NON-NEGATIVE MATRIX FACTORIZATION

AP

REVIEW OF SPECTRAL ANALYSIS/SVD

Decompose the data matrix and interpret its ‘first’ eigenvalues as concepts/topics for user and activity classification:

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

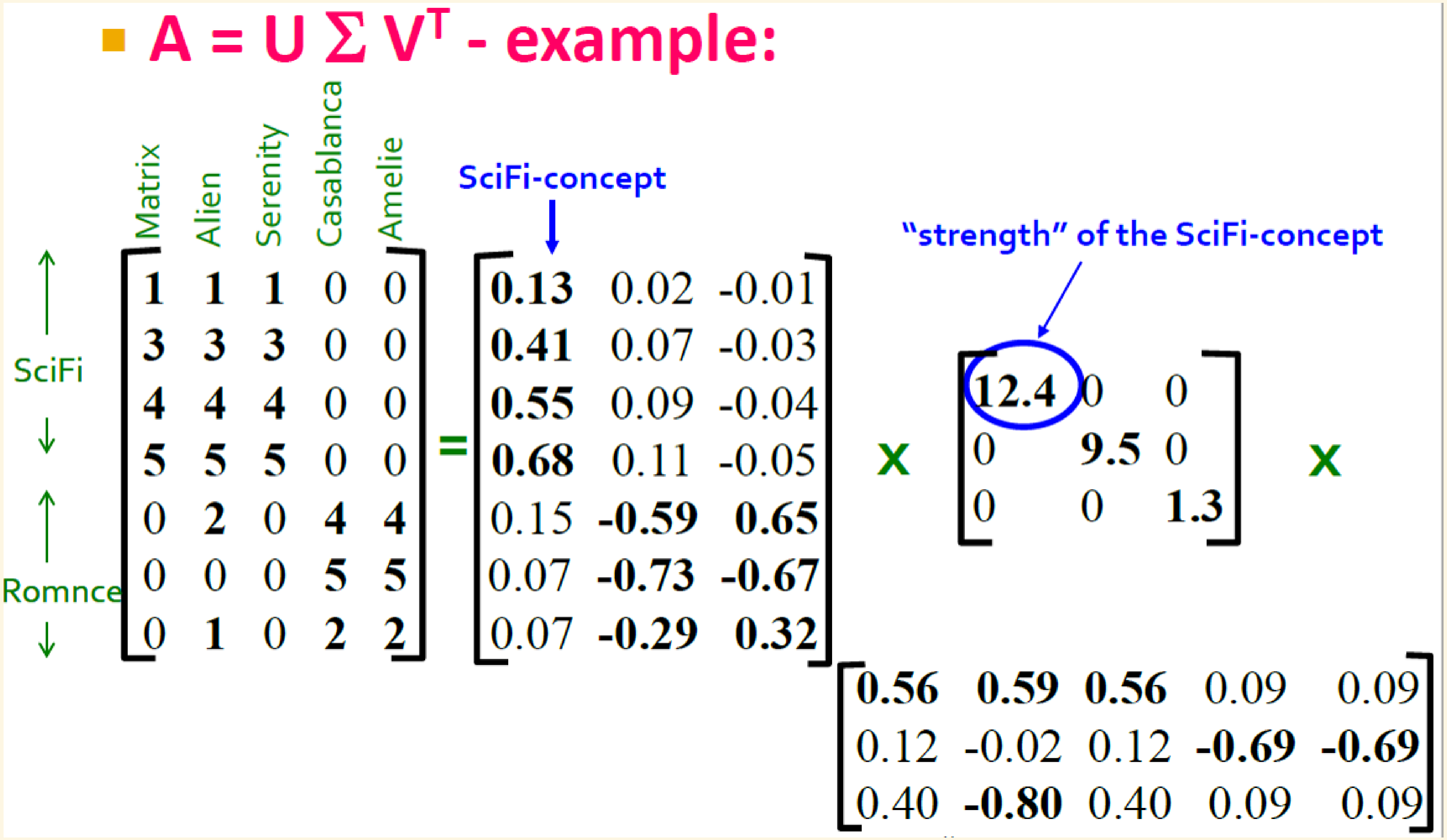
$U_{(m\times r)}$ is column-orthonormal: $u_i \cdot u_j^T = 0$

$V^T_{(r\times n)}$ is row-orthonormal: $v_i^T \cdot v_j = 0$

$\Sigma_{(r\times r)}$ (or $D\Sigma_{(r\times r)}$ is diagonal, σ_{ij} are the singular values

dimension *r* will depend on the no. of singular values found

HURDLE: INTERPRETATION OF NEGATIVE VALUES



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

SciFi-concept

"strength" of the SciFi-concept

$$\begin{array}{c}
\\ \uparrow \\ \text{SciFi} \\ \downarrow \\ \uparrow \\ \text{Romnce} \\ \downarrow
\end{array}
\begin{array}{ccccc}
\text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
\end{array}$$

$$\left[\begin{array}{ccccc}
\mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
\mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
\mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
\mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{array}\right]
=
\left[\begin{array}{ccc}
\overset{\text{SciFi-concept}}{\overset{\downarrow}{\mathbf{0.13}}} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{array}\right]
\times
\left[\begin{array}{ccc}
\boxed{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{array}\right]
\times
\left[\begin{array}{ccccc}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{array}\right]$$

With negative values from SVD we cannot distinguish between

i. lack of information,

ii. lack of interest or

ii. outright *repulsion*.

A non-negative decomposition of the activity matrix would be *interpretable:*

$$A_{(m\times n)} = P_{(m\times r)} \cdot Q_{(r\times n)}$$

- A: activity
- P: user participation to a topic
- Q: quantity of the topic in product

user/product profiling and reccommender sys. would be half-done already!

NON-NEGATIVE DECOMPOSITION

CONTEXT

Lee and Seung sought a new approach to the problem of learning images (how to recognise them) via visual patters:

reduce a specific image to the combination of known patters: face $\approx$ eyes, eyebrows, nose, chin etc.

Could each face be decomposed into basic, repeated visual patterns?

In the neural network model of brain activity, firing rates of neurons cannot be negative: patterns are either activated or ignored

Sparsity: only few patterns should be needed to re-create an image

Example dataset: faces (not Olivetti)

2429 mugshots, each having $19 \times 19 = 361$ pixels, rescaled to [0,1]: $V_{361 \times 2429}$

Lee-Seung fix $r = 49$ and compute $V_{361 \times 2429} = W_{361 \times 49} \times H_{49 \times 2429}$

For comparison, they also try PCA (diagonalisation of VV^T) and VQ

THE NUMERICAL PROBLEM

Istance: a non-negative matrix V

Solution: non-negative matrix factors W and H s.t.

$$V \approx W \cdot H$$

with $w_{ij}, h_{rs} \geq 0$

EXAMPLE

$$V_{3\times 3} = \begin{pmatrix} 4 & 0 & 2 \\ 0 & 3 & 1 \\ 2 & 1 & 3 \end{pmatrix}$$

choose $r = 2$

$$V_{3\times 3} \approx W_{3\times 2} \times H_{2\times 3} = \begin{pmatrix} 2 & 0 \\ 0 & 1.5 \\ 1 & 1 \end{pmatrix} \times \begin{pmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 4 & 0 & 2 \\ 0 & 3 & 1 \\ 2 & 1 & 3 \end{pmatrix} \approx \begin{pmatrix} 4 & 0 & 2 \\ 0 & 3 & 1.5 \\ 2 & 2 & 2 \end{pmatrix}$$

NOTATION

$$A = B \cdot C$$

Let $\mathbf{a}_i$ be the i-th column of A. It can be expressed as

$$\mathbf{a}_i = B \cdot \mathbf{c}_i$$

each col. of the result is seen as a linear combination of the cols. of B, with $\mathbf{c}_i$ supplying the *weights*:

$$\mathbf{a}_i = B \cdot \mathbf{c}_i = c_{1,i}\mathbf{b}_1 + c_{2,i}\mathbf{b}_2 + \cdots + c_{n,i}\mathbf{b}_n$$

EXAMPLE

$$\begin{pmatrix} 4 & 0 & 2 \\ 0 & 3 & 1 \\ 2 & 1 & 3 \end{pmatrix} \approx \begin{pmatrix} 2 & 0 \\ 0 & 1.5 \\ 1 & 1 \end{pmatrix} \times \begin{pmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 4 \\ 0 \\ 2 \end{pmatrix} \approx \begin{pmatrix} 2 & 0 \\ 0 & 1.5 \\ 1 & 1 \end{pmatrix} \times \begin{pmatrix} 2 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} 4 \\ 0 \\ 2 \end{pmatrix} \approx \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix} \cdot 2 + \begin{pmatrix} 0 \\ 1.5 \\ 1 \end{pmatrix} \cdot 0$$

INTERPRETABILITY OF NMF

HOW WE READ NMF

Let $\mathbf{v}_i$ be the i-th column of V.

If V is an activity m., $\mathbf{v}_i$ represent the *consumption* of i

$$v_i \approx W \cdot h_i$$

Consumption of i is given by a linear combination of the cols. of W, with h_i supplying the weights.

Each $\mathbf{w}_i$ is interpretable as a pattern (or mask)

[Lee & Seung, Nature, 1999]: “Learning the parts of objects by non-negative matrix factorization.”

$$\mathbf{v}_i \approx \mathbf{w}_1 \cdot h_{1,i} + \ldots \mathbf{w}_r \cdot h_{1,r}$$

> W can be regarded as containing a basis that is optimized for the linear approximation of the data in V.

> Since relatively few basis vectors are used to represent many data vectors, good approximation can only be achieved if the basis vectors discover structure that is latent in the data.

NORM NOTATION

Frobenius' element-wise norm:

$$||A_{m\times n}||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2} = \sqrt{\sum_{i,j} a_{ij}^2}$$

Notation for error:

$$||X - Y||_F^2 = ||X - Y||^2 = \sum_{i,j}(x_{ij} - y_{ij})^2$$

NMF AS ERROR-MINIMIZATION

COMPUTATIONAL PROBLEM

Input: $V_{n \times m}$

Minimize $||V - WH||^2$

subject to $W, H \geq 0.$

- choose the new dimension *r* s.t. $(n+m)r < nm$;
- calculate $W_{n \times r}$ and $H_{r \times m}$.

ASIDE: INFORMATION-THEORETIC VIEW

If the input matrix can be (somehow) normalised then we see the search for the perfect non-negative decomposition in terms of minimizing *divergence:*

$$D_I(X||Y) = \Sigma_{i,j}(x_{ij} \cdot \log(\frac{x_{ij}}{y_{ij}}) - x_{ij} + y_{ij}))$$

Minimize $D_I(V||WH)$

subject to $W, H \geq 0$.

Recommended version for sparse counting data.

The Kullback-Leibler divergence, D_{KL}, may also be used.

GRADIENT DESCENT MAY NOT FIND THE MINIMAL-ERROR SOLUTION

> Although [error func.] are convex in W only or H only, they are not convex in both variables together.
>
> Therefore it is unrealistic to expect an algorithm to solve [the problem] in the sense of finding global minima.

> However, there are many techniques from numerical optimization for finding local minima.
>
> Gradient descent is perhaps the simplest technique to implement, but convergence can be slow.

LEE-SEUNG'S METHOD

ITERATED ERROR BALANCING

1. start from random W and H
2. compute the error
3. update W and H with the **multiplicative update** rule
4. repeat normalisation of W: each column sum to 1

$$\boxed{\begin{gathered} W_{ia} \leftarrow W_{ia} \sum_{\mu} \frac{V_{i\mu}}{(WH)_{i\mu}} H_{a\mu} \\ W_{ia} \leftarrow \frac{W_{ia}}{\sum_{j} W_{ja}} \end{gathered}} \qquad \boxed{H_{a\mu} \leftarrow H_{a\mu} \sum_{i} W_{ia} \frac{V_{i\mu}}{(WH)_{i\mu}}}$$

MULTIPLICATIVE UPDATE

Classical Gradient descent: we *move around* by adding/subtracting some quantity

NMF: we *move around* by multiplying by a *local* error measure $\frac{v_{i\mu}}{(wh)_{i\mu}}$

$$\boxed{\begin{gathered} W_{ia} \leftarrow W_{ia} \sum_{\mu} \frac{V_{i\mu}}{(WH)_{i\mu}} H_{a\mu} \\ W_{ia} \leftarrow \frac{W_{ia}}{\sum_{j} W_{ja}} \end{gathered}} \quad \boxed{H_{a\mu} \leftarrow H_{a\mu} \sum_{i} W_{ia} \frac{V_{i\mu}}{(WH)_{i\mu}}}$$

$$W_{ia} \leftarrow W_{ia} \sum_{\mu} \boxed{\frac{V_{i\mu}}{(WH)_{i\mu}}} H_{a\mu}$$

$$W_{ia} \leftarrow \frac{W_{ia}}{\sum_{j} W_{ja}}$$

$$H_{a\mu} \leftarrow H_{a\mu} \sum_{i} W_{ia} \boxed{\frac{V_{i\mu}}{(WH)_{i\mu}}}$$

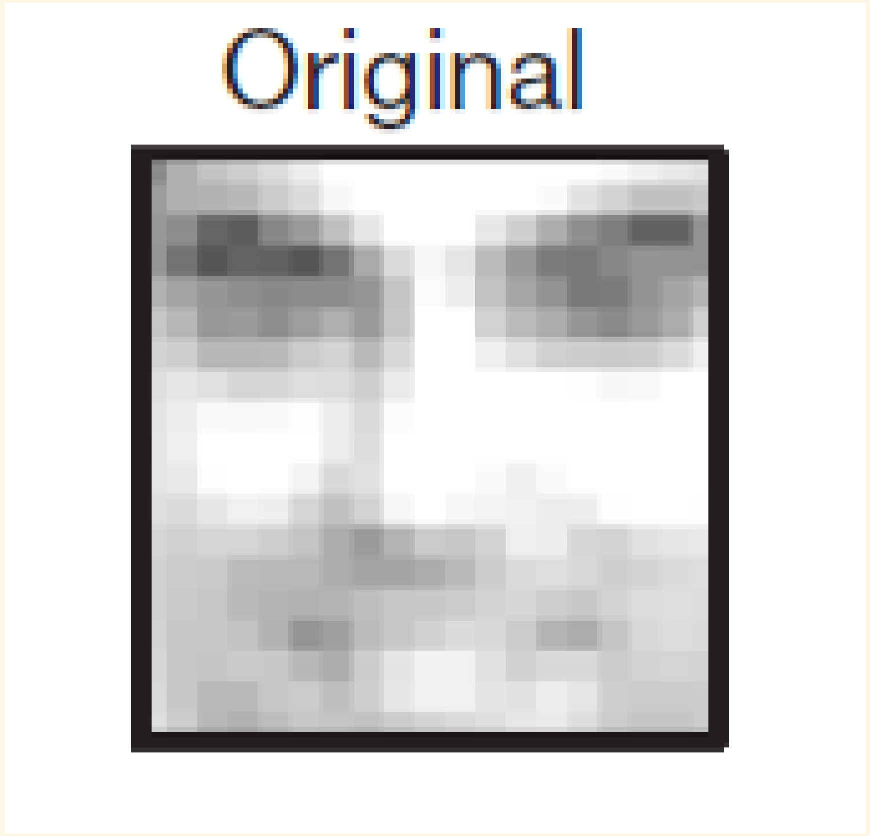
- through iteration, the $\frac{v_{i\mu}}{(wh)_{i\mu}}$ factors vanish and we stop.
- the update rules maintain non-negativity and force the $\mathbf{w}_i$ columns to sum to 1.

INTERPRETABILITY OF NMF

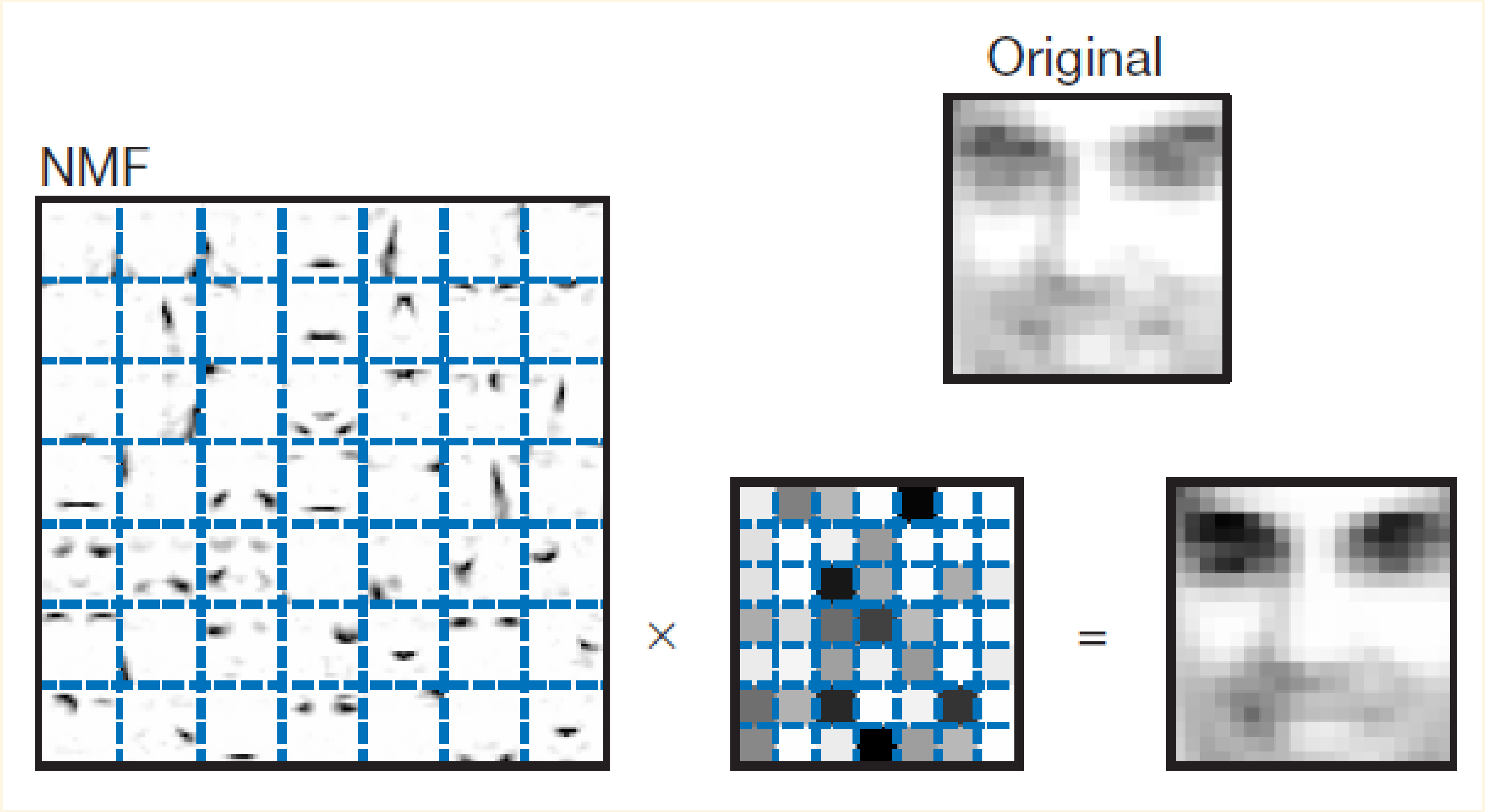
EXTRACTING COMMON PARTS: FACIAL FEATURES

Each column V_i represent a 19x19 mugshots

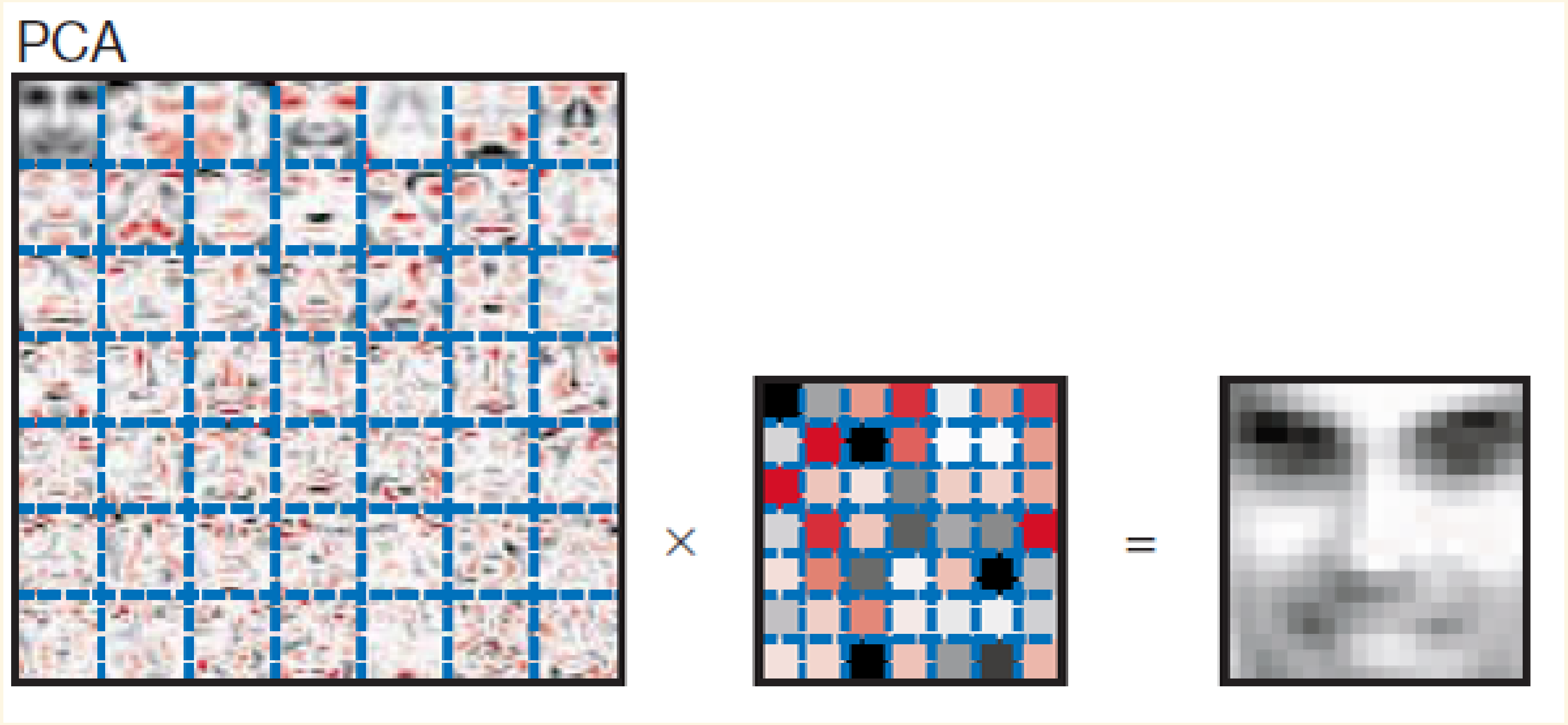
Lee-Seung render it as a linear combination of 49 common facial features, each weighted by an entry in the H matrix.



W AND H IN A 7X7 MONTAGE



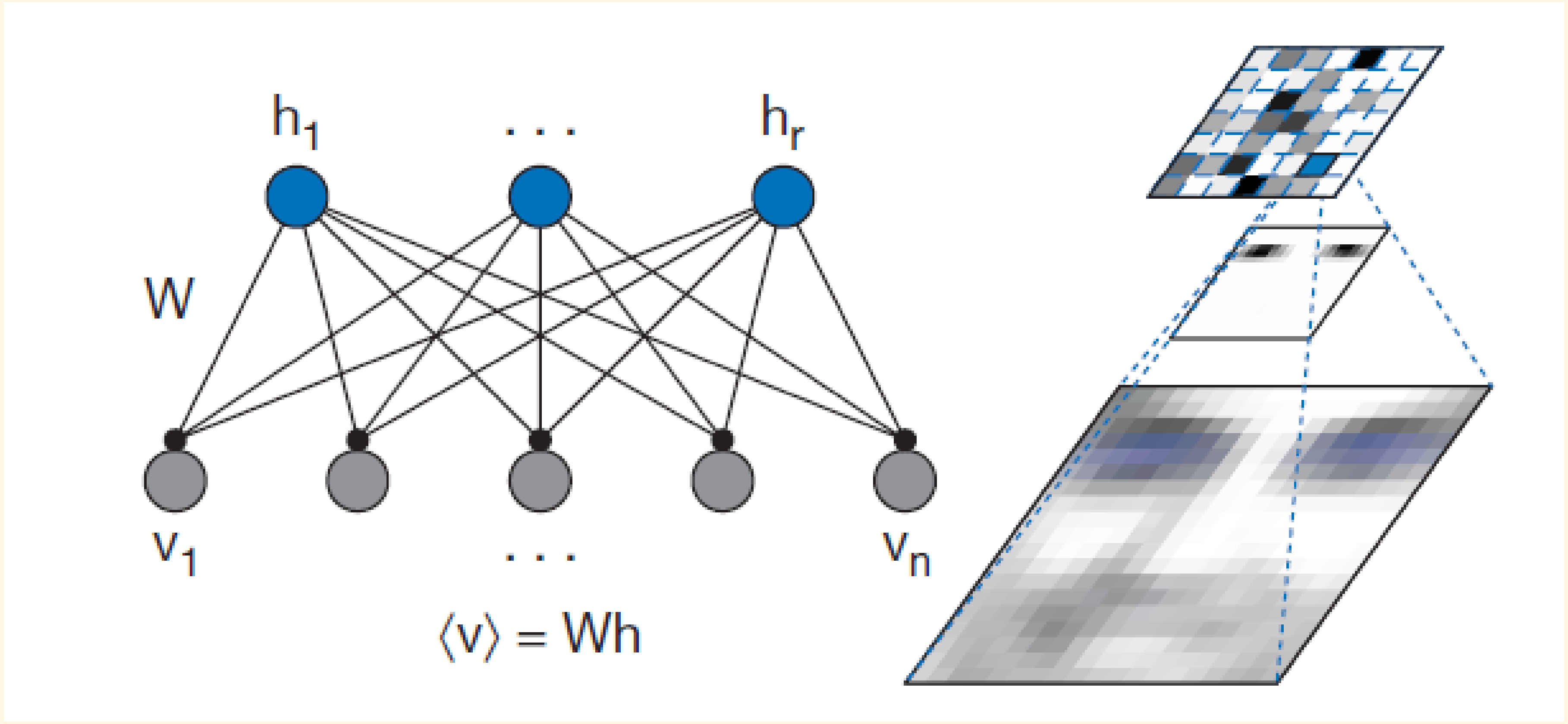
The *eigenfaces* might have negative values



A PROBABILISTIC HIDDEN-VARIABLES MODEL:

The columns. of W are *bases* that are combined to form the reconstruction

As in neural networks, the influence of $\mathbf{h}_a$ on $\mathbf{v}_i$ is represented by a connection **of strength** w_{ia}



NMF IN SCIKIT-LEARN

THE OLIVETTI FACES

Today Lee-Seung method is implemented in Scikit-learn via the `mu` (multiplicative update) method

try sklearn.decomposition.NMF

```
faces = fetch_olivetti_faces()

nmf_estimator = decomposition.NMF(solver='mu')

nmf_estimator.fit(faces)
```

For reference: a visual comparison of the decomposition methods available in Scikit-learn

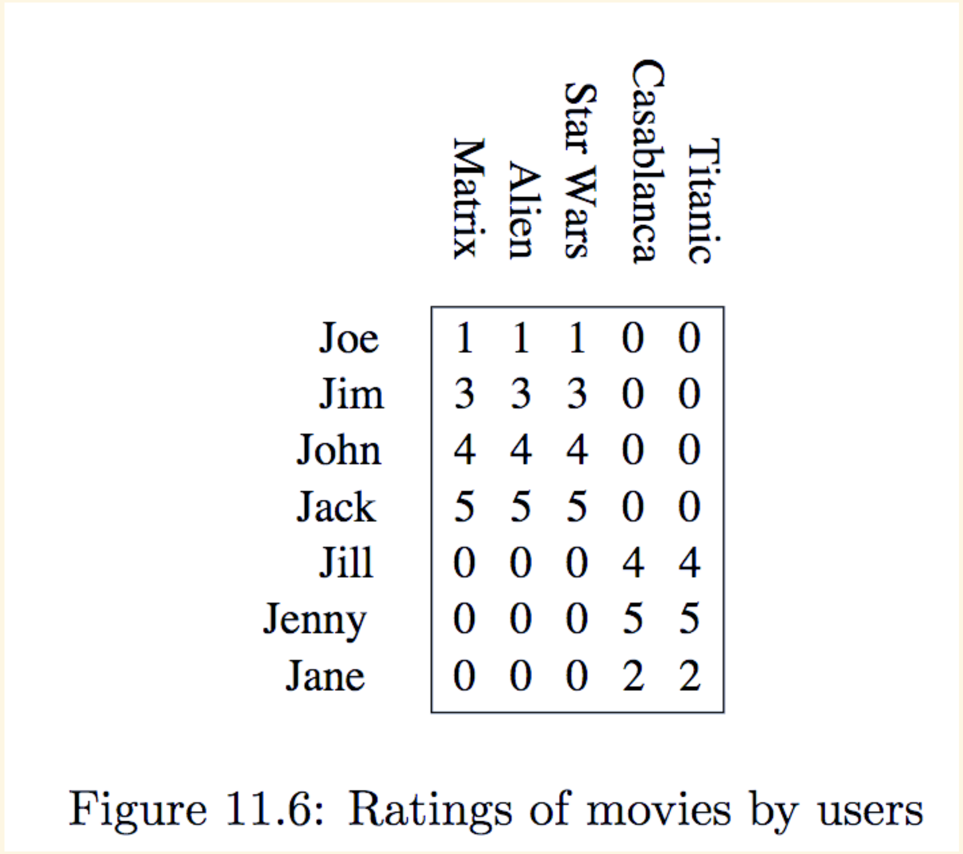
CODA: SIMPLE NMF EXAMPLE

DECOMPOSING THE NETFLIX EXAMPLE

Another look at the running example with film ratings

W: users' committment to a topic.

H: films' pertinence to a specific topic (binary, why?)



	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

Figure 11.6: Ratings of movies by users

Let R be the film ratings matrix and compute $R = PQ$

$M = 7, N = 5$ and we fix $K = 2$ to run NMF:

```
@INPUT:
    R: input matrix to be factorized, dim. M x N
    P: an initial m. of dim. M x K
    Q: an initial m. of dim. K x N
    K: the no. of latent features
    steps: the max no. of steps to perform the optimisation
    alpha: the learning rate
    beta: the regularization parameter

@OUTPUT:
    the final matrices P and Q
```

DIRECT IMPLEMENTATION (1 RUN)

```
nP=
[[ 0.33104196  0.39332058]
 [ 1.08079793  1.08397306]
 [ 1.59267325  1.27929568]
 [ 1.87852789  1.72209575]
 [ 0.67146598  1.76523621]
 [ 1.04872774  2.10824903]
 [ 0.94419145  0.59698619]]
```

```
nQ.T=
[[ 1.27381876  1.3870236   1.67315614  0.9855609   0.81578369]
 [ 1.50953822  1.38352352  1.06501557  1.87281749  1.96189735]]
```

ANALYSIS OF THE ERROR, I

```
np.dot(nP, nQ.T) = [
    [ 1.01541991  1.00333129  0.97277743  1.06287968  1.04171324]
        [ 3.01303945  2.99879446  2.96279189  3.09527589  3.0083412 ]
        [ 3.95992279  3.97901104  4.02726085  3.9655638   3.80912366]
        [ 4.99247343  4.98812249  4.97712927  5.07657468  4.91104751]
        [ 3.52001748  3.37358497  3.00347147  3.96773585  4.01098322]
        [ 4.51837154  4.37142223  4.00000329  4.98195069  4.99170315]
        [ 2.10390225  2.13556026  2.21557931  2.04860435  1.94148161]
  ]
```

```
ratings =  [[1, 1, 1, 0, 0],
            [3, 3, 3, 0, 0],
            [4, 4, 4, 0, 0],
            [5, 5, 5, 0, 0],
            [0, 0, 0, 4, 4],
            [0, 0, 0, 5, 5],
            [0, 0, 0, 2, 2]
           ]
```

ANALYSIS OF THE ERROR, II

```
np.rint(np.dot(nP, nQ.T))= [
      [ 1.  1.  1.  1.  1.]
            [ 3.  3.  3.  3.  3.]
            [ 4.  4.  4.  4.  4.]
            [ 5.  5.  5.  5.  5.]
            [ 4.  4.  4.  4.  4.]
            [ 5.  5.  5.  5.  5.]
            [ 2.  2.  2.  2.  2.]
      ]
```

```
ratings =  [[1, 1, 1, 0, 0],
            [3, 3, 3, 0, 0],
            [4, 4, 4, 0, 0],
            [5, 5, 5, 0, 0],
            [0, 0, 0, 4, 4],
            [0, 0, 0, 5, 5],
            [0, 0, 0, 2, 2]]
```

ANALYSIS OF THE RESULT

```
W(user x topic) = [
    [ 0.          0.82037571]
    [ 0.          2.46112713]
    [ 0.          3.28150284]
    [ 0.          4.10187855]
    [ 1.62445593  0.        ]
    [ 2.03056992  0.        ]
    [ 0.81222797  0.        ]
    ]
```

```
H(topic x film) =
[[ 0.          0.          0.          2.46236289  2.46236289]
 [ 1.21895369  1.21895369  1.21895369  0.          0.        ]]
```

W: users' committment to a topic.

H: films' pertinence to a specific topic (binary, why?)