Topics in Data Analytics

Collaborative Filtering, Finding Similar Items in Very High Dimension, Sentiment Analysis

Debapriyo Majumdar
Indian Statistical Institute Kolkata
debapriyo@isical.ac.in

Introduction

- We all know about big data
- We all know about online marketing / stores
- We all know about social media
- In this talk, we will see some techniques relevant in modern days data analytics / processing
  - Recommender systems
  - Finding similar items in very high dimensional data
  - Sentiment analysis: starting to make sense of what people write

Recommender systems

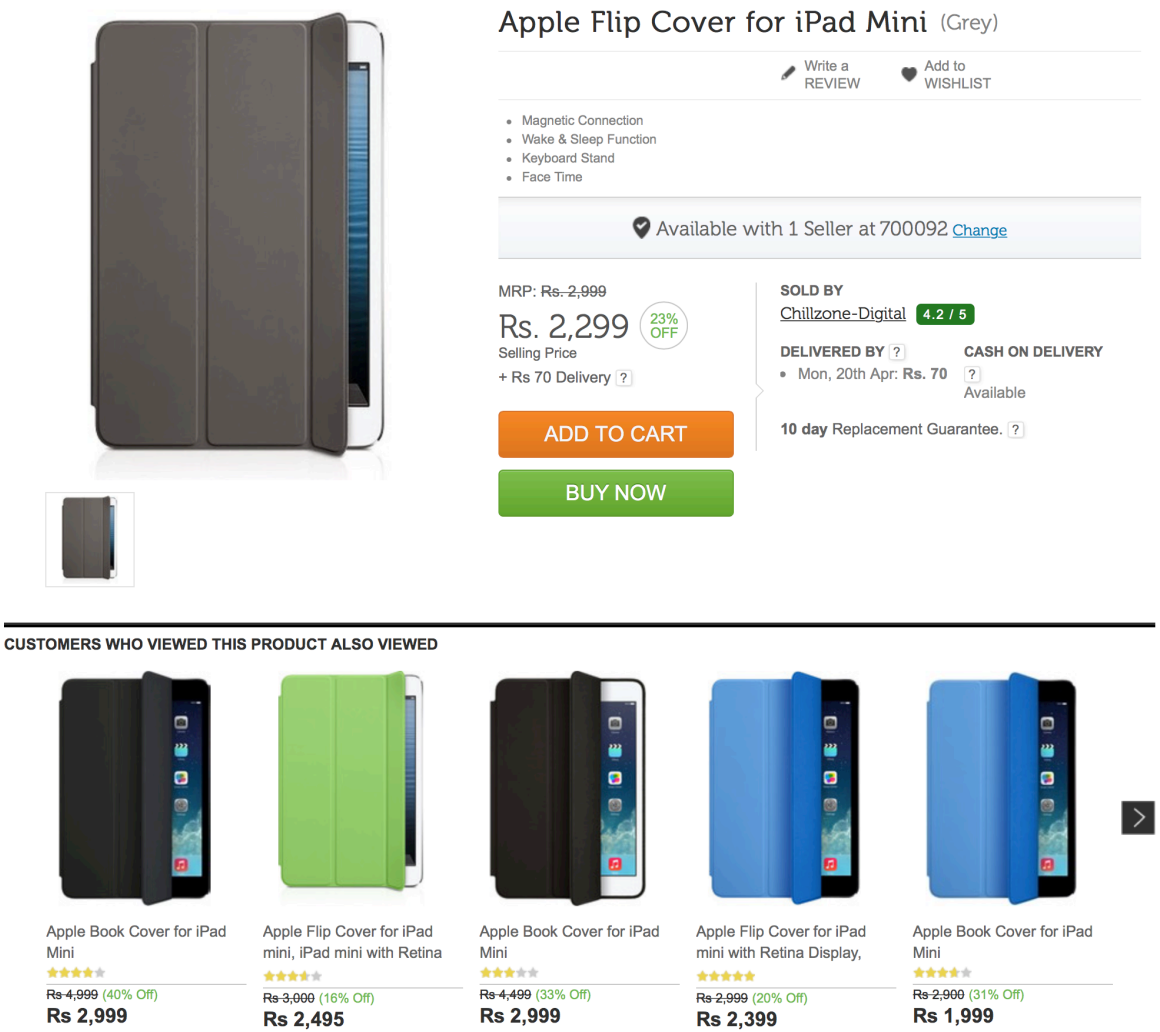
Business

- How to increase revenue?
- How to recommend items customers like?
- May be then they'll buy more

Customer

- Too many options
- How to choose the right one?

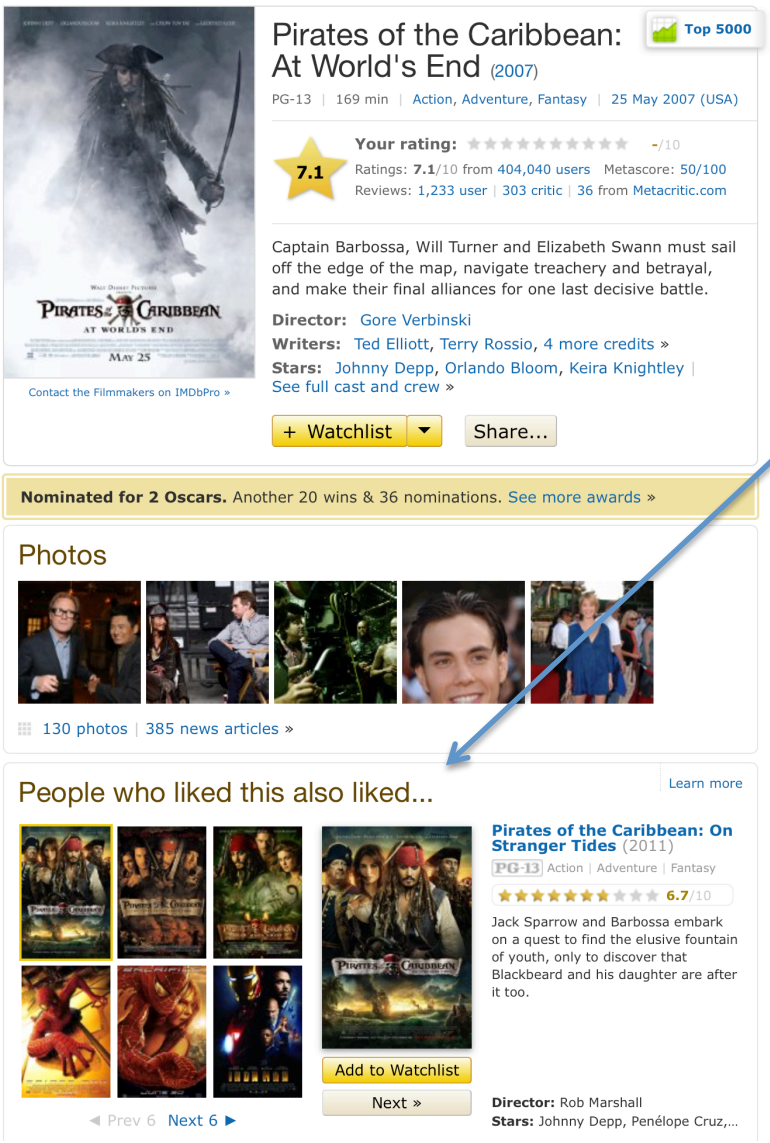
Recommender systems



Customers who viewed / bought this product also bought

Since you are looking at this, you may also look at …

Recommender systems



Viewers who liked this movie also liked the other movies

Since you are looking at this page, you may also like…

The Recommendation Problem

- We have a set of users U and a set of items S to be recommended to the users.
- Let p be an utility function that measures the usefulness of item s ($\in S$) to user u ($\in U$), i.e.,
  - $p : U \times S \rightarrow R$, where R is a totally ordered set (e.g., non-negative integers or real numbers in a range)
- Objective
  - Learn p based on the past data
  - Use p to predict the utility value of each item s ($\in S$) to each user u ($\in U$)

Two main formulations

- Rating prediction: predict the rating score that a user is likely to give to an item that (s)he has not seen or used before
  - Rating on an unseen movie
  - In this case, the utility of item s to user u is the rating given to s by u
- Item prediction: predict a ranked list of items that a user is likely to buy or use

Approaches

<u>Content-based recommendations:</u>

- The user will be recommended items similar to the ones the user preferred in the past

<u>Collaborative filtering (or collaborative recommendations):</u>

- The user will be recommended items that people with similar tastes and preferences liked in the past

<u>Hybrids:</u> Combine collaborative and content-based methods

Content based recommendation

- Will user *u* like item *s*?
- Look at items similar to *s*; does *u* like them?
  - Similarity based on content
  - Example: a movie represented based on features as specific actors, director, genre, subject matter, etc
- The user's interest or preference is also represented by the same set of features (the <u>user profile</u>)
- Candidate item *s* is compared with the user profile of *u* in the same feature space
- Determine if *u* would like *s*, or
- Top *k* similar items are recommended

Collaborative filtering

- Collaborative filtering (CF): more studied and widely used recommendation approach in practice
  - k-nearest neighbor
  - association rules based prediction
  - matrix factorization
- Key characteristic: predicts the utility of items for a user based on the items previously rated by other like-minded users (thus, *collaborative*)

k nearest neighbor approach

- No model building
- Utilizes the entire user-item database to generate predictions directly, i.e., there is no model building.
- This approach includes both
  - User-based methods
  - Item-based methods

User based kNN CF

- Let the record (or profile) of the target user be **u** (represented as a vector), and the record of another user be **v** (**v** $\in T$).
- The similarity between the target user, **u**, and a neighbor, **v**, can be calculated using the **Pearson's correlation coefficient**:

$$sim(\mathbf{u},\mathbf{v}) = \frac{\sum_{i \in C}(r_{\mathbf{u},i} - \overline{r}_{\mathbf{u}})(r_{\mathbf{v},i} - \overline{r}_{\mathbf{v}})}{\sqrt{\sum_{i \in C}(r_{\mathbf{u},i} - \overline{r}_{\mathbf{u}})^2}\sqrt{\sum_{i \in C}(r_{\mathbf{v},i} - \overline{r}_{\mathbf{v}})^2}},$$

where V is the set of k similar users, $r_{\mathbf{v},i}$ is the rating of user **v** given to item i

- Compute the rating prediction of item i for target user **u**

$$p(\mathbf{u},i) = \overline{r}_{\mathbf{u}} + \frac{\sum_{\mathbf{v} \in V} sim(\mathbf{u},\mathbf{v}) \times (r_{\mathbf{v},i} - \overline{r}_{\mathbf{v}})}{\sum_{\mathbf{v} \in V} \left| sim(\mathbf{u},\mathbf{v}) \right|}$$

Problems with user based CF

- The problem with the user-based formulation of collaborative filtering is the lack of scalability:
  - it requires the real-time comparison of the target user to all user records in order to generate predictions
- A variation of this approach that remedies this problem is called item-based CF

Item-based CF

- The item-based approach works by comparing items based on their pattern of ratings across users. The similarity of items *i* and *j* is computed as follows:

$$sim(i,j) = \frac{\sum_{\mathbf{u}\in U}(r_{\mathbf{u},i} - \overline{r}_{\mathbf{u}})(r_{\mathbf{u},j} - \overline{r}_{\mathbf{u}})}{\sqrt{\sum_{\mathbf{u}\in U}(r_{\mathbf{u},i} - \overline{r}_{\mathbf{u}})^2}\sqrt{\sum_{\mathbf{u}\in U}(r_{\mathbf{u},j} - \overline{r}_{\mathbf{u}})^2}}$$

- After computing the similarity between items we select a set of *k* most similar items to the target item and generate a predicted value of user **u**' s rating

$$p(\mathbf{u}, i) = \frac{\sum_{j\in J} r_{\mathbf{u},j} \times sim(i,j)}{\sum_{j\in J} sim(i,j)}$$

where *J* is the set of *k* similar items

Association rule-based CF

- Transaction database: users, items
  - User → Item: viewed, bought, liked
- Find association rules such as
  - Bought *X,* bought *Y* → Bought *Z*
  - Confidence and support (how strong is this association)
- Rank items based on measures such as confidence, subject to some minimum support
- Further reading: association rule mining

Matrix factorization based CF

- Gained popularity for CF in recent years due to its superior performance both in terms of recommendation quality and scalability.
- Part of its success is due to the Netflix Prize contest for movie recommendation
- Popularized a Singular Value Decomposition (SVD) based matrix factorization algorithm
  - The prize winning method of the Netflix Prize Contest employed an adapted version of SVD

Linear algebra review

- Rank of a matrix: number of linearly independent columns (or rows)
- If A is an $m \times n$ matrix, $\text{rank}(A) \leq \min(m, n)$

Rank of

	m_1	m_2	m_3	m_4	m_5
sourav	1	2	0	0	1
debapriyo	1	2	0	0	0
ansuman	1	2	1	0.2	0
arijit	0	0	1	0.2	0.8

= ?

Linear algebra review

- A square matrix M is called orthogonal if its rows and columns are orthogonal unit vectors (orthonormal vectors)
  - Each column (row) has norm 1
  - Any two columns (rows) have dot product 0

- For a square matrix A, if there is a vector v such that

$$Av = \lambda v$$

for some scalar λ, then v is called an eigenvector of A

λ is the corresponding eigenvalue

Singular value decomposition

If A is an $m \times n$ matrix with rank r

Then there exists a factorization of A as:

$$\underbrace{A}_{m\times n} = \underbrace{U}_{m\times m} \underbrace{\Sigma}_{m\times n} \underbrace{V^T}_{n\times n}$$

where U $(m \times m)$ and V $(n \times n)$ are orthogonal, and

Σ $(m \times n)$ is a diagonal-like matrix

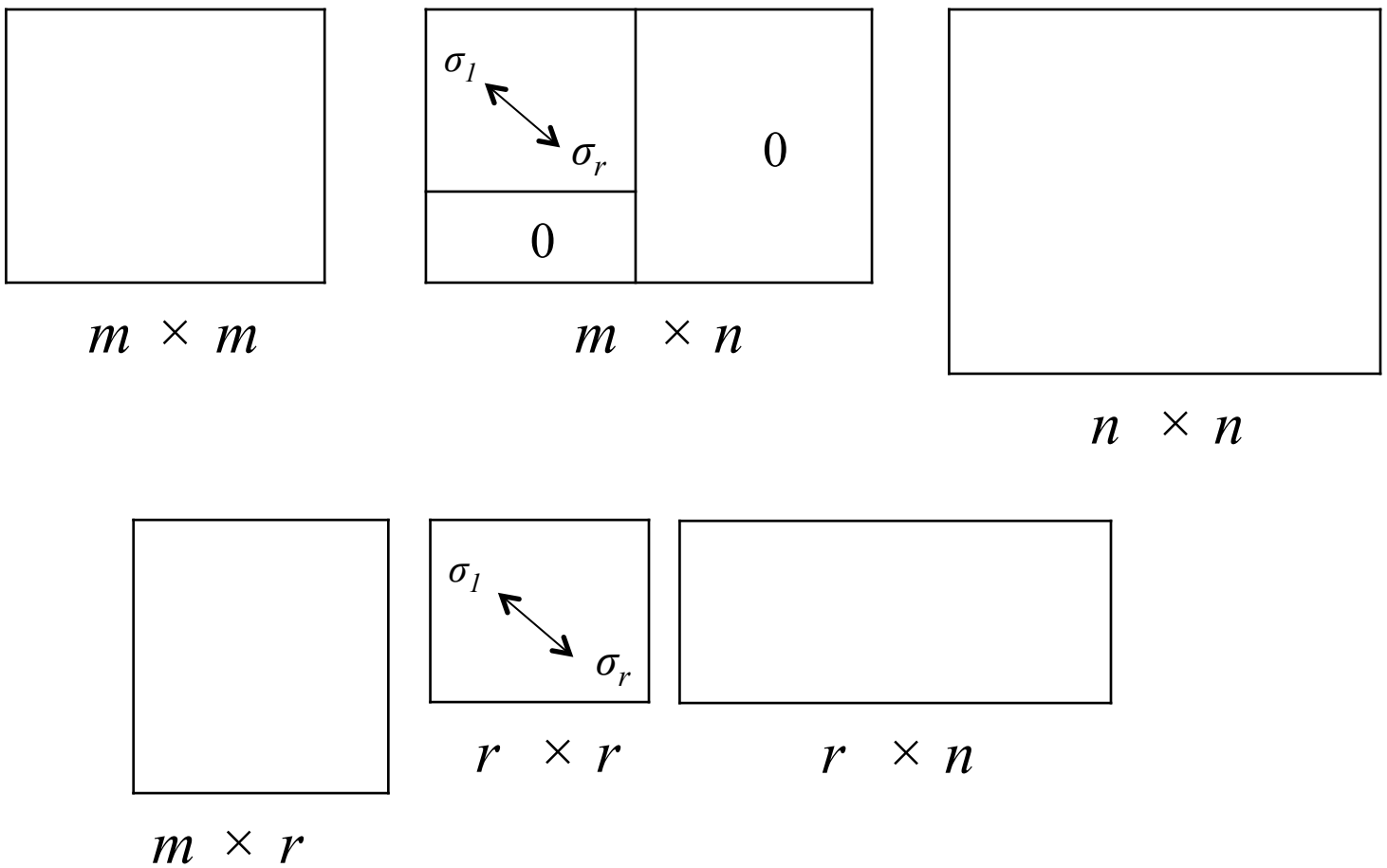
$\Sigma = (\sigma_{ij})$, where $\sigma_{ii} = \sigma_i$, for $i = 1, \ldots, r$ are the singular values of A, all non-diagonal entries of Σ are zero

$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq 0$

Columns of U are the left singular vectors of A

Singular value decomposition

$$\underbrace{A}_{m\times n} = \underbrace{U}_{m\times m} \underbrace{\Sigma}_{m\times n} \underbrace{V^T}_{n\times n}$$



Matrix digonalization for symmetric matrix

If A is an $m \times m$ matrix with rank r

Consider $C = AA^T$. Then:

$$\underbrace{C}_{m\times m} = \underbrace{A}_{m\times n}\underbrace{A^T}_{n\times m}$$

C has rank r

$$= \underbrace{U}_{m\times r}\underbrace{\Sigma}_{r\times r}\underbrace{V^T}_{r\times n}\left(\underbrace{U}_{m\times r}\underbrace{\Sigma}_{r\times r}\underbrace{V^T}_{r\times n}\right)^T$$

Σ^2 is a diagonal matrix with entries σ_i^2, for $i = 1, \ldots, r$

$$= \underbrace{U}_{m\times r}\underbrace{\Sigma}_{r\times r}\underbrace{V^T}_{r\times n}\underbrace{V}_{n\times r}\underbrace{\Sigma^T}_{r\times r}\underbrace{U^T}_{r\times m}$$

Columns of U are the eigenvectors of C

$$= \underbrace{U}_{m\times r}\underbrace{\Sigma}_{r\times r}\underbrace{\Sigma^T}_{r\times r}\underbrace{U^T}_{r\times m}$$

$$= \underbrace{U}_{m\times r}\underbrace{\Sigma^2}_{r\times r}\underbrace{U^T}_{r\times m}$$

σ_i^2 are the corresponding eigenvalues of C

SVD of term – document matrix

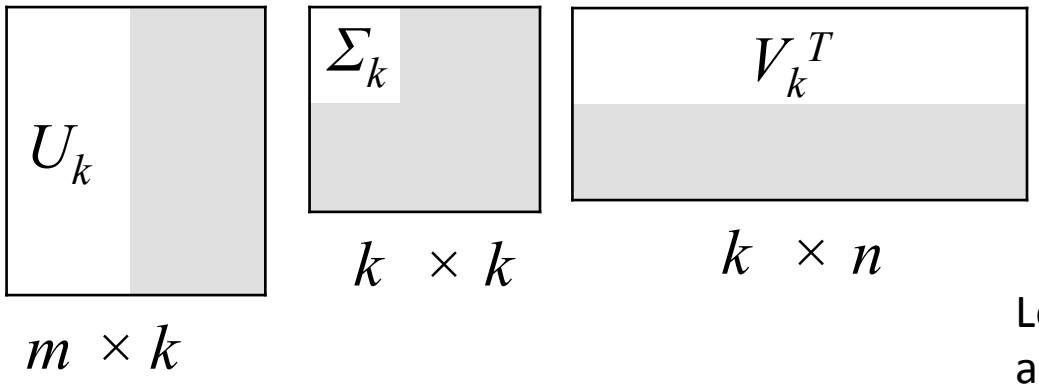
$$A = [d_1, \ldots, d_n]$$

Documents are vectors in the m dimensional term space

But we would think there are less number of concepts associated with the collection

m terms, k concepts. $k << m$

Ignore all but the first k singular values, singular vectors



Low rank approximation

Low-rank approxmation

$$\underbrace{A}_{m\times n} = \underbrace{U}_{m\times m}\ \underbrace{\Sigma}_{m\times n}\ \underbrace{V^T}_{n\times n}$$

$$\Downarrow$$

$$\underbrace{A_k}_{m\times n} = \underbrace{U_k}_{m\times k}\ \underbrace{\Sigma_k}_{k\times k}\ \underbrace{V_k^T}_{k\times n}$$

Rank k

Still m dimensional vectors

Back to the topic: How do we choose a movie?

- Possibly, we look at a few factors
  - Genre (Action, Thriller, Drama, Horror, ...)
  - Actor (Leo, Aamir Khan, Amitabh, …)
  - Director (Nolan, Spielberg, Mani Ratnam, ...)

- There are only a few factors that helps decide our choice (remember: content based)

- But say, we do not know (and we don't want to know) exactly which factors …

Latent Factor Model

- Assumes that the factors affecting the choices are hidden / latent.
- These factors need not be exactly known
  - The item-*j* is characterized by *k*-factors

$$v_j = [v_j^{(1)}, v_j^{(2)}, \ldots . v_j^{(k)}]^T$$

  - The user-*i* is characterized by his / her affinity towards these factors

$$u_i = [u_i^{(1)}, u_i^{(2)}, \ldots . u_i^{(k)}]^T$$

Mathematical Formalism

- Latent factor model assumes that the rating of a user on an item is just an inner-product of the users' and items' latent factors.

$$r_{i,j} = u_i^T v_j$$

- How do we use this model for prediction?

A holistic view

- The matrix of interactions M
- There are missing entries; they are not zeros!

Items →

Users ↓

0.09	–	–	–	–	–	–	0.05	–	–
–	–	0.02	–	0.03	–	–	–	–	0.06
–	0.07	–	–	–	0.04	–	–	–	0.04
–	0.05	–	–	–	–	0.06	–	–	–
–	–	0.03	0.05	–	–	–	0.01	–	–
0.01	–	–	–	0.07	–	–	–	–	–
–	–	–	–	0.06	–	–	0.10	–	–
0.02	–	–	–	–	–	0.07	–	–	–
–	–	0.12	0.05	–	–	–	–	–	0.11
–	0.11	–	–	–	0.07	–	0.08	–	–

- The goal is to predict the missing entries

Think of the low-rank model

- The matrix of ratings (user – item) can be expressed as $Z = (z_{ij})$
  - The rating by user i to item j is z_{ij}
- According to our assumption, the matrix is of low rank (k)
- We think …

$$z_{i,j} = [u_i^{(1)}, u_i^{(2)}, \ldots . u_i^{(k)}] \begin{bmatrix} v_j^{(1)} \\ v_j^{(2)} \\ \ldots \\ v_j^{(k)} \end{bmatrix}$$

SVD-CF

- We approximate Z by a low rank approximation of the user – item matrix M (a bit modified) that we have
- Method:
  - Compute the column averages to impute the missing values in M
  - Compute the row averages and subtract the row average from each element
  - Call this matrix A. Each row of A has average zero
  - Compute SVD of $A = R\ S\ L^T$
  - Compute best m-rank approximation of A

$$A_m = R(1:m)S(1:m,1:m)L^T(1:m) = Z$$

  - Predict missing value as

$$\hat{r}_{i,j} = \overline{r}_i + z_{ij}$$

High Dimensional Search
Min-Hashing
Locality Sensitive Hashing

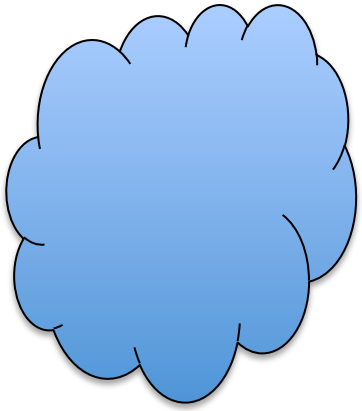
High Support Rules vs Correlation of Rare Items

- Association rule mining
  - Items, transactions
  - Itemsets: items that occur together
  - Consider itemsets (items that occur together) with minimum support
  - Form association rules
- Very sparse high dimensional data
  - Several *interesting* itemsets have negligible support
  - If support threshold is very low, many itemsets are frequent → high memory requirement
  - Correlation: *rare* pair of items, but high *correlation*
  - One item occurs → High *chance* that the other may occur

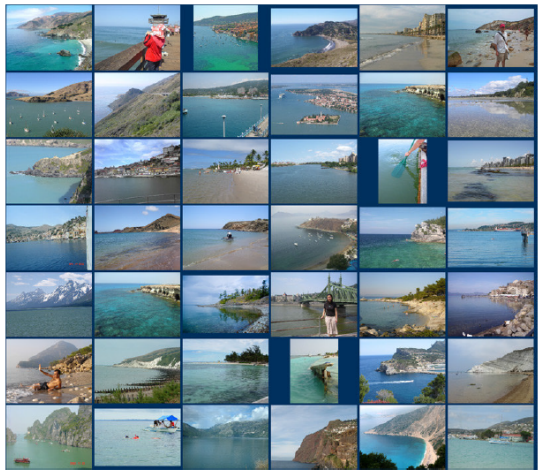
Source of this slide's material: http://www.eecs.berkeley.edu/~efros

Scene Completion: Hyes and Efros (2007)

Search for similar images among *many* images



Find *k* most similar images



Reconstruct the missing part of the image

Use Cases of Finding Nearest Neighbors

- Product recommendation
  - Products bought by same or similar customers
- Online advertising
  - Customers who visited similar webpages
- Web search
  - Documents with similar terms (e.g. the query terms)
- Graphics
  - Scene completion

Use Cases of Finding Nearest Neighbors

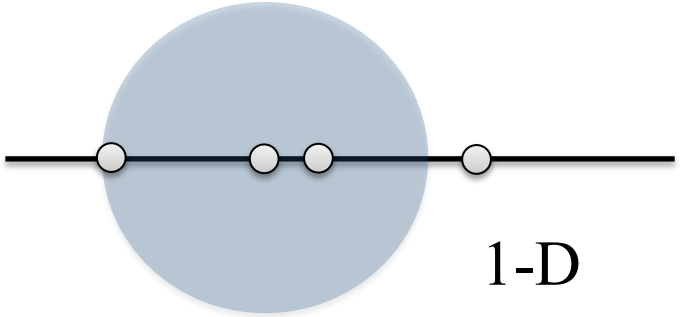
- Product recommendation
- Online advertising
- Web search
- Graphics

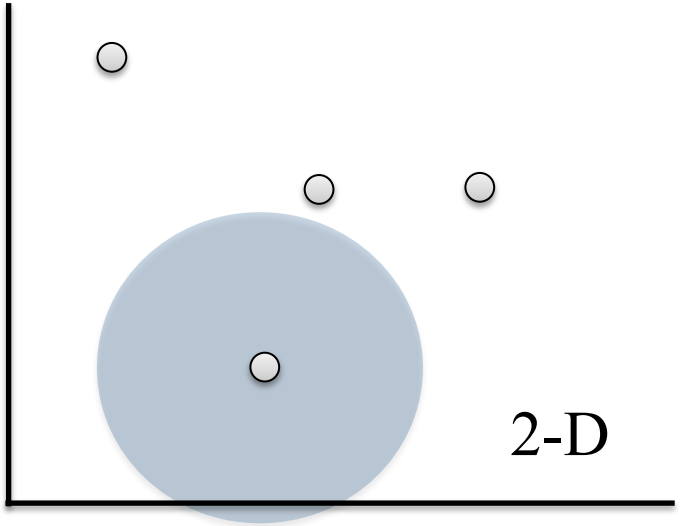
Use Cases of Finding Nearest Neighbors

- Product recommendation
  - Millions of products, millions of customers
- Online advertising
  - Billions of websites, Billions of customer actions, log data
- Web search
  - Billions of documents, millions of terms
- Graphics
  - Huge number of image features

All are very high dimensional spaces

The High Dimension Story





As dimension increases

- The average *distance* between points increases
- Less number of neighbors in the same radius

Data Sparseness

- Product recommendation
  - Most customers do not buy most products
- Online advertising
  - Most uses do not visit most pages
- Web search
  - Most terms are not present in most documents
- Graphics
  - Most images do not contain most features

But a lot of data are available nowadays

Distance

- Distance (metric) is a function defining distance between elements of a set X
- A distance measure $d : X \times X \rightarrow \mathbf{R}$ (real numbers) is a function such that
  1. For all $x, y \in X$, $d(x,y) \geq 0$
  2. For all $x, y \in X$, $d(x,y) = 0$ if and only if $x = y$ (reflexive)
  3. For all $x, y \in X$, $d(x,y) = d(y,x)$ (symmetric)
  4. For all $x, y, z \in X$, $d(x,z) + d(z,y) \geq d(x,y)$ (triangle inequality)

Distance measures

- Euclidean distance (L_2 norm)
  - Manhattan distance (L_1 norm)
  - Similarly, L_∞ norm
- Cosine distance
  - Angle between vectors to x and y drawn from the origin
- Edit distance between string of characters
  - (Minimum) number of *edit* operations (insert, delete) to obtain one string to another
- Hamming distance
  - Number of positions in which two bit vectors differ

Problem: Find Similar Documents

- Given a text document, find other documents which are very similar
  - Very similar set of words, or
  - Several sequences of words overlapping

- Applications
  - Clustering (grouping) search results, news articles
  - Web spam detection

- Broder et al. (WWW 2007)

Shingles

- *Syntactic Clustering of the Web:* Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, Geoffrey Zweig
- A document
  - A sequence of words, a canonical sequence of tokens (ignoring formatting, html tags, case)
  - Every document D is a set of subsequences or tokens $S(D,w)$
- Shingle: a contiguous subsequence contained in D
- For a document D, define its *w-shingling* $S(D, w)$ as the set of all unique shingles of size w contained in D
  - Example: the 4-shingling of (a,car,is,a,car,is,a,car) is the set

    { (a,car,is,a), (car,is,a,car), (is,a,car,is) }

Resemblance

- Fix a large enough w, the size of the shingles
- Resemblance of documents A and B

$$r(A,B) = \frac{|S(A,w) \cap S(B,w)|}{|S(A,w) \cup S(B,w)|}$$

Jaccard similarity between two sets

- *Resemblance distance* is a *metric*

$$d(A,B) = 1 - r(A,B)$$

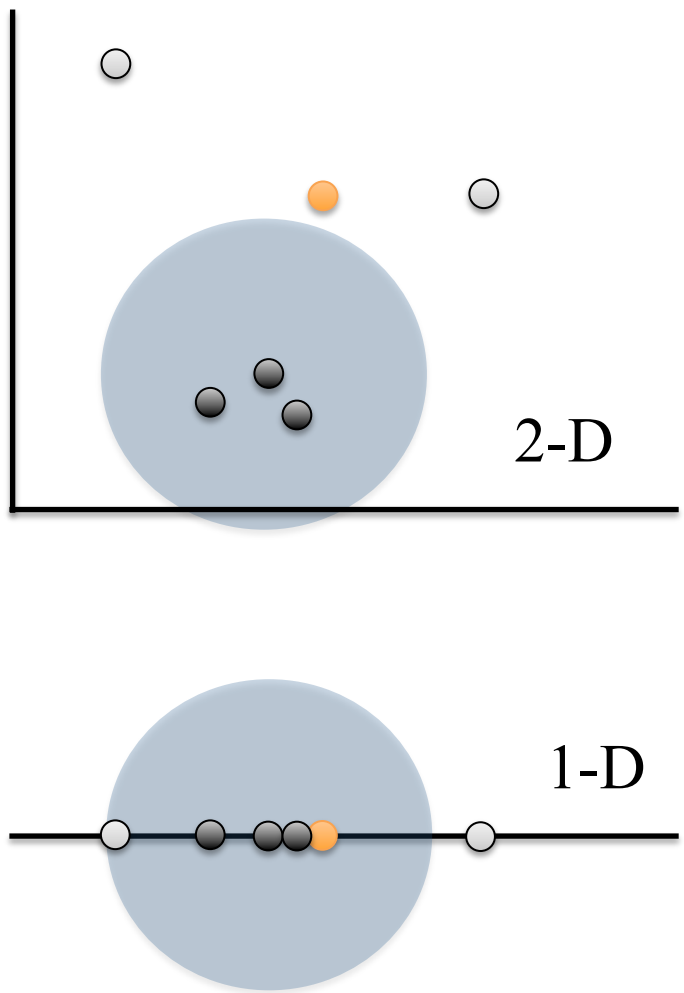
- Containment of document A in document B

$$c(A,B) = \frac{|S(A,w) \cap S(B,w)|}{|S(A,w)|}$$

Brute Force Method

- We have: N documents, similarity / distance metric
- Finding similar documents in brute force method is expensive
  - Finding similar documents for one given document: $O(N)$
  - Finding pairwise similarities for all pairs: $O(N^2)$

Locality Sensitive Hashing (LSH): Intuition



- Two points are close to each other in a high dimensional space → They remain close to each other after a "projection" (map)
- If two points are not close to each other in a high dimensional space, they may come close after the mapping
- However, it is quite likely that two points that are far apart in the high dimensional space will preserve some distance after the mapping also

LSH for Similar Document Search

- Documents are represented as set of shingles
  - Documents D_1 and D_2 are points at a (very) high dimensional space
  - Documents as vectors, the set of all documents as a matrix
  - Each row corresponds to a shingle,
  - Each column corresponds to a document
  - The matrix is *very* sparse
- Need a hash function h, such that
  1. If $d(D_1, D_2)$ is high, then $dist(h(D_1), h(D_2))$ is high, with high probability
  2. If $d(D_1, D_2)$ is low, then $dist(h(D_1), h(D_2))$ is low, with high probability
- Then, we can apply h on all documents, put them into hash buckets
- Compare only documents in the same bucket

Some appropriate distance function, not the same as d

Min-Hashing

- Defining the hash function h as:
  1. Choose a random permutation σ of m = number of shingles
  2. Permute all rows by σ
  3. Then, for a document D, $h(D)$ = index of the first row in which D has 1

	D1	D2	D3	D4	D5
S1	0	1	1	1	0
S2	0	0	0	0	1
S3	1	0	0	0	0
S4	0	0	1	0	0
S5	0	0	0	1	0
S6	0	1	1	0	0
S7	1	0	0	0	0
S8	1	0	0	0	1
S9	0	1	1	0	0
S10	0	0	1	0	0

σ
3
1
7
10
6
2
5
9
8
4

	D1	D2	D3	D4	D5
S2	0	0	0	0	1
S6	0	1	1	0	0
S1	0	1	1	1	0
S10	0	0	1	0	0
S7	1	0	0	0	0
S5	0	0	0	1	0
S3	1	0	0	0	0
S9	0	1	1	0	0
S8	1	0	0	0	1
S4	0	0	1	0	0

$h(D)$

D1	D2	D3	D4	D5
5	2	2	3	1

Property of Min-hash

- How does Min-Hashing help us?
- Do we retain some important information after hashing high dimensional vectors to one dimension?

- Property of MinHash
- The probability that D_1 and D_2 are hashed to the same value is same as the *resemblance* of D_1 and D_2
- In other words,

$$\mathrm{P}[h(D_1) = h(D_2)] = r(D_1, D_2)$$

Proof

	D1	D2
Type 11	1	1
Type 10	1	0
Type 01	0	1
Type 00	0	0

- There are four types of rows
- Let n_x be the number of rows of type x $\in \{11, 01, 10, 00\}$
- Note: $r(D_1, D_2) = \dfrac{n_{11}}{n_{11} + n_{10} + n_{01}}$

- Now, let σ be a *random* permutation. Consider $\sigma(D_1)$ and $\sigma(D_2)$
- Let j be the index of the first 1 across $\sigma(D_1)$ and $\sigma(D_2)$
- Let x_j be the type of the j-th row
- Observe: If $h(D_1) = h(D_2) = j$ then $x_j = 11$
- Also, If $x_j = 11$, then $h(D_1) = h(D_2) = j$
- Also, $x_j \neq 00$ in any case
- So,

$$P\left[x_j = 11\right] = \frac{n_{11}}{n_{11} + n_{10} + n_{01}} = r(D_1, D_2)$$

Using one min-hash function

- *High* similarity documents go to same bucket with *high* probability
- Task: Given D_1, find similar documents with at least 75% similarity
- Apply min-hash:
  - Documents which are 75% similar to D_1 fall in the same bucket with D_1 with 75% probability
  - Those documents do not fall in the same bucket with about 25% probability
  - Missing similar documents and false positives

Min-hash Signature

Hundreds, but still less than the number of dimensions

- Create a signature for a document D using *many* independent min-hash functions
- Compute similarity of columns by the similarity in their signatures

Signature matrix

		D1	D2	D3	D4	D5
SIG(1)	h_1	5	2	2	3	1
SIG(2)	h_2	3	1	1	5	2
SIG(3)	h_3	1	4	4	1	3
	…	…	…	…	…	…
SIG(n)	h_n	…	…	…	…	…

Example (considering only 3 signatures):

$\text{Sim}_{\text{SIG}}(D_2,D_3) = 1$
$\text{Sim}_{\text{SIG}}(D_1,D_4) = 1/3$

Observe:
$E[\text{Sim}_{\text{SIG}}(D_i,D_j)] = r(D_i,D_j)$ for any $0 < i, j < N$ (#documents)

Computational Challenge

- Computing signature matrix of a large matrix is expensive
  - Accessing random permutation of billions of rows is also time consuming
- Solution:
  - Pick a hash function $h : \{1, \ldots, m\} \rightarrow \{1, \ldots, m\}$
  - Some pairs of integers will be hashed to the same value, some values (buckets) will remain empty
  - Example: $m = 10$, $h : k \rightarrow (k + 1) \bmod 10$
  - *Almost* equivalent to a permutation

Computing a Signature Matrix

- Pick n hash functions on the rows (not permutations): $h_1, h_2, \ldots, h_n$
- Let SIG(i,j) = the (i,j)-th entry of the signature matrix (i-th hash function, j-th document)

```
For each row r BEGIN
    Compute h1(r), h2(r),…, hn(r)
    For each column j BEGIN
        If the j-th column has 1 in row r
            For each i = 1, 2, … , n BEGIN
                set SIG(i,j) = min{SIG(i,j), hi(r)}
            END
        END IF
    END
END
```

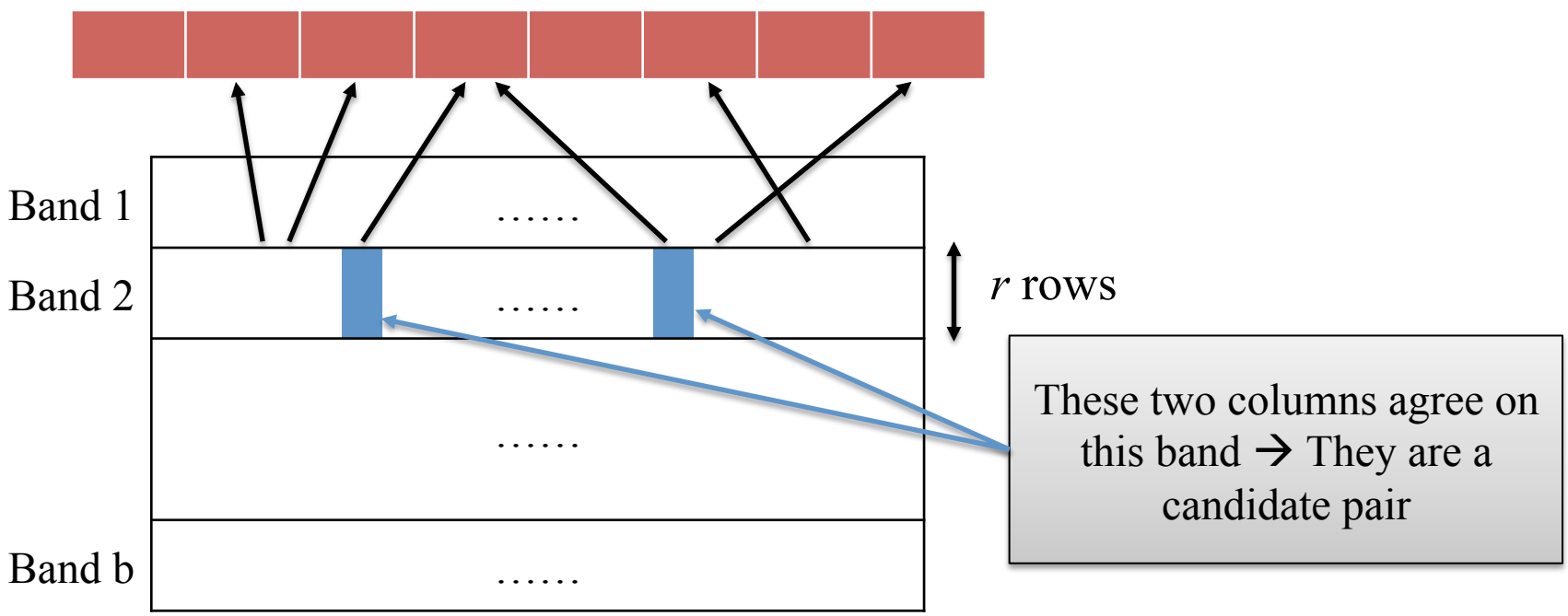
Example

Locality Sensitive Hashing

- Suppose there is a hashing scheme such that
  - Each time hashing: similar documents are *likely* to fall into same bucket, dissimilar documents are *less likely* to fall into same bucket
- Main idea
  - Hash several times: Dissimilar documents are *very unlikely* to fall into the same bucket *several times*
  - Two documents fall into the same bucket several times → They are likely to be similar
  - Candidate pair: a pair of documents which goes to the same bucket at least some number of times

Locality Sensitive Hashing

- Banding of hash functions: *b* bands, *r* rows each
- For each band, hash portion of each column to some bucket (*k* buckets)
- Two columns agree on at least one band → the corresponding pair of documents is a candidate pair



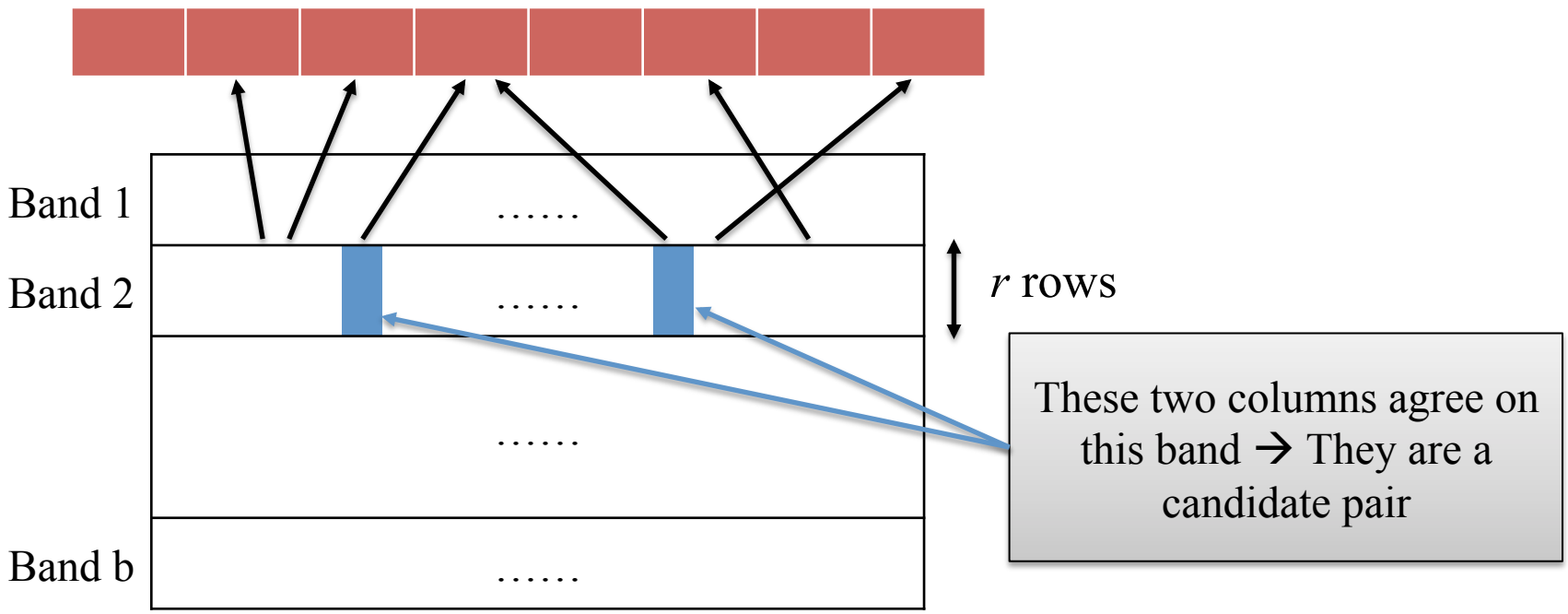
Locality Sensitive Hashing

A band (portion) of two columns hashing to same bucket

→ High probability that those bands of those columns are identical

→ Signature of two documents matching significantly

→ Candidate pairs



Analysis of LSH

Signature: n (hash functions) $\times$ N, b bands, r rows per band

For two documents, let the resemblance (similarity) be s

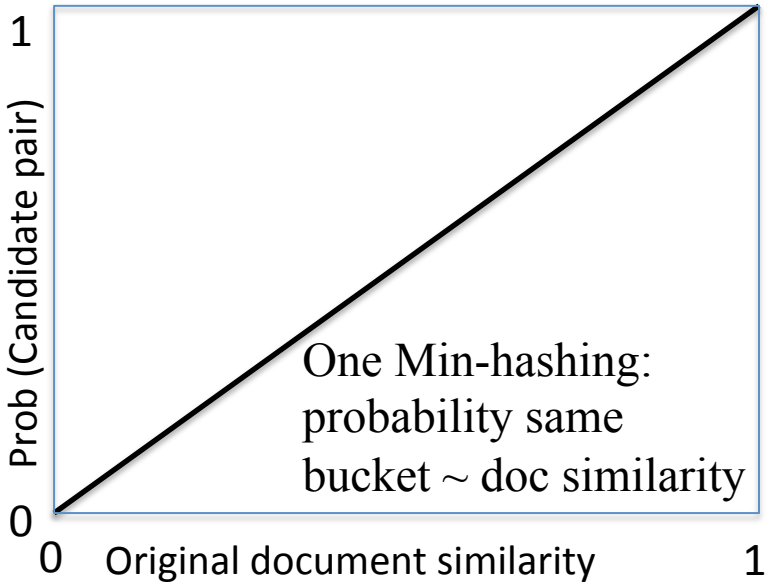
P[Signature agree in all rows of one particular band]
$= s^r$

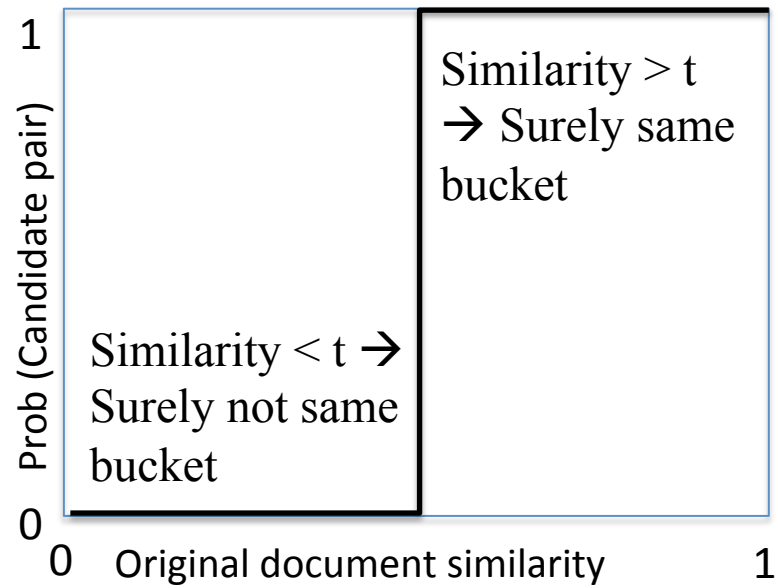
P[Signature don't agree in at least one row of one particular band]
$= 1 - s^r$

P[Signature don't agree in all rows of any of the bands]
$= (1 - s^r)^b$

P[Signature agree in all rows of at least one band]
$1 - (1 - s^r)^b$

What we want

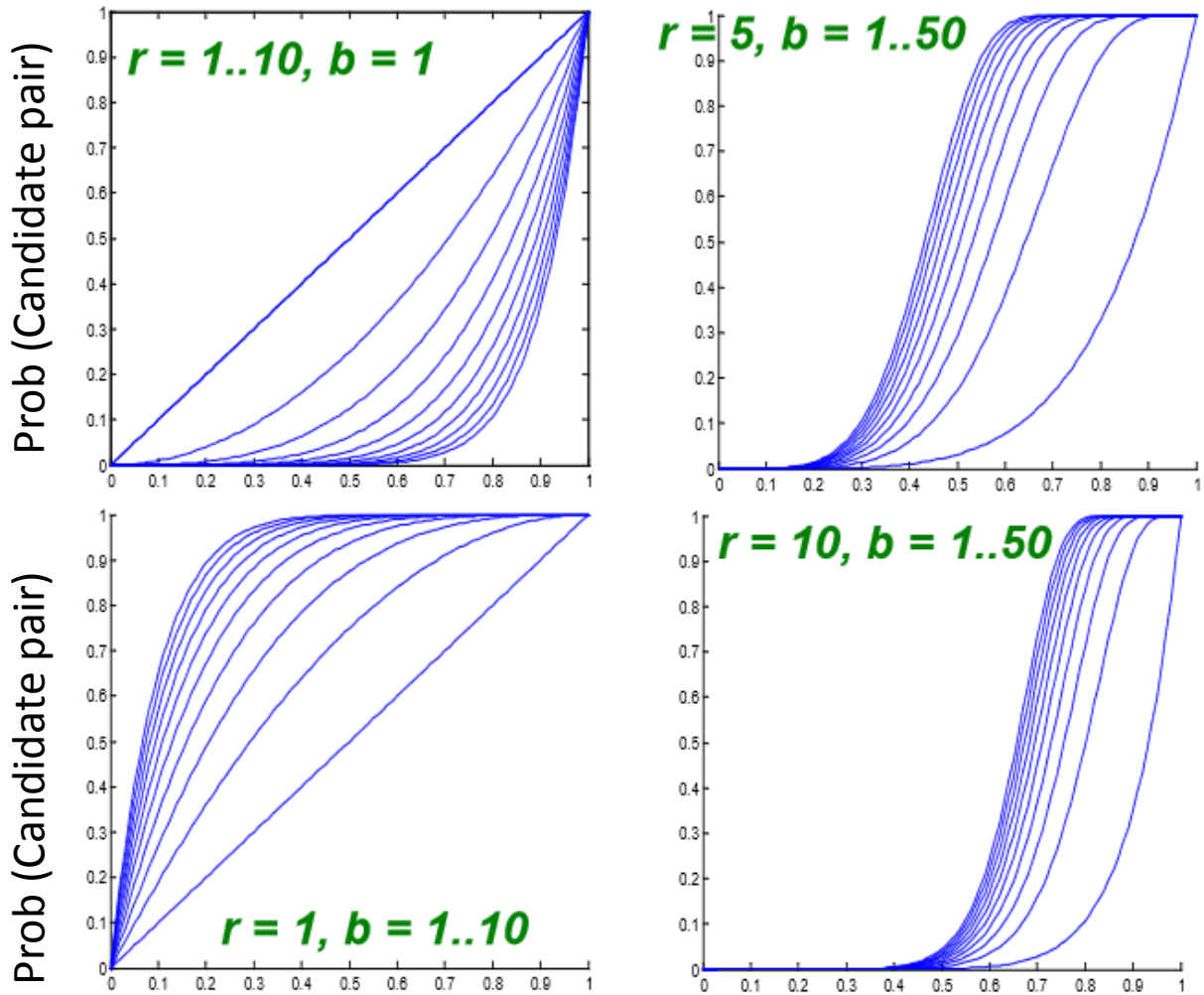




Tune *r* and *b* to get the desired step function

Tuning *b* and *r*

By tuning *b* and *r* we can get a desired step function



Resemblance (similarity) of documents

Generalization: LSH Family of Functions

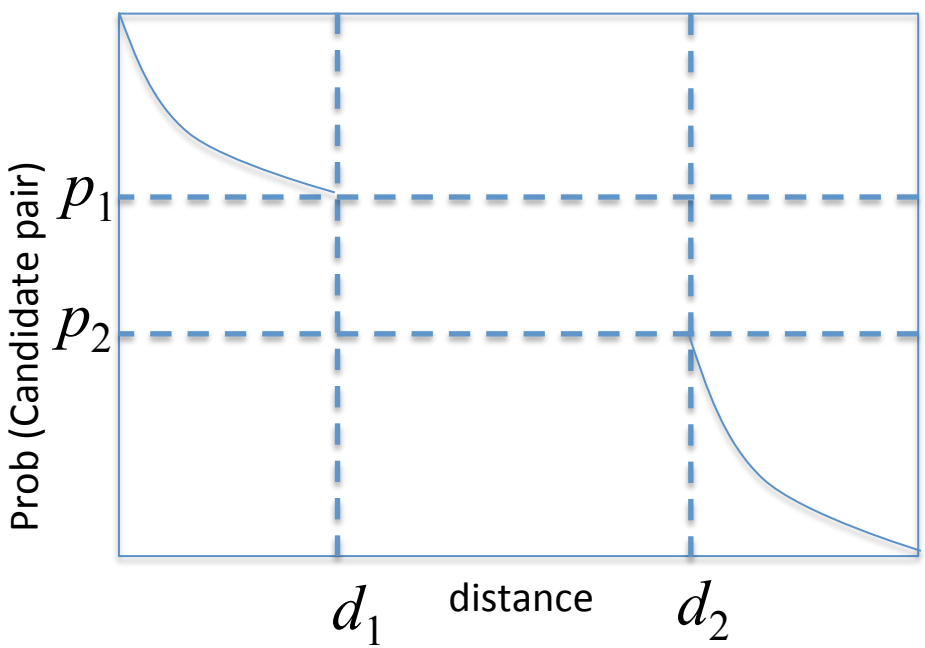
- Conditions for the family of functions
  1. Declares closer pairs as candidate pairs with higher probability than a pair that are not close to each other
  2. Statistically independent: product rule for independent events can be used
  3. Efficient in identifying candidate pairs much faster than exhaustive pairwise computation
  4. Efficient in combining for avoiding false positives and false negatives

General Definition

- Let $d_1 < d_2$ be two distances (say between two pairs of points)
- A family **F** of functions is said to be (d_1, d_2, p_1, p_2)-sensitive, if for every $f \in \mathbf{F}$, and for some $0 \le p_1, p_2 \le 1$ we have:
  1. If $d(x,y) \le d_1$ then $P[f(x) = f(y)] \ge p_1$
  2. If $d(x,y) \ge d_2$ then $P[f(x) = f(y)] \le p_2$

If two points are close enough, then probability that they are mapped to the same value is high enough

If two points are far enough, then probability that they are mapped to the same value is small enough



The Min-Hash Function

Class exercise:

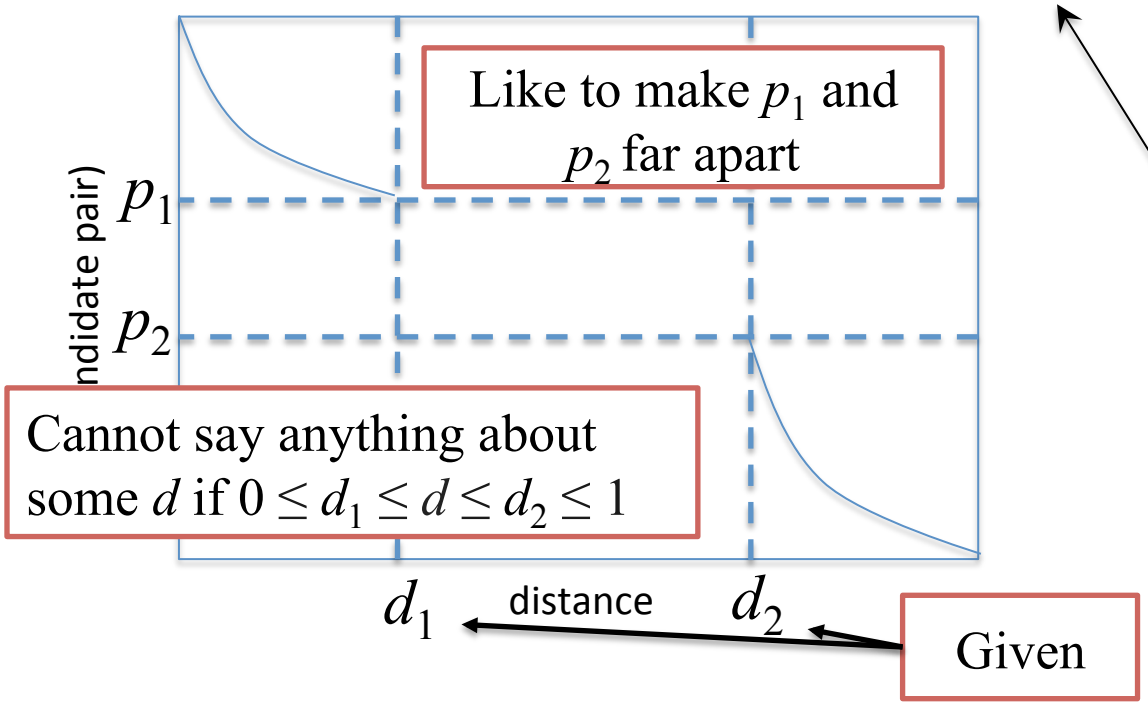
- The family of min-hash functions is $(d_1, d_2, 1 - d_1, 1 - d_2)$-sensitive for any d_1 and d_2 such that $0 \leq d_1 \leq d_2 \leq 1$

General Definition

- Let $d_1 < d_2$ be two distances (say between two pairs of points)
- A family **F** of functions is said to be (d_1, d_2, p_1, p_2)-sensitive, if for every $f \in \mathbf{F}$, and for some $0 \le p_1, p_2 \le 1$ we have:
  1. If $d(x,y) \le d_1$ then $P[f(x) = f(y)] \ge p_1$
  2. If $d(x,y) \ge d_2$ then $P[f(x) = f(y)] \le p_2$

If two points are close enough, then probability that they are mapped to the same value is high enough

If two points are far enough, then probability that they are mapped to the same value is small enough



Amplifying an LSH Family

- Suppose **F** is a family (d_1, d_2, p_1, p_2)-sensitive functions
- Then another family **F'** of functions can be constructed from **F** such that **F'** is (d_1, d_2, p_1^r, p_2^r) sensitive for some integer $r > 0$
- *AND-Construction*:

  Fix any $0 < r < |\mathbf{F}|$

  Define each $f \in \mathbf{F'}$ such that if and only if for some set of r indices $i = 1, \ldots, r$

  Now:

  1. If $d(x,y) \leq d_1$ then $\geq p_1$ for all $i = 1, \ldots, r$ $\geq$
  2. If $d(x,y) \geq d_2$ then $\leq p_2$ for all $i = 1, \ldots, r$ $\leq$

  Since s are independent

Amplifying an LSH Family

- Suppose **F** is a family (d_1, d_2, p_1, p_2)-sensitive functions
- Then another family **F'** of functions can be constructed from **F** such that **F'** is (d_1, d_2, p_1^r, p_2^r) sensitive for some integer $r > 0$
- *AND-Construction*:

  Fix any $0 < r < |\mathbf{F}|$

  Define each $f \in \mathbf{F'}$ such that if and only if for some set of r indices $i = 1, \ldots, r$

  Now:

  1. If $d(x,y) \leq d_1$ then $\geq p_1$ for all $i = 1, \ldots, r$ $\geq$
  2. If $d(x,y) \geq d_2$ then $\leq p_2$ for all $i = 1, \ldots, r$ $\leq$

  Since s are independent

Amplifying

- The AND Construction is the effect of combining r rows into a single band
  - Two documents form a candidate pair if and only if they are hashed to the same bucket in all rows of the band
- Similarly, an OR-Construction gives us a $(d_1, d_2, 1-(1-p_1)^b, 1-(1-p_2)^b)$-sensitive family
  - The effect of b bands
  - Two documents form a candidate pair if and only if they are hashed to the same bucket in at least one band
- The AND-construction lowers probabilities, OR-construction increases probabilities
- With carefully chosen r and b
  - For AND, push p_2 very close to 0, keep p_1 significantly higher
  - For OR, push p_1 very close to 1, keep p_2 significantly lower

Sentiment Analysis

Making sense of what people write

Background

Once upon a time in the world

- There was mostly edited content available
- It was not so easy to express our opinion

Nowadays

- People can express themselves on the web easily
- Blogs, Twitter, Facebook, Review sites, …

Blogs

Anyone can write anything, and people do …

Last evening, we were told that we are being investigated upon for match fixing. I mean, this is really funny. When I joined this team I had no clue that we will make so much news off the field. I mean, if there was an IPL for off-field screw ups, we'd have won pads down. Unbelievable. Reminds me of that 3-patti game, muflis or something. The worst hand wins. Wish we had muflis in IPL. We'd bee millionaires by now.

You know what's funnier abt the match fixing investigation? No one seems to be asking the most logical question. Why would someone pay us to lose a match when we are doing the same for free? Anyway, all these allegations are bollocks. It's just a way for some officials & their families to get free tickets and stay for the semis-finals weekend.

fakeiplplayer.blogspot.com, 2009

Reviews in review site

EUREKA FORBES LTD.

3% 4% 10% 13% 70%

Write Your Review

Analyze Ratings

Reviews & Ratings (107) | Photos | Discussion

USER REVIEWS ON EUREKA FORBES

Ratings and textual reviews

first reviewed by koolravs

Sort Reviews By : Review Date

mauryasantkumar
Robertsganj, India

Reviews: 1

AVOID EUREKA FORBES products

Mar 18, 2014 11:43 AM

Read 98 times
Comments (0)

Customer Service: Staff Courtesy:

We are having Eureka Forbes Amrit water purifier and have placed a complaint(75394894) one month back.
No one responds to the complaint. The call center is placing request and no one responds. This is cheap POOR, SHABBY, IRRESPONSIBLE service

Reviews in review site

42 people have reviewed this hotel

Write a Review

Traveller rating

Excellent	5
Very good	21
Average	10
Poor	5
Terrible	1

See reviews for

Families	18
Couples	9
Solo	3
Business	6

Rating summary

Location	●●●●○
Sleep Quality	●●●◐○
Rooms	●●●◐○
Service	●●●◐○
Value	●●●◐○
Cleanliness	●●●◐○

Traveller tips help you choose the right room. Room tips (14)

Ratings and textual reviews

tripadvisor.in

Reviews in shopping site

REVIEWS OF APPLE IPHONE 5S

4.3

Average Rating
Based on 443 ratings
Read certified buyer reviews

5 star	331
4 star	43
3 star	10
2 star	9
1 star	50

PRODUCT FEATURES USERS TALKED ABOUT IN THEIR REVIEWS

Screen/display (in 48 reviews)

Value for money (in 45 reviews)

Camera (in 38 reviews)

Battery (in 22 reviews)

Build quality/design (in 18 reviews)

Ratings and textual reviews

flipkart.com

Social networking sites

Twitter

Facebook

No explicit rating, more free text. What do they mean?

Making sense of so much data

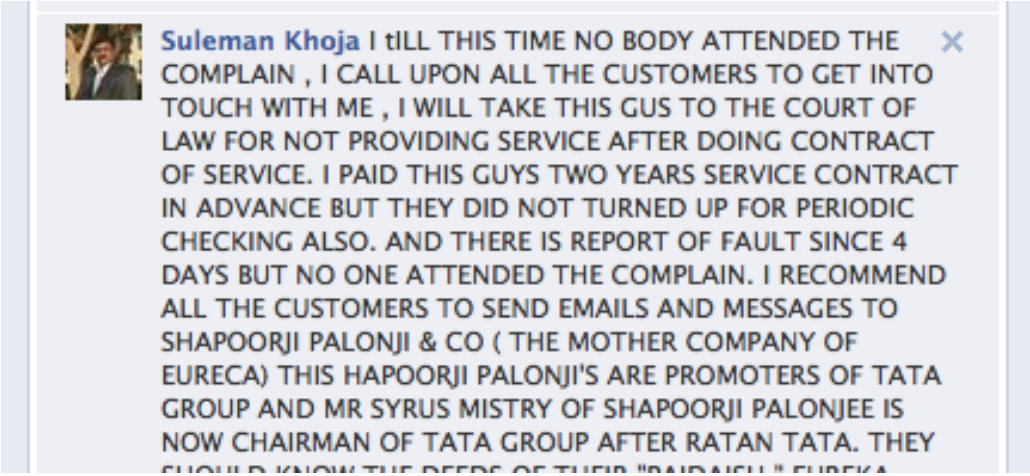
- Review sites: semi structured, ratings and details
- Blogs: unstructured
- Social networks: very unstructured, noisy
- Sentiment analysis
  - Making sense of so much of unstructured expressions by people
  - Giving some structure to these expressions
  - Quantifying qualitative statements by people

Sentiment analysis

MouthShut.com @MouthShut_com · Feb 20
1/5 #Review on **#EurekaForbes** by prasadraoj : Unresponsive-sales-team- - bit.ly/1gZ2oiN

Expand Reply Retweet Favorite More

Some customer is **unhappy** about the **sales team**


Suleman Khoja I tILL THIS TIME NO BODY ATTENDED THE COMPLAIN , I CALL UPON ALL THE CUSTOMERS TO GET INTO TOUCH WITH ME , I WILL TAKE THIS GUS TO THE COURT OF LAW FOR NOT PROVIDING SERVICE AFTER DOING CONTRACT OF SERVICE. I PAID THIS GUYS TWO YEARS SERVICE CONTRACT IN ADVANCE BUT THEY DID NOT TURNED UP FOR PERIODIC CHECKING ALSO. AND THERE IS REPORT OF FAULT SINCE 4 DAYS BUT NO ONE ATTENDED THE COMPLAIN. I RECOMMEND ALL THE CUSTOMERS TO SEND EMAILS AND MESSAGES TO SHAPOORJI PALONJI & CO (THE MOTHER COMPANY OF EURECA) THIS HAPOORJI PALONJI'S ARE PROMOTERS OF TATA GROUP AND MR SYRUS MISTRY OF SHAPOORJI PALONJEE IS NOW CHAIRMAN OF TATA GROUP AFTER RATAN TATA. THEY

Some customer is **unhappy** about the **service.**

What do people express?

EXPRESSIONS AND STATES

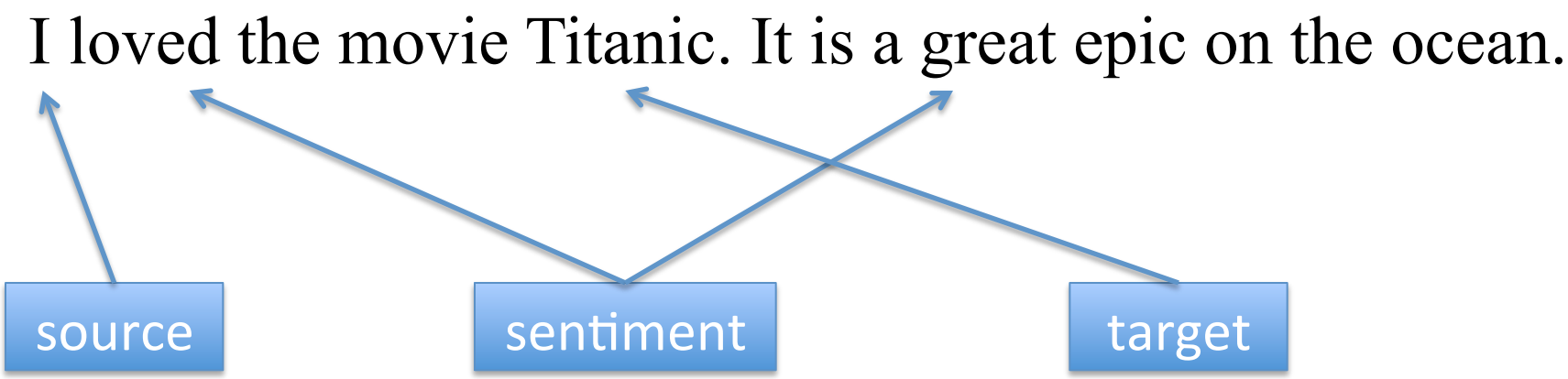
Typology of Affective States [Scherer 1984]

- **Emotion**: brief organically synchronized … evaluation of a major event
  - *angry, sad, joyful, fearful, ashamed, proud, elated*
- **Mood**: diffuse non-caused low-intensity long-duration change in subjective feeling
  - *cheerful, gloomy, irritable, listless, depressed, buoyant*
- **Interpersonal stances**: affective stance toward another person in a specific interaction
  - *friendly, flirtatious, distant, cold, warm, supportive, contemptuous*
- **Attitudes**: enduring, affectively colored beliefs, dispositions towards objects or persons
  - *liking, loving, disliking, hating, valuing, desiring*
- **Personality traits**: stable personality dispositions and typical behavior tendencies
  - *nervous, anxious, reckless, morose, hostile, jealous*

Scope of sentiment analysis

- **Attitudes**: enduring, affectively colored beliefs, dispositions towards objects or persons
  - *liking, loving, disliking, hating, valuing, desiring*
- Simplify to liking and not liking (hating)
- Simple task: Detect polarity of a text
  - Positive sentiment / negative sentiment
- Complex task: rate the sentiment in a more granular scale 1 – 5 (for example)
  - Strongly positive, weakly positive, neutral, weakly negative, strongly negative
- More complex task: detect the sentiment, the source and the target

More complex task



- In some cases the target (aspect) may not be in the text as well, it may be understood from the context or metadata
  - A movie review, the name of the movie may not be explicitly mentioned

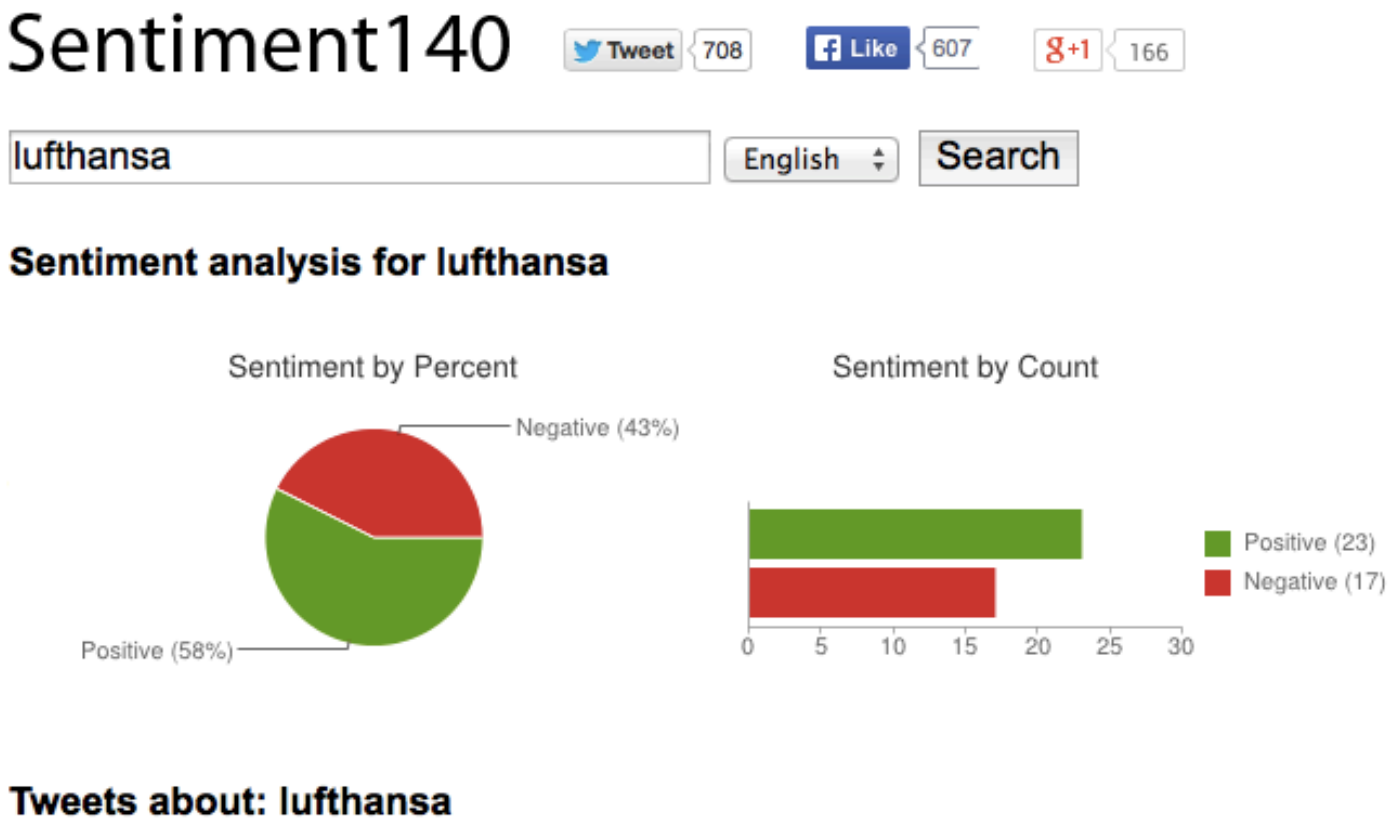
What do people do with it? Let’s see a few examples.

APPLICATIONS

Twitter sentiment analysis

Analyzing recent sentiments about a brand or product

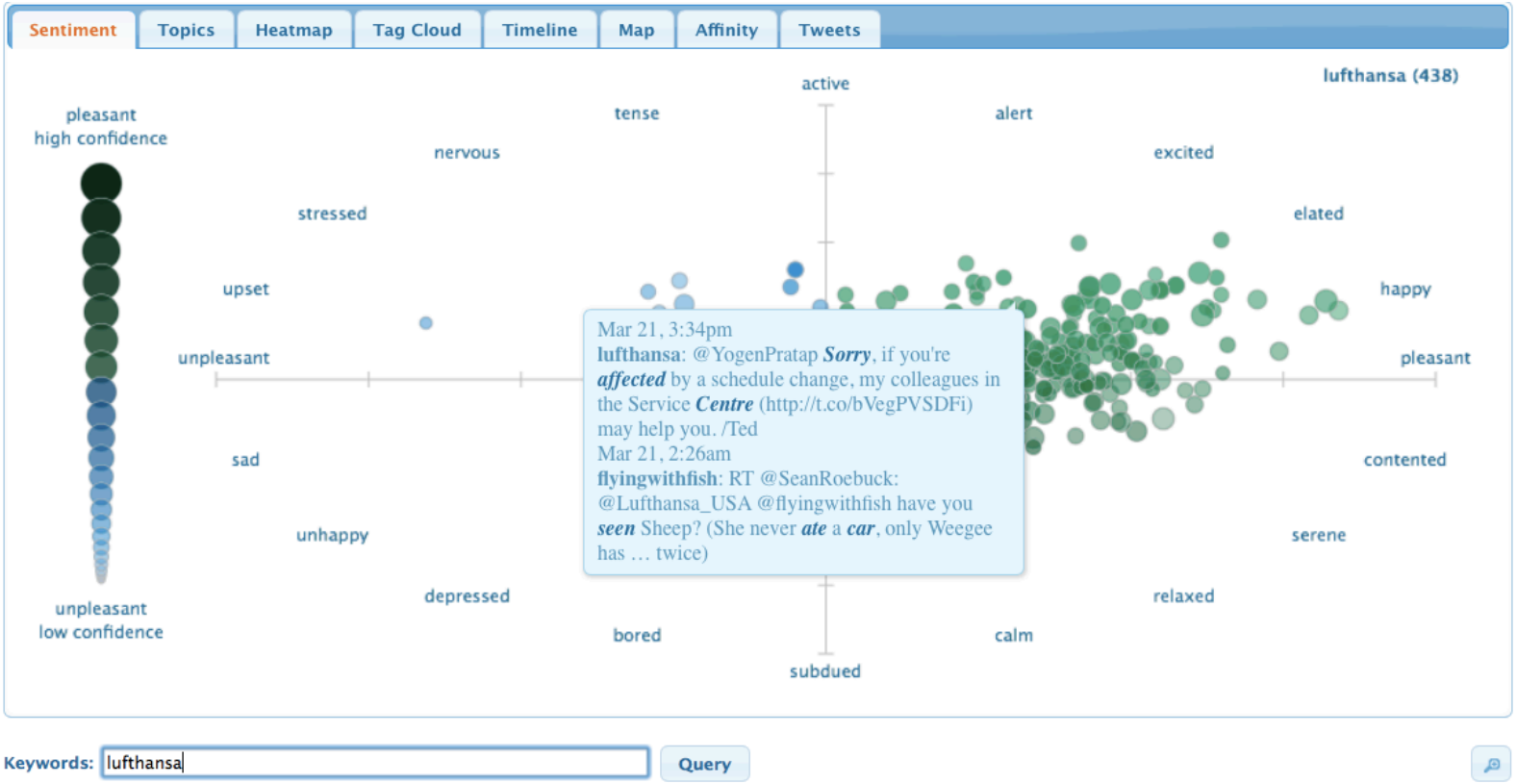
http://www.sentiment140.com



Twitter sentiment analysis

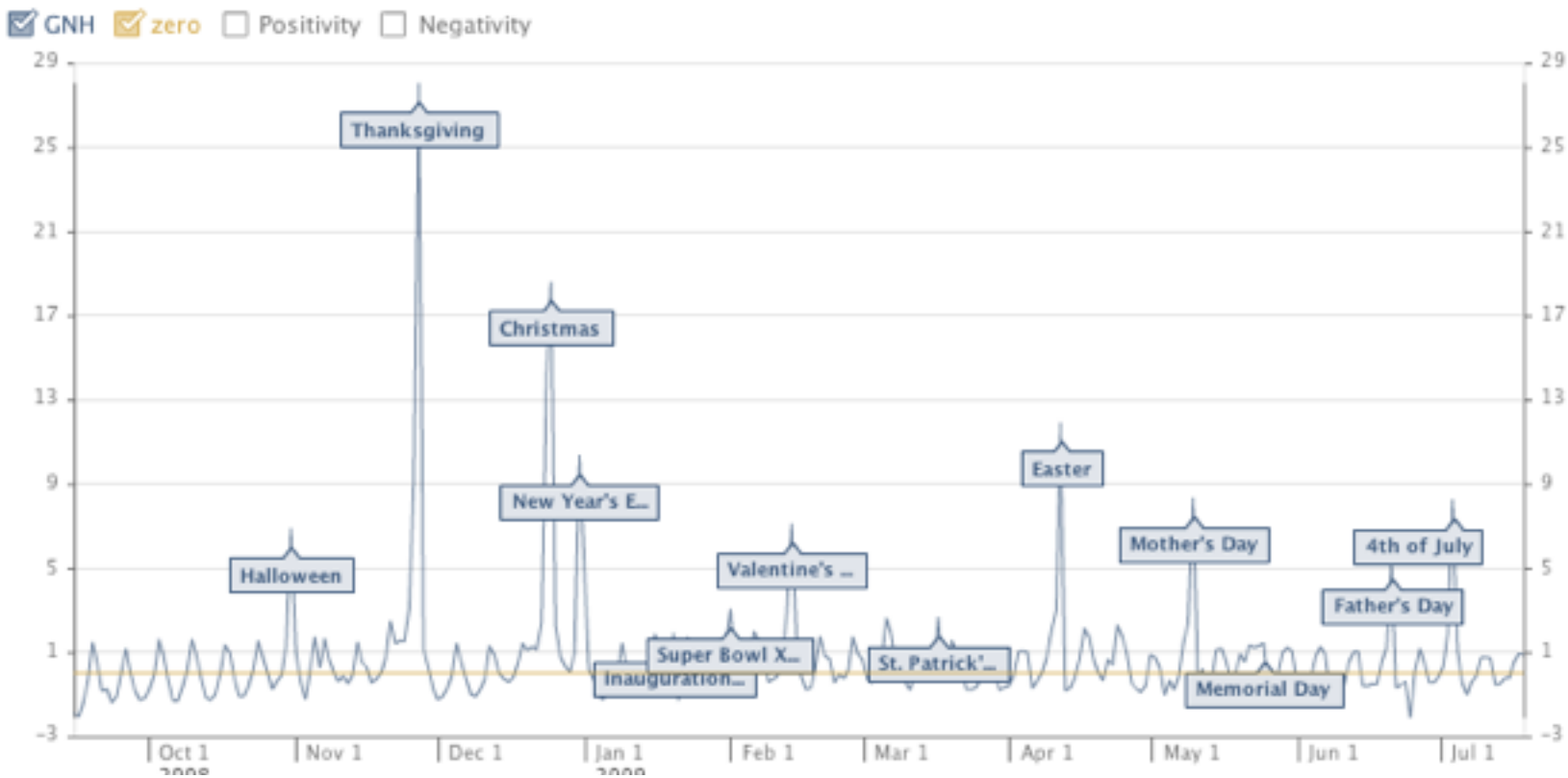
Analyzing recent sentiments about a brand or product

http://www.csc.ncsu.edu/faculty/healey/tweet_viz/tweet_app/



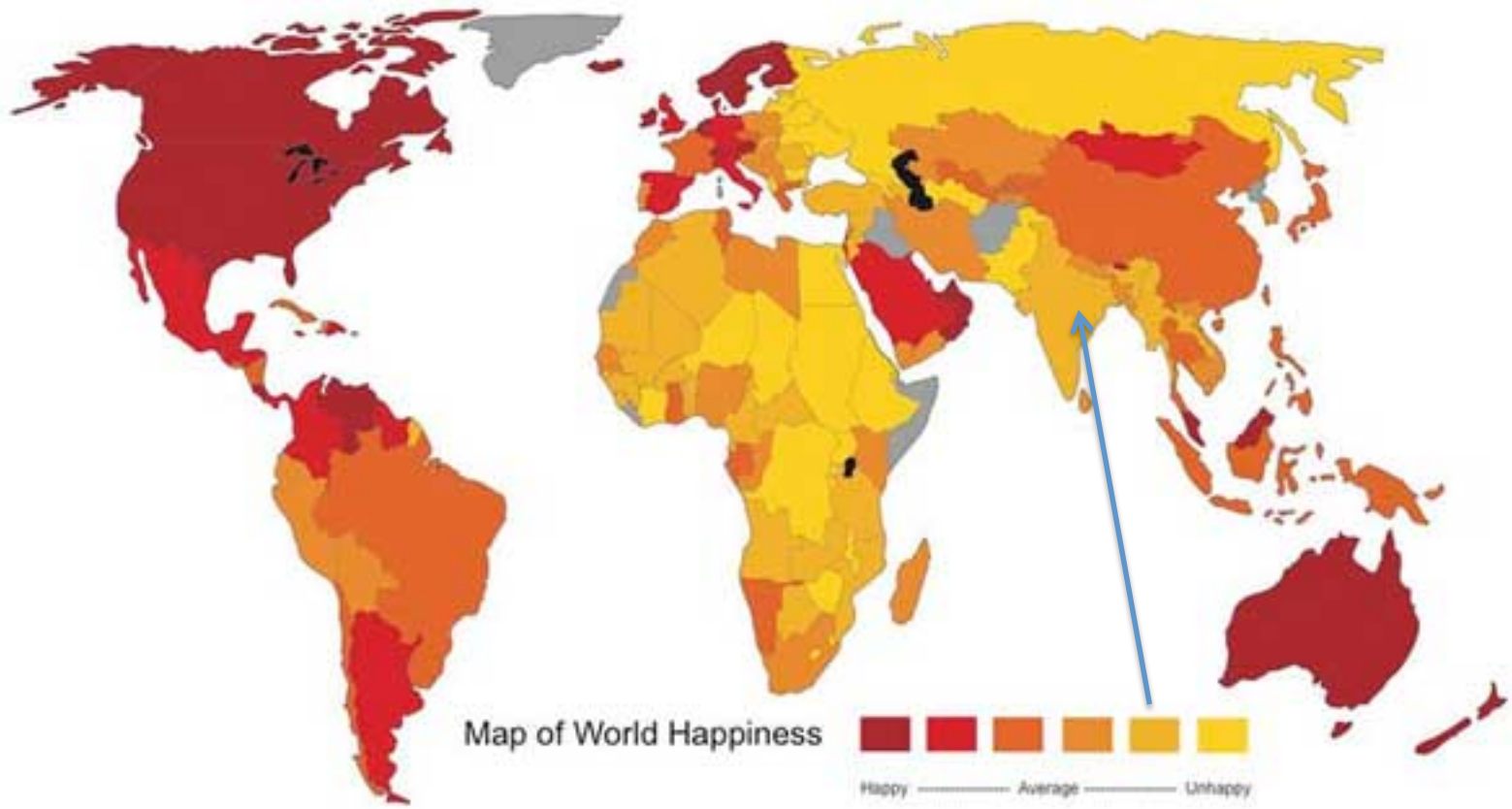
Facebook Gross National Happiness

Facebook used to have a GNH index app



Festivals make people happier (at least they express it publicly), and some catastrophic events make them unhappy

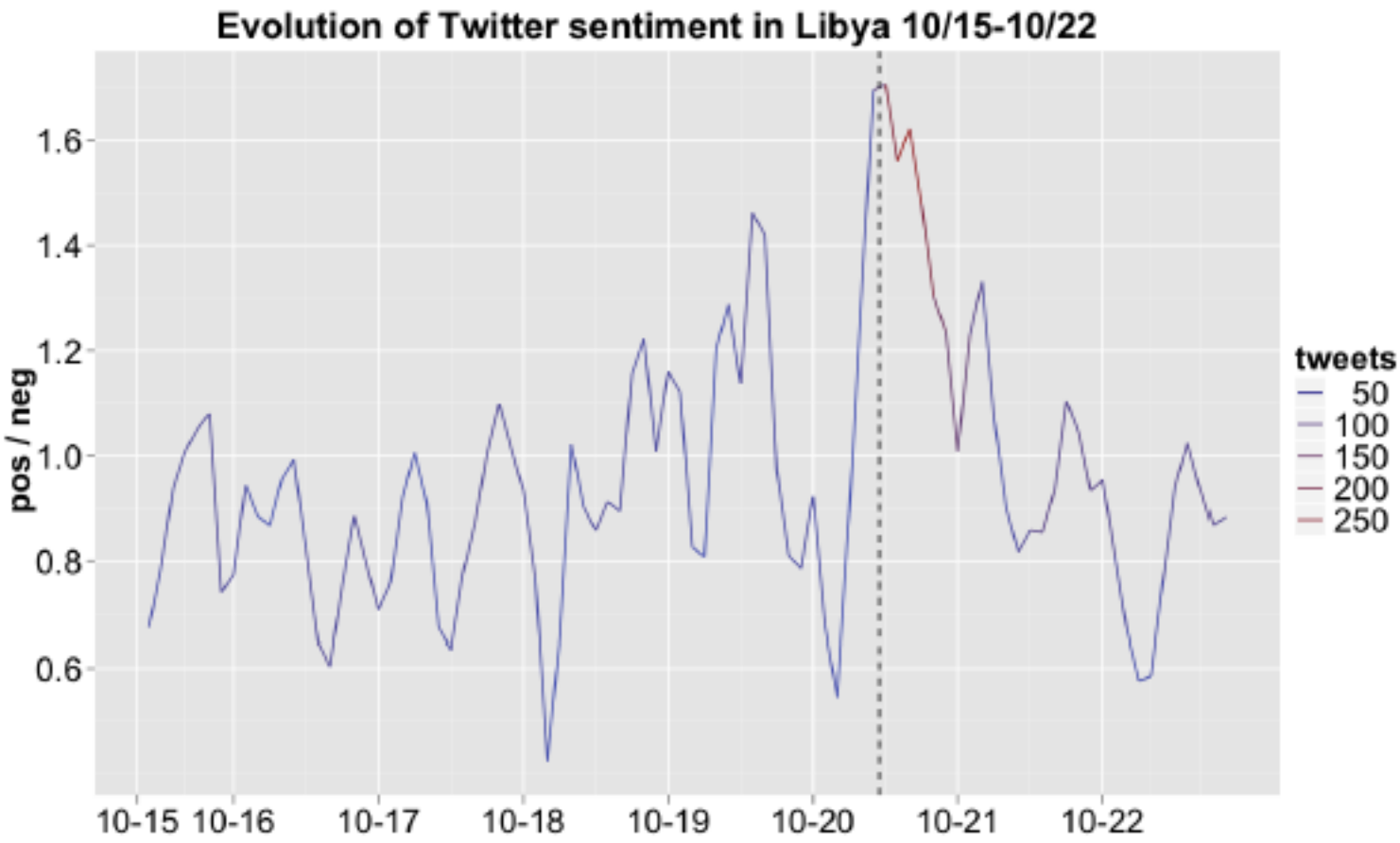
Facebook GNH map of the world



We are almost among the most unhappy people.

Do we agree?

Twitter sentiments



- Twitter sentiments in Libya
- The vertical line is the time when it was announced that Gaddafi was killed

Detect actionable items

Detect actionable items in social media [IBM Research]

http://researcher.watson.ibm.com/researcher/view_project.php?id=4290

Nothing to do

Take action

Take action

One exercise – let's do it by hand

A hotel review by a customer:

At a first glance, the hotel looked somewhat old to me. The room we got was spacious and decent. The facilities were more or less satisfactory. The food was good, my son liked the fish curry a lot.

How will we do a sentiment analysis?

Challenges

- Subtlety
  - Someone reviews a perfume: "If you are reading this because it is your darling fragrance, please use it at home exclusively, and keep the windows shut."
- Expectation and reality mismatch
  - "This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it **can't hold up**."

Challenges

- Ordering effects:
  - They said the movie would be great, and they were right
  - They said the movie would be great, and they were wrong
- More subtlety
  - Oh! you're terrible!! [read – oh you are amazing]

নিবারণ । বাপ না হয় রাজী হল কিন্তু মেয়ে কি বলে?

সত্য । বড় গোলমেলে জবাব দিচ্ছে।

নিবারণ । কি বলে বুঁচকী?

সত্য । বললে - যাঃ ।

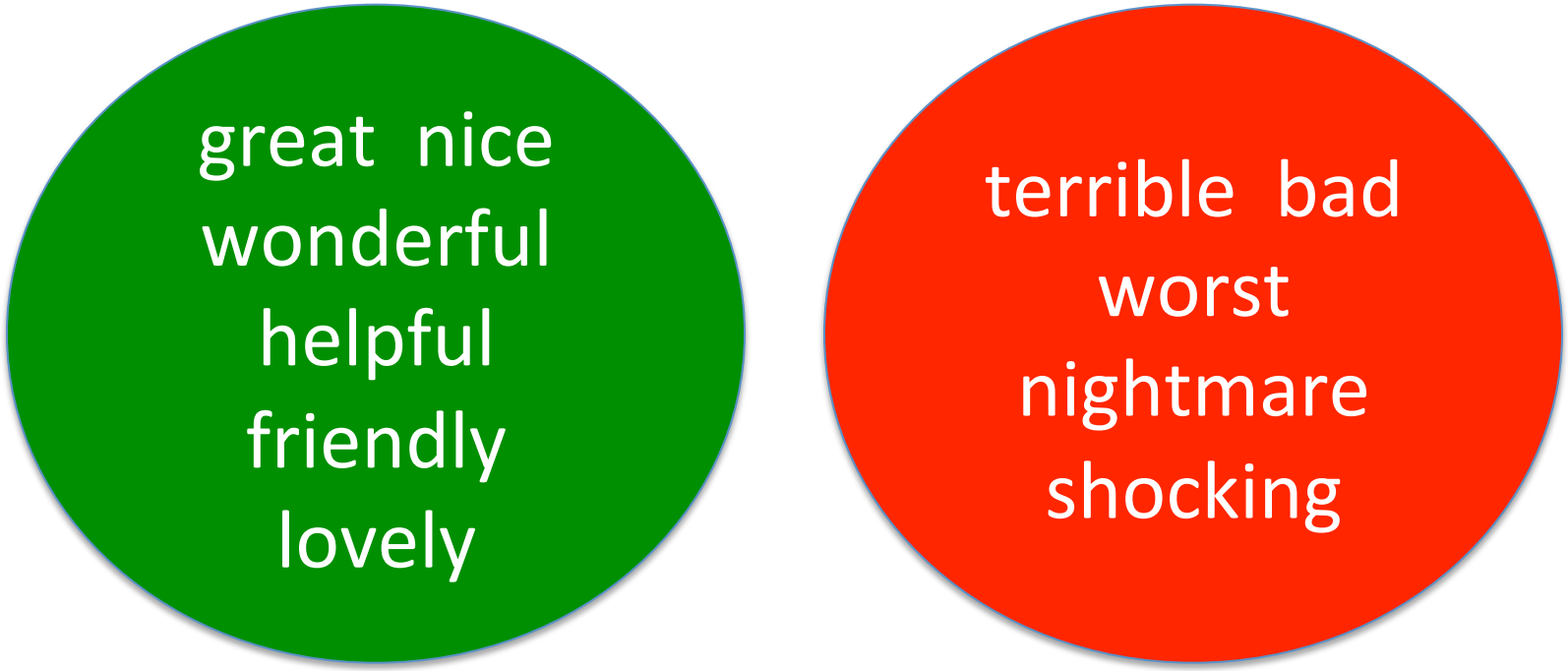
নিবারণ । দূর গাধা, যাঃ মানেই হ্যাঁঃ ।

Classification algorithms

- Naïve Bayes
- Support Vector Machine
- MaxEnt – maximum entropy classifier

  http://www.kamalnigam.com/papers/maxent-ijcaiws99.pdf

Sentiment classification



It is great to be here. The weather is nice.

MH370 is still not found. It is shocking and it must be terrible for those whose families were in that plane.

Tokenization

Woowwwww #India won vs #Pak today!!!! Can't wait 4 da match vs #WI on Mar 23.

Whitespace tokenizing:

Woowwwww
#India
won
vs
#Pak
today!!!!
Can't
wait

4
da
match
vs
#WI
on
Mar
23

Aspects of sentiment aware tokenizing

- Emoticons: very common, particularly in social media
- Twitter style mark-up: usernames, hashtags
- Informative HTML tags
  - absolute mess up
  - , tags
  - 2
- Masked curses: @#$*!*&*
- Punctuations: !!!, !?!????? – likely to be negative
- Capitalization – higher weightage
- Lengthening: "soooooooo much"
- Multi word expression: Named entities, dates, idioms
  - out of this world

Tokenization

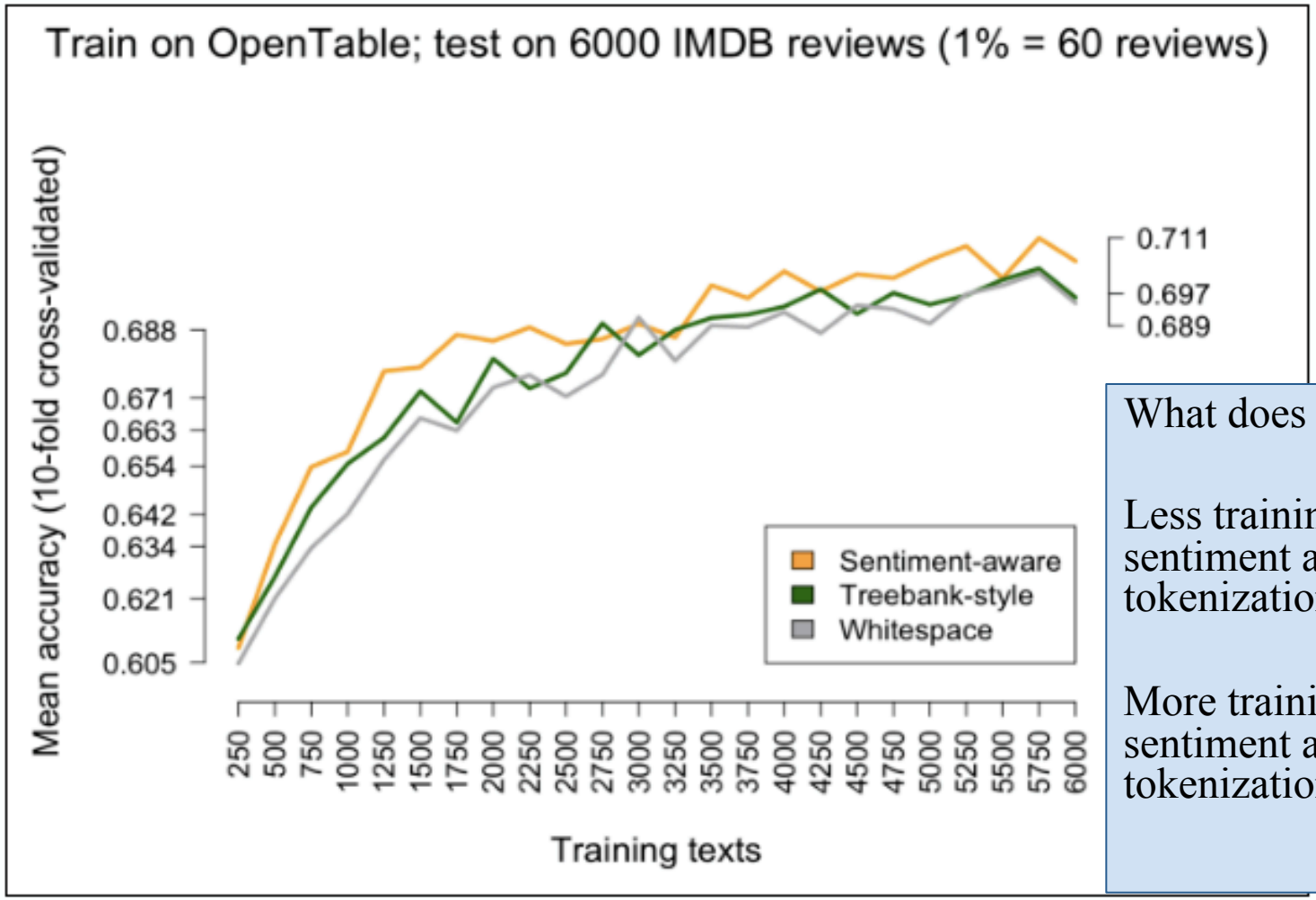
Woowwwww #India won vs #Pak today!!!! Can't wait 4 da match vs #WI on Mar 23.

Semantic-aware tokenizing:

http://sentiment.christopherpotts.net/tokenizing.html

Wow	wait
#India	4
won	da
vs	match
#Pak	vs
today	#WI
!!!!	on
Can't	**Mar_23**

Tokenizing – experiments



What does it mean?

Less training data → sentiment aware tokenization helps

More training data → sentiment aware tokenization matters less

Stemming

Reducing inflected or derived words to its stem (root)

- Stemming, Stemmer → stem
- Objective, Objection → object
- Billing, Billable → bill

Helps a search engine in particular

- *billing error* is essentially same as the query *bill error*

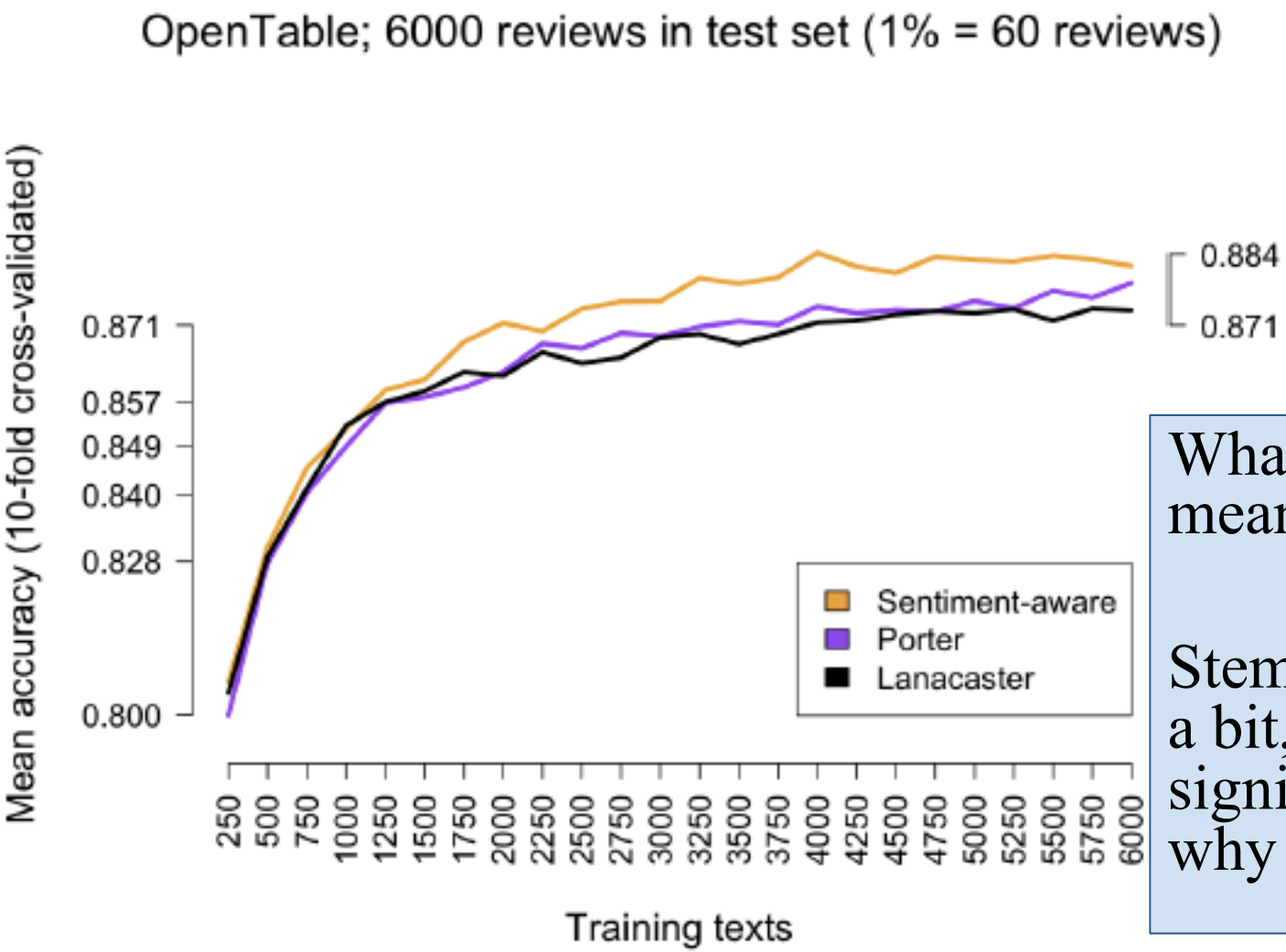
Stemming for sentiment analysis

Porter stemmer (suffix stripping)

Positive sentiment	Negative sentiment	Stem
defense	defensive	defens
affection	affect	affect
objective	objection	object
tolerant	tolerable	toler
extravagance	extravagant	extravag

- Different forms of the same stem may carry different sentiments
- WordNet stemmer does not have this problem, but still it removes comparative morphology, e.g. **happiest, happy** → **happy**

Stemming – experiments



What does it mean?

Stemming hurts a bit, but not significant. Still why even do it?

Negation

- Negation plays a very important role in expressions
  - I don’t like it
  - I never like it
  - I hardly like it
  - No one likes it
  - I am yet to like it
  - I don’t think I will like it

Negation marking

- Append a _NEG suffix to every word appearing between a negation and a clause level punctuation mark.
- Keep track of negative expression.

No one likes it →

no one_NEG likes_NEG it_NEG.

I don't think I will like it →

I don't think_NEG I_NEG will_NEG like_NEG it_NEG .

Part of speech (POS) tagging

- For each word in a text, tag the word by the part of speech

  NN – Noun

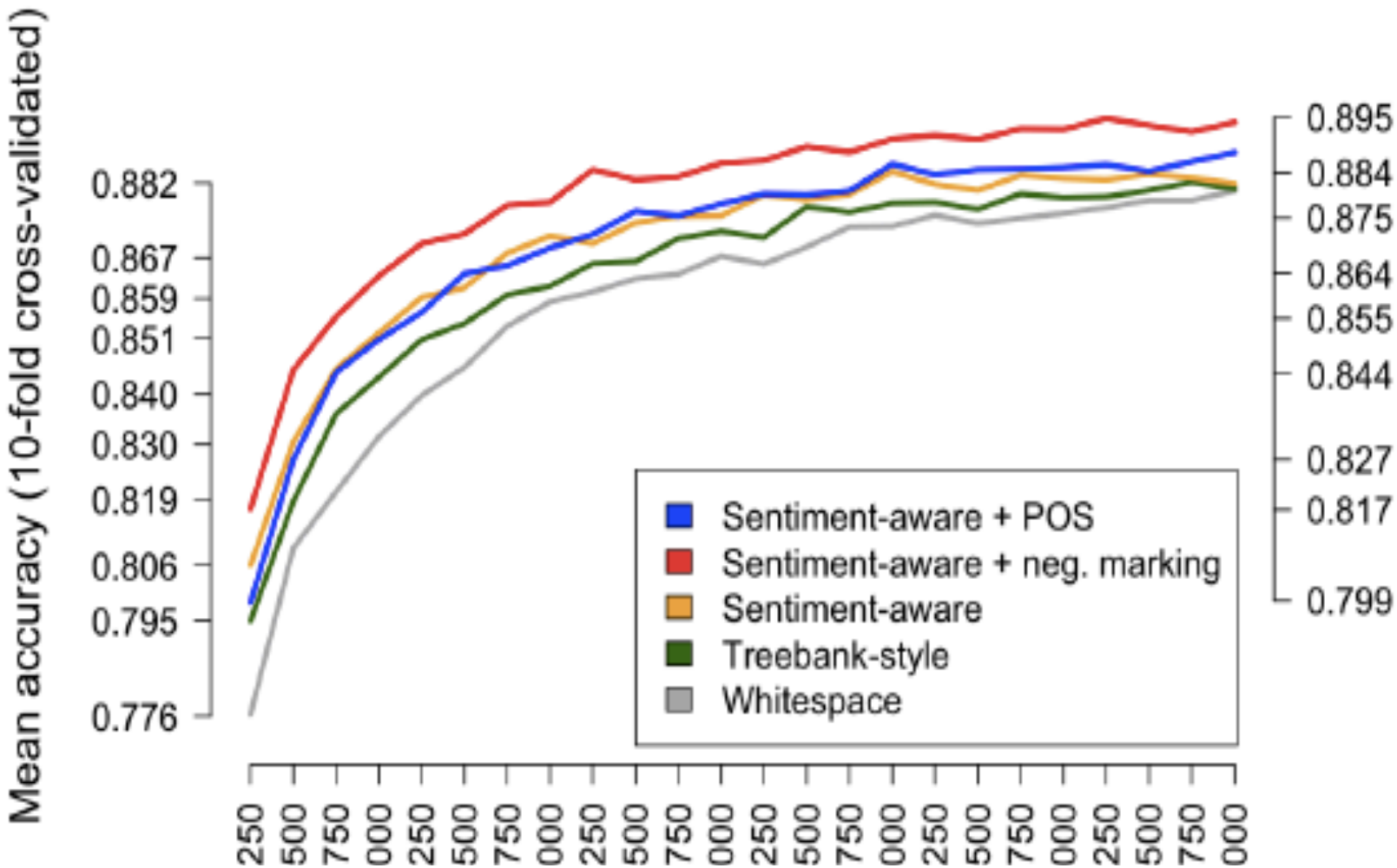
  JJ – Adjective

  RB – Adverb

  VB – Verb

- Part of speech matters for sentiment
  - *That was a **fine** shot by Kohli*
  - *The police took a **fine** from me because I was above 60kmph*
  - fine (jj) → positive, fine (nn or vb) → negative

POS tagging does help



Occurrence vs count

- Often the occurrence of a word matters more than how many times it occurs
  - There may be several criticism about a movie but if there is a sentence "However the movie is wonderful", it means more than several negative words.
- In fact, using the occurrence (boolean: occurs → 1, does not occur → 0) works better in some cases

Sentiment Analysis Method Workflow

1. Tokenize the text, if possible sentiment aware

   [Do not use stemming]

2. Use negation marking
3. Use POS tagging
4. Use a classifier

   [SVM or MaxEnt classifier work better than Naïve Bayes]

   – Train the classifier using the labeled set
   – Use on new data

Some sentiment analysis algorithms

SOME ALGORITHMS

Semi-supervised learning of lexicons

- Manual labeling of lexicons is tough
- Small set of lexicons would not produce a good sentiment analysis
- Can we start with a few manually labeled and let the system learn more?

- Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the Semantic Orientation of Adjectives. ACL, 174–181

Semi supervised learning of lexicons

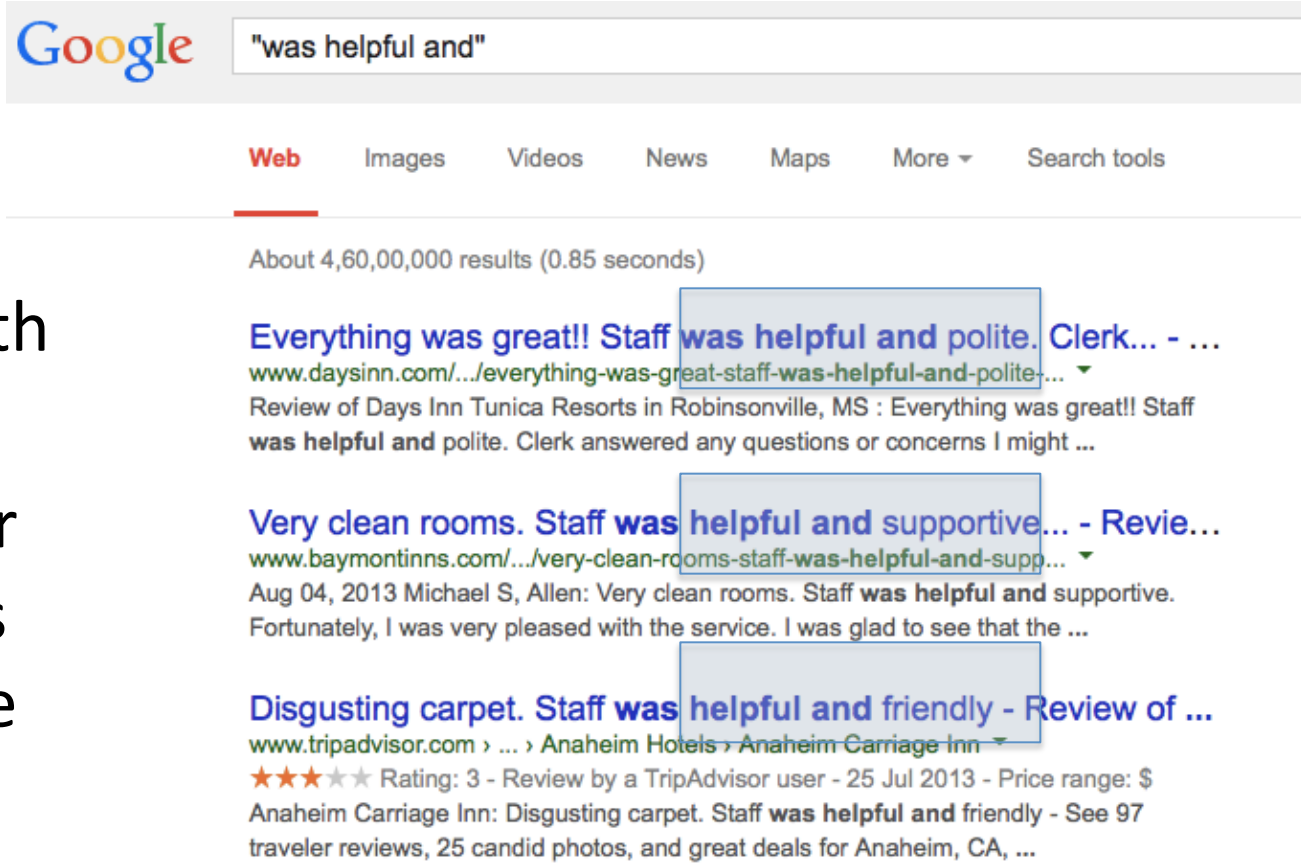
Observe:

- Adjectives conjoined by “and” usually have same polarity
  - helpful and friendly
  - dangerous and brutal
- Adjectives conjoined by “but” usually have opposite polarity
  - friendly but deceptive

Semi supervised learning of lexicons

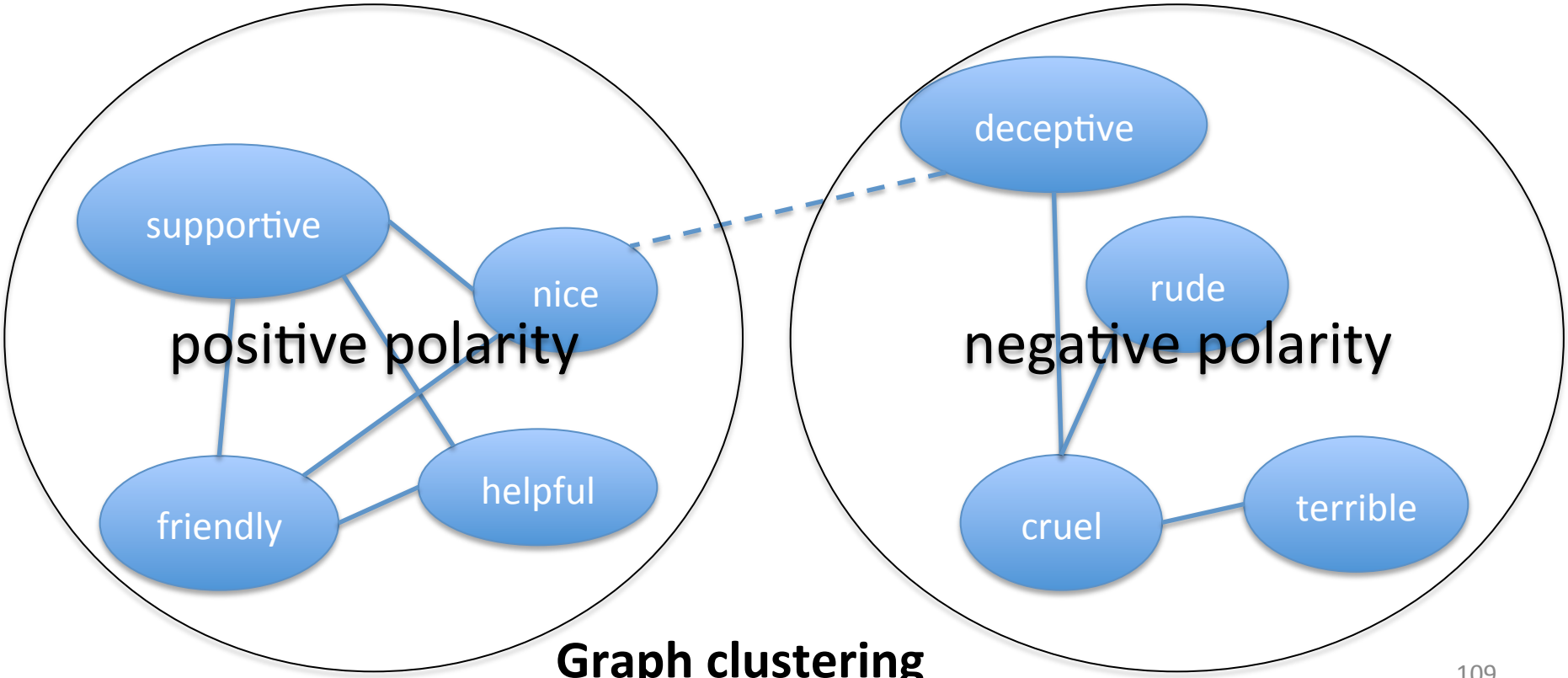
Expand

- Use one lexicon as query, with "and"
- Find other adjectives with same polarity



Semi supervised learning of lexicons

- A graph: similar polarity lexicon pairs have positive weight edges, opposite polarity pairs have negative weight edges



Graph clustering

Semi supervised learning of lexicons

<u>Results</u>

- Positive
  - bold decisive disturbing generous good honest important large mature patient peaceful positive proud sound stimulating straightforward strange talented vigorous witty…
- Negative
  - ambiguous cautious cynical evasive harmful hypocritical inefficient insecure irrational irresponsible minor outspoken pleasant reckless risky selfish tedious unsupported vulnerable wasteful…

What do you see?

Semi supervised learning of lexicons

<u>Results</u>

- Positive
  - bold decisive **disturbing** generous good honest important large mature patient peaceful positive proud sound stimulating straightforward **strange** talented vigorous witty…
- Negative
  - ambiguous **cautious** cynical evasive harmful hypocritical inefficient insecure irrational irresponsible minor **outspoken pleasant** reckless risky selfish tedious unsupported vulnerable wasteful…

Some wrong outputs, but mostly these are fine!

Turney's Algorithm

1. Extract a *phrasal lexicon* from reviews
2. Learn polarity of each phrase
3. Rate a review by the average polarity of its phrases

- Turney (2002): Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews

Extract two-word phrases with adjectives

First Word	Second Word	Third Word (not extracted)
JJ	NN or NNS	anything
RB, RBR, RBS	JJ	Not NN nor NNS
JJ	JJ	Not NN or NNS
NN or NNS	JJ	Nor NN nor NNS
RB, RBR, or RBS	VB, VBD, VBN, VBG	anything

How to measure polarity of a phrase?

- Positive phrases co-occur more with *"excellent"*
- Negative phrases co-occur more with *"poor"*
- But how to measure co-occurrence?

Pointwise Mutual Information

- Pointwise mutual information:
  - How much more do events x and y co-occur *than* if they were independent?

$$PMI(x,y) = \log \frac{P(x,y)}{P(x)P(y)} = \log \frac{P(x \mid y)}{P(x)} = \log \frac{P(y \mid x)}{p(y)}$$

- Estimate PMI between two words

Estimate Pointwise Mutual Information

- Query some search engine (Altavista, Google)
  - *P(word)* estimated by *hits(word)/N*
  - $P(word_1, word_2)$ by $hits(word_1\ NEAR\ word_2)/N$ [denote NEAR by ~]

$$PMI(word_1, word_2) = \log \frac{\frac{1}{N} hits(word_1 \sim word_2)}{\frac{1}{N} hits(word_1) \frac{1}{N} hits(word_2)}$$

Does phrase appear more with “poor” or “excellent”?

$$Polarity(phrase) = PMI(phrase, excellent) - PMI(phrase, poor)$$

$$= \log \frac{\frac{1}{N} hits(phrase \sim excellent)}{\frac{1}{N} hits(phrase) \frac{1}{N} hits(excellent)} - \log \frac{\frac{1}{N} hits(phrase \sim poor)}{\frac{1}{N} hits(phrase) \frac{1}{N} hits(poor)}$$

$$= \log \frac{hits(phrase \sim excellent) \times hits(phrase) \times hits(poor)}{hits(phrase) \times hits(excellent) \times hits(phrase \sim poor)}$$

$$= \log \frac{hits(phrase \sim excellent) \times hits(poor)}{hits(excellent) \times hits(phrase \sim poor)}$$

Phrases from a thumbs-up review

Phrase	POS tags	Polarity
online service	JJ NN	2.8
online experience	JJ NN	2.3
direct deposit	JJ NN	1.3
local branch	JJ NN	0.42
…		
low fees	JJ NNS	0.33
true service	JJ NN	-0.73
other bank	JJ NN	-0.85
inconveniently located	JJ NN	-1.5
Average		0.32

Phrases from a thumbs-down review

Phrase	POS tags	Polarity
direct deposits	JJ NNS	5.8
online web	JJ NN	1.9
very handy	RB JJ	1.4
…		
virtual monopoly	JJ NN	-2.0
lesser evil	RBR JJ	-2.3
other problems	JJ NNS	-2.8
low funds	JJ NNS	-6.8
unethical practices	JJ NNS	-8.5
Average		-1.2

Results of Turney algorithm

- 410 reviews from Epinions
  - 170 (41%) negative
  - 240 (59%) positive
- Majority class baseline: 59%
- Turney algorithm: 74%

- Phrases rather than words
- Learns domain-specific information by itself

But it’s not easy yet

MORE CHALLENGES

Problems with classification

- Assumptions:
  - Each text unit (paragraph, document, sentence) either does not have, or has each sentiment label
  - Usually it has exactly one sentiment label
  - The set of all labels are ranked and are not continuous

Reality of sentiment

- Some text may be partially aligned with some sentiment label
- The expression of emotion in language and human expressions is blended and continuous (Russel 1980, Ekman 1992, Wilson et al 2006)
- A single label often does not do justice to an expression!
- **Project confession!**

Sentiment and context

- Sentiment is target/topic relative
  - I loved the hotel room but the food was terrible
- Sentiment vocabulary is domain or topic dependent
  - "What sets Martin apart is his sheer, brutal, mind-numbing honesty. This is life, in all its pain and glory. … The novel is a masterpiece; beautifully crafted, shockingly realistic and a joy to read."

Data and tools

RESOURCES

The Harvard General Inquirer

- Home page: http://www.wjh.harvard.edu/~inquirer
- List of Categories: http://www.wjh.harvard.edu/~inquirer/homecat.htm
- Spreadsheet: http://www.wjh.harvard.edu/~inquirer/inquirerbasic.xls

- Categories:
  - Positiv (1915 words) and Negativ (2291 words)
  - Strong vs Weak, Active vs Passive, Overstated versus Understated
  - Pleasure, Pain, Virtue, Vice, Motivation, Cognitive Orientation, etc

- Philip J. Stone, Dexter C Dunphy, Marshall S. Smith, Daniel M. Ogilvie. 1966. The General Inquirer: A Computer Approach to Content Analysis. MIT Press
- Free for Research Use

MPQA Subjectivity Lexicon

- Multi-Perspective Question Answering
- Maintained by Wilson, Wiebe, Hoffmann

  http://mpqa.cs.pitt.edu
- GPL License

	Strength	Length	Word	POS	Stemmed	Polarity
1	weaksub	1	abondoned	adj	n	negative
2	strongsub	1	abash	verb	y	negative
…						

Bing Liu Opinion Lexicon

- Bing Liu's Page on Opinion Mining
- http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar
- Minqing Hu and Bing Liu. Mining and Summarizing Customer Reviews. ACM SIGKDD-2004.
- 6786 words
  - 2006 positive
  - 4783 negative

SentiWordNet

- Home page: http://sentiwordnet.isti.cnr.it/
- All WordNet synsets automatically annotated for degrees of positivity, negativity, and neutrality/ objectiveness
- Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010 SENTIWORDNET 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. LREC-2010

Disagreements between polarity lexicons

Christopher Potts, Sentiment Tutorial, 2011

	Opinion Lexicon	General Inquirer	SentiWordNet	LIWC
MPQA	33/5402 **(0.6%)**	49/2867 **(2%)**	1127/4214 **(27%)**	12/363 **(3%)**
Opinion Lexicon		32/2411 **(1%)**	1004/3994 **(25%)**	9/403 **(2%)**
General Inquirer			520/2306 **(23%)**	1/204 **(0.5%)**
SentiWordNet				174/694 **(25%)**
LIWC				

Tools

- Basic sentiment tokenizer

  http://sentiment.christopherpotts.net

- Twitter NLP and POS Tagging

  http://ark.cs.cmu.edu/TweetNLP

- Stanford Core NLP

  http://nlp.stanford.edu/software/corenlp.shtml

Sources and Acknowledgements

- Mining of Massive Datasets: Leskovec, Rajaraman and Ulmann
- Credits to Bing Liu (UIC) and Angshul Majumdar (IIITD) for some slides in the Collaborative Filtering part
- Min-Hashing: Slides by Leskovec, Rajaraman and Ullman from the courses taught in the Stanford University
- Dan Jurafsky's lectures and slides:

  www.youtube.com/watch?v=sxPBv4Skj98

  www.stanford.edu/class/cs124/lec/sentiment.pdf
- Christopher Potts' wonderful website and material:

  http://sentiment.christopherpotts.net
- Survey paper by Pang and Lee: Opinion Mining and Sentiment Analysis, Foundations and Trends in Information Retrieval, Vol. 2, Nos. 1–2 (2008) 1–135
- Stanford NLP Course at Coursera:

  www.coursera.org/course/nlp