Clustering Algorithms

Data mining lab 8

Clustering

- Clustering – discovering groups of people, things, ideas, which are closely related

You'll learn

- Distance metrics
- Two clustering algorithms
- Retrieving data from blogs
- Visualization techniques

Tutorial on clustering documents and words

- Input data (titles of the papers):

d1: Human machine interface for ABC computer applications

d2: A survey of user opinion of computer system response time

d3: The EPS user interface management system

d4: System and human system engineering testing of EPS

d5: Relation of user perceived response time to error measurement

d6: The generation of random binary ordered trees

d7: The intersection graph of paths in trees

d8: Graph minors IV: widths of trees and well-quasi-ordering

d9: Graph minors: A survey

From documents to list of words

- d8 : ['graph', 'minors', 'iv', 'widths', 'of', 'trees', 'and', 'well-quasi-ordering']
- d9 : ['graph', 'minors', 'a', 'survey']
- d6 : ['the', 'generation', 'of', 'random', 'binary', 'ordered', 'trees']
- d7 : ['the', 'intersection', 'graph', 'of', 'paths', 'in', 'trees']
- d4 : ['system', 'and', 'human', 'system', 'engineering', 'testing', 'of', 'eps']
- d5 : ['relation', 'of', 'user', 'perceived', 'response', 'time', 'to', 'error', 'measurement']
- d2 : ['a', 'survey', 'of', 'user', 'opinion', 'of', 'computer', 'system', 'response', 'time']
- d3 : ['the', 'eps', 'user', 'interface', 'management', 'system']
- d1 : ['human', 'machine', 'interface', 'for', 'abc', 'computer', 'applications']

Remove stop words and infrequent words

d8 : ['graph', 'minors', 'iv', 'widths', 'of', 'trees', 'and', 'well-quasi-ordering']

d9 : ['graph', 'minors', 'a', 'survey']

d6 : ['the', 'generation', 'of', 'random', 'binary', 'ordered', 'trees']

d7 : ['the', 'intersection', 'graph', 'of', 'paths', 'in', 'trees']

d4 : ['system', 'and', 'human', 'system', 'engineering', 'testing', 'of', 'eps']

d5 : ['relation', 'of', 'user', 'perceived', 'response', 'time', 'to', 'error', 'measurement']

d2 : ['a', 'survey', 'of', 'user', 'opinion', 'of', 'computer', 'system', 'response', 'time']

d3 : ['the', 'eps', 'user', 'interface', 'management', 'system']

d1 : ['human', 'machine', 'interface', 'for', 'abc', 'computer', 'applications']

Word-document matrix

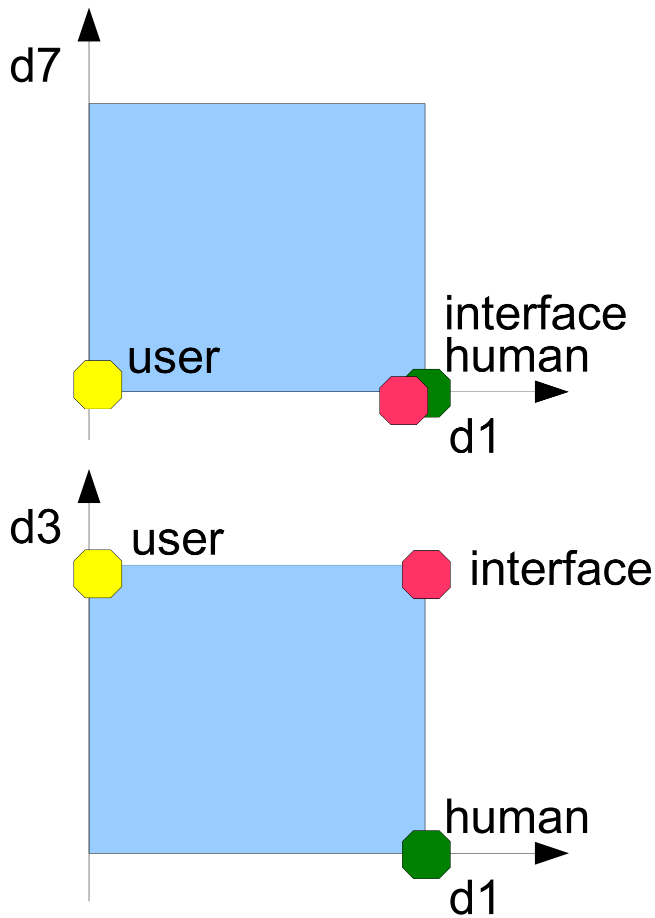
Word dimensions

Items	minors	human	graph	trees	user	interface	response	eps	survey	time
d8	1	0	1	1	0	0	0	0	0	0
d9	1	0	1	0	0	0	0	0	1	0
d6	0	0	0	1	0	0	0	0	0	0
d7	0	0	1	1	0	0	0	0	0	0
d4	0	1	0	0	0	0	0	1	0	0
d5	0	0	0	0	1	0	1	0	0	1
d2	0	0	0	0	1	0	1	0	1	1
d3	0	0	0	0	1	1	0	1	0	0
d1	0	1	0	0	0	1	0	0	0	0

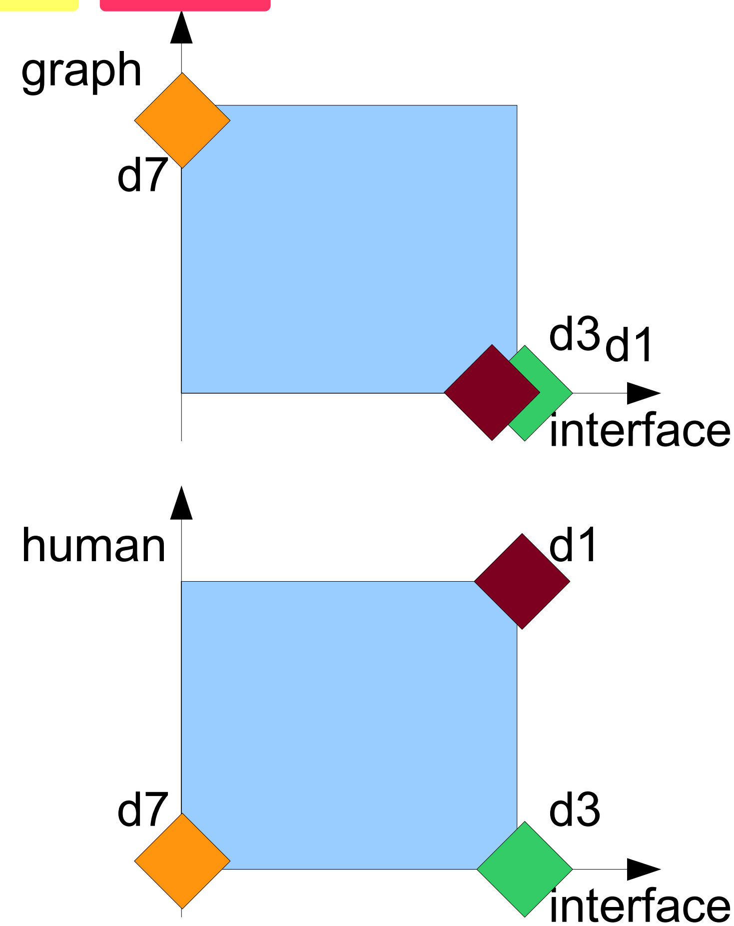
Document dimensions

Distance

Items	minors	human	graph	trees	user	interface	response	eps	survey	time
d3	0	0	0	0	1	1	0	1	0	0
d7	0	0	1	1	0	0	0	0	0	0
d1	0	1	0	0	0	1	0	0	0	0

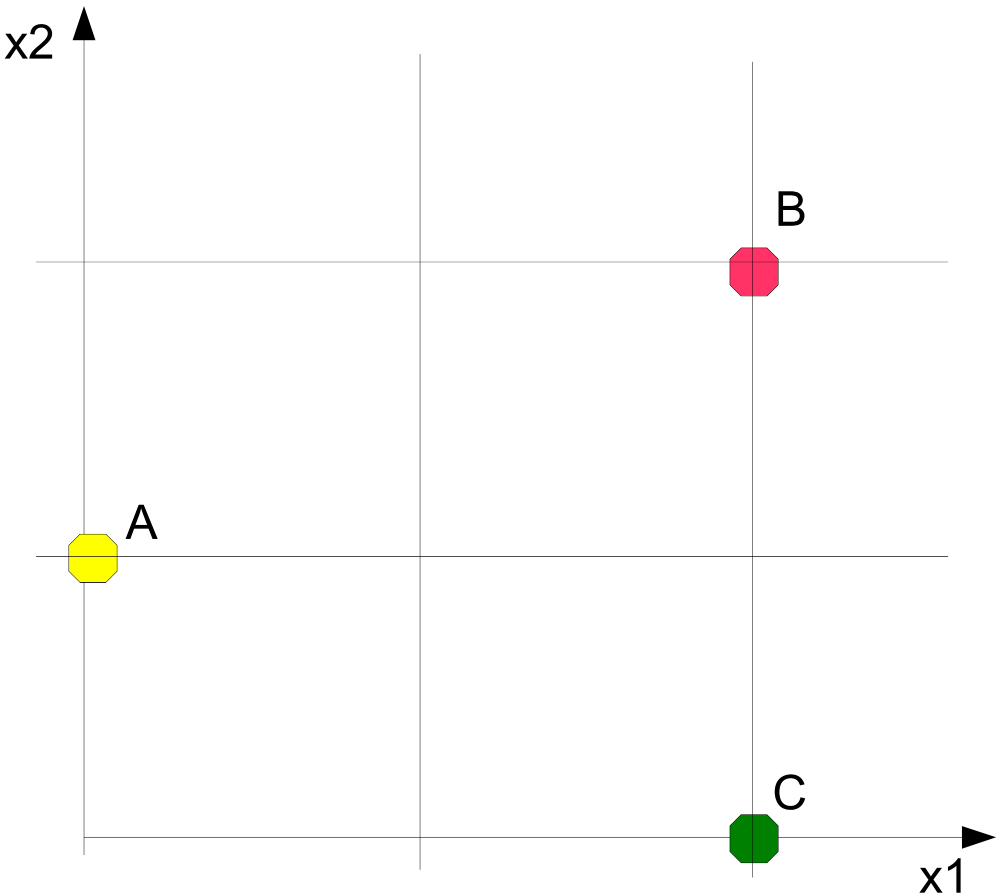


Distance between words



Distance between documents

Distance metrics.
Manhattan distance



dist(A,B)=|Ax1-Bx1|+|Ax2-Bx2|

dist(A,B)=3
dist(B,C)=2

For N dimensions:
dist(A,B)=SUM$_{\text{(i from 1 to N)}}$|Axi-Bxi|

For similarity:
sim(A,B)=1/1+dist(A,B)

Document similarities using Manhattan distance

execfile('manhattan.py')

manhattan distance between documents d8 d9 = 2 and similarity = 0.3333333333

manhattan distance between documents d6 d8 = 2 and similarity = 0.3333333333

manhattan distance between documents d6 d9 = 4 and similarity = 0.2

manhattan distance between documents d6 d7 = 1 and similarity = 0.5

manhattan distance between documents d7 d8 = 1 and similarity = 0.5

manhattan distance between documents d7 d9 = 3 and similarity = 0.25

manhattan distance between documents d4 d8 = 5 and similarity = 0.1666666666

manhattan distance between documents d4 d9 = 5 and similarity = 0.1666666666

manhattan distance between documents d4 d6 = 3 and similarity = 0.25

manhattan distance between documents d4 d7 = 4 and similarity = 0.2

manhattan distance between documents d4 d5 = 5 and similarity = 0.1666666666

manhattan distance between documents d5 d8 = 6 and similarity = 0.1428571428

manhattan distance between documents d5 d9 = 6 and similarity = 0.1428571428

manhattan distance between documents d5 d6 = 4 and similarity = 0.2

Word similarity using Manhattan distance

manhattan distance between words minors trees = 3 and similarity = 0.25

manhattan distance between words minors survey = 2 and similarity = 0.333333333333

manhattan distance between words minors user = 5 and similarity = 0.166666666667

manhattan distance between words minors time = 4 and similarity = 0.2

manhattan distance between words minors response = 4 and similarity = 0.2

manhattan distance between words graph minors = 1 and similarity = 0.5

manhattan distance between words graph trees = 2 and similarity = 0.333333333333

manhattan distance between words graph survey = 3 and similarity = 0.25

manhattan distance between words graph user = 6 and similarity = 0.142857142857

manhattan distance between words graph human = 5 and similarity = 0.166666666667

manhattan distance between words graph time = 5 and similarity = 0.166666666667

manhattan distance between words graph interface = 5 and similarity = 0.166666666667

manhattan distance between words human minors = 4 and similarity = 0.2

manhattan distance between words human trees = 5 and similarity = 0.166666666667

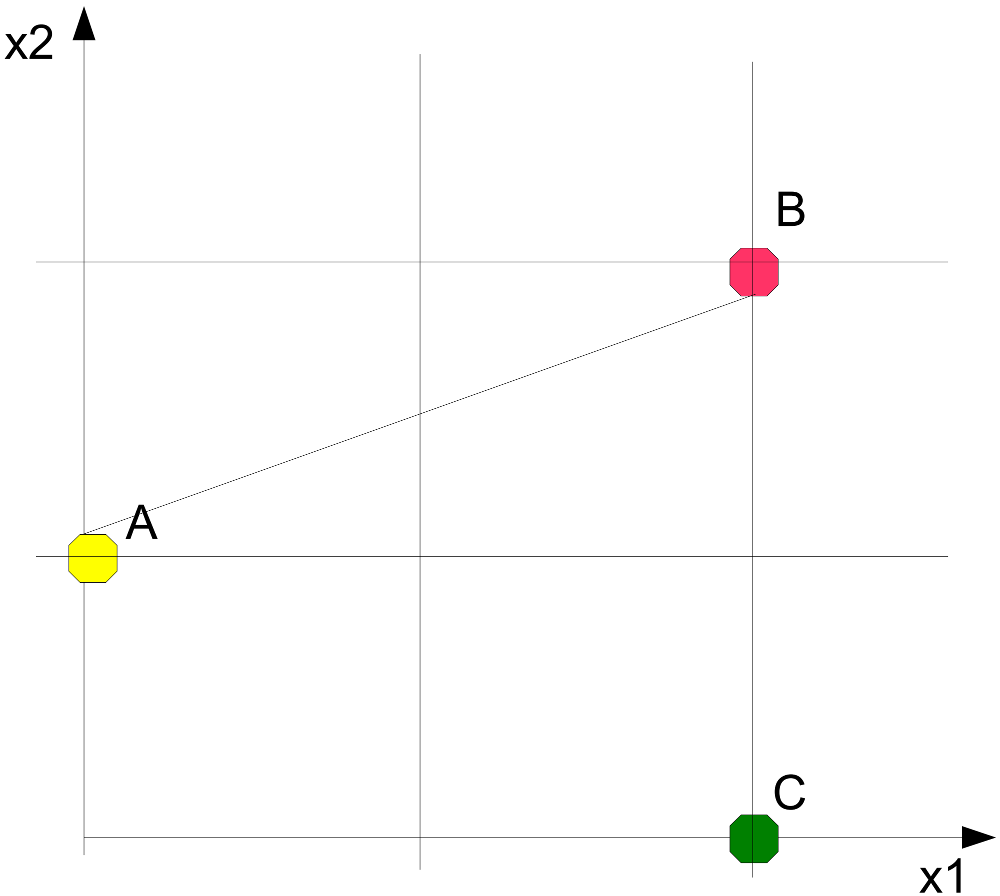
manhattan distance between words human survey = 4 and similarity = 0.2

manhattan distance between words human user = 5 and similarity = 0.166666666667

manhattan distance between words human time = 4 and similarity = 0.2

manhattan distance between words human interface = 2 and similarity = 0.333333333333

Distance metrics. Euclidean distance



dist(A,B)=sqrt ($|Ax1-Bx1|^2+|Ax2-Bx2|^2$)

dist(A,B)=sqrt(5)
dist(B,C)=2

For N dimensions:
dist(A,B)=sqrt ($SUM_{(i\ from\ 1\ to\ N)}|Axi-Bxi|^2$)

For similarity:
sim(A,B)=1/(1+dist(A,B))

Document similarity using Euclidean distance

execfile('euclidean.py')

euclidean distance between documents d8 d9 = 1.41421356237 and similarity =0.414

euclidean distance between documents d6 d8 = 1.41421356237 and similarity =0.414

euclidean distance between documents d6 d9 = 2.0 and similarity = 0.333333

euclidean distance between documents d6 d7 = 1.0 and similarity = 0.5

euclidean distance between documents d7 d8 = 1.0 and similarity = 0.5

euclidean distance between documents d7 d9 = 1.73205080757 and similarity =0.366

euclidean distance between documents d4 d8 = 2.2360679775 and similarity = 0.309

euclidean distance between documents d4 d9 = 2.2360679775 and similarity = 0.309

euclidean distance between documents d4 d6 = 1.73205080757 and similarity =0.366

euclidean distance between documents d4 d7 = 2.0 and similarity = 0.33333

euclidean distance between documents d4 d5 = 2.2360679775 and similarity = 0.309

euclidean distance between documents d5 d8 = 2.44948974278 and similarity =0.289

euclidean distance between documents d5 d9 = 2.44948974278 and similarity =0.289

euclidean distance between documents d5 d6 = 2.0 and similarity = 0.3333

euclidean distance between documents d5 d7 = 2.2360679775 and similarity = 0.309

Word similarity using Euclidean distance

euclidean distance between words minors trees = 1.73205080757 and similarity = 0.366025403784

euclidean distance between words minors survey = 1.41421356237 and similarity = 0.414213562373

euclidean distance between words minors user = 2.2360679775 and similarity = 0.309016994375

euclidean distance between words minors time = 2.0 and similarity = 0.333333333333

euclidean distance between words minors response = 2.0 and similarity = 0.333333333333

euclidean distance between words graph minors = 1.0 and similarity = 0.5

euclidean distance between words graph trees = 1.41421356237 and similarity= 0.414213562373

euclidean distance between words graph survey = 1.73205080757 and similarity = 0.366025403784

euclidean distance between words graph user = 2.44948974278 and similarity = 0.289897948557

euclidean distance between words graph human = 2.2360679775 and similarity = 0.309016994375

euclidean distance between words graph time = 2.2360679775 and similarity =0.309016994375

euclidean distance between words graph interface = 2.2360679775 and similarity = 0.309016994375

euclidean distance between words human minors = 2.0 and similarity = 0.333333333333

euclidean distance between words human trees = 2.2360679775 and similarity = 0.309016994375

euclidean distance between words human survey = 2.0 and similarity = 0.333333333333

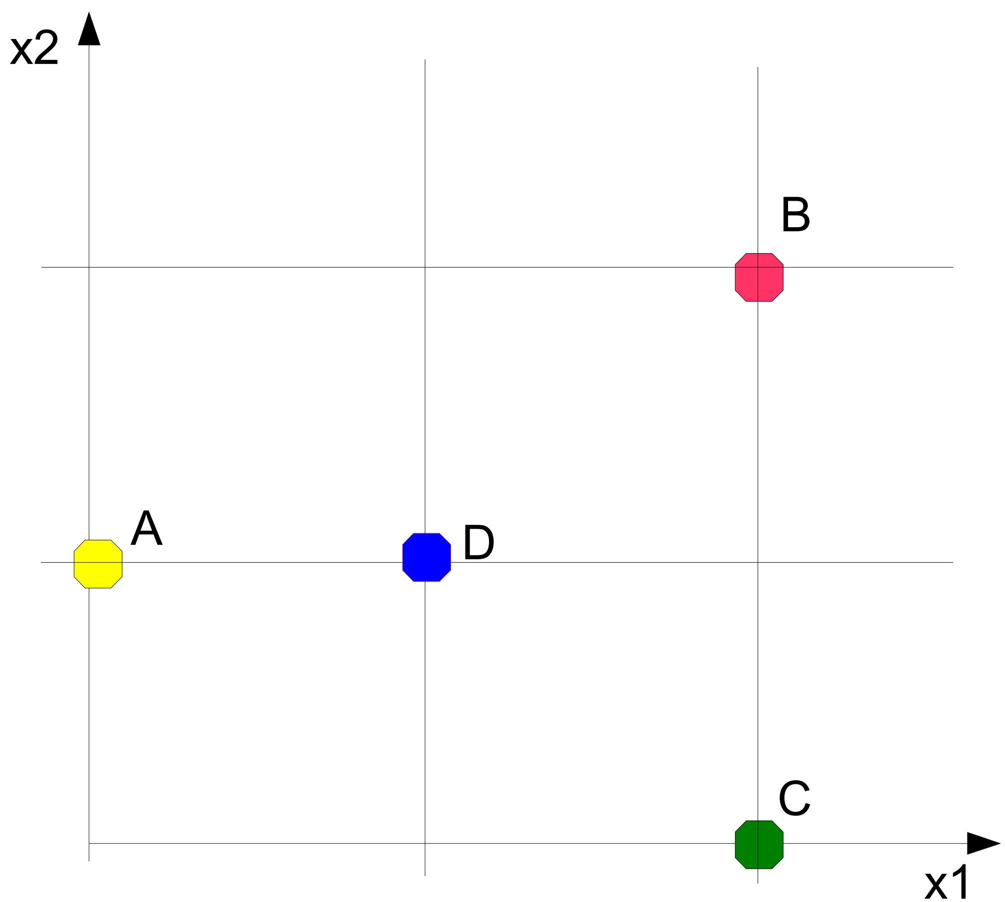
euclidean distance between words human user = 2.2360679775 and similarity =0.309016994375

euclidean distance between words human time = 2.0 and similarity = 0.333333333333

euclidean distance between words human interface = 1.41421356237 and similarity = 0.414213562373

euclidean distance between words human response = 2.0 and similarity = 0.333333333333

Distance metrics. Pearson correlation

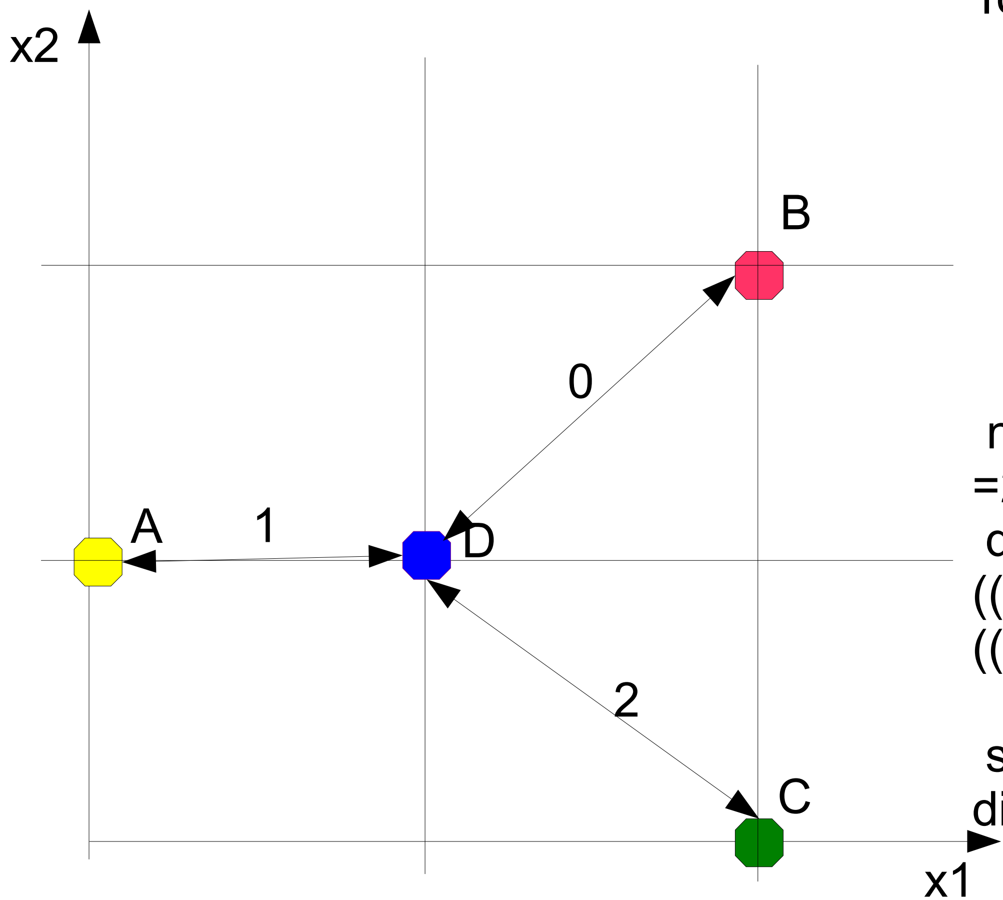


A correlation is a number between -1 and +1 that measures the degree of association between two variables A positive value for the correlation implies a positive association (large values of x1 tend to be associated with large values of x2 and small values of x1 tend to be associated with small values of x2). A negative value for the correlation implies a negative or inverse association

D and B are perfectly correlated in dimensions x1,x2.
Pearson coefficient is 1.0
sim(D,B)=1
dist(D,B)=1-sim(D,B)=0

D and C are perfectly uncorrelated in dimensions x1,x2.
Pearson coefficient is -1.0
sim(D,B)=-1
dist(D,B)=1-(-1)=2

Distance metrics. Pearson correlation



To calculate correlation (A,D):

1. SumA=0+1=1
2. SumD=1+1=2
3. SumAA=0^2+1^2=1
4. SumDD=1^2+1^2=2
5. SumAD=0x1+1x1=1
6. The dimensions (N=2)

numerator= (N x SumAD) – (SumA x SumD) =2x1-1x2=0
denominator = sqrt [((N*SumAA)-(SumA*SumA)) x ((N*SumDD)-(SumD*SumD))]

sim(A,D)=numerator/denominator=0
dist(A,D)=1-sim(A,D)=1

Document similarity using Pearson distance

execfile('pearson.py')

pearson distance between documents d8   d9 = 0.047619047619 and similarity = 0.952380952381

pearson distance between documents d6   d8 = 0.272607032547 and similarity = 0.727392967453

pearson distance between documents d6   d9 = 1.0 and similarity = 0.0

pearson distance between documents d6   d7 = 0.166666666667 and similarity = 0.833333333333

pearson distance between documents d7   d8 = 0 and similarity = 1.0

pearson distance between documents d7   d9 = 0.45445527441 and similarity = 0.54554472559

pearson distance between documents d4   d8 = 1.0 and similarity = 0.0

pearson distance between documents d4   d9 = 1.0 and similarity = 0.0

pearson distance between documents d4   d6 = 1.0 and similarity = 0.0

pearson distance between documents d4   d7 = 1.0 and similarity = 0.0

pearson distance between documents d4   d5 = 1.0 and similarity = 0.0

pearson distance between documents d5   d8 = 1.0 and similarity = 0.0

pearson distance between documents d5   d9 = 1.0 and similarity = 0.0

pearson distance between documents d5   d6 = 1.0 and similarity = 0.0

pearson distance between documents d5   d7 = 1.0 and similarity = 0.0

Word similarity using Pearson distance

pearson distance between words minors trees = 0.433053290486 and similarity= 0.566946709514

pearson distance between words minors survey = 0.357142857143 and similarity= 0.642857142857

pearson distance between words minors user = 1.0 and similarity = 0.0

pearson distance between words minors time = 1.0 and similarity = 0.0

pearson distance between words minors response = 1.0 and similarity = 0.0

pearson distance between words graph minors = 0 and similarity = 1.0

pearson distance between words graph trees = 0.5 and similarity = 0.5

pearson distance between words graph survey = 0.433053290486 and similarity= 0.566946709514

pearson distance between words graph user = 1.5 and similarity = -0.5

pearson distance between words graph human = 1.0 and similarity = 0.0

pearson distance between words graph interface = 1.0 and similarity = 0.0

pearson distance between words graph response = 1.0 and similarity = 0.0

pearson distance between words human minors = 1.0 and similarity = 0.0

pearson distance between words human trees = 1.0 and similarity = 0.0

pearson distance between words human survey = 1.0 and similarity = 0.0

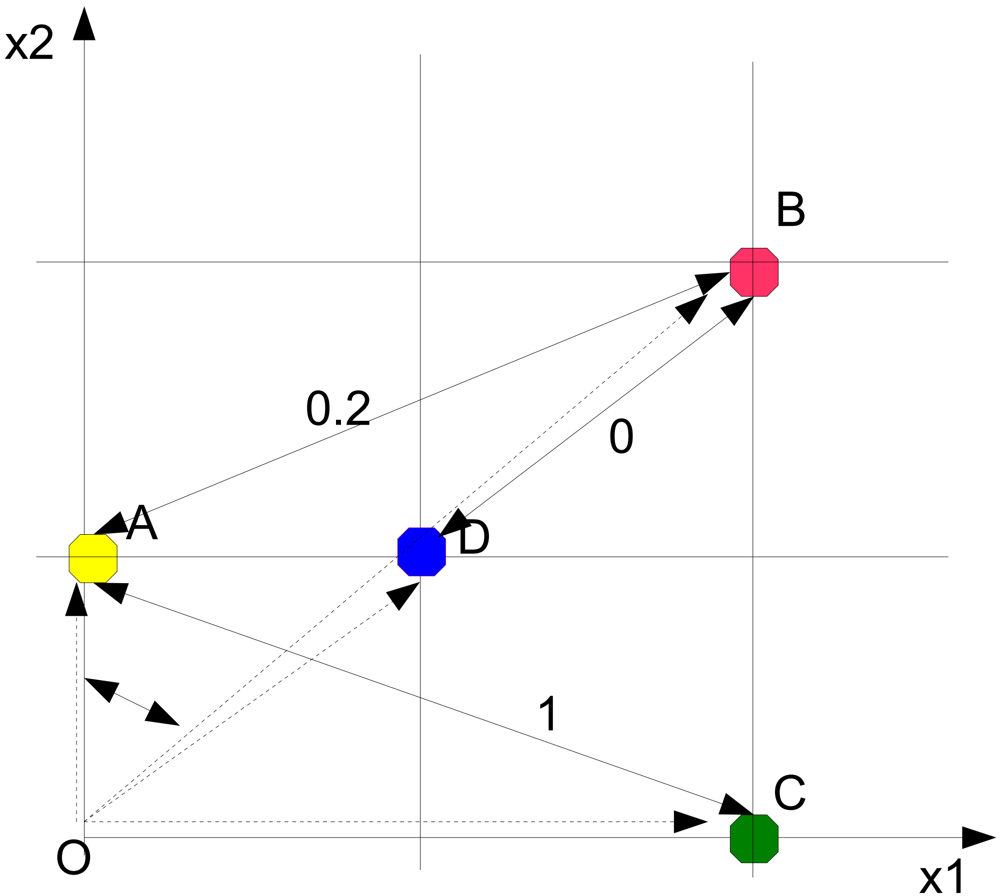
pearson distance between words human user = 1.0 and similarity = 0.0

pearson distance between words human time = 1.0 and similarity = 0.0

pearson distance between words human interface = 0.357142857143 and similarity =0.642857142857

pearson distance between words human response = 1.0 and similarity = 0.0

Distance metrics. Cosine similarity



sim(**A**,**B**)=cosine(AOB)=
(**A***B**)/(|**A**|*|**B**|)

The bigger cosine, the less is the angle, the more similar are 2 vectors

sim(A,B)=[0*2+1*2]/[1*sqrt(8)] =0.71
dist(A,B)=1-sim(A,B)=0.29

sim(D,B)=(1*2+1*2)/(sqrt(2)*sqrt(8))
=4/4=1.0
dist(D,B)=0

sim(A,C)=(0*2+2*0)/(1*2)=0
dist(A,C)=1

Document similarity using Cosine similarity

execfile('cosine.py')

cosine distance between documents d8   d9 = 0.333333333333 and similarity = 0.666666666667

cosine distance between documents d6   d8 = 0.42264973081 and similarity = 0.57735026919

cosine distance between documents d6   d9 = 1.0 and similarity = 0.0

cosine distance between documents d6   d7 = 0.292893218813 and similarity = 0.707106781187

cosine distance between documents d7   d8 = 0.183503419072 and similarity = 0.816496580928

cosine distance between documents d7   d9 = 0.591751709536 and similarity = 0.408248290464

cosine distance between documents d4   d8 = 1.0 and similarity = 0.0

cosine distance between documents d4   d9 = 1.0 and similarity = 0.0

cosine distance between documents d4   d6 = 1.0 and similarity = 0.0

cosine distance between documents d4   d7 = 1.0 and similarity = 0.0

cosine distance between documents d4   d5 = 1.0 and similarity = 0.0

cosine distance between documents d5   d8 = 1.0 and similarity = 0.0

cosine distance between documents d5   d9 = 1.0 and similarity = 0.0

cosine distance between documents d5   d6 = 1.0 and similarity = 0.0

cosine distance between documents d5   d7 = 1.0 and similarity = 0.0.

Word similarity using Cosine similarity

cosine distance between words minors   trees = 0.591751709536 and similarity = 0.408248290464

cosine distance between words minors   survey = 0.5 and similarity = 0.5

cosine distance between words minors   user = 1.0 and similarity = 0.0

cosine distance between words minors   time = 1.0 and similarity = 0.0

cosine distance between words minors   response = 1.0 and similarity = 0.0

cosine distance between words graph   minors = 0.183503419072 and similarity = 0.816496580928

cosine distance between words graph   trees = 0.333333333333 and similarity =0.666666666667

cosine distance between words graph   user = 1.0 and similarity = 0.0

cosine distance between words graph   human = 1.0 and similarity = 0.0

cosine distance between words graph   time = 1.0 and similarity = 0.0

cosine distance between words graph   interface = 1.0 and similarity = 0.0

cosine distance between words graph   response = 1.0 and similarity = 0.0

cosine distance between words human   minors = 1.0 and similarity = 0.0

cosine distance between words human   trees = 1.0 and similarity = 0.0

cosine distance between words human   survey = 1.0 and similarity = 0.0

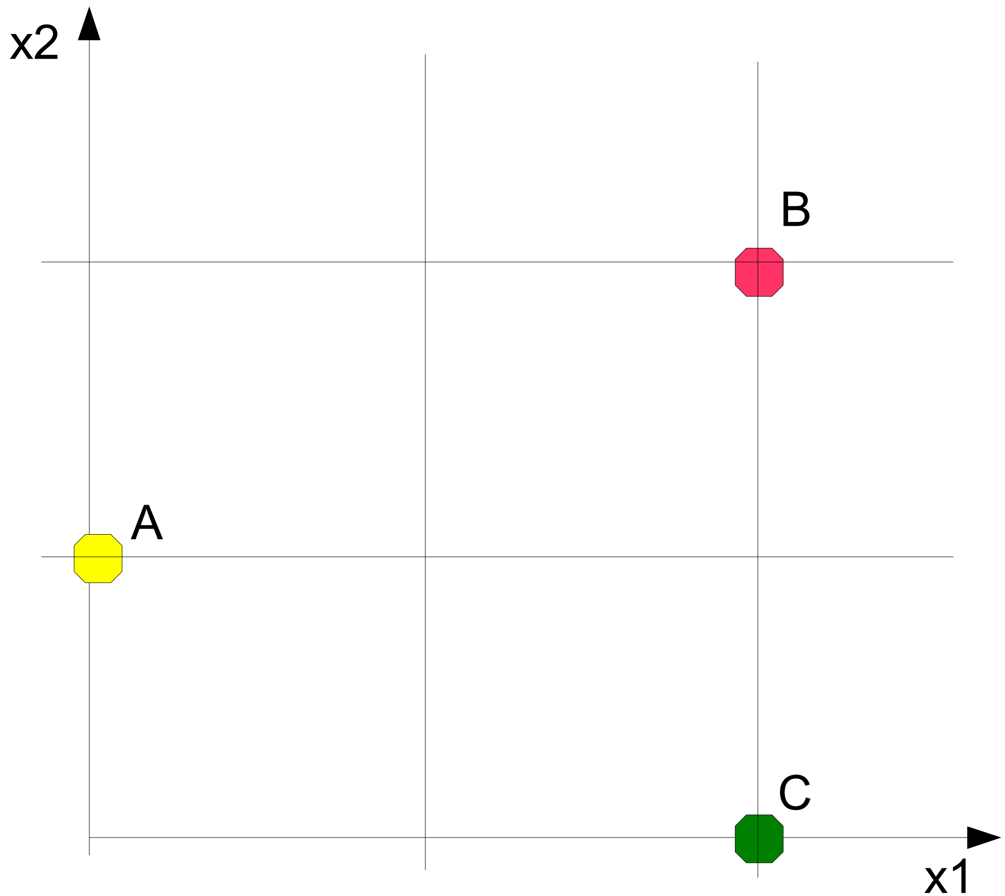
cosine distance between words human   user = 1.0 and similarity = 0.0

cosine distance between words human   time = 1.0 and similarity = 0.0

cosine distance between words human   interface = 0.5 and similarity = 0.5

cosine distance between words human   response = 1.0 and similarity = 0.0

Distance metrics. Tanimoto coefficient



$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$

Jaccard coefficient

Tanimoto coefficient is a Jaccard coefficient for binary attributes

$$T(A,B) = \frac{A \cdot B}{\|A\|^2 + \|B\|^2 - A \cdot B}.$$

sim(A,B)=[0*2+1*2]/[1+8-(0*2+1*2)]
=2/7=0.285

dist(A,B)=1-sim(A,B)=0.715

Document similarity using Tanimoto coefficient

execfile('tanimoto.py')

tanimoto distance between documents d8   d9 = 0.5 and similarity = 0.5

tanimoto distance between documents d6   d8 = 0.666666666667 and similarity =0.333333333333

tanimoto distance between documents d6   d9 = 1.0 and similarity = 0.0

tanimoto distance between documents d6   d7 = 0.5 and similarity = 0.5

tanimoto distance between documents d7   d8 = 0.333333333333 and similarity =0.666666666667

tanimoto distance between documents d7   d9 = 0.75 and similarity = 0.25

tanimoto distance between documents d4   d8 = 1.0 and similarity = 0.0

tanimoto distance between documents d4   d9 = 1.0 and similarity = 0.0

tanimoto distance between documents d4   d6 = 1.0 and similarity = 0.0

tanimoto distance between documents d4   d7 = 1.0 and similarity = 0.0

tanimoto distance between documents d4   d5 = 1.0 and similarity = 0.0

tanimoto distance between documents d5   d8 = 1.0 and similarity = 0.0

tanimoto distance between documents d5   d9 = 1.0 and similarity = 0.0

tanimoto distance between documents d5   d6 = 1.0 and similarity = 0.0

tanimoto distance between documents d5   d7 = 1.0 and similarity = 0.0

Word similarity using Tanimoto coefficient

tanimoto distance between words minors trees = 0.75 and similarity = 0.25

tanimoto distance between words minors survey = 0.666666666667 and similarity = 0.333333333333

tanimoto distance between words minors user = 1.0 and similarity = 0.0

tanimoto distance between words minors response = 1.0 and similarity = 0.0

tanimoto distance between words graph minors = 0.333333333333 and similarity = 0.666666666667

tanimoto distance between words graph trees = 0.5 and similarity = 0.5

tanimoto distance between words graph survey = 0.75 and similarity = 0.25

tanimoto distance between words graph user = 1.0 and similarity = 0.0

tanimoto distance between words graph time = 1.0 and similarity = 0.0

tanimoto distance between words graph interface = 1.0 and similarity = 0.0

tanimoto distance between words graph response = 1.0 and similarity = 0.0

tanimoto distance between words human minors = 1.0 and similarity = 0.0

tanimoto distance between words human trees = 1.0 and similarity = 0.0

tanimoto distance between words human survey = 1.0 and similarity = 0.0

tanimoto distance between words human user = 1.0 and similarity = 0.0

tanimoto distance between words human time = 1.0 and similarity = 0.0

tanimoto distance between words human interface = 0.666666666667 and similarity = 0.333333333333

tanimoto distance between words human response = 1.0 and similarity = 0.0

What is the best distance metric for clustering documents?

manhattan distance between documents d7   d8 = 1 and similarity = 0.5

manhattan distance between documents d4   d8 = 5 and similarity = 0.1666666666

euclidean distance between documents d7   d8 = 1.0 and similarity = 0.5

euclidean distance between documents d4   d8 = 2.2360679775 and similarity = 0.309

pearson distance between documents d7   d8 = 0 and similarity = 1.0

pearson distance between documents d4   d8 = 1.0 and similarity = 0.0

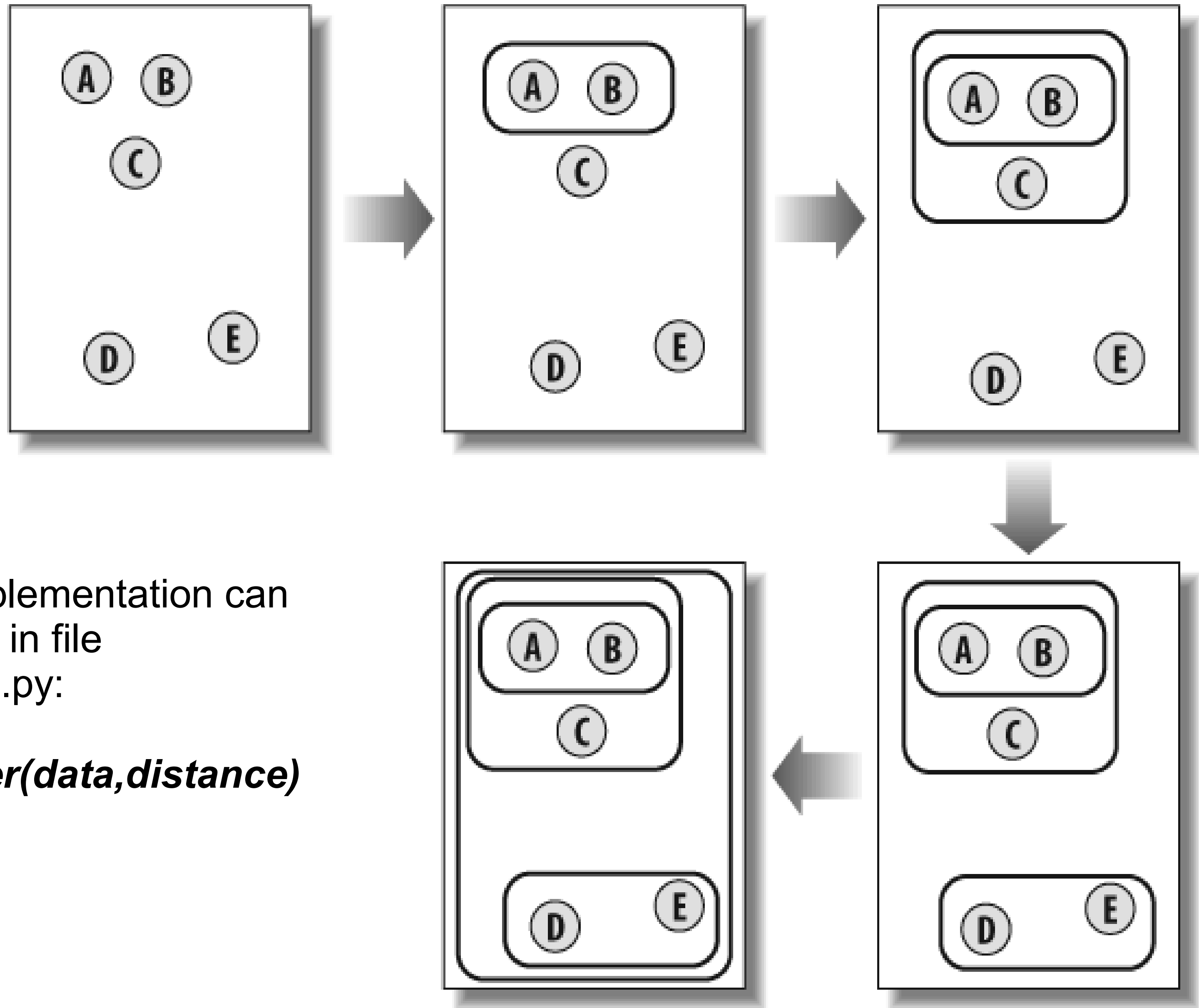
cosine distance between documents d7   d8 = 0.183503419072 and similarity = 0.816496580928

cosine distance between documents d4   d8 = 1.0 and similarity = 0.0

tanimoto distance between documents d7   d8 = 0.333333333333 and similarity =0.666666666667

tanimoto distance between documents d4   d8 = 1.0 and similarity = 0.0

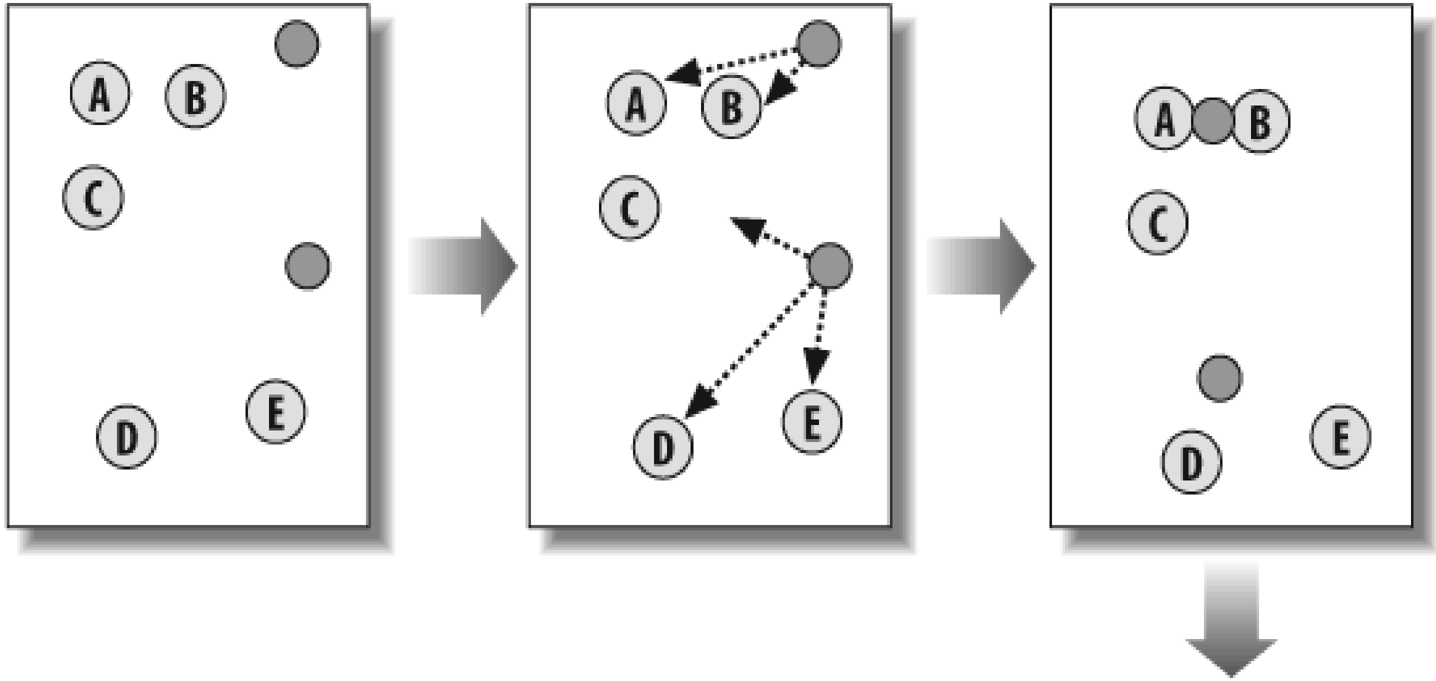
Hierarchical clustering



The implementation can be read in file clusters.py:

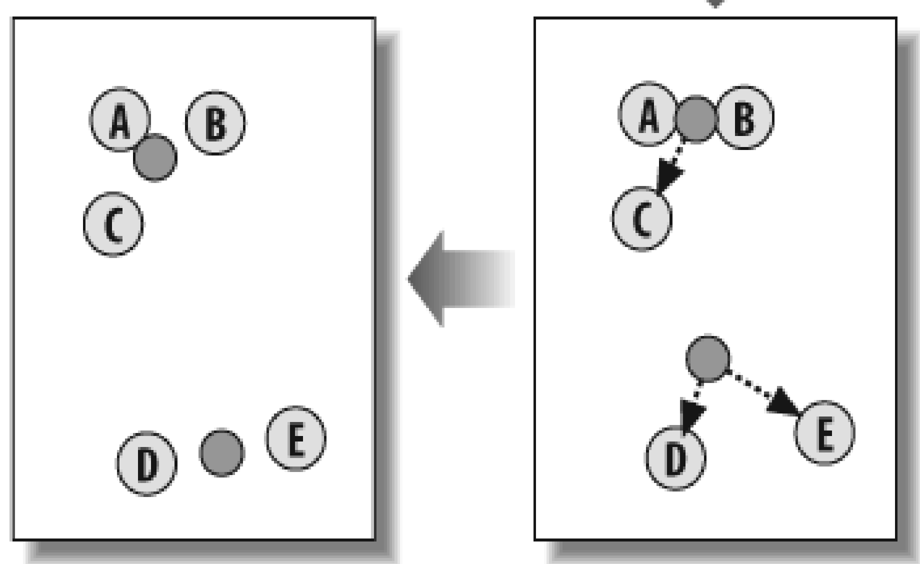
hcluster(data,distance)

K-means clustering



The implementation can be read in file clusters.py:

kcluster(data,distance,k):



Clustering titles

Read file 'hclustertitles.py'

```
import clusters
docs,words,data=clusters.readfile('titlesdata.txt')

clust=clusters.hcluster(data,distance=clusters.pearson)
print 'clusters by pearson correlation'
clusters.printclust(clust,labels=docs)
clusters.drawdendrogram(clust,docs,jpeg='docsclustpearson.jpg')
```

```
execfile('hclustertitles.py')
```

Read file 'kmlustertitles.py'

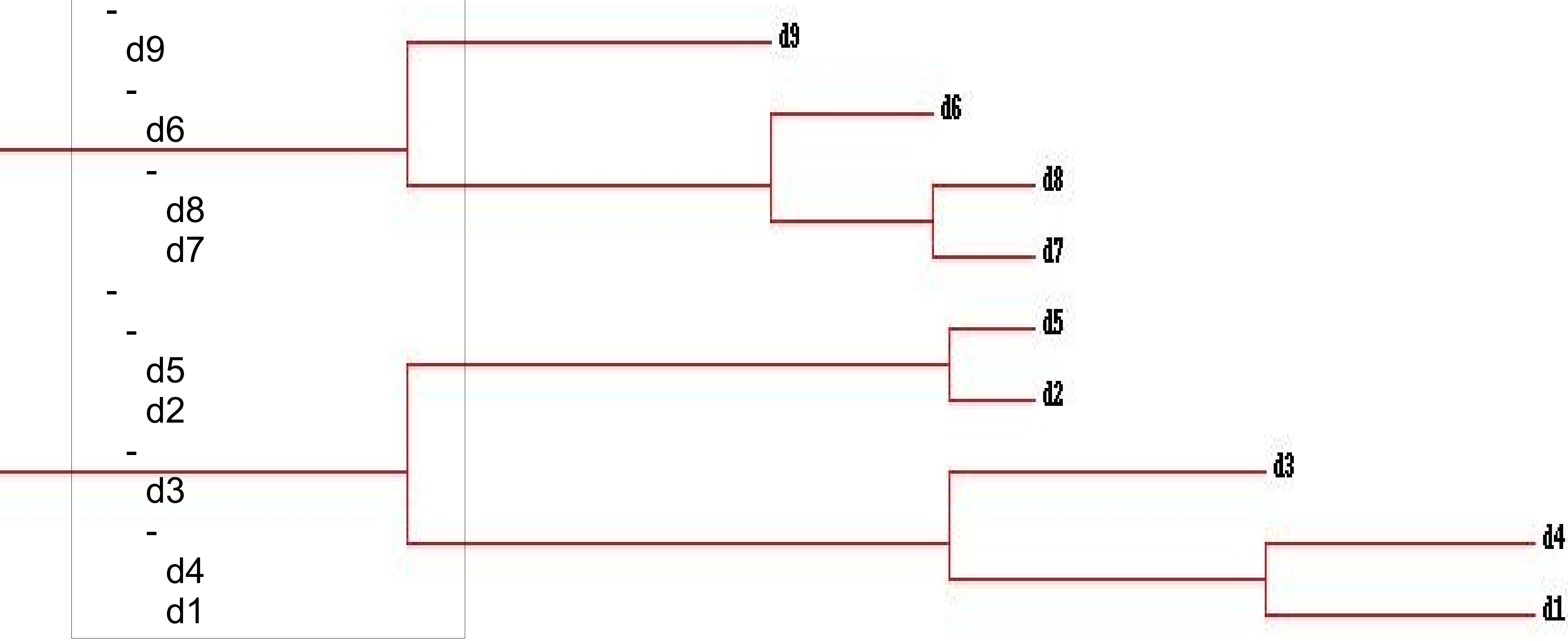
```
docs,words,data=clusters.readfile('titlesdata.txt')

print '2 clusters:'
clust=clusters.kcluster(data,distance=clusters.pearson,k=2)
print 'clusters by pearson correlation'
print 'cluster 1:'
print [docs[r] for r in clust[0]]
print 'cluster 2:'
print [docs[r] for r in clust[1]]
```

```
execfile('kmclustertitles.py')
```

Hierarchical clustering results for titles

clusters by pearson correlation

-
  -
    d9
    -
      d6
      -
        d8
        d7
  -
    -
      d5
      d2
    -
      d3
      -
        d4
        d1



Clustering words

Read file 'hclusterwords.py'

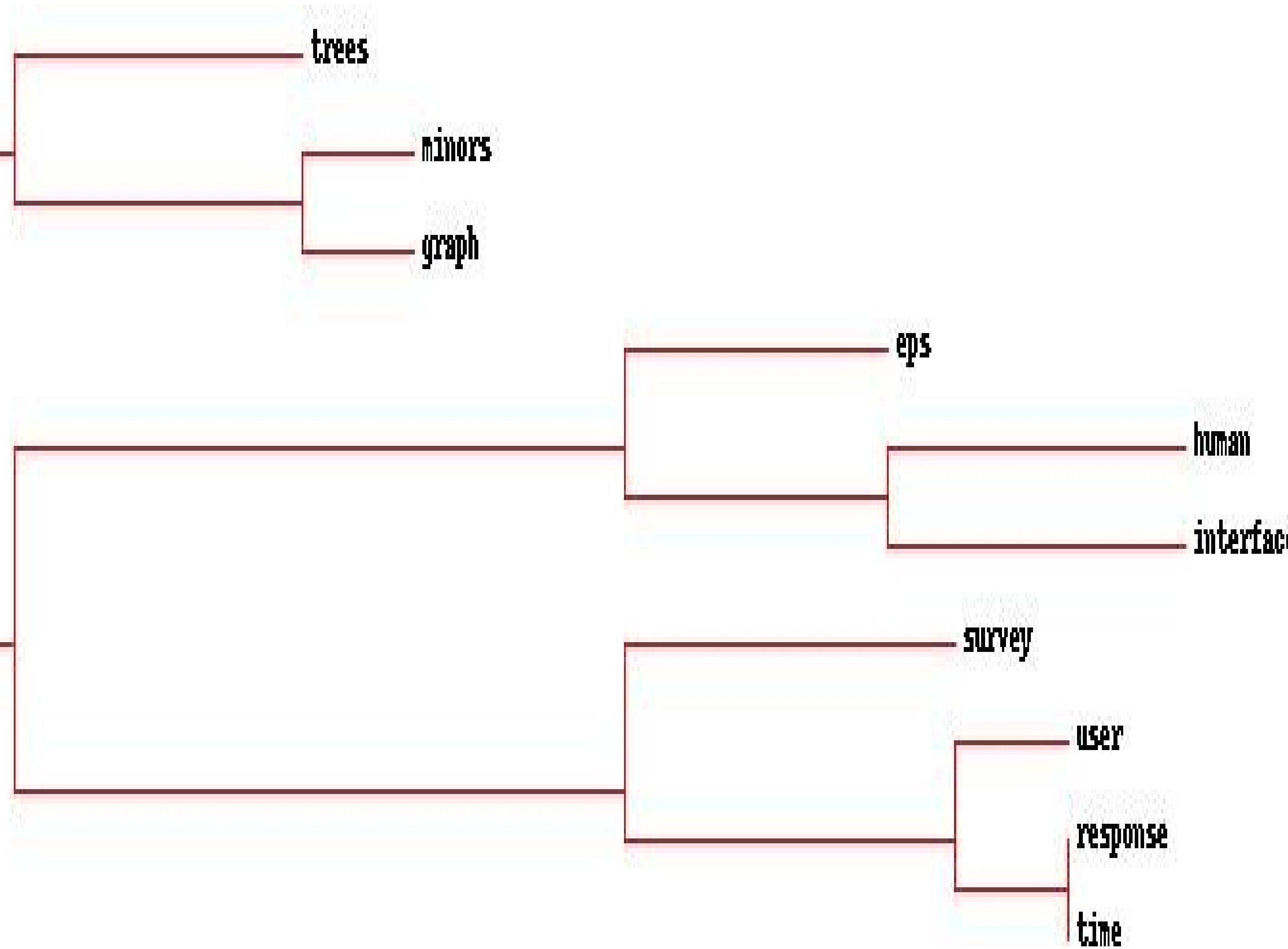
```
import clusters
docs,words,data=clusters.readfile('titlesdata.txt')
rdata=clusters.rotatematrix(data)

clust=clusters.hcluster(rdata,distance=clusters.pearson)
print 'clusters by pearson correlation'
clusters.printclust(clust,labels=words)
clusters.drawdendrogram(clust,words,jpeg='wordsclustpearson.jpg')
```

Read file 'kmclusterwords.py'

```
idocs,words,data=clusters.readfile('titlesdata.txt')
rdata=clusters.rotatematrix(data)

print '3 clusters:'
clust=clusters.kcluster(rdata,distance=clusters.pearson,k=3)
print 'clusters by pearson correlation'
print 'cluster 1:'
print [words[r] for r in clust[0]]
print 'cluster 2:'
print [words[r] for r in clust[1]]
print 'cluster 3:'
print [words[r] for r in clust[2]]
```

Hierarchical clustering results for words

clusters by pearson correlation
-
  -
    trees
    -
      minors
      graph
  -
    -
      eps
      -
        human
        interface
    -
      survey
      -
        user
        -
          response
          time



Task 1. Improve k-mean clustering results

- Try to improve k-mean clustering results on words and titles

Task 2. Download blog data

- Almost all blogs can be read online or via their RSS feeds. An RSS feed is a simple XML document that contains information about the blog and all the entries. The first step in generating word counts for each blog is to parse these feeds. Fortunately, there is an excellent module for doing this called Universal Feed Parser, which you can download from http://www.feedparser.org
- This module makes it easy to get the title, links, and entries from any RSS or Atom feed. The next step is to create a function that will extract all the words from a feed.
- execfile ("generatefeedvector.py")

  OR
- use file "blogdata1.txt"

Task 3. Perform hierarchical clustering on blogs

- Create and execute script similar to 'hclustertitles.py' for word-document matrix of file 'blogdata1.txt'

Task 4. Clustering of preferences

- File 'zebo.txt' contains list of items people would like to have. This list has been downloaded from ***zebo.com*** WEB site.
- Perform hierarchical clustering on 'zebo.txt' data.
- What groups of items people want?