CMU SCS

15-826: Multimedia Databases and Data Mining

SVD - part II (case studies)

C. Faloutsos

CMU SCS

Outline

Goal: 'Find similar / interesting things'

- Intro to DB
- Indexing - similarity search
- Data Mining

15-826 Copyright: C. Faloutsos (2005) 2

CMU SCS

Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
- spatial access methods
- fractals
- text
- Singular Value Decomposition (SVD)
- multimedia
- ...

15-826 Copyright: C. Faloutsos (2005) 3

CMU SCS

SVD - Detailed outline

- Motivation
- Definition - properties
- Interpretation
- Complexity
- Case studies
- SVD properties
- Conclusions

15-826 Copyright: C. Faloutsos (2005) 4

CMU SCS

SVD - Case studies

- multi-lingual IR; LSI queries
- compression
- PCA - 'ratio rules'
- Karhunen-Lowe transform
- query feedbacks
- google/Kleinberg algorithms

15-826 Copyright: C. Faloutsos (2005) 5

CMU SCS

Case study - LSI

Q1: How to do queries with LSI?

Q2: multi-lingual IR (english query, on spanish text?)

15-826 Copyright: C. Faloutsos (2005) 6

Case study - LSI

Q1: How to do queries with LSI?

Problem: Eg., find documents with ‘data’

$$
\begin{array}{c|ccccc}
 & \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \\
\text{CS} & & & & & \\
\text{MD} & & & & &
\end{array}
\quad
\left[\begin{array}{ccccc}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{array}\right]
=
\left[\begin{array}{cc}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{array}\right]
\text{x}
\left[\begin{array}{cc}
9.64 & 0 \\
0 & 5.29
\end{array}\right]
\text{x}
\left[\begin{array}{ccccc}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{array}\right]
$$

Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into ‘concept space’ – how?

$$
\begin{array}{c|ccccc}
 & \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \\
\text{CS} & & & & & \\
\text{MD} & & & & &
\end{array}
\quad
\left[\begin{array}{ccccc}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{array}\right]
=
\left[\begin{array}{cc}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{array}\right]
\text{x}
\left[\begin{array}{cc}
9.64 & 0 \\
0 & 5.29
\end{array}\right]
\text{x}
\left[\begin{array}{ccccc}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{array}\right]
$$

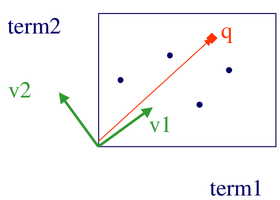
Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into ‘concept space’ – how?

$$
\begin{array}{ccccc}
\text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung}
\end{array}
$$

$$ q = \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \end{array}\right] $$



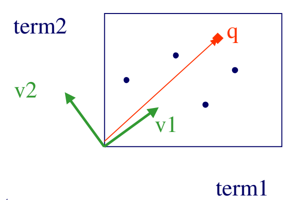
Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into ‘concept space’ – how?

$$
\begin{array}{ccccc}
\text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung}
\end{array}
$$

$$ q = \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \end{array}\right] $$



A: inner product
(cosine similarity)
with each ‘concept’ vector v_i

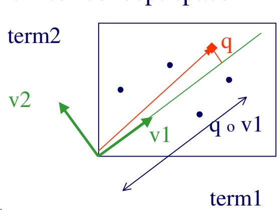
Case study - LSI

Q1: How to do queries with LSI?

A: map query vectors into ‘concept space’ – how?

$$
\begin{array}{ccccc}
\text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung}
\end{array}
$$

$$ q = \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \end{array}\right] $$



A: inner product
(cosine similarity)
with each ‘concept’ vector v_i

Case study - LSI

compactly, we have:

$q_{concept} = q\ \mathbf{V}$

Eg:

$$
q = \begin{array}{c} \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \\ \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}
\underbrace{\left[\begin{array}{cc}
0.58 & 0 \\
0.58 & 0 \\
0.58 & 0 \\
0 & 0.71 \\
0 & 0.71
\end{array}\right]}_{\text{term-to-concept similarities}}
=
\begin{array}{c} \text{CS-concept} \\ \downarrow \\ \left[\begin{array}{cc} 0.58 & 0 \end{array}\right] \end{array}
$$

CMU SCS

Case study - LSI

Drill: how would the document ('information', 'retrieval') handled by LSI?

term-to-concept similarities

CMU SCS

Case study - LSI

Drill: how would the document ('information', 'retrieval') handled by LSI? A: SAME:

$d_{concept} = d\ \mathbf{V}$

Eg:

$$d = \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array} \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{bmatrix} = \begin{bmatrix} 1.16 & 0 \end{bmatrix}$$

CS-concept

term-to-concept similarities

CMU SCS

Case study - LSI

Observation: document ('information', 'retrieval') will be retrieved by query ('data'), although it does not contain 'data'!!

CS-concept

$$\begin{array}{lccccc} & \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} \end{array}$$

$$d = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \dashrightarrow \begin{bmatrix} 1.16 & 0 \end{bmatrix}$$

$$q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \dashrightarrow \begin{bmatrix} 0.58 & 0 \end{bmatrix}$$

CMU SCS

Case study - LSI

Q1: How to do queries with LSI?

➡ Q2: multi-lingual IR (english query, on spanish text?)

CMU SCS

Case study - LSI

- Problem:
  - given many documents, translated to both languages (eg., English and Spanish)
  - answer queries across languages

CMU SCS

Case study - LSI

- Solution: ~ LSI

$$\begin{array}{c|ccccc:ccccc|} & \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} & \text{datos} & \text{informacion} & & & \\ \hline \text{CS} & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ \text{CS} & 2 & 2 & 2 & 0 & 0 & 1 & 2 & 2 & 0 & 0 \\ \text{CS} & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ \text{CS} & 5 & 5 & 5 & 0 & 0 & 5 & 5 & 4 & 0 & 0 \\ \text{MD} & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 2 & 2 \\ \text{MD} & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 2 & 3 \\ \text{MD} & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{array}$$

CMU SCS

Case study - LSI

- Solution: ~ LSI

$$
\begin{array}{c|ccccc:ccccc}
 & \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lung} & \text{datos} & \text{informacion} & & & \\
\hline
\text{CS} & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
 & 2 & 2 & 2 & 0 & 0 & 1 & 2 & 2 & 0 & 0 \\
 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
 & 5 & 5 & 5 & 0 & 0 & 5 & 5 & 4 & 0 & 0 \\
\text{MD} & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 & 2 & 2 \\
 & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 2 & 3 \\
 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1
\end{array}
$$

CMU SCS

SVD - Case studies

- multi-lingual IR; LSI queries
- compression
- PCA - 'ratio rules'
- Karhunen-Lowe transform
- query feedbacks
- google/Kleinberg algorithms

CMU SCS

Case study: compression

[Korn+97]

Problem:

- given a matrix
- compress it, but maintain 'random access'

(surprisingly, its solution leads to data mining and visualization...)

CMU SCS

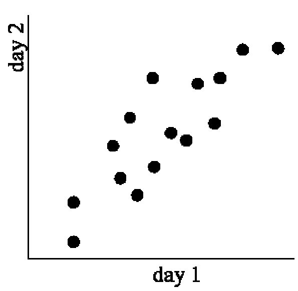
Problem - specs

- ~10**6 rows; ~10**3 columns; no updates;
- random access to any cell(s) ; small error: OK

day customer	We 7/10/96	Th 7/11/96	Fr 7/12/96	Sa 7/13/96	Su 7/14/96
ABC Inc.	1	1	1	0	0
DEF Ltd.	2	2	2	0	0
GHI Inc.	1	1	$\boxed{1}$	0	0
KLM Co.	5	5	5	0	0
Smith	0	0	0	2	2
Johnson	0	0	0	3	3
Thompson	0	0	0	1	1

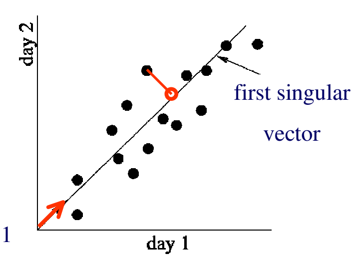
CMU SCS

Idea



CMU SCS

SVD - reminder

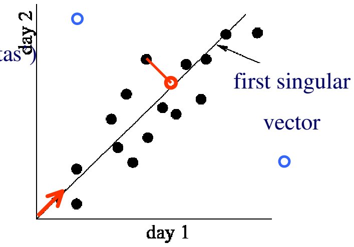


- space savings: 2:1
- minimum RMS error

CMU SCS

Case study: compression

outliers?
A: treat separately
(SVD with 'Deltas')



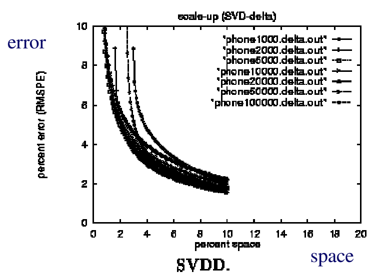
CMU SCS

Compression - Performance

- 3 pass algo (-> scalability) (HOW?)
- random cell(s) reconstruction
- 10:1 compression with < 2% error

CMU SCS

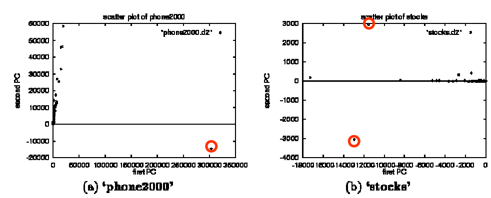
Performance - scaleup



CMU SCS

Compression - Visualization

- no Gaussian clusters; Zipf-like distribution



CMU SCS

SVD - Case studies

- multi-lingual IR; LSI queries
- compression
- PCA - 'ratio rules'
- Karhunen-Lowe transform
- query feedbacks
- google/Kleinberg algorithms

CMU SCS

PCA - 'Ratio Rules'

[Korn+00]

Typically: '*Association Rules*' (eg.,
{bread, milk} -> {butter}

But:

- which set of rules is 'better'?
- how to reconstruct missing/corrupted values?
- need binary/bucketized values

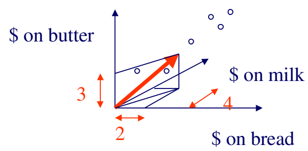
CMU SCS

PCA - 'Ratio Rules'

Idea: try to find 'concepts':

- singular vectors dictate rules about ratios:

bread:milk:butter = 2:4:3



CMU SCS

PCA - 'Ratio Rules'

Identical to PCA = Principal Components Analysis

- Q1: which set of rules is 'better'?
- Q2: how to reconstruct missing/corrupted values?
- Q3: is there need for binary/bucketized values?
- Q4: how to interpret the rules (= 'principal components')?

CMU SCS

PCA - 'Ratio Rules'

Q2: how to reconstruct missing/corrupted values?

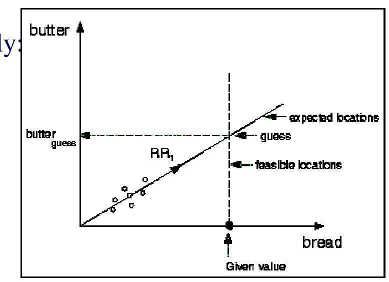
Eg:

- rule: bread:milk = 3:4
- a customer spent $6 on bread - how about milk?

CMU SCS

PCA - 'Ratio Rules'

pictorially:



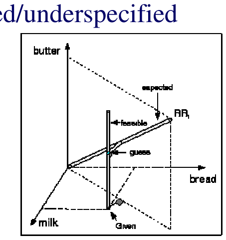
CMU SCS

PCA - 'Ratio Rules'

harder cases: overspecified/underspecified

over-specified:
•milk:bread:butter = 1:2:3
•a customer got
  • $2 bread and $4 milk
•how much milk?

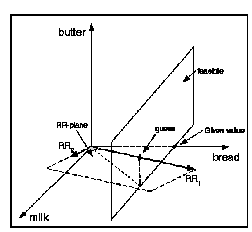
Answer: minimize distance between 'feasible' and 'expected' values (using SVD...)



CMU SCS

PCA - 'Ratio Rules'

harder cases: underspecified



CMU SCS

PCA - 'Ratio Rules'

bottom line: we can reconstruct any count of missing values

This is very useful:

- can spot outliers (how?)
- can measure the 'goodness' of a set of rules (how?)

CMU SCS

PCA - 'Ratio Rules'

Identical to PCA = Principal Components Analysis

- Q1: which set of rules is 'better'?
- √ – Q2: how to reconstruct missing/corrupted values?
- Q3: is there need for binary/bucketized values?
- Q4: how to interpret the rules (= 'principal components')?

CMU SCS

PCA - 'Ratio Rules'

- Q1: which set of rules is 'better'?
- A: the ones that needs the fewest outliers:
  - pretend we don't know a value (eg., $ of 'Smith' on 'bread')
  - reconstruct it
  - and sum up the squared errors, for all our entries
- (other Answers are also reasonable)

CMU SCS

PCA - 'Ratio Rules'

Identical to PCA = Principal Components Analysis

- √ – Q1: which set of rules is 'better'?
- √ – Q2: how to reconstruct missing/corrupted values?
- Q3: is there need for binary/bucketized values?
- Q4: how to interpret the rules (= 'principal components')?

CMU SCS

PCA - 'Ratio Rules'

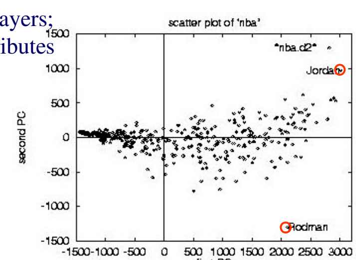
Identical to PCA = Principal Components Analysis

- √ – Q1: which set of rules is 'better'?
- √ – Q2: how to reconstruct missing/corrupted values?
- √ – Q3: is there need for binary/bucketized values? NO
- Q4: how to interpret the rules (= 'principal components')?

CMU SCS

PCA - Ratio Rules

NBA dataset
~500 players;
~30 attributes



PCA - Ratio Rules

- PCA: get singular vectors v1, v2, ...
- ignore entries with small abs. value
- try to interpret the rest

PCA - Ratio Rules

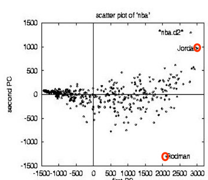
NBA dataset - **V** matrix (term to ‘concept’ similarities)

field	RR_1	RR_2	RR_3
minutes played	.808	$-.4$	
field goals			
goal attempts			
points	.406	.199	
total rebounds		$-.489$	.602
assists			$-.486$
steals			$-.07$

v1

Ratio Rules - example

- RR1: minutes:points = 2:1
- corresponding concept?



v1

Ratio Rules - example

- RR1: minutes:points = 2:1
- corresponding concept?
- A: ‘goodness’ of player

Ratio Rules - example

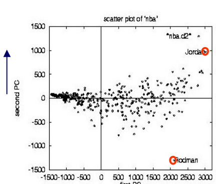
- RR2: points: rebounds negatively correlated(!)

field	RR_1	RR_2	RR_3
minutes played	.808	$-.4$	
field goals			
goal attempts			
points	.406	.199	
total rebounds		$-.489$	.602
assists			$-.486$
steals			$-.07$

Ratio Rules - example

- RR2: points: rebounds negatively correlated(!) - concept?

v2



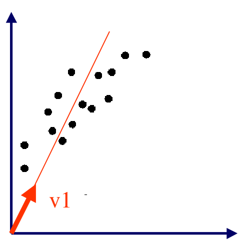
Ratio Rules - example

- RR2: points: rebounds negatively correlated(!) - concept?
- A: position: offensive/defensive

SVD - Case studies

- multi-lingual IR; LSI queries
- compression
- PCA - ‘ratio rules’
- Karhunen-Lowe transform
- query feedbacks
- google/Kleinberg algorithms

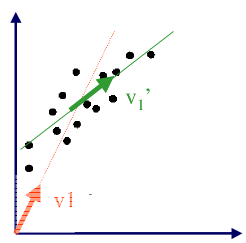
K-L transform



[Duda & Hart]; [Fukunaga]

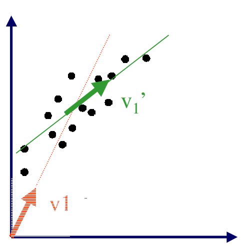
A subtle point:
SVD will give vectors that go through the origin

K-L transform



A subtle point:
SVD will give vectors that go through the origin
Q: how to find v_1’ ?

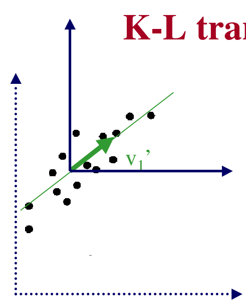
K-L transform



A subtle point:
SVD will give vectors that go through the origin
Q: how to find v_1’ ?

A: ‘centered’ PCA, ie., move the origin to center of gravity

K-L transform



A subtle point:
SVD will give vectors that go through the origin
Q: how to find v_1’ ?

A: ‘centered’ PCA, ie., move the origin to center of gravity and THEN do SVD

CMU SCS

K-L transform

- How to 'center' a set of vectors (= data matrix)?
- What is the covariance matrix?
- A: see textbook
- ('whitening transformation')

CMU SCS

Conclusions

- SVD: popular for dimensionality reduction / compression
- SVD is the 'engine under the hood' for PCA (principal component analysis)
- ... as well as the Karhunen-Lowe transform
- (and there is more to come ...)

CMU SCS

References

- Duda, R. O. and P. E. Hart (1973). Pattern Classification and Scene Analysis. New York, Wiley.
- Fukunaga, K. (1990). Introduction to Statistical Pattern Recognition, Academic Press.
- Jolliffe, I. T. (1986). Principal Component Analysis, Springer Verlag.

CMU SCS

References

- Korn, F., H. V. Jagadish, et al. (May 13-15, 1997). Efficiently Supporting Ad Hoc Queries in Large Datasets of Time Sequences. ACM SIGMOD, Tucson, AZ.
- Korn, F., A. Labrinidis, et al. (1998). Ratio Rules: A New Paradigm for Fast, Quantifiable Data Mining. VLDB, New York, NY.

CMU SCS

References

- Korn, F., A. Labrinidis, et al. (2000). "Quantifiable Data Mining Using Ratio Rules." VLDB Journal 8(3-4): 254-266.
- Press, W. H., S. A. Teukolsky, et al. (1992). Numerical Recipes in C, Cambridge University Press.