DATA MINING
LECTURE 8

Dimensionality Reduction

PCA -- SVD

The curse of dimensionality

- Real data usually have thousands, or millions of dimensions
  - E.g., web documents, where the dimensionality is the vocabulary of words
  - Facebook graph, where the dimensionality is the number of users
- Huge number of dimensions causes problems
  - Data becomes very sparse, some algorithms become meaningless (e.g. density based clustering)
  - The complexity of several algorithms depends on the dimensionality and they become infeasible.

Dimensionality Reduction

- Usually the data can be described with fewer dimensions, without losing much of the meaning of the data.
  - The data reside in a space of lower dimensionality

- Essentially, we assume that some of the data is noise, and we can approximate the useful part with a lower dimensionality space.
  - Dimensionality reduction does not just reduce the amount of data, it often brings out the useful part of the data

Dimensionality Reduction

- We have already seen a form of dimensionality reduction

- LSH, and random projections reduce the dimension while preserving the distances

Data in the form of a matrix

- We are given n objects and d attributes describing the objects. Each object has d numeric values describing it.

- We will represent the data as a $n \times d$ real matrix A.
  - We can now use tools from linear algebra to process the data matrix

- Our goal is to produce a new $n \times k$ matrix B such that
  - It preserves as much of the information in the original matrix A as possible
  - It reveals something about the structure of the data in A

Example: Document matrices

$$
\begin{array}{cc}
 & \textbf{d} \text{ terms} \\
 & \text{(e.g., theorem, proof, etc.)} \\
\begin{array}{c}\textbf{n} \\ \text{documents}\end{array} & \begin{pmatrix} \\ \\ \quad\quad\quad\quad A \quad\quad\quad\quad \\ \\ \\ \end{pmatrix}
\end{array}
$$

A_{ij} = frequency of the **j**-th term in the **i**-th document

Find subsets of terms that bring documents together

Example: Recommendation systems

d movies

n customers

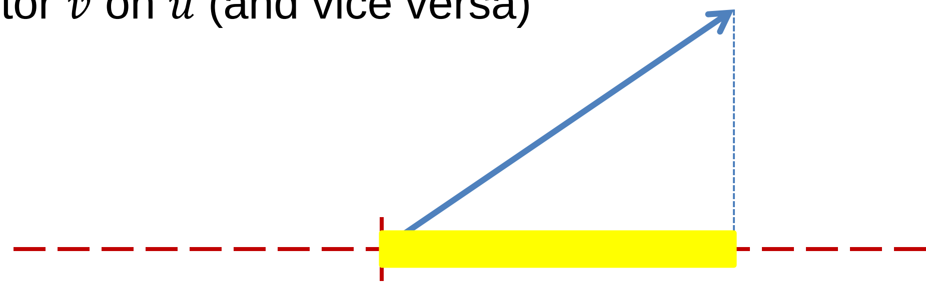
$$\begin{pmatrix} \\ \\ A \\ \\ \end{pmatrix}$$

A_{ij} = rating of **j**-th product by the **i**-th customer

Find subsets of movies that capture the behavior or the customers

Linear algebra

- We assume that vectors are column vectors.
- We use v^T for the transpose of vector v (row vector)
- Dot product: $u^T v$ $(1 \times n, n \times 1 \;\rightarrow\; 1 \times 1)$
  - The dot product is the projection of vector v on u (and vice versa)
- $[1, 2, 3] \begin{bmatrix} 4 \\ 1 \\ 2 \end{bmatrix} = 12$
- $u^T v = \|v\| \|u\| \cos(u, v)$



  - If $||u|| \;=\; 1$ (unit vector) then $u^T v$ is the projection length of v on u
- $[-1, 2, 3] \begin{bmatrix} 4 \\ -1 \\ 2 \end{bmatrix} = 0$ orthogonal vectors
  - Orthonormal vectors: two unit vectors that are orthogonal

Matrices

- An n×m matrix A is a collection of n row vectors and m column vectors

$$A = \begin{bmatrix} | & | & | \\ a_1 & a_2 & a_3 \\ | & | & | \end{bmatrix} \qquad A = \begin{bmatrix} - & \alpha_1^T & - \\ - & \alpha_2^T & - \\ - & \alpha_3^T & - \end{bmatrix}$$

- Matrix-vector multiplication
  - Right multiplication Au: projection of u onto the row vectors of A, or projection of row vectors of A onto u.
  - Left-multiplication $u^T A$: projection of u onto the column vectors of A, or projection of column vectors of A onto u
- Example:

$$[1,2,3] \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} = [1,2]$$

Rank

- Row space of A: The set of vectors that can be written as a linear combination of the rows of A
  - All vectors of the form $v = u^T A$

- Column space of A: The set of vectors that can be written as a linear combination of the columns of A
  - All vectors of the form $v = Au$.

- Rank of A: the number of linearly independent row (or column) vectors
  - These vectors define a basis for the row (or column) space of A

Rank-1 matrices

- In a rank-1 matrix, all columns (or rows) are multiples of the same column (or row) vector

$$A = \begin{bmatrix} 1 & 2 & -1 \\ 2 & 4 & -2 \\ 3 & 6 & -3 \end{bmatrix}$$

- All rows are multiples of $r = [1,2,-1]$
- All columns are multiples of $c = \ [1,2,3]^T$
- External product: uv^T $(n{\times}1\,,1{\times}m\ \rightarrow\ n{\times}m)$
  - The resulting $n{\times}m$ has rank 1: all rows (or columns) are linearly dependent
  - $A = rc^T$

Eigenvectors

- (Right) Eigenvector of matrix A: a vector v such that $Av = \lambda v$
- λ: eigenvalue of eigenvector v

- A square matrix A of rank r, has r orthonormal eigenvectors $u_1, u_2, \ldots, u_r$ with eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_r$.
- Eigenvectors define an orthonormal basis for the column space of A

Singular Value Decomposition

$$A = U \quad \Sigma \quad V^T = [u_1, u_2, \cdots, u_r] \begin{bmatrix} \sigma_1 & & & 0 \\ & \sigma_2 & & \\ & & \ddots & \\ 0 & & & \sigma_r \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_r^T \end{bmatrix}$$

$$[n \times m] = [n \times r] \quad [r \times r] \quad [r \times m]$$

r: rank of matrix A

- $\sigma_1, \geq \sigma_2 \geq \cdots \geq \sigma_r$: singular values of matrix A (also, the square roots of eigenvalues of AA^T and $A^T A$)
- $u_1, u_2, \ldots, u_r$: left singular vectors of A (also eigenvectors of AA^T)
- $v_1, v_2, \ldots, v_r$: right singular vectors of A (also, eigenvectors of $A^T A$)

$$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$$

Symmetric matrices

- Special case: A is symmetric positive definite matrix

$$A = \lambda_1 u_1 u_1^T + \lambda_2 u_2 u_2^T + \cdots + \lambda_r u_r u_r^T$$

- $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r \geq 0$: Eigenvalues of A
- $u_1, u_2, \ldots, u_r$: Eigenvectors of A

Singular Value Decomposition

- The left singular vectors are an orthonormal basis for the row space of A.
- The right singular vectors are an orthonormal basis for the column space of A.
- If A has rank r, then A can be written as the sum of r rank-1 matrices
- There are r "linear components" (trends) in A.
  - Linear trend: the tendency of the row vectors of A to align with vector **v**
  - Strength of the i-th linear trend: $||A\boldsymbol{v_i}|| = \boldsymbol{\sigma_i}$

An (extreme) example

- Document-term matrix
  - Blue and Red rows (colums) are linearly dependent

A = 

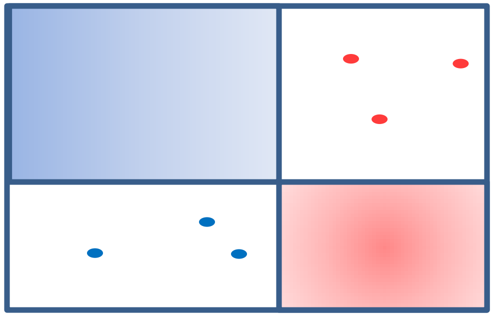
- There are two prototype documents (vectors of words): blue and red
  - To describe the data is enough to describe the two prototypes, and the projection weights for each row

- A is a rank-2 matrix

$$A = [w_1, w_2]\begin{bmatrix} d_1^T \\ d_2^T \end{bmatrix}$$

An (more realistic) example

- Document-term matrix

A = 

- There are two prototype documents and words but they are noisy
  - We now have more than two singular vectors, but the strongest ones are still about the two types.
  - By keeping the two strongest singular vectors we obtain most of the information in the data.
    - This is a rank-2 approximation of the matrix A

Rank-**k** approximations (A_k)

$$\begin{pmatrix} \\ \\ A_k \\ \\ \\ \end{pmatrix} = \begin{pmatrix} \\ \\ U_k \\ \\ \\ \end{pmatrix} \cdot \begin{pmatrix} \\ \Sigma_k \\ \\ \end{pmatrix} \cdot \begin{pmatrix} \\ V_k^T \\ \\ \end{pmatrix}$$

n x d **n x k** **k x k** **k x d**

U_k (V_k): orthogonal matrix containing the top ***k*** left (right) singular vectors of **A**.
Σ_k: diagonal matrix containing the top ***k*** singular values of **A**

A_k is an approximation of **A**

A_k is the **best** approximation of **A**

SVD as an optimization

- The rank-k approximation matrix A_k produced by the top-k singular vectors of A minimizes the Frobenious norm of the difference with the matrix A

$$A_k = \arg \max_{B:rank(B)=k} ||A - B||_F^2$$

$$||A - B||_F^2 = \sum_{i,j} \left(A_{ij} - B_{ij}\right)^2$$

What does this mean?

- We can project the row (and column) vectors of the matrix A into a k-dimensional space and preserve most of the information
- (Ideally) The k dimensions reveal latent features/aspects/topics of the term (document) space.
- (Ideally) The A_k approximation of matrix A, contains all the useful information, and what is discarded is noise

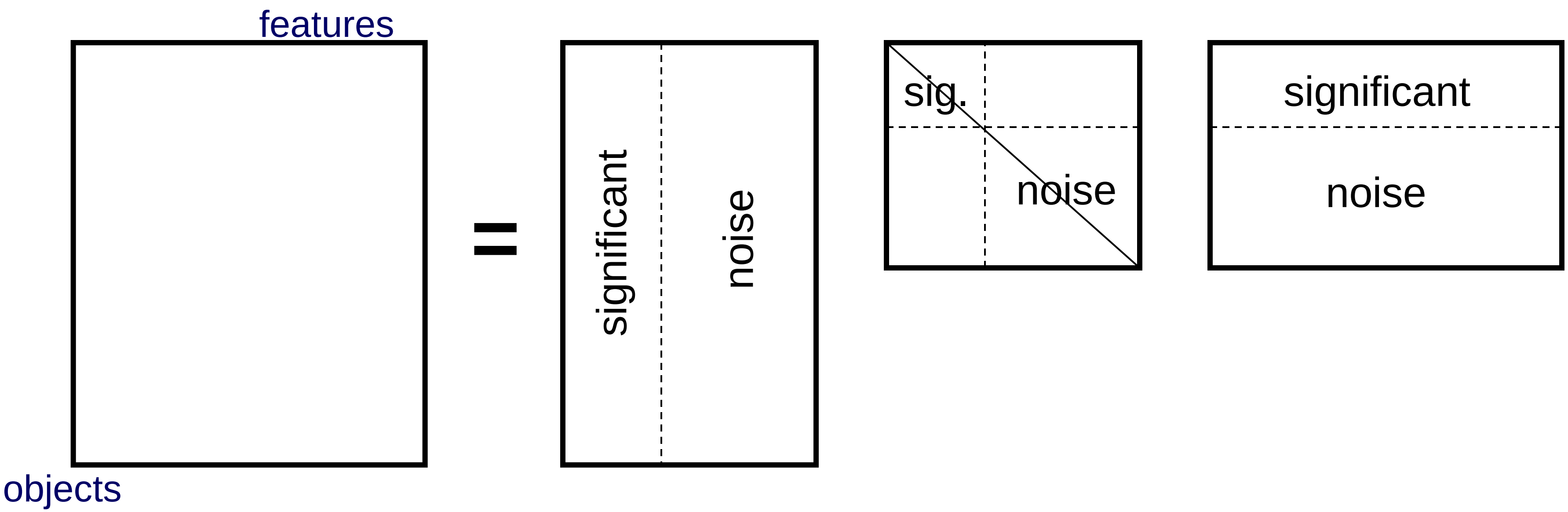
Latent factor model

- Rows (columns) are linear combinations of k latent factors
  - E.g., in our extreme document example there are two factors
- Some noise is added to this rank-k matrix resulting in higher rank

- SVD retrieves the latent factors (hopefully).

SVD and Rank-**k** approximations

$$\mathbf{A} \quad = \quad \mathbf{U} \quad \mathbf{\Sigma} \quad \mathbf{V}^{T}$$



Application: Recommender systems

- Data: Users rating movies
  - Sparse and often noisy
- Assumption: There are k basic user profiles, and each user is a linear combination of these profiles
  - E.g., action, comedy, drama, romance
  - Each user is a weighted cobination of these profiles
  - The "true" matrix has rank k
- What we observe is a noisy, and incomplete version of this matrix $\tilde{A}$
  - The rank-k approximation $\tilde{A}_k$ is provably close to A_k
- Algorithm: compute $\tilde{A}_k$ and predict for user u and movie m, the value $\tilde{A}_k[m, u]$.
  - Model-based collaborative filtering

SVD and PCA

- PCA is a special case of SVD on the centered covariance matrix.

Covariance matrix

- Goal: reduce the dimensionality while preserving the "information in the data"
- Information in the data: variability in the data
  - We measure variability using the covariance matrix.
  - Sample covariance of variables X and Y

$$\sum_i (x_i - \mu_X)^T (y_i - \mu_Y)$$

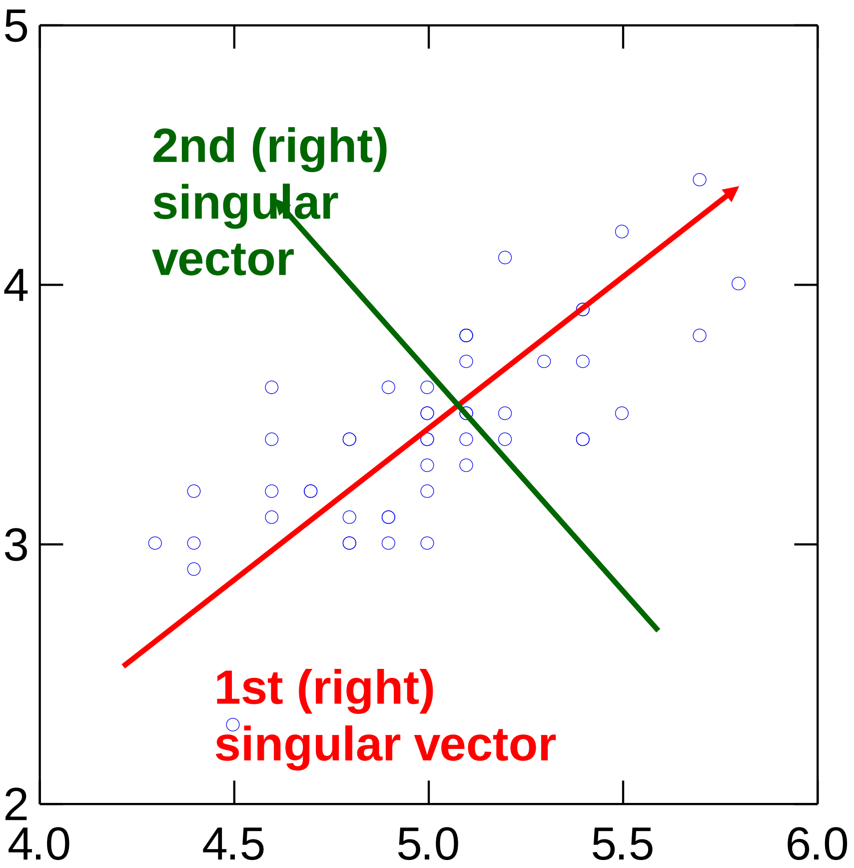
- Given matrix A, remove the mean of each column from the column vectors to get the centered matrix C
- The matrix $V = C^T C$ is the covariance matrix of the row vectors of A.

PCA: Principal Component Analysis

- We will project the rows of matrix A into a new set of attributes (dimensions) such that:
  - The attributes have zero covariance to each other (they are orthogonal)
  - Each attribute captures the most remaining variance in the data, while orthogonal to the existing attributes
    - The first attribute should capture the most variance in the data

- For matrix C, the variance of the rows of C when projected to vector x is given by $\sigma^2 = ||Cx||^2$
  - The right singular vector of C maximizes σ^2!

PCA



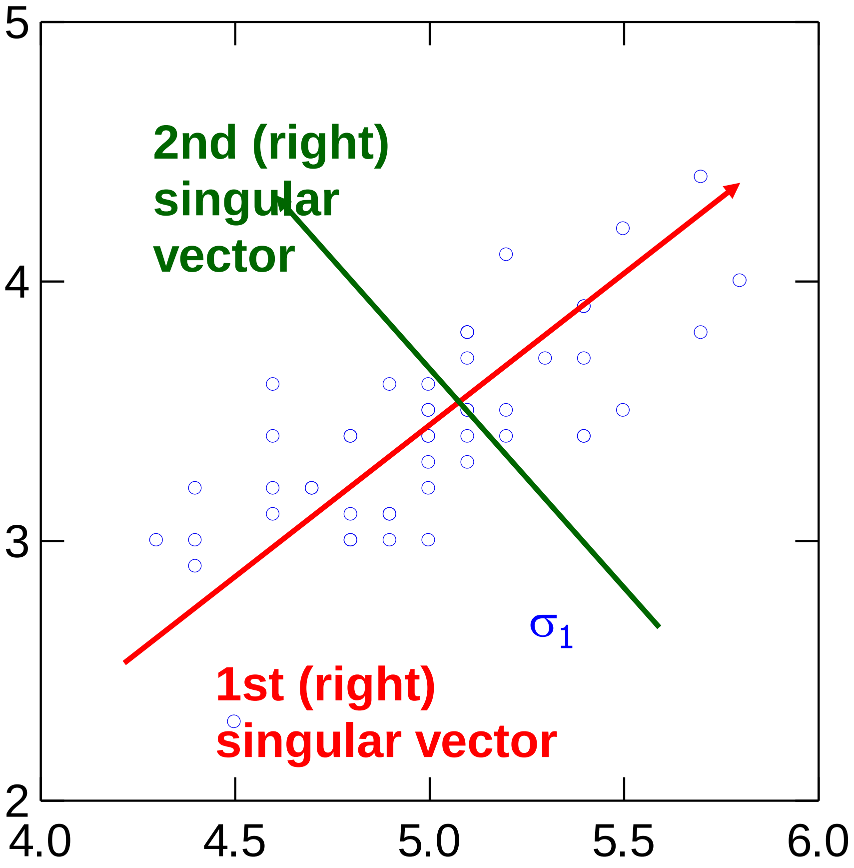
Input: **2-d** dimensional points

Output:

<u>1st (right) singular vector:</u>
direction of maximal variance,

<u>2nd (right) singular vector:</u>
direction of maximal variance, after removing the projection of the data along the first singular vector.

Singular values



σ_1: measures how much of the data variance is explained by the first singular vector.

σ_2: measures how much of the data variance is explained by the second singular vector.

Singular values tell us something about the variance

- The variance in the direction of the **k**-th principal component is given by the corresponding singular value σ_k^2

- Singular values can be used to estimate how many components to keep

- ***Rule of thumb:*** keep enough to explain ***85%*** of the variation:

$$\frac{\sum_{j=1}^{k} \sigma_j^2}{\sum_{j=1}^{n} \sigma_j^2} \approx 0.85$$

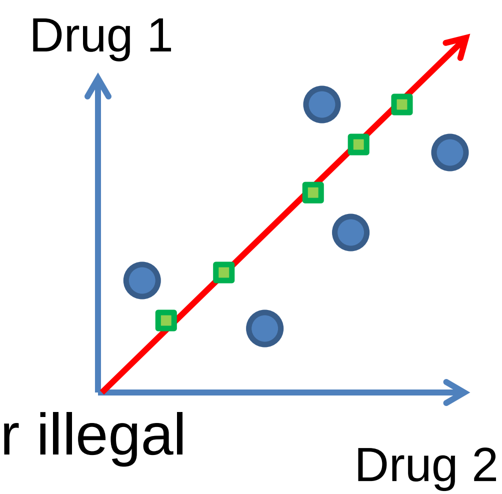
Example

$$
\begin{array}{ccc}
 & \text{drugs} & \\
A = & \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} & \text{students} \\
 & \text{legal} \quad\quad \text{illegal} &
\end{array}
$$

a_{ij}: usage of student i of drug j

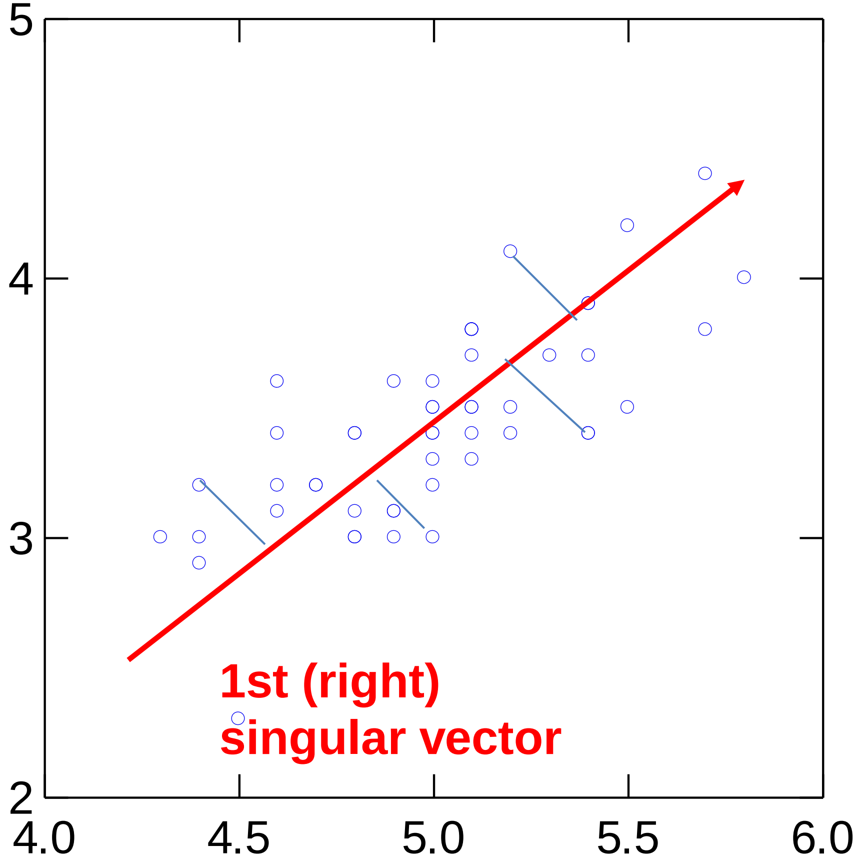
$$A = U\Sigma V^T$$

- First right singular vector v_1
  - More or less same weight to all drugs
  - Discriminates heavy from light users
- Second right singular vector
  - Positive values for legal drugs, negative for illegal



Another property of PCA/SVD

- The chosen vectors are such that minimize the sum of square differences between the data vectors and the low-dimensional projections



Application

- Latent Semantic Indexing (LSI):
  - Apply PCA on the document-term matrix, and index the k-dimensional vectors
  - When a query comes, project it onto the k-dimensional space and compute cosine similarity in this space
  - Principal components capture main topics, and enrich the document representation

SVD is “**the Rolls-Royce and the Swiss Army Knife of Numerical Linear Algebra.”***

*Dianne O’Leary, MMDS ’06