Clustering

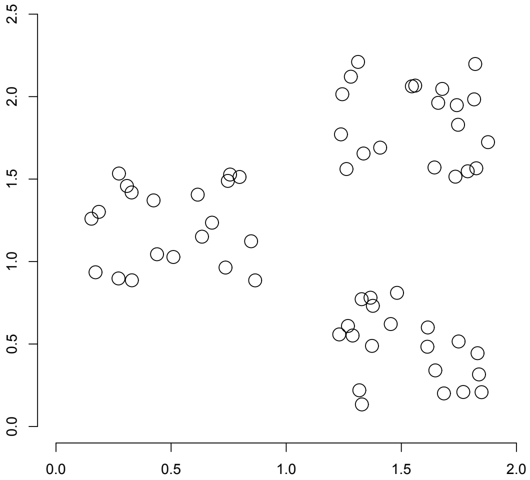
Outline

1. Clustering: Introduction
2. Clustering in IR
3. K-means
4. Evaluation

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 $\Rightarrow$ there are no labelled 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.
  - However, there are many ways of influencing the outcome of clustering: number of clusters, similarity measure, representation of documents, . . .

Outline

The cluster hypothesis

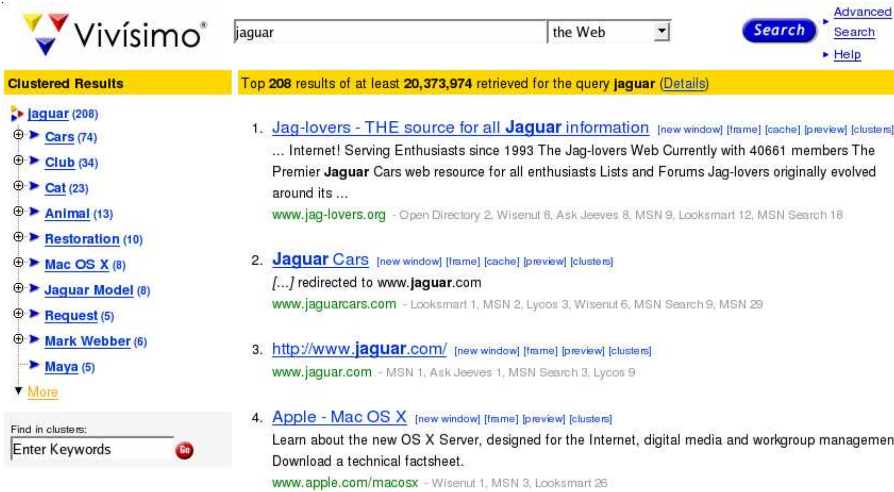
Cluster hypothesis.

- Documents in the same cluster behave similarly with respect to relevance to information needs.
- All applications of clustering in IR are based (directly or indirectly) on the cluster hypothesis.
- Van Rijsbergen's original wording: "closely associated documents tend to be relevant to the same requests".

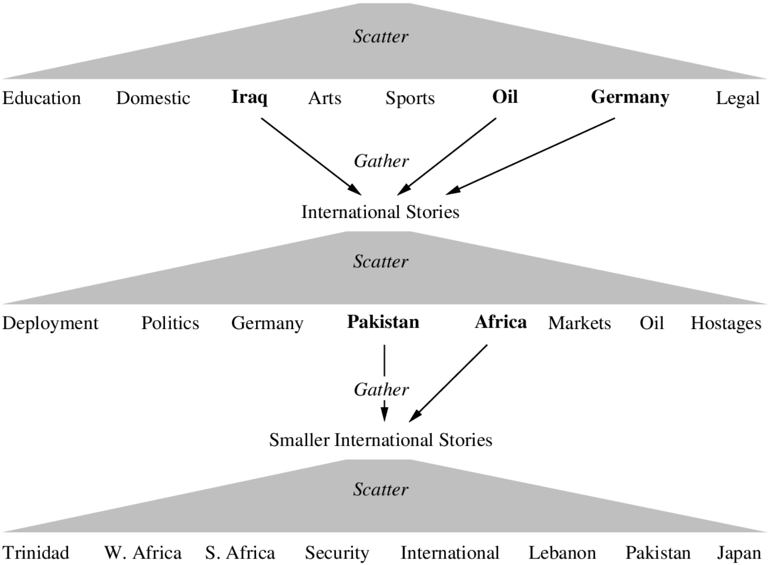
Applications of clustering in IR

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



Scatter-Gather



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".

Desiderata for clustering

- General goal: put related docs in the same cluster, put unrelated docs in different clusters.
  - How do we formalise this?
- The number of clusters should be appropriate for the data set we are clustering.
- 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
  - Main 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 optimises the chosen partitioning criterion
- Global optimisation: exhaustively enumerate partitions, pick optimal one
  - Not tractable
- Effective heuristic method: K-means algorithm

Outline

1. Clustering: Introduction
2. Clustering in IR
3. **K-means**
4. Evaluation

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
- We can measure relatedness between vectors by Euclidean distance

K-means

- Each cluster in K-means is def ned by a centroid.
- Objective/partitioning criterion: minimise 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 algorithm

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 $j \leftarrow \arg\min_{j'} |\vec{\mu}_{j'} - \vec{x}_n|$

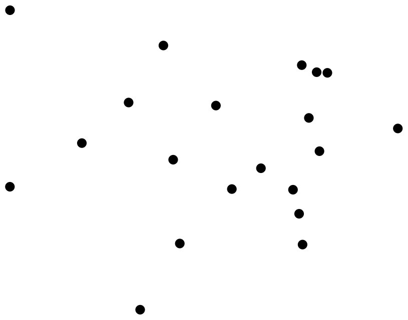
    $\omega_j \leftarrow \omega_j \cup \{\vec{x}_n\}$ *(reassignment of vectors)*

  for $k \leftarrow 1$ **to** K

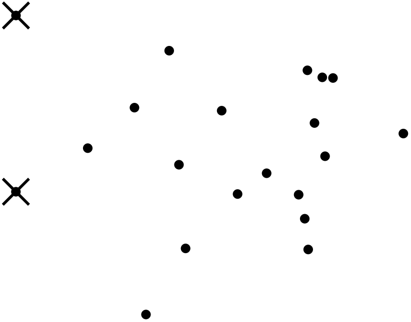
  do $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$ *(recomputation of centroids)*

return $\{\vec{\mu}_1, \ldots, \vec{\mu}_K\}$

Worked Example : Set of points to be clustered

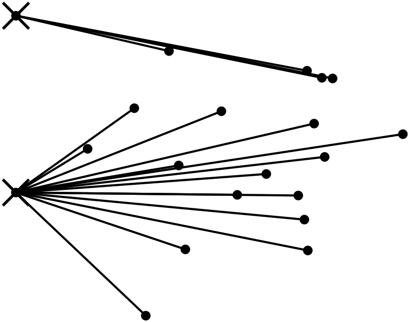


Worked Example: Random selection of initial centroids

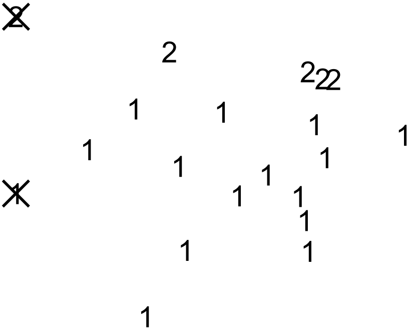


Exercise: (i) clustering into two clusters

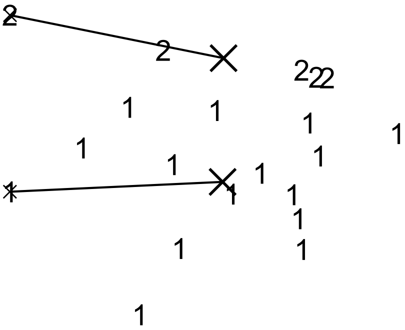
Worked Example: Assign points to closest center



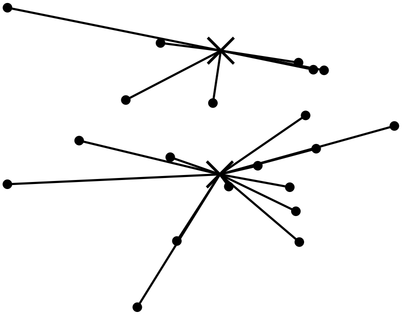
Worked Example: Assignment



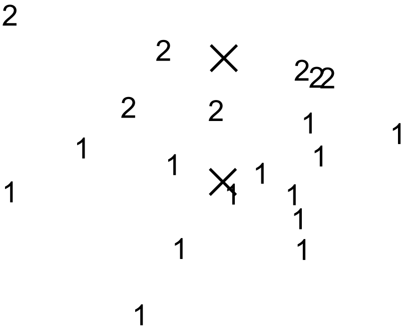
Worked Example: Recompute cluster centroids



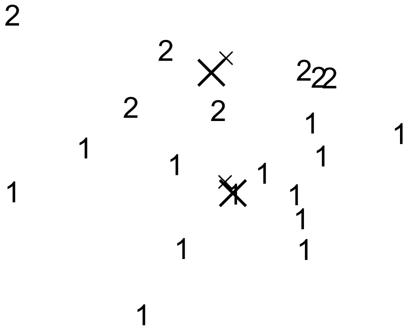
Worked Example: Assign points to closest centroid



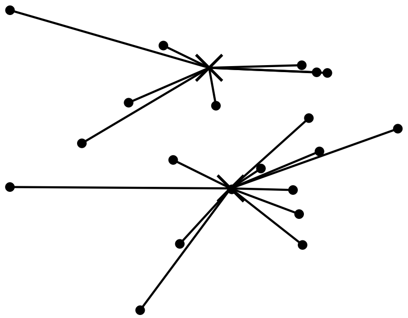
Worked Example: Assignment



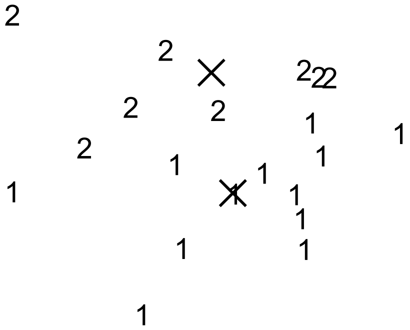
Worked Example: Recompute cluster centroids



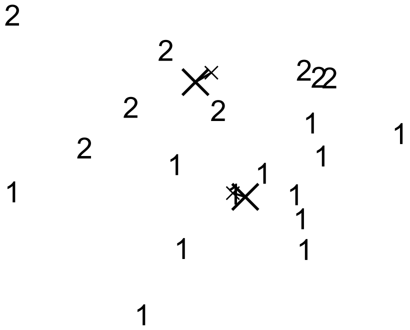
Worked Example: Assign points to closest centroid



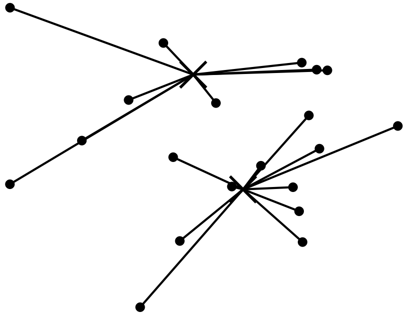
Worked Example: Assignment



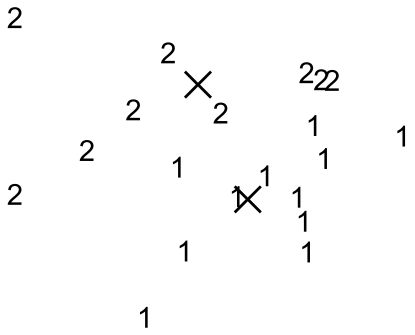
Worked Example: Recompute cluster centroids



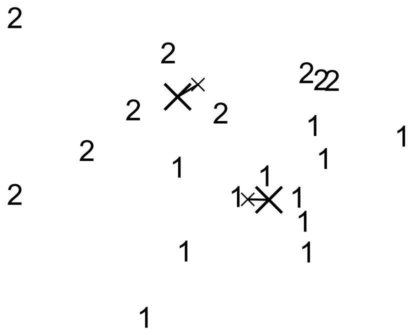
Worked Example: Assign points to closest centroid



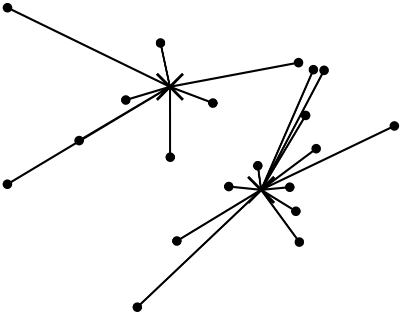
Worked Example: Assignment



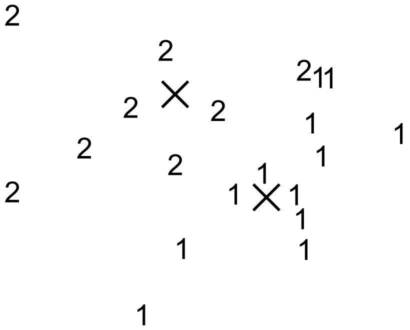
Worked Example: Recompute cluster centroids



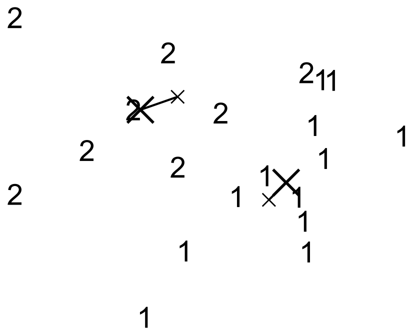
Worked Example: Assign points to closest centroid



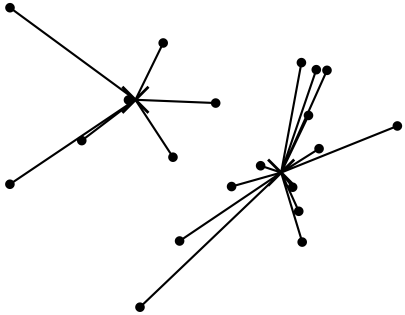
Worked Example: Assignment



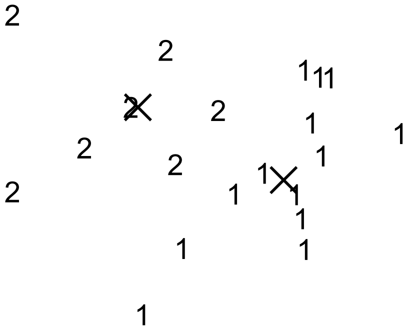
Worked Example: Recompute cluster centroids



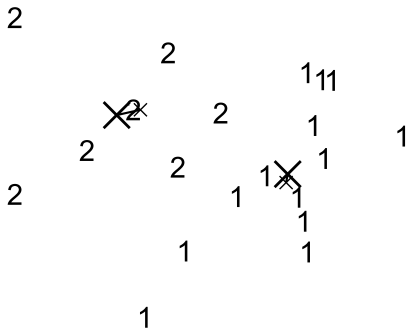
Worked Example: Assign points to closest centroid



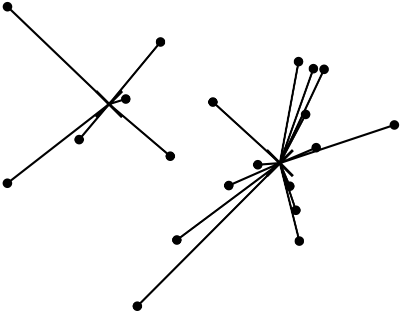
Worked Example: Assignment



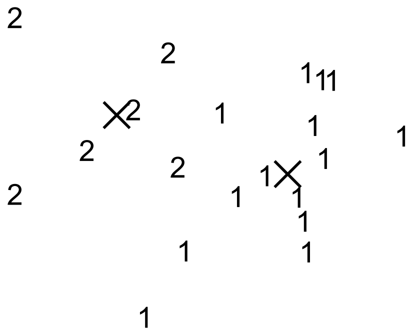
Worked Example: Recompute cluster centroids



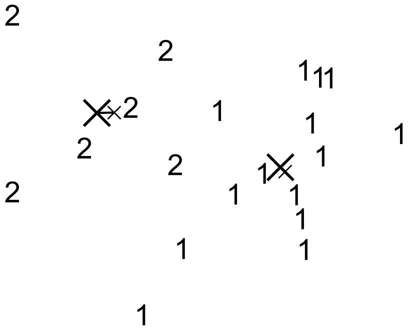
Worked Example: Assign points to closest centroid



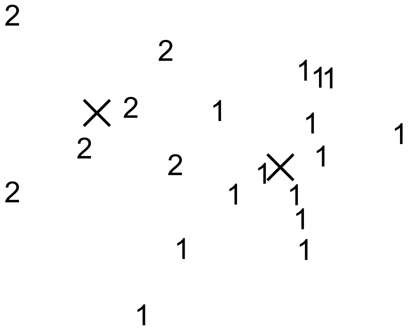
Worked Example: Assignment



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.
  -
- There is only a finite number of clusterings.
- Thus: We must reach a fixed point.

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 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 poor.

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

Outline

What is a good clustering?

- Internal criteria
  - Example of an internal criterion: RSS in K-means
- 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 classification

External criteria for clustering quality

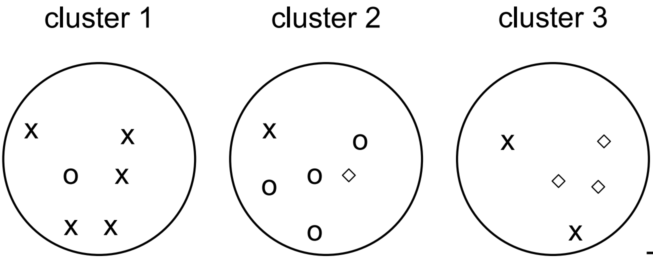
- Based on a gold standard data set
- Goal: Clustering should reproduce the classes in the gold standard
- First measure for how well we were able to reproduce the classes: purity

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: f nd 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 = $\max_j |\omega_1 \cap c_j|$ (class x, cluster 1); 4 = $\max_j |\omega_2 \cap c_j|$ (class o, cluster 2); and 3 = $\max_j |\omega_3 \cap c_j|$ (class $\diamond$, cluster 3). Purity is $(1/17) \times (5+4+3) \approx 0.71$.

Rand index

- Definition: $\text{RI} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{FP}+\text{FN}+\text{TN}}$
- 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.
- There are $\binom{N}{2}$ pairs for N documents.
- 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.

How many clusters?

- Number of clusters K is given in many applications.
  - E.g., there may be an external constraint on K. Example: In the case of Scatter-Gather, it was 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 optimisation criterion
  - Given docs, find K for which the optimum is reached.
  - What optimisation criterion can we use?
  - We can't use RSS or average squared distance from centroid as criterion: always chooses $K = N$ clusters.

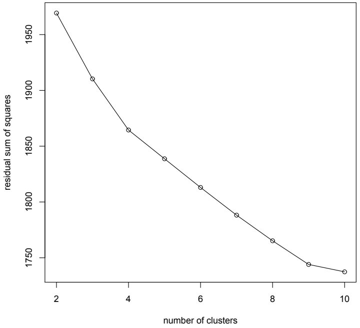
Simple objective function for K (1)

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

Simple objective function for K (2)

- 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: penalise 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: RSS(K)+ $K\lambda$
- Select K that minimises (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.

Summary

- Clustering has many applications in IR
- Many approaches - K-means one such approach
- Issues with choosing optimal K
- Many other approaches exist for clustering - hierarchical, soft etc.