Data Preprocessing & Dimensionality reduction

M. VAZIRGIANNIS
https://www.lix.polytechnique.fr/~mvazirg/

DASCIM, LIX
https://www.lix.polytechnique.fr/dascim/

February 2022

Outline

Distance Measures

Dimensionality Reduction

Distance Measures

Machine Learning algorithms capitalize on similarity or distance measures between objects.

Similarity or distance between data points can be expressed as:

- Explicit similarity for each pair of objects
- Similarity obtained indirectly based on data vector attributes.

A distance $d(i,j)$ is a metric iff

1. $d(i,j) \geq 0$ for all i, j and $d(i,j)=0$ iff $i=j$
2. $d(i,j)=d(j,i)$ for all i and j
3. $d(i,j) \leq d(i,k)+d(k,j)$ for all i,j and k

It has to have the *shuffling invariant property*

Distance

- Notation: n objects with p attributes

$$x(i) = (x_1(i), x_2(i), ...x_p(i))$$

- Most common distance metric is *Euclidean* distance:

$$d_E(i,j) = (\sum (x_k(i) - x_k(j))^2)^{1/2}$$

- Makes sense in the case where the different measurements are are proportional; each variable measured in the same units.
- If the measurements are different, say length and weight, it is not clear – need for standardization

Weighted Euclidean distance

Finally, if we have some idea of the relative importance of each variable, we can weight them:

$$d_E(i,j) = (\sum w_k(x_k(i) - x_k(j))^2)^{1/2}$$

Other Distance Metrics

Minkowski or L_p metric:

$$d_E(i,j) = (\sum_{k=1}^{p} (x_k(i) - x_k(j))^{\lambda})^{1/\lambda}$$

Manhattan, city block or L_1 metric:

$$d_E(i,j) = \sum_{k=1}^{p} |x_k(i) - x_k(j)|$$

Chebyshev L_∞

$$d_E(i,j) = max_k |x_k(i) - x_k(j)|$$

Variants of the L_1 family

Sorensen $$d_{sor}(i,j) = \frac{\sum_{k=1}^{p} |x_k(i) - x_k(j)|}{\sum_{i=1} |x_k(i) + x_k(j)|}$$

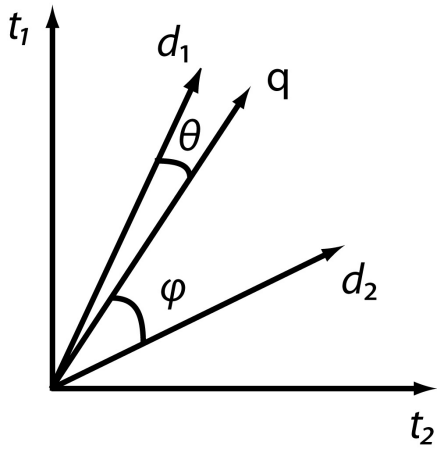
Gowers $$d_{gow}(i,j) = 1/p \sum_{k=1}^{p} |x_k(i) - x_k(j)|$$

Lorentzian $$d_{Lor}(i,j) = \sum_{k=1}^{p} ln(1 + |x_k(i) - x_k(j)|)$$

Inner product family

Inner product $s_{IP}(i,j) = \sum_{k=1}^{p} x_k(i)x_k(j)$

Harmonic Mean $s_{HM}(i,j) = 2\sum_{k=1}^{p} \frac{x_k(i)x_k(j)}{x_k(i) + x_k(j)}$



Cosine based similarity

$$sim(q,d) = \frac{q \cdot d}{|q||d|} = \frac{\sum_{k=1}^{p} w_{k,q} \cdot w_{k,d}}{\sqrt{\sum_{k=1}^{p} w_{k,q}^2} \cdot \sqrt{\sum_{k=1}^{p} w_{k,d}^2}}$$

Intersection family

Intersection $$s_{IS}(i,j) = \sum_{k=1}^{p} min((x_k(i), x_k(j))$$

Czekanowski $$s_{Cze}(i,j) = \frac{2\sum_{k=1}^{p} min(x_k(i), x_k(j))}{\sum_{k=1}^{p}(x_k(i) + x_k(j))}$$

Jaccard $$s_{Jac}(i,j) = \frac{\sum_{k=1}^{p} x_k(i)x_k(j)}{\sum_{k=1}^{p} x_k(i)^2 + \sum_{k=1}^{p} x_k(j)^2 - \sum_{k=1}^{p} x_k(i)x_k(j)}$$

Dice $$s_{Dice}(i,j) = \frac{2\sum_{k=1}^{p} x_k(i)x_k(j)}{\sum_{k=1}^{p} x_k(i)^2 + \sum_{k=1}^{p} x_k(j)^2}$$

Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions
Sung-Hyuk Cha, INT. J. OF MATHEMATICAL MODELS AND METHODS IN APPLIED SCIENCES

Squared L2 family

Squared Euclidean

$$d_{sqe}(i,j) = \sum_{k=1}^{p} (x_k(i) - x_k(j))^2$$

Pearson x^2

$$d_{pre}(i,j) = \frac{\sum_{k=1}^{p} (x_k(i) - x_k(j))^2}{x_k(j)}$$

Divergence

$$d_{DIV}(i,j) = 2\sum_{k=1}^{p} \frac{(x_k(i) - x_k(j))^2}{(x_k(i) + x_k(j))^2}$$

Shannon's entropy family

Kullback Leibler $$d_{KL}(i,j) = \sum_{k=1}^{p} x_k(i) ln \frac{x_k(i)}{x_k(j)}$$

Jeffreys $$d_{JF}(i,j) = \sum_{k=1}^{p} (x_k(i) - x_k(j)) ln \frac{x_k(i)}{x_k(j)}$$

K- divergence $$d_{kids}(i,j) = \sum_{k=1}^{p} x_k(i) ln \frac{2x_k(i)}{x_k(i) + x_k(j)}$$

Jensen Shannon

$$d_{JS}(i,j) = 1/2[\sum_{k=1}^{p} x_k(i) ln \frac{2x_k(i)}{x_k(i) + x_k(j)} + \sum_{k=1}^{p} x_k(j) ln \frac{2x_k(j)}{x_k(i) + x_k(j)}]$$

Distance metrics – Nominal values / text

Nominal variables

Number of matches divided by number of dimensions

A	A	B	B	C	B	B	C	C	A
A	B	B	A	C	B	B	C	C	C

7/10

- Edit (Levenshtein) distance

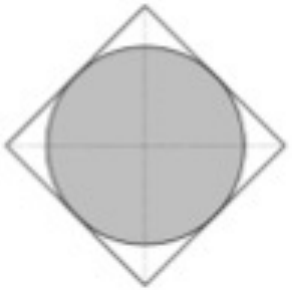
kitten → **s**itten (substitution of "s" for "k")

sitt**e**n → sitt**i**n (substitution of "i" for "e")

sittin → sittin**g** (insertion of "g" at the end)”

Dimensionality Reduction

Empty space phenomenon



Hyper sphere within a hyper rectangle

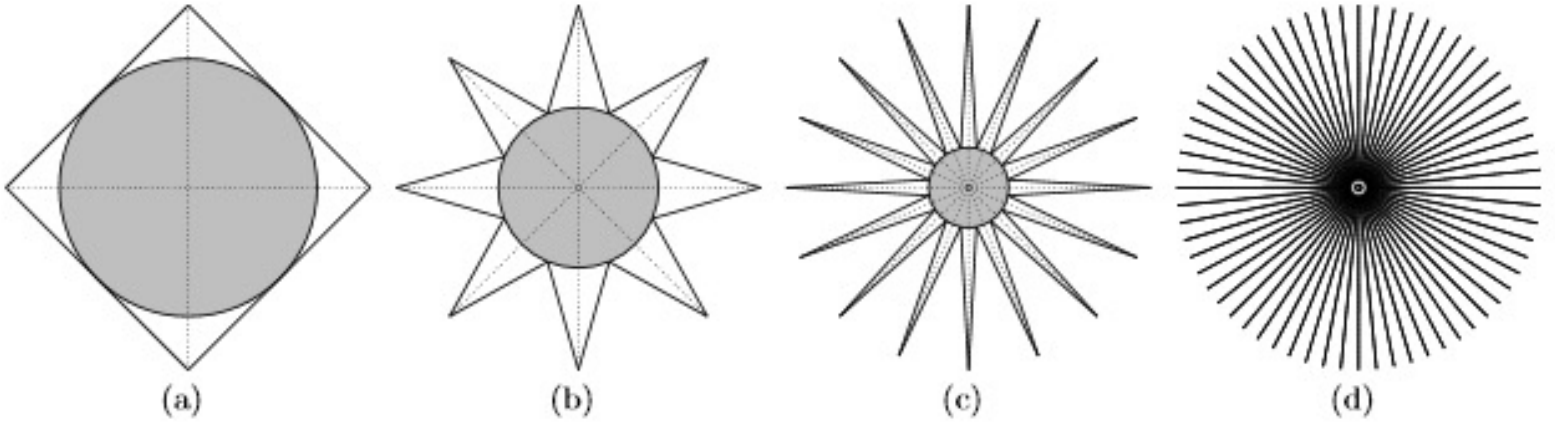
Respective volumes: $\mathbf{V}(\boldsymbol{S}) = {}^{2r^d\pi^{\frac{d}{2}}}\Big/_{d\Gamma(^d/_2)}, \boldsymbol{V}(\boldsymbol{R}) = (\boldsymbol{2r})^{\boldsymbol{d}}$

The fraction of the sphere within the rectangle becomes insignificant with d increasing:

$$\lim_{d\to\infty}\left({}^{\pi^{\frac{d}{2}}}\Big/_{d2^{d-1}\Gamma(^d/_2)}\right) = 0$$

- the normal distribution in high dimensions
- longest/shortest distances converge.
- clustering becomes infeasible

Inscription of hyper sphere in a hypercube



The radius of the inscribed circle accurately reflects the difference between the volume of the hypercube and the inscribed hypersphere in d-dimensions.

http://www.cs.rpi.edu/~zaki/www-new/uploads/Dmcourse/Main/chap6.pdf

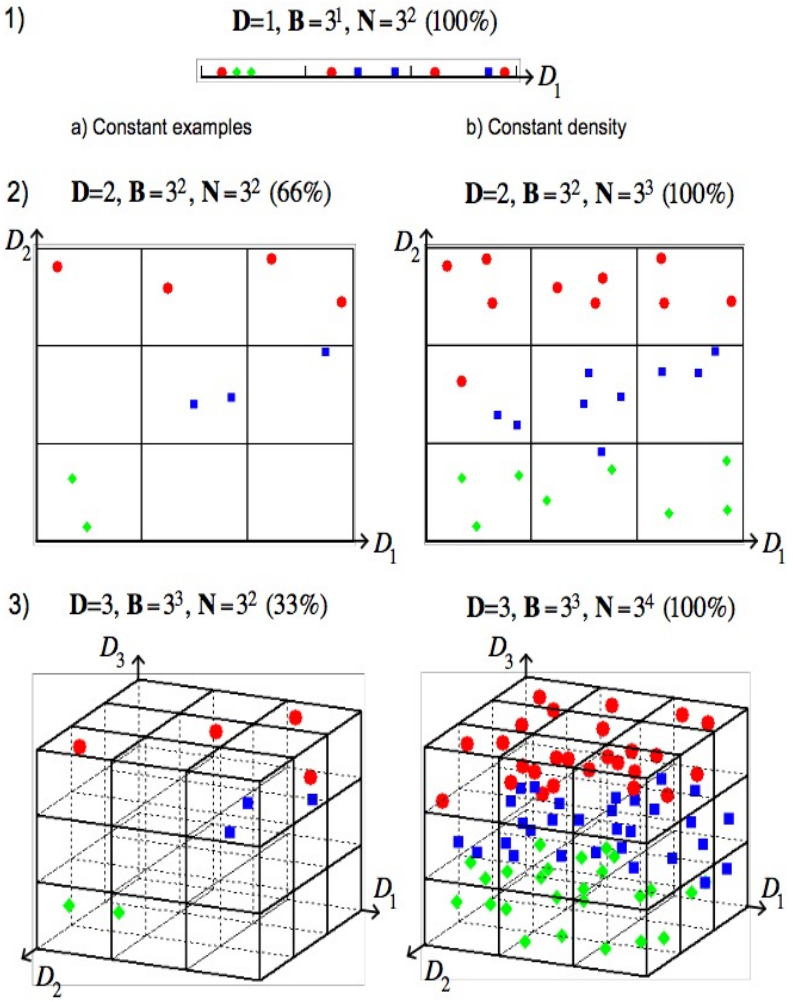
Curse of Dimensionality [Belmann 1961]

- Some coordinates do not contribute to the data representation.
- Subsets of the dimensions may be highly correlated.
- Nearest neighbor is distorted in a high dimensional space
- Low dimension intuitions do not apply to high dimensions

Curse of Dimensionality

Assuming 3 classes (colors)

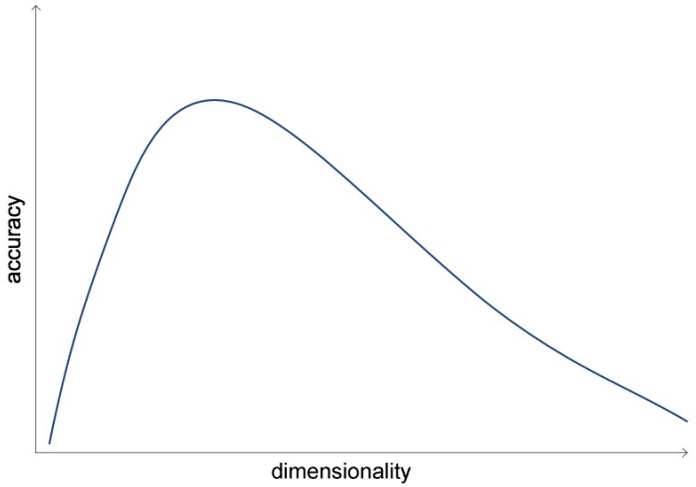
- The same number of points in higher dimensions (sparsity)
- need exponentially more points to maintain density in higher dimensions and be able to estimate density necessary for non parametric classification (curse of dimensionality)
- Data tends to gather in extremes of small areas of the multidimensional space (empty space phenomenon)



Curse of Dimesionality

Point queries

- distance to the nearest neighbour tends to converge distance to the farthest neighbour as dimensionality increases Beyer et al., [1999]
- these properties manifest themselves by a decrease of overall accuracy of system according to the statistical learning theory approach [Vapnik, 1998].
- for a given dataset, there is a maximum number of dimensions above which the quality of data analysis degrades when the number of training samples is small relative to dimensionality



Deterministic dimensionality reduction

- methods optimize an objective function that does not contain any local optima - the solution space is convex [Boyd and Vandenberghe, 2004].
- The objective function has usually the form of a generalized Rayleigh-Ritz and therefore optimized by solving the generalized eigenvalue problem.
- The final embedded space is formed by eigenvectors which correspond to smallest or largest eigenvalues.

Deterministic methods classification

- *Global methods:* eigen-decomposition of a dense cost matrix
  - Methods: Principal Component Analysis, Multidimensional Scaling, Kernel Principal Component Analysis, Isomap, Maximum Variance Unfolding
- *Local methods:* eigen-decomposition of a sparse cost matrix
  - Methods: Locally Linear Embedding, Laplacian Eigenmaps.

Dim. Reduction – Linear Algorithms

Matrix Factorization methods

- Principal Components Analysis (PCA)
- Singular Value Decomposition (SVD)
- Multidimensional Scaling (MDS)
- Non negative Matrix Factorization (NMF)
- Latent Semantic Indexing (LSI)

Low Rank Approximation

Data: $\boldsymbol{X = \{x_i \in R^{mxn} | x_i}$ **columns of** $\boldsymbol{X\}}$

Goal: approximate $\boldsymbol{X = UV^T}$,

$\boldsymbol{U \in R^{mxr}}$, $\boldsymbol{V \in R^{nxr}}$, , $\boldsymbol{r<<n}$

- each data vector $\boldsymbol{x_i}$: $\boldsymbol{x_i \sim U v_i^T}$, $\boldsymbol{v_i}$ is the i-th column of $\boldsymbol{V}$.

Geometric interpretation:

- each data vector $\boldsymbol{x_i \in R^m}$, $\boldsymbol{_i \sim U v_i^T}$, is approximated by its projection to an r-dimesional space spanned by the column vectors of U
- $\boldsymbol{Y = U\ V^T}$ the approximation matrix, max rank r

Evaluating the approximation

- Assuming two matrices A_{mxn}, B_{mxk} we need to define their similarity/distance.
- A popular matrix norm is the Frobenius one

$$|A|_F = \sum_{i=1}^{m}\sum_{j=1}^{n}\alpha_{ij}^2 = \sum_{i=1}^{min(m,n)}\sigma_i^2$$

- Other potential matrix distance measures:

$$d_1(A,B) = \sum_{i=1}^{n}\sum_{j=1}^{n}|\alpha_{ij} - \beta_{ij}|$$

$$d_2(A,B) = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_{ij} - \beta_{ij})^2}$$

SOME CONTRIBUTIONS TO DIMENSIONALITY REDUCTION, Wei Tong, Ph.D. thesis, 2010, Michigan State University
http://www.ece.uprm.edu/~domingo/teaching/ciic8996/SOME%20CONTRIBUTIONS%20TO%20DIMENSIONALITY%20REDUCTION.pdf

Dim. Reduction–Eigenvectors

A: nxn matrix

- eigenvalues λ: $|A-\lambda I|=0$
- Eigenvectors x : $Ax=\lambda x$
- Matrix rank: # linearly independent rows or columns
- A real symmetric table A nxn can be expressed as: $A=U\Lambda U^T$
- U's columns are A's eigenvectors
- Λ' s diagonal contains A's eigenvalues
- $\boldsymbol{A=U\Lambda U^T=\lambda_1 x_1 x^T{}_1+\lambda_2 x_2 x^T{}_2+\ldots+\lambda_n x_n x^T{}_n}$
- $x_1 x^T{}_1$ represents projection via x_1 (λ_i eigenvalue, x_i eigenvector)
- Interpretations: xx^T vs. x^Tx

Singular Value Decomposition (SVD)

Eigen values and eigenvectors decomposition is applied to square matrices. For non square matrices we apply ***Singular Value Decomposition.***

Let **X** a **mxn table**, $X = U\Sigma V^T$

U : orthogonal mxm, its columns are the eigenvectors of XX^T.

U,V define orthogonal basis: $U^T U = VV^T = 1$

Σ: mxn contains A's singular values (square roots of XX^T eigenvalues)

V : nxn, its columns are the eigenvectors of $X^T X$

Singular Value Decomposition (SVD) - I

Proof:

$X = U\Sigma V^T, X^T = V\Sigma^T U^T =>$

$XX^T = U\Sigma(V^T V)\Sigma U^T = U\Sigma\Sigma^T U^T$

Similarly: $X^T X = V\Sigma^T \Sigma V^T$

Therefore: U: eigenvectors of XX^T (V: eigenvectors of $X^T X$)

Σ: sqrt of the eigenvalues of XX^T

X k-dimensional representation: $X_k = U_k \Sigma_k V_k^T$

Singular Value Decomposition (SVD) - II

Matrix approximation $$X_k = U_k \Sigma_k {V_k}^T$$

The best rank k approximation Y' of a matrix X. (minimizing the Frobenius norm)

$$\|A\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2 = \operatorname{trace}(AA^H) = \sum_{i=1}^{\min\{m,n\}} \sigma_i^2$$

where A^H transpose of A, σ_i are the singular values of A, and the trace function is used.

Multidimensional Scaling (MDS)

- Decomposition of the data similarity matrix: XX^T
- Aim to minimize the stress:

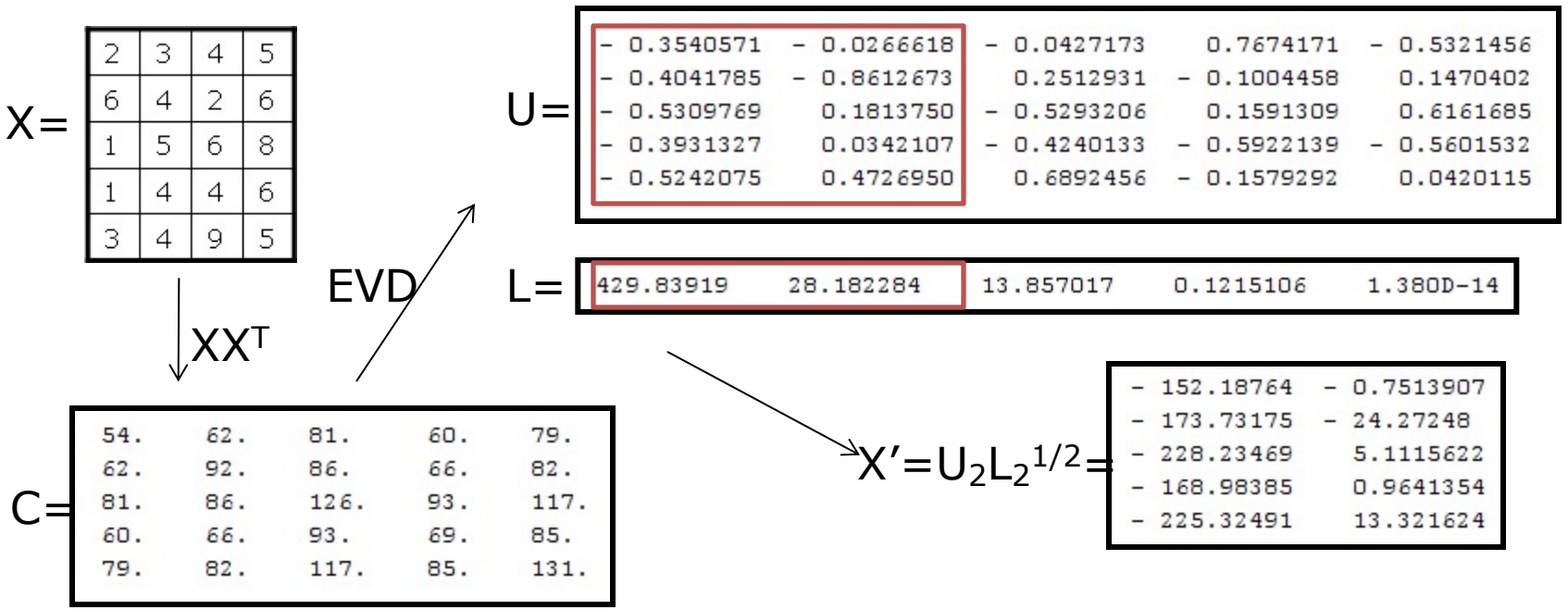
$$stress = \frac{\sum_{ij}(d(i,j) - d'(i,j))^2}{\sum_{ij}(d(i,j))^2}$$

Complexity $O(N^3)$ (N: number of vectors)

- Result:
  - A new representation of the data in a lower dimensional space.
- Implement usually by:
  - Eigen decomposition of the inner product matrix
  - projection on the k eigenvectors corresponding to the k largest eigenvalues.

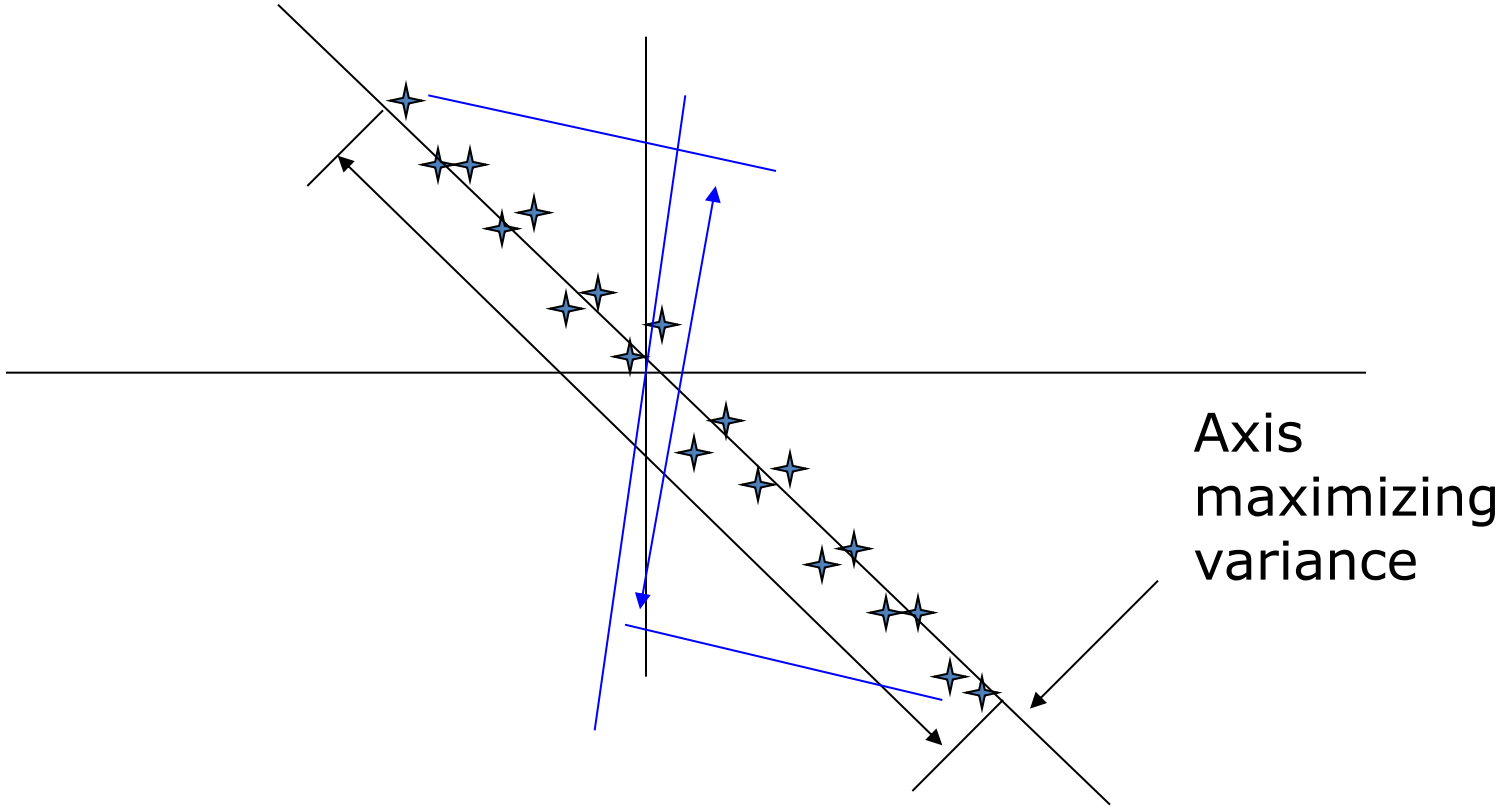
Multidimensional Scaling

- **Data is given as rows in X**
  - $C=XX^T$ (inner product of x_i with x_j)
  - Eigen decomposition of $C' = ULU^{-1}$
  - Eventually $X' = U_k L_k^{1/2}$, where k is the projection dimension



PCA: Dimensionality reduction based on variance maintenance

The main concept is maintaining as much as possible data's variance.



Covariance Matrix

Let Matrix $X = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}$ where Xi vectors

covariance matrix Σ is the matrix whose (i, j) entry is the covariance

$$\Sigma = \begin{bmatrix} \mathrm{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \mathrm{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \mathrm{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \mathrm{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \mathrm{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$

Also: $cov(X) = X'^{T}X'$, where $X' = X\text{-}M$

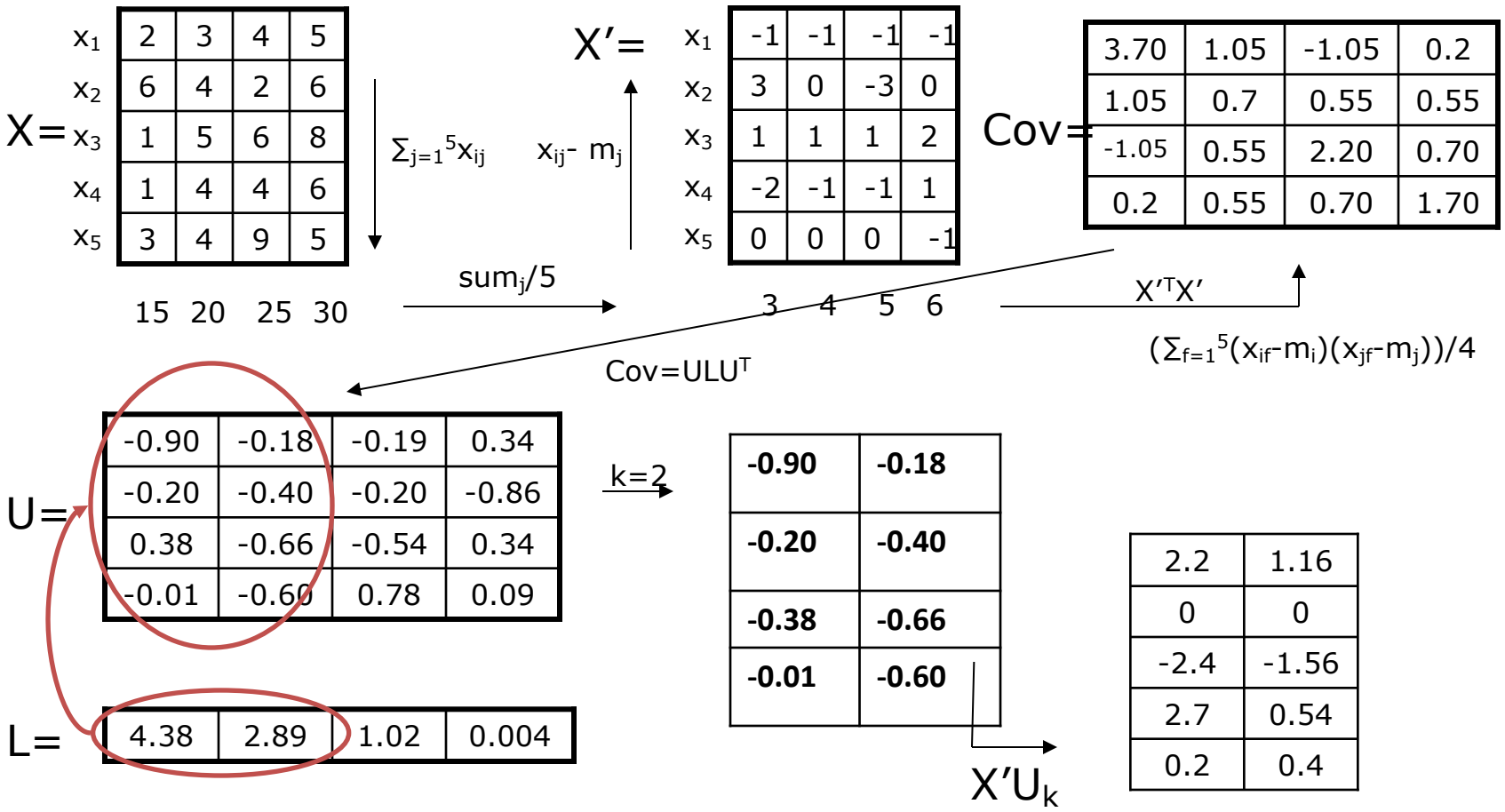
Principal Components Analysis (PCA)

- Assumption: n data rows x p variables in the matrix X_{nxp}
- "centralize" the matrix: subtract mean values from columns: $X'=(X-M)$
- Calculate covariance matrix - nxn, in each cell of which (i,j) we have the covariance of X_i,X_j: $W = X'^T X'$
- Calculate eigenvalues and eigenvectors of $W= UAU^T$
- Retain k largest eigenvalues and corresponding eigenvectors

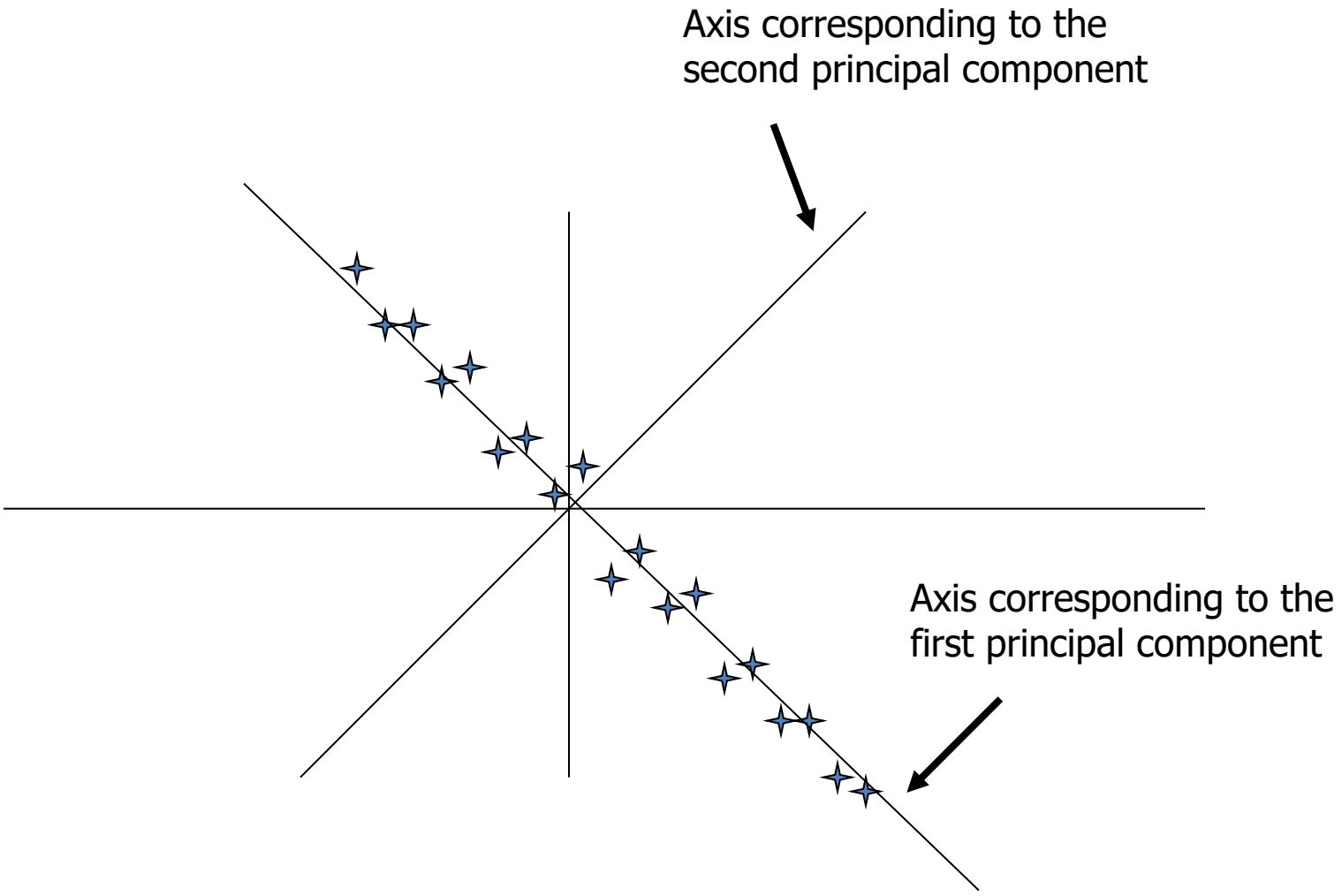
$$\Sigma^p_{j=k+1}\lambda_j/\Sigma^p_{j=1}\lambda_j > 85\%$$

Projection : $X'U_k$

Principal Components Analysis



PCA, example


Axis corresponding to the
second principal component
Axis corresponding to the
first principal component

Non Negative Matrix factorization (NMF)

- Applying SVD results in factorized matrices with positive and negative elements may contradict the physical meaning of the result.

Example:

- X gray-scale image intensities, Y its SVD approximation

- difficult to interpret the reconstructed matrix Y for a gray-scale image with negative elements.

- *Nonnegative matrix factorization (NMF)*

find the reduced rank *nonnegative factors* to approximate a given nonnegative data matrix.

Non Negative Matrix factorization (NMF)

Assume X mxn data matrix $(X_{ij} \geq 0), r \ll \min(m,n)$

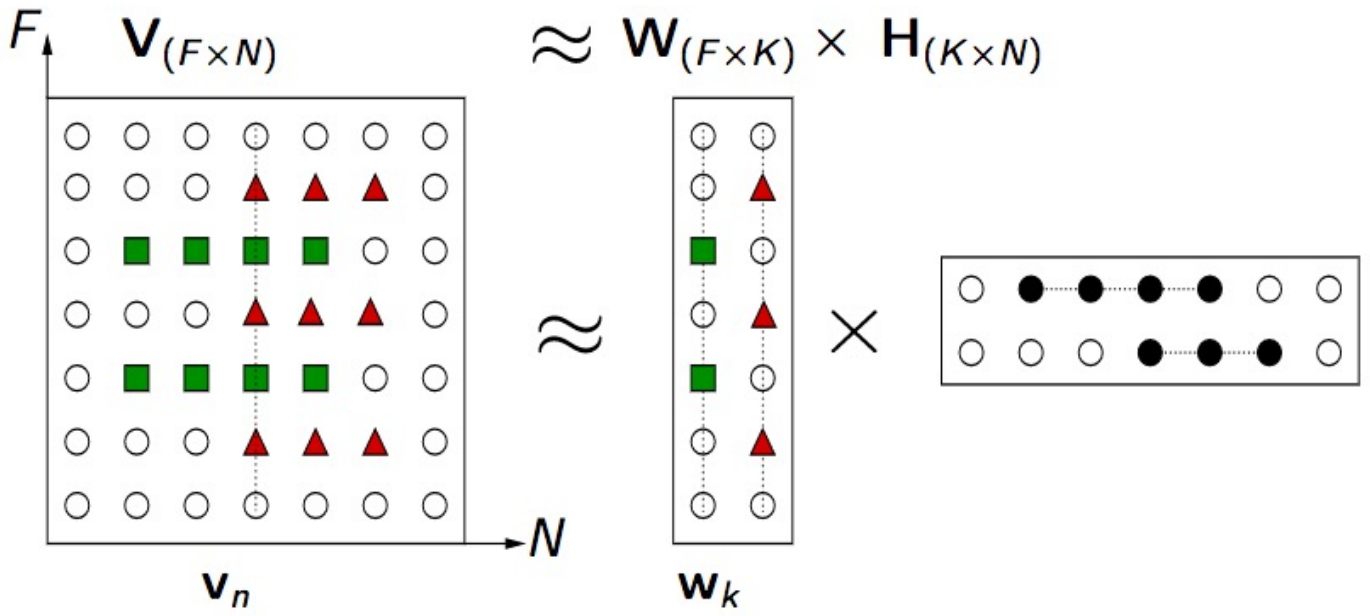
Then NMF finds non negative matrices

$$U \in R^{mxr}, V \in R^{nxr} : X \approx UV^T$$

To find U, V is to minimize Euclidian Distance $X - UV^T$:

$$\min_{U,V} \quad f(U,V) = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(X_{ij}\log\frac{X_{ij}}{(UV^\top)_{ij}} - X_{ij} + (UV^\top)_{ij}\right)$$
$$\text{s. t.} \quad U_{ia} \geq 0, V_{jb} \geq 0, \forall i, a, b, j.$$

Explaining data by factorization



data matrix

"explanatory variables"
"basis", "dictionary",
"patterns", "topics"

"regressors",
"activation coefficients",
"expansion coefficients"

Illustration by C. Févotte

Non Negative Matrix factorization (NMF)

- Data is often nonnegative by nature
    - pixel intensities; occurrence counts; food or energy consumption; user scores; stock market values;
- Interpretability of the results, optimal processing of nonnegative data may call for processing under Nonnegativity constraints

.

- Applying SVD results in factorized matrices with positive and negative elements may contradict the physical meaning of the result.

    - *Nonnegative matrix factorization (NMF)*

    find the reduced rank *nonnegative factors* to approximate a given nonnegative data matrix.

NMF

Assume X (mxn) data matrix and r<< m,n

NMF aims to find non negative matrices

$$U \in R^{mxr}, V \in R^{rxn} : X \approx UV^T$$

To find U.V, optimization problem:

$$min_{(U,V)} ||X - UV^T||_2$$

Alternative error function:

$$\min_{U,V} \quad f(U,V) = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(X_{ij}\log\frac{X_{ij}}{(UV^\top)_{ij}} - X_{ij} + (UV^\top)_{ij}\right)$$
$$\text{s. t.} \quad U_{ia} \geq 0, V_{jb} \geq 0, \forall i, a, b, j.$$

Alternating Least squares

1. Suppose we know U, *with* V unknown.

for each j we could minimize $||X_{.j} - UV_{.j}^T||_2$

- find $V_{.j}$ that minimizes with $X_{.j}$ and U known.
- Frobenius norm: sum of squares,
  - minimization is a least-squares problem, i.e. linear regression
  - "predicting" $X.j$ from W.

$$V_{.j} = (U^T U)^{-1} U^T X.j$$

  - repeat for all columns $V_{.j}$

2. assume V, *with* U unknown: $X^T = VU^T$

  - Interchange roles of U, V in the above optimization
  - Compute a row of U, repeat for all rows

Alternating Least squares

Putting all this together

- first choose initial guesses, random numbers,for U and V
- alternate:
  - Compute U assuming V known
  - Compute V based on that new U
  - …
- may generate some negative values: simply truncate to 0

Other NMF Algorithms

Multiplicative: updating solutions U and V

$$V_{bj}^{\top} \leftarrow V_{bj}^{\top} \frac{\left(U^{\top} X\right)_{bj}}{\left(U^{\top} U V^{\top}\right)_{bj}} \qquad U_{ia} \leftarrow U_{ia} \frac{(XV)_{ia}}{(UV^{\top} V)_{ia}}$$

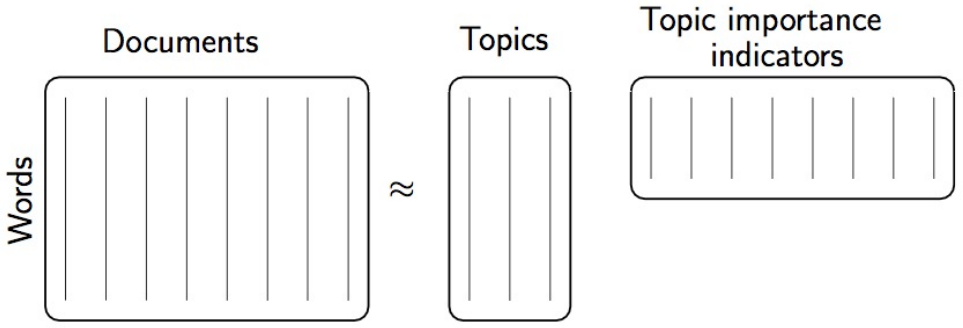
Gradient descent algorithms

$$V_{bj}^{\top} \leftarrow V_{bj}^{\top} - \epsilon_V \frac{\partial f}{\partial V_{bj}^{\top}} \qquad U_{ia} \leftarrow U_{ia} - \epsilon_U \frac{\partial f}{\partial U_{ia}}$$

ε_V and ε_U are the step sizes.

NMF issues, applications

- Uniqueness and Convergence
- U_{mxr} , r (rank) choice: via SVD…
- Applications
  - Topic detection
  - Source separation (music , speech)
  - Clustering
  - Recommendations



Latent Semantic Indexing (LSI) -I

- Finding similarity with exact keyword matching is problematic.
- Using SVD we process the initial document-term document.
- Then we choose the k larger singular values. The resulting matrix is of order k and is the most similar to the original one based on the Frobenius norm than any other k-order matrix.

Latent Semantic Indexing (LSI) - II

- The initial matrix is SVD decomposed as: $A=ULV^T$
- Choosing the top-k singular values from L we have:

$$A_k=U_kL_kV_k^T\ ,$$

- L_k is square kxk containing the top-k singular values of the diagonal in matrix L,
- U_k, the mxk matrix containing the first k columns in U (left singular vectors)
- V_k^T, the kxn matrix containing the first k lines of V^T (right singular vectors)

Typical values for $\varkappa$~200-300 (empirically chosen based on experiments appearing in the bibliography)

LSI capabilities

- Term to term similarity: $A_k A_k^T = U_k L_k^2 U_k^T$

  $A_k = U_k L_k V_t$

- Document-document similarity: $A_k^T A_k = V_k L_k^2 V_k^T$

- Term document similarity (as an element of the transformed

  – document matrix)

- Extended query capabilities transforming initial query q to

  $q_n : \ q_n = q^T U_k L_k^{-1}$

- Thus q_n can be regarded a line in matrix V_k

LSI – an example

LSI application on a term – document matrix

C1: Human machine Interface for Lab ABC computer application

C2: A survey of user opinion of computer system response time

C3: The EPS user interface management system

C4: System and human system engineering testing of EPS

C5: Relation of user-perceived response time to error measurements

M1: The generation of random, binary unordered trees

M2: The intersection graph of path in trees

M3: Graph minors IV: Widths of trees and well-quasi-ordering

M4: Graph minors: A survey

- The dataset consists of 2 classes, 1st: "human – computer interaction" (c1-c5) 2nd: related to graph (m1-m4). After feature extraction the titles are represented as follows.

Indexing by Latent Semantic Analysis (1990) Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Richard Harshman, Journal of the American Society of Information Science

LSI – an example

	C1	C2	C3	C4	C5	M1	M2	M3	M4
human	1	0	0	1	0	0	0	0	0
Interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
User	0	1	1	0	1	0	0	0	0
System	0	1	1	2	0	0	0	0	0
Response	0	1	0	0	1	0	0	0	0
Time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
Survey	0	1	0	0	0	0	0	0	1
Trees	0	0	0	0	0	1	1	1	0
Graph	0	0	0	0	0	0	1	1	1
Minors	0	0	0	0	0	0	0	1	1

LSI – an example

$A = ULV^T$

$A =$

1	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0
0	1	1	2	0	0	0	0	0
0	1	0	0	1	0	0	0	0
0	1	0	0	1	0	0	0	0
0	0	1	1	0	0	0	0	0
0	1	0	0	0	0	0	0	1
0	0	0	0	0	1	1	1	0
0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	0	1	1

LSI – an example

$A = ULV^T$

U=

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41	0	0	0
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11	0	0	0
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49	0	0	0
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01	0	0	0
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27	0	0	0
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05	0	0	0
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05	0	0	0
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17	0	0	0
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58	0	0	0
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23	0	0	0
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23	0	0	0
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18	0	0	0

LSI – an example

$A=ULV^T$

L=

3.34	0	0	0	0	0	0	0	0
0	2.54	0	0	0	0	0	0	0
0	0	2.35	0	0	0	0	0	0
0	0	0	1.64	0	0	0	0	0
0	0	0	0	1.50	0	0	0	0
0	0	0	0	0	1.31	0	0	0
0	0	0	0	0	0	0.85	0	0
0	0	0	0	0	0	0	0.56	0
0	0	0	0	0	0	0	0	0.36
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

LSI – an example

$A=ULV^T$

V=

0.20	-0.06	0.11	-0.95	0.05	-0.08	0.18	-0.01	-0.06
0.61	0.17	-0.50	-0.03	-0.21	-0.26	-0.43	0.05	0.24
0.46	-0.13	0.21	0.04	0.38	0.72	-0.24	0.01	0.02
0.54	-0.23	0.57	0.27	-0.21	-0.37	0.26	-0.02	-0.08
0.28	0.11	-0.51	0.15	0.33	0.03	0.67	-0.06	-0.26
0.00	0.19	0.10	0.02	0.39	-0.30	-0.34	0.45	-0.62
0.01	0.44	0.19	0.02	0.35	-0.21	-0.15	-0.76	0.02
0.02	0.62	0.25	0.01	0.15	0.00	0.25	0.45	0.52
0.08	0.53	0.08	-0.03	-0.60	0.36	0.04	-0.07	-0.45

LSI – an example

Choosing the 2 largest singular values we have

$U_k =$

0.22	-0.11
0.20	-0.07
0.24	0.04
0.40	0.06
0.64	-0.17
0.27	0.11
0.27	0.11
0.30	-0.14
0.21	0.27
0.01	0.49
0.04	0.62
0.03	0.45

$L_k =$

3.34	0
0	2.54

$V_k^T =$

0.20	0.61	0.46	0.54	0.28	0.00	0.02	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53

LSI (2 singular values)

$A_k =$

	C1	C2	C3	C4	C5	M1	M2	M3	M4
human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
Interface	0.14	0.37	0.33	0.40	0.16	-0.03	-0.07	-0.10	-0.04
Computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
User	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
System	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
Response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
Time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
Survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
Trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
Graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
Minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62

LSI Example

- Query: "human computer interaction" retrieves documents: c_1,c_2, c_4 but *not* c_3 and c_5.
- If we submit the same query (based on the transformation shown before) to the transformed matrix we retrieve (using cosine similarity) all c_1-c_5 even if c_3 and c_5 have no common keyword to the query.
- According to the transformation for the queries we have:

Query transformation

	query
human	1
Interface	0
computer	1
User	0
System	0
Response	0
Time	0
EPS	0
Survey	0
Trees	0
Graph	0
Minors	0

q=

1
0
1
0
0
0
0
0
0
0
0
0

Query transformation

$q^T =$

1	0	1	0	0	0	0	0	0	0	0	0

$U_k =$

0.22	-0.11
0.20	-0.07
0.24	0.04
0.40	0.06
0.64	-0.17
0.27	0.11
0.27	0.11
0.30	-0.14
0.21	0.27
0.01	0.49
0.04	0.62
0.03	0.45

$L_k =$

0.3	0
0	0.39

$q_n = q^T U_k L_k =$

0.138	-0.0273

Query transformation

Map docs to the 2 dim space $V_k L_k =$

0.20	-0.06
0.61	0.17
0.46	-0.13
0.54	-0.23
0.28	0.11
0.00	0.19
0.01	0.44
0.02	0.62
0.08	0.53

3.34	0
0	2.54

=

0.67	-0.15
2.04	0.43
1.54	-0.33
1.80	-0.58
0.94	0.28
0.00	0.48
0.03	1.12
0.07	1.57
0.27	1.35

$q_n L_k =$

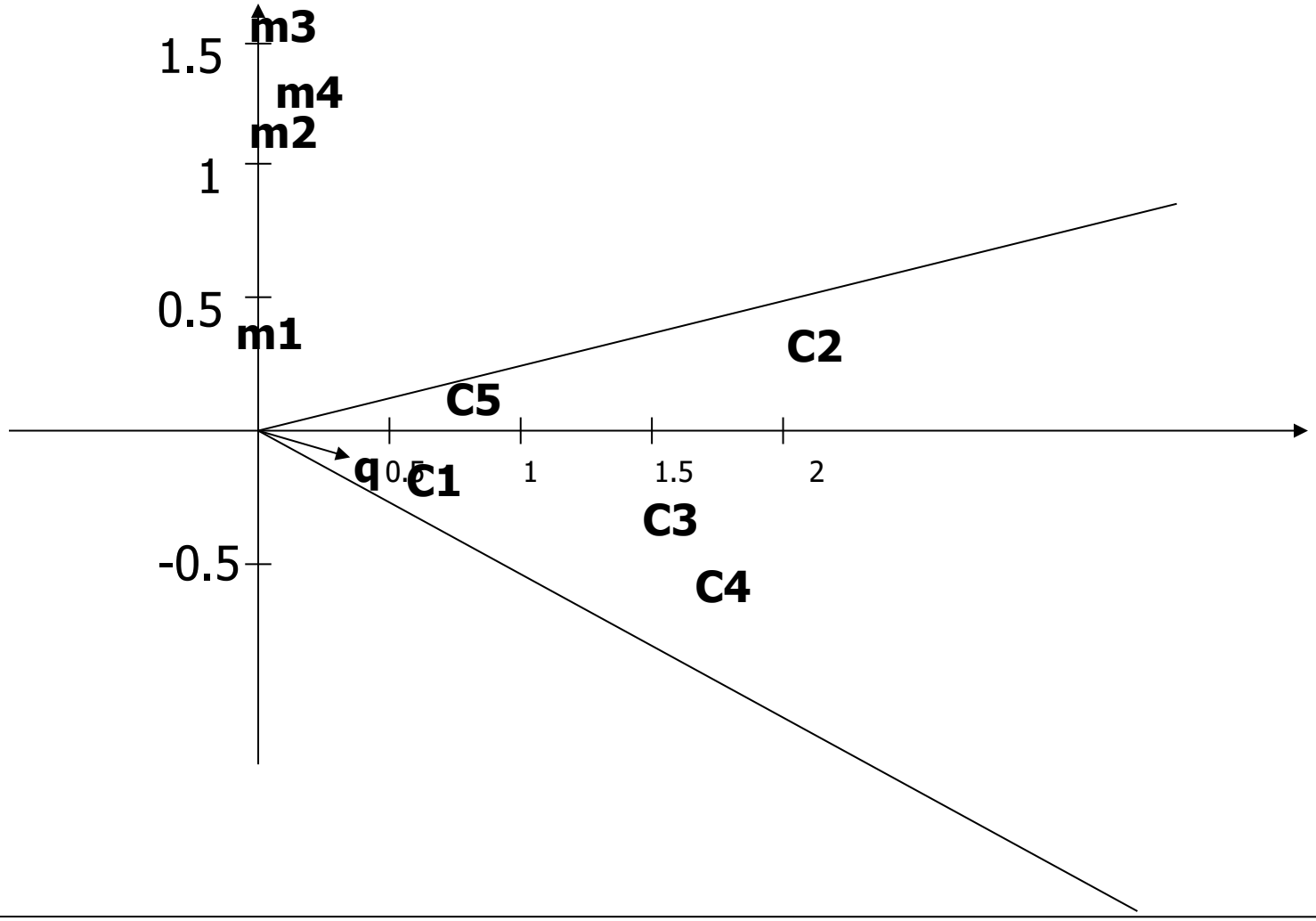
0.138	-0.0273

3.34	0
0	2.54

=

0.46	-0.069

Query transformation



Query transformation

- Comparison of the transformed query to the new document vectors based on cosine similarity, where the similarity is computed as:

$$Cos(x,y)=<x,y>/||x||.||y||$$

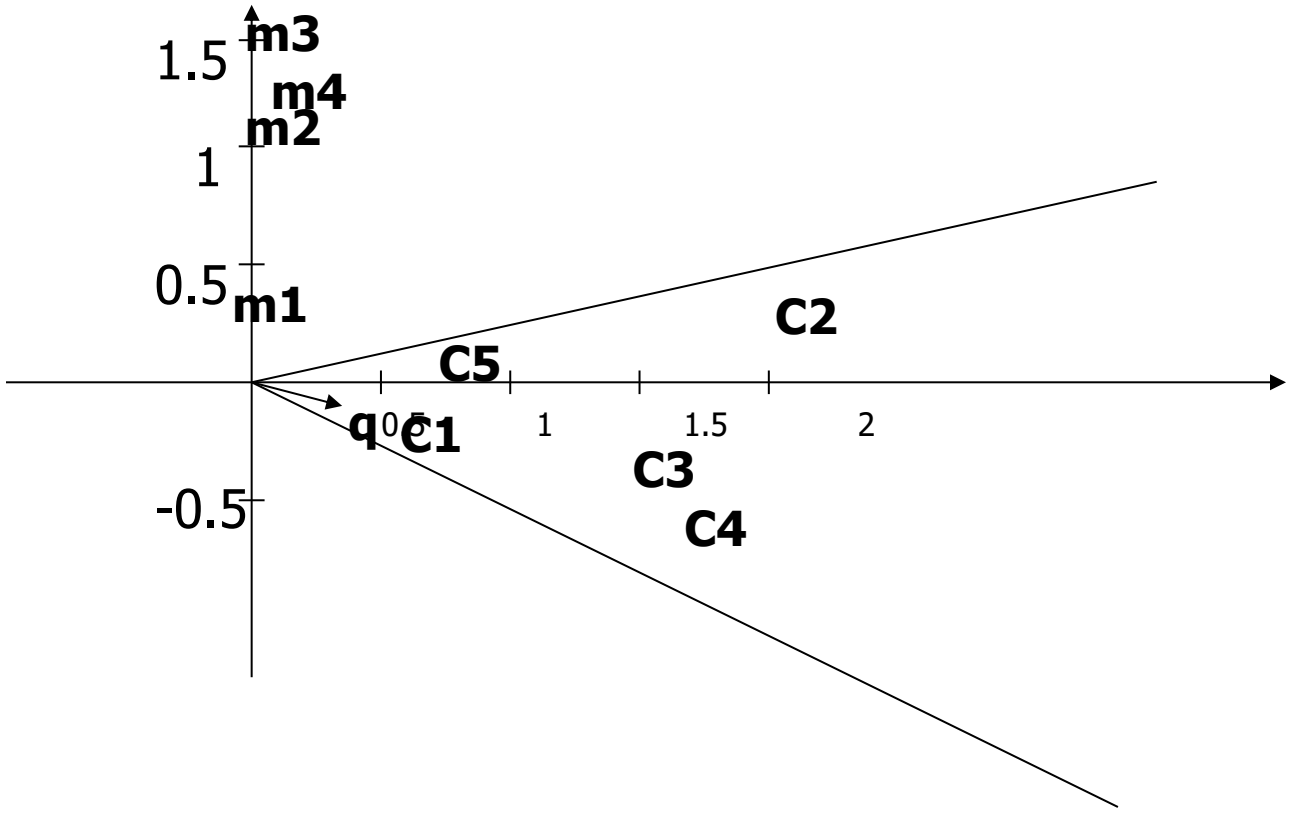
Where $x=(x_1,\ldots,x_n)$, $y=(y_1,\ldots,y_n)$

$$<x,y>=x_1*y_1+\ldots+x_n*y_n$$

Query transformation

- The cosine similarity matrix of query vector to the documents is:

	query
C1	0.99
C2	0.94
C3	0.99
C4	0.99
C5	0.90
M1	-0.14
M2	-0.13
M3	-0.11
M4	0.05



Useful References

- **Principles of Data Mining, David J. Hand, Heikki Mannila and Padhraic Smyth** MIT Press 2001
- **T. Hastie, R. Tibshirani, and J. Friedman, Elements of Statistical Learning,** Springer Verlag, 2001
- **Dash, Manoranjan, and Huan Liu**. "Feature selection for classification." *Intelligent data analysis* 1.1-4 (1997): 131-156.
- **N. R. Draper and H. Smith, Applied Regression Analysis, 2nd edition, Wiley, 1981** (the "bible" for classical regression methods in statistics
- **An introduction to variable and feature selection, Isabelle Guyon, André Elisseeff**, The Journal of Machine Learning Research archive Volume 3, 3/1/2003, pp. 1157-1182
- **Mohammed J. Zaki, course notes, High Dimesional Notes** http://www.cs.rpi.edu/~zaki/www-new/uploads/Dmcourse/Main/chap6.pdf
- http://www.bmva.org/thesis-archive/2011/2011-lewandowski.pdf
- Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions, **Sung-Hyuk Cha**, Int. J. Of Mathematical Models And Methods In Applied Sciences