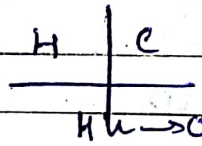
Date / /20

Web- Collection of docs.
Internet - Interconnected Network.
HTTP

WI → AI + Advanced IT.

Turing Test → In, AI, Turing Test is a method to determine whether a computer is Human Being.




Reverse Turing Test is:
Computer asks question eg: CAPTCHA

(1) Intelligent Web Page System
→ BI → CRM (Cust. Rel. Marketing) → Web Marketing / Service / Publishing

(2) Knowledge Network & Management.
→ Electronic Library.
→ Ontology engineering.
→ Semantic Web.

(3) Web Mining.
→ Multimedia datamining.
→ Text Mining.
→ Web based Ontology
→ Web log mining.
→ Web log structure.
→ Web warehousing

Index →
List of words / names / terms

Query → Text Indexing.

1) Documents
2) Tokenizing ~ Breaking Down Sentence.
3) Normalizing
4) Stemming ~ Root Word.
5) Stop Words. ~ Common Word.

| Serial no. | term | Doc ID |
|---|---|---|
| 1 | bag. | 1 → 2 |
| 2 | ball | 3 → 12 |
| 3 | ball | 1 → 12 |

Class Index
create(D) (populates list of documents) ← list of docs.
for d in D
(each document in D)
for w in d.
(each word in every document d)
i = index.lookup(w)
(check whether word w is present in index)
(if w is not present in index, lookup function returns -ve(1))

Date / /20

if i<0

j = index.add(w)

(j will return to which position (or serial no.) the new word w is added to)

index.append(j, d.id);

(add the document id to the dict of doc id

4 cat [2]

else

index.append(d.id)

---

D1:

In the end it doesn't even matter

D2:

It might matter who knows

1) Tokenize.

| Terms | DocID |
|---|---|
| In | 1 |
| the | 1 |
| end | 1 |
| it | 1 |
| doesn't | 1 |
| even | 1 |
| matter | 1 |

| Terms | DocID |
|---|---|
| It | 1 |
| might | 1 |
| matter | 1 |
| who | 1 |
| knows. | 1 |

2) Sort & merge both lists.

| | |
|---|---|
| Doesn't | 1 |
| and | 1 |
| even | 1 |
| In | 1 |
| It | 1 |
| It | 2 |
| Knows | 2 |
| matter | 1 |

Inverted Index.

Now, frequency comes into picture. Posting.

Dictionary.

| Sl. no. | term | Docfq | DocID list |
|---|---|---|---|
| 1 | Doesn't | 1 | 1 |
| 2 | and | 1 | 1 |
| 3 | In | 1 | 1 |
| 4 | It | 2 | 1 → 2 |
| 5 | Knows | 1 | 2 |

Document frequency

Not Term frequency.

'It' twice in doc1

| | |
|---|---|
| It | 1 |

'It' twice in doc1 & 'It' once in doc 2.

| | |
|---|---|
| It | 2 |

# Complexity

Date / /20

'n' documents. 'm' words. 'w' words
(Total docs.) in index. per doc.

(1) complexity of reading every word

$$O(nw)$$

(2) for lookup function, at most 'm' words

$$O(\log m)$$

Assump: Structure used to store Term index is balanced Binary Tree

(3) One more function to append to docID the new document.

considering all of these, the Total Time complexity is : $O(nw \log m)$

I love pets
pets are good

I don't like
pets.

S1:

| term | DocID |
|---|---|
| I | 1 |
| love | 1 |
| pets | 1 |
| pets | 1 |
| are | 1 |
| good | 1 |

| term | DocID |
|---|---|
| I | 2 |
| don't | 2 |
| like | 2 |
| pets. | 2 |

S2:

| Term | ID |
|---|---|
| are | 1 |
| don't | 2 |
| good | 1 |
| I | 1 |
| I | 2 |
| like | 2 |
| love | 1 |
| Pets | 1 |
| pets | 1 |
| pets | 2 |

S3:

| Sl. no. | term | DocID |
|---|---|---|
| 1 | are \| 1 | 1 |
| 2 | don't \| 1 | 2 |
| 3 | good \| 1 | 1 |
| 4 | I \| 2 | 1 → 2 |
| 5 | like \| 1 | 2 |
| 6 | love \| 1 | 1 |
| 7 | pets \| 2 | 1 → 2 |

Dictionary

posting

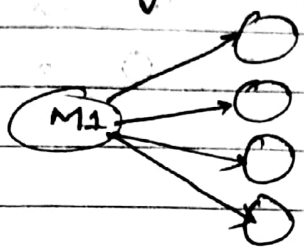
Date / /20

Web Intelligence :

Page Rank, Hyperlinks, Link Analysis.

(Q) How to tell if a mail account is a spam. account, without looking into contents.

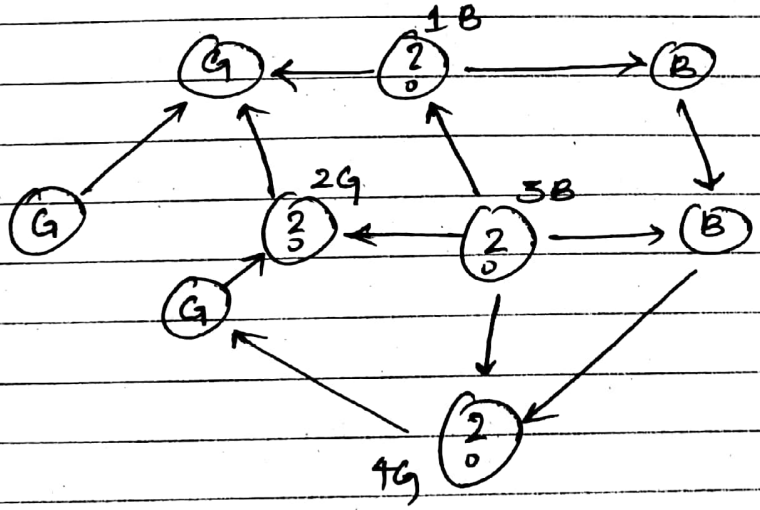
(A) No In-Link.




(1) No Good Node points to a bad node.

(2) If a node is pointing to a bad Node, then, it is bad.

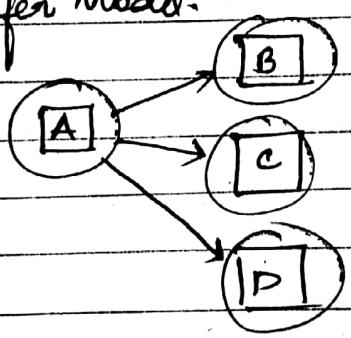
(3) If good nodes points to a Node, then, it is good.




# Page Rank

Random Surfer Model.

Page Rank:




Prob = 1/n

n → No. of outlinks

Once you reach a node that has no outlink then, we use teleport operation

(1) Navigate to a different page

(2) Go Back

Teleport used when:

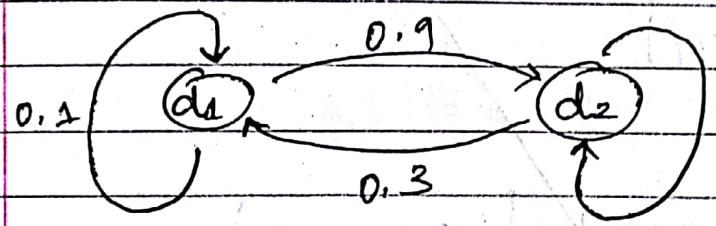
(1) Node has no outlink

(2) $0 < \alpha < 1$.

Date / /20

# To convert Adjacency Matrix to TPM

(1) Divide each '1' in A by total no. of 1's in the row.
(2) Multiply resulting matrix by $1-\alpha$.
(3) Add $\alpha/N$ to every entry to the resulting matrix, to obtain




$x_1$ $x_2$

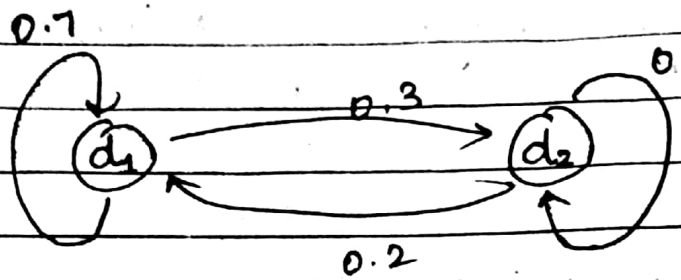
$P_t(d_1)$ $P_t(d_2)$.

$$[0.1] \begin{bmatrix} 0.1 & 0.9 \\ 0.3 & 0.7 \end{bmatrix}$$

$P_{11} = 0.1 \quad P_{12} = 0.9$
$P_{21} = 0.3 \quad P_{22} = 0.7$

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 0.3 | 0.7 | $\vec{x}P$ |
| 0.3 | 0.7 | 0.24 | 0.76 | $\vec{x}P^2$ |
| 0.24 | 0.76 | 0.252 | 0.748 | $\vec{x}P^3$ |
| 0.252 | 0.748 | 0.2496 | 0.7504 | $\vec{x}P^4$ |
| ... | | ... | | |
| 0.25 | 0.75 | 0.25 | 0.75 | $xP^5$ |

# Power Method.




I $\quad [1 \;\; 0] \begin{matrix} d1 \\ d2 \end{matrix} \begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$ $\quad$ Ans = $[0.4 \;\; 0.6]$

II $= [0.3 \quad 0.7] \begin{bmatrix} 0.3 & 0.7 \\ 0.2 & 0.8 \end{bmatrix}$

$= 0.09 + 0.14 \quad 0.21 + 0.56$

$= [0.23 \;\; 0.77]$

$[0.23 \;\; 0.77] \begin{bmatrix} 0.3 & 0.7 \\ 0.2 & 0.8 \end{bmatrix}$

III $[0.223 \;\; 0.777] \begin{bmatrix} 0.3 & 0.7 \\ 0.2 & 0.8 \end{bmatrix}$

$0.2223 \quad 0.7777$

$[0.22^2 \quad 0.77]$

$[1 \;\; 0] \begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$ $\quad$ Ans = $[0.4 \;\; 0.6]$

PageRank Vector = $\vec{\pi} = (\pi_1, \pi_2)$

$= (0.4, 0.6)$

Date / /20

$$\pi = \pi \begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$$

$$\pi = [x \quad y]$$

$$[x \quad y] = [x \quad y] \begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$$

$$[x \quad y] = [0.7x + 0.2y \quad 0.3x + 0.8y$$

$$x = 0.7x + 0.2y$$

$$x + y = 1$$

(a) $0.3x = 0.2y$

$$y = \frac{3}{2}x$$

$$x + \frac{3}{2}x = 1$$

$$\frac{5x}{2} = 1$$

$$x = \frac{2}{5} = 0.4$$

(1) $\alpha = 0.4$ consider Graph

$1 \to 2$ $2 \to 1$ $2 \to 3$ $3 \to 2$

$4 \to 3$ $4 \to 2$ $4 \to 1$ $1 \to 3$

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 |
| 2 | 1 | 0 | 1 | 0 |
| 3 | 0 | 1 | 0 | 0 |
| 4 | 1 | 1 | 1 | 0 |

Step1: Divide by total no. of 1's in row

$$\begin{bmatrix} 0 & 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/1 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 \end{bmatrix}$$

$\alpha = 0.4 \Rightarrow 1 - \alpha = 0.6$.

Step(2) Multiply the resulting matrix by $1-\alpha$

$$[0.6] \begin{bmatrix} 0 & 0.5 & 0.5 & 0 \\ 0.5 & 0 & 0.5 & 0 \\ 0 & 1 & 0 & 0 \\ 0.33 & 0.33 & 033 & 0 \end{bmatrix}$$

Date / /20

$$\begin{bmatrix} 0 & 0.3 & 0.3 & 0 \\ 0.3 & 0 & 0.3 & 0 \\ 0 & 0.6 & 0 & 0 \\ 0.198 & 0.198 & 0.198 & 0 \end{bmatrix}$$

Step (3): Add $\alpha/n$ to everything:

$\alpha = 0.4 \quad N = 4$

N → Total no. of nodes.

$$\begin{bmatrix} 0.1 & 0.4 & 0.4 & 0.1 \\ 0.4 & 0.1 & 0.4 & 0.1 \\ 0.1 & 0.7 & 0.1 & 0.1 \\ 0.3 & 0.3 & 0.3 & 0.1 \end{bmatrix}$$

↕ TPM from Adjacency Matrix

Now, to get PageRank

$$\begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0.1 & 0.4 & 0.4 & 0.1 \\ 0.4 & 0.1 & 0.4 & 0.1 \\ 0.1 & 0.7 & 0.1 & 0.1 \\ 0.3 & 0.3 & 0.3 & 0.1 \end{bmatrix}$$

Page Rank → human memory

Word - word Association

Eg. when 'Apple' is given people think 'Orange' as the next word
Experiment were done & made a symantic network which works better than web for searching

Letter - word Association

'a' → apple
→ aeroplane

Algo. → ends up relearning a bunch of words of words

google indexing & human memory is not same
human memory is poor in recalling facts but google does it easily
Context - human needs context to recall
fuzzy - each & every detail not possible
sparse - a small hint can also trigger to recall a whole incident

LSH → Locality Sensitive Hashing

Shingling → divide doc. into sets where no element is repeated

Given 2 n/w assignments, how to find if they have been plagiarized.
doc. similarity

#PageRank

more → human memory.

"Google & the mind"

Psychological.

(1) Word-Word Association

formed semantic network

web-graphs => hyperlink

(2) Algorithm:

Ends up retrieving a bunch of words of words

Search works better for a semantic network as compared to a web graph

(Q) Is human memory same as Google Indexing

(A) No.

- Human mind is poor at recalling facts.
- " " needs context
- fuzzy
- sparse

Searching: Structured Data: complex in nature.

LSH: Locality Sensitive Hashing

(Q) Given 2 network assignments, how to find if they have been plagiarized.

(A) Document Similarity.

Date / /20

Shingling : A common technique of representing sets.

LSH → Locality Sensitive Hashing.

General idea of LSH →
Hashing items such that similar items fall into the same bin.

Starting point = Similar docs → Similar sets

hard part → Arranging similar items such that they fall into the same bin

Starting point = Similar docs → Similar sets.

# Applications of set similarities :

(1) Grouping of docs, pages with similar content
→ topic

(2) Netflix → Recommending.

(3) Movie fans.

(4) Entropy Matching.

# Applications of Document Similarity:

(1) Mirror Sites

(2) Plagiarism.

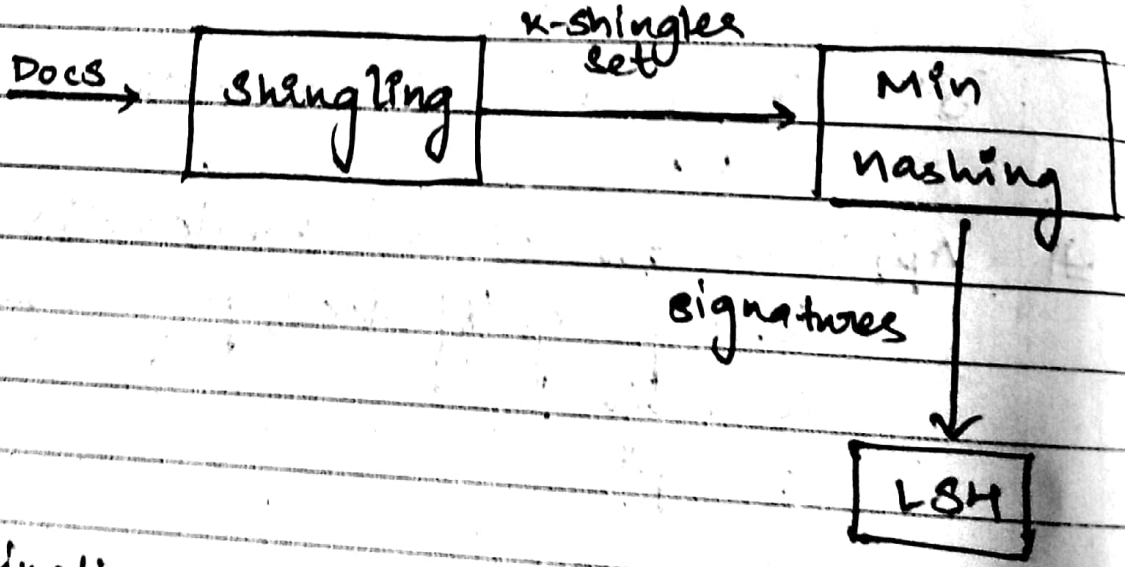
(3) e-news.

# Calculate Similarity

1) Shingling.

(2) Min-Hashing.

(3) LSH.




Shingling: set of k-sensitive words

2K abcab → {ab, bc, ca, ~~cb~~}

Set

Date / /20

Shingles & Doc Similarity

→ most of the shingles.

→ 2k shingles (inter-changing paragraphs)

Min Hashing:

→ Jaccard similarity

→ Signature Matrix

*

$JC \rightarrow sim(C_1, C_2) = |C_1 \cap C_2| / |C_1 \cup C_2|$

Rows → universal elements:

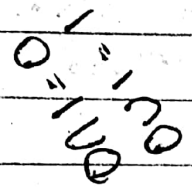
Cols → sets

$JC = A/(A+B+C)$

| C1 | C2 | |
|---|---|---|
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | * |
| 0 | 0 | |
| 1 | 1 | * |
| 0 | 1 | |

2/5

| | C1 | C2 |
|---|---|---|
| A | 1 | 1 |
| B | 1 | 0 |
| C | 0 | 1 |
| D | 0 | 0 |



Maturity level of an organization refers to the ability of an org for continuous improvement in a particular

# Min Hashing :

| Element | $S_1$ | $S_2$ | $S_3$ | $S_4$ |
|---|---|---|---|---|
| b | 0 | 0 | 1 | 0 |
| e | 0 | 0 | 1 | 0 |
| a | 1 | 0 | 0 | 1 |
| d | 1 | 0 | 1 | 1 |
| c | 0 | 1 | 0 | 1 |

$h(S_1) = a$

$h(S_2) = e$

$h(S_3) = b$

$h(S_4) = a$

| Row | $c_1$ | $c_2$ |
|---|---|---|
| 1 | 1 | 0 |
| 2 | 0 | 1 |
| 3 | 1 | 1 |
| 4 | 1 | 0 |
| 5 | 0 | 1 |

$h(x) = x \bmod 5$

$g(x) = 2x + 1) \bmod 5$

Minhash Signature Matrix

JC = 1/5

Date / /20

| | Sig 1 | Sig 2 |
|---|---|---|
| h(x) | ∞ | ∞ |
| g(x) | ∞ | ∞ |

h(1) = 1
g(1) = 3

| | Sig 1 | Sig 2 |
|---|---|---|
| h(1)=1 | 1 | ∞ |
| g(1)=3 | 3 | ∞ |
| h(2)=2 | 1 | 2 |
| g(2)=0 | 3 | 0 |
| h(3)=3 | 1 | 2 |
| g(3)=2 | 2 | 0 |
| h(4)=4 | 1 | 2 |
| g(4)=4 | 2 | 0 |
| h(5)=0 | 1 | 0 |
| g(5)=1 | 2 | 0 |

| Raw | $S_1$ | $S_2$ | $S_3$ | $S_4$ | |
|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | $h_1(x) = (x+1) \bmod 5$ |
| 1 | 0 | 0 | 1 | 0 | $h_2(x) = (3x+1) \bmod 5$ |
| 2 | 0 | 1 | 0 | 1 | |
| 3 | 1 | 0 | 1 | 1 | |
| 4 | 0 | 0 | 1 | 0 | |

| $S_1$ | $S_2$ | $S_3$ | $S_4$ | ~~h(x)~~ |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | $h_1(0) = 1$ |
| ∞ | ∞ | ∞ | ∞ | $h_2(0) = 1$ |

$h_1(0)=1$ 1 ∞ ∞ 1

$h_2(0)=2$ 1 ∞ ∞ 1

$h_1(1)=2$ 1 ∞ 2 1

$h_2(1)=4$ 1 ∞ 4 1

$h_1(2)=3$ 1 3 2 1

$h_2(2)=2$ 1 2 4 1

$h_1(3)=4$ 1 3 2 1

$h_2(3)=0$ 0 2 0 0

$h_2(4)=0$ 1 3 0 1

$h_2(4)=3$ 0 2 0 0

$Sim(S_1, S_4) =$

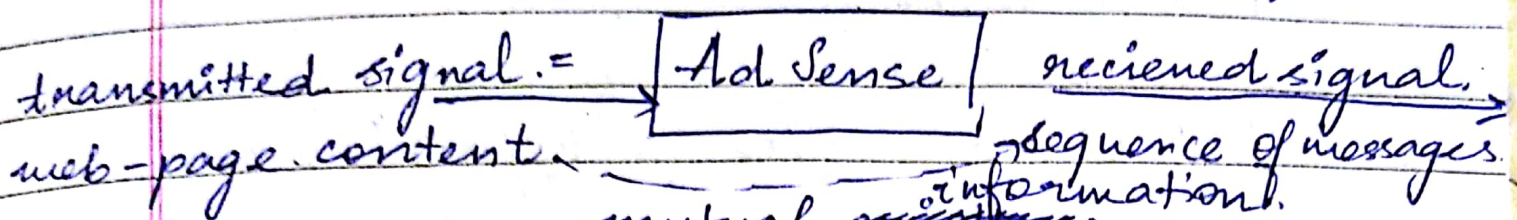
Date / /20

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | $h_1(x) = (2x+1) \bmod 6$ |
| 1 | 0 | 1 | 0 | 0 | $h_2(x) = (3x+2) \bmod 6$ |
| 2 | 1 | 0 | 0 | 1 | $h_3(x) = (3x+2) \bmod 6$ |
| 3 | 0 | 0 | 1 | 0 | |
| 4 | 0 | 0 | 1 | 1 | |
| 5 | 1 | 0 | 0 | 0 | |

| | $S_1$ | $S_2$ | $S_3$ | $S_4$ |
|---|---|---|---|---|
| $h_1(0) = 1$ | ∞ | 1 | ∞ | 1 |
| $h_2(0) = 2$ | ∞ | 2 | ∞ | 2 |
| $h_3(0) = 2$ | ∞ | 2 | ∞ | 2 |
| $h_1(1) = 3$ | ∞ | 1 | ∞ | 1 |
| $h_2(1) = 5$ | ∞ | 2 | ∞ | 2 |
| $h_3(1) = 1$ | ∞ | 1 | ∞ | 2 |
| $h_1(2) = 5$ | 5 | 1 | ∞ | 1 |
| $h_2(2) = 2$ | 2 | 2 | ∞ | 2 |
| $h_3(2) = 0$ | 0 | 1 | ∞ | 0 |
| $h_1(3) = 1$ | 5 | 1 | 1 | 1 |
| $h_2(3) = 5$ | 2 | 2 | 5 | 2 |
| $h_3(3) = 5$ | 0 | 1 | 5 | 0 |
| $h_1(4) = 3$ | 5 | 1 | 1 | 1 |
| $h_2(4) = 2$ | 2 | 2 | 2 | 2 |
| $h_3(4) = 4$ | 0 | 1 | 4 | 0 |
| $h_1(5) = 5$ | 5 | 1 | 1 | 1 |
| $h_2(5) = 5$ | 2 | 2 | 2 | 2 |
| $h_3(5) = 3$ | 0 | 1 | 4 | 0 |

$2/3 = 66\%$.
kind of similar

# Adsense, Keywords & mutual information




Advertising Messages

# Inverse Search.

pages to keywords.

query words to pages.

Tf-Idf.

- How important a word is
- Tf-idf increases proportionality to the number of times.
- Rarer words make better keywords

2 ways to search (or show ads)

(1) Search based on keyword

(2) & Inverse Search

(Based on imp word in website)

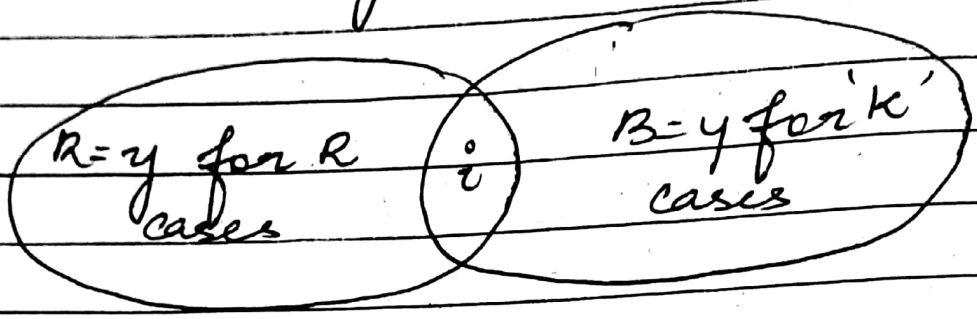
$P(B|R) = i/r$

$P(R) = r/n$

$P(R \text{ and } B) = i/n = (i/r) * (r/n)$

$P(B,R) = P(B|R) \cdot P(R)$

This is Baye's rule.




$P(B,R) = P(B|R) \cdot P(R)$

$= P(R|B) \cdot P(B)$

→

Types of prob:

- → Conditional. $P(x_1|x_2)$
- → Posterior $P(x|x_1)$
- → Apriori $P(x_1)\ P(x_2)\ P(x)$

Independence:

$i/c = r/n \equiv q/r = c/n$

$P(R) = r/n \quad P(C) = c/n$

$P(R|C) = i/c \quad P(C|R) = i/r$

R & C are independent

Date / /20

(Q) Machine A & B produce produce 10% & 90% effectively. of the prod of component. intended for motor industry. From the experience, it is known that prob. that A produces a defective comp is 0.01
B -> defective component -> 0.05
Pick random comp
find the prob. that it was made by A or by B.

Prob (A) = (produced by A)

$P(A) = 0.1$ $P(D|A) = 0.01$
$P(B) = 0.9$ $P(D|B) = 0.05$

$$P(A|D) = \frac{P(D|A).P(A)}{P(D|A).P(A) + P(D|B)P(B)}$$

$= 0.02.$

# Language & Information

transmitted signal. → meaning | language | recieved signal
spoken / written word

Language is redundant.
Language tries to maintain "uniform information density"

- Besides Tf-idf.
  - keyword is present in how many docs.
  - semantics of a word.
  - bipartite graph.

- Machine learning.

Prediction using conditional probability

# Sentiment Analysis:

Lib

Date / /20

limiting physical.
Internal control.

Mutual information:

double summation.

$$I(F,B) = \sum_{f,b} p(f,b) \log \frac{p(f,b)}{p(f)\,p(b)}$$

Limitations of ML:

# ASSIGNMENT - 1

$P(A) = .1$ , $P(B) = .9$

$P(D/A) = .01$ , $P(D/B) = .05$

$$P(A/D) = \frac{P(D/A) \cdot P(A)}{P(D/A) \cdot P(A) + P(D/B) \cdot P(B)} = \frac{.01 \times .1}{(.01 \times .1) + (.05 \times .9)} = \frac{.001}{.046} =$$

$$P(B/D) = \frac{P(D/B) \cdot P(B)}{P(D/B) \cdot P(B) + P(D/A) \cdot P(A)} = \frac{.05 \times .9}{.046} = .978$$

$h1(x) = (2x + 4) \% 5$

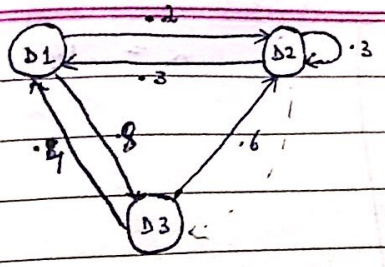
$h2(x) = (3x - 1) \% 5$

| Row | $S_1$ | $S_2$ | $S_3$ | $S_4$ |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 1 |
| 3 | 1 | 0 | 1 | 1 |
| 4 | 0 | 0 | 1 | 0 |

## Minhash SM

| | sig1 | sig2 | sig3 | sig4 |
|---|---|---|---|---|
| $h1(x)$ | ∞ | ∞ | ∞ | ∞ |
| $h2(x)$ | ∞ | ∞ | ∞ | ∞ |
| $h1(0)$ | 4 | ∞ | ∞ | 4 |
| $h2(0)$ | 4 | ∞ | ∞ | 4 |
| $h1(1)$ | 4 | ∞ | 1 | 4 |
| $h2(1)$ | 4 | ∞ | 2 | 4 |
| $h1(2)$ | 4 | 3 | 1 | 3 |
| $h2(2)$ | 4 | 0 | 2 | 0 |
| $h1(3)$ | 0 | 3 | 0 | 0 |
| $h2(3)$ | 3 | 0 | 2 | 0 |
| $h1(4)$ | 0 | 3 | 0 | 0 |
| $h2(4)$ | 3 | 0 | 1 | 0 |

| | | | |
|---|---|---|---|
| 0 | 3 | 0 | 0 |
| 3 | 0 | 1 | 0 |




$$P = \begin{bmatrix} 0 & .2 & .8 \\ .3 & .3 & 0 \\ .4 & .6 & 0 \end{bmatrix}$$

$$t = [\,0 \quad 0 \quad 1\,]$$

Power Method

| X(D1) | X(D2) | X(D3) | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | .4 | .6 | 0 | → XP |
| .4 | .6 | 0 | .18 | .26 | .32 | → XP² |
| .18 | .26 | .32 | .206 | .306 | .144 | → XP³ |
| .206 | .306 | .144 | .1494 | .2194 | .1648 | → XP⁴ |

$$P = \begin{bmatrix} 0 & .2 & .8 \\ .3 & .3 & .4 \\ .4 & .6 & 0 \end{bmatrix}$$

Power Method

| X(D1) | X(D2) | X(D3) | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | .4 | .6 | 0 | → XP |
| .4 | .6 | 0 | .18 | .26 | .56 | → XP² |
| .18 | .26 | .56 | .302 | .45 | .248 | → XP³ |
| .302 | .45 | .248 | .2342 | .3442 | .4216 | → XP⁴ |
| .2342 | .3442 | .4216 | .2719 | .403 | .325 | → XP⁵ |
| .2719 | .403 | .325 | .2509 | .3702 | .3787 | → XP⁶ |

$\alpha = .14$

| | $d_0$ | $d_1$ | $d_2$ | $d_3$ | $d_4$ | $d_5$ | $d_6$ |
|---|---|---|---|---|---|---|---|
| $d_0$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $d_1$ | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| $d_2$ | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| $d_3$ | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| $d_4$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $d_5$ | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $d_6$ | 0 | 0 | 0 | 1 | 1 | 0 | 1 |

divide by no. 1's in each row

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | 0 | 0 | 0 |
| $\frac{1}{2}$ | 0 | $\frac{1}{2}$ | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | $\frac{1}{2}$ | $\frac{1}{2}$ |
| 0 | 0 | 0 | $\frac{1}{3}$ | $\frac{1}{3}$ | 0 | $\frac{1}{3}$ |

multiply by $(1-\alpha)$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | .86 | 0 | 0 | 0 | 0 |
| 0 | .43 | .43 | 0 | 0 | 0 | 0 |
| .43 | 0 | .43 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | .43 | .43 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | .86 |
| 0 | 0 | 0 | 0 | 0 | .43 | .43 |
| 0 | 0 | .287 | .287 | 0 | .287 | |

add $\alpha/N$

| | | | | | | |
|---|---|---|---|---|---|---|
| .02 | .02 | .88 | .02 | .02 | .02 | .02 |
| .02 | .45 | .45 | .02 | .02 | .02 | .02 |
| .45 | .02 | .45 | .02 | .02 | .02 | .02 |
| .02 | .02 | .02 | .45 | .45 | .02 | .02 |
| .02 | .02 | .02 | .02 | .02 | .02 | .88 |
| .02 | .02 | .02 | .02 | .02 | .45 | .45 |

5) Doc1 → I did not
Doc 2 → Et tu
Term Doc #
I 1
did 1
not 1
Julius
Caesar
I
was
killed
I'
the
capitol
brutus
killed
me
do
not
let
it
be
with
Caesar
the
noble
brutus
hath
told
you
Caesar
was
ambi
Inverted

Doc 1 → I did enact Julius Caesar : I was killed i' the capitol, Brutus killed me.

Doc 2 → So let it be with Caesar. The noble brutus hath told you Caesar was ambitious

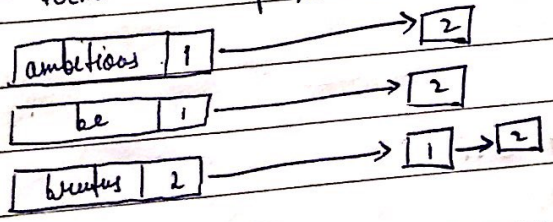
| Term | Doc # |
|---|---|
| I | 1 |
| did | 1 |
| enact | 1 |
| Julius | 1 |
| caesar | 1 |
| I | 1 |
| was | 1 |
| killed | 1 |
| i' | 1 |
| the | 1 |
| capitol | 1 |
| brutus | 1 |
| killed | 1 |
| me | 1 |
| so | 2 |
| let | 2 |
| it | 2 |
| be | 2 |
| with | 2 |
| caesar | 2 |
| the | 2 |
| noble | 2 |
| brutus | 2 |
| hath | 2 |
| told | 2 |
| you | 2 |
| caesar | 2 |
| was | 2 |
| ambitious | 2 |

→

| Term | Doc # |
|---|---|
| ambitious | 2 |
| be | 2 |
| brutus | 1 |
| brutus | 2 |
| capitol | 1 |
| caesar | 1 |
| caesar | 2 |
| caesar | 2 |
| did | 1 |
| enact | 1 |
| hath | 2 |
| I | 1 |
| I | 1 |
| i' | 1 |
| it | 2 |
| julius | 1 |
| killed | 1 |
| killed | 1 |
| let | 2 |
| me | 1 |
| noble | 2 |
| so | 2 |
| the | 1 |
| the | 2 |
| told | 2 |
| you | 2 |
| was | 1 |
| was | 2 |
| with | 2 |

## Inverted Index :-

term  doc. freq.  postings list



Notes.

Shingles:

→ Define a k shingle for a doc to be any substring of length k found in the doc.

→ Then we associate with each doc the set of k shingles that appear one or more times in that doc.

How to pick k (Shingle size).

→ k should be picked large enough so that probability of any given shingle appearing in any given doc is low.

→ For large docs, such as research articles k=9 is considered safe.

Hashing Shingles: We can map each of the 9-shingles and hash them down to 4 bytes.

Shingles Built from Words. Using say, a stop word followed by any two words as a shingle.

Ex 3.3.2 Minhash Signatures.

| Ele | $S_1$ | $S_2$ | $S_3$ | $S_4$ |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 | 1 |
| 3 | 0 | 0 | 1 | 0 |
| 4 | 0 | 0 | 1 | 1 |
| 5 | 1 | 0 | 0 | 0 |

| | | | | |
|---|---|---|---|---|
| $h_1(2)$ | 3 | 2 | $\infty$ | 3 |
| $h_2(2)$ | 0 | 2 | $\infty$ | 0 |
| $h_1(3)$ | 3 | 2 | 0 | 3 |
| $h_2(3)$ | 0 | 2 | 3 | 0 |
| $h_1(4)$ | 3 | 2 | 0 | 2 |
| $h_2(4)$ | 0 | 2 | 1 | 0 |
| $h_1(5)$ | 3 | 2 | 0 | 2 |
| $h_2(5)$ | 0 | 2 | 1 | 0 |

∴ Signature matrix - $\begin{bmatrix} 3 & 2 & 0 & 2 \\ 0 & 2 & 1 & 0 \end{bmatrix}$

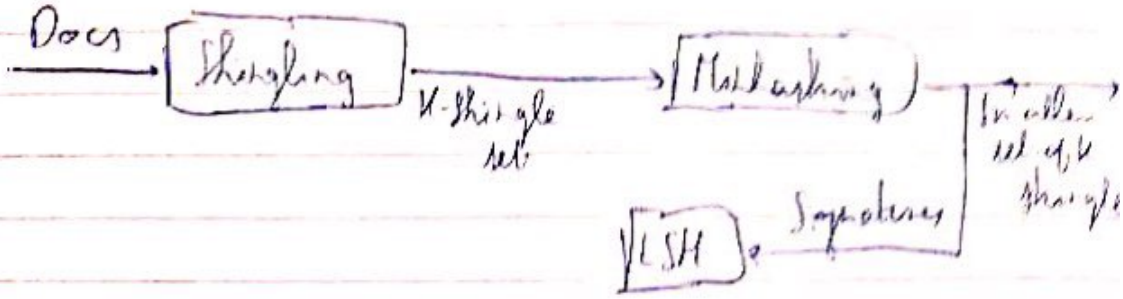
Locality Sensitive Hashing (LSH): Instead of finding similarity of every pair (tedious), often we want to find only the most similar pairs or all pairs that are above some lower bound in similarity. For this we use LSH.

Approach: "Hash" items several times, so that similar items are more likely to be hashed to the same bucket than dissimilar items are.

→ Then, pairs within the same bucket are compared (candidate pairs). Only these are checked for similarity

Note. Dissimilar pairs that hash to the same bucket are called

False Positives




~~Check~~

Check 3.4.3 from the slides.

Associative memory: A type of computer memory from which items may be retrieved by matching some part of their content, rather than by specifying their address.
→ AM is much slower than RAM.
→ Used in multilevel memory systems.
→ To retrieve a word from AM, a search key (or descriptor) is compared with tags or tag bits of all stored words, and corresponding matches are found.
→ AM is expensive to implement.

Sparse distributed memory: Generalized RAMs for long binary words
→ Main attribute is that it is sensitive to similarity: A word can be read back by not only giving the address but also by giving one close to it, as measured by Hamming distance.

**Jaccard Similarity**:

We shall focus initially on a particular notion of “similarity”: the similarity of sets by looking at the relative size of their intersection. This notion of similarity is called “Jaccard similarity”.

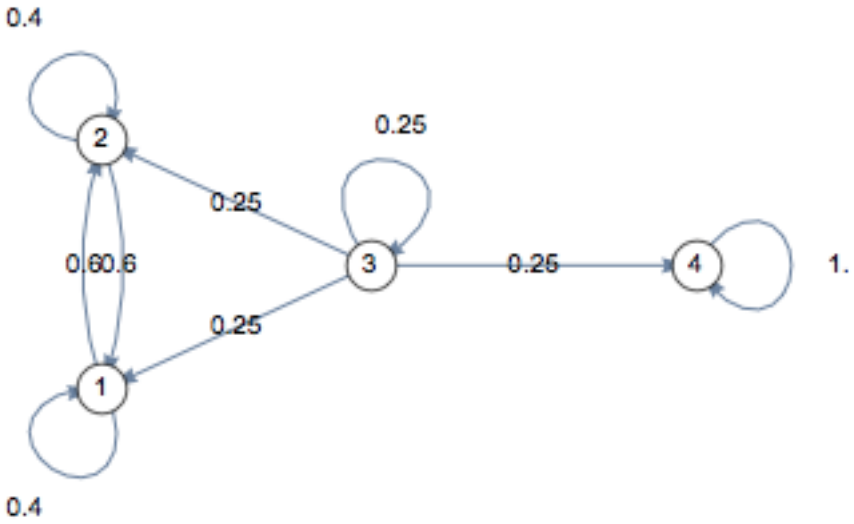
**Formula**: $|S \cap T|/|S \cup T|$

**Markov Chain**:

A Markov chain is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules. The defining characteristic of a Markov chain is that no matter how the process arrived at its present state, the possible future states are fixed. In other words, the probability of transitioning to any particular state is dependent solely on the current state and time elapsed. The state space, or set of all possible states, can be anything: letters, numbers, weather conditions, baseball scores, or stock performances.

Markov chains may be modelled by finite state machines, and random walks provide a prolific example of their usefulness in mathematics. They arise broadly in statistical and information-theoretical contexts and are widely employed in economics, game theory, queueing (communication) theory, genetics, and finance. While it is possible to discuss Markov chains with any size of state space, the initial theory and most applications are focused on cases with a finite (or countably infinite) number of states.

Markov chain Monte Carlo (MCMC) is a technique for estimating by simulation the expectation of a statistic in a complex model. Successive random selections form a Markov chain, the stationary distribution of which is the target distribution. It is particularly useful for the evaluation of posterior distributions in complex Bayesian models.




**Semantic Graphs:**

A **semantic graph** is a network of heterogeneous nodes and links annotated with a domain ontology. In information analysis, **semantic graphs** are generated and applied in a visual analysis approach known as link analysis.
Example: RDF.

**Conditional Probability**:
P(a/y) = (P(y/a)*P(a))/P(y).

**Shannon Coding:**

In the field of data compression, **Shannon coding**, named after its creator is a lossless data compression technique for constructing a prefix code based on a set of symbols and their probabilities (estimated or measured). It is suboptimal in the sense that it does not achieve the lowest possible expected code word length like Huffman coding does, and never better but sometime equal to the Shannon-Fano coding.

Easy Video of 3 minutes: https://www.youtube.com/watch?v=dJCck1OgsIA