İSTANBUL ŞEHİR UNIVERSITY

ENGR 212: Programming Practice

Week 7

İSTANBUL ŞEHİR UNIVERSITY

Structuring

&

Visualization

Example

İSTANBUL ŞEHİR UNIVERSITY

	"china"	"kids"	"music"	"yahoo"
Gothamist	0	3	3	0
GigaOM	6	0	0	2
Quick Online Tips	0	2	2	22

Preprocessing

İSTANBUL ŞEHİR ÜNİVERSİTY

- 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. Universal Feed Parser is an excellent module. Install via:
- sudo pip install feedparser

Install feedparser

İSTANBUL ŞEHİR ÜNİVERSİTY

- PyCharm (like any other module installation)

or

- pip install feedparser

Sample Data Set

İSTANBUL ŞEHİR UNIVERSITY

- Highly referenced blogs with clean data (mostly text)
  - feedlist.txt
  - Available on LMS (/Lectures/Week 06/Code)

RSS Feed or Atom Feed

İSTANBUL ŞEHİR UNIVERSITY

- RSS and Atom feeds always have a title and a list of entries.
- Each entry usually has either a summary or description tag that contains the actual text of the entries.

Get word counts

İSTANBUL ŞEHİR ÜNİVERSİTY

```
# Returns title and dictionary of word counts for an RSS feed
def getwordcounts(url):
    # Parse the feed
    d =feedparser.parse(url)
    wc ={}

    # Loopover  all the entries
    for e in d.entries:
        if 'summary' in e:
            summary = e.summary
        else:
            summary = e.description

        # Extract a list of words
        words = getwords(e.title+' '+summary)
        for wordin  words:
            wc.setdefault(word,0)
            wc[word] += 1
    return d.feed.title, wc
```

Tokenize: Get Words

İSTANBUL ŞEHİR ÜNIVERSITY

```
import re

# Strips out all of the HTML and splits the words by
# nonalphabetical characters and returns them as a list.

def getwords(html):
    # Remove all the HTML tags
    txt=re.compile(r'<[^>]+>').sub('',html)

    # Split words by all non-alpha characters
    words=re.compile(r'[^A-Z^a-z]+').split(txt)

    # Convert to lowercase
    return [word.lower() for word in words if word!='']
```

Generate word counts

İSTANBUL ŞEHİR UNIVERSITY

```
apcount={}
wordcounts={}

feedlist=[line for line in file('feedlist.txt')]

for feedurl in feedlist:
    title,wc=getwordcounts(feedurl)
    wordcounts[title]=wc
    for word,count in wc.items():
      apcount.setdefault(word,0)
      if count>1:
        apcount[word]+=1

wordlist=[]
for w,bc in apcount.items():
  frac=float(bc)/len(feedlist)
  if frac>0.1 and frac<0.5:
    wordlist.append(w)
```

Compute word appearance counts

Eliminate common & rare words

Create word matrix

İSTANBUL ŞEHİR UNIVERSITY

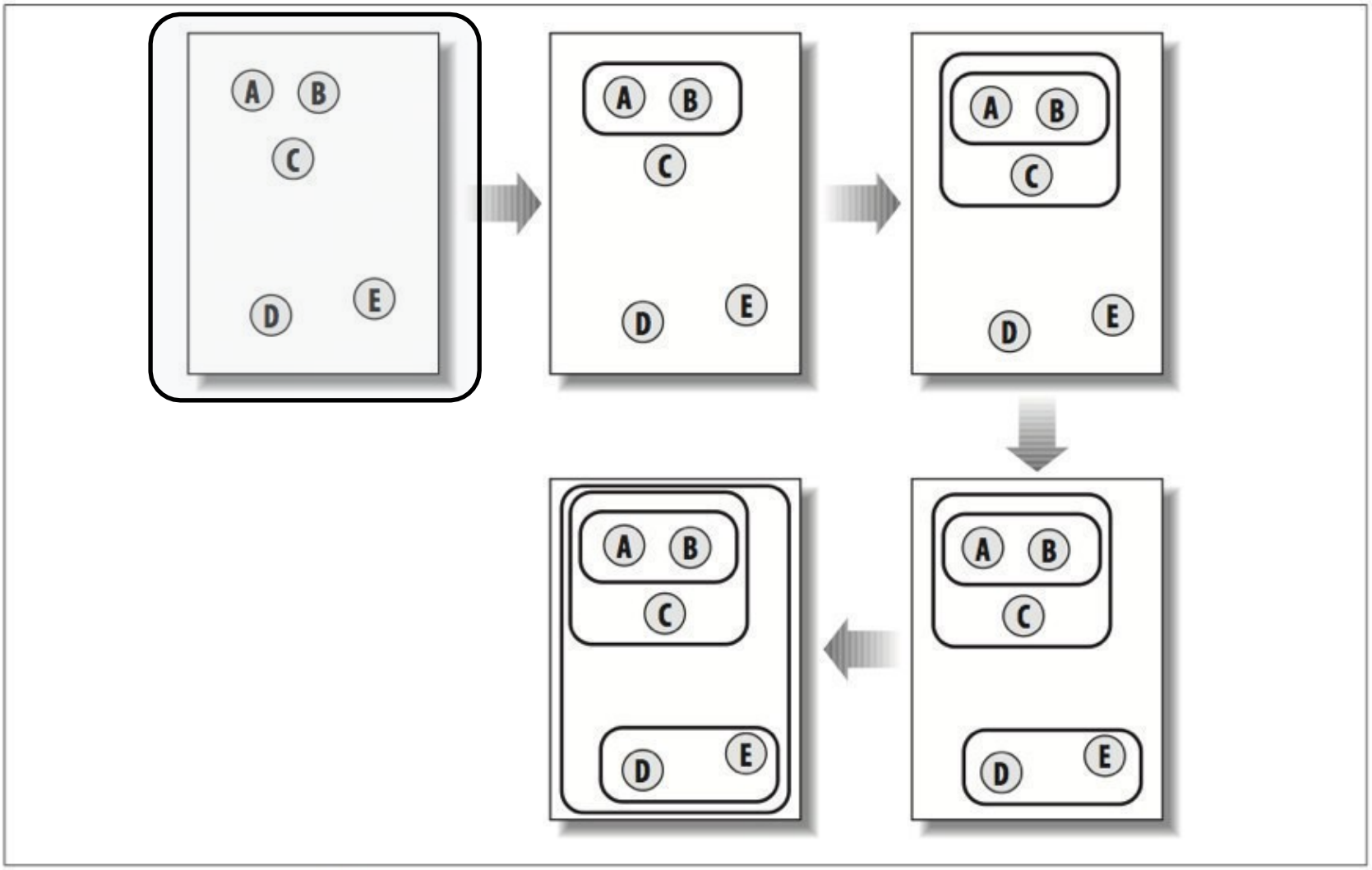
	"china"	"kids"	"music"	"yahoo"
Gothamist	0	3	3	0
GigaOM	6	0	0	2
Quick Online Tips	0	2	2	22

```
out=file('blogdata.txt','w')
out.write('Blog')
for word in wordlist:
  out.write('\t%s' % word)
out.write('\n')

for blog,wc in wordcounts.items():
  print blog
  out.write(blog)
  for word in wordlist:
    if word in wc: out.write('\t%d' % wc[word])
    else: out.write('\t0')
  out.write('\n')
```

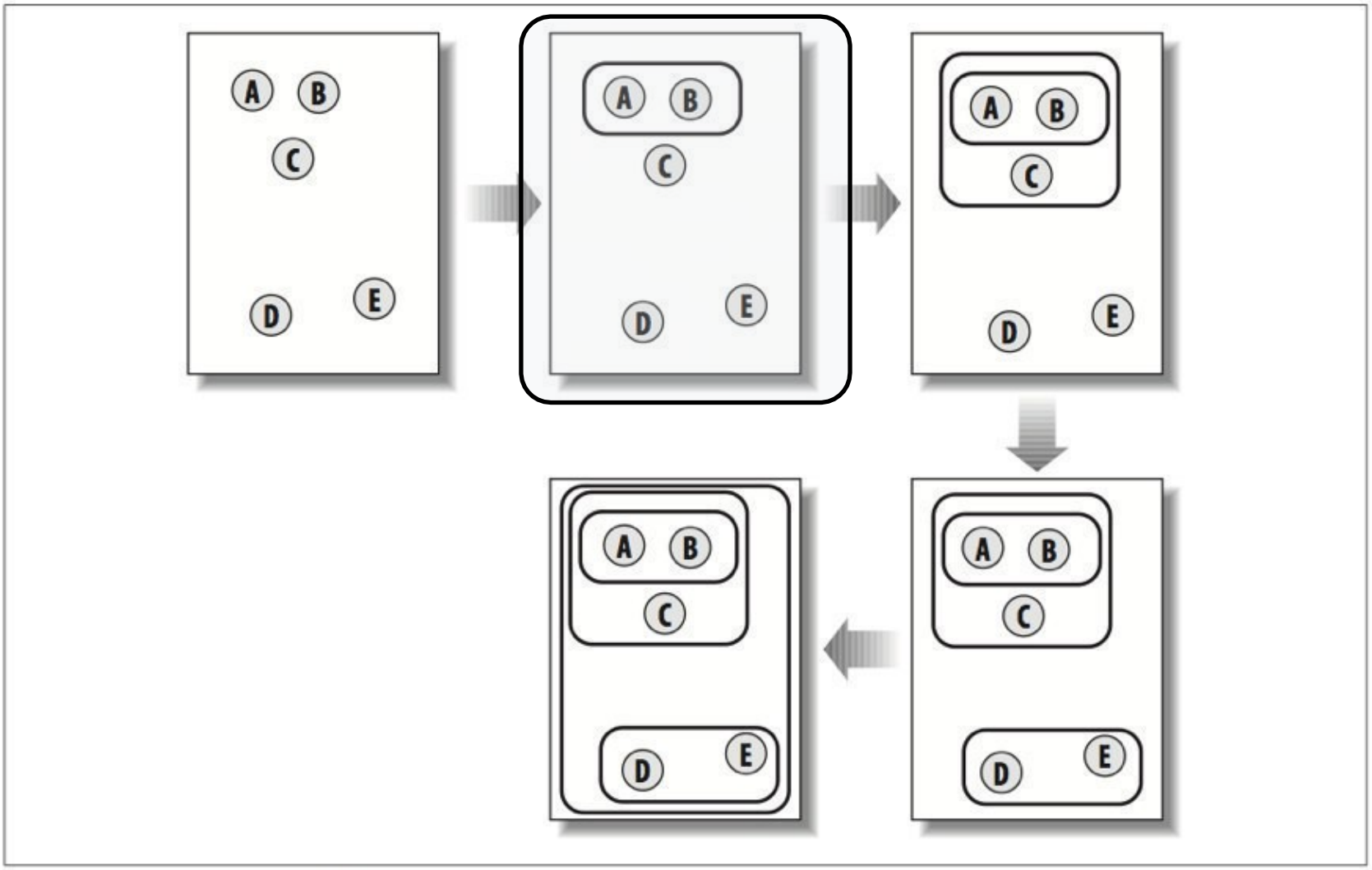
Hierarchical Clustering

İSTANBUL ŞEHİR UNIVERSITY



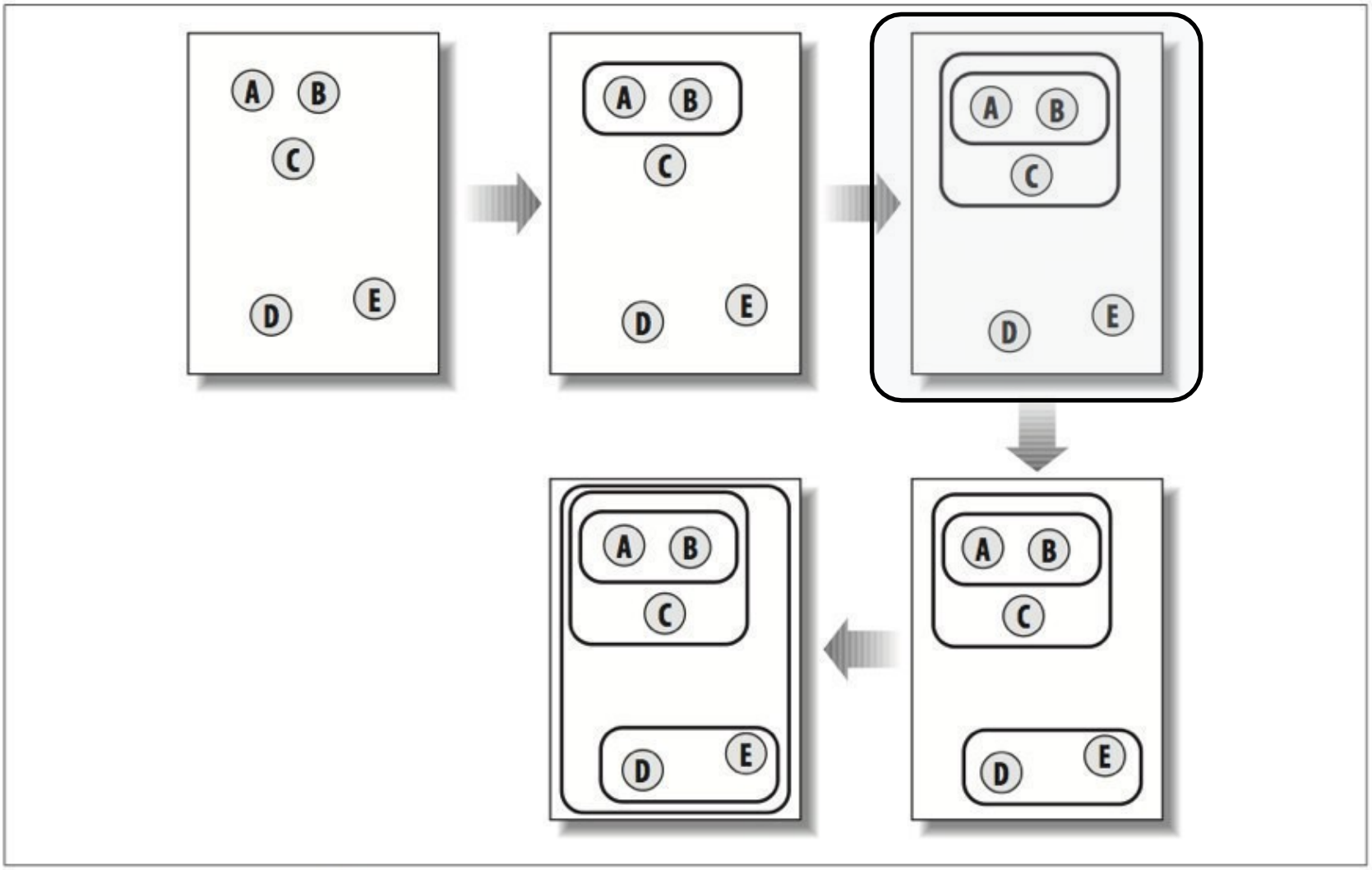
Hierarchical Clustering

İSTANBUL ŞEHİR UNIVERSITY



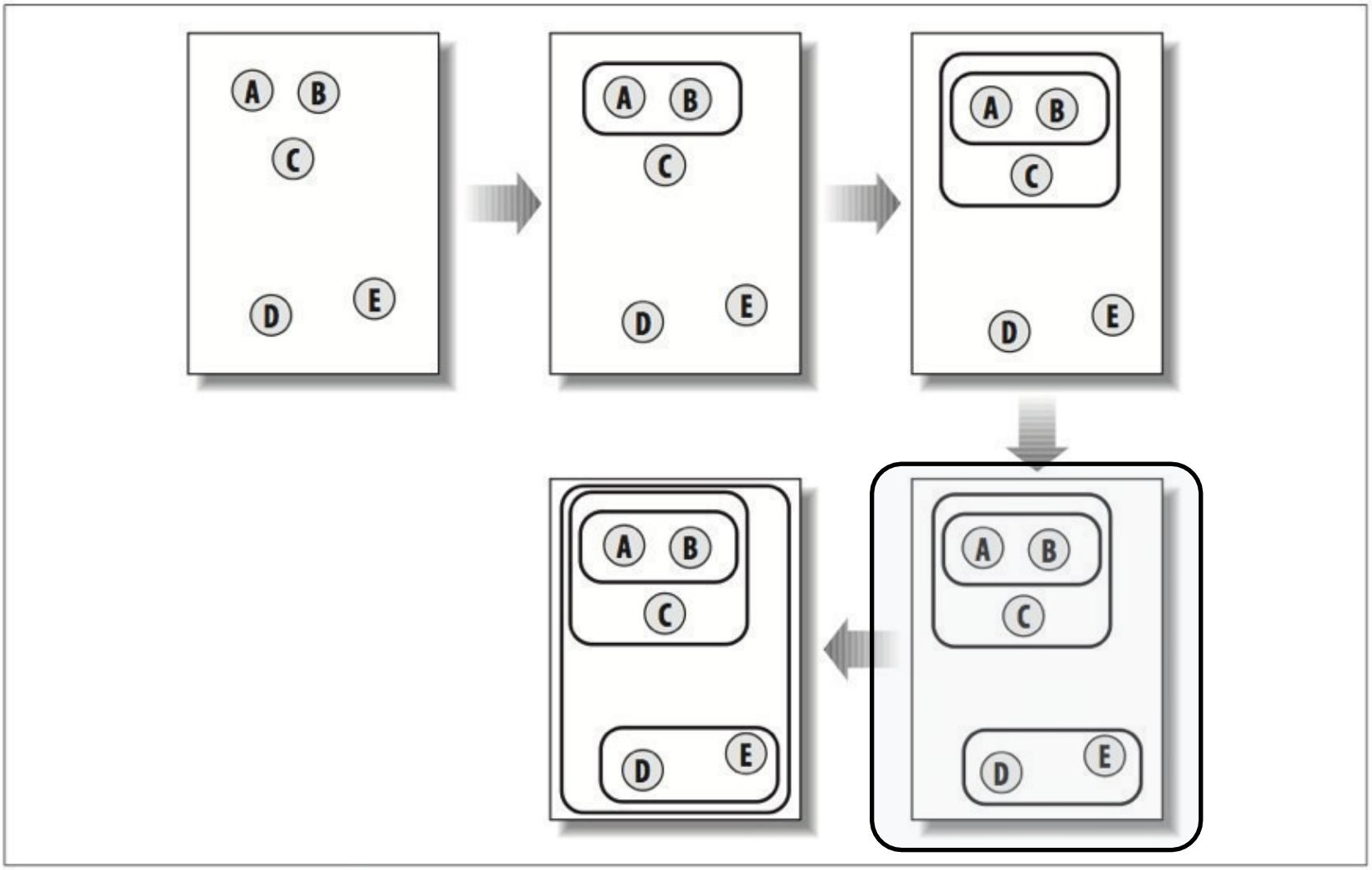
Hierarchical Clustering

İSTANBUL ŞEHİR UNIVERSITY



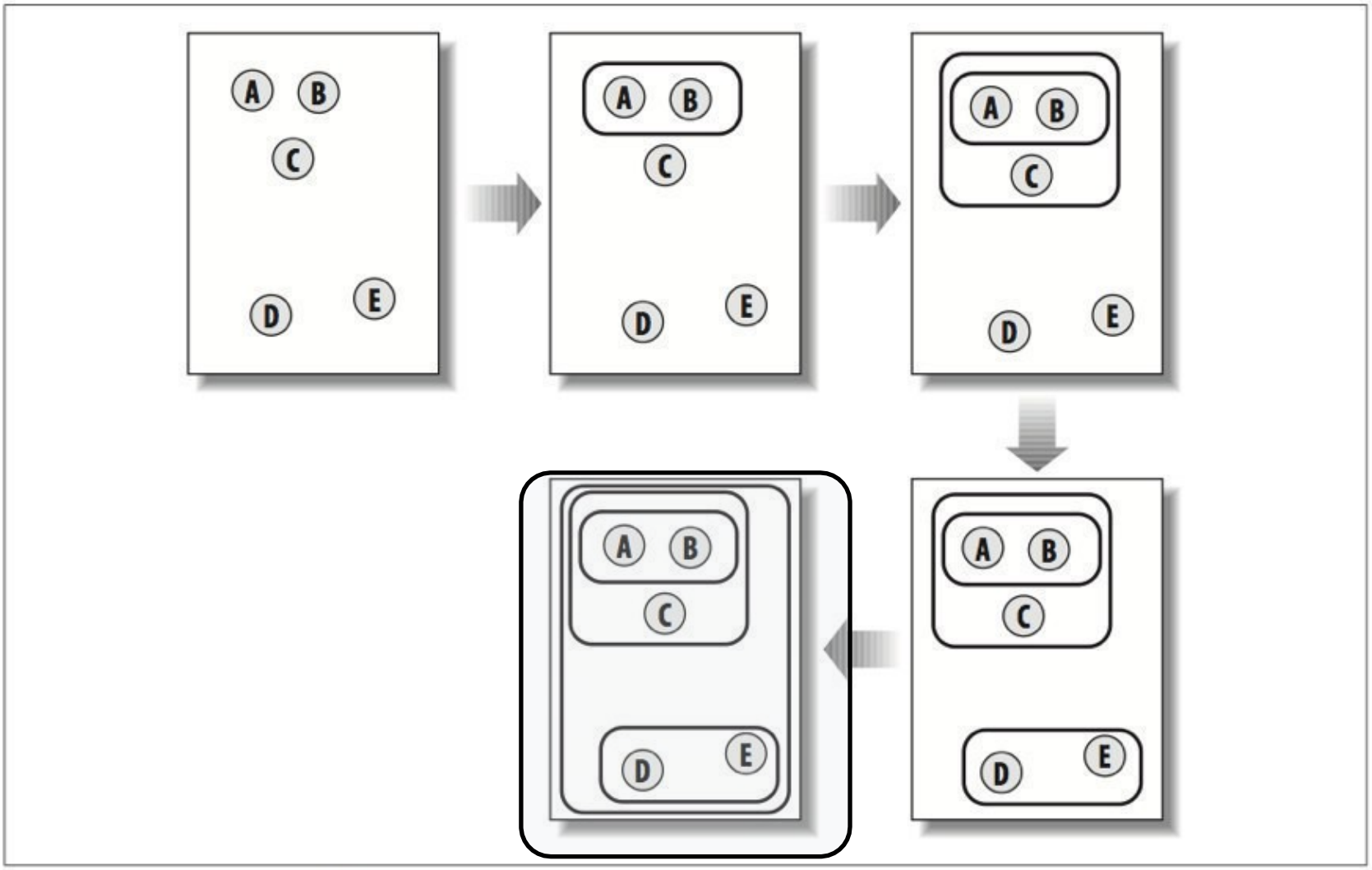
Hierarchical Clustering

İSTANBUL ŞEHİR UNIVERSITY



Hierarchical Clustering

İSTANBUL ŞEHİR UNIVERSITY



Similarity (closeness)

İSTANBUL ŞEHİR UNIVERSITY

- Pearson correlation coefficient.
- Others can be used as well.
  - e.g., Tanimoto (same as Jaccard)

Play with

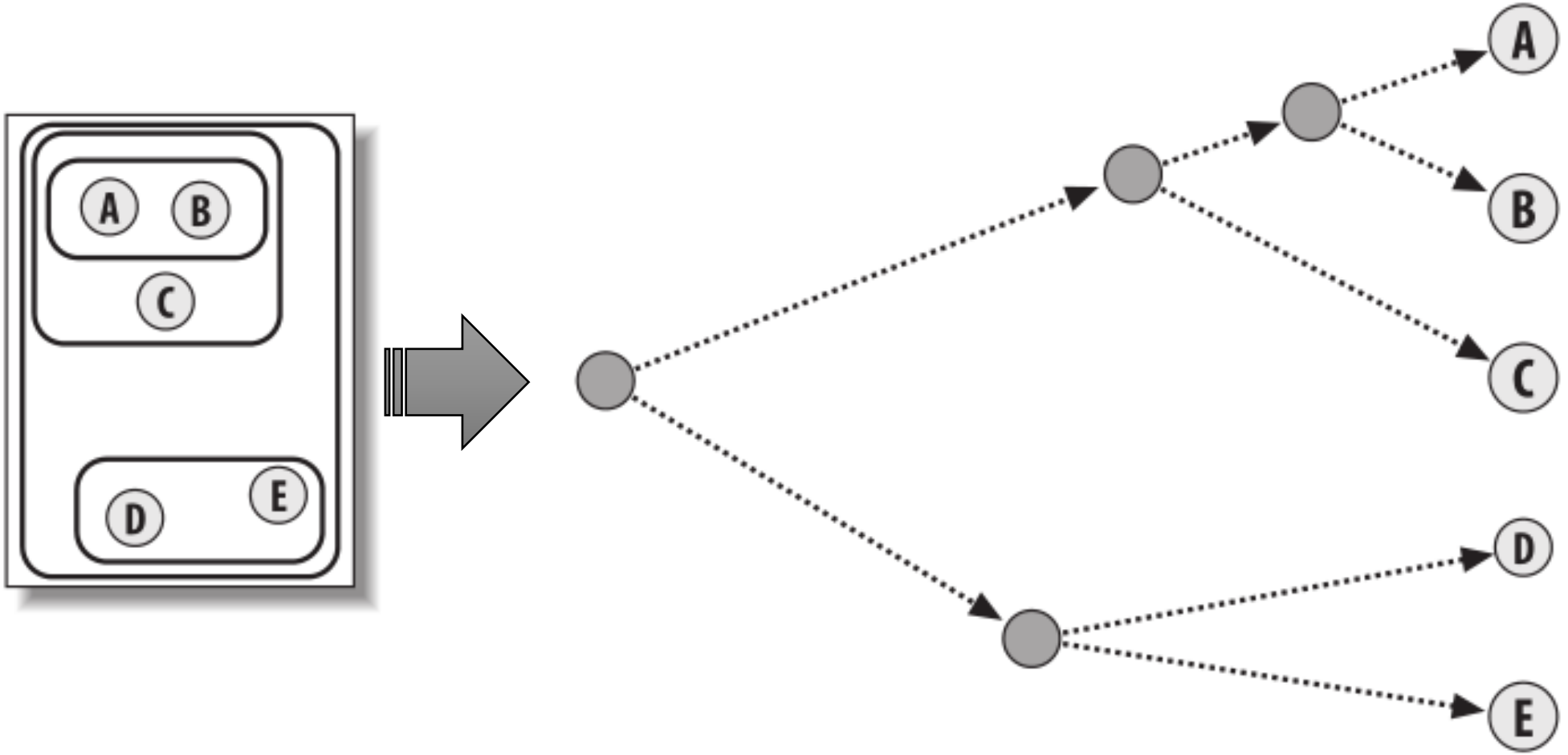
İSTANBUL ŞEHİR UNIVERSITY

- /Week 07/Code folder on LMS contains all the code. Dow
  them into a directory.

```
>>> import clusters
>>> blognames, words, data = clusters.readfile('blogdata.txt')
>>> clust=clusters.hcluster(data)
>>> clusters.printclust(clust, labels = blognames)
```

Visualizing Clusters - Dendograms

İSTANBUL ŞEHİR UNIVERSITY



Viewing Clusters - printclust

İSTANBUL ŞEHİR UNIVERSITY

```
John Battelle's Searchblog
-
  Search Engine Watch Blog
  -
    Read/WriteWeb
    -
      Official Google Blog
      -
        Search Engine Roundtable
        -
          Google Operating System
          Google Blogoscoped
```

Visualizing Clusters – Drawing Dendograms

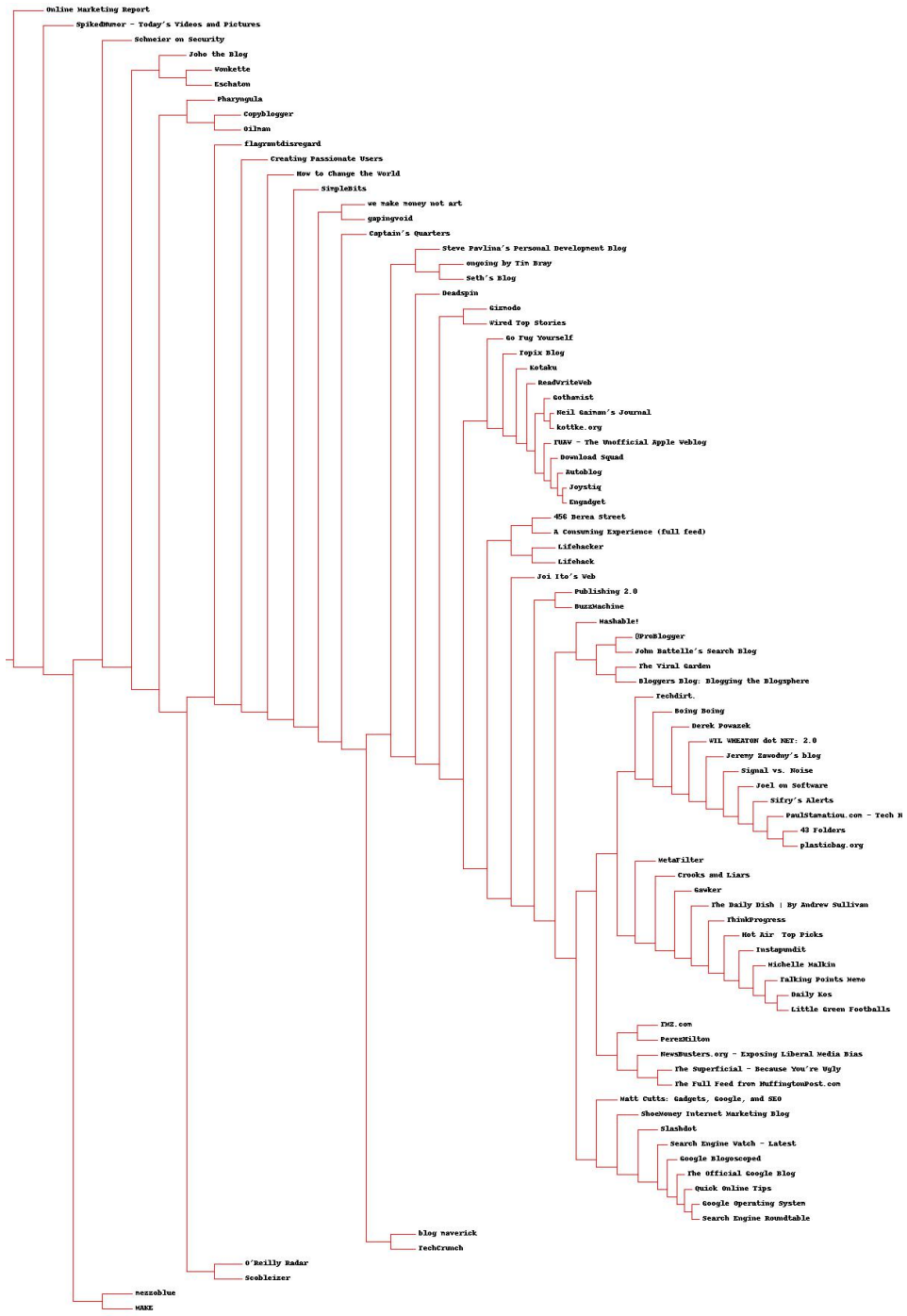
İSTANBUL ŞEHİR UNIVERSITY

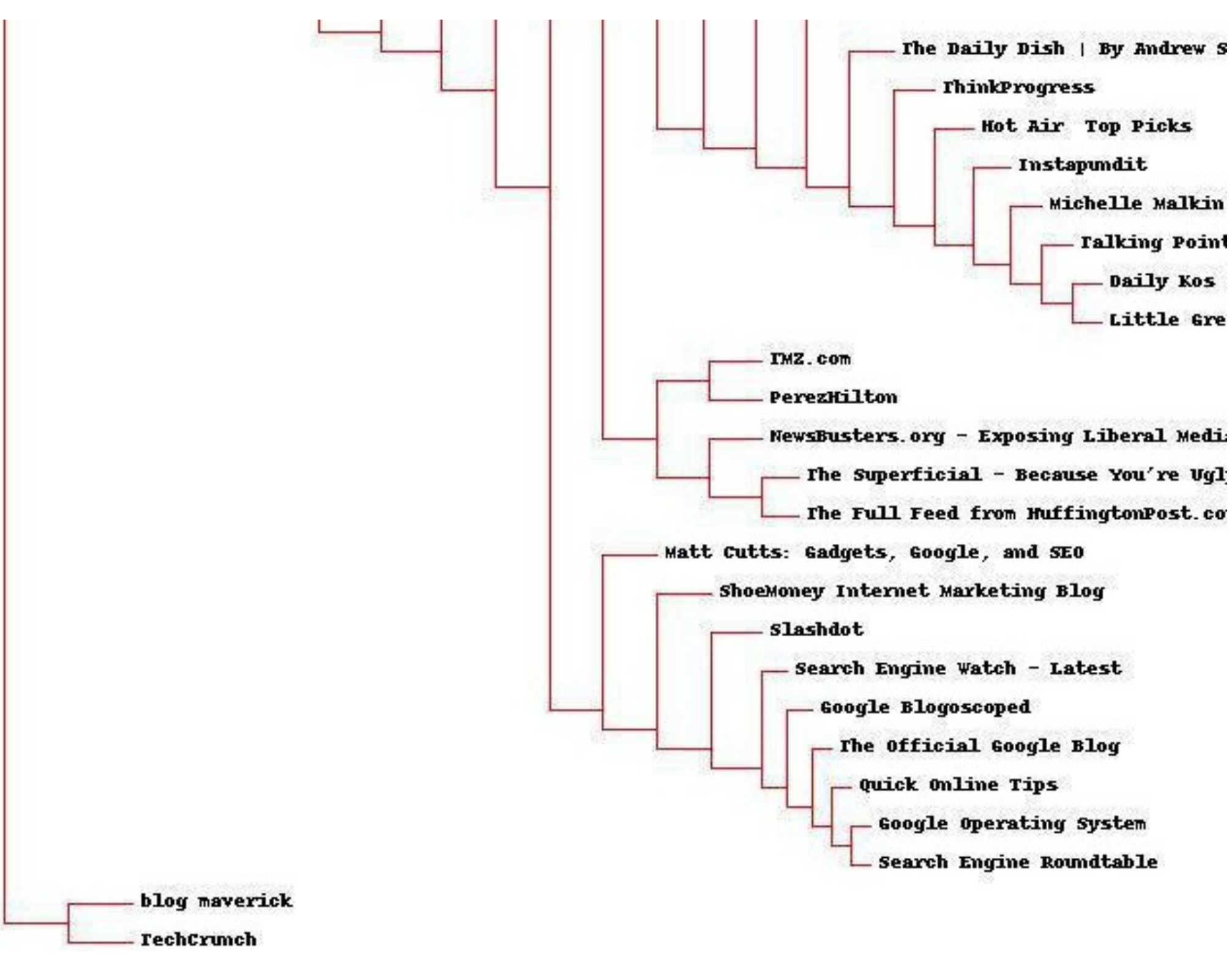
- Install PIL module
  - Follow the instructions posted on LMS
    - /Lectures/Week 07/PIL.Installation.txt

Play with

- /Week 07/Code folder on LMS contains all the code.
- Download them into a directory.

```
>>> import clusters
>>> blognames, words, data = clusters.readfile('blogdata.txt')
>>> clust=clusters.hcluster(data)
>>> clusters.drawdendrogram(clust, blognames, jpeg = ‘cl.jpg')
```


Online Marketing Report
SpikedHumor - Today's Videos and Pictures
Schneier on Security
Joho the Blog
Wonkette
Eschaton
Pharyngula
Copyblogger
Oilman
flagrantdisregard
Creating Passionate Users
How to Change the World
SimpleBits
we make money not art
gapingvoid
Captain's Quarters
Steve Pavlina's Personal Development Blog
ongoing by Tim Bray
Seth's Blog
Deadspin
Gizmodo
Wired Top Stories
Go Fug Yourself
Topix Blog
Kotaku
ReadWriteWeb
Gothamist
Neil Gaiman's Journal
Kottke.org
TUAW - The Unofficial Apple Weblog
Download Squad
Autoblog
Joystiq
Engadget
456 Berea Street
A Consuming Experience (full feed)
Lifehacker
Lifehack
Joi Ito's Web
Publishing 2.0
BuzzMachine
Mashable!
ProBlogger
John Battelle's Search Blog
The Viral Garden
Bloggers Blog: Blogging the Blogsphere
Techdirt.
Boing Boing
Derek Powazek
WIL WHEATON dot NET: 2.0
Jeremy Zawodny's blog
Signal vs. Noise
Joel on Software
Sifry's Alerts
Paulstamatiou.com - Tech N
43 Folders
plasticbag.org
MetaFilter
Crooks and Liars
Gawker
The Daily Dish | By Andrew Sullivan
ThinkProgress
Hot Air Top Picks
Instapundit
Michelle Malkin
Talking Points Memo
Daily Kos
Little Green Footballs
TMZ.com
PerezHilton
NewsBusters.org - Exposing Liberal Media Bias
The Superficial - Because You're Ugly
The Full Feed from HuffingtonPost.com
Matt Cutts: Gadgets, Google, and SEO
ShoeMoney Internet Marketing Blog
Slashdot
Search Engine Watch - Latest
Google Blogoscoped
The Official Google Blog
Quick Online Tips
Google Operating Systen
Search Engine Roundtable
blog maverick
TechCrunch
O'Reilly Radar
Scobleizer
mezzoblue
MAKE


The Daily Dish | By Andrew S
ThinkProgress
Hot Air Top Picks
Instapundit
Michelle Malkin
Talking Point
Daily Kos
Little Gre
TMZ.com
PerezHilton
NewsBusters.org - Exposing Liberal Media
The Superficial - Because You're Ugly
The Full Feed from HuffingtonPost.com
Matt Cutts: Gadgets, Google, and SEO
ShoeMoney Internet Marketing Blog
Slashdot
Search Engine Watch - Latest
Google Blogoscoped
The Official Google Blog
Quick Online Tips
Google Operating System
Search Engine Roundtable
blog maverick
TechCrunch

Clusters out of hierarchical clustering

İSTANBUL ŞEHİR ÜNİVERSİTY

- Each cluster in a hierarchical clustering algorithm is either a point in the tree with two branches, or an endpoint associated with an actual row from the dataset.
- Each cluster also contains data about its location, which is the raw data for the endpoints and the merged data for points.

Implementation details


```
class bicluster:
  def __init__(self,vec,left=None,right=None,distance=0.0,id=None):
    self.left=left
    self.right=right
    self.vec=vec
    self.id=id
    self.distance=distance
```

Clustering - I

İSTANBUL ŞEHİR UNIVERSITY

```
def readfile(filename):
  lines=[line for line in file(filename)]

  # First line is the column titles
  colnames=lines[0].strip( ).split('\t')[1:]
  rownames=[]
  data=[]
  for line in lines[1:]:
    p=line.strip( ).split('\t')
    # First column in each row is the rowname
    rownames.append(p[0])
    # The data for this row is the remainder of the row
    data.append([float(x) for x in p[1:]])
  return rownames,colnames,data
```

Clustering - II

İSTANBUL ŞEHİR UNIVERSITY

```
def hcluster(rows,distance=pearson):
  distances={}
  currentclustid=-1

  # Clusters are initially just the rows
  clust=[bicluster(rows[i],id=i) for i in range(len(rows))]

  while len(clust)>1:
    lowestpair=(0,1)
    closest=distance(clust[0].vec,clust[1].vec)

    # loop through every pair looking for the smallest distance
    for i in range(len(clust)):
      for j in range(i+1,len(clust)):
        # distances is the cache of distance calculations
        if (clust[i].id,clust[j].id) not in distances:
          distances[(clust[i].id,clust[j].id)]=distance(clust[i].vec,clust[j].vec)

        d=distances[(clust[i].id,clust[j].id)]

        if d<closest:
          closest=d
          lowestpair=(i,j)
```

Clustering - III

İSTANBUL ŞEHİR UNIVERSITY

```
    # calculate the average of the two clusters
    mergevec=[
    (clust[lowestpair[0]].vec[i]+clust[lowestpair[1]].vec[i])/2.0
    for i in range(len(clust[0].vec))]

    # create the new cluster
    newcluster=bicluster(mergevec,left=clust[lowestpair[0]],
                         right=clust[lowestpair[1]],
                         distance=closest,id=currentclustid)

    # cluster ids that weren't in the original set are negative
    currentclustid-=1
    del clust[lowestpair[1]]
    del clust[lowestpair[0]]
    clust.append(newcluster)

  return clust[0]
```