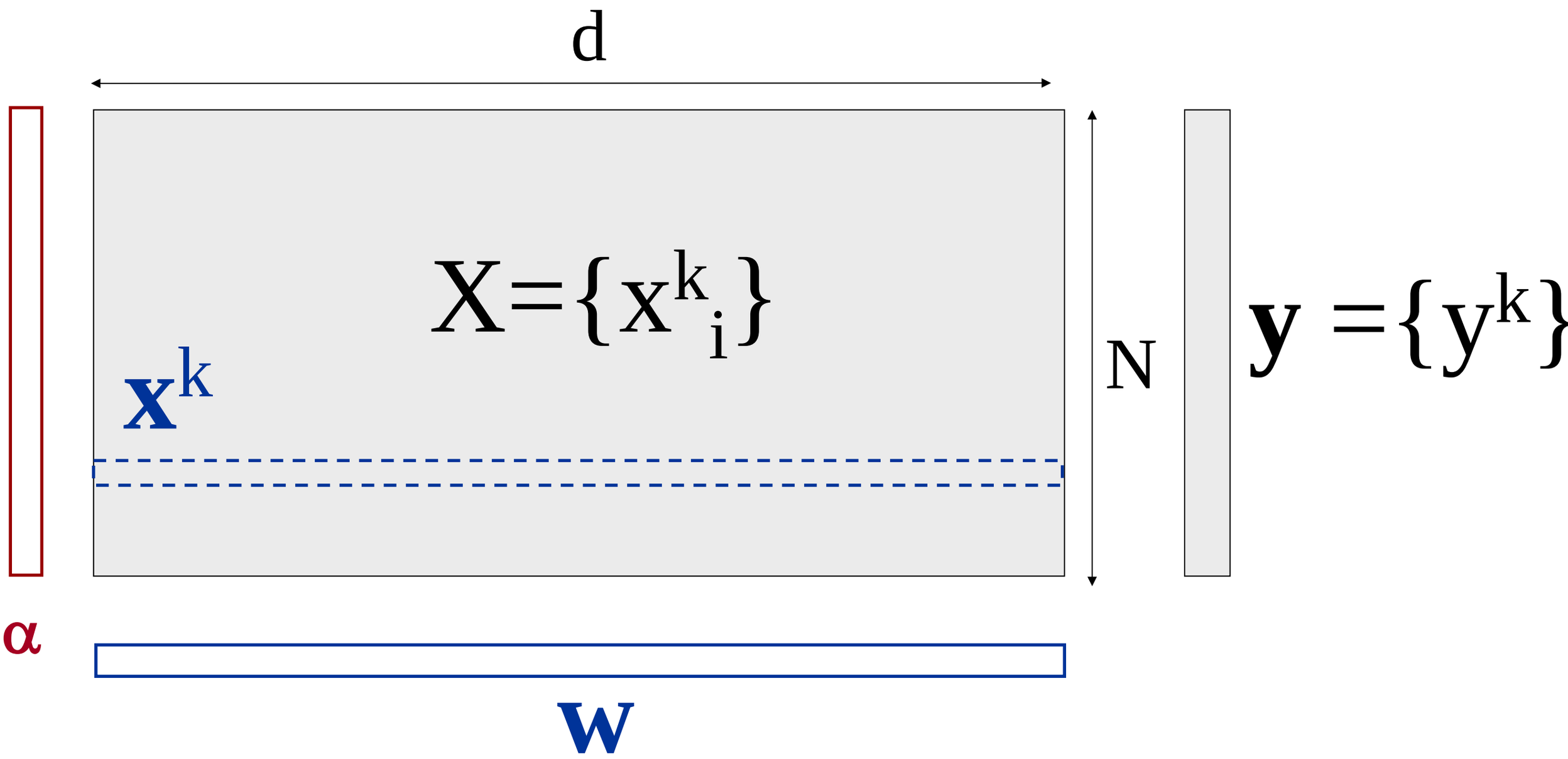
Dimensionality Reduction by PCA

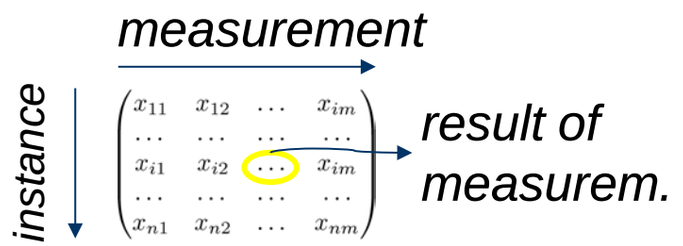
Pattern Matrix



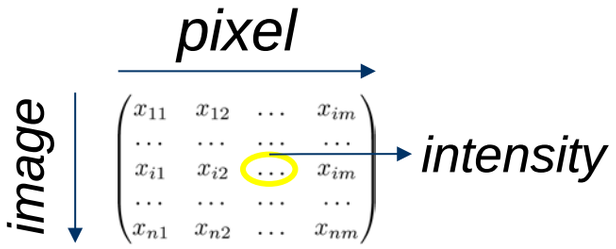
Examples: Pattern Matrices

$$\mathbf{X} \in \mathbb{R}^{n \times m}$$

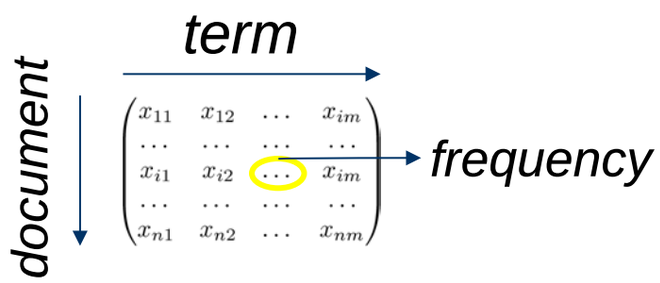
- **Measurement** vectors
  - i: instance number, e.g. a house
  - j: measurement, e.g. the area of a house



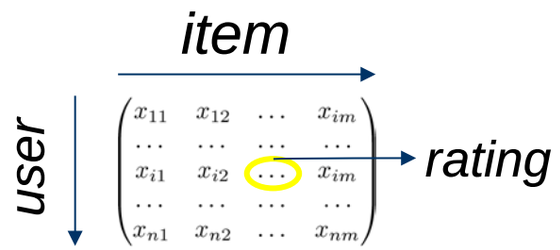
- **Digital images** as gray-scale **vectors**
  - i: image number
  - j: pixel value at location $j=(k,l)$



- **Text documents** in **bag-of-words** representation
  - i: document number
  - j: term (word or phrase) in a vocabulary



- **User rating** data
  - i: user number
  - j: item (book, movie)



Document-Term Matrix

D = Document collection

W = Lexicon/Vocabulary

Texas Instruments said it has developed the first 32-bit computer chip designed specifically for artificial ***intelligence*** *applications [...]*

d_i

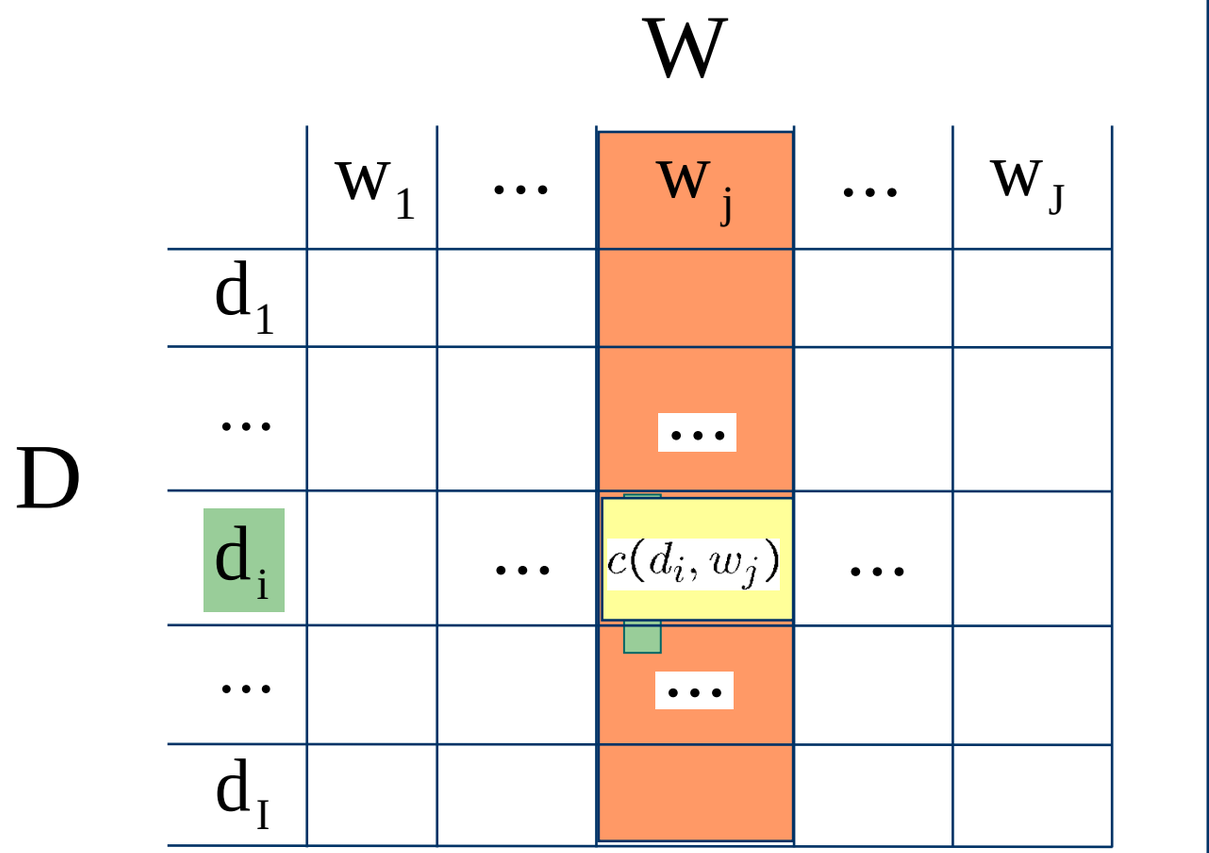
artifact artificial intelligence interest

d_i = … 0 1 … 2 0 … t

X

term weighting

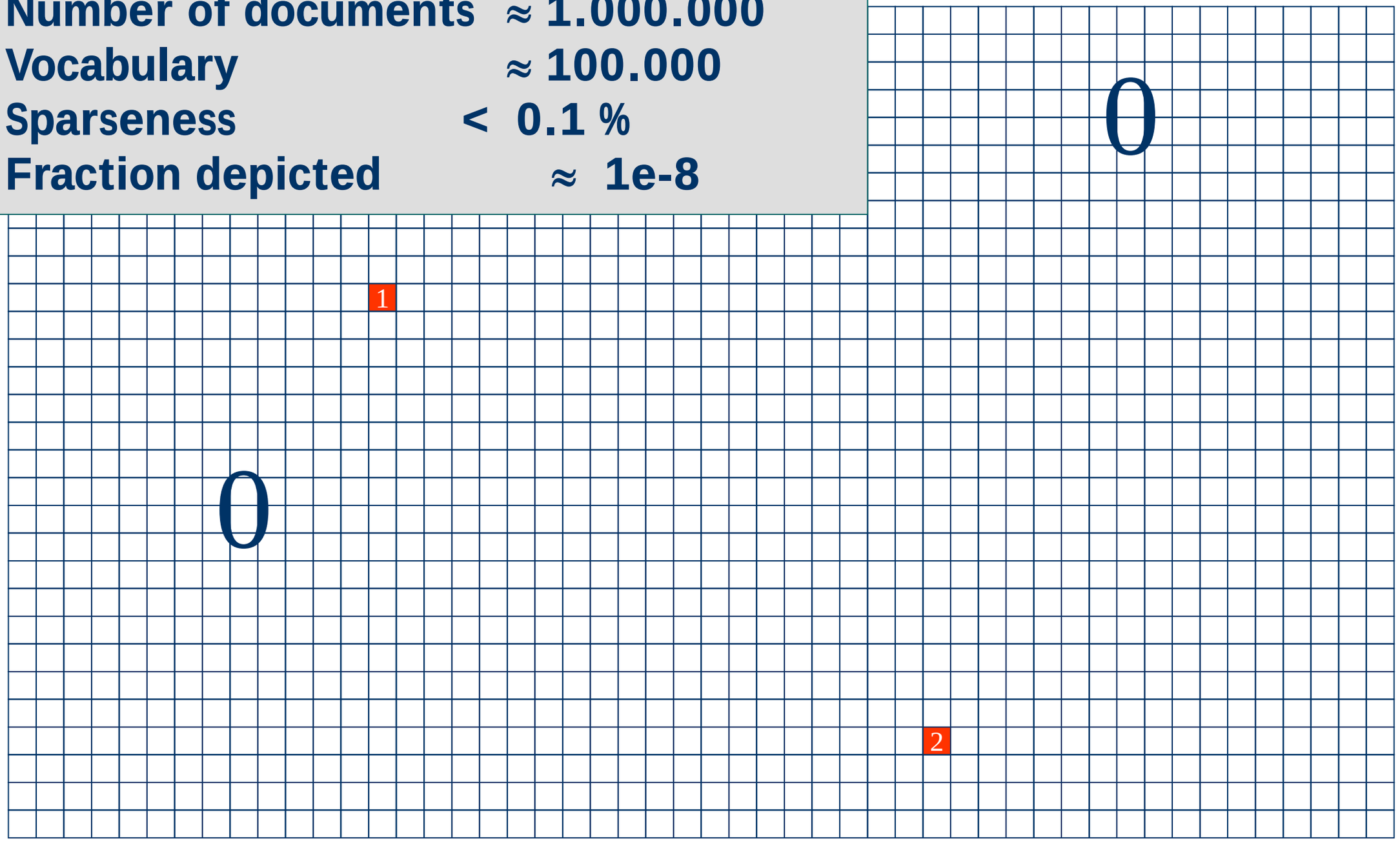
Document-Term Matrix



A 100 Million^{ths} of a Typical Document-term Matrix

Typical:

- **Number of documents** ≈ **1.000.000**
- **Vocabulary** ≈ **100.000**
- **Sparseness** < **0.1 %**
- **Fraction depicted** ≈ **1e-8**



Vocabulary Mismatch & ~~Robustness~~

Document-Term Matrix

D = Document collection

W = Lexicon/Vocabulary

intelligence w_j

Texas Instruments said it has developed the first 32-bit computer chip designed specifically for artificial ***intelligence*** *applications […]*

d_i

artifact artificial intelligence interest

d_i = … 0 1 … 2 0 … t

X

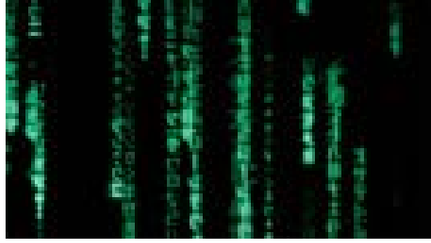
term weighting

Document-Term Matrix

D	w_1	…	w_j	…	w_J
d_1					
…			…		
d_i		…	$c(d_i, w_j)$	…	
…			…		
d_I					

Clustering rows?

Clustering columns?



Latent Structure

- Given a matrix that “encodes” data …
- Potential problems
  - too large
  - too complicated
  - missing entries
  - noisy entries
  - lack of structure
  - …

$$\mathbf{A} = \begin{pmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1m} \\ \dots & \dots & \dots & \dots & \dots \\ a_{i1} & \dots & a_{ij} & \dots & a_{im} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & \dots & a_{nj} & \dots & a_{nm} \end{pmatrix}$$

- Is there a **simpler** way to **explain** entries?
- There might be a **latent structure** underlying the data.
- How can we “find” or “reveal” this structure?

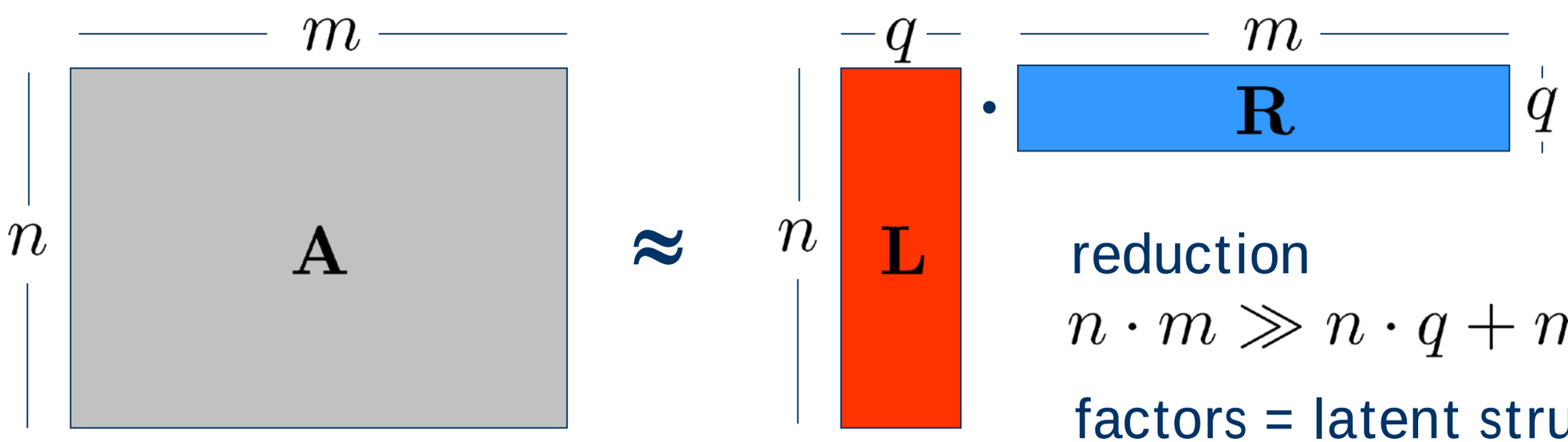
Matrix Decomposition

- Common approach: approximately **factorize** matrix

$$\mathbf{A} \approx \hat{\mathbf{A}} = \mathbf{L} \cdot \mathbf{R}$$

approximation left factor right factor

- Factors are typically constrained to be "**thin**"



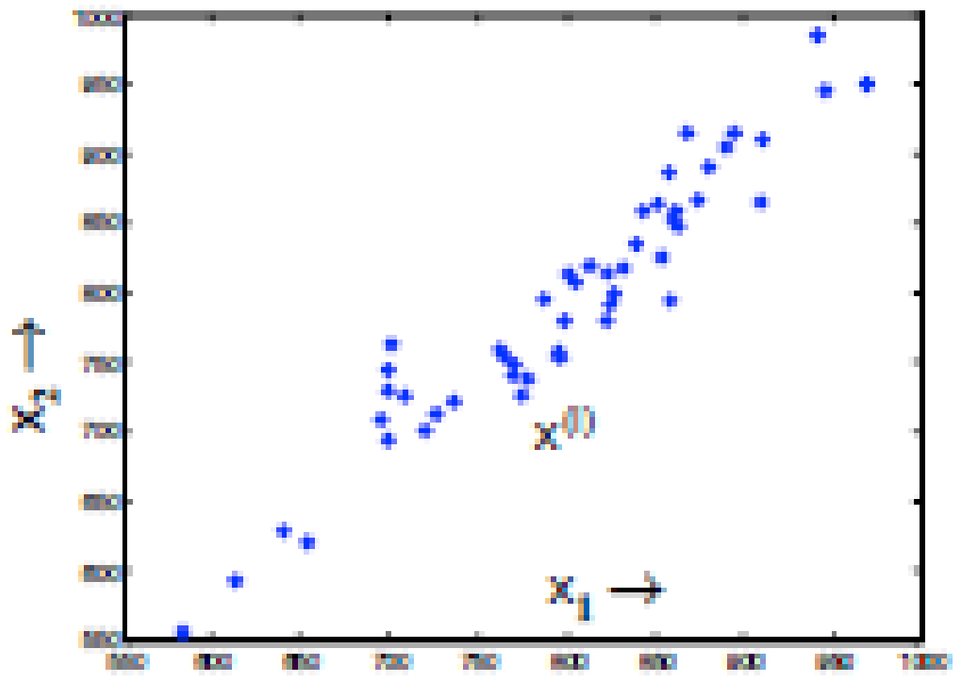
reduction

$$n \cdot m \gg n \cdot q + m \cdot q$$

factors = latent structure (?)

Dimensionality reduction

- Ex: data with two real values $[x_1,x_2]$
- We'd like to describe each point using only one value $[z_1]$
- We'll communicate a "model" to convert: $[x_1,x_2] \sim f(z_1)$

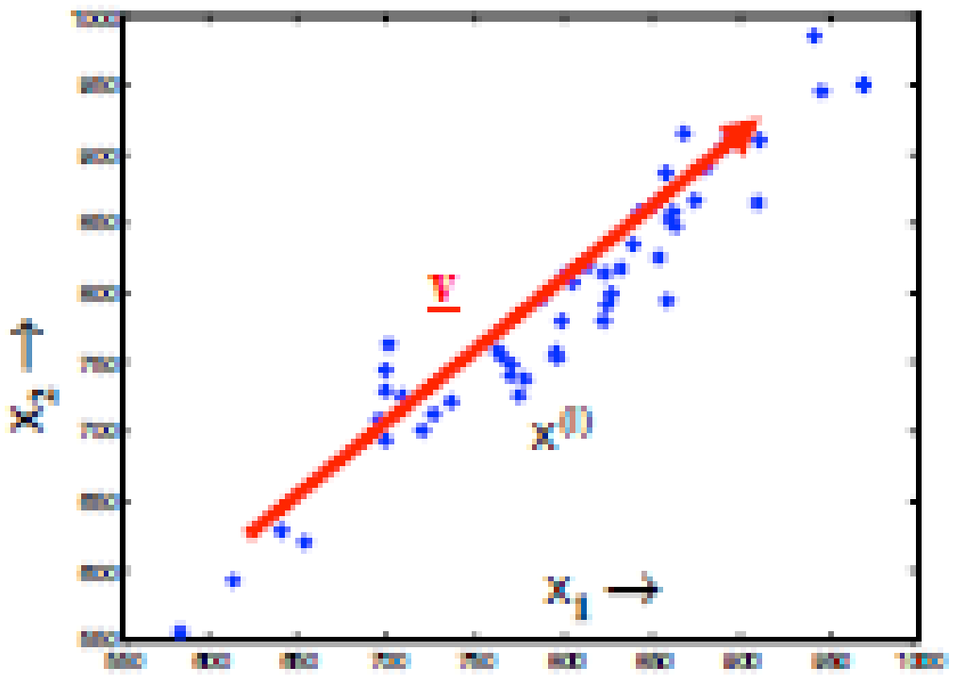


Slides by Alexander Ihler

Dimensionality reduction

- Ex: data with two real values $[x_1,x_2]$
- We'd like to describe each point using only one value $[z_1]$
- We'll communicate a "model" to convert: $[x_1,x_2] \sim f(z_1)$

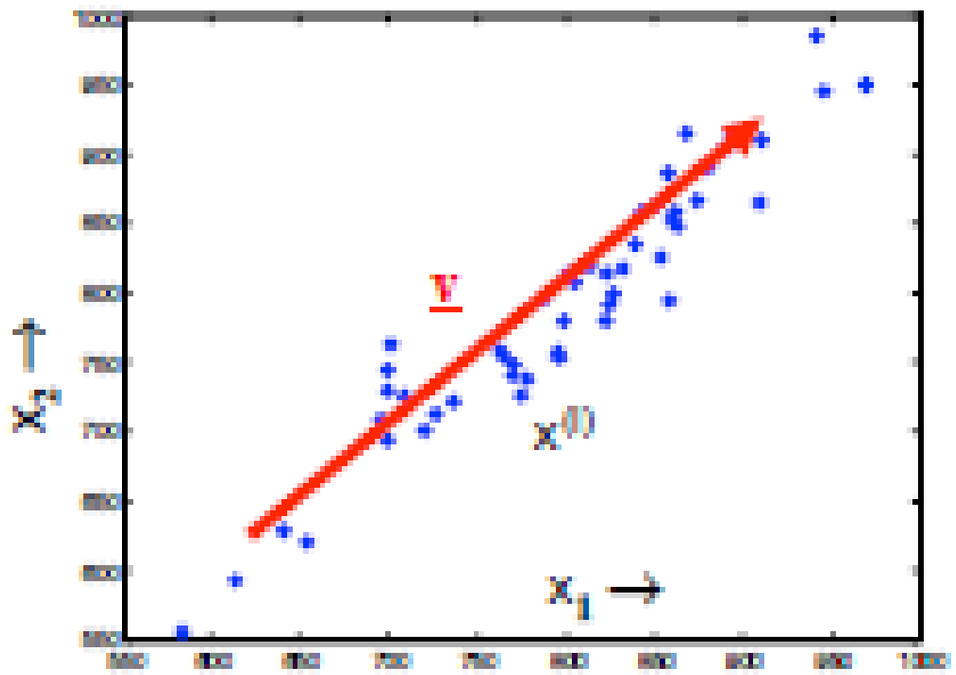
- Ex: linear function f(z): $[x_1,x_2] = z * \underline{v} = z * [v_1,v_2]$
- $\underline{v}$ is the same for all data points (communicate once)
- z tells us the closest point on v to the original point $[x_1,x_2]$

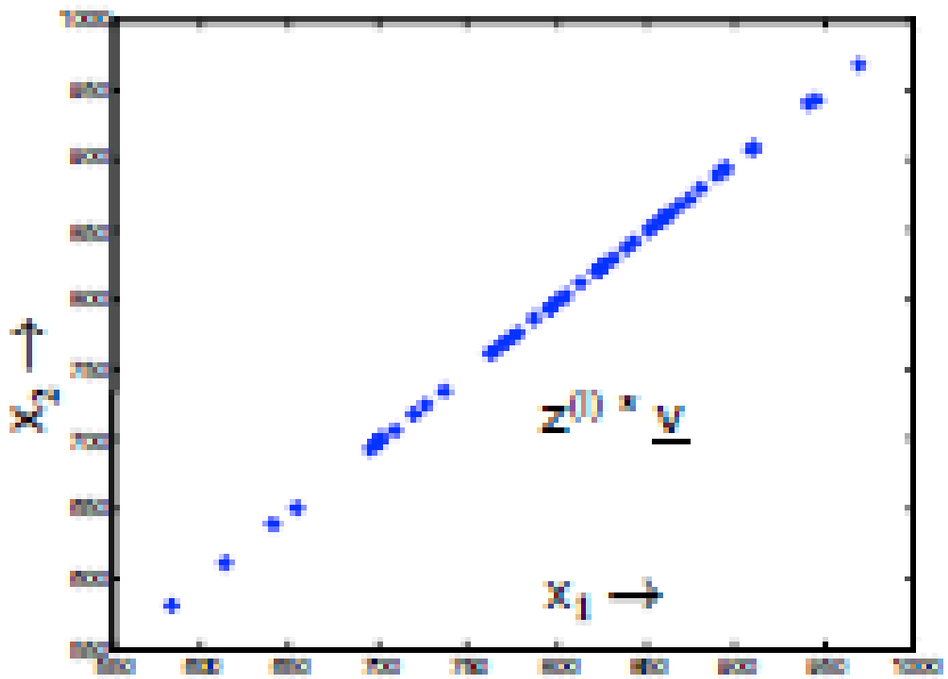


Dimensionality reduction

- Ex: data with two real values $[x_1,x_2]$
- We'd like to describe each point using only one value $[z_1]$
- We'll communicate a "model" to convert: $[x_1,x_2] \sim f(z_1)$

- Ex: linear function f(z): $[x_1,x_2] = z * \underline{v} = z * [v_1,v_2]$
- $\underline{v}$ is the same for all data points (communicate once)
- z tells us the closest point on v to the original point $[x_1,x_2]$



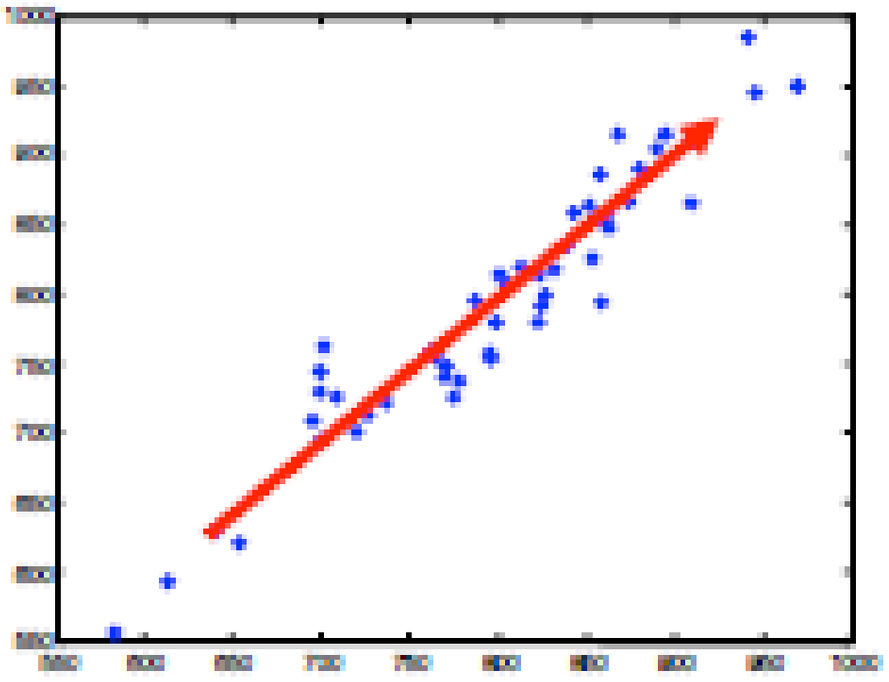


Principal Components Analysis

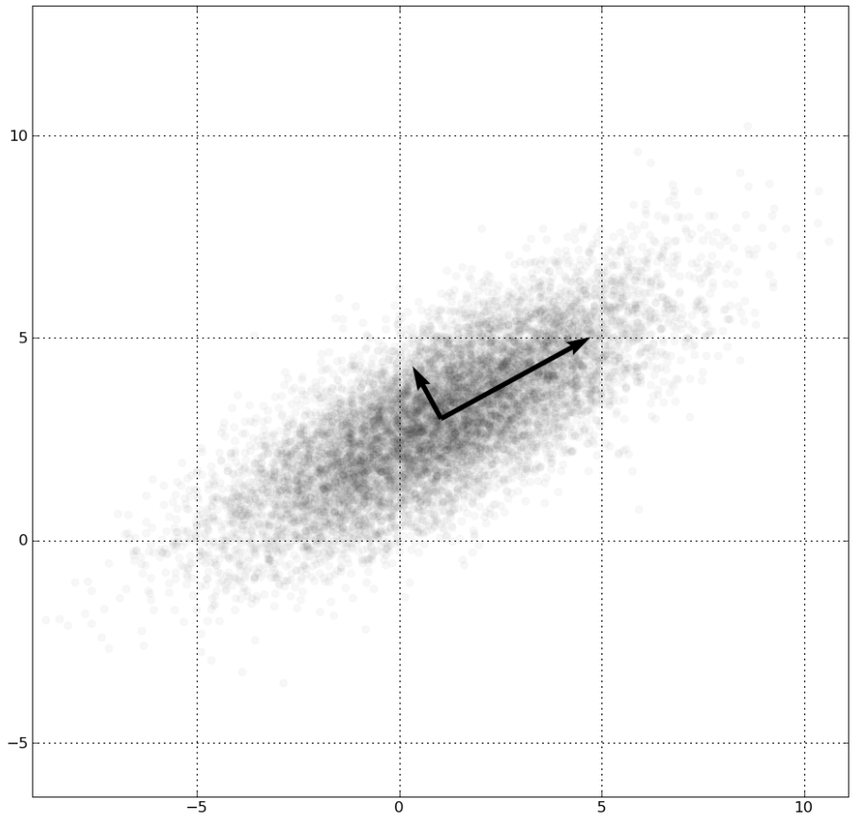
- What is the vector that would most closely reconstruct X?

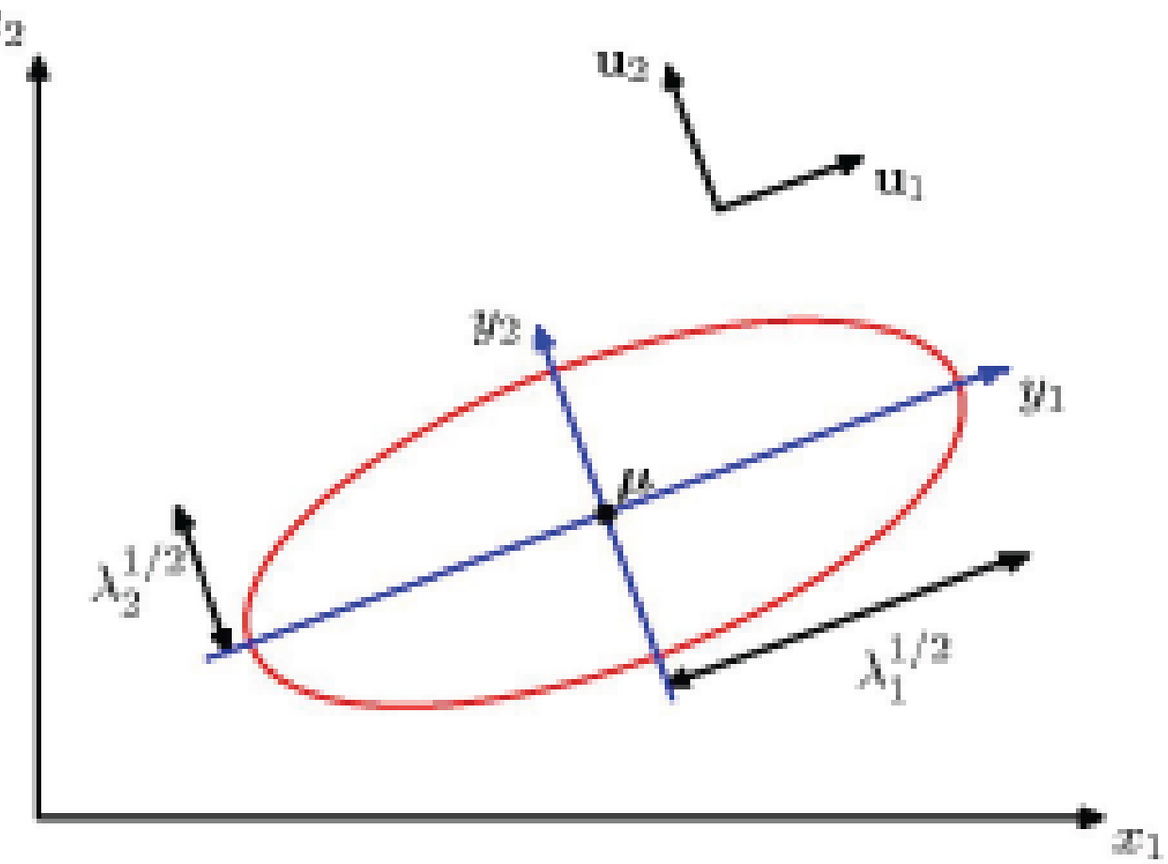
$$\min_{a,v} \sum_i (x^{(i)} - a^{(i)} v)^2$$

  - Given v: $a^{(i)}$ is the projection of each point $x^{(i)}$ onto v
  - v chosen to minimize the residual variance
  - Equivalently, v is the direction of maximum variance
  - Extensions: best two dimensions: xi= ai*v + bi*w + m



Eigenvectors





PCA

Given pattern matrix X,

1. Subtract mean from each point
2. (sometimes) scale each dimension by its variance
3. Compute covariance matrix C=$X^T X$
4. Compute k largest eigenvectors of C

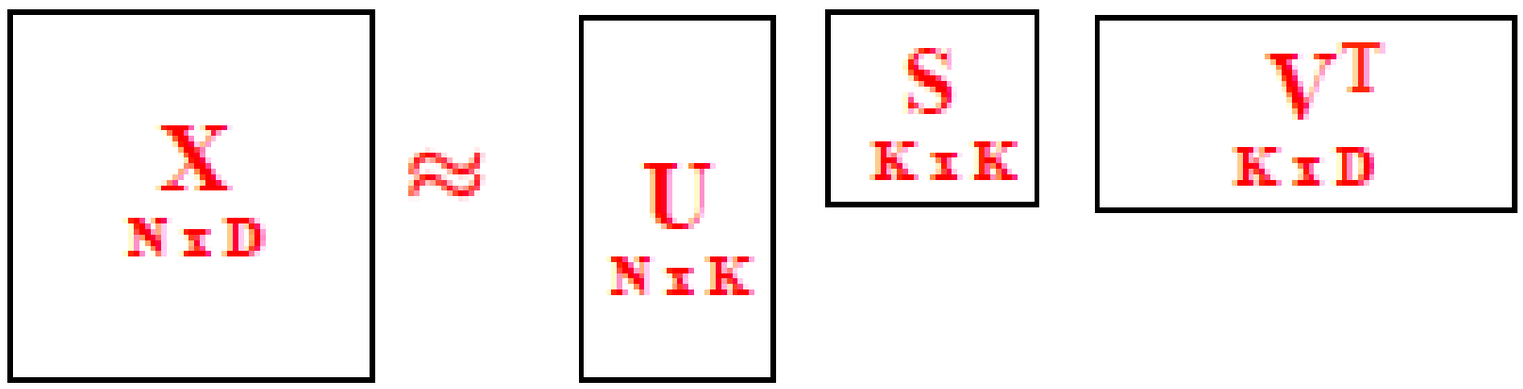
   C = VDV^T

Singular Value Decomposition

- Alternative method to calculate (still subtract mean 1st)
- Decompose $X = U\ S\ V^T$
  - Orthogonal: $X^T X = V\ S\ S\ V^T = V\ D\ V^T$
  - $X\ X^T = U\ S\ S\ U^T = U\ D\ U^T$

- U*S matrix provides coefficients
  - Example $x_i = U_{i,1}\ S_{11}\ v_1 + U_{i,2}\ S_{22}\ v_2 + \ldots$

- Gives the least-squares approximation to X of this form



Glorious SVD

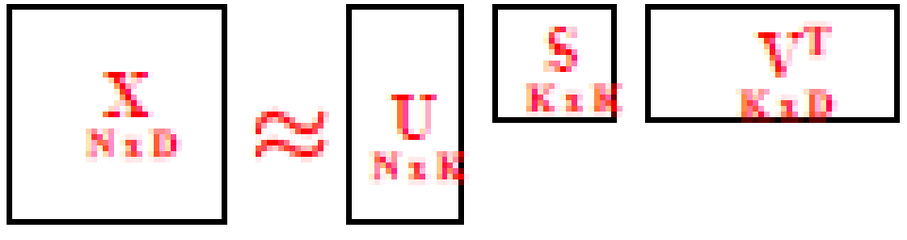
$$X = USV^T$$

- XX^T and X^TX share the same eigenvalues
- Even better: their eigenvectors are related
  - Xv_i is an eigenvector of XX^T

Collaborative Filtering (Netflix)

From Y. Koren of BellKor team

movies \ users	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

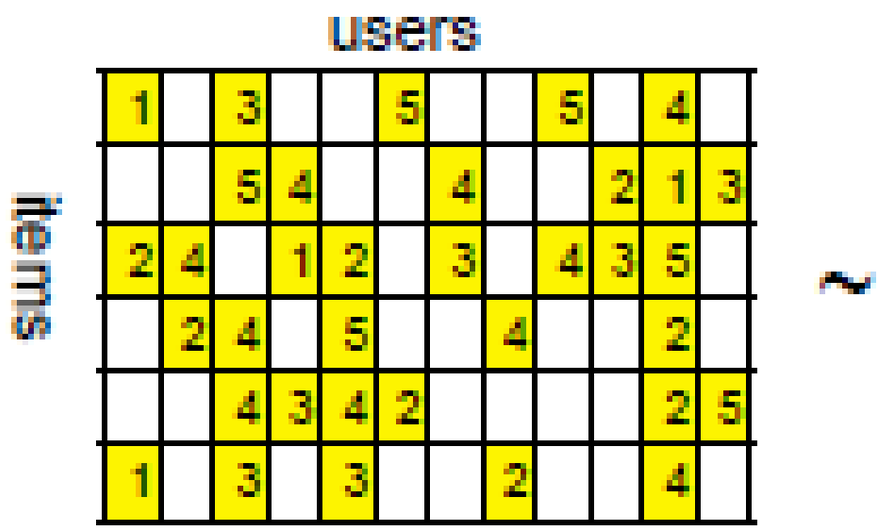


From Y. Koren of BellKor team

Latent Space Models

Model ratings matrix as "user" and "movie" positions

Infer values from known ratings



Extrapolate to unranked

~

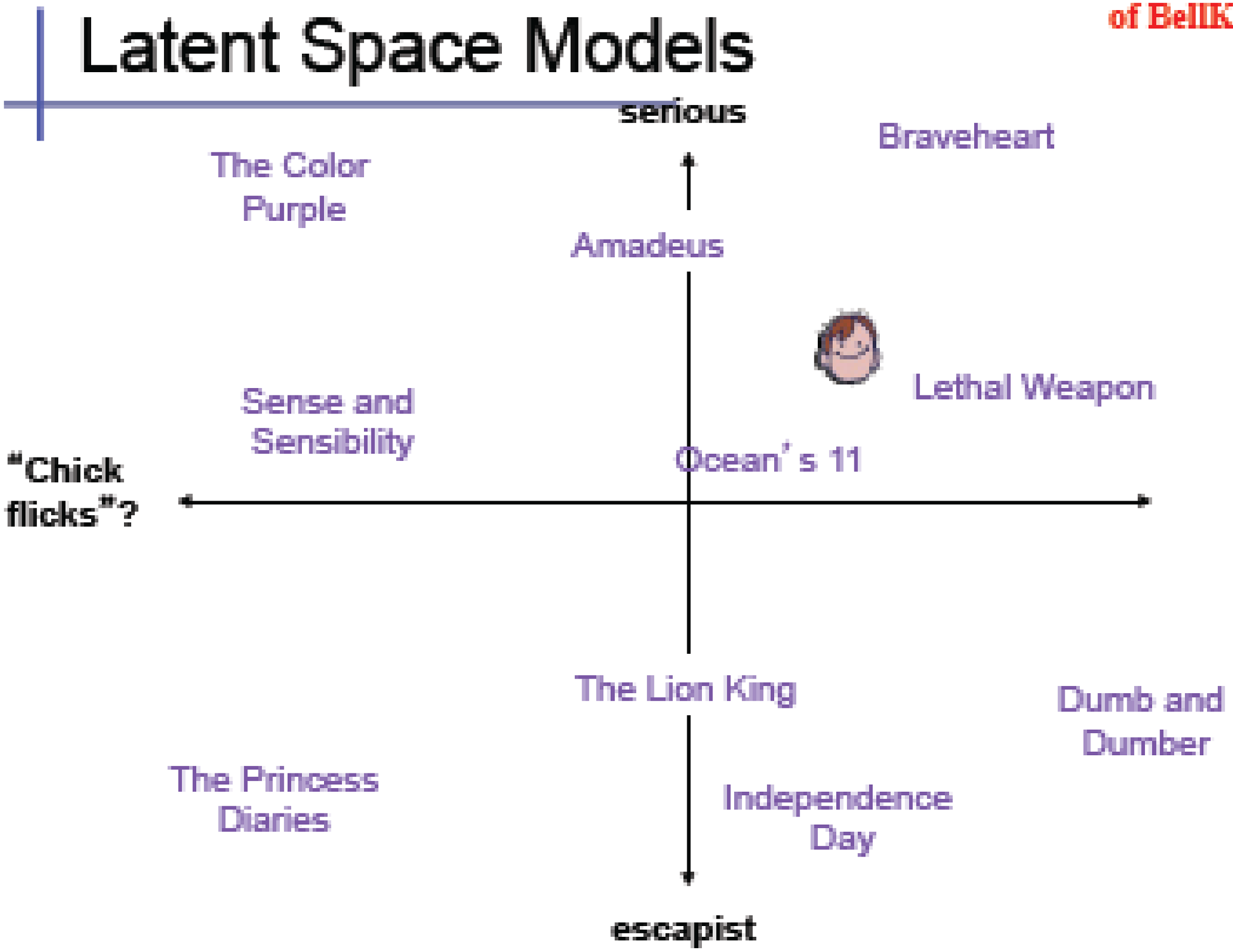
items			
	.1	-.4	.2
	-.5	.6	.5
	-.2	.3	.5
	1.1	2.1	.3
	-.7	2.1	-2
	-1	.7	.3

•

users											
1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

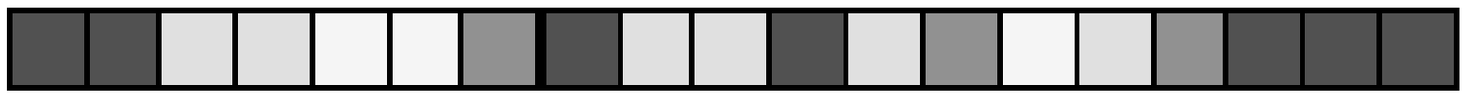
From Y. Koren
of BellKor team

Latent Space Models



“Eigen-faces”

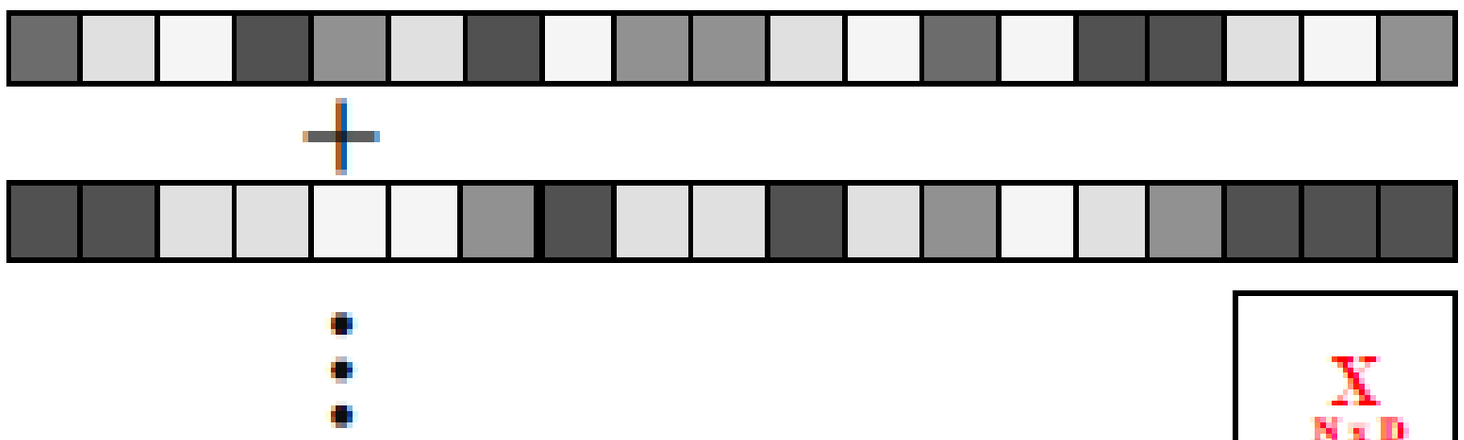
- “Eigen-X” = represent X using PCA
- Ex: Viola Jones data set
  - 24x24 images of faces = 576 dimensional measurements



$$X_{N \times D}$$

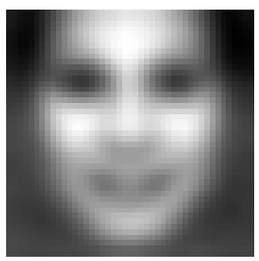
“Eigen-faces”

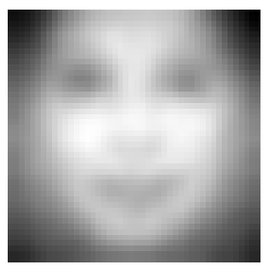
- “Eigen-X” = represent X using PCA
- Ex: Viola Jones data set
  - 24x24 images of faces = 576 dimensional measurements
  - Take first K PCA components

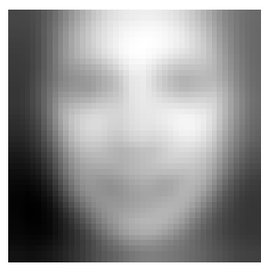


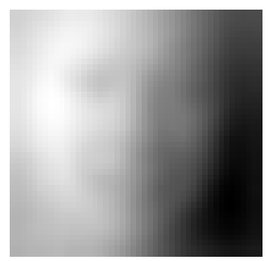
“Eigen-faces”

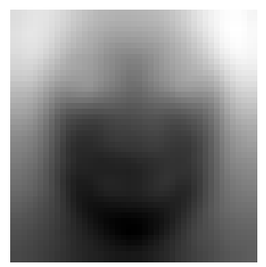
- “Eigen-X” = represent X using PCA
- Ex: Viola Jones data set
  - 24x24 images of faces = 576 dimensional measurements
  - Take first K PCA components


Mean

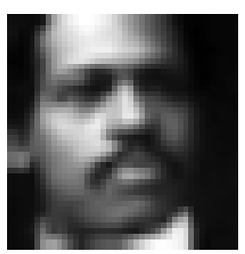

Dir 1

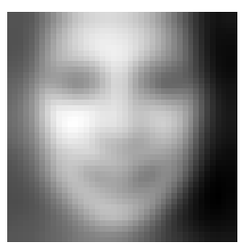

Dir 2

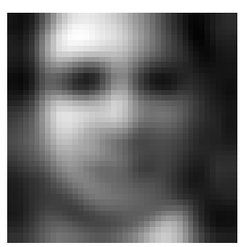

Dir 3

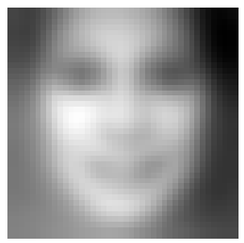

Dir 4

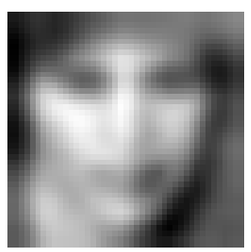
…




K=4


K=50


K=4


K=50