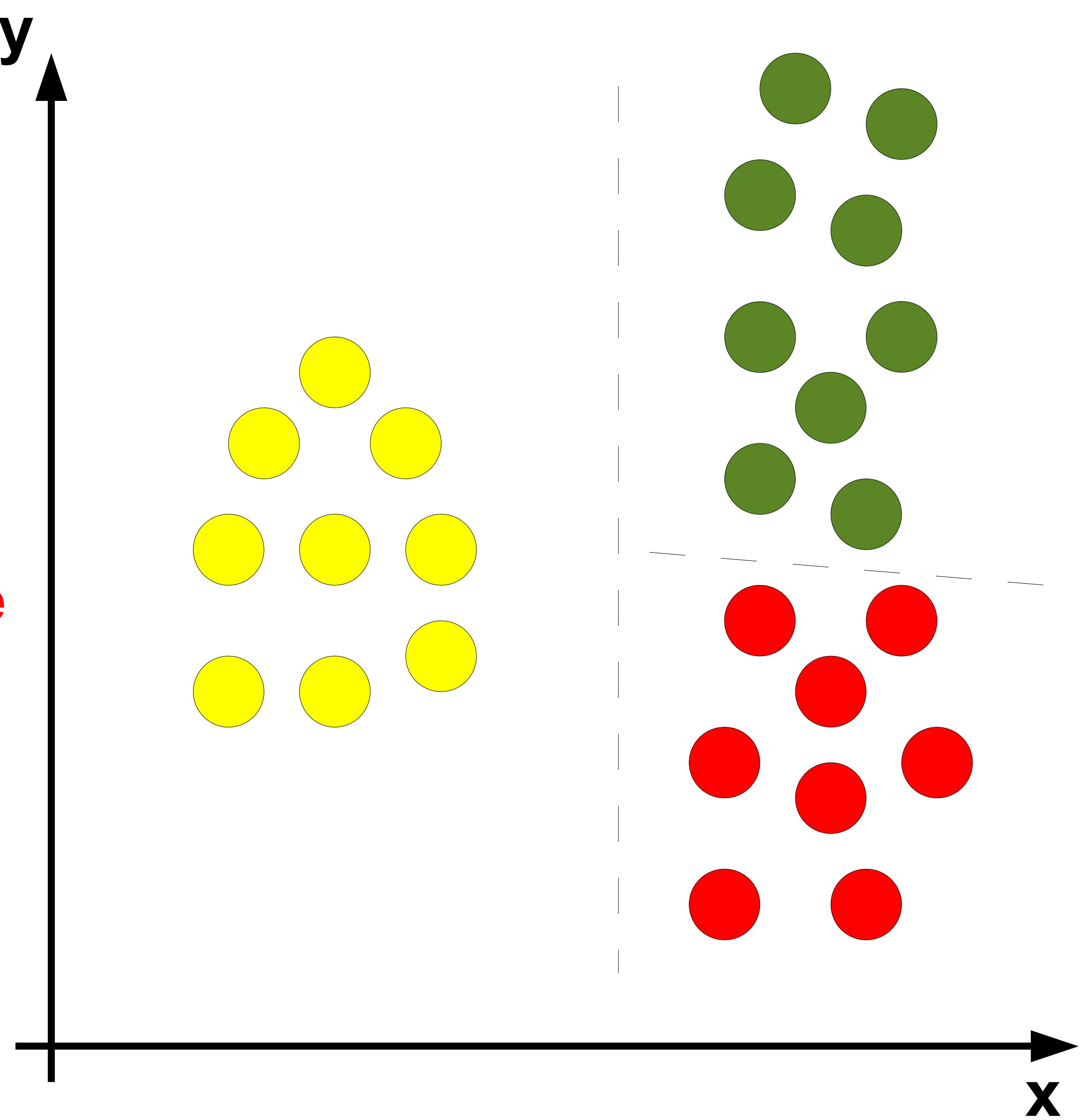
Using stability to find **k**:

the <u>**undesired**</u> case: stable performance on bad *k*

<u>**Clustering with k=3**</u>

- Here, all common algorithms would go breaking the larger group through-out different samples.
- **Here, stable performance suggests that k=3 would be a good choice, although it actually isn't.**

k – the number of clusters



=>

USING STABILITY TO FIND THE NUMBER OF CLUSTERS IS ABSOLUTELY UNRELIABLE

NOW, FORMALLY..

Clustering distance

- (<u>def</u>) Let ***P*** be a family of probability distributions over some domain **X**.
- Let ***S*** be a family of clusterings of **X**.
- A **clustering distance** is a function **d** : ***P*** x ***S*** x ***S*** → [0,1] that satisfying for any P ∈ ***P*** and C_1, C_2, C_3 ∈ ***S***:
  - $d_P(C_1, C_1) = 0$
  - $d_P(C_1, C_2) = d_P(C_2, C_1)$ *(symmetry)*
  - $d_P(C_1, C_3) \geq d_P(C_1, C_2) + d_P(C_2, C_3)$ *(triangle inequality)*

Hamming distance

- Defined as:

$$d_P(C_1,C_2) = \mathrm{Pr}_{x\sim P,\, y\sim P}\left[(x \sim_{C_1} y) \ \otimes (x \sim_{C_2} y)\right]$$

- ($\otimes$ denotes logical XOR)
- In English: probability of random points x and y being clustered differently in clusterings C_1 and C_2.
- Hamming distance satisfies all conditions for being a valid clustering distance measure.

Risk optimizing clustering algorithms

- Algorithms that choose the clustering by optimizing some risk function.
- The largest class of clustering algorithms.
- Examples:
  - all center-based algorithms (k-means, k-medians,..)
  - spectral clustering, if re-formulated a bit
- Explicitly define a „quality“ for any clustering.
- The following theorems apply to all ROCAs.

The **stability** theorem

- Let P be a probability distribution.
- Let C be a clustering on P.
- „If P has a unique minimizer C, then any R-minimizing clustering algorithm which is risk-converging is stable on P."
- (details & proof in the article)

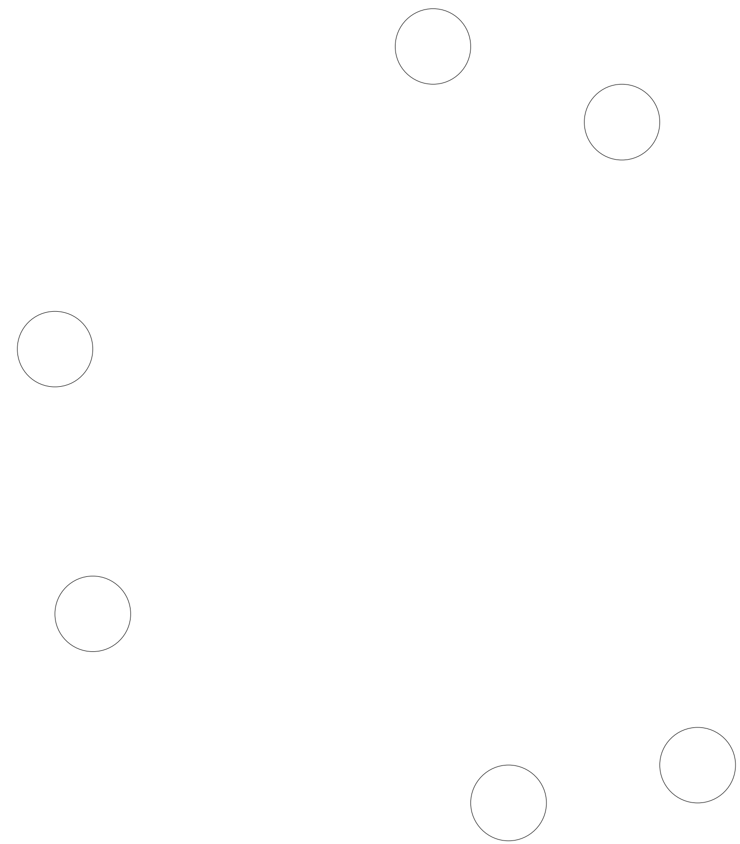
- [Example 1: the uniform distribution]

The **instability** theorem

- Let **R** be an ODD risk function.
- Let **d** be an ODD clustering distance measure.
- Let **P** be a probability distribution that has n distinct minimizers.
- Let **g** be a P-symmetry, so that for each minimizer the distance **d** from clustering **C** to symmetric clustering **g(C)** is strictly greater than zero.
- Then any R-minimizing algorithm which is convergent is unstable on **P**.
- (details & proof in the article)

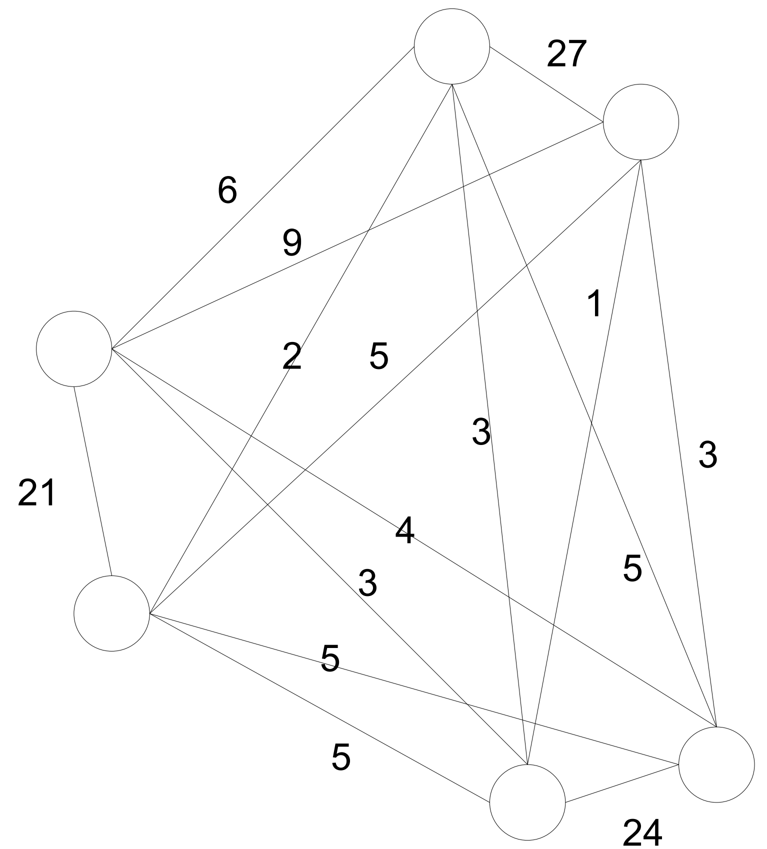
- [Example 2]

Spectral Clustering: the NCuts version



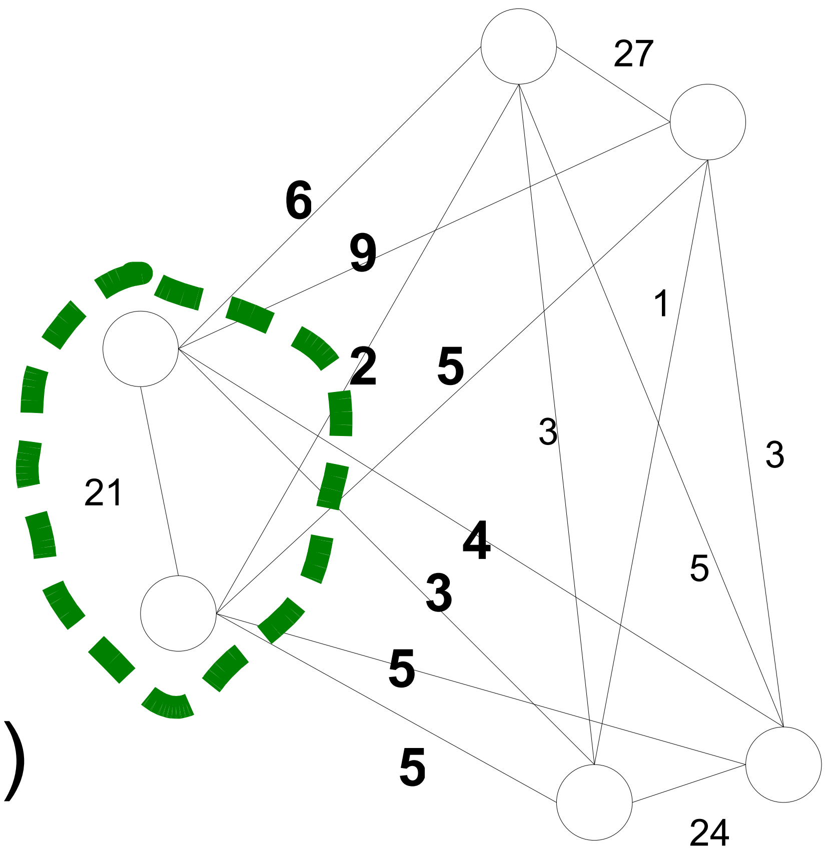
Spectral Clustering: the NCuts version

- STEP 1: Construction of a full graph between training data points. Weights on the edges are the similarity measure values from the kernel function.



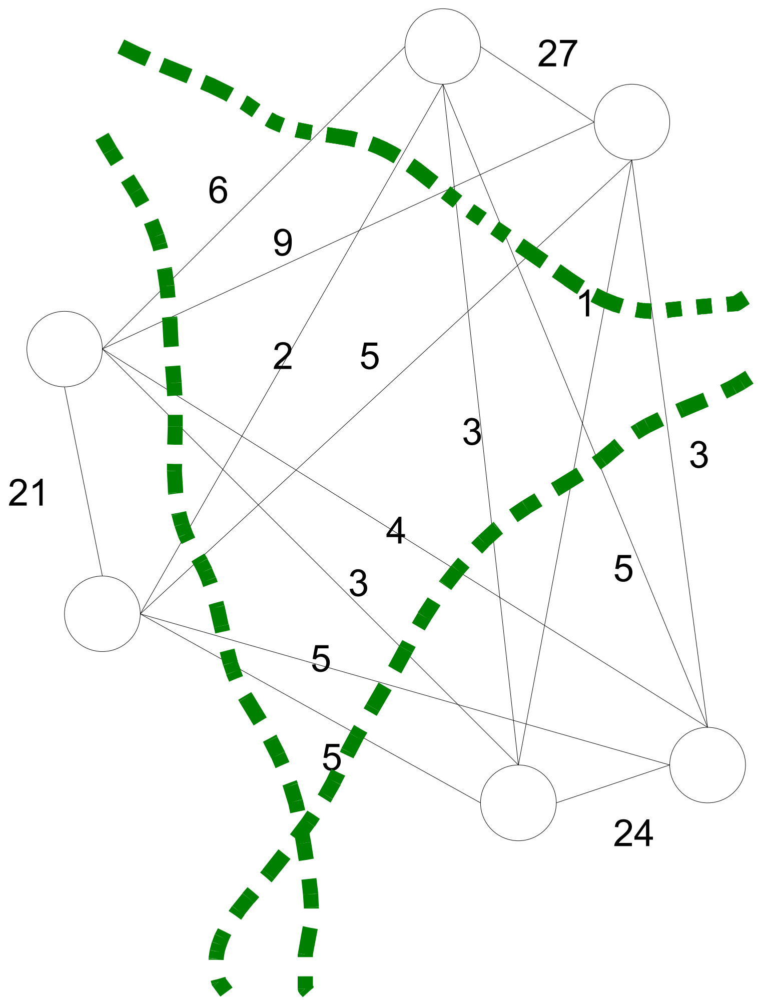
Spectral Clustering: the NCuts version

- Normalized cuts: minimize the sum of weights of edges that escape a cluster and normalize it relative to the total „weight“ of that cluster.
- (6+9+2+5+4+3+5+5) / (6+9+2+5+4+3+5+5+21)



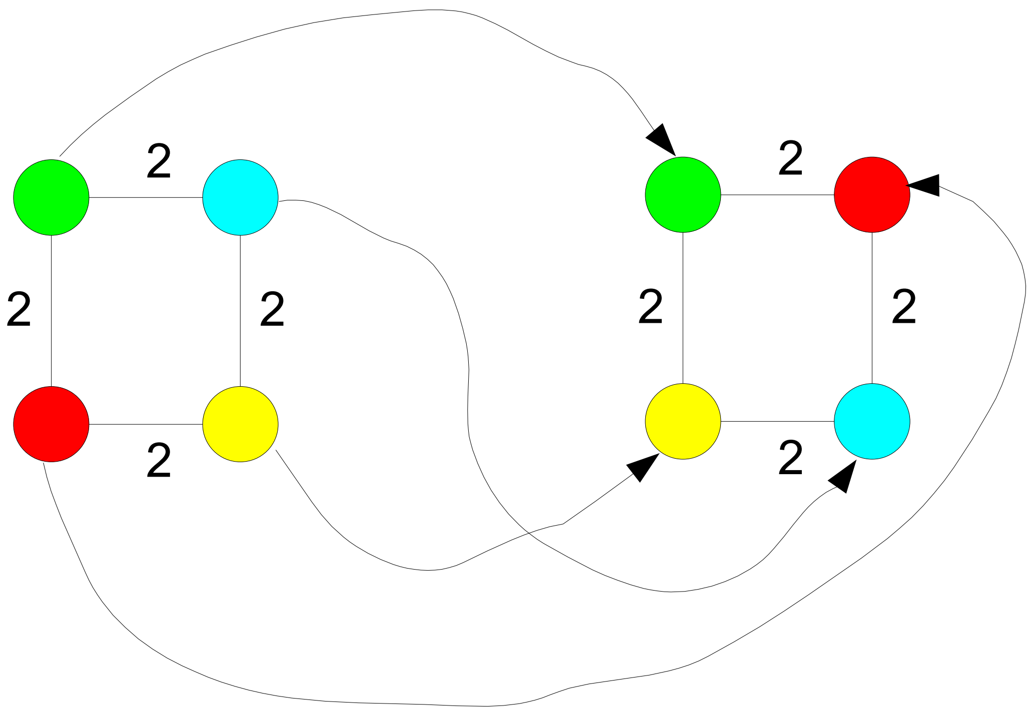
Spectral Clustering: the NCuts version

- STEP 2: Find the clustering that minimizes the normalized cuts.



Spectral Clustering: the NCuts version

- For a graphs, the **symmetry** is usually defined as the existance of non-trivial automorphisms.



- [Example 3: The C_4 graph]

Spectral Clustering: the NCuts version

- In real life and for large amounts of training data points, exact automorphisms are unlikely to occur.
- Graphs based on data from a symmetric probability distribution will be „nearly“ non-trivially automorphic.
- The uncertainty from the random sampling process will be enough to make the algorithm pick one clustering for one sample and another (possibly very different) clustering for another sample.

Conclusions

- The existence of a **unique minimizer** implies **stability**.
- The existence of a **symmetry** permuting such minimizers implies **instability**.
- Real world data rarely has a symmetric probability distribution.
- When searching for a meaningful number of clusters on the basis of stability, success should be viewed as an exception rather than a rule.