Singular Value Decomposition (SVD)

Y. Kotidis

Athens University of Economics and Business

Acknowledgments

- Some examples adapted from Christos Faloutsos’ class material (CMU) and also from the following work:
  - *Quantifiable Data Mining Using Ratio Rules.*

    F. Korn, A. Labrinidis, Y. Kotidis, C. Faloutsos. The VLDB Journal, Volume 8(3+4), February 2000.

    (available at http://pages.aueb.gr/users/kotidis/Publications/index.html)

Applications

- Find similar "*concepts*" in large datasets
  - Basket analysis
    - Explore customer-product relationships
    - Find similar customers, products
  - Document indexing & retrieval
- Dimensionality reduction/feature selection
  - Reduce data size by projecting items into a lower-dimensionality *concept-space*
- Remove noise, detect outliers, visualization
- Web-link analysis
  - Compute "importance" of web-pages

Basket Data Analysis

Vegetables

Meat

Vegetarians

$$\begin{array}{l|ccccc} & \text{asparagus} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork} \\ \hline \text{Customer 1} & 1 & 1 & 1 & 0 & 0 \\ \text{Customer 2} & 2 & 2 & 2 & 0 & 0 \\ \text{Customer 3} & 1 & 1 & 1 & 0 & 0 \\ \text{Customer 4} & 5 & 5 & 5 & 0 & 0 \\ \text{Customer 5} & 0 & 0 & 0 & 2 & 2 \\ \text{Customer 6} & 0 & 0 & 0 & 3 & 3 \\ \text{Customer 7} & 0 & 0 & 0 & 1 & 1 \end{array}$$

Meat-eaters

Singular Value Decomposition (SVD)

- *Factorization* of matrix A into three matrices

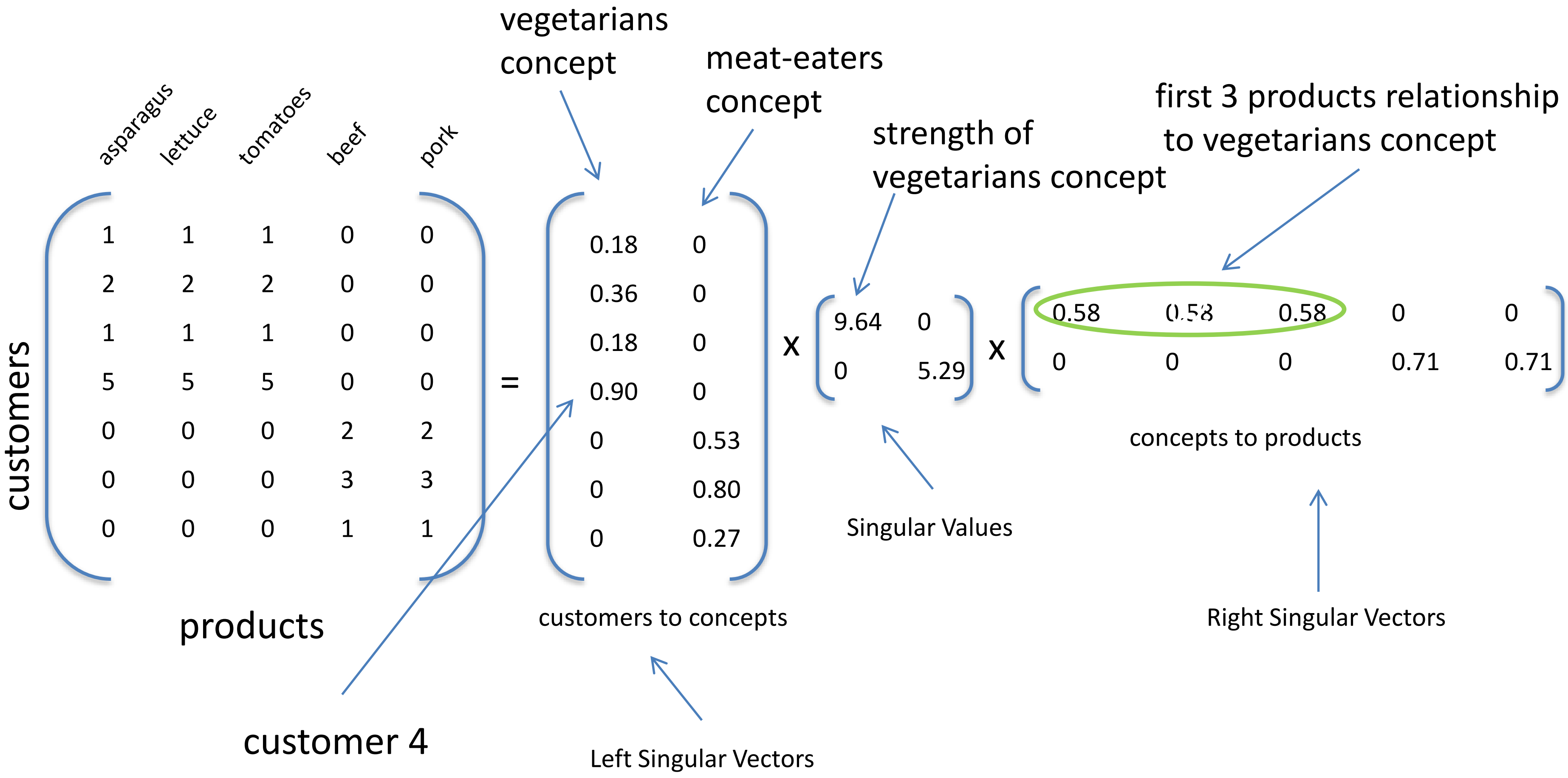
$$\mathbf{A = U\, \Lambda\, V^T}$$

- Such that:
  - A: n x m matrix (e.g. n customers, m products)
  - U: n x r matrix (customers to concepts)
  - Λ: r x r *diagonal* matrix
  - V: m x r matrix (products to concepts)
  - U (resp. V) is a *column-orthonormal* matrix
    - Its columns are mutually orthogonal unit vectors

Στα Ελληνικά…

- SVD - Παραγοντοποίηση Ιδιαζουσών Τιμών
- Concepts: Έννοιες
- Singular value: Ιδιάζουσα τιμή
- Eigen value: Ιδιοτιμή

SVD Example



$$
\begin{array}{c|ccccc}
 & \text{asparagus} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork} \\
\hline
 & 1 & 1 & 1 & 0 & 0 \\
 & 2 & 2 & 2 & 0 & 0 \\
 & 1 & 1 & 1 & 0 & 0 \\
\text{customers} & 5 & 5 & 5 & 0 & 0 \\
 & 0 & 0 & 0 & 2 & 2 \\
 & 0 & 0 & 0 & 3 & 3 \\
 & 0 & 0 & 0 & 1 & 1
\end{array}
=
\begin{pmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{pmatrix}
\times
\begin{pmatrix}
9.64 & 0 \\
0 & 5.29
\end{pmatrix}
\times
\begin{pmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{pmatrix}
$$

products

customers

vegetarians concept

meat-eaters concept

strength of vegetarians concept

first 3 products relationship to vegetarians concept

customer 4

customers to concepts

Left Singular Vectors

Singular Values

concepts to products

Right Singular Vectors

Different Column Ratios

1:2:3 ⟺ 0.27:0.53:0.80

$$\text{customers}\;\begin{array}{ccccc} \text{asparagus} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork} \\ \hline 1 & 2 & 3 & 0 & 0 \\ 2 & 4 & 6 & 0 & 0 \\ 1 & 2 & 3 & 0 & 0 \\ 5 & 10 & 15 & 0 & 0 \\ 0 & 0 & 0 & 2 & 4 \\ 0 & 0 & 0 & 3 & 6 \\ 0 & 0 & 0 & 1 & 2 \end{array} =\sim \begin{pmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{pmatrix} \times \begin{pmatrix} 20.8 & 0 \\ 0 & 8.3 \end{pmatrix} \times \begin{pmatrix} 0.27 & 0.53 & 0.80 & 0 & 0 \\ 0 & 0 & 0 & 0.45 & 0.89 \end{pmatrix}$$

products

customers to concepts

concepts to products

Mixed preferences (rounding near-zero values)

$$\text{customers}\ \begin{array}{ccccc} \text{asparagus} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork} \\ 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 0.58 & 0.58 & 0.58 & 0.7 & 0.7 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{array} =\sim \begin{pmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.11 & 0.18 \\ 0.91 & 0 \\ 0 & 0.53 \\ 0 & 0.79 \\ 0 & 0.26 \end{pmatrix} \text{x} \begin{pmatrix} 9.54 & 0 \\ 0 & 5.38 \end{pmatrix} \text{x} \begin{pmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{pmatrix}$$

products

customers to concepts

concepts to products

What are the concepts now?

$$\text{customers}\ \begin{array}{c} \begin{array}{ccccc} \text{asparagus} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork} \end{array} \\ \left(\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 577 & 577 & 577 & 707 & 707 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right) \end{array} =\sim \begin{pmatrix} 0.008 & 0.16 \\ 0.017 & 0.32 \\ 1 & 0 \\ 0.04 & 0.80 \\ 0.01 & -0.26 \\ 0.02 & -0.39 \\ 0.007 & -0.13 \end{pmatrix} \times \begin{pmatrix} 1413 & 0 \\ 0 & 7.68 \end{pmatrix} \times \begin{pmatrix} 0.41 & 0.41 & 0.41 & 0.5 & 0.5 \\ 0.41 & 0.41 & 0.41 & -0.5 & -0.5 \end{pmatrix}$$

products

customers to concepts

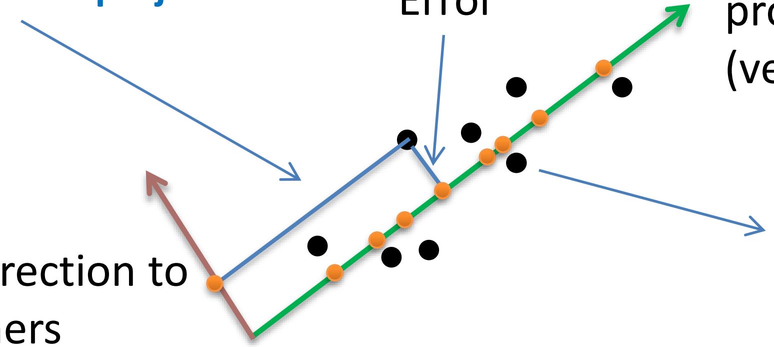
concepts to products

Geometric Interpretation

- Recall V: products to concepts matrix

$$V^T = \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix} \text{ concepts}$$

products



(U Λ=AV) gives the coordinates of the projection

“Error”

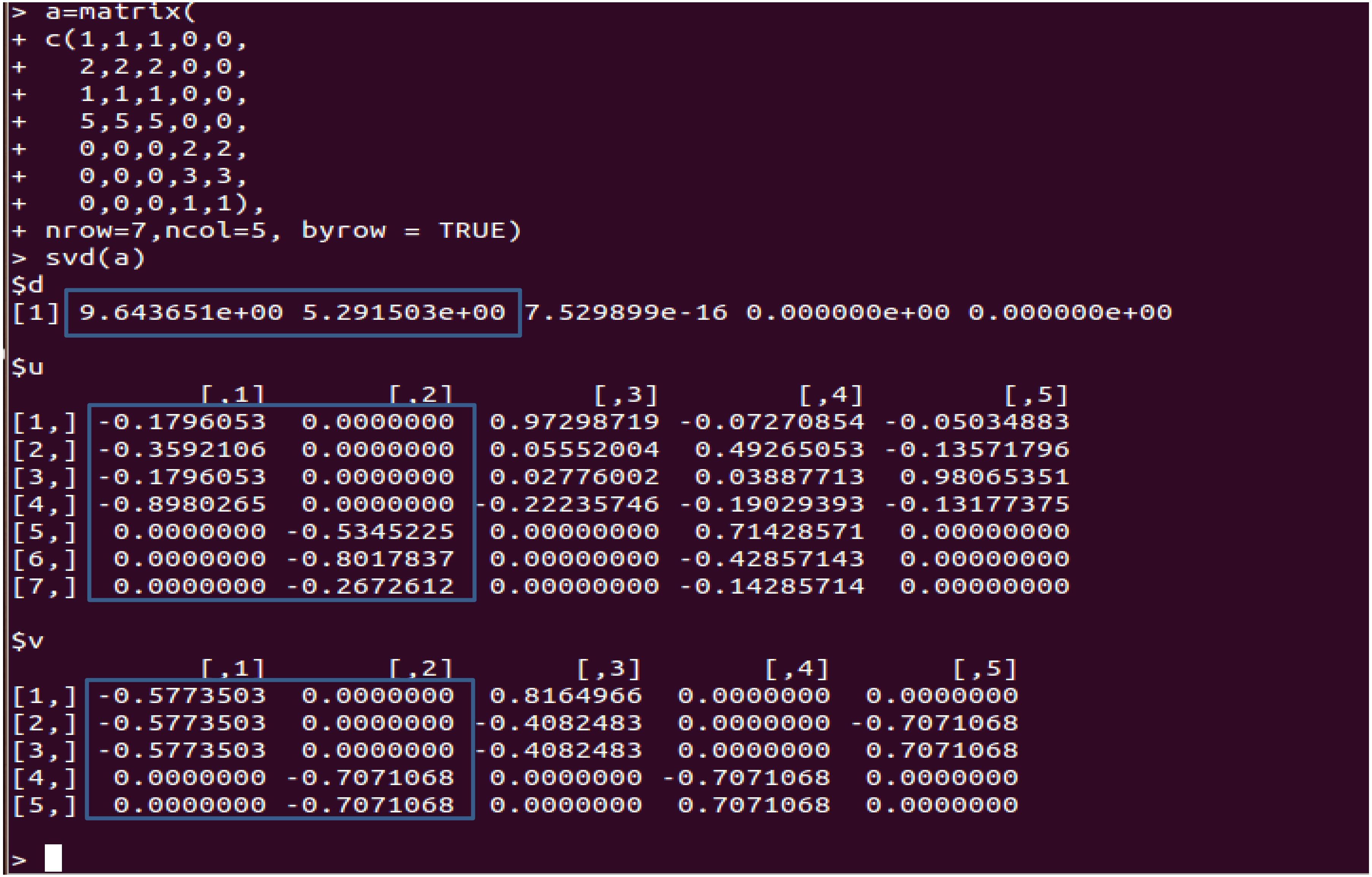
“best” direction to project customers (vegetarians concept)

Second-best direction to project customers (meat-eaters concept)

Each point is a customer depicted in the products space

#dims = #products !

SVD using R



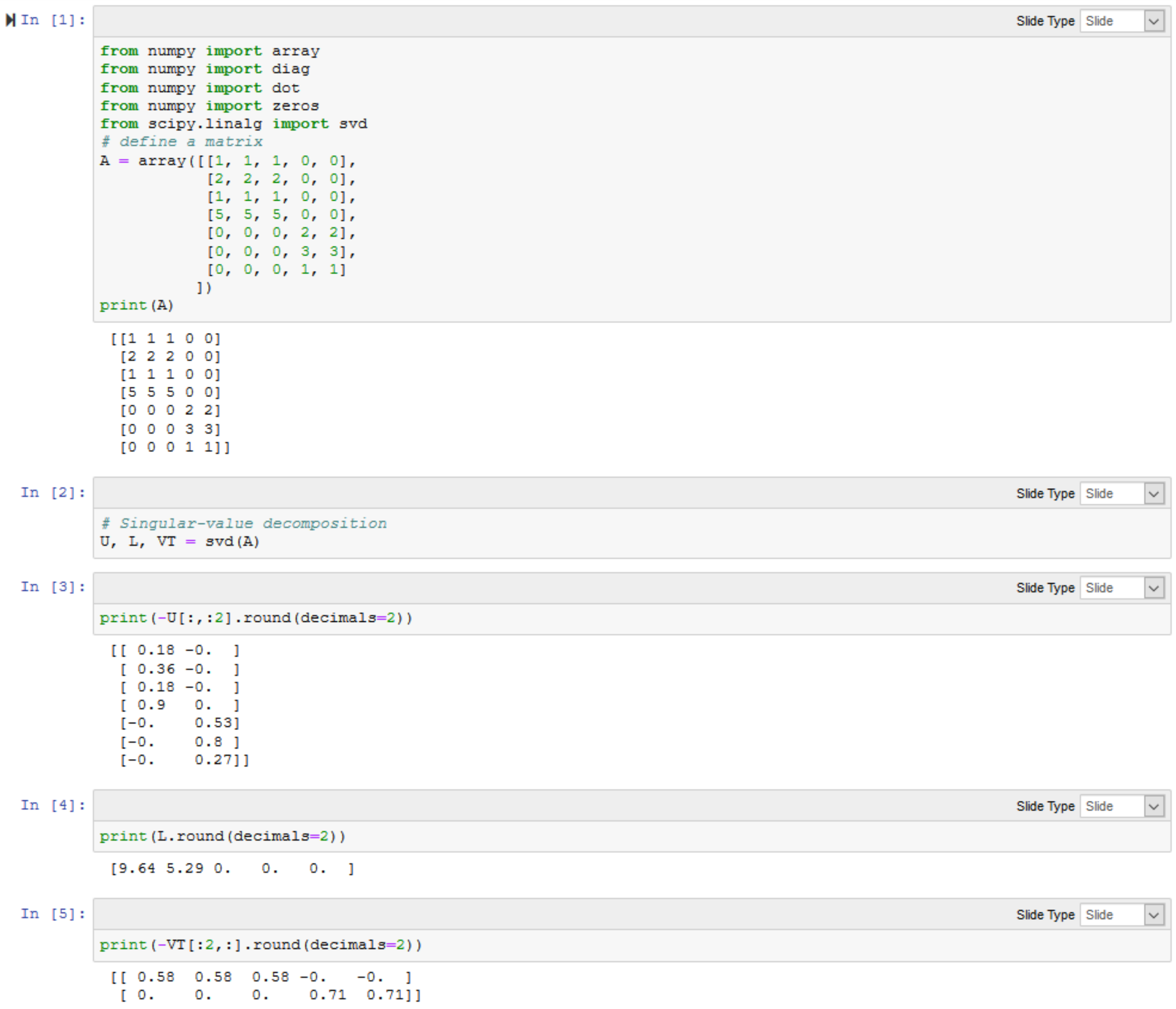
```
> a=matrix(
+ c(1,1,1,0,0,
+   2,2,2,0,0,
+   1,1,1,0,0,
+   5,5,5,0,0,
+   0,0,0,2,2,
+   0,0,0,3,3,
+   0,0,0,1,1),
+ nrow=7,ncol=5, byrow = TRUE)
> svd(a)
$d
[1] 9.643651e+00 5.291503e+00 7.529899e-16 0.000000e+00 0.000000e+00

$u
           [,1]       [,2]        [,3]        [,4]        [,5]
[1,] -0.1796053  0.0000000  0.97298719 -0.07270854 -0.05034883
[2,] -0.3592106  0.0000000  0.05552004  0.49265053 -0.13571796
[3,] -0.1796053  0.0000000  0.02776002  0.03887713  0.98065351
[4,] -0.8980265  0.0000000 -0.22235746 -0.19029393 -0.13177375
[5,]  0.0000000 -0.5345225  0.00000000  0.71428571  0.00000000
[6,]  0.0000000 -0.8017837  0.00000000 -0.42857143  0.00000000
[7,]  0.0000000 -0.2672612  0.00000000 -0.14285714  0.00000000

$v
           [,1]       [,2]       [,3]       [,4]       [,5]
[1,] -0.5773503  0.0000000  0.8164966  0.0000000  0.0000000
[2,] -0.5773503  0.0000000 -0.4082483  0.0000000 -0.7071068
[3,] -0.5773503  0.0000000 -0.4082483  0.0000000  0.7071068
[4,]  0.0000000 -0.7071068  0.0000000 -0.7071068  0.0000000
[5,]  0.0000000 -0.7071068  0.0000000  0.7071068  0.0000000

> █
```

SVD in Python



```
from numpy import array
from numpy import diag
from numpy import dot
from numpy import zeros
from scipy.linalg import svd
# define a matrix
A = array([[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]
          ])
print(A)
```

```
[[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]]
```

```
# Singular-value decomposition
U, L, VT = svd(A)
```

```
print(-U[:,:2].round(decimals=2))
```

```
[[ 0.18 -0.  ]
 [ 0.36 -0.  ]
 [ 0.18 -0.  ]
 [ 0.9   0.  ]
 [-0.    0.53]
 [-0.    0.8 ]
 [-0.    0.27]]
```

```
print(L.round(decimals=2))
```

```
[9.64 5.29 0.   0.   0.  ]
```

```
print(-VT[:2,:].round(decimals=2))
```

```
[[ 0.58  0.58  0.58 -0.   -0.  ]
 [ 0.    0.    0.    0.71  0.71]]
```

Applications of SVD

- John : “I like tomatoes”
- Think of John as a customer (row) vector
- $\vec{c}_{John}$ = [0 0 3 0 0]

- Mary : “I like lettuce and asparagus”
- $\vec{c}_{Mary}$ = [2 2 0 0 0]

- Vectors don’t look similar (inner-product = zero)
  - Sim($\vec{c}_{John}$,$\vec{c}_{Mary}$) = 0*2+0*2+3*0+0*0+0*0 = 0 (equivalently their cosine similarity = 0)

Map to the concepts space

Also… this is an example of dim-reduction (from 5-dims → 2-dims)

$$\begin{bmatrix} 0 & 0 & 3 & 0 & 0 \end{bmatrix} \times \begin{pmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{pmatrix} = \begin{bmatrix} 1.74 & 0 \end{bmatrix}$$

products-to-concepts V

⬆

Products that John/Mary buy (or prefer)

⬇

$$\begin{bmatrix} 2 & 2 & 0 & 0 & 0 \end{bmatrix} \times \begin{pmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{pmatrix} = \begin{bmatrix} 2.32 & 0 \end{bmatrix}$$

products-to-concepts V

Now John & Marry look similar (cos=1) even though their baskets don’t contain common products

Building Recommendations

- John : “I like tomatoes”
- Think of John as a customer (row) vector
- $\vec{c}_{John}$ = [0 0 3 0 0]

- Which additional products could John find appealing?

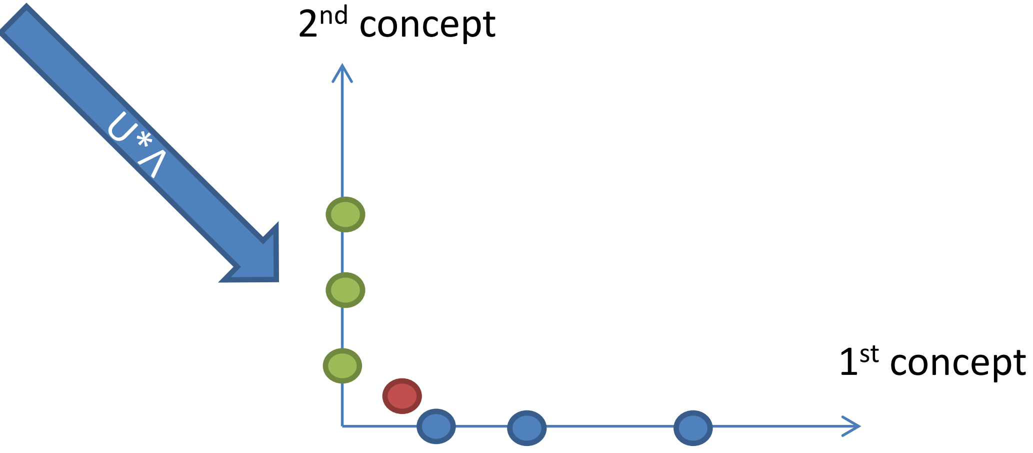
Map to the concepts space (a), and back (b)

a) $$\begin{bmatrix} 0 & 0 & 3 & 0 & 0 \end{bmatrix} \times \underset{\text{products-to-concepts V}}{\begin{pmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{pmatrix}} = \begin{bmatrix} 1.74 & 0 \end{bmatrix}$$

b) $$\begin{bmatrix} 1.74 & 0 \end{bmatrix} \times \underset{\text{concepts-to-products } V^T}{\begin{pmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{pmatrix}} = \begin{bmatrix} 1.01 & \underbrace{1.01 \quad 1.01}_{\text{John may also like these two!}} & 0 & 0 \end{bmatrix}$$

Map to the concepts' space

$$\text{customers}\ \begin{array}{c} \begin{array}{ccccc} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork} \end{array} \\ \left(\begin{array}{ccccc} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 0.58 & 0.58 & 0.58 & 0.7 & 0.7 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right) \\ \text{products} \end{array} =\sim \begin{array}{c} \left(\begin{array}{cc} 0.18 & 0 \\ 0.36 & 0 \\ 0.11 & 0.18 \\ 0.91 & 0 \\ 0 & 0.53 \\ 0 & 0.79 \\ 0 & 0.26 \end{array}\right) \\ \text{customers to concepts} \end{array} \times \left(\begin{array}{cc} 9.54 & 0 \\ 0 & 5.38 \end{array}\right) \times \begin{array}{c} \left(\begin{array}{ccccc} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{array}\right) \\ \text{concepts to products} \end{array}$$



Note: all customers lie on a plane in the 5th dimensional space. This plane is defined by the first two columns of V.

Rank of Matrix

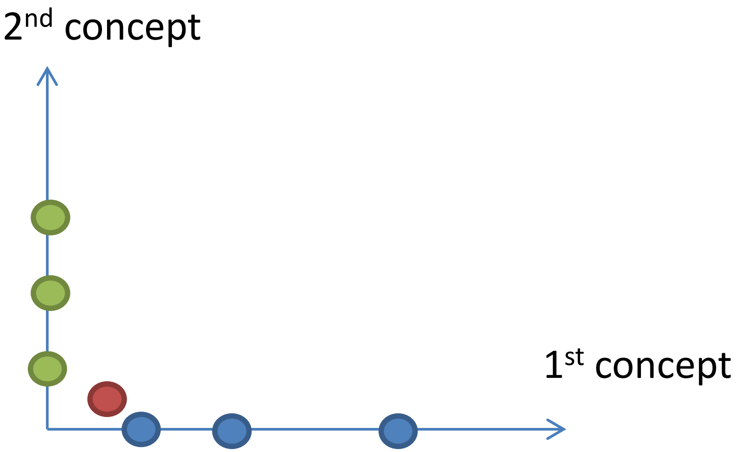
$$\text{customers}\ \left(\begin{array}{ccccc} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork} \\ 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 0.58 & 0.58 & 0.58 & 0.7 & 0.7 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right) =\sim \begin{pmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.11 & 0.18 \\ 0.91 & 0 \\ 0 & 0.53 \\ 0 & 0.79 \\ 0 & 0.26 \end{pmatrix} \text{x} \begin{pmatrix} 9.54 & 0 \\ 0 & 5.38 \end{pmatrix} \text{x} \begin{pmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{pmatrix}$$

products

customers to concepts

concepts to products

The rank of this matrix is two



Dimensionality Reduction

vegetarians concept

meat-eaters concept

$$\text{customers}\ \begin{array}{c}\begin{array}{ccccc}\text{asparagus} & \text{lettuce} & \text{tomatoes} & \text{beef} & \text{pork}\end{array}\\ \left(\begin{array}{ccccc}1 & 1 & 1 & 0 & 0\\ 2 & 2 & 2 & 0 & 0\\ 1 & 1 & 1 & 0 & 0\\ 5 & 5 & 5 & 0 & 0\\ 0 & 0 & 0 & 2 & 2\\ 0 & 0 & 0 & 3 & 3\\ 0 & 0 & 0 & 1 & 1\end{array}\right)\\ \text{products}\end{array} = \left(\begin{array}{cc}0.18 & \text{[redacted]}\\ 0.36 & \text{[redacted]}\\ 0.18 & \text{[redacted]}\\ 0.90 & \text{[redacted]}\\ 0 & \text{[redacted]}\\ 0 & \text{[redacted]}\\ 0 & \text{[redacted]}\end{array}\right) \times \left(\begin{array}{cc}9.64 & 0\\ 0 & 0\end{array}\right) \times \left(\begin{array}{ccccc}0.58 & 0.58 & 0.58 & 0 & 0\\ \text{[redacted]} & \text{[redacted]} & \text{[redacted]} & \text{[redacted]} & \text{[redacted]}\end{array}\right)$$

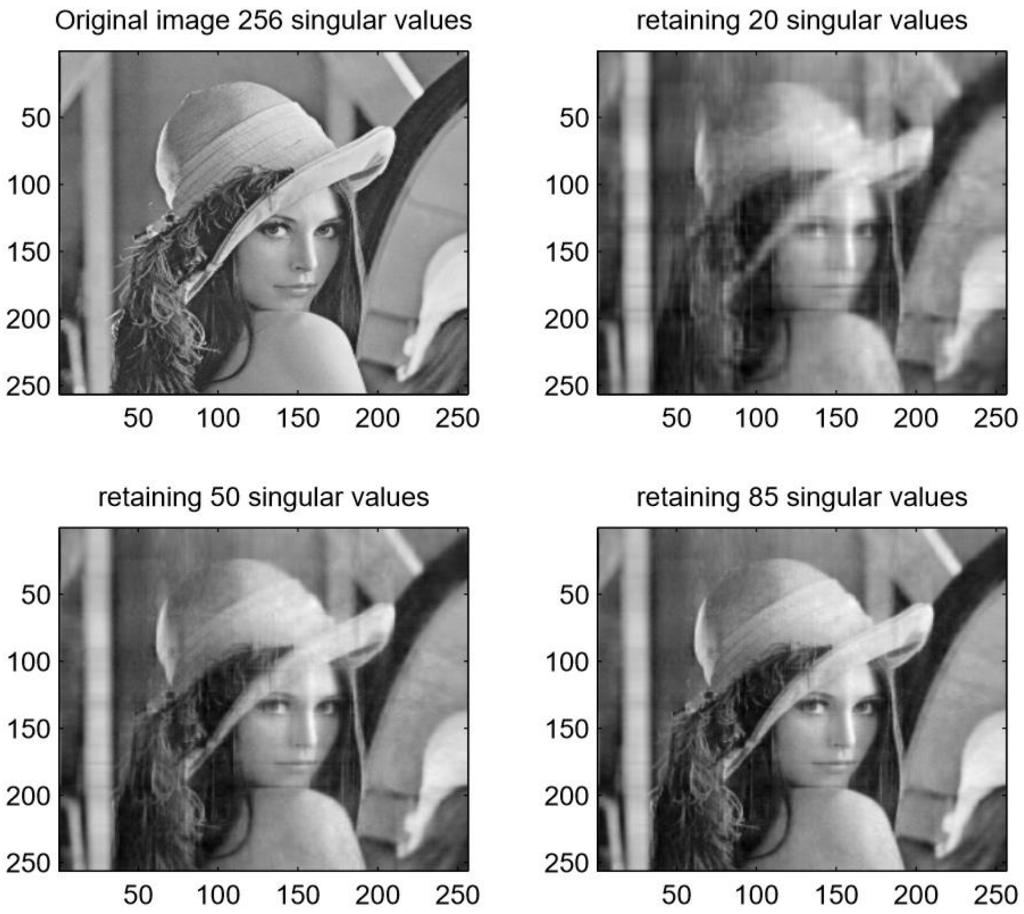
$$= \begin{array}{c}\left(\begin{array}{ccccc}1 & 1 & 1 & 0 & 0\\ 2 & 2 & 2 & 0 & 0\\ 1 & 1 & 1 & 0 & 0\\ 5 & 5 & 5 & 0 & 0\\ 0 & 0 & 0 & \mathbf{0} & \mathbf{0}\\ 0 & 0 & 0 & \mathbf{0} & \mathbf{0}\\ 0 & 0 & 0 & \mathbf{0} & \mathbf{0}\end{array}\right)\\ \text{products}\end{array}\ \text{customers}$$

Best 1-rank approximation

Truncated SVD

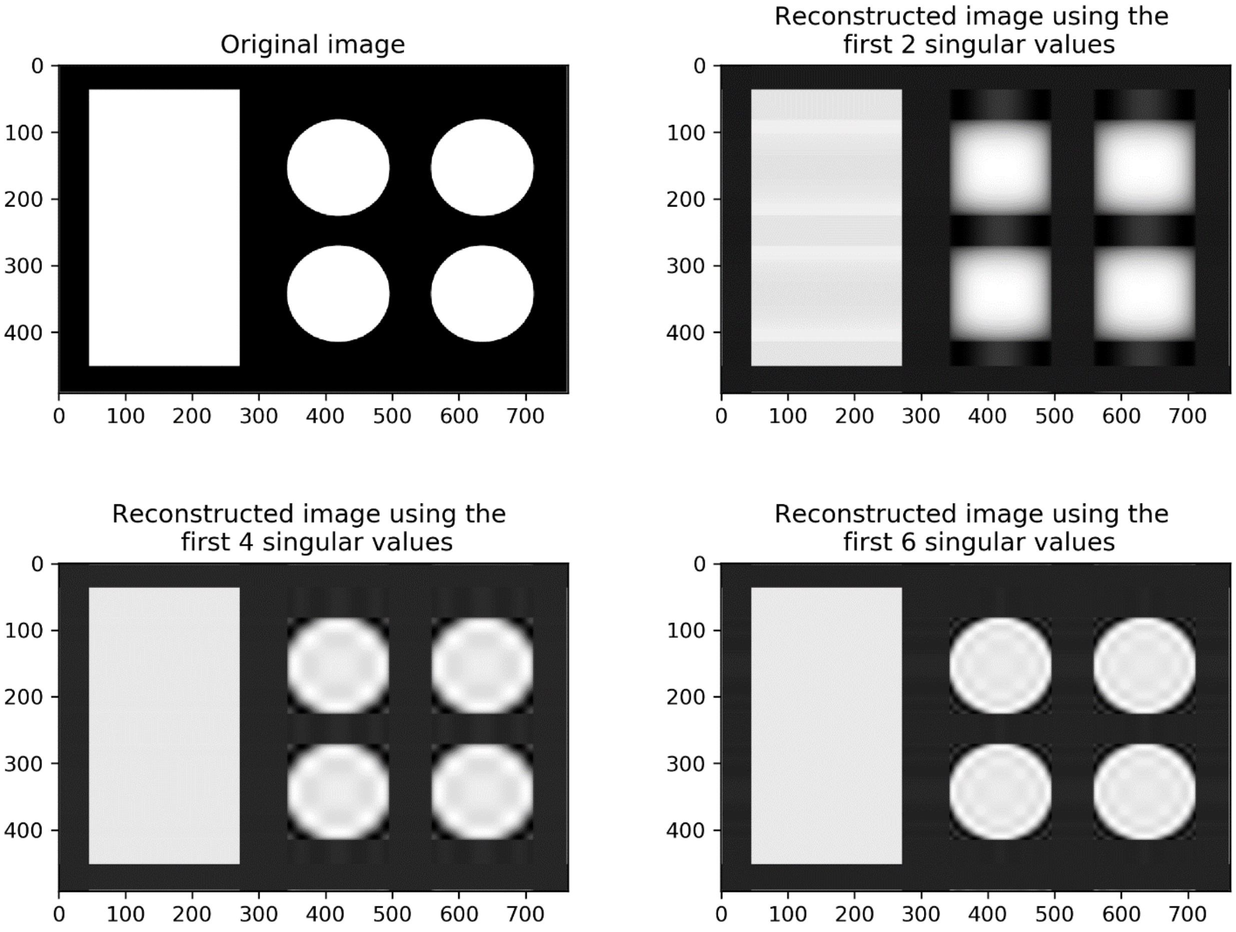
- Concentrate on the most important parts using a rank-**k** approximation of the matrix
- How to select the value of k?

Example (Image Compression)


Original image 256 singular values
retaining 20 singular values
retaining 50 singular values
retaining 85 singular values
50
100
150
200
250

Images from: Singular value decomposition in image noise filtering and reconstruction
T Workalemahu - 2008

Another example*



Original image

Reconstructed image using the first 2 singular values

Reconstructed image using the first 4 singular values

Reconstructed image using the first 6 singular values

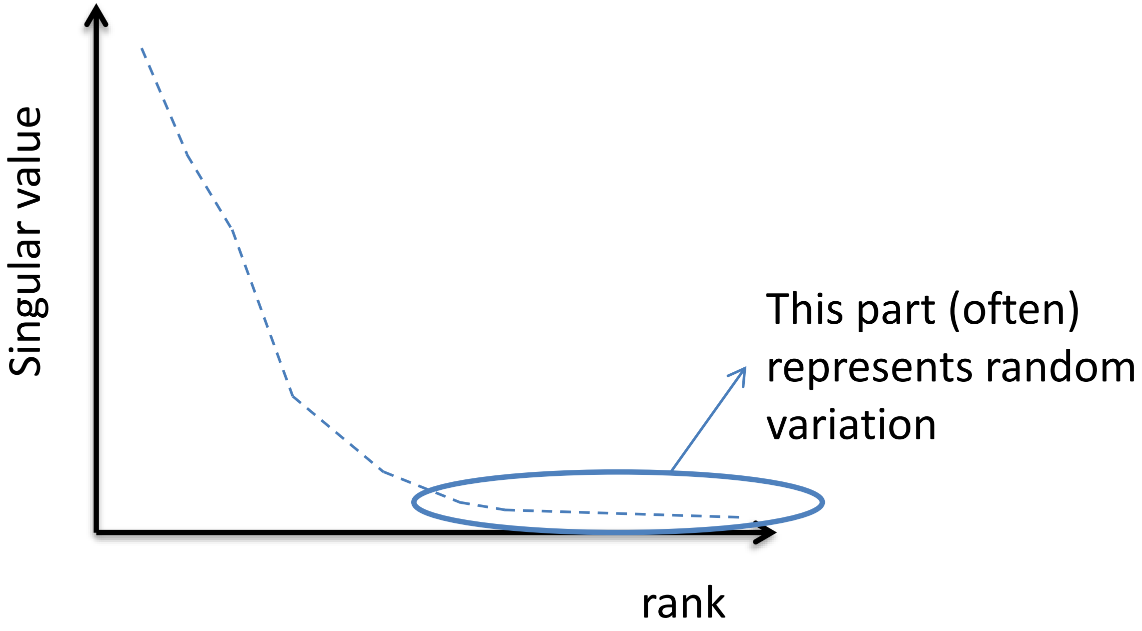
* credit: https://towardsdatascience.com/understanding-singular-value-decomposition-and-its-application-in-data-science-388a54be95d

How do we achieve compression?

- Original image: n x m values
- By selecting the first k singular values we store
  - k singular values
  - First k columns of matrix U (n*k values)
  - First k rows of matrix V^T (m*k values)
  - Thus, we need a total of k*(1+n+m) values
    - Compare with n x m

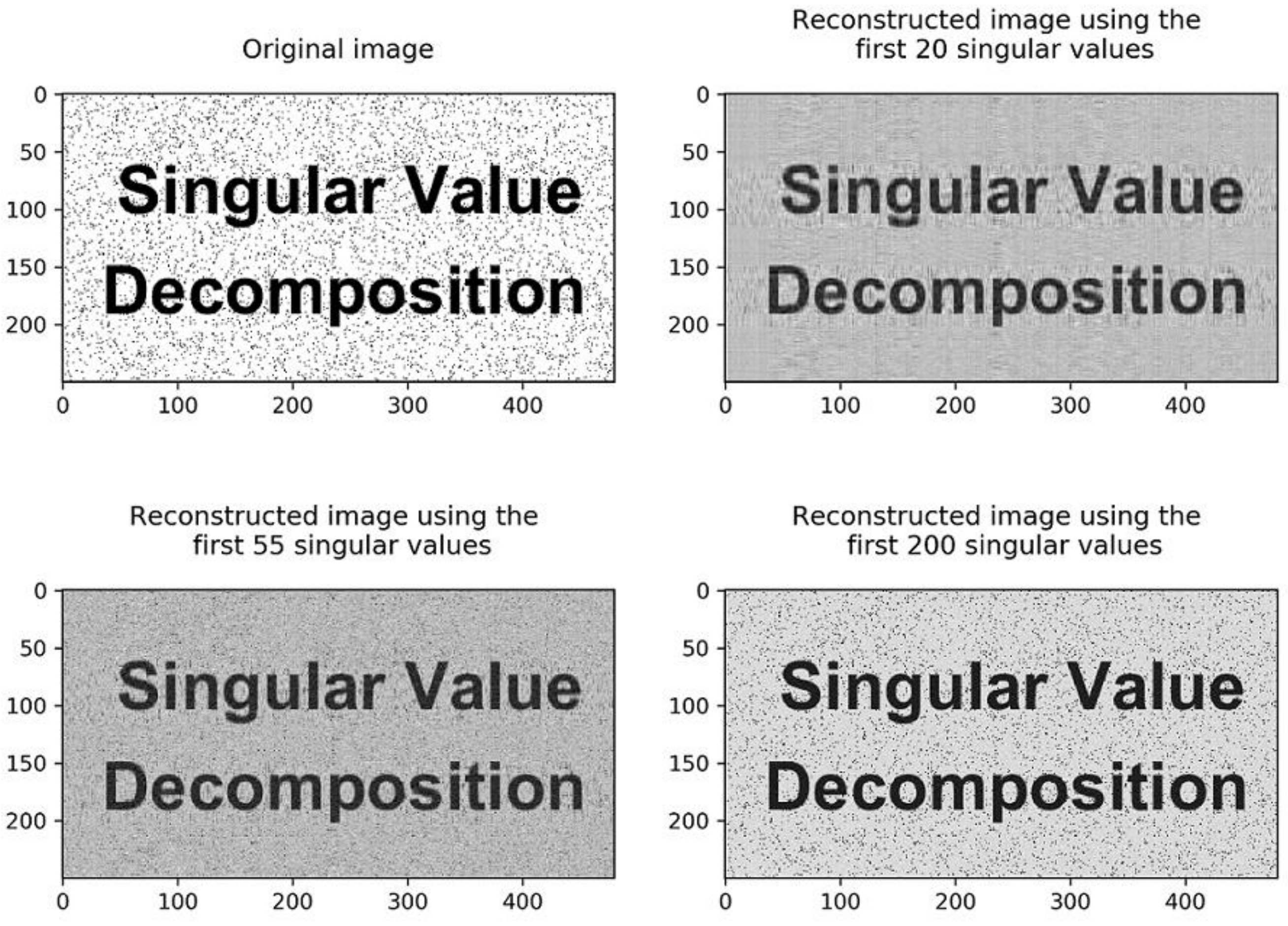
Scree plot

- Plot the value of successive singular values (or their squared values) against the rank order
  - Decision is subjective



This part (often) represents random variation

Use truncated SVD for removing noise



Original image

Reconstructed image using the first 20 singular values

Reconstructed image using the first 55 singular values

Reconstructed image using the first 200 singular values

* credit: https://towardsdatascience.com/understanding-singular-value-decomposition-and-its-application-in-data-science-388a54be95d

More on denoising

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & \boxed{\mathbf{0}} & \boxed{1} \end{pmatrix} \xrightarrow{K=2} \begin{matrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & \boxed{2.9} & \boxed{3.1} \\ 0 & 0 & 0 & \boxed{\mathbf{0.5}} & \boxed{\mathbf{0.5}} \end{matrix}$$

$$V^T = \begin{pmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.69 & 0.72 \\ 0 & 0 & 0 & -0.72 & 0.69 \end{pmatrix} \qquad \Lambda = \begin{pmatrix} 9.64 & 0 & 0 \\ 0 & 5.15 & 0 \\ 0 & 0 & 0.70 \end{pmatrix}$$

Frobenious norm
(a.k.a. Euclidean Norm)

$$\|A\|_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{i\,j}|^2}$$

original matrix

customers

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \xrightarrow{\text{SVD}} \overset{\Lambda}{\begin{pmatrix} 9.64 & 0 \\ 0 & 5.29 \end{pmatrix}}$$

products

$$\sigma_1^2 + \sigma_2^2 = 121$$

$$\|A\|_F^2 = 3*1^2+3*2^2+3*1^2+3*5^2+2*2^2+2*3^2+2*1^2=121$$

$||A||_F$ v.s. singular values

original matrix

$$\text{customers}\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

products

1-rank approximation

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}\text{customers}$$

products

$3*1^2+3*2^2+3*1^2+3*5^2+2*2^2+2*3^2+2*1^2=121$

$\sigma_1^2 + \sigma_2^2 = 121$

$3*1^2+3*2^2+3*1^2+3*5^2=93$

$\sigma_1^2 = 93$

Select k based on $\|A\|_F^2$

- Recall $\|A\|_F^2 = \Sigma(\sigma_i^2)$
- List singular values in decreasing order:

  $\sigma_1, \sigma_2, \sigma_3 \ldots$

- Select k singular values such that the sum of their squares is (for example) ≥ 80% of the total sum of the squared singular values
  - Ensures that 80% of the squared Frobenious norm is preserved

$$\Lambda = \begin{pmatrix} \mathbf{9.64} & \mathbf{0} \\ \mathbf{0} & \mathbf{5.29} \end{pmatrix}$$

$\sigma_1^2 + \sigma_2^2 = 121$

$\sigma_1^2 = 93$

→

1st-rank approximation preserves 93/121=77% of the energy

Consideration

- Results depend on magnitude of data values
- E.g.: first column is customer age (18..65)

  second column is income (5000..100000)

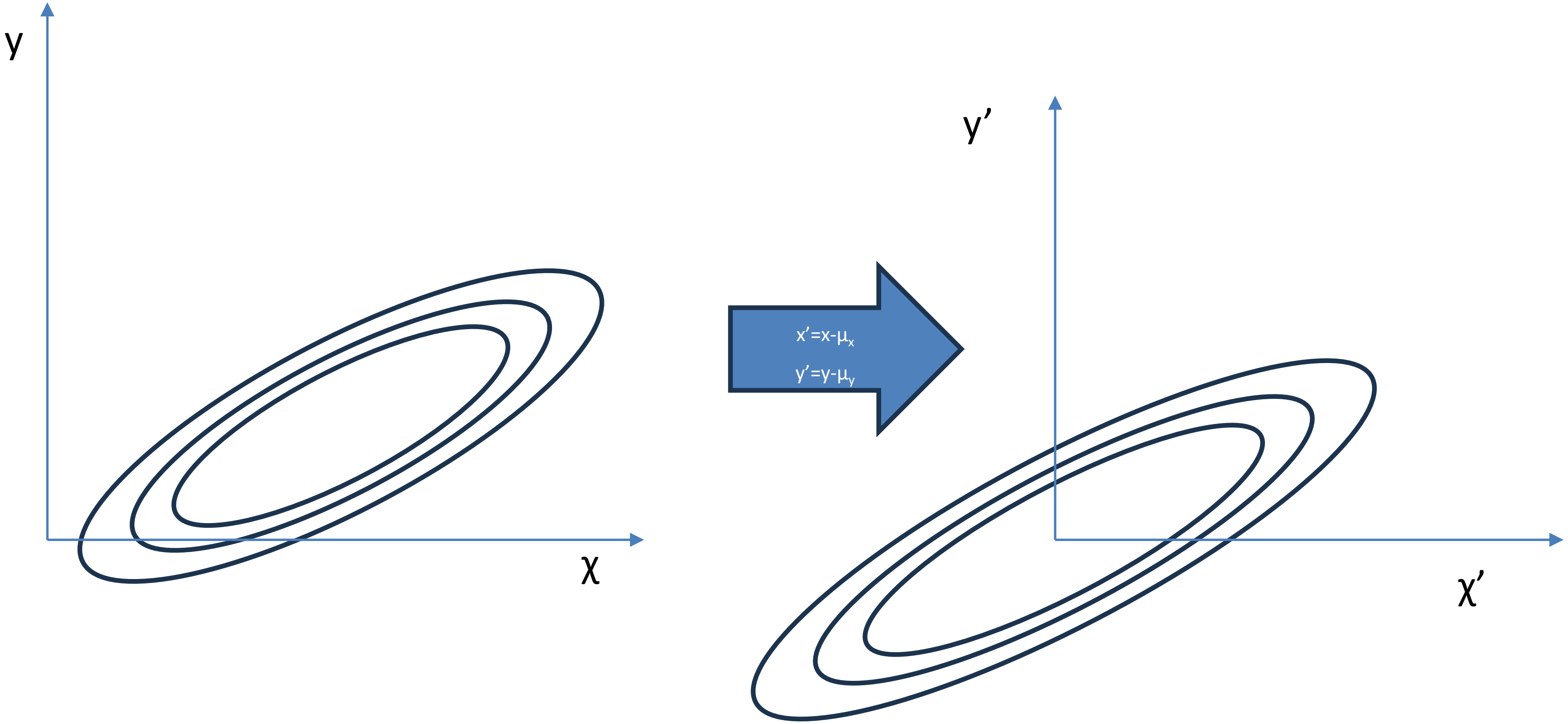
- Then, income seems more important than age
  - For some datasets this may be true, but not for all

Suggestion

- First do mean normalization (center to zero) by subtracting the mean value per column

- Optionally do feature scaling
  - e.g. divide by standard deviation (normalize magnitude) per column
  - other approach is to reduce features with large magnitude (e.g. take square roots/logarithms)

Mean Normalization

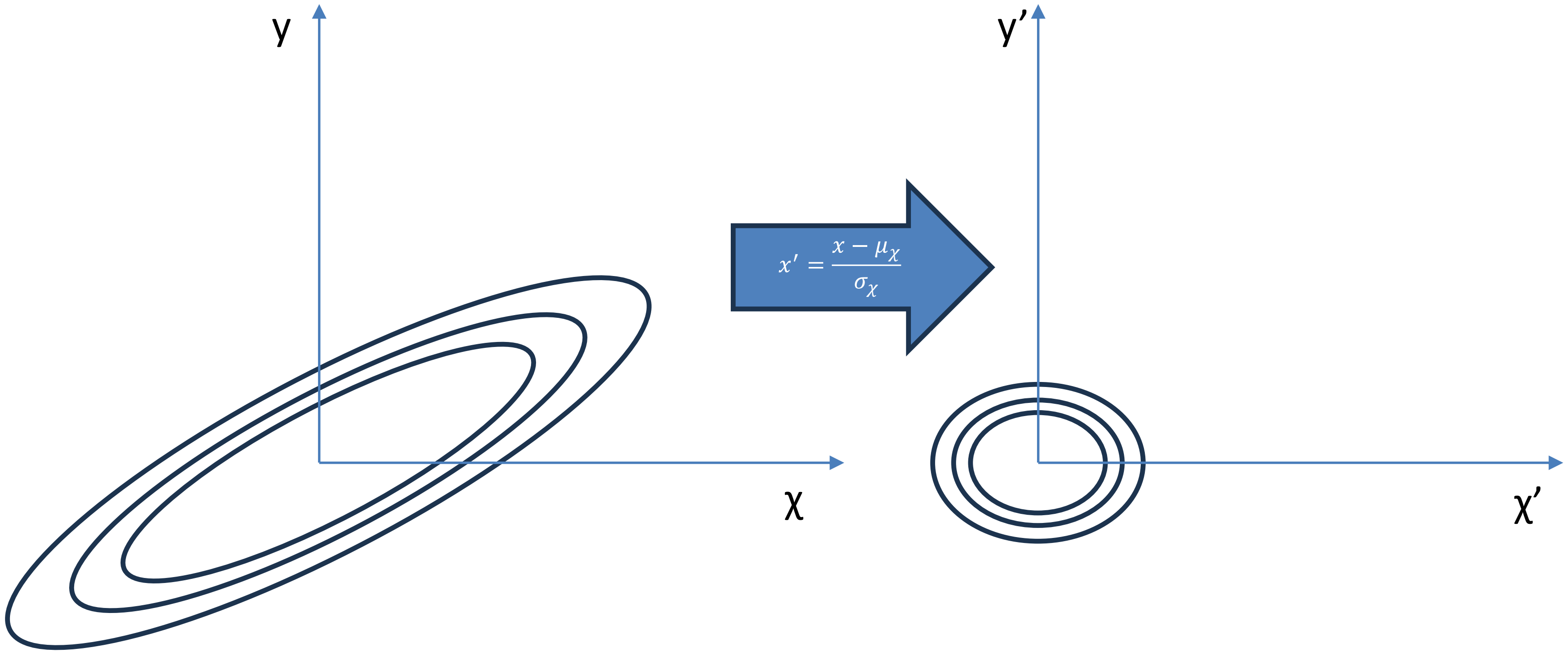


$x' = x - \mu_x$

$y' = y - \mu_y$

Feature Scaling

(e.g. standardization in this example)



$$x' = \frac{x - \mu_\chi}{\sigma_\chi}$$

Another interpretation: Ratio Rules

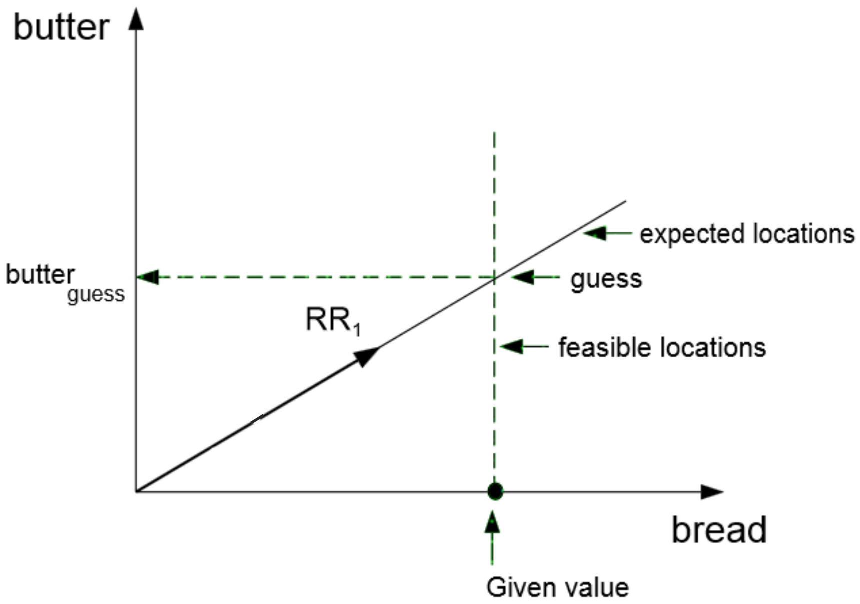
$$V^T = \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix} \text{ concepts}$$

products

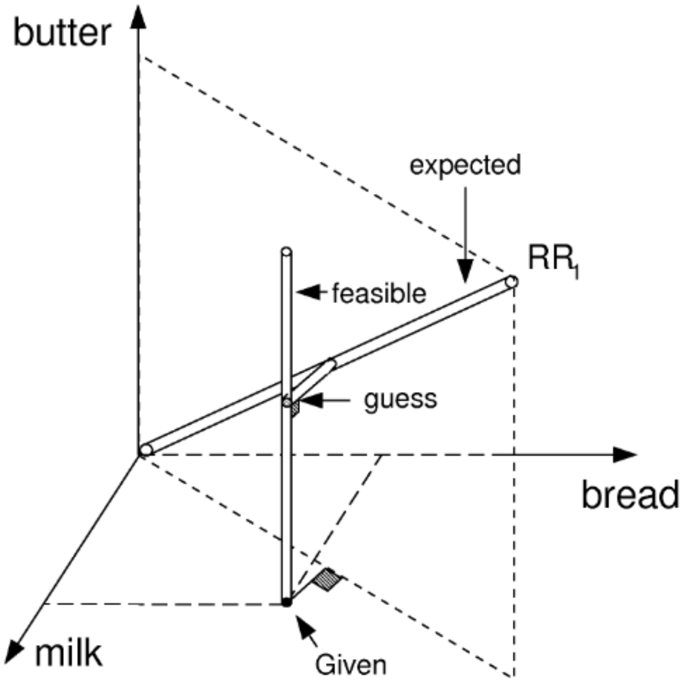
- RR1: “A (vegetarian)customer spends 0.58:0.58:0.58 € on asparagus:lettuce:tomatoes”
- Ratio Rules Construction
  - Obtain matrix A
  - Compute A_c (zero mean) from A by subtracting column averages
  - Compute $C = A_c^T A$ product-to-product similarity matrix
  - Ratio Rules in V are the eigenvectors of C

What-if-Analysis

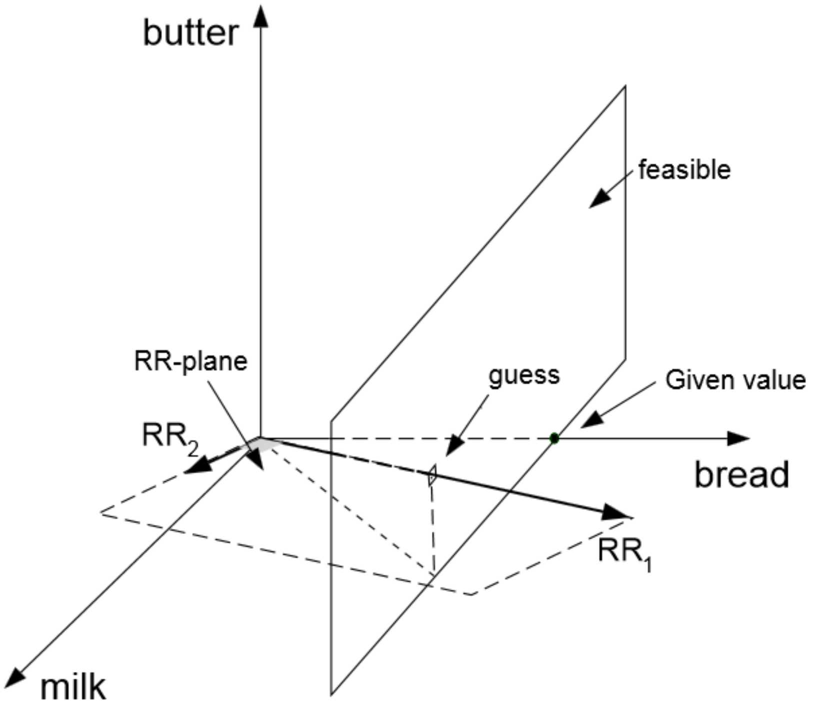
- Simple case: exactly-specified
- Example
  - RR1: bread:butter
  - A customer spends 5$ on bread
  - How much is she expected to spend on butter?



Harder cases*



over-specified



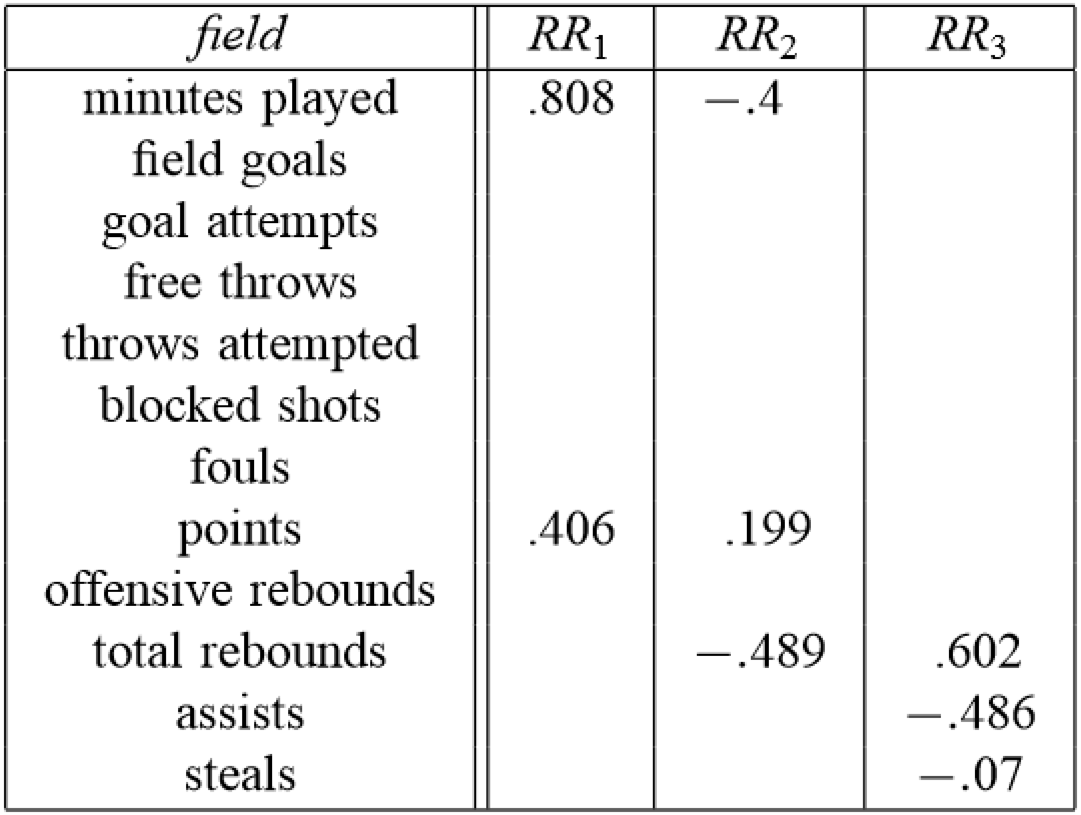
under-specified

*See: ”Quantifiable Data Mining Using Ratio Rules”

Other use: detect outlier values

- Inspect Customer x:
  - (5,2,3,0,51) → Looks suspicious
- Remove suspected value(s), replace with unknown (5,2,3,0,?)
- Run previous algorithm, reconstruct missing value from RRs
  - (5,2,3,0,48)
- Compare against guessed value

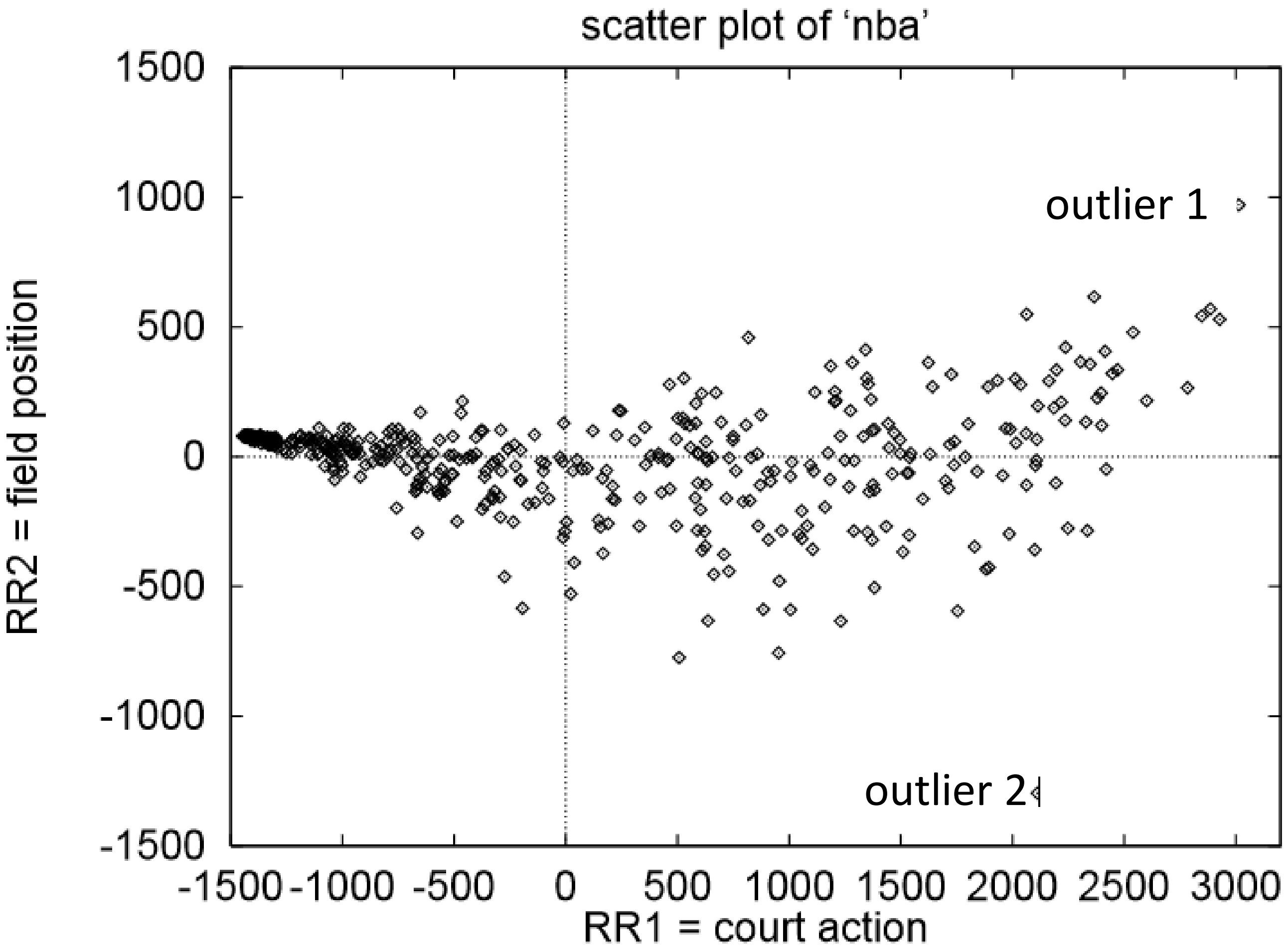
Sample of V matrix for NBA 1991-2 season stats



field	RR_1	RR_2	RR_3
minutes played	.808	$-.4$	
field goals			
goal attempts			
free throws			
throws attempted			
blocked shots			
fouls			
points	.406	.199	
offensive rebounds			
total rebounds		$-.489$	.602
assists			$-.486$
steals			$-.07$

RR1: level of activity (stars vs bench players)
RR2: field position (offensive/defensive players)
defensive players are good rebounders but score less points per minutes played than offensive players
RR3: height of a players (tall guys get more rebounds but have less steals/assists

Visualization



Latent Semantic Indexing (Information Retrieval)

- Assume A is a (terms) x (documents) matrix
  - Values denote the term frequencies (or e.g. tf*idf scores)

	d1	d2	d3	…
Term1	4	0	2	…
Term2	0	2	0	…
Term3	3	2	0	…
Term4	2	0	1	…
Term5	11	7	0	…
…	0	0	0	…
…	0	0	1	…

How to define similarity between docs?

	d1	d2	d3	…
Term1	4	0	2	…
Term2	0	2	0	…
Term3	3	2	0	…
Term4	2	0	1	…
Term5	11	7	0	…
…	0	0	0	…
…	0	0	1	…

Issue 1: Synonymy

- Different words with the same meaning

	d1	d2	d3	…
Car	4	0	2	…
Term2	0	2	0	…
Automobile	3	2	0	…
Term4	2	0	1	…
Term5	11	7	0	…
…	0	0	0	…
…	0	0	1	…

Issue 2: Polysemy

- The same word having other meanings

	d1	d2	d3	…
Term1	4	0	2	…
Term2	0	2	0	…
Term3	3	2	0	…
Term4	2	0	1	…
Apple	11	7	0	…
…	0	0	0	…
…	0	0	1	…

mentioned as a fruit

mentioned as a technology company

Latent Semantic Indexing

- SVD (LSI) discovers the “topics” (=concepts) discussed
- Handle synonymy: e.g. car, automobile, αυτοκίνητο, αμάξι all refer to the same topic
- Handle polysemy: if apple is mentioned along with other fruits in a document it will be mapped to the apple-the-fruit concept (crude analogy)

SVD vs PCA

- SVD: $A=U \Lambda V^T$
- PCA : $A=X L X^T$
  - A is symmetric
  - U, V, X are orthonormal → $X^TX=I$, $V^TV=I$, $U^TU=I$
- Given a non-symmetric matrix A
  - A^TA is symmetric (thus PCA applies)
  - $A^TA= V\Lambda^T U^T(U\Lambda V^T) = V \Lambda^2 V^T = X L X^T$
    - for $X = V$, $L= \Lambda^2$
    - also, $l_i = \lambda_\iota^2$

eigen-values

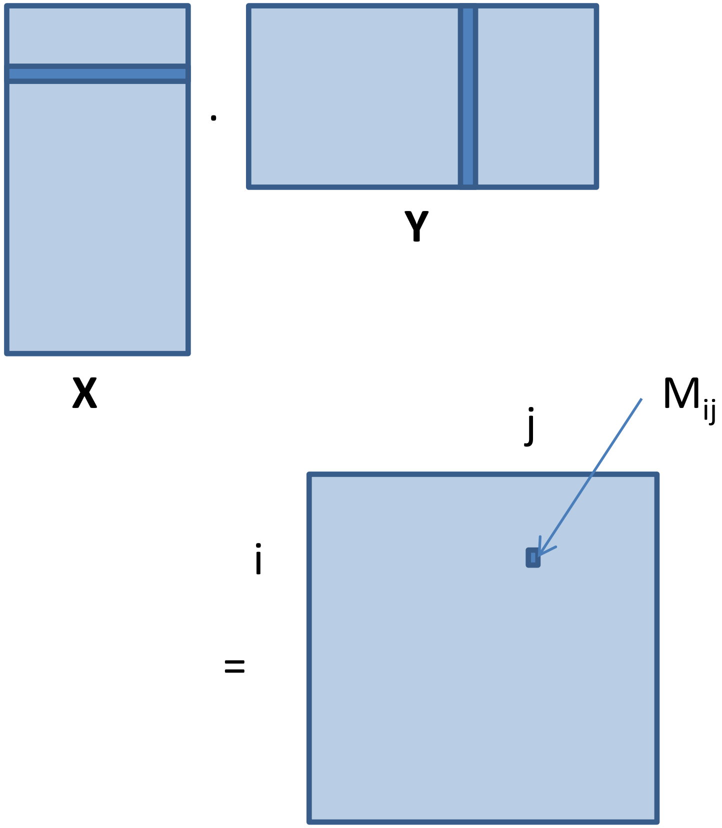
singular-values

Quick Note: table multiplication

- Let $M_{nxm} = X_{nxl} * Y_{lxm}$
- Then:

$$M_{ij} = \sum_{k=1}^{l} (X_{ik} * Y_{kj})$$

$= \vec{x}_i \cdot \vec{y}_j$



Dot products

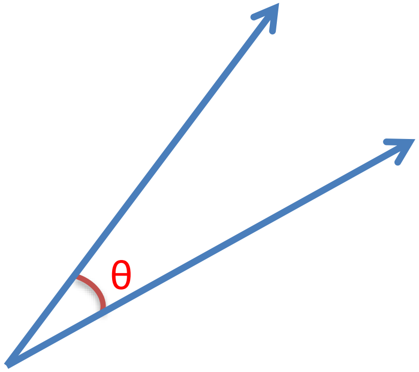
- $\vec{x} \,.\, \vec{y} = \Sigma(x_k * y_k)$
- Example:

  $\vec{x} = (1,3,0,5)$
  $\vec{y} = (1,0,1,6)$

- Then:

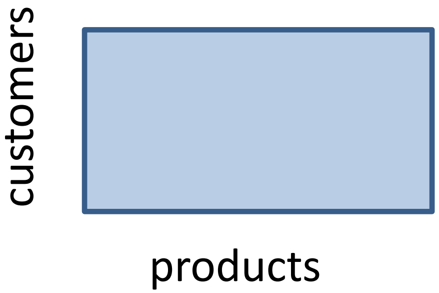
  $\vec{x} \,.\, \vec{y} = 1*1+3*0+0*1+5*6=31$

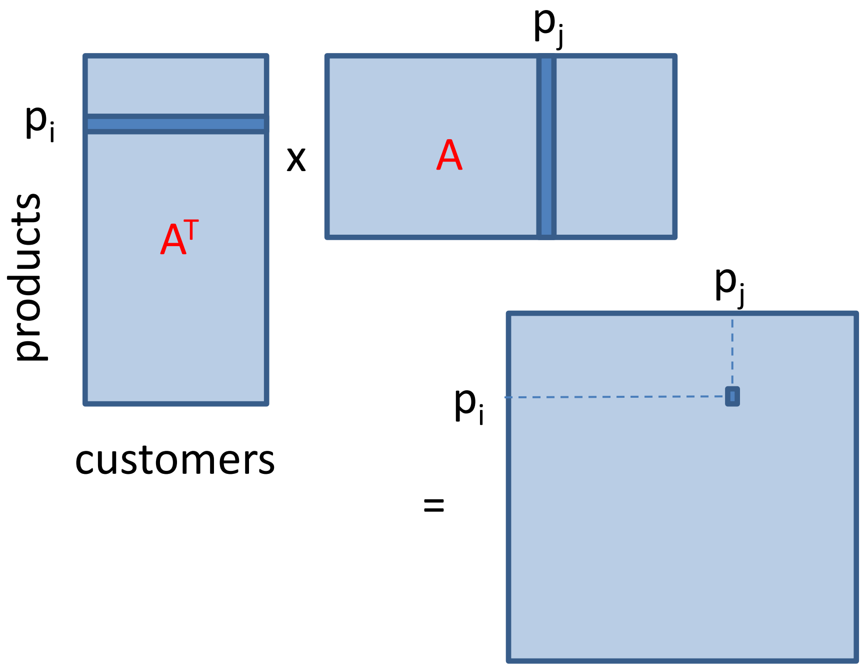
  $= |\vec{x}| * |\vec{y}| * \cos(\theta(\vec{x}, \vec{y}))$



More interesting properties of SVD

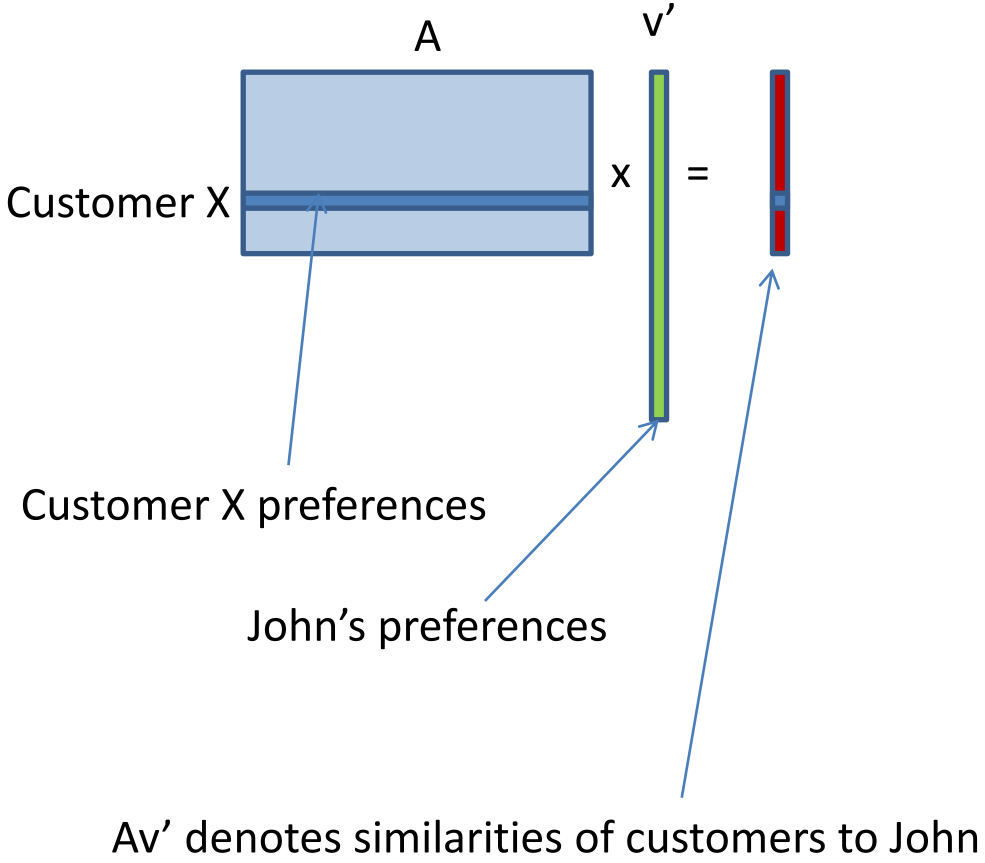
- Recall $A_{n \times m}$: customers to products matrix
- SVD: $A = U \Lambda V^T$
- Then $A^T A = V \Lambda^2 V^T$ (product-to-product similarity matrix)
    - Similarity based on their customers





Exploring User’s similarities

- Let v’ be a mx1 vector denoting the preferences of customer “John”
- What is Av’ ?



Customer X preferences

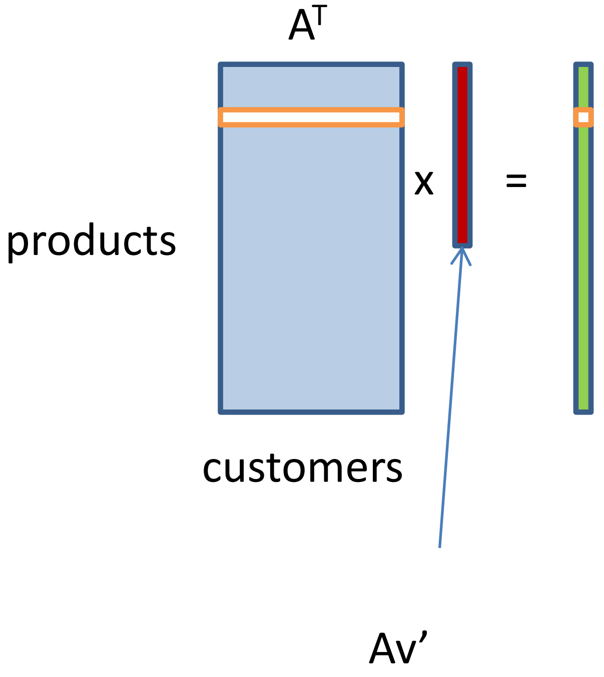
John’s preferences

Av’ denotes similarities of customers to John

Exploring User’s similarities

- Av’ denotes similarities of customers to John
- What is $A^T(Av')$?

- Products that customers similar to John buy
  - a high value indicates that the customers for this product are similar to the customers that are similar to John



In “facebook terms” what friends of John like
(assuming “friend of John” means similar to John)

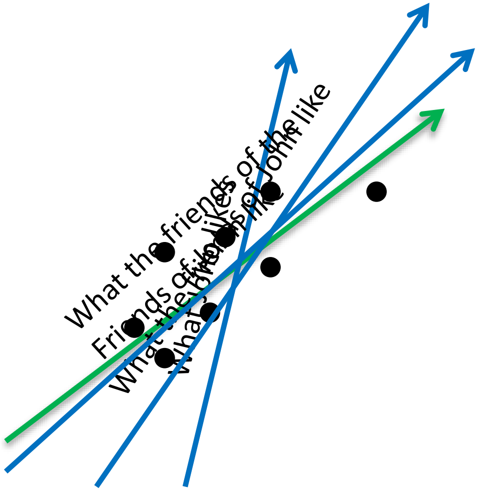
Thus, $A^T(Av')$ measures how the “friends” of John like each product

Take it further

- $(A^TA)v'$ = what the friends of John like
- Via similar arguments:
- $(A^TA)^2v$ ‘ = what the friends of the friends of John like

- $(A^TA)^kv$ ‘ = what k-hops away friends like

Visually



WHAT IS THIS?

Interesting property

- Computation converges to a vector parallel to v_1
  - $(A^TA)^k = V\ \Lambda^{2k}\ V^T\ =\sim v_1\ \lambda_1^{2k}\ v_1^T$ for k>>1
    - Recall that v_1 is also the 1st eigenvector of A^TA
    - By definition $(A^TA)v_1 = \lambda_1^2\ v_1$
  - Therefore: $(A^TA)^k v\ ' =\sim$ constant $*\ v_1$
- After k steps it doesn't matter where we started from. Will converge to the first (strongest) eigenvector
  - property used for computing pageRank (A is derived from the adjacency matrix of the web)

Example

Dataset A (Customer x Products) · v' = Av'

$$\begin{array}{|c|c|c|c|c|}
\hline
1 & 1 & 1 & 0 & 0 \\ \hline
2 & 2 & 2 & 0 & 0 \\ \hline
1 & 1 & 1 & 0 & 0 \\ \hline
5 & 5 & 5 & 0 & 0 \\ \hline
0 & 0 & 0 & 2 & 2 \\ \hline
0 & 0 & 0 & 3 & 3 \\ \hline
0 & 0 & 0 & 1 & 1 \\ \hline
\end{array}
\cdot
\begin{array}{|c|}
\hline
0 \\ \hline
0 \\ \hline
3 \\ \hline
0 \\ \hline
0 \\ \hline
\end{array}
=
\begin{array}{|c|l}
\hline
3 & \text{Customer 1} \\ \hline
6 & \text{Customer 2} \\ \hline
3 & \text{Customer 3} \\ \hline
15 & \text{Customer 4} \\ \hline
0 & \text{Customer 5} \\ \hline
0 & \text{Customer 6} \\ \hline
0 & \text{Customer 7} \\ \hline
\end{array}$$

John: I like tomatoes

Customers-to-John similarity vector

Conclusion: first four customers are similar to John

Dataset A (Customer x Products) · v' = Av'

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ \color{red}{3} \\ 0 \\ 0 \end{bmatrix} = \begin{array}{|c|l} \hline 3 & \textbf{Customer 1} \\ \hline 6 & \textbf{Customer 2} \\ \hline 3 & \textbf{Customer 3} \\ \hline 15 & \textbf{Customer 4} \\ \hline 0 & \text{Customer 5} \\ \hline 0 & \text{Customer 6} \\ \hline 0 & \text{Customer 7} \\ \hline \end{array}$$

John: I like tomatoes

Customers-to-John similarity vector

Moreover, customers that are like John buy vegetables

Dataset A^T (Product x Customer) Av' $A^T(Av')$

$$\begin{bmatrix} 1 & 2 & 1 & 5 & 0 & 0 & 0 \\ 1 & 2 & 1 & 5 & 0 & 0 & 0 \\ 1 & 2 & 1 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 3 & 1 \\ 0 & 0 & 0 & 0 & 2 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 6 \\ 3 \\ 15 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{array}{c|r} \mathbf{93} & \textbf{asparagus} \\ \mathbf{93} & \textbf{lettuce} \\ \mathbf{93} & \textbf{tomato} \\ 0 & \\ 0 & \end{array}$$

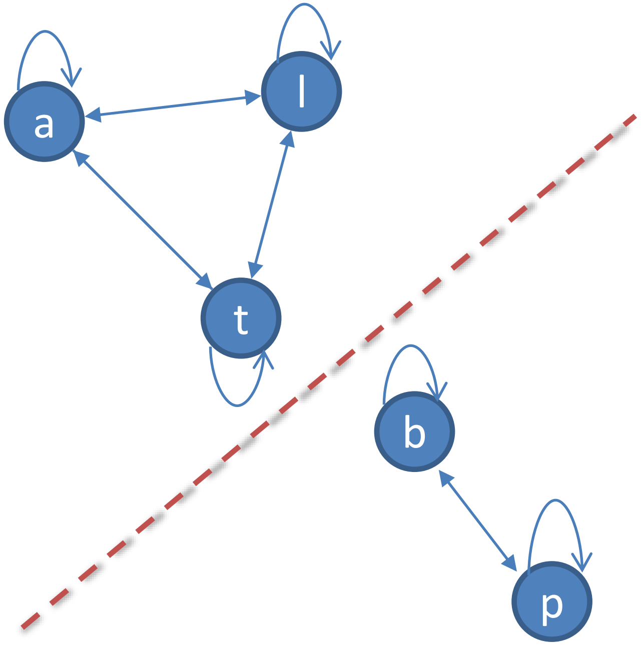
$\Downarrow$

Which products customers that are similar to John like

Complication

A^T A (Product x Product)

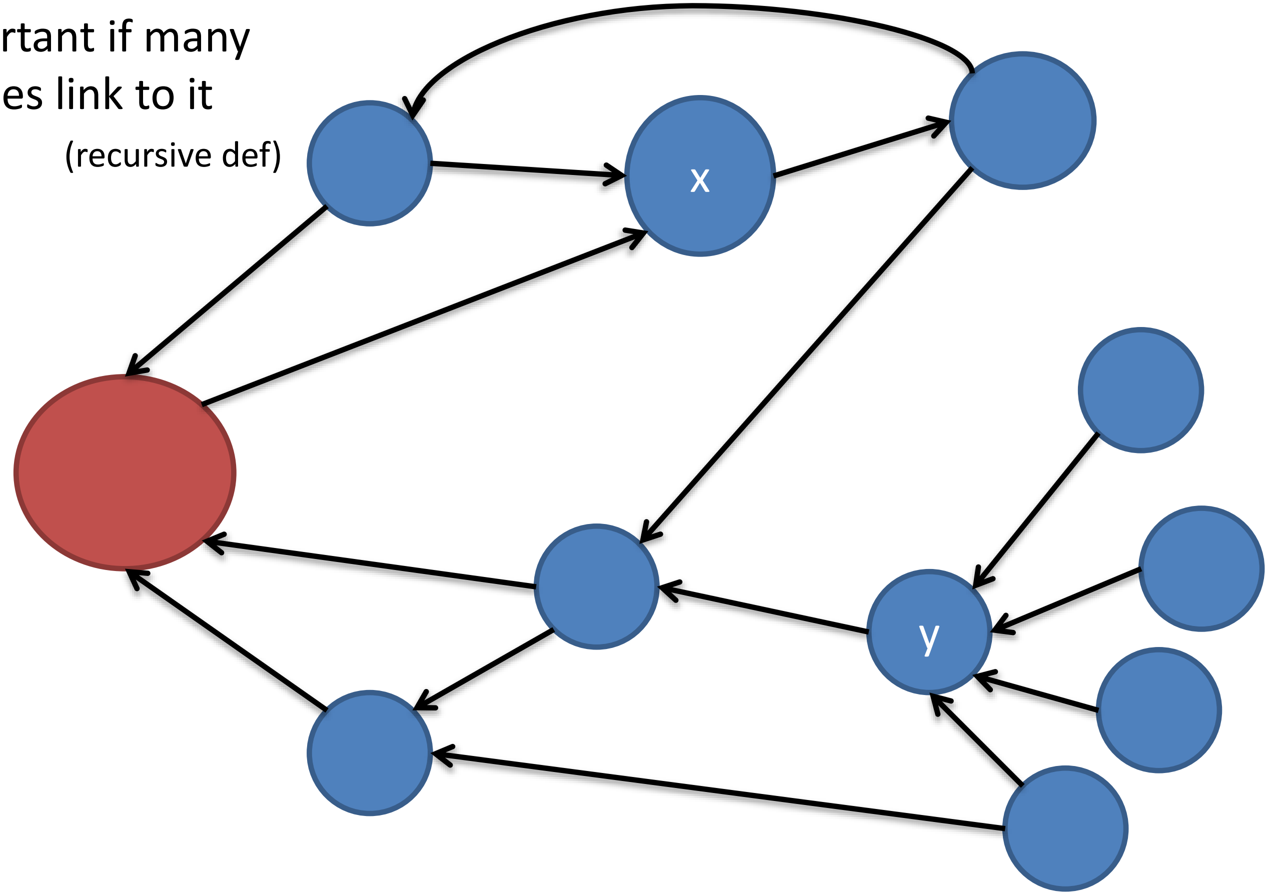
	a	l	t	b	p
a	31	31	31	0	0
l	31	31	31	0	0
t	31	31	31	0	0
b	0	0	0	14	14
p	0	0	0	14	14



PageRank* (informal)

Page x is important if many important pages link to it

(recursive def)

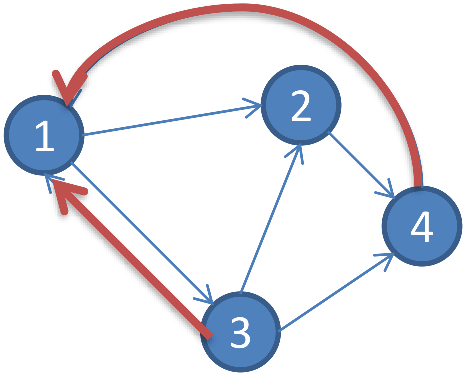


Page y has more incoming links but lower PageRank than page x

*Larry Page and Sergey Brin

PageRank
(less informal but not complete)

- Let matrix W denote the web graph



From:

To:

		1/3	1
1/2			
1/2		1/3	
	1	1/3	

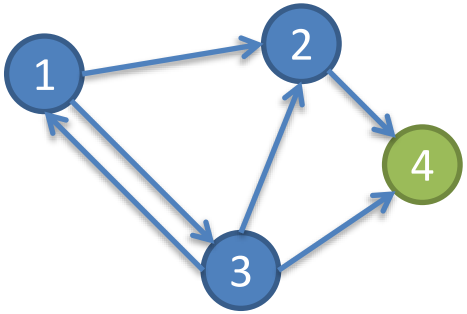
$p_1 = 1/3p_3+p_4$

- Look for vector p such that W.p = 1.p

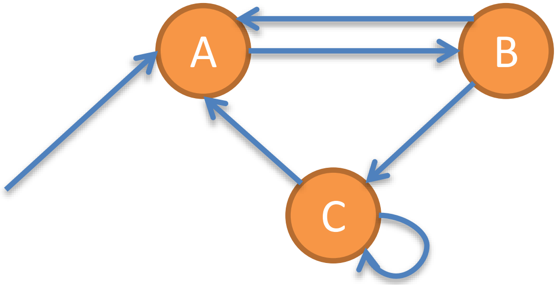
$$\begin{array}{|c|c|c|c|} \hline & & 1/3 & 1 \\ \hline 1/2 & & & \\ \hline 1/2 & & 1/3 & \\ \hline & 1 & 1/3 & \\ \hline \end{array} \cdot \begin{array}{|c|} \hline p_1 \\ \hline p_2 \\ \hline p_3 \\ \hline p_4 \\ \hline \end{array} = \begin{array}{|c|} \hline p_1 \\ \hline p_2 \\ \hline p_3 \\ \hline p_4 \\ \hline \end{array}$$

Complications

- Some (even important) pages may have no outgoing links
  - PageRank leaks out from these nodes instead of being re-distributed back to the net



- Existence of cycles
  - spider traps/closed communities
  - PageRank get's "trapped"



Summary

- SVD: algebraic tool that has many potential uses
  - Dimensionality reduction
  - Indexing (LSI)
  - Visualization/clustering of high-dimensional objects
  - Similarity computations/outlier detection
  - Rule mining, treatment of missing/wrong values