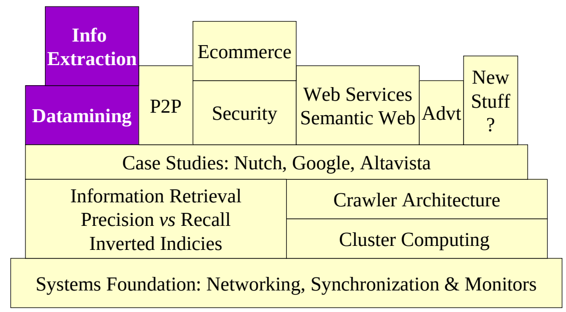
Course Overview



Tentative Schedule

11/1 Machine learning & datamining
11/3 Text categorization & evaluation methods
11/8 Information extraction
11/10 KnowItAll
11/15 ... continued
11/17 Clustering & Focused crawling
11/22 AJAX – Denise Draper
11/24 ---
11/29 Outbreak
12/1 Cryptography / Security
12/6 P2P & Advertising
12/8 Semantic Web

Today's Outline

- Overfitting
- Ensembles
  Learners: The more the merrier
- Co-Training
  Supervised learning with few labeled training ex
- Clustering
  No training examples

Bias

- The nice word for prejudice is "bias".
- What kind of hypotheses will you consider?
  What is allowable ***range*** of functions you use when approximating?
- What kind of hypotheses do you prefer?

Learning = Function Approximation

- E.g., Checkers
  V: boards -> evaluation
- E.g., Handwriting recognition
  V: image -> word
- E.g., Mushrooms
  V: mushroom-attributes -> {E, P}
- OPINE ?

Supervised Learning

- **Inductive learning** or "**Prediction**":
  Given examples of a function *(X, F(X))*
  Predict function *F(X)* for new examples *X*

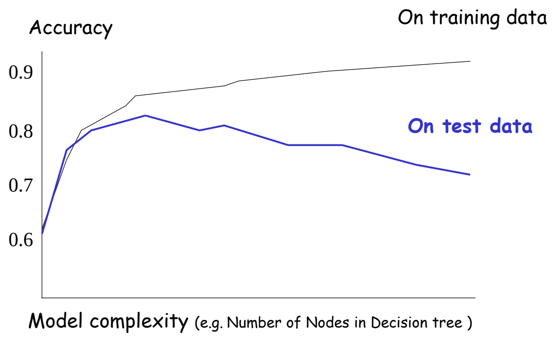
- Classification
  F(X) = Discrete
- Regression
  F(X) = Continuous
- Probability estimation
  F(X) = Probability*(X):*

Task
Performance Measure
Experience

(Some) Datamining Issues

- What feedback (experience) is available?
- How to represent this experience?
- How avoid overfitting?

Overfitting



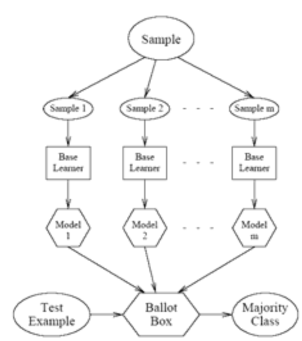
Overfitting…

- Hypothesis H is *overfit* when ∃ H' and
  H has ***smaller*** error on training examples, but
  H has ***bigger*** error on test examples
- Causes of overfitting
  Noisy data, or
  Training set is too small
- Huge problem in practice
  Take class in ML or datamining…

Ensembles of Classifiers

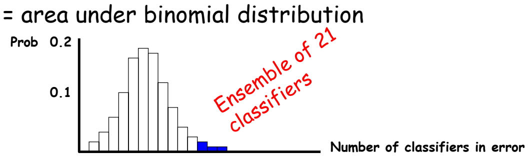
Bagging
Cross-validated committees
Boosting
Stacking

Voting



Ensembles of Classifiers

- Assume
  - Errors are independent (suppose 30% error)
  - Majority vote
- Probability that majority is wrong…
  - = area under binomial distribution

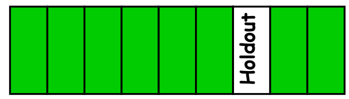


- If individual area is 0.3
- **Area under curve for ≥11 wrong is 0.026**
- Order of magnitude improvement!

Constructing Ensembles

Cross-validated committees

- Partition examples into *k* disjoint equiv classes
- Now create *k* training sets
  - Each set is union of all equiv classes ***except one***
  - So each set has (k-1)/k of the original training data
- Now train a classifier on each set



Ensemble Construction II

Bagging

- *Generate k sets of training examples*
- For each set
  - Draw m examples randomly (with replacement)
  - From the original set of m examples
- Each training set corresponds to
  - 63.2% of original
  - (+ duplicates)
- Now train classifier on each set

Ensemble Creation III

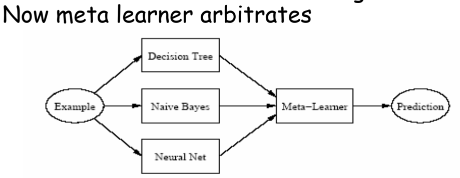
Boosting

- Maintain prob distribution over set of training ex
- Create k sets of training data iteratively:
- On iteration *i*
  - Draw m examples randomly (like bagging)
  - But use probability distribution to bias selection
  - Train classifier number *i* on this training set
  - Test partial ensemble (of *i* classifiers) on all training exs
  - Modify distribution: increase P of each error ex
- Create harder and harder learning problems...
- "Bagging with ***optimized*** choice of examples"

Ensemble Creation IV

Stacking

- Train several base learners
- Next train meta-learner
  - Learns when base learners are right / wrong
  - Now meta learner arbitrates



Train using cross validated committees
- Meta-L inputs = base learner predictions
- Training examples = 'test set' from cross validation

Co-Training Motivation

- Learning methods need labeled data
  - Lots of <x, f(x)> pairs
  - Hard to get… (who wants to label data?)
- But unlabeled data is usually plentiful…
  - Could we use this instead??????

Co-training

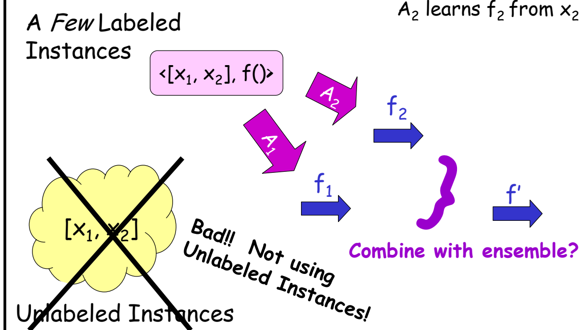
Suppose

- Have *little* labeled data + *lots* of unlabeled
- Each instance has two parts:
  x = [x1, x2]
  x1, x2 conditionally independent given f(x)
- Each half can be used to classify instance
  ∃f1, f2 such that f1(x1) ~ f2(x2) ~ f(x)
- Both f1, f2 are learnable
  f1 ∈ H1, f2 ∈ H2, ∃ learning algorithms A1, A2

Without Co-training

$f_1(x_1) \sim f_2(x_2) \sim f(x)$

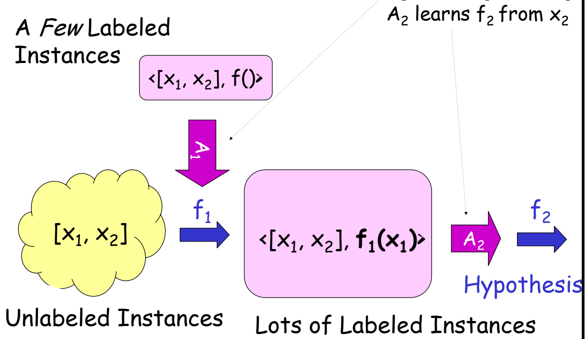
A_1 learns f_1 from x_1
A_2 learns f_2 from x_2



Co-training

$f_1(x_1) \sim f_2(x_2) \sim f(x)$

A_1 learns f_1 from x_1
A_2 learns f_2 from x_2



Observations

- Can apply A_1 to generate as much training data as one wants
  If x_1 is conditionally independent of x_2 / f(x),
  then the error in the labels produced by A_1
  will look like random noise to A_2 !!!
- Thus ***no limit*** to quality of the hypothesis A_2 can make

It really works!

- Learning to classify web pages as course pages
  x1 = bag of words on a page
  x2 = bag of words from all anchors pointing to a page
- Naïve Bayes classifiers
  12 labeled pages
  1039 unlabeled

	Page-based classifier	Hyperlink-based classifier	Combined classifier
Supervised training	12.9	12.4	11.1
Co-training	6.2	11.6	5.0

Table 2: Error rate in percent for classifying web pages as course home pages. The top row shows errors when training on only the labeled examples. Bottom row shows errors when co-training, using both labeled and unlabeled examples.

Choosing the Training Experience

- Credit assignment problem:
  Direct training examples: ← Expensive!
  - E.g. individual checker boards + correct move for each
  - Supervised learning

  Indirect training examples :
  - E.g. complete sequence of moves and final result
  - Reinforcement learning

  Unlabeled training examples
  - Clustering
- Which examples:
  Random, teacher chooses, learner chooses

Clustering Outline

- Motivation
- Document Clustering
- Offline evaluation
- Grouper I
- Grouper II
- Evaluation of deployed systems

Low Quality of Web Searches

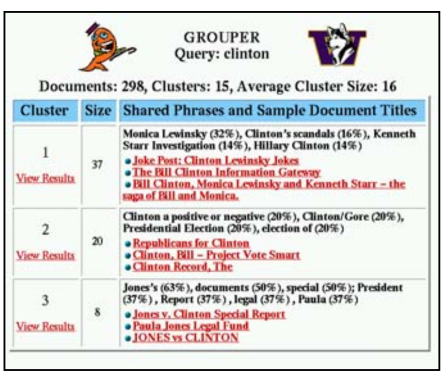
- System perspective:
  - small coverage of Web (<16%)
  - dead links and out of date pages
  - limited resources
- IR perspective
  (relevancy of doc ~ similarity to query):
  - very short queries
  - huge database
  - novice users

Document Clustering

- User receives many (200 - 5000) documents from Web search engine
- Group documents in clusters
  - by topic
- Present clusters as interface

Grouper

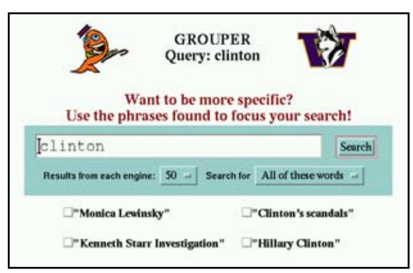
www.cs.washington.edu/research/clustering

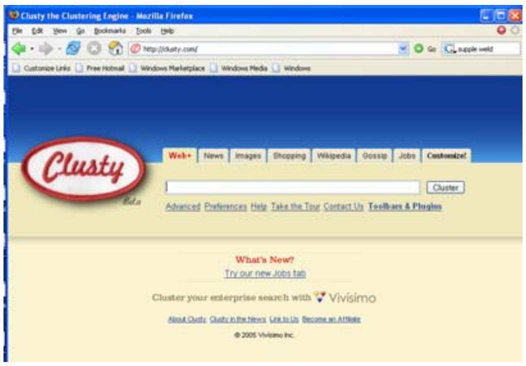


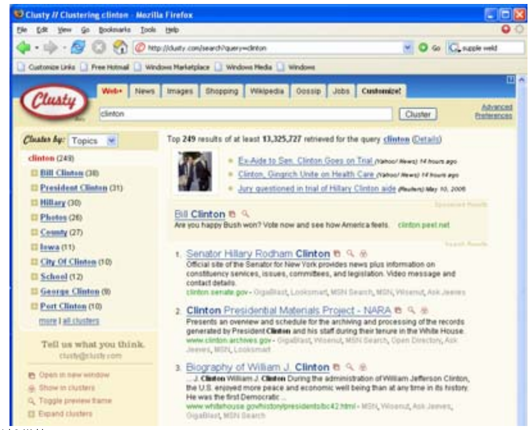
GROUPER
Query: clinton

Documents: 298, Clusters: 15, Average Cluster Size: 16

Cluster	Size	Shared Phrases and Sample Document Titles
1 View Results	37	Monica Lewinsky (32%), Clinton's scandals (16%), Kenneth Starr Investigation (14%), Hillary Clinton (14%) • Joke Post: Clinton Lewinsky Jokes • The Bill Clinton Information Gateway • Bill Clinton, Monica Lewinsky and Kenneth Starr – the saga of Bill and Monica.
2 View Results	20	Clinton a positive or negative (20%), Clinton/Gore (20%), Presidential Election (20%), election of (20%) • Republicans for Clinton • Clinton, Bill – Project Vote Smart • Clinton Record, The
3 View Results	8	Jones's (63%), documents (50%), special (50%); President (37%), Report (37%), legal (37%), Paula (37%) • Jones v. Clinton Special Report • Paula Jones Legal Fund • JONES vs CLINTON







Desiderata

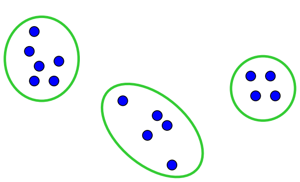
- Coherent cluster
- Speed
- Browsable clusters
  - Naming

Main Questions

- Is document clustering feasible for Web search engines?
- Will the use of phrases help in achieving high quality clusters?
- Can phrase-based clustering be done quickly?

1. Clustering

group together similar items
(words or documents)

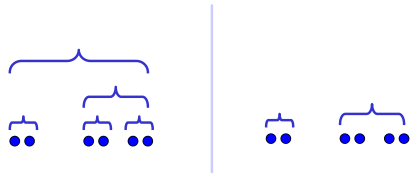


Clustering Algorithms

- Hierarchical Agglomerative Clustering
  - $O(n^2)$
- Linear-time algorithms
  - K-means (Rocchio, 66)
  - Single-Pass (Hill, 68)
  - Fractionation (Cutting et al, 92)
  - Buckshot (Cutting et al, 92)

Basic Concepts - 1

- Hierarchical vs. Flat



Basic Concepts - 2

- hard clustering:
  each item in only one cluster
- soft clustering:
  each item has a probability of membership in each cluster
- disjunctive / overlapping clustering:
  an item can be in more than one cluster

Basic Concepts - 3

distance / similarity function
(for documents)

dot product of vectors
number of common terms
co-citations
access statistics
share common phrases

Basic Concepts - 4

- What is "right" number of clusters?
  apriori knowledge
  default value: "5"
  clusters up to 20% of collection size
  choose best based on external criteria
  Minimum Description Length
  Global Quality Function
- no good answer

K-means

- Works when we know k, the number of clusters
- Idea:
  Randomly pick k points as the "centroids" of the k clusters
  Loop:
  - ∀ points, add to cluster w/ nearest centroid
  - Recompute the cluster centroids
  - Repeat loop (until no change)

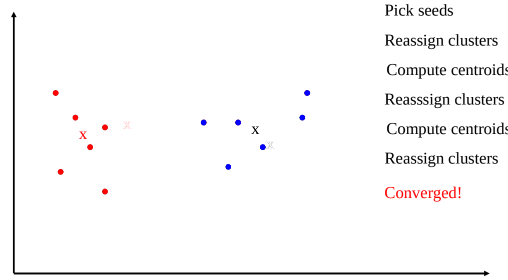
Iterative improvement of the objective function:
Sum of the squared distance from each point to the centroid of its cluster

K-means Example

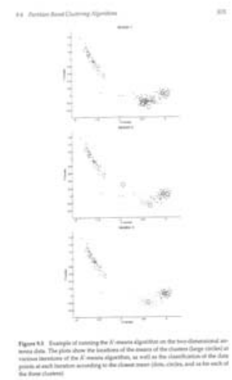
- For simplicity, 1-dimension objects and k=2.
  Numerical difference is used as the distance
- Objects: 1, 2, 5, 6,7
- K-means:
  Randomly select 5 and 6 as centroids;
  => Two clusters {1,2,5} and {6,7}; meanC1=8/3, meanC2=6.5
  => {1,2}, {5,6,7}; meanC1=1.5, meanC2=6
  => no change.
  Aggregate dissimilarity
  - (sum of squares of distanceeach point of each cluster from *its* cluster center--(intra-cluster distance)
    $= 0.5^2 + 0.5^2 + 1^2 + 0^2 + 1^2 = 2.5$

$|1-1.5|^2$

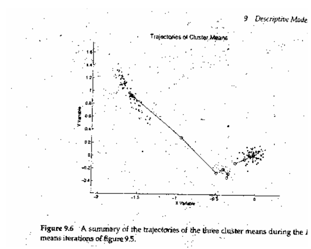
K Means Example
(K=2)



Pick seeds
Reassign clusters
Compute centroids
Reasssign clusters
Compute centroids
Reassign clusters
Converged!



Example of K-means in operation



Time Complexity

- Assume computing distance between two instances is $O(m)$ where m is the dimensionality of the vectors.
- Reassigning clusters: $O(kn)$ distance computations, or $O(knm)$.
- Computing centroids: Each instance vector gets added once to some centroid: $O(nm)$.
- Assume these two steps are each done once for I iterations: $O(Iknm)$.
- Linear in all relevant factors, assuming a fixed number of iterations,
  - more efficient than $O(n^2)$ HAC (to come next)

Vector Quantization: K-means as Compression

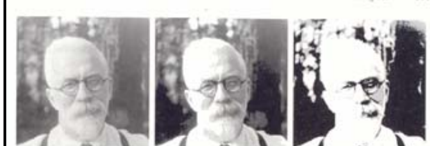


FIGURE 14.9. Sir Ronald A. Fisher (1890-1962) was one of the founders of modern day statistics, to whom we owe maximum-likelihood, sufficiency, and many other fundamental concepts. The image on the left is a 1024×1024 grayscale image at 8 bits per pixel. The center image is the result of 2×2 block VQ, using 200 code vectors, with a compression rate of 1.9 bits/pixel. The right image uses only four code vectors, with a compression rate of 0.50 bits/pixel

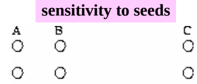
---

Problems with K-means

Why not the minimum value?

- Need to know k in advance
  - Could try out several k?
    - Cluster tightness increases with increasing K.
      - Look for a kink in the tightness vs. K curve
- Tends to go to local minima that are sensitive to the starting centroids
  - Try out multiple starting points
- Disjoint and exhaustive
  - Doesn't have a notion of "outliers"
    - Outlier problem can be handled by K-medoid or neighborhood-based algorithms
- Assumes clusters are spherical in vector space
  - Sensitive to coordinate changes, weighting etc.

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}

Hierarchical Clustering

- Agglomerative
  - bottom-up

Initialize: - *each item a cluster*
Iterate: - *select two most similar clusters*
- *merge them*
Halt: *when have required # of clusters*

Hierarchical Clustering

- Divisive
  top-bottom

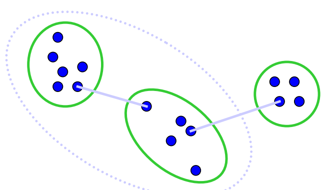
Initialize: *-all items one cluster*
Iterate: *- select a cluster (least coherent)*
- divide it into two clusters
Halt: *when have required # of clusters*

HAC Similarity Measures

- Single link
- Complete link
- Group average
- Ward's method

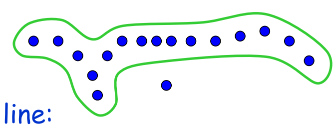
Single Link

- cluster similarity = similarity of two most similar members



Single Link

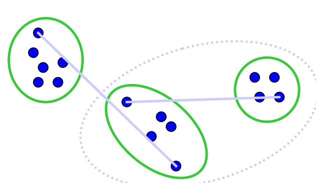
- $O(n^2)$
- chaining:



- bottom line:
  simple, fast
  often low quality

Complete Link

- cluster similarity = similarity of two least similar members

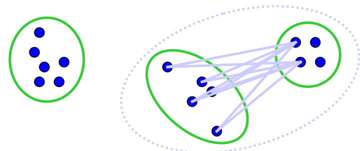


Complete Link

- worst case $O(n^3)$
- fast algo requires $O(n^2)$ space
- no chaining
- bottom line:
  typically much faster than $O(n^3)$,
  often good quality

Group Average

- cluster similarity
  = average similarity of all pairs

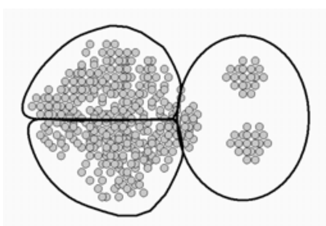


HAC Often Poor Results - Why?

- Often produces single large cluster
- Work best for:
  spherical clusters; equal size; few outliers
- Text documents:
  no model
  not spherical; not equal size; overlap
- Web:
  many outliers; lots of noise

Example: Clusters of Varied Sizes

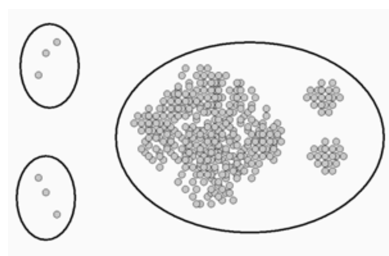
k-means; complete-link; group-average:



single-link: chaining,
but succeeds on this example

Example - Outliers

HAC:



Suffix Tree Clustering

(KDD'97; SIGIR'98)

- Most clustering algorithms aren't ***specialized*** for text:
  Model document as **set** of words
- STC:
  document = **sequence** of words

STC Characteristics

- Coherent
  phrase-based
  overlapping clusters
- Speed and Scalability
  linear time; incremental
- Browsable clusters
  phrase-based
  simple cluster definition

STC - Central Idea

- Identify ***base clusters***
  - a group of documents that share a phrase
  - use a ***suffix tree***
- Merge base clusters as needed

STC - Outline

Three logical steps:

1. "Clean" documents
2. Use a ***suffix tree*** to identify ***base clusters*** - a group of documents that share a phrase
3. Merge base clusters to form clusters

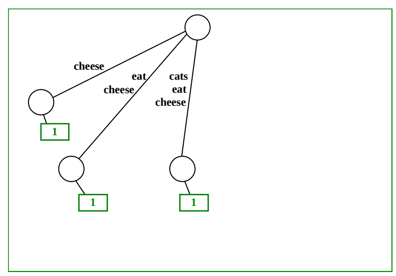
Step 1 - Document "Cleaning"

- Identify sentence boundaries
- Remove
  - HTML tags,
  - JavaScript,
  - Numbers,
  - Punctuation

Suffix Tree

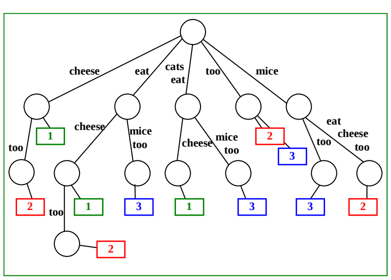
(Weiner, 73; Ukkonen, 95; Gusfield, 97)

Example - suffix tree of the string: (1) "**cats eat cheese**"



Example - suffix tree of the strings:

(1) "**cats eat cheese**",
(2) "**mice eat cheese too**" and
(3) "**cats eat mice too**"



Step 2 - Identify Base Clusters via Suffix Tree

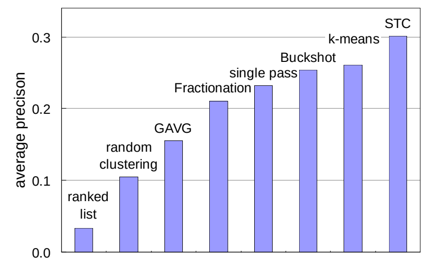
- Build one suffix tree from all sentences of all documents
- Suffix tree node = base cluster
- Score all nodes
- Traverse tree and collect top k (500) base clusters

Step 3 - Merging Base Clusters

- Motivation: similar documents share multiple phrases
- Merge base clusters based on the overlap of their document sets
- Example (query: "salsa")

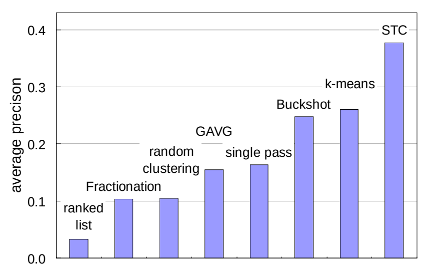
"tabasco sauce"	docs: **3,4,5,6**
"hot pepper"	docs: **1,3,5,6**
"dance"	docs: **1,2,7**
"latin music"	docs: **1,7,8**

Average Precision - WSR-SNIP



16% increase over k-means (not stat. sig.)

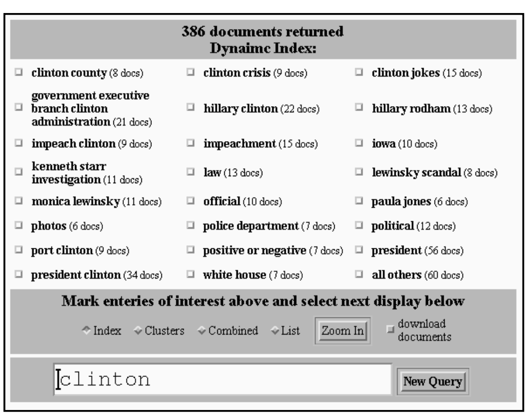
Average Precision - WSR-DOCS



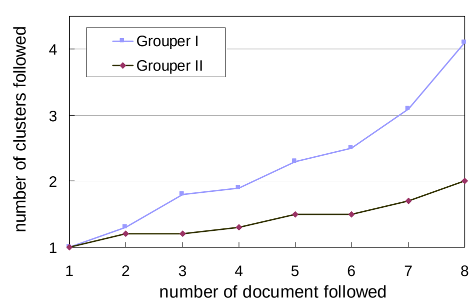
45% increase over k-means (stat. sig.)

Grouper II

- Dynamic Index:
  Non-merged based clusters
- Multiple interfaces:
  List, Clusters + Dynamic Index (key phrases)
- Hierarchical:
  Interactive "Zoom In" feature
  (similar to Scatter/Gather)

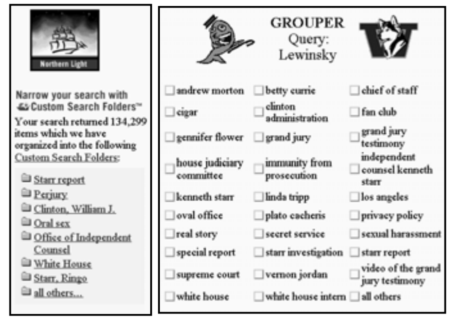


Evaluation - Log Analysis



Northern Light

- "Custom Folders"
- 20000 predefined topics in a manually developed hierarchy
- Classify document into topics
- Display "dominant" topics in search results



Summary

- Post-retrieval clustering
  to address low precision of Web searches
- STC
  phrase-based; overlapping clusters; fast
- Offline evaluation
  Quality of STC,
  advantages of using phrases vs. n-grams, FS
- Deployed two systems on the Web
  Log analysis: Promising initial results

www.cs.washington.edu/research/clustering