Flat Clustering

September 6, 2023

Overview

Outline

1. Recap
2. Clustering: Introduction
3. Clustering in IR
4. *K*-means
5. Evaluation
6. How many clusters?

Take-away today

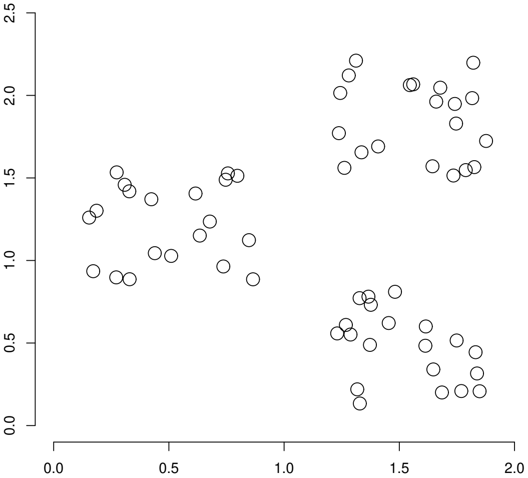
- What is clustering?
- Applications of clustering in information retrieval
- K-means algorithm
- Evaluation of clustering
- How many clusters?

Outline

Clustering: Definition

- (Document) clustering is the process of grouping a set of documents into clusters of similar documents.
- Documents within a cluster should be similar.
- Documents from different clusters should be dissimilar.
- Clustering is the most common form of unsupervised learning.
- Unsupervised = there are no labeled or annotated data.

Data set with clear cluster structure



Propose algorithm for finding the cluster structure in this example

Classification vs. Clustering

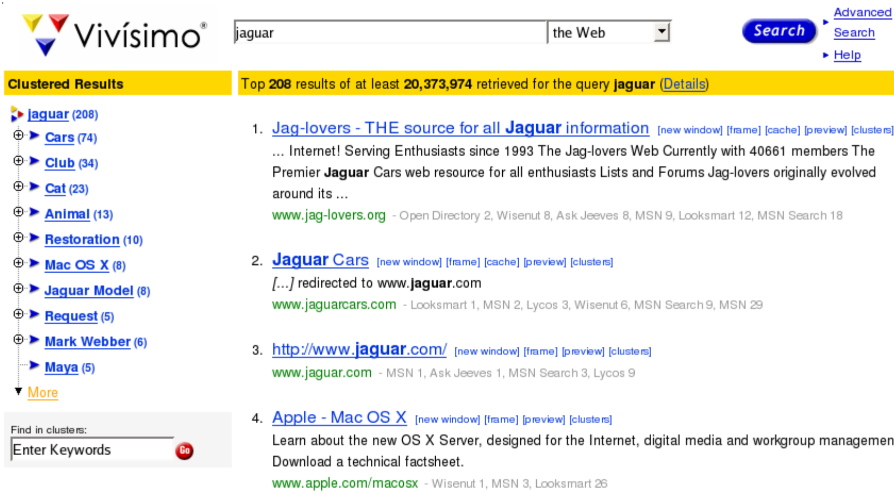
- Classification: supervised learning
- Clustering: unsupervised learning
- Classification: Classes are human-defined and part of the input to the learning algorithm.
- Clustering: Clusters are inferred from the data without human input.
- Many ways of influencing the outcome of clustering:
    - number of clusters,
    - similarity measure,
    - representation of documents,
    - ...

Outline

Applications of clustering in IR

application	what is clustered?	benefit
search result clustering	search results	more effective information presentation to user
Scatter-Gather	(subsets of) collection	alternative user interface: “search without typing”
collection clustering	collection	effective information presentation for exploratory browsing
cluster-based retrieval	collection	higher efficiency: faster search

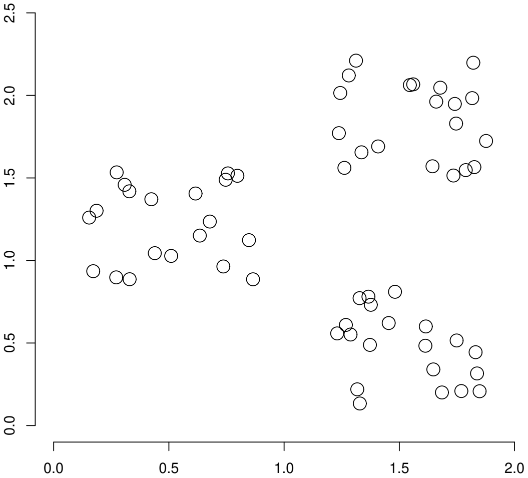
Search result clustering for better navigation



Clustering for improving recall

- To improve search recall:
  - Cluster docs in collection a priori
  - When a query matches a doc d, also return other docs in the cluster containing d
- Hope: if we do this: the query "car" will also return docs containing "automobile"
  - Because the clustering algorithm groups together docs containing "car" with those containing "automobile".
  - Both types of documents contain words like "parts", "dealer", "mercedes", "road trip".

Data set with clear cluster structure



Propose algorithm for finding the cluster structure in this example

Desiderata for clustering

- General goal: put related docs in the same cluster, put unrelated docs in different clusters.
  - We'll see different ways of formalizing this.
- The number of clusters should be appropriate for the data set we are clustering.
  - Initially, we will assume the number of clusters K is given.
  - Later: Semiautomatic methods for determining K
- Secondary goals in clustering
  - Avoid very small and very large clusters
  - Define clusters that are easy to explain to the user
  - Many others ...

Flat vs. Hierarchical clustering

- Flat algorithms
  - Usually start with a random (partial) partitioning of docs into groups
  - Refine iteratively
  - Algorithm: K-means
- Hierarchical algorithms
  - Create a hierarchy
  - Bottom-up, agglomerative
  - Top-down, divisive

Hard vs. Soft clustering

- Hard clustering: Each document belongs to exactly one cluster.
  - More common and easier to do
- Soft clustering: A document can belong to more than one cluster.
  - Makes more sense for applications like creating browsable hierarchies
  - You may want to put *sneakers* in two clusters:
    - sports apparel
    - shoes
  - You can only do that with a soft clustering approach.

Flat algorithms

- Flat algorithms compute a partition of N documents into a set of K clusters.
- Given: a set of documents and the number K
- Find: a partition into K clusters that optimizes the chosen partitioning criterion
- Global optimization: exhaustively enumerate partitions, pick optimal one
  - Not tractable
- Effective heuristic method: K-means algorithm

Outline

K-means

- Perhaps the best known clustering algorithm
- Simple, works well in many cases
- Use as default / baseline for clustering documents

Document representations in clustering

- Vector space model
- doc2vec
- relatedness between vectors can be measured by cosine similarity, etc.

K-means: Basic idea

- Each cluster in *K*-means is defined by a centroid.
- Objective/partitioning criterion: minimize the average squared difference from the centroid
- Recall definition of centroid:

$$\vec{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\vec{x} \in \omega} \vec{x}$$

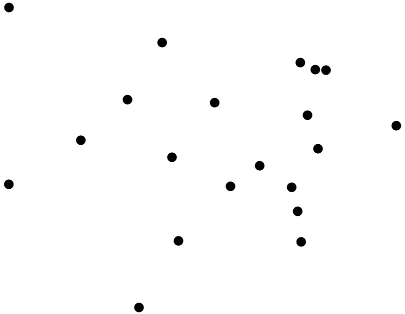
  where we use ω to denote a cluster.
- We try to find the minimum average squared difference by iterating two steps:
  - reassignment: assign each vector to its closest centroid
  - recomputation: recompute each centroid as the average of the vectors that were assigned to it in reassignment

K-means pseudocode (μ_k is centroid of ω_k)

K-MEANS$(\{\vec{x}_1, \ldots, \vec{x}_N\}, K)$

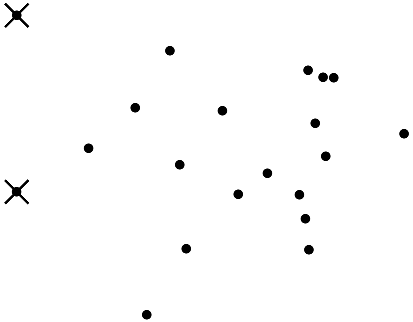
$(\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_K) \leftarrow$ SELECTRANDOMSEEDS$(\{\vec{x}_1, \ldots, \vec{x}_N\}, K)$
for $k \leftarrow 1$ **to** K
do $\vec{\mu}_k \leftarrow \vec{s}_k$
while stopping criterion has not been met
do for $k \leftarrow 1$ **to** K
    do $\omega_k \leftarrow \{\}$
    for $n \leftarrow 1$ **to** N
    do $j \leftarrow \arg\min_{j'} |\vec{\mu}_{j'} - \vec{x}_n|$
        $\omega_j \leftarrow \omega_j \cup \{\vec{x}_n\}$ *(reassignment of vectors)*
    for $k \leftarrow 1$ **to** K
    do $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$ *(recomputation of centroids)*
return $\{\vec{\mu}_1, \ldots, \vec{\mu}_K\}$

Worked Example: Set of points to be clustered

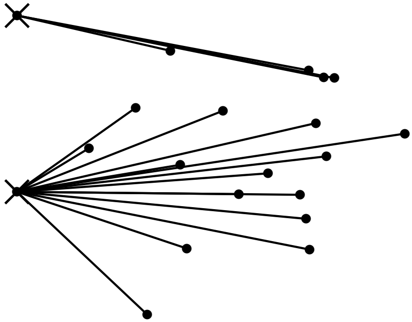


- what are the two clusters?
- compute the centroids of the clusters

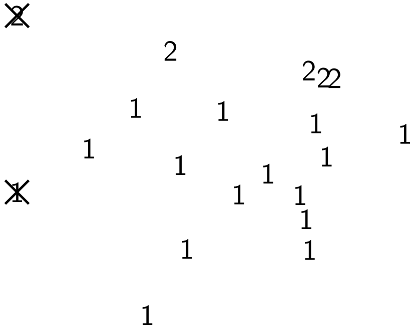
Worked Example: Random selection of initial centroids



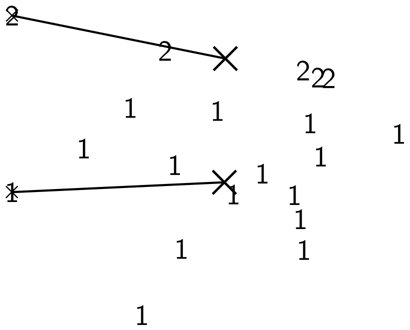
Worked Example: Assign points to closest center



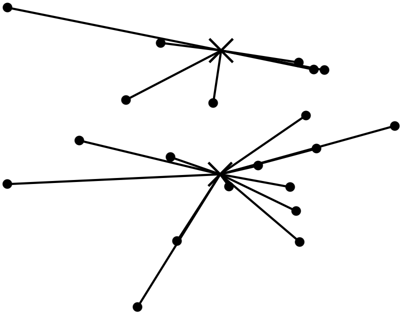
Worked Example: Assignment



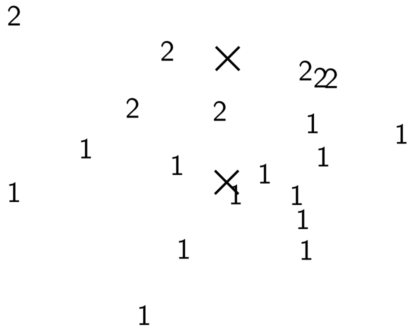
Worked Example: Recompute cluster centroids



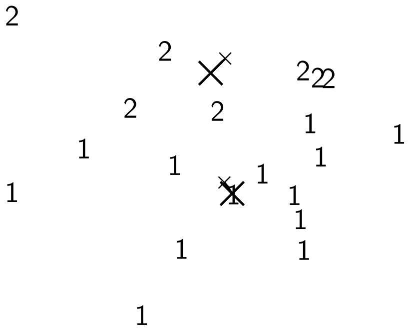
Worked Example: Assign points to closest centroid



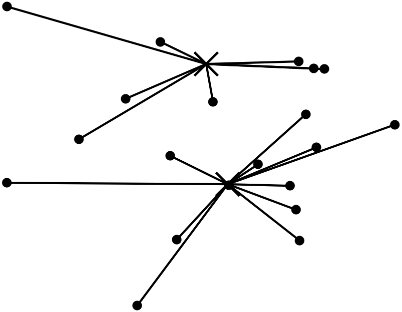
Worked Example: Assignment



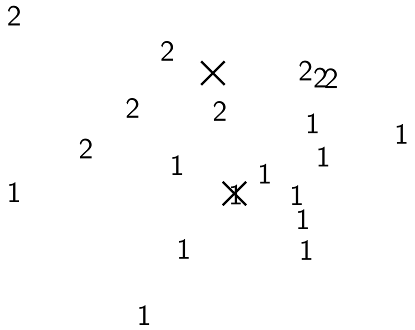
Worked Example: Recompute cluster centroids



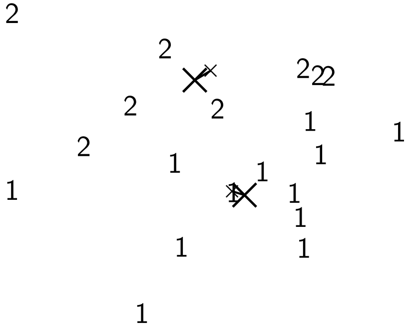
Worked Example: Assign points to closest centroid



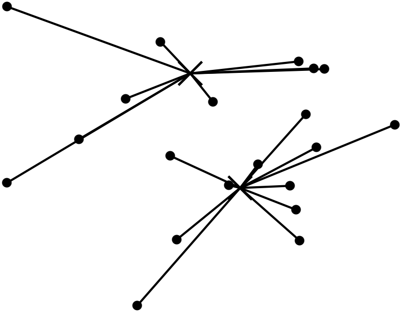
Worked Example: Assignment



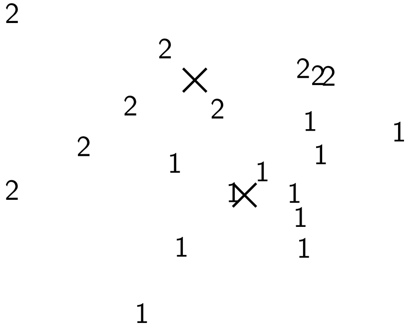
Worked Example: Recompute cluster centroids



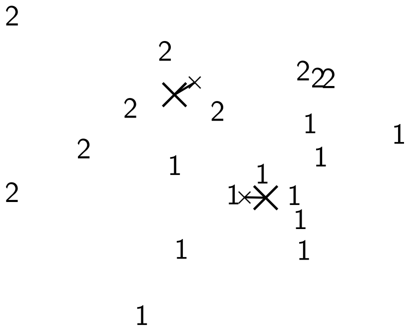
Worked Example: Assign points to closest centroid



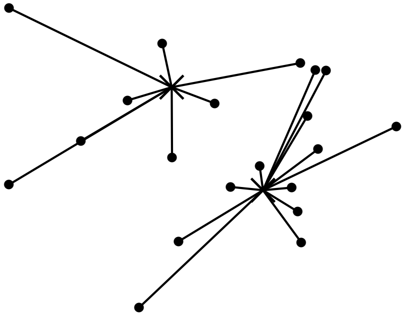
Worked Example: Assignment



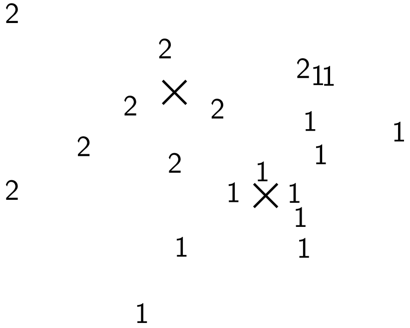
Worked Example: Recompute cluster centroids



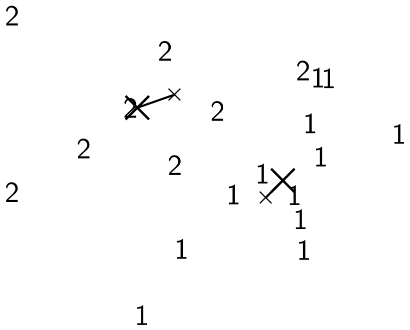
Worked Example: Assign points to closest centroid



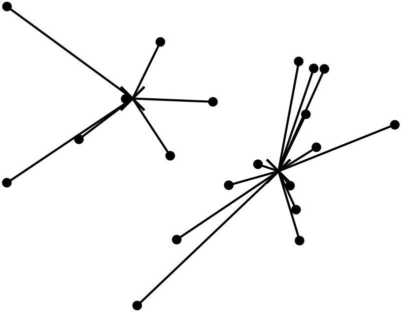
Worked Example: Assignment



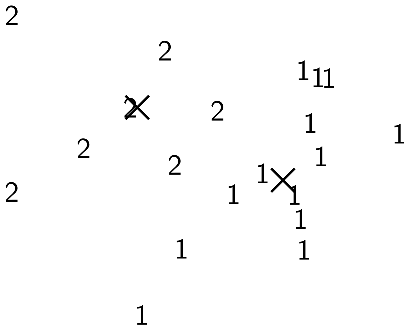
Worked Example: Recompute cluster centroids



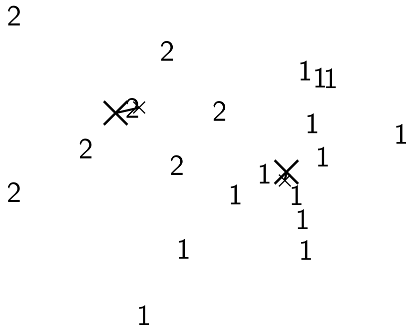
Worked Example: Assign points to closest centroid



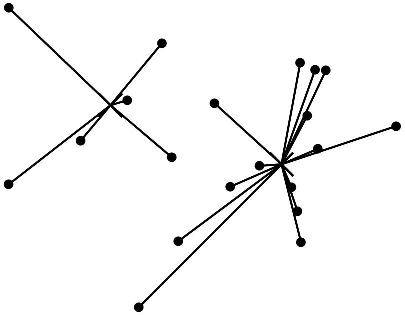
Worked Example: Assignment



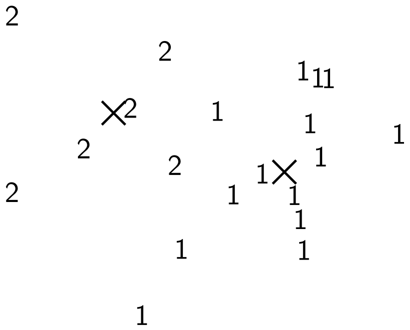
Worked Example: Recompute cluster centroids



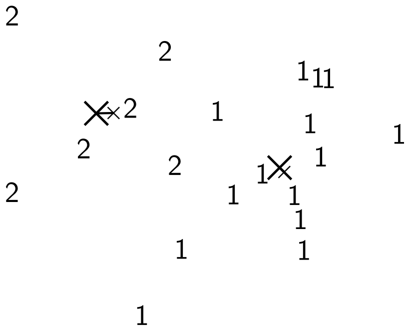
Worked Example: Assign points to closest centroid



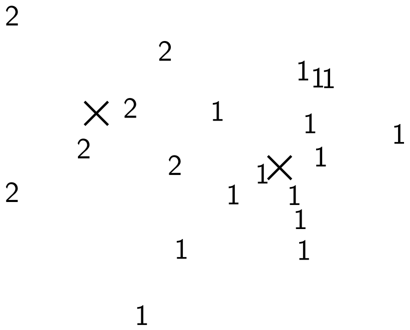
Worked Example: Assignment



Worked Example: Recompute cluster centroids



Worked Ex.: Centroids and assignments after convergence



K-means is guaranteed to converge: Proof

- RSS = sum of all squared distances between document vector and closest centroid
- RSS decreases during each reassignment step.
  - because each vector is moved to a closer centroid
- RSS decreases during each recomputation step.
  - see next slide
- There is only a finite number of clusterings.
- Thus: We must reach a fixed point.
- Assumption: Ties are broken consistently.
- Finite set & monotonically decreasing $\rightarrow$ convergence

Re-computation decreases average distance

$\text{RSS} = \sum_{k=1}^{K} \text{RSS}_k$ – the residual sum of squares (the "goodness" measure)

$$\begin{aligned} \text{RSS}_k(\vec{v}) &= \sum_{\vec{x}\in\omega_k} \|\vec{v}-\vec{x}\|^2 = \sum_{\vec{x}\in\omega_k}\sum_{m=1}^{M}(v_m - x_m)^2 \\ \frac{\partial \text{RSS}_k(\vec{v})}{\partial v_m} &= \sum_{\vec{x}\in\omega_k} 2(v_m - x_m) = 0 \end{aligned}$$

$$v_m = \frac{1}{|\omega_k|}\sum_{\vec{x}\in\omega_k} x_m$$

The last line is the componentwise definition of the centroid! We minimize RSS_k when the old centroid is replaced with the new centroid. RSS, the sum of the RSS_k, must then also decrease during recomputation.

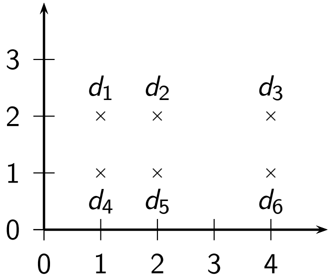
K-means is guaranteed to converge

- But we don't know how long convergence will take!
- If we don't care about a few docs switching back and forth, then convergence is usually fast ($<$ 10-20 iterations).
- However, complete convergence can take many more iterations.

Optimality of K-means

- Convergence $\neq$ optimality
- Convergence does not mean that we converge to the optimal clustering!
- This is the great weakness of K-means.
- If we start with a bad set of seeds, the resulting clustering can be horrible.

Exercise: Suboptimal clustering



- What is the optimal clustering for $K = 2$?
- Do we converge on this clustering for arbitrary seeds d_i, d_j?

Initialization of K-means

- Random seed selection is just one of many ways K-means can be initialized.
- Random seed selection is not very robust: It's easy to get a suboptimal clustering.
- Better ways of computing initial centroids:
  - Select seeds not randomly, but using some heuristic (e.g., filter out outliers or find a set of seeds that has "good coverage" of the document space)
  - Use hierarchical clustering to find good seeds
  - Select i (e.g., $i = 10$) different random sets of seeds, do a K-means clustering for each, select the clustering with lowest RSS

Time complexity of *K*-means

- Computing one distance of two vectors is $O(M)$.
- Reassignment step: $O(KNM)$ (we need to compute KN document-centroid distances)
- Recomputation step: $O(NM)$ (we need to add each of the document's $< M$ values to one of the centroids)
- Assume number of iterations bounded by I
- Overall complexity: $O(IKNM)$ – linear in all important dimensions

M: Vector length; N: Number of documents.

Outline

What is a good clustering?

- Internal criteria
  - RSS
  - Modularity in graph
- But an internal criterion often does not evaluate the actual utility of a clustering in the application.
- Alternative: External criteria
  - Evaluate with respect to a human-defined clustering

External criteria for clustering quality

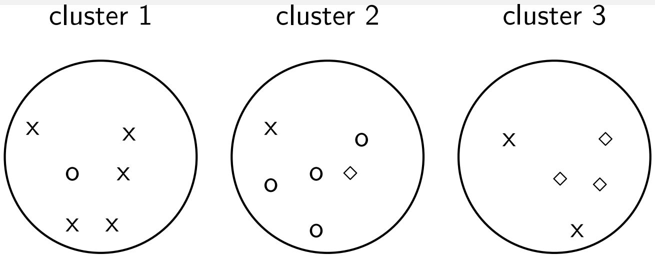
- Based on a gold standard data set, e.g., the Reuters collection we also used for the evaluation of classification
- Goal: Clustering should reproduce the classes in the gold standard
- (But we only want to reproduce how documents are divided into groups, not the class labels.)
- First measure for how well we were able to reproduce the classes: purity

External criterion: Purity

$$\text{purity}(\Omega, C) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$$

- $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ is the set of clusters and $C = \{c_1, c_2, \ldots, c_J\}$ is the set of classes.
- For each cluster ω_k: find class c_j with most members n_{kj} in ω_k
- Sum all n_{kj} and divide by total number of points

Example for computing purity



To compute purity:

$$5 = \max_j |\omega_1 \cap c_j| \qquad \text{(class x, cluster 1)}$$

$$4 = \max_j |\omega_2 \cap c_j| \qquad \text{(class o, cluster 2)}$$

$$3 = \max_j |\omega_3 \cap c_j| \qquad \text{(class } \diamond \text{, cluster 3)}$$

$$Purity = \frac{5+4+3}{17} \approx 0.71.$$

Another external criterion: Rand index

- Purity can be increased easily by increasing K – a measure that does not have this problem: Rand index.
- Definition: $\mathrm{RI} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{FP}+\mathrm{FN}+\mathrm{TN}}$
- Based on 2x2 contingency table of all pairs of documents:

	same cluster	different clusters
same class	true positives (TP)	false negatives (FN)
different classes	false positives (FP)	true negatives (TN)

- TP+FN+FP+TN is the total number of pairs.
- $\mathrm{TP}+\mathrm{FN}+\mathrm{FP}+\mathrm{TN} = \binom{N}{2}$ for N documents.
- Example: $\binom{17}{2} = 136$ in o/$\diamond$/x example
- Each pair is either positive or negative (the clustering puts the two documents in the same or in different clusters) ...
- ...and either "true" (correct) or "false" (incorrect): the clustering decision is correct or incorrect.

Rand Index: Example

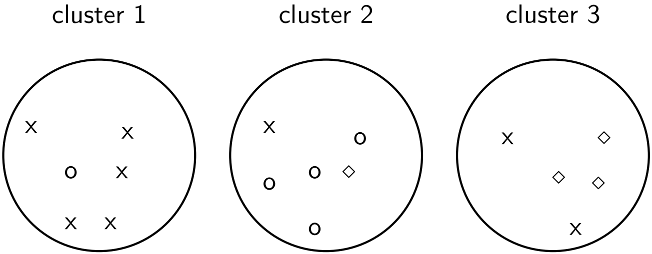
The three clusters contain 6, 6, and 5 points.

$$\text{TP} + \text{FP} = \binom{6}{2} + \binom{6}{2} + \binom{5}{2} = 40$$

Of these, the x pairs in cluster 1, the o pairs in cluster 2, the $\diamond$ pairs in cluster 3, and the x pair in cluster 3 are true positives:

$$\text{TP} = \binom{5}{2} + \binom{4}{2} + \binom{3}{2} + \binom{2}{2} = 20$$

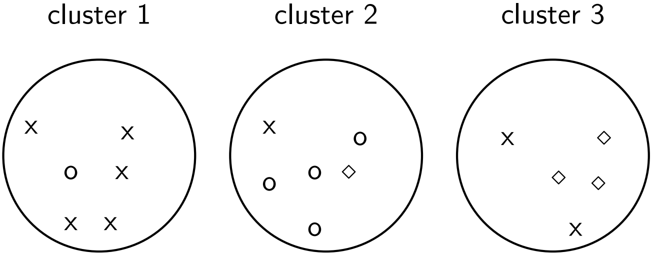
Thus, $\text{FP} = 40 - 20 = 20$.



Rand measure for the o/◇/x example

	same cluster	different clusters
same class	TP = 20	FN = 24
different classes	FP = 20	TN = 72

$$RI = \frac{20 + 72}{20 + 20 + 24 + 72} = \frac{92}{136} \approx 0.68.$$



Normalized mutual information (NMI)

$$NMI(\Omega, C) = \frac{I(\Omega; C)}{(H(\Omega) + H(C))/2}$$

$$\begin{aligned} I(\Omega; C) &= \sum_k \sum_j P(\omega_k \cap c_j) \log \frac{P(\omega_k \cap c_j)}{P(\omega_k)P(c_j)} && (1) \\ &= \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} \log \frac{N|\omega_k \cap c_j|}{|\omega_k||c_j|} && (2) \end{aligned}$$

$$H(\Omega) = \sum_k P(\omega_k) \log P(\omega_k) \qquad (3)$$

- H: entropy
- I: Mutual Information
- the denominator: normalize the value to be within -1 to 1.

Evaluation results for the o/$\diamond$/x example

	purity	NMI	RI	F_5
lower bound	0.0	0.0	0.0	0.0
maximum	1.0	1.0	1.0	1.0
value for example	0.71	0.36	0.68	0.46

- All four measures range from 0 (really bad clustering) to 1 (perfect clustering).
- What is $F5$

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \tag{4}$$

- Give stronger weight to recall when $\beta > 1$.

Outline

1. Recap
2. Clustering: Introduction
3. Clustering in IR
4. *K*-means
5. Evaluation
6. How many clusters?

How many clusters?

- Number of clusters K is given in many applications.
  - E.g., there may be an external constraint on K.
  - Example: it is hard to show more than 10–20 clusters on a monitor in the 90s.
- What if there is no external constraint? Is there a "right" number of clusters?
- One way to go: define an optimization criterion
  - Given docs, find K for which the optimum is reached.
  - What optimization criterion can we use?
  - We can't use RSS or average squared distance from centroid as criterion: always chooses $K = N$ clusters.

Exercise

- Your job is to develop the clustering algorithms for a competitor to news.google.com
- You want to use K-means clustering.
- How would you determine K?

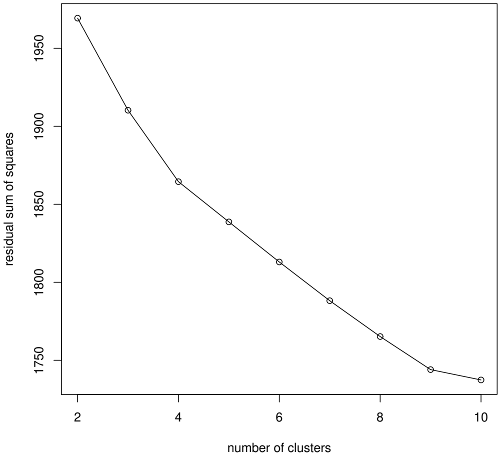
Simple objective function for K: Basic idea

- Start with 1 cluster ($K = 1$)
- Keep adding clusters (= keep increasing K)
- Add a penalty for each new cluster
- Then trade off cluster penalties against average squared distance from centroid
- Choose the value of K with the best tradeoff

Simple objective function for K: Formalization

- Given a clustering, define the cost for a document as (squared) distance to centroid
- Define total distortion RSS(K) as sum of all individual document costs (corresponds to average distance)
- Then: penalize each cluster with a cost λ
- Thus for a clustering with K clusters, total cluster penalty is $K\lambda$
- Define the total cost of a clustering as distortion plus total cluster penalty: $\text{RSS(K)} + K\lambda$
- Select K that minimizes $(\text{RSS(K)} + K\lambda)$
- Still need to determine good value for λ ...

Finding the "knee" in the curve



Pick the number of clusters where curve "flattens". Here: 4 or 9.

Take-away today

- What is clustering?
- Applications of clustering in information retrieval
- K-means algorithm
- Evaluation of clustering
- How many clusters?