Applied Data Science

Session 30: Dimension Reduction with PCA & SVD

Dr. Soharab Hossain Shaikh

1

Principal Component Analysis (PCA)

2

Principal Component Analysis

Principal Component Analysis, or **PCA** for short, is a method for reducing the dimensionality of data.

It can be thought of as a ***projection method where data with p-columns (features) is projected into a subspace with k of fewer columns***, whilst retaining the essence of the original data.

Steps:

Begin by standardizing the data.

- Generate the Covariance Matrix.
- Perform Eigen Decomposition
- Sort the Eigen pairs in descending order and select the largest one.

3

Principal Component Covariance Matrix

- Variance is measured within the dimensions and covariance is among the dimensions.
- Express total variance (variance and cross variance between dimensions as a matrix)
- Covariance Matrix is a mathematical representation of the total variance of individual dimension and across dimensions.

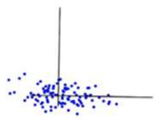
$$\mathrm{var}(X) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})}{(n-1)}$$

$$\mathrm{cov}(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

4

Improving Performance through PCA

- The mean is subtracted from all the points on both dimensions.
- The dimensions are transformed using algebra into new set of dimensions.
- The transformation is a rotation of axes in mathematical space.



5

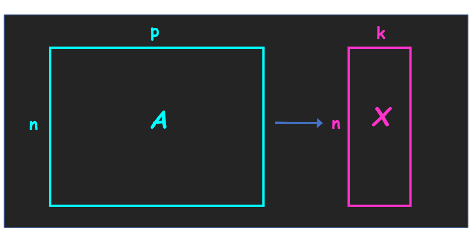
PCA for Dimensionality Reduction

- PCA can also be used to reduce dimensions.
- Arrange all Eigen vectors along with corresponding eigen values in descending order of Eigen values.
- Plot a cumulative Eigen-value graph.
- Eigen vectors with insignificant contributions to total Eigen values can be removed from analysis.

6

Data Reduction

Summarization of data with many (p) variables by a smaller set of (k) derived (synthetic, composite) variables.



7

Data Reduction

> “Residual” variation is information in *A* that is not retained in *X*

> Balancing between

- Clarity of Representation: ease of understanding
- Oversimplification: loss of important or relevant information.

8

Principal Component Analysis (PCA)

- Takes a data matrix of *n* objects by *p* variables, which may be correlated, and summarizes it by uncorrelated axes (principal components or principal axes) that are linear combinations of the original *p* variables.
- the first *k* components display as much as possible of the variation among objects.

9

Geometric Rationale of PCA

- Objects are represented as a cloud of *n* points in a multidimensional feature space with an axis for each of the *p* variables.
- The centroid of the points is defined by the mean of each variable.
- The variance of each variable is the average squared deviation of its *n* values around the mean of that variable.

$$V_i = \frac{1}{n-1}\sum_{m=1}^{n}\left(X_{im} - \overline{X}_i\right)^2$$

10

Geometric Rationale of PCA

> Degree to which the variables are linearly correlated is represented by their covariances.

$$C_{ij} = \frac{1}{n-1}\sum_{m=1}^{n}\left(X_{im} - \overline{X}_i\right)\left(X_{jm} - \overline{X}_j\right)$$

Covariance of variables *i* and *j*

Sum over all *n* objects

Value of variable *i* in object *m*

Mean of variable *i*

Value of variable *j* in object *m*

Mean of variable *j*

11

Geometric Rationale of PCA

- objective of PCA is to rigidly rotate the axes of this *p*-dimensional space to new positions (principal axes) that have the following properties:
  - ordered such that principal axis 1 has the highest variance, axis 2 has the next highest variance, ... , and axis *p* has the lowest variance
  - covariance among each pair of the principal axes is zero (the principal axes are uncorrelated).

12

The Dissimilarity Measure Used in PCA is Euclidean Distance

- PCA uses Euclidean Distance calculated from the *p* variables as the measure of dissimilarity among the n objects
- PCA derives the best possible *k* dimensional ($k < p$) representation of the Euclidean distances among objects.

13

Generalization to *p*-dimensions

- In practice nobody uses PCA with only 2 variables
- The algebra for finding principal axes readily generalizes to *p* variables
- PC 1 is the direction of maximum variance in the *p*-dimensional cloud of points
- PC 2 is in the direction of the next highest variance, subject to the constraint that it has zero covariance with PC 1.

14

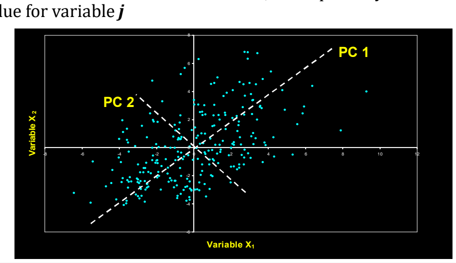
Generalization to *p*-dimensions

- PC3 is in the direction of the next highest variance, subject to the constraint that it has zero covariance with both PC1 and PC2
- and so on... up to PC*p*

15

PCA

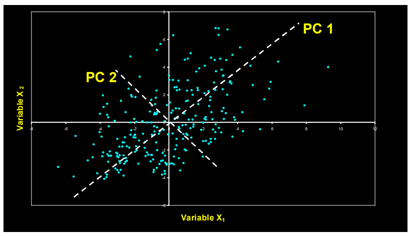
- each principal axis is a linear combination of the original variables
- $\mathbf{PC_j = a_{i1}X_1 + a_{i2}X_2 + \ldots a_{in}X_n}$
- a_{ij}'s are the coefficients for factor ***i***, multiplied by the measured value for variable ***j***



16

PCA

- PC axes are a rigid rotation of the original variables
- PC 1 is simultaneously the direction of maximum variance and a least-squares "line of best fit" (squared distances of points away from PC 1 are minimized).



17

Generalization to *p*-dimensions

> if we take the first k principal components, they define the k-dimensional "hyperplane of best fit" to the point cloud of the total variance of all p variables:

- PCs 1 to k represent the maximum possible proportion of that variance that can be displayed in k dimensions
- *i.e.* the squared Euclidean distances among points calculated from their coordinates on PCs 1 to k are the best possible representation of their squared Euclidean distances in the full p dimensions.

18

Eigen-values and Eigen-vectors

- An eigen-values and eigen-vectors of the square matrix A are a *scalar* λ and a non-zero vector **v** that satisfy $\mathbf{Av} = \boldsymbol{\lambda}\mathbf{v}$
- In this equation, **A** is a n-by-n matrix, **v** is non-zero n-by-1 **vector,** and $\boldsymbol{\lambda}$ is the **scalar** (which might be either real or complex).
- Any value of the $\boldsymbol{\lambda}$ for which this equation has a solution known as *eigen-values* of the matrix **A**.
- It is also called the **characteristic value.**
- The vector, **v**, which corresponds to this equation, is called eigen-vectors**.**
- The eigen-values problem can be written as
- $\mathbf{A.v-\lambda \cdot v=0}$
  $\mathbf{A \cdot v - \lambda \cdot I \cdot v=0}$
  $\mathbf{(A-\lambda * I) \cdot v=0}$
- If **v** is non-zero, this equation will only have the solutions if $\mathbf{det(A-\lambda \cdot I)}=0$
- This equation is called the characteristic equations of **A** and is a n^{th} order polynomial in $\boldsymbol{\lambda}$ with n roots. These roots are called the eigenvalue of **A**.
- We will only handle the case of n distinct roots; through which they may be repeated. For each eigenvalue, there will be eigenvectors for which the eigenvalue equations are true.

19

Example

If

$$A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$$

then the characteristic equation is

$$\left| A - \lambda . I = \left| \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right| \right| = 0$$

$$\left| \begin{bmatrix} -\lambda & 1 \\ -2 & -3 - \lambda \end{bmatrix} \right| = \lambda^2 + 3\lambda + 2 = 0$$

and the two eigenvalues are

$$\lambda_1 = -1, \lambda_2 = -2$$

20

Example

- All that's left is to find two eigenvectors.
- Let's find the eigenvector, v_1, connected with the eigenvalue, λ_1=-1, first.

$$A.v_1 = \lambda_1.v_1$$

$$(A\text{-}\lambda_1).v_1 = 0$$

$$\begin{bmatrix} -\lambda_1 & 1 \\ -2 & -3-\lambda_1 \end{bmatrix}.v_1 = 0$$

$$\begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix}.v_1 = \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix}.\begin{bmatrix} v_{1,1} \\ v_{1,2} \end{bmatrix} = 0$$

so clearly from the top-row of the equation, we get

$$v_{1,1} + v_{1,2} = 0$$

$$v_{1,1} = -v_{1,2}$$

If we take the second row, we will get

$$-2.v_{1,1} \pm 2.v_{1,2} = 0, \text{ so again}$$

$$v_{1,1} = -v_{1,2}$$

In this case, we find that the first eigenvector is any 2 components column vector in which the two items have equal magnitude and opposite sign.

$$v_1 = k_1 \begin{bmatrix} +1 \\ -1 \end{bmatrix}$$

where k_1 is an arbitrary constant. If we didn't have to use +1 and -1, we have used any two quantities of equal magnitude and opposite sign.

21

Example

Going through the same process for the second eigenvalue:

$$A.v_2 = \lambda_2.v_2$$

$$(A\text{-}\lambda_2).v_2 = \begin{bmatrix} -\lambda_2 & 1 \\ -2 & -3-\lambda_2 \end{bmatrix}.v_2 = \begin{bmatrix} 2 & 1 \\ -2 & -1 \end{bmatrix}.\begin{bmatrix} v_{2,1} \\ v_{2,2} \end{bmatrix} = 0$$

$$2.v_{2,1} + 1.v_{2,2} = 0, \quad \text{from bottom line: } -2.v_{2,1} - 1.v_{2,2} = 0$$

$$2.v_{2,1} = -v_{2,2}$$

$$v_2 = k_2 \begin{bmatrix} +1 \\ -2 \end{bmatrix}$$

Again, the choice of the +1 and -2 for the eigenvectors was arbitrary; only their ratio is essential.

22

The Algebra of PCA

- First step is to calculate the cross-products matrix of variances and covariances among every pair of the *p* variables
- Covariance Matric is a square matrix
- Diagonals are the variances, off-diagonals are the covariances.

	X_1	X_2
X_1	6.6707	3.4170
X_2	3.4170	6.2384

Variance-covariance Matrix

23

The Algebra of PCA

- in matrix notation, this is computed as

$$S = X'X$$

- where X is the *n* x *p* data matrix, with each variable centered. X' is the transpose of X.

	X_1	X_2
X_1	6.6707	3.4170
X_2	3.4170	6.2384

Variance-covariance Matrix

24

Manipulating Matrices

- Transposing: could change the columns to rows or the rows to columns

$$X = \begin{bmatrix} 10 & 0 & 4 \\ 7 & 1 & 2 \end{bmatrix} \qquad X' = \begin{bmatrix} 10 & 7 \\ 0 & 1 \\ 4 & 2 \end{bmatrix}$$

- Multiplying Matrices
  - must have the same number of columns in the pre-multiplicand matrix as the number of rows in the post-multiplicand matrix

25

The Algebra of PCA

- sum of the diagonals of the variance-covariance matrix is called the **Trace**.
- **Trace** represents the *total variance in the data*
- it is the mean squared Euclidean distance between each object and the centroid in *p*-dimensional space.

	X_1	X_2
X_1	6.6707	3.4170
X_2	3.4170	6.2384

Trace = 12.9091

26

The Algebra of PCA

- finding the principal axes involves Eigen analysis of the cross-products matrix (S)
- the Eigen values (latent roots) of S are solutions (λ) to the characteristic equation

$$\left|\mathbf{S} - \lambda\mathbf{I}\right| = 0$$

Determinant

27

The Algebra of PCA

- the *eigenvalues, λ_1, λ_2, ... λ_p are the variances of the coordinates on each principal component axis*
- the sum of all *p* eigenvalues equals the trace of S (the sum of the variances of the original variables).

	X_1	X_2
X_1	6.6707	3.4170
X_2	3.4170	6.2384

$\lambda_1 = 9.8783$
$\lambda_2 = 3.0308$

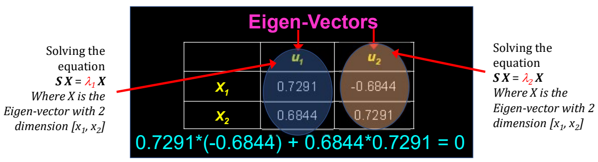
Trace = 12.9091

Note: $\lambda_1 + \lambda_2 = 12.9091$

28

The Algebra of PCA

- each eigenvector consists of p values which represent the "contribution" of each variable to the principal component axis
- eigenvectors are uncorrelated (orthogonal)
  - their cross-products are zero.



29

The Algebra of PCA

- coordinates of each object i on the k^{th} principal axis, known as the scores on PC k, are computed as

$$z_{ki} = u_{1k}x_{1i} + u_{2k}x_{2i} + \cdots + u_{pk}x_{pi}$$

- where Z is the n x k matrix of PC scores
- X is the n x p centered data matrix
- U is the p x k matrix of eigenvectors.

30

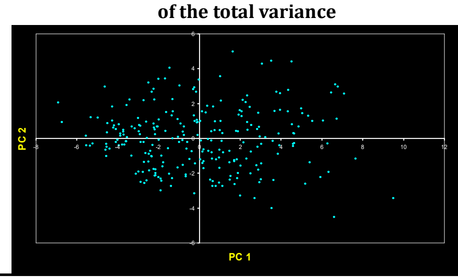
The Algebra of PCA

- variance of the scores on each PC axis is equal to the corresponding eigenvalue for that axis
- the eigenvalue represents the variance displayed ("explained" or "extracted") by the k^{th} axis
- the sum of the first k eigenvalues is the variance explained by the k-dimensional ordination.

31

The Algebra of PCA

λ_1 = 9.8783 λ_2 = 3.0308 Trace = 12.9091
PC 1 displays ("explains") 9.8783/12.9091 = 76.5% of the total variance



32

The Algebra of PCA

> The cross-products matrix computed among the *p* principal axes has a simple form:
- all off-diagonal values are zero (the principal axes are uncorrelated)
- the diagonal values are the eigenvalues.

	PC_1	PC_2
PC_1	9.8783	0.0000
PC_2	0.0000	3.0308

Variance-covariance Matrix
of the PC axes

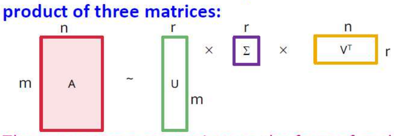
33

Singular Value Decomposition (SVD)

34

Reducing Matrix Dimension

- **Gives a decomposition of any matrix into a product of three matrices:**

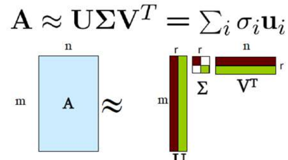


- There are strong constraints on the form of each of these matrices
  - Results in a unique decomposition
- From this decomposition, you can choose any number r of intermediate concepts (latent factors) in a way that minimizes the reconstruction error

35

SVD Definition

$$\mathbf{A} \approx \mathbf{U}\Sigma\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^{\mathrm{T}}$$

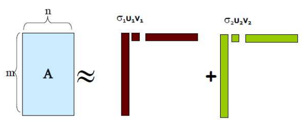


- **A**: Input data matrix
  - *m* x *n* matrix (e.g., *m* documents, *n* terms)
- **U**: Left singular vectors
  - *m* x *r* matrix (*m* documents, *r* concepts)
- **Σ**: Singular values
  - *r* x *r* diagonal matrix (strength of each 'concept') (*r* : rank of the matrix A)
- **V**: Right singular vectors
  - *n* x *r* matrix (*n* terms, *r* concepts)

36

SVD Definition

$$\mathbf{A} \approx \mathbf{U}\Sigma\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^T$$



If we set $\sigma_2 = 0$, then the green columns may as well not exist.

σ_i … scalar
$\mathbf{u}_i$ … vector
$\mathbf{v}_i$ … vector

37

SVD Properties

It is **always** possible to decompose a real matrix ***A*** into $A = U \Sigma V^T$, where

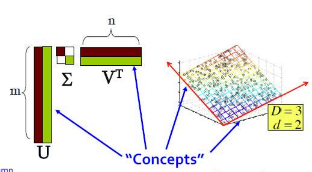
- U, Σ, V: unique
- U, V: column orthonormal
  - $U^T U = I$; $V^T V = I$ (*I*: identity matrix)
  - (Columns are orthogonal unit vectors)
- Σ: diagonal
  - Entries (**singular values**) are non-negative, and sorted in decreasing order ($\sigma_1 \geq \sigma_2 \geq \ldots \geq 0$)

38

SVD – Example – User-to-Movies

- **Consider a matrix. What does SVD do?**

$$\begin{array}{c|ccccc}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
 & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
\text{Romance} & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
 & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
 & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array} =$$



Ratings matrix where each column corresponds to a movie and each row to a user. First 4 users prefer SciFi, while others prefer Romance.

39

SVD – Example – User-to-Movies

- **$A = U \Sigma V^T$ - example: Users to Movies**

$$\begin{array}{c|ccccc}
 & \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\hline
\text{SciFi} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
 & \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
 & \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
 & \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
\text{Romance} & 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
 & 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
 & 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}$$

40

SVD – Example – User-to-Movies

- $A = U \Sigma V^T$ - example: Users to Movies

SciFi-concept

Romance-concept

$$\begin{array}{c|c} & \begin{matrix} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{matrix} \\ \begin{matrix} \text{SciFi} \\ \\ \\ \\ \text{Romance} \\ \\ \\ \end{matrix} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \end{array} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

41

SVD – Example – User-to-Movies

- $A = U \Sigma V^T$ - example: *U* is "user-to-concept" factor matrix

SciFi-concept

Romance-concept

$$\begin{array}{c|c} & \begin{matrix} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{matrix} \\ \begin{matrix} \text{SciFi} \\ \\ \\ \\ \text{Romance} \\ \\ \\ \end{matrix} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \end{array} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

42

SVD – Example – User-to-Movies

- $A = U \Sigma V^T$ - example:

SciFi-concept

"strength" of the SciFi-concept

$$\begin{array}{c|c} & \begin{matrix} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{matrix} \\ \begin{matrix} \text{SciFi} \\ \\ \\ \\ \text{Romance} \\ \\ \\ \end{matrix} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \end{array} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

43

SVD – Example – User-to-Movies

- $A = U \Sigma V^T$ - example: *V* is "movie-to-concept" factor matrix

SciFi-concept

$$\begin{array}{c|c} & \begin{matrix} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{matrix} \\ \begin{matrix} \text{SciFi} \\ \\ \\ \\ \text{Romance} \\ \\ \\ \end{matrix} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \end{array} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SciFi-concept

44

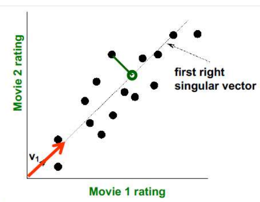
SVD – Interpretation

Movies, users and concepts:

- U: user-to-concept matrix
- V: movie-to-concept matrix
- Σ: its diagonal elements: 'strength' of each concept

45

SVD – Dimension Reduction

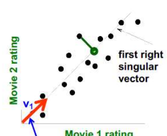


- Instead of using two coordinates (x, y) to describe point positions, let's use only one coordinate
- Point's position is its location along vector v_1

46

SVD – Dimension Reduction

- A = U Σ V^T - example:
  - U: "user-to-concept" matrix
  - V: "movie-to-concept" matrix

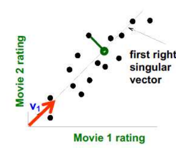


$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

47

SVD – Dimension Reduction

- A = U Σ V^T - example:



variance ('spread') on the v_1 axis

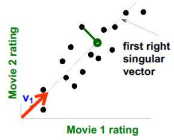
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

48

SVD – Dimension Reduction

$A = U \Sigma V^T$ - example:

- **$U \Sigma$:** Gives the coordinates of the points in the projection axis



$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}$$

Projection of users on the "Sci-Fi" axis $U \Sigma$:

$$\begin{bmatrix} \boxed{1.61} & 0.19 & -0.01 \\ 5.08 & 0.66 & -0.03 \\ 6.82 & 0.85 & -0.05 \\ 8.43 & 1.04 & -0.06 \\ 1.86 & -5.60 & 0.84 \\ 0.86 & -6.93 & -0.87 \\ 0.86 & -2.75 & 0.41 \end{bmatrix}$$

49

SVD – Dimension Reduction

More details

- **Q: How is dim. reduction done?**

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

50

SVD – Interpretation

More details

- **Q: How exactly is dim. reduction done?**
- **A: Set smallest singular values to zero**

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & \cancel{1.3} \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

51

SVD – Interpretation

More details

- **Q: How exactly is dim. reduction done?**
- **A: Set smallest singular values to zero**

This is Rank 2 approximation to A. We could also do Rank 1 approx. The larger the rank the more accurate the approximation.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 0.13 & 0.02 & \cancel{-0.01} \\ 0.41 & 0.07 & \cancel{-0.03} \\ 0.55 & 0.09 & \cancel{-0.04} \\ 0.68 & 0.11 & \cancel{-0.05} \\ 0.15 & -0.59 & \cancel{0.65} \\ 0.07 & -0.73 & \cancel{-0.67} \\ 0.07 & -0.29 & \cancel{0.32} \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & \cancel{1.3} \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ \cancel{0.40} & \cancel{-0.80} & \cancel{0.40} & \cancel{0.09} & \cancel{0.09} \end{bmatrix}$$

52

SVD – Interpretation

More details

- **Q: How exactly is dim. reduction done?**
- **A: Set smallest singular values to zero**

This is Rank 2 approximation to A. We could also do Rank 1 approx. The larger the rank the more accurate the approximation.

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} \approx \begin{bmatrix} \mathbf{0.13} & 0.02 \\ \mathbf{0.41} & 0.07 \\ \mathbf{0.55} & 0.09 \\ \mathbf{0.68} & 0.11 \\ 0.15 & \mathbf{-0.59} \\ 0.07 & \mathbf{-0.73} \\ 0.07 & \mathbf{-0.29} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 \\ 0 & \mathbf{9.5} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \end{bmatrix}$$

53

SVD – Interpretation

More details

- **Q: How exactly is dim. reduction done?**
- **A: Set smallest singular values to zero**

This is Rank 2 approximation to A. We could also do Rank 1 approx. The larger the rank the more accurate the approximation

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ \mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\ \mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\ \mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\ 0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\ 0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\ 0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2} \end{bmatrix} \approx \begin{bmatrix} \mathbf{0.92} & \mathbf{0.95} & \mathbf{0.92} & 0.01 & 0.01 \\ \mathbf{2.91} & \mathbf{3.01} & \mathbf{2.91} & -0.01 & -0.01 \\ \mathbf{3.90} & \mathbf{4.04} & \mathbf{3.90} & 0.01 & 0.01 \\ \mathbf{4.82} & \mathbf{5.00} & \mathbf{4.82} & 0.03 & 0.03 \\ 0.70 & \mathbf{0.53} & 0.70 & \mathbf{4.11} & \mathbf{4.11} \\ -0.69 & 1.34 & -0.69 & \mathbf{4.78} & \mathbf{4.78} \\ 0.32 & \mathbf{0.23} & 0.32 & \mathbf{2.01} & \mathbf{2.01} \end{bmatrix}$$

Reconstructed data matrix B

Reconstruction Error is quantified by the Frobenius norm:

$\|M\|_F = \sqrt{\Sigma_{ij} M_{ij}^2}$

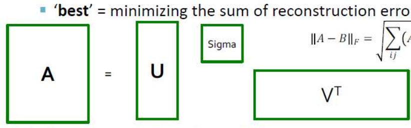
$\|A-B\|_F = \sqrt{\Sigma_{ij} (A_{ij}-B_{ij})^2}$ is "small"

54

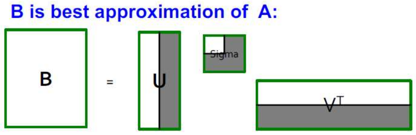
SVD – Best Low Rank Approximation

- **Fact: SVD gives 'best' axis to project on:**
  - '**best**' = minimizing the sum of reconstruction errors

$$\|A - B\|_F = \sqrt{\sum_{ij} (A_{ij} - B_{ij})^2}$$



B is best approximation of A:



55

SVD – Conclusions

- **SVD: A= U Σ V^T: unique**
  - **U**: user-to-concept factors
  - **V**: movie-to-concept factors
  - **Σ** : strength of each concept
- **Q: So what's a good value for r (# of latent factors)?**
- Let the *energy* of a set of singular values be the sum of their squares.
- Pick r so the retained singular values have at least 90% of the total energy.
- **Back to our example:**
  - With singular values 12.4, 9.5, and 1.3, total energy = 245.7
  - If we drop 1.3, whose square is only 1.7, we are left with energy 244, or over 99% of the total

56

SVD – How to Compute?

57

Finding Eigen Pairs

- **How do we actually compute SVD?**
- First we need a method for finding the **principal eigenvalue** (the largest one) and the corresponding **eigenvector** of a symmetric matrix
  - M is *symmetric* if $m_{ij} = m_{ji}$ for all i and j
- **Method:**
  - Start with any "guess eigenvector" $\boldsymbol{x}_0$
  - Construct $\boldsymbol{x}_{k+1} = \frac{M\boldsymbol{x}_k}{||M\boldsymbol{x}_k||}$ for $k = 0, 1, \ldots$
    - || … || denotes the Frobenius norm
  - Stop when consecutive $\boldsymbol{x}_k$ show little change

58

Iterative Eigen Vector

$$M = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \quad \mathbf{x}_0 = \begin{matrix} 1 \\ 1 \end{matrix}$$

$$\frac{M\mathbf{x}_0}{||M\mathbf{x}_0||} = \begin{pmatrix} 3 \\ 5 \end{pmatrix} / \sqrt{34} = \begin{pmatrix} 0.51 \\ 0.86 \end{pmatrix} = \mathbf{x}_1$$

$$\frac{M\mathbf{x}_1}{||M\mathbf{x}_1||} = \begin{pmatrix} 2.23 \\ 3.60 \end{pmatrix} / \sqrt{17.93} = \begin{pmatrix} 0.53 \\ 0.85 \end{pmatrix} = \mathbf{x}_2$$

59

Finding the Principal Eigen Value

- Once you have the principal eigenvector $\boldsymbol{x}$, you find its eigenvalue λ by $\lambda = \boldsymbol{x}^T M \boldsymbol{x}$.
  - In proof: We know $\boldsymbol{x}\lambda = M\boldsymbol{x}$ if λ is the eigenvalue; multiply both sides by $\boldsymbol{x}^T$ on the left.
  - Since $\boldsymbol{x}^T\boldsymbol{x} = 1$ we have $\lambda = \boldsymbol{x}^T M \boldsymbol{x}$
- **Example:** If we take $\mathbf{x}^T = [0.53, 0.85]$, then

$$\lambda = [0.53\ 0.85] \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 0.53 \\ 0.85 \end{bmatrix} = 4.25$$

60

Finding More Eigen Pairs

- Eliminate the portion of the matrix M that can be generated by the first eigenpair, λ and $\boldsymbol{x}$:
  $$M^{*} := M - \lambda\, x\, x^{T}$$
- Recursively find the principal eigenpair for M^{*}, eliminate the effect of that pair, and so on
- **Example:**

$$M^{*} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} - 4.25 \begin{bmatrix} 0.53 \\ 0.85 \end{bmatrix} [0.53\ 0.85] = \begin{bmatrix} -0.19 & 0.09 \\ 0.09 & 0.07 \end{bmatrix}$$

61

How to Compute the SVD?

- **Start by supposing $A = U\Sigma V^{T}$**
- $A^{T} = (U\Sigma V^{T})^{T} = (V^{T})^{T}\Sigma^{T}U^{T} = V\Sigma U^{T}$
  - **Why?** (1) Rule for transpose of a product; (2) the transpose of the transpose and the transpose of a diagonal matrix are both the identity functions
- $\boldsymbol{A^{T}A = V\Sigma U^{T}U\Sigma V^{T} = V\Sigma^{2}V^{T}}$
  - **Why?** U is orthonormal, so $U^{T}U$ is an identity matrix
  - Also note that Σ^{2} is a diagonal matrix whose i-th element is the square of the i-th element of Σ
- $\boldsymbol{A^{T}AV = V\Sigma^{2}V^{T}V = V\Sigma^{2}}$
  - **Why?** V is also orthonormal

62

Computing SVD

- Starting with $\boldsymbol{(A^{T}A)V = V\Sigma^{2}}$
  - **Note** that therefore the i-th column of V is an eigenvector of $A^{T}A$, and its eigenvalue is the i-th element of Σ^{2}
- Thus, we can find V and Σ by finding the eigenpairs for $A^{T}A$
  - Once we have the eigenvalues in Σ^{2}, we can find the singular values by taking the square root of these eigenvalues
- Symmetric argument, AA^{T} gives us U

63

SVD – Time Complexity

- **To compute the full SVD using specialized methods:**
  - **O(nm²)** or **O(n²m)** (whichever is less)
- **But:**
  - Less work, if we just want singular values
  - or if we want the first *k* singular vectors
  - or if the matrix is sparse
- **Implemented in** linear algebra packages like
  - LINPACK, Matlab, SPlus, Mathematica ...

64

How to Query?

- **Q: Find users that like 'Matrix'**
- **A: Map query into a 'concept space' – how?**

$$
\begin{array}{c}
 \\ \uparrow \\ \text{SciFi} \\ \downarrow \\ \uparrow \\ \text{Romance} \\ \downarrow
\end{array}
\begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix}
\mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
\mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
\mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
\mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}
$$

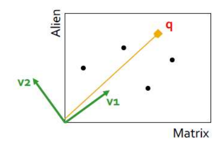
65

How to Query?

- **Q: Find users that like 'Matrix'**
- **A: Map query into a 'concept space' – how?**

$$
q = \begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix}
\end{array}
$$

Project into concept space:
Inner product with each 'concept' vector $\mathbf{v_i}$



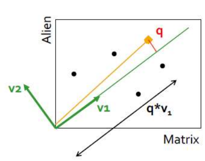
66

How to Query?

- **Q: Find users that like 'Matrix'**
- **A: Map query into a 'concept space' – how?**

$$
q = \begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix}
\end{array}
$$

Project into concept space:
Inner product with each 'concept' vector $\mathbf{v_i}$



67

How to Query?

Compactly, we have:
$q_{concept} = q\ V$

E.g.:

$$
q = \begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix}
\end{array}
\times
\underbrace{\begin{bmatrix}
0.56 & 0.12 \\
0.59 & -0.02 \\
0.56 & 0.12 \\
0.09 & -0.69 \\
0.09 & -0.69
\end{bmatrix}}_{\text{movie-to-concept factors (V)}}
=
\begin{bmatrix} \overset{\text{SciFi-concept}}{2.8} & 0.6 \end{bmatrix}
$$

68

How to Query?

- **How would the user *d* that rated ('Alien', 'Serenity') be handled?**
  $d_{concept}$ = d V

E.g.:

$$d = \begin{array}{c} \begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\ \begin{bmatrix} 0 & 4 & 5 & 0 & 0 \end{bmatrix} \end{array} \times \begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix} = \begin{bmatrix} 5.2 & 0.4 \end{bmatrix}$$

movie-to-concept factors (V)

SciFi-concept

69

How to Query?

- **Observation:** User ***d*** that rated ('*Alien*', '*Serenity*') will be **similar** to user **q** that rated ('*Matrix*'), although ***d*** and **q** have **zero ratings in common**!

Matrix, Alien, Serenity, Casablanca, Amelie

$$d = \begin{bmatrix} 0 & 4 & 5 & 0 & 0 \end{bmatrix} \dashrightarrow \begin{bmatrix} 5.2 & 0.4 \end{bmatrix}$$

$$q = \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix} \dashrightarrow \begin{bmatrix} 2.8 & 0.6 \end{bmatrix}$$

SciFi-concept

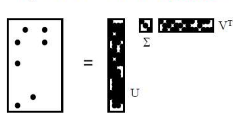
Zero ratings in common

Similarity > 0

70

SVD – Drawbacks

+ **Optimal low-rank approximation** in terms of Frobenius norm
− **Interpretability problem:**
  - A singular vector specifies a linear combination of all input columns or rows
− **Lack of sparsity:**
  - Singular vectors are **dense!**



71

72

Case-study

73

Bank Note Authentication

Data Source: https://archive.ics.uci.edu/ml/datasets/banknote+authentication

Abstract:

Data were extracted from images that were taken for the evaluation of an authentication procedure for banknotes.

Data Set Information:

Data were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images.

74

Case Study - PCA

Dataset information

1. variance of Wavelet Transformed image (continuous)
2. skewness of Wavelet Transformed image (continuous)
3. curtosis of Wavelet Transformed image (continuous)
4. entropy of image (continuous)
5. class (integer)

75

Let's go to the Coding Demo…

76

To be continued in the next session…..

77