CS6322: Information Retrieval
Sanda Harabagiu

Lecture 13: Clustering
Today’s Topic: Clustering

- Document clustering
  - Motivations
  - Document representations
  - Success criteria
- Clustering algorithms
  - Partitional
  - Hierarchical
What is clustering?

- **Clustering**: the process of grouping a set of objects into classes of similar objects
  - Documents within a cluster *should be similar*.
  - Documents from *different clusters should be dissimilar*.
- The commonest form of *unsupervised learning*
  - Unsupervised learning = learning from raw data, as opposed to supervised data where a classification of examples is given
  - A common and important task that finds many applications in IR and other places
A data set with clear cluster structure

- How would you design an algorithm for finding the three clusters in this case?
Applications of clustering in IR

- Whole corpus analysis/navigation
  - Better user interface: search without typing
- For improving recall in search applications
  - Better search results (like pseudo RF)
- For better navigation of search results
  - Effective “user recall” will be higher
- For speeding up vector space retrieval
  - Cluster-based retrieval gives faster search
Yahoo! Hierarchy *isn’t* clustering but *is* the kind of output you want from clustering

www.yahoo.com/Science
Google News: automatic clustering gives an effective news presentation metaphor
Scatter/Gather: Cutting, Karger, and Pedersen
For visualizing a document collection and its themes

- Wise et al, “Visualizing the non-visual” PNNL
- ThemeScapes, Cartia
  - [Mountain height = cluster size]
For improving search recall

- **Cluster hypothesis** - Documents in the same cluster behave similarly with respect to relevance to information needs

- Therefore, to improve search recall:
  - Cluster docs in corpus 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 clustering grouped together docs containing *car* with those containing *automobile*.

Why might this happen?
For better navigation of search results

- For grouping search results thematically
  - clusty.com / Vivisimo
Problem Statement

- Define the goal of hard flat clustering. Given:
  1. A set of documents \( D =\{d_1, d_2, \ldots, d_N\} \)
  2. A desired number of clusters \( K \)
  3. An objective function that evaluates the quality of clustering

Compute an assignment \( \gamma: D \rightarrow \{1, 2, \ldots, K\} \) that minimizes the objective function

- Sometimes we want \( \gamma \) to be surjective (none of the \( K \) clusters are empty!!!)
- The objective function is defined in terms of similarity or distance between documents.
Issues for clustering

- **Representation for clustering**
  - Document representation
    - Vector space? Normalization?
      - Centroids aren’t length normalized
  - Need a notion of similarity/distance

- **How many clusters?**
  - Fixed a priori?
  - Completely data driven?
    - Avoid “trivial” clusters - too large or small
      - If a cluster's too large, then for navigation purposes you've wasted an extra user click without whittling down the set of documents much.
Notion of similarity/distance

- Ideal: semantic similarity.
- Practical: term-statistical similarity
  - We will use cosine similarity.
  - Docs as vectors.
  - For many algorithms, easier to think in terms of a distance (rather than similarity) between docs.
  - We will mostly speak of Euclidean distance
    - But real implementations use cosine similarity
Clustering Algorithms

- **Flat algorithms**
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - $K$ means clustering
    - (Model based clustering)

- **Hierarchical algorithms**
  - 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 a pair of sneakers in two clusters: (i) sports apparel and (ii) shoes
  - You can only do that with a soft clustering approach.
- We won’t do soft clustering today. See IIR 16.5, 18
Partitioning Algorithms

- Partitioning method: Construct a partition of $n$ documents into a set of $K$ clusters
- Given: a set of documents and the number $K$
- Find: a partition of $K$ clusters that optimizes the chosen partitioning criterion
  - Globally optimal
    - Intractable for many objective functions
    - Ergo, exhaustively enumerate all partitions
  - Effective heuristic methods: $K$-means and $K$-medoids algorithms
K-Means

- Assumes documents are real-valued vectors.
- Clusters based on centroids (aka the center of gravity or mean) of points in a cluster, $c$:

$$
\widehat{\mu}(c) = \frac{1}{|c|} \sum_{\tilde{x} \in c} \tilde{x}
$$

- Reassignment of instances to clusters is based on distance to the current cluster centroids.
  - (Or one can equivalently phrase it in terms of similarities)
K-Means Algorithm

1. Select $K$ random docs $\{s_1, s_2, \ldots, s_K\}$ as seeds.
2. Until clustering converges (or other stopping criterion):
   a) For each doc $d_i$:
      Assign $d_i$ to the cluster $c_j$ such that $dist(x_i, s_j)$ is minimal.
      
      *(Next, update the seeds to the centroid of each cluster)*
   b) For each cluster $c_j$
      
      $s_j = \mu(c_j)$
K Means Example

(K=2)

Pick seeds
Reassign clusters
Compute centroids
Reassign clusters
Compute centroids
Reassign clusters

Converged!
Termination conditions

- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Doc partition unchanged.
  - Centroid positions don’t change.

Does this mean that the docs in a cluster are unchanged?
Convergence

- Why should the $K$-means algorithm ever reach a fixed point?
  - A state in which clusters don’t change.
- $K$-means is a special case of a general procedure known as the *Expectation Maximization (EM) algorithm*.
  - EM is known to converge.
  - Number of iterations could be large.
    - But in practice usually isn’t
Convergence of $K$-Means

- Define goodness measure of cluster $k$ as sum of squared distances from cluster centroid:
  - $G_k = \sum_i (d_i - c_k)^2$ (sum over all $d_i$ in cluster $k$)
  - $G = \sum_k G_k$
- Reassignment monotonically decreases $G$ since each vector is assigned to the closest centroid.
- A measure of how well the centroids represent the members of their clusters is the residual sum of squares (RSS) (the squared distance of each vector from its centroid summed over all vectors):

$$RSS_k = \sum_{\tilde{x} \in \omega_k} |\tilde{x} - \bar{\mu}(\omega_k)|^2$$

$$RSS = \sum_{k=1}^{K} RSS_k$$
The $K$-Means Algorithm

$K$-MEANS$(\{\bar{x}_1, \ldots, \bar{x}_N\}, K)$
1. $(\bar{s}_1, \bar{s}_2, \ldots, \bar{s}_K) \leftarrow$ SELECTRANDOMSEEDS$(\{\bar{x}_1, \ldots, \bar{x}_N\}, K)$
2. for $k \leftarrow 1$ to $K$
3. do $\bar{\mu}_k \leftarrow \bar{s}_k$
4. while stopping criterion has not been met
5. do for $k \leftarrow 1$ to $K$
6. do $\omega_k \leftarrow \{\}$
7. for $n \leftarrow 1$ to $N$
8. do $j \leftarrow \text{arg min}_{j'} |\bar{\mu}_{j'} - \bar{x}_n|$
9. $\omega_j \leftarrow \omega_j \cup \{\bar{x}_n\}$ (reassignment of vectors)
10. for $k \leftarrow 1$ to $K$
11. do $\bar{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\bar{x} \in \omega_k} \bar{x}$ (recomputation of centroids)
12. return $\{\bar{\mu}_1, \ldots, \bar{\mu}_K\}$

Select the seeds!!!

Move the cluster centers around to minimize the RSS
selection of seeds

assignment of documents (iter. 1)

recomputation/movement of $\bar{\mu}$'s (iter. 1)

$\bar{\mu}$'s after convergence (iter. 9)
Example for K=2

movement of \( \bar{\mu} \)'s in 9 iterations
Time Complexity

- Computing distance between two docs is $O(M)$ where $M$ is the dimensionality of the vectors.
- Reassigning clusters: $O(KN)$ distance computations, or $O(KNM)$.
- Computing centroids: Each doc gets added once to some centroid: $O(NM)$.
- Assume these two steps are each done once for $I$ iterations: $O(INKM)$. 
Seed Choice

- Results can vary based on random seed selection.
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
  - Select good seeds using a heuristic (e.g., doc least similar to any existing mean)
  - Try out multiple starting points
  - Initialize with the results of another method.

Example showing sensitivity to seeds

In the above, if you start with B and E as centroids you converge to \{A,B,C\} and \{D,E,F\}.
If you start with D and F you converge to \{A,B,D,E\} \{C,F\}.
**K-means issues, variations, etc.**

- Recomputing the centroid after every assignment (rather than after all points are re-assigned) can improve speed of convergence of *K*-means
- Assumes clusters are spherical in vector space
  - Sensitive to coordinate changes, weighting etc.
- Disjoint and exhaustive
  - Doesn’t have a notion of “outliers” by default
  - But can add outlier filtering
How Many Clusters?

- Number of clusters $K$ is given
  - Partition $n$ docs into predetermined number of clusters
- Finding the “right” number of clusters is part of the problem
  - Given docs, partition into an “appropriate” number of subsets.
  - E.g., for query results - ideal value of $K$ not known up front - though UI may impose limits.
- Can usually take an algorithm for one flavor and convert to the other.
$K$ not specified in advance

- Say, the results of a query.
- Solve an optimization problem: penalize having lots of clusters
  - application dependent, e.g., compressed summary of search results list.
- Tradeoff between having more clusters (better focus within each cluster) and having too many clusters
Given a clustering, define the Benefit for a doc to be the cosine similarity to its centroid.

Define the Total Benefit to be the sum of the individual doc Benefits.

Why is there always a clustering of Total Benefit $n$?
Penalize lots of clusters

- For each cluster, we have a Cost $C$.
- Thus for a clustering with $K$ clusters, the Total Cost is $KC$.
- Define the Value of a clustering to be $\text{Value} = \text{Total Benefit} - \text{Total Cost}$.
- Find the clustering of highest value, over all choices of $K$.
  - Total benefit increases with increasing $K$. But can stop when it doesn’t increase by “much”. The Cost term enforces this.
Evaluation of Clustering

- Typical Objective: *attain a high inter-cluster similarity and lower inter-cluster similarity* INTERNAL CRITERION
- EXTERNAL CRITERION: use a gold standard, e.g. set of classes

- Four external criteria for evaluating the quality of clustering:
  1. Purity
  2. Normalized mutual information
  3. Rand index
  4. F-measure
Purity

- To compute purity each cluster is assigned to the class which is most frequent in the cluster.
- The accuracy of this assignment is measured by counting the number of correctly assigned documents and dividing by \( N \).
- Formally:

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

- Where \( \Omega=\{\omega_1, \omega_2, \ldots, \omega_k\} \) is the set of clusters
- \( C=\{c_1, c_2, \ldots, c_j\} \) is the set of classes
Computing the Purity

- **FORMALLY:**

\[
purity(\Omega, C) = \frac{1}{N} \sum_{k} \max_{j} | \omega_k \cap c_j |
\]

Where \( \Omega = \{\omega_1, \omega_2, \ldots, \omega_k\} \) is the set of clusters
- \( C = \{c_1, c_2, \ldots, c_J\} \) is the set of classes

High purity is possible to be achieved when the number of clusters is high!!!
Normalized Mutual Information

- **FORMALLY:**

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

- Where \( I \) is the mutual information and \( H \) is the entropy

\[
I(\Omega, C) = \sum_k \sum_j P(w_k \cap c_j) \log \frac{P(w_k \cap c_j)}{P(w_k)P(c_j)} = \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} \log \frac{N |\omega_k \cap c_j|}{|\omega_k \cap c_j|}
\]

\[
H(\Omega) = -\sum_k P(w_k) \log P(w_k) = \sum_k \frac{|\omega_k|}{N} \log \frac{|\omega_k|}{N}
\]
Rand Index

- Clustering is viewed as a series of decision, one of each of the $N \times (N-1)/2$ pairs of documents in the collection:
- A **TRUE POSITIVE (TP)** decision assigns two similar documents to the same cluster
- A **TRUE NEGATIVE (TN)** decision assigns two dissimilar documents to different clusters
- A **FALSE POSITIVE (FP)** decision assigns to dissimilar document to the same cluster
- A **FALSE NEGATIVE (FN)** decision assigns to similar document to different clusters

$$RI(Ω, C) = \frac{TP + TN}{TP + FP + FN + TN}$$
F1 Measure

- RI gives the same weight to FP and FN.
- Separating similar documents is much worse than putting pairs of dissimilar documents in the same cluster. To penalize FN more, we use the F1 measure by selecting $\beta > 1$

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN}$$

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

- Cluster center defined as the mean or centroid:

\[
\bar{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\bar{x} \in \omega} \bar{x}
\]

- Where \( \Omega = \{\omega_1, \omega_2, \ldots, \omega_k\} \) is the set of clusters

- A measure of how well the centroids represent the members of the clusters is the residual sum of squares (RSS), i.e. the squared distance of each vector from its centroid summed over all vectors:

\[
RSS_k = \frac{1}{|\omega|} \sum_{\bar{x} \in \omega_k} |\bar{x} - \bar{\mu}(\omega_k)|^2
\]

\[
RSS = \sum_{k=1}^{K} RSS_k
\]

K-means tries to minimize it!!!
The *K*-Means Algorithm

\[ \text{K-MEANS}(\{\bar{x}_1, \ldots, \bar{x}_N\}, K) \]

1. \((\bar{s}_1, \bar{s}_2, \ldots, \bar{s}_K) \leftarrow \text{SELECTRANDOMSEEDS}(\{\bar{x}_1, \ldots, \bar{x}_N\}, K)\)
2. \text{for } k \leftarrow 1 \text{ to } K \text{ do}
   3. \quad \bar{\mu}_k \leftarrow \bar{s}_k
4. \text{while stopping criterion has not been met do}
5. \quad \text{for } k \leftarrow 1 \text{ to } K \text{ do}
   6. \quad \omega_k \leftarrow \{\}
   7. \quad \text{for } n \leftarrow 1 \text{ to } N \text{ do}
   8. \quad \omega_j \leftarrow \omega_j \bigcup \{\bar{x}_n\} \quad \text{(reassignment of vectors)}
   9. \quad \omega_j \leftarrow \omega_j \bigcup \{\bar{x}_n\} \quad \text{(reassignment of vectors)}
10. \quad \text{for } k \leftarrow 1 \text{ to } K \text{ do}
11. \quad \bar{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{x \in \omega_k} \bar{x} \quad \text{(recomputation of centroids)}
12. \text{return } \{\bar{\mu}_1, \ldots, \bar{\mu}_K\} \]
Hierarchical Clustering

- Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of documents.

- One approach: recursive application of a partitional clustering algorithm.

```
  animal
 /     \
vertebrate  invertebrate
 /  \
fish reptile amphib. mammal  worm insect crustacean
```
Dendrogram: Hierarchical Clustering

- Clustering obtained as a dendogram when using single-link clustering on 30 documents from Reuters-RCV1.

- **Two possible cuts in the dendogram are shown:** at 0.4 (into 24 clusters) and at 0.1 (into 12 clusters).

- By cutting the dendrogram at a desired level: each connected component forms a cluster.
Hierarchical Agglomerative Clustering (HAC)

- Starts with each doc in a separate cluster
  - then repeatedly joins the closest pair of clusters, until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.
- It does not require a pre-specified number of clusters!!
- Some applications may require a partition of disjoint clusters – what do we do??? *Cut the hierarchy at some point***!!
Cutting the Dendogram

**Criteria for determining the cutting point:**

- Cut at a **pre-specified level of similarity**
- Cut the dendogram where the gap between two successive combinations of similarities is largest.
- Apply the equation:

\[
K = \arg\min_{K'} [RSS(K') + \lambda K']
\]

\[
RSS_k = \frac{1}{|\omega|} \sum_{\bar{x} \in \omega_k} |\bar{x} - \mu(\omega_k)|^2
\]

- Where \(K'\) refers to the cut in the hierarchy that results into \(K'\) clusters
- Pre-specify the number of clusters
Simple but inefficient HAC algorithm

\[
\text{SIMPLEHAC}(d_1, \ldots, d_N)
\]
\[
\begin{align*}
1 & \text{ for } n \leftarrow 1 \text{ to } N \\
2 & \text{ do for } i \leftarrow 1 \text{ to } N \\
3 & \quad \text{ do } C[n][i] \leftarrow \text{SIM}(d_n, d_i) \\
4 & \quad I[n] \leftarrow 1 \text{ (keeps track of active clusters)} \\
5 & \quad A \leftarrow [] \text{ (assembles clustering as a sequence of merges)} \\
6 & \text{ for } k \leftarrow 1 \text{ to } N - 1 \\
7 & \quad \langle i, m \rangle \leftarrow \arg \max \{(i, m) : i \neq m \land I[i] = 1 \land I[m] = 1\} \ C[i][m] \\
8 & \quad A.\text{APPEND}((i, m)) \text{ (store merge)} \\
9 & \quad \text{ for } j \leftarrow 1 \text{ to } N \\
10 & \quad \text{ do } C[i][j] \leftarrow \text{SIM}(i, m, j) \\
11 & \quad \text{ and } C[j][i] \leftarrow \text{SIM}(i, m, j) \\
12 & \quad I[m] \leftarrow 0 \text{ (deactivate cluster)} \\
13 & \text{ return } A
\end{align*}
\]

First compute a $N \times N$ similarity matrix $C$

Merge the two most similar clusters

Rows and columns of the merged clusters are updated

IMPORTANT QUESTION: How do we compute the similarity???
Closest pair of clusters

- Many variants to defining closest pair of clusters
  - Single-link
    - Similarity of the most cosine-similar (single-link)
  - Complete-link
    - Similarity of the “furthest” points, the least cosine-similar
  - Centroid
    - Clusters whose centroids (centers of gravity) are the most cosine-similar
  - Average-link
    - Average cosine between pairs of elements
Single Link Agglomerative Clustering

- Use maximum similarity of pairs:
  \[
  \text{sim}(c_i, c_j) = \max_{x \in c_i, y \in c_j} \text{sim}(x, y)
  \]

- Can result in “straggly” (long and thin) clusters due to chaining effect.

- After merging \(c_i\) and \(c_j\), the similarity of the resulting cluster to another cluster, \(c_k\), is:
  \[
  \text{sim}((c_i \cup c_j), c_k) = \max(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k))
  \]
Single Link Example
Complete Link

- Use minimum similarity of pairs:

\[
sim(c_i, c_j) = \min_{x \in c_i, y \in c_j} \sim(x, y)
\]

- Makes “tighter,” spherical clusters that are typically preferable.

- After merging \(c_i\) and \(c_j\), the similarity of the resulting cluster to another cluster, \(c_k\), is:

\[
sim((c_i \cup c_j), c_k) = \min\{\sim(c_i, c_k), \sim(c_j, c_k)\}
\]
Complete Link Example
Computational Complexity

- In the first iteration, all HAC methods need to compute similarity of all pairs of \( N \) initial instances, which is \( O(N^2) \).
- In each of the subsequent \( N-2 \) merging iterations, compute the distance between the most recently created cluster and all other existing clusters.
- In order to maintain an overall \( O(N^2) \) performance, computing similarity to each other cluster must be done in constant time.
  - Often \( O(N^3) \) if done naively or \( O(N^2 \log N) \) if done more cleverly
The rows of the similarity matrix $C$ are sorted in decreasing order of similarity in the priority queues $P$.

$P[k].\text{MAX}()$ returns the cluster in $P[k]$ that currently has the highest similarity with the $k^{th}$ cluster.

---

**Figure 17.8** The priority-queue algorithm for HAC. Top: The algorithm. Center: Four different similarity measures. Bottom: An example for processing steps 6 and 16–19. This is a made-up example showing $P[5]$ for a $5 \times 5$ matrix $C$. 

```
for n ← 1 to N
  do for i ← 1 to N
      do $C[n][i].\text{sim} ← \vec{d}_n \cdot \vec{d}_i$
      $C[n][i].\text{index} ← i$
  $I[n] ← 1$
  $P[n].\text{index} ← n$
  $P[n].\text{MAX}().\text{sim} ← \text{self-similarities}$
  $A ← []$
  for $k ← 1$ to $N - 1$
    do $k_1 ← \text{arg max}_{k} P[k].\text{MAX}().\text{sim}$
    do $k_2 ← P[k_1].\text{MAX}().\text{index}$
    do $A.\text{APPEND}((k_1, k_2))$
    do $I[k_2] ← 0$
    do $P[k_1] ← []$
    for each $i$ with $I[i] = 1$ and $i ≠ k_1$
      do $P[i].\text{DELETE}(C[i][k_1])$
      do $P[i].\text{DELETE}(C[i][k_2])$
      do $C[i][k_1].\text{sim} ← \text{SIM}(i, k_1, k_2)$
      do $P[i].\text{INSERT}(C[i][k_1])$
      do $C[k_1][i].\text{sim} ← \text{SIM}(i, k_1, k_2)$
      do $P[k_1].\text{INSERT}(C[k_1][i])$
return $A$
```
Group Average

- Similarity of two clusters = average similarity of all pairs within merged cluster.

\[
sim(c_i, c_j) = \frac{1}{|c_i \cup c_j|(|c_i \cup c_j| - 1)} \sum_{\bar{x} \in (c_i \cup c_j)} \sum_{\bar{y} \in (c_i \cup c_j): \bar{y} \neq \bar{x}} \text{sim}(\bar{x}, \bar{y})
\]

- Compromise between single and complete link.

- Two options:
  - Averaged across all ordered pairs in the merged cluster
  - Averaged over all pairs between the two original clusters

- No clear difference in efficacy
Computing Group Average Similarity

- Always maintain sum of vectors in each cluster.

\[ \tilde{s}(c_j) = \sum_{\tilde{x} \in c_j} \tilde{x} \]

- Compute similarity of clusters in constant time:

\[
sim(c_i, c_j) = \frac{(\tilde{s}(c_i) + \tilde{s}(c_j)) \cdot (\tilde{s}(c_i) + \tilde{s}(c_j)) - (|c_i| + |c_j|)}{(|c_i| + |c_j|)(|c_i| + |c_j| - 1)}
\]
What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  - the **intra-class** (that is, intra-cluster) similarity is high
  - the **inter-class** similarity is low
  - The measured quality of a clustering depends on both the document representation and the similarity measure used
External criteria for clustering quality

- Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
- Assesses a clustering with respect to ground truth ... requires labeled data
- Assume documents with $C$ gold standard classes, while our clustering algorithms produce $K$ clusters, $\omega_1, \omega_2, ..., \omega_K$ with $n_i$ members.
External Evaluation of Cluster Quality

- Simple measure: **purity**, the ratio between the dominant class in the cluster $\pi_i$ and the size of cluster $\omega_i$

\[
Purity(\omega_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C
\]

- Biased because having $n$ clusters maximizes purity

- Others are entropy of classes in clusters (or mutual information between classes and clusters)
Purity example

Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5
Rand Index measures between pair decisions. Here RI = 0.68

<table>
<thead>
<tr>
<th>Number of points</th>
<th>Same Cluster in clustering</th>
<th>Different Clusters in clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same class in ground truth</td>
<td>20</td>
<td>24</td>
</tr>
<tr>
<td>Different classes in ground truth</td>
<td>20</td>
<td>72</td>
</tr>
</tbody>
</table>
Rand index and Cluster F-measure

\[ RI = \frac{A + D}{A + B + C + D} \]

Compare with standard Precision and Recall:

\[ P = \frac{A}{A + B} \quad R = \frac{A}{A + C} \]

People also define and use a cluster F-measure, which is probably a better measure.
Final word and resources

- In clustering, clusters are inferred from the data without human input (unsupervised learning).
- However, in practice, it’s a bit less clear: there are many ways of influencing the outcome of clustering: number of clusters, similarity measure, representation of documents, ...

Resources
- IIR 16 except 16.5
- IIR 17.1–17.3