Precept 8: Dimension Reduction: PCA, SVD and NMF

COS 424/524, SML 302

Sulin Liu, Xiaoyan Li

Outline

Why dimension reduction?

Principle component analysis (PCA)

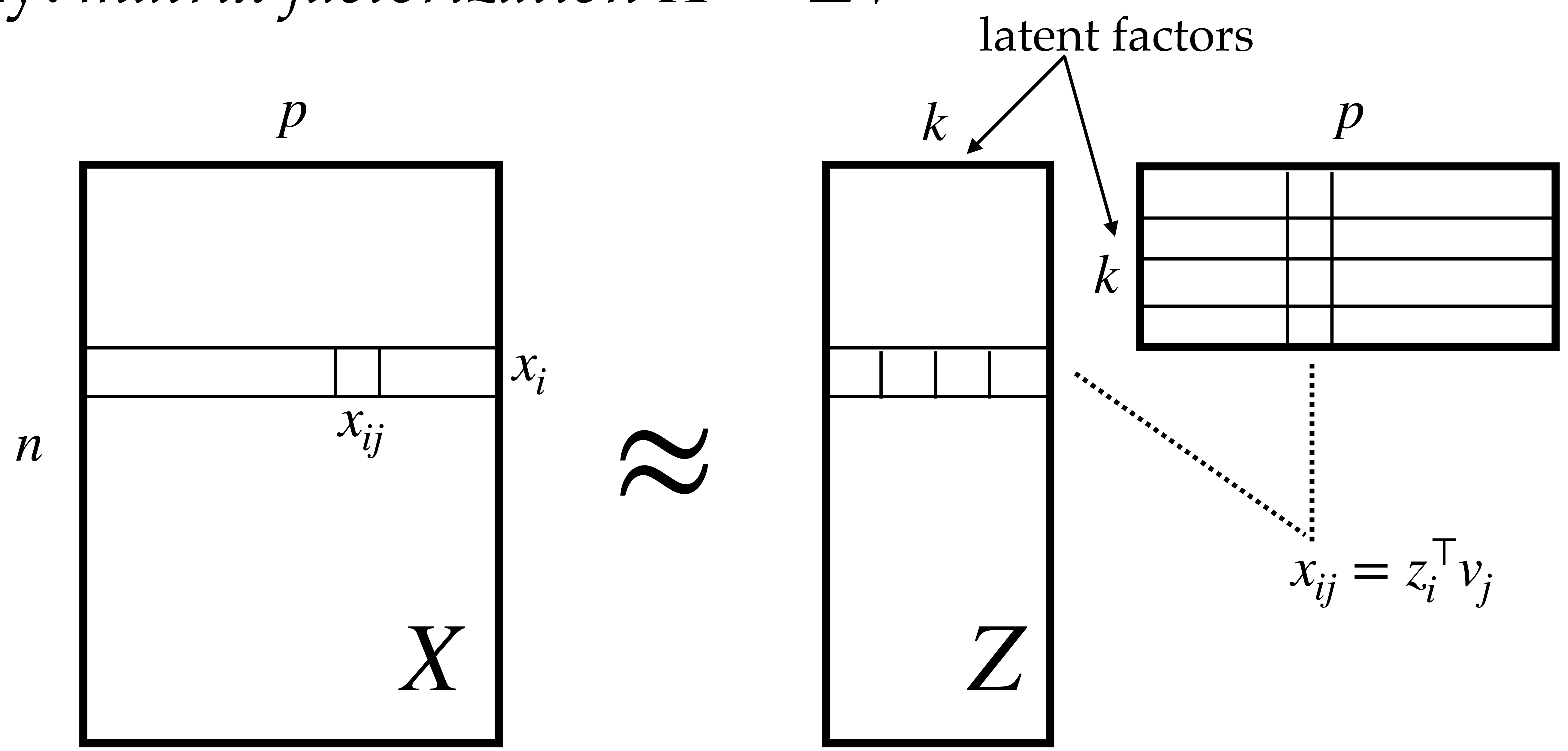
Singular value decomposition (SVD)

Non-negative matrix factorization (NMF)

Dealing with huge data and missing data

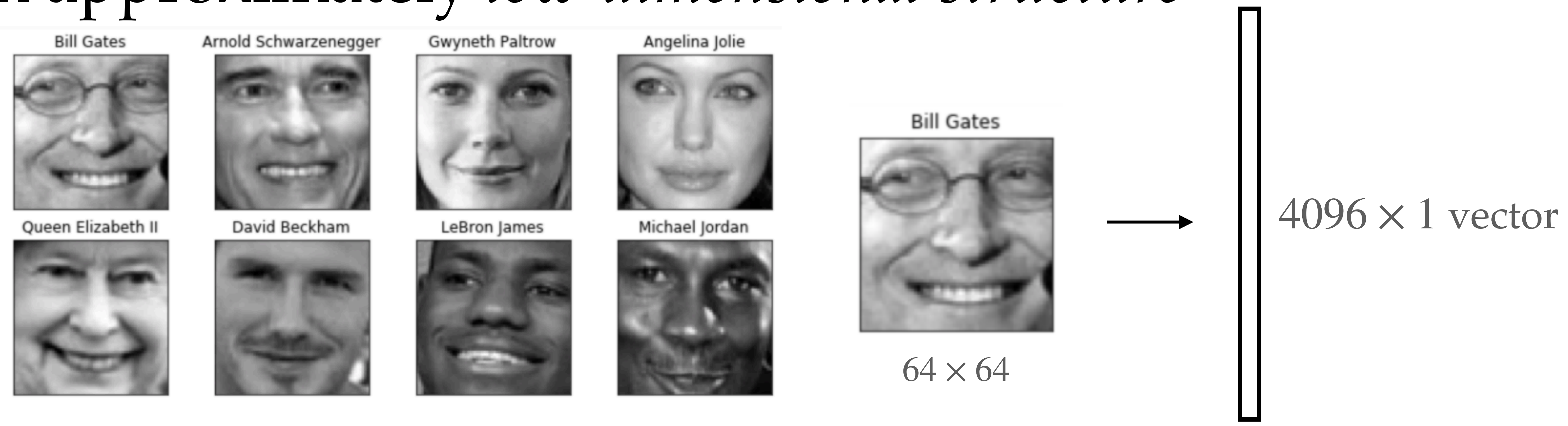
Dimension reduction

- Represent high-dimensional data with low-dimensional representations
  - $x_i \in \mathbb{R}^p \to z_i \in \mathbb{R}^k, k \ll p$
- *Focus* of today: *matrix factorization* $X = ZV^\top$



Dimension reduction

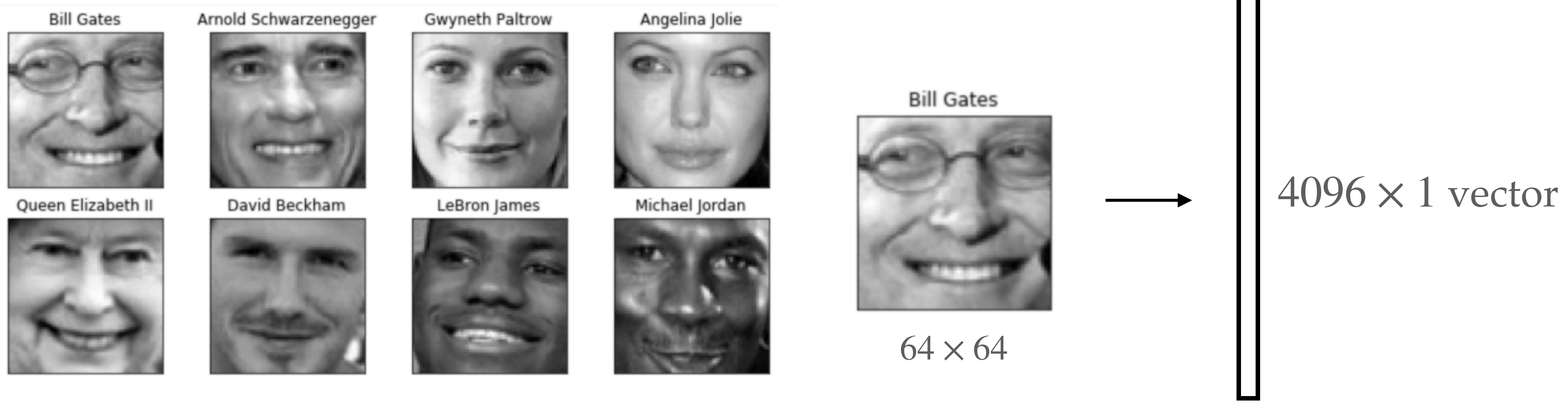
- Motivation?
  - Oftentimes the data have an approximately *low-dimensional structure*
    - Example: face images



- Potential benefits?

Dimension reduction

- Motivation?
  - Oftentimes the data have an approximately *low-dimensional structure*
    - Example: face images



- Potential benefits?

1. Data compression: the important information are kept with much less memory
2. De-noising: the less important information (hopefully noise) are discarded
3. Visualization: lower dimension means easier to visualize
4. *Useful latent structure in the data*

Outline

Why dimension reduction?

Principle component analysis (PCA)

Singular value decomposition (SVD)

Non-negative matrix factorization (NMF)

Dealing with huge data and missing data

Principle component analysis (PCA)

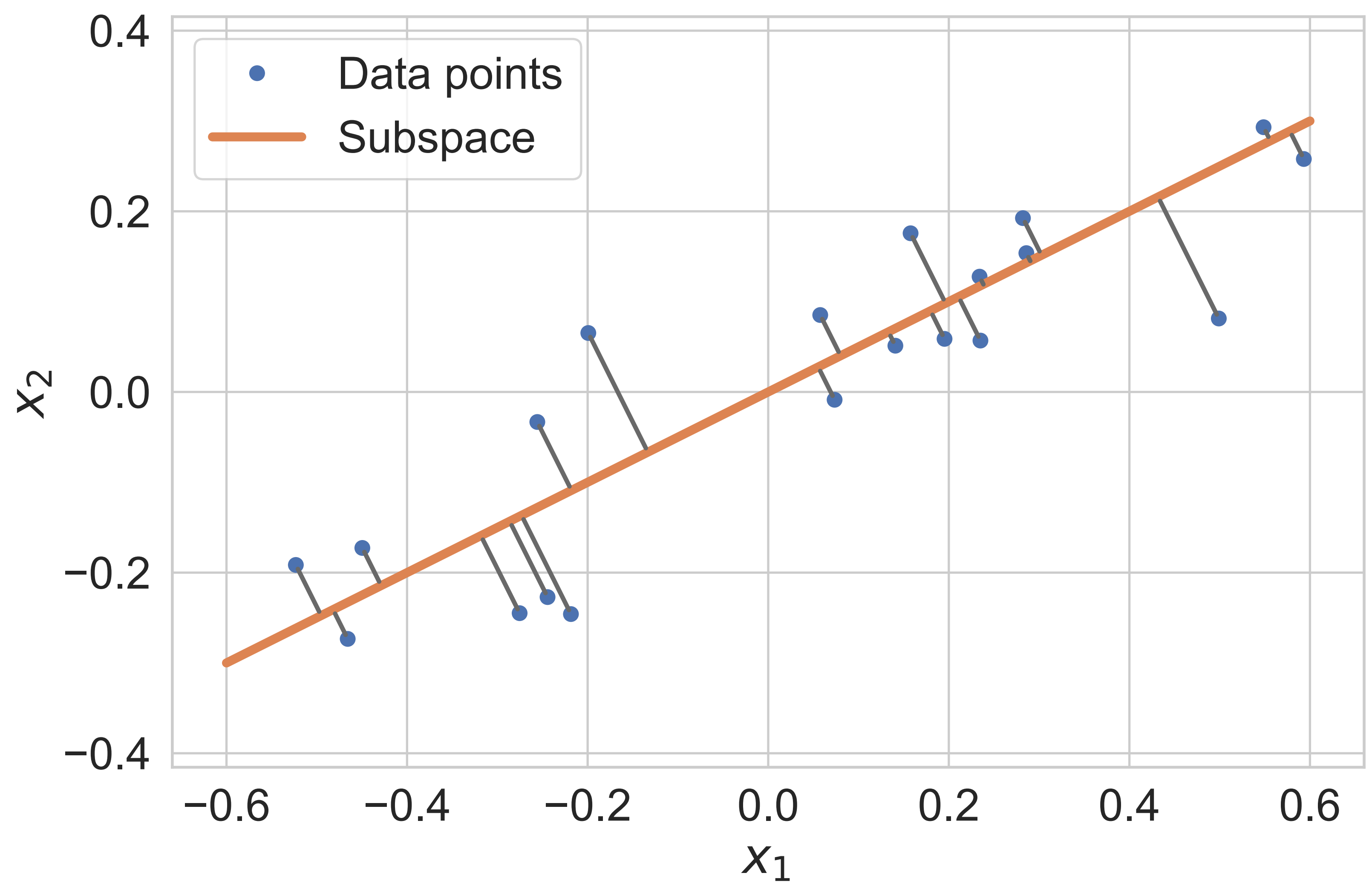
- Idea:

  1. Project data onto a set of orthogonal basis vectors (principle components): $z_i = U^\top x_i$

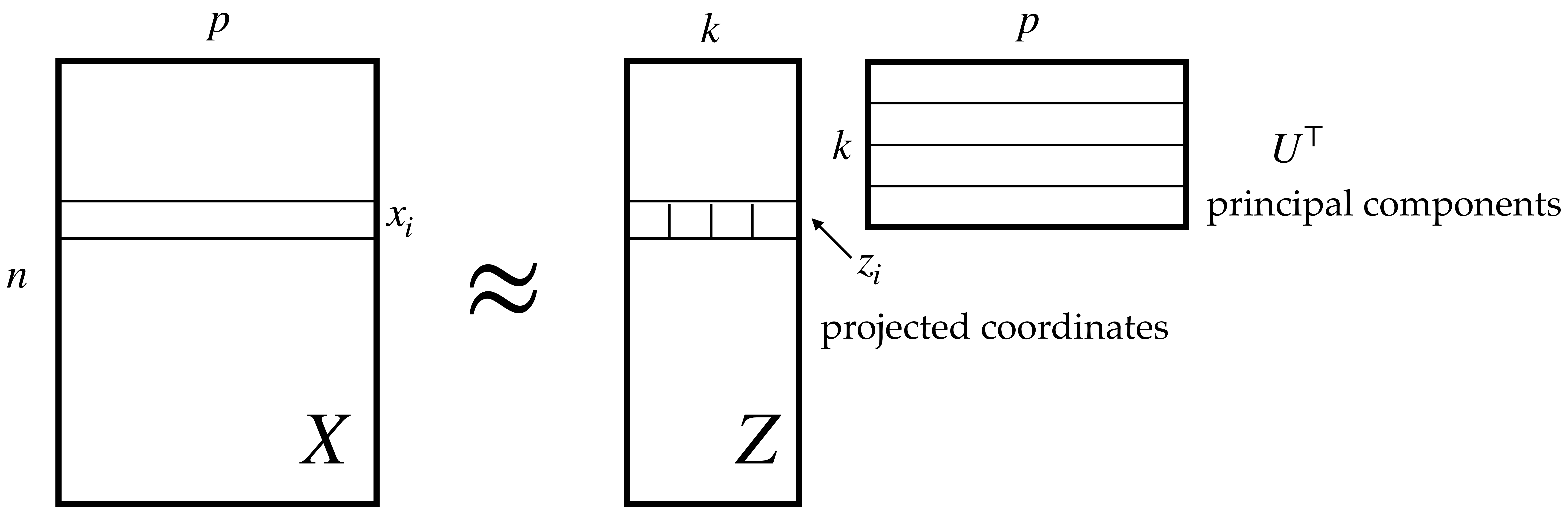
     $$u_i^\top u_j = 0, \forall i \neq j, \|u_i\| = 1, \forall i$$

  2. The principle components are chosen to be directions that capture the most variance of the data

  3. To reconstruct the data from the projected coordinates: $x_i \approx \hat{x}_i = U z_i$ (the PCs also minimize the reconstruction error)



Principle component analysis (PCA)



Principle component analysis (PCA)

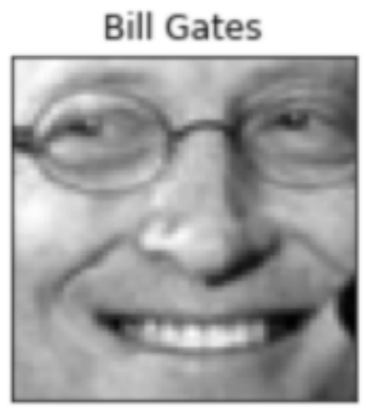
- How to compute the principal components?
  - Turns out PCs are the eigenvectors of the covariance matrix! ($XX^\top$ for mean-centered data)
  - The eigenvalues correspond to the variances explained by those directions
- Why do eigenvectors capture the most variance in the data?
  - Let's try to find the first principal component u, data $x_i \in \mathbb{R}^d$ is projected to $u^\top x_i \in \mathbb{R}$

$$\max_u \quad \frac{1}{n}\sum_{i=1}^{n}\left(u^T x_i\right)^2$$
$$= u^T X X^\top u$$

The first eigenvector maximizes this!

Principle component analysis (PCA)

- Eigenfaces:

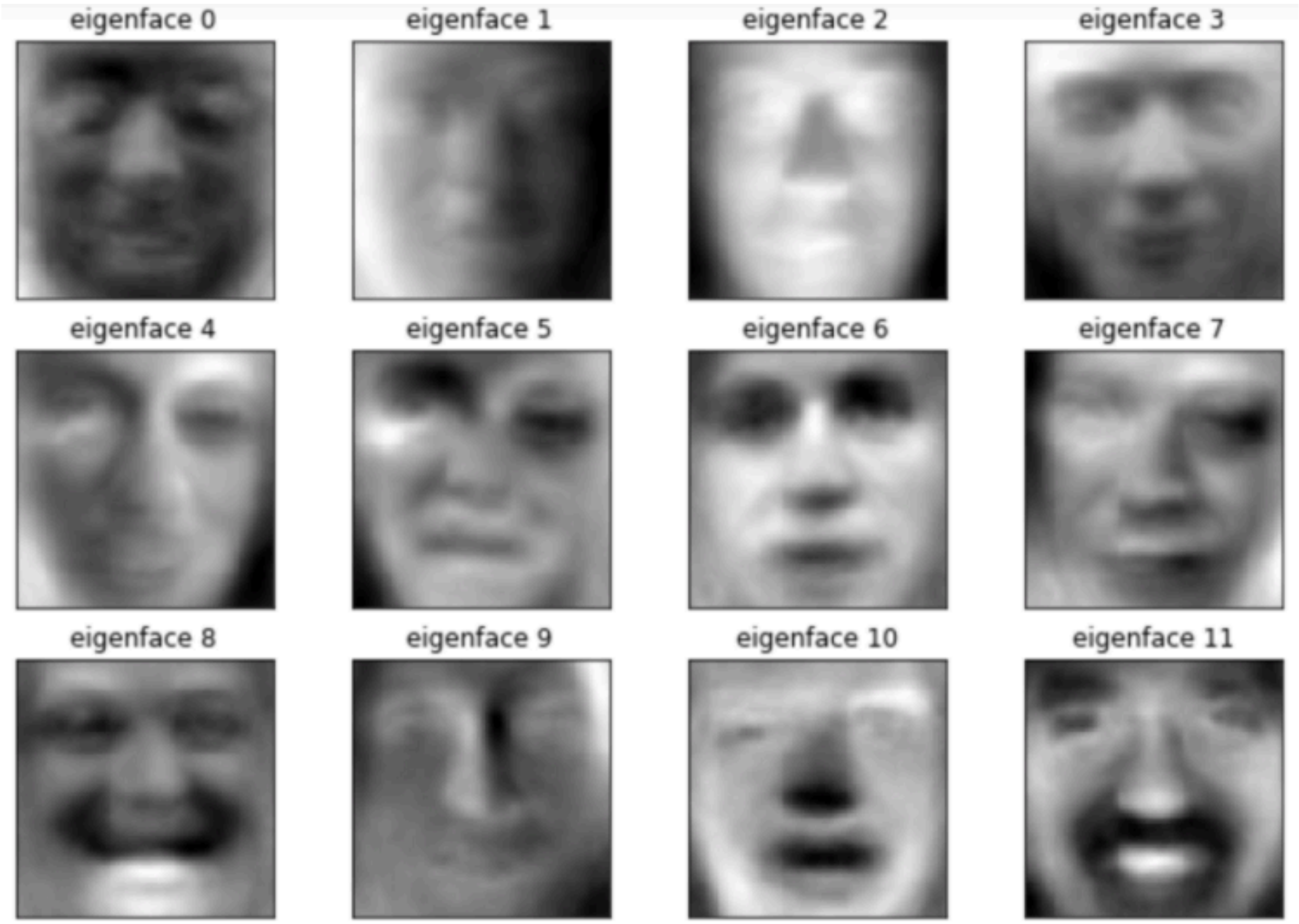


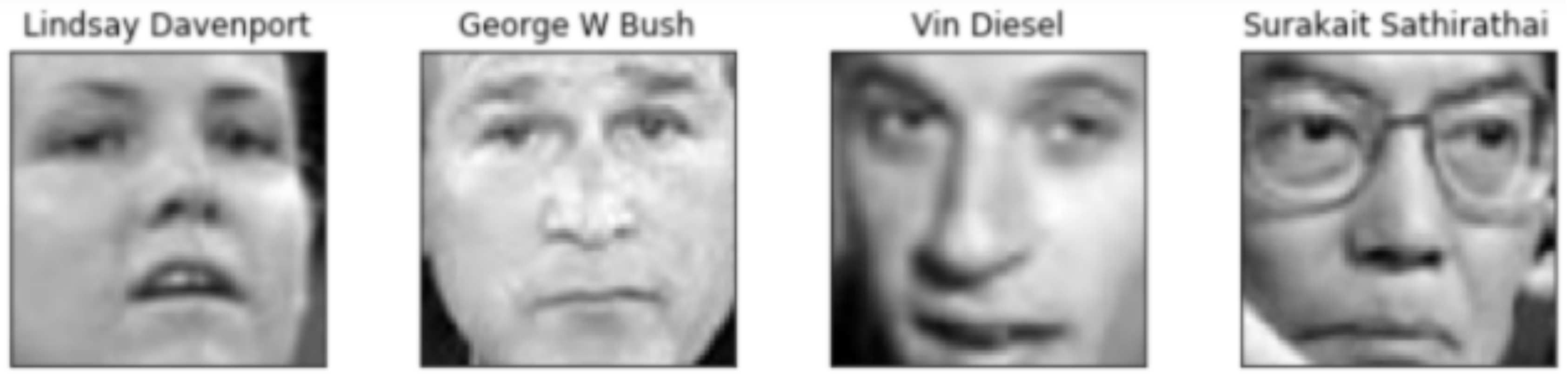
64×64

$\longrightarrow$

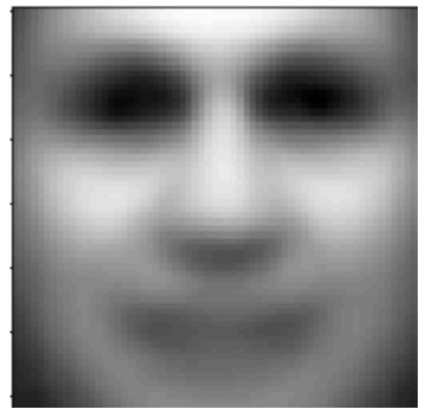
4096×1 vector

Vectorize the faces, subtract the mean and compute eigenvectors of the covariance matrix





...

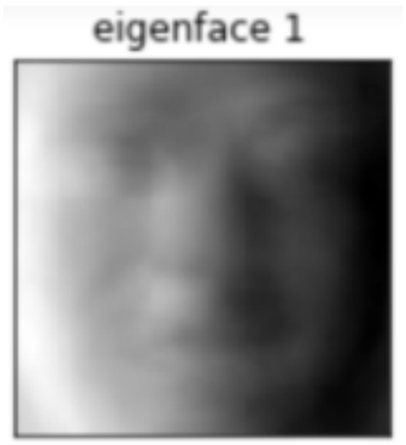


mean face

$=$

$+\ 0.7 \times$

$-\ 0.2 \times$



$+\ 0.2 \times$

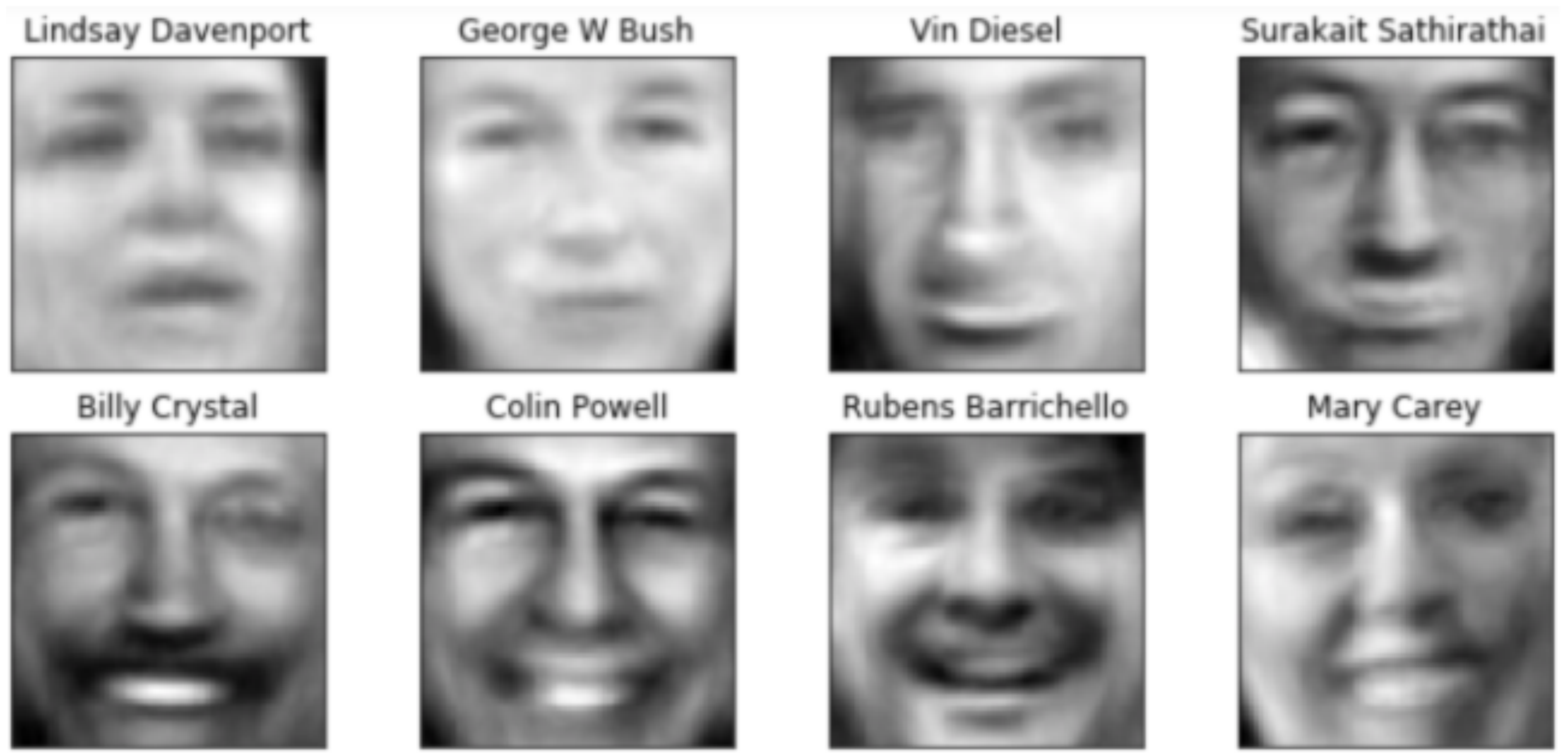
$+\ \cdots$

https://towardsdatascience.com/eigenfaces-recovering-humans-from-ghosts-17606c328184

Principle component analysis (PCA)

- Eigenfaces:

Original faces



Reconstructed faces, k=50

Reconstructed faces, k=250

Reconstructed faces, k=1000

Principle component analysis (PCA)

- How many components shall I pick?

http://www.improvedoutcomes.com/docs/WebSiteDocs/PCA/Creating_a_Scree_Plot.htm

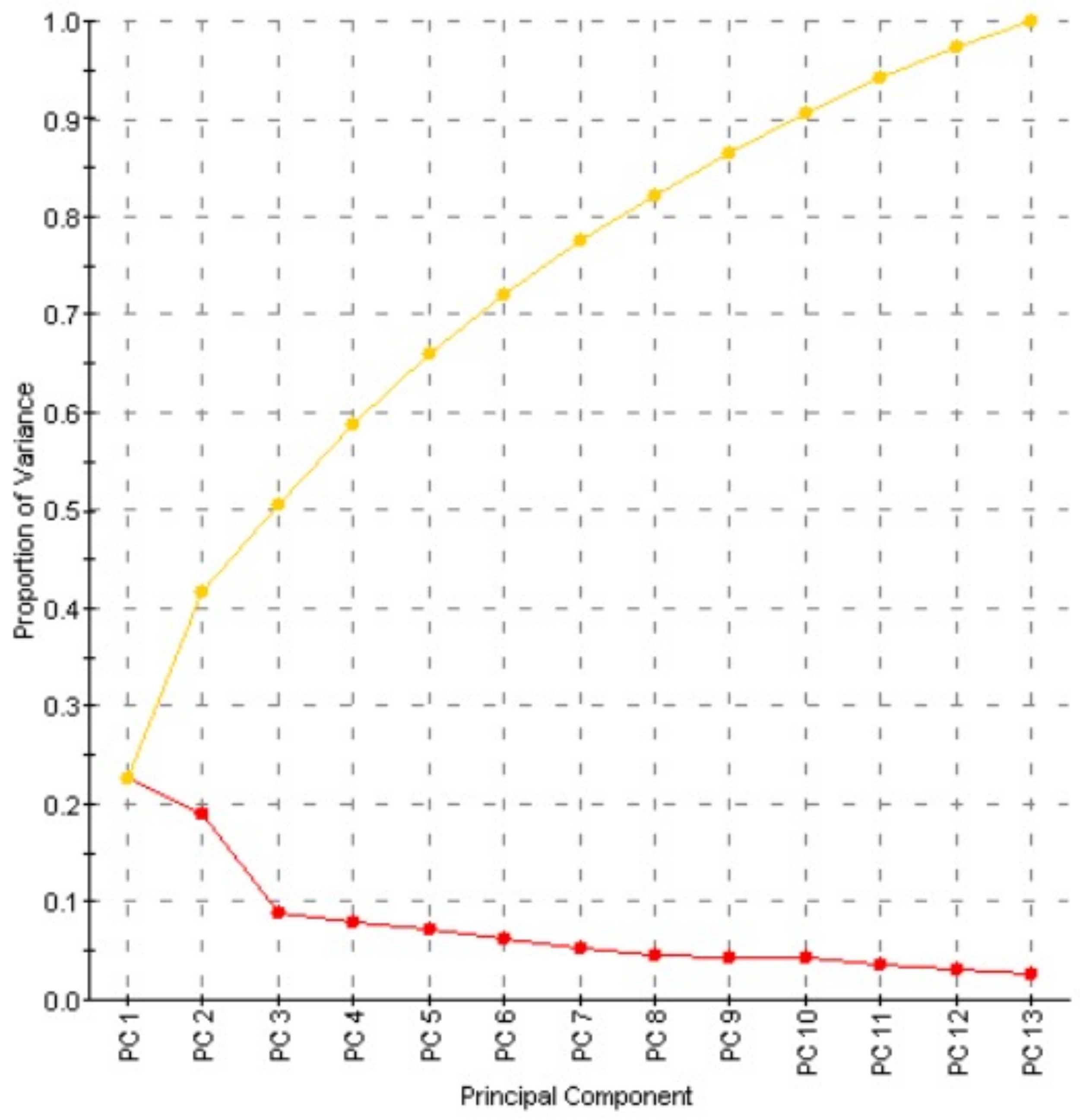
Principle component analysis (PCA)

- How many components shall I pick?

1. Percentage of the variance explained (say pick the first k until 90% variance is explained)

2. Create a scree plot, identify the PC at the "elbow".

3. Treat as hyperparameter, use cross-validation for tuning



http://www.improvedoutcomes.com/docs/WebSiteDocs/PCA/Creating_a_Scree_Plot.htm

Outline

Why dimension reduction?

Principle component analysis (PCA)

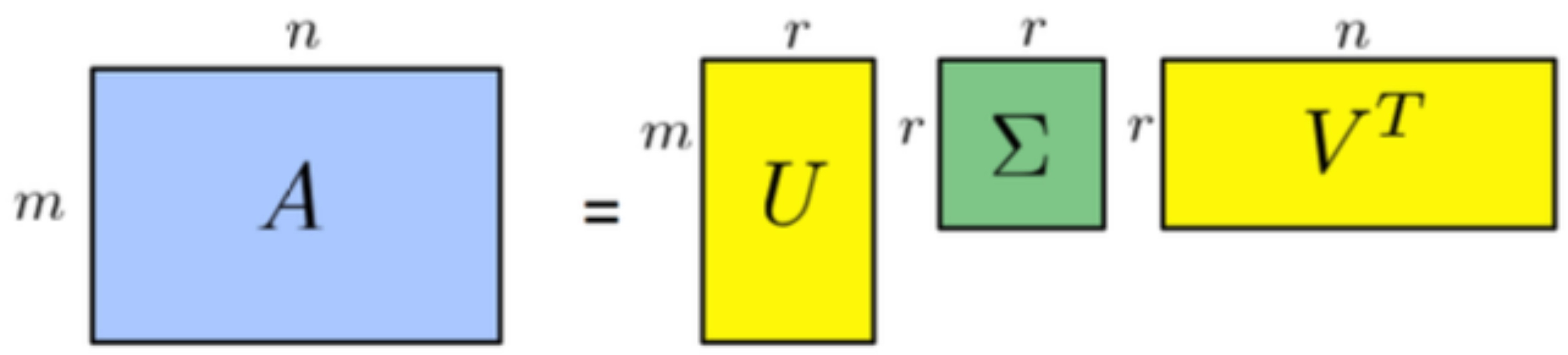
Singular value decomposition (SVD)

Non-negative matrix factorization (NMF)

Dealing with huge data and missing data

Singular value decomposition (SVD)

- SVD is a fundamental linear algebra tool: can be used for calculating PCs, low-rank matrix approximation, matrix factorization etc.
- Factorization of a rectangular matrix into three parts: $A = U\Sigma V^\top$



  - U, V are orthogonal matrices (orthogonal columns)
  - Σ is a diagonal matrix that contains singular values (non-negative)

Singular value decomposition (SVD)

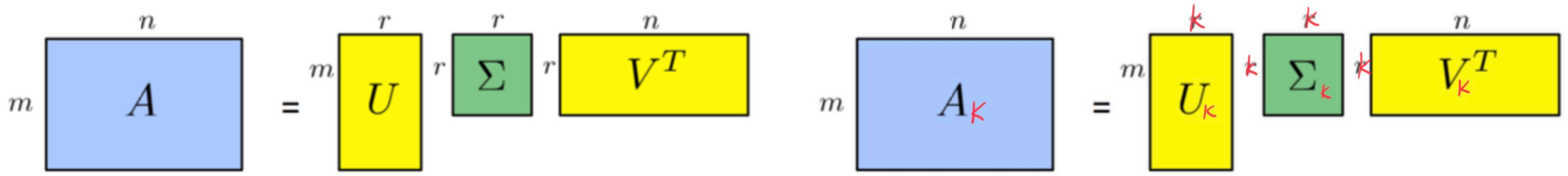
- Relationship with eigen-decomposition:
  - U are the eigenvectors of $AA^\top$
  - Short proof: $AA^\top = U\Sigma V^\top (U\Sigma V^\top)^\top = U\Sigma \underbrace{V^\top V}_{I} \Sigma U^\top = U\Sigma^2 U^\top$
- Relationship with PCA?

Singular value decomposition (SVD)

- Relationship with eigen-decomposition:
  - U are the eigenvectors of $AA^\top$
  - Short proof: $AA^\top = U\Sigma V^\top (U\Sigma V^\top)^\top = U\Sigma \underbrace{V^\top V}_{I} \Sigma U^\top = U\Sigma^2 U^\top$
- Relationship with PCA?
  - U gives the principle components if A is centered, σ_i^2's are the eigenvalues

Truncated SVD

- Approximate the data with the top k components: $A_k = U_k \Sigma_k V_k^\top = \sum_{i=1}^{k} \sigma_i u_i v_i^\top \approx \sum_{i=1}^{r} \sigma_i u_i v_i^\top = U \Sigma V^\top = A$



- This gives a low rank (rank=k) approximation to A
  - in fact the best approximation to A in terms of matrix Frobenius norm
- How do we select k? Similar to PCA.
- It is also used as a initialization for many of the more complicated matrix factorization problems

Outline

Why dimension reduction?

Principle component analysis (PCA)

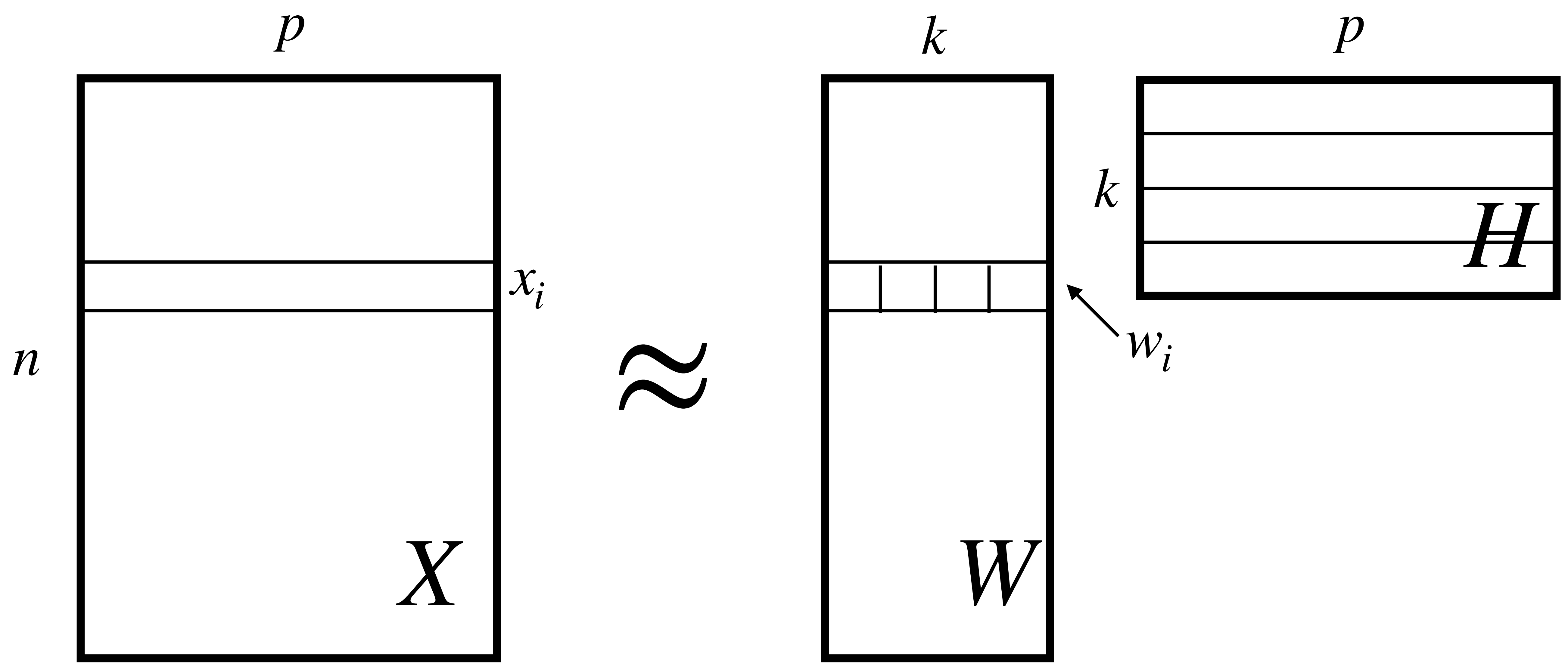
Singular value decomposition (SVD)

Non-negative matrix factorization (NMF)

Dealing with huge data and missing data

Non-negative matrix factorization (NMF)

- Factorizes a non-negative matrix X into two non-negative matrices



- Example use case: text data topic modeling
    - x_i is the bag-of-words representation, w_i contains the weights for k different topics, each row of H is bag-of-words representation for each topic

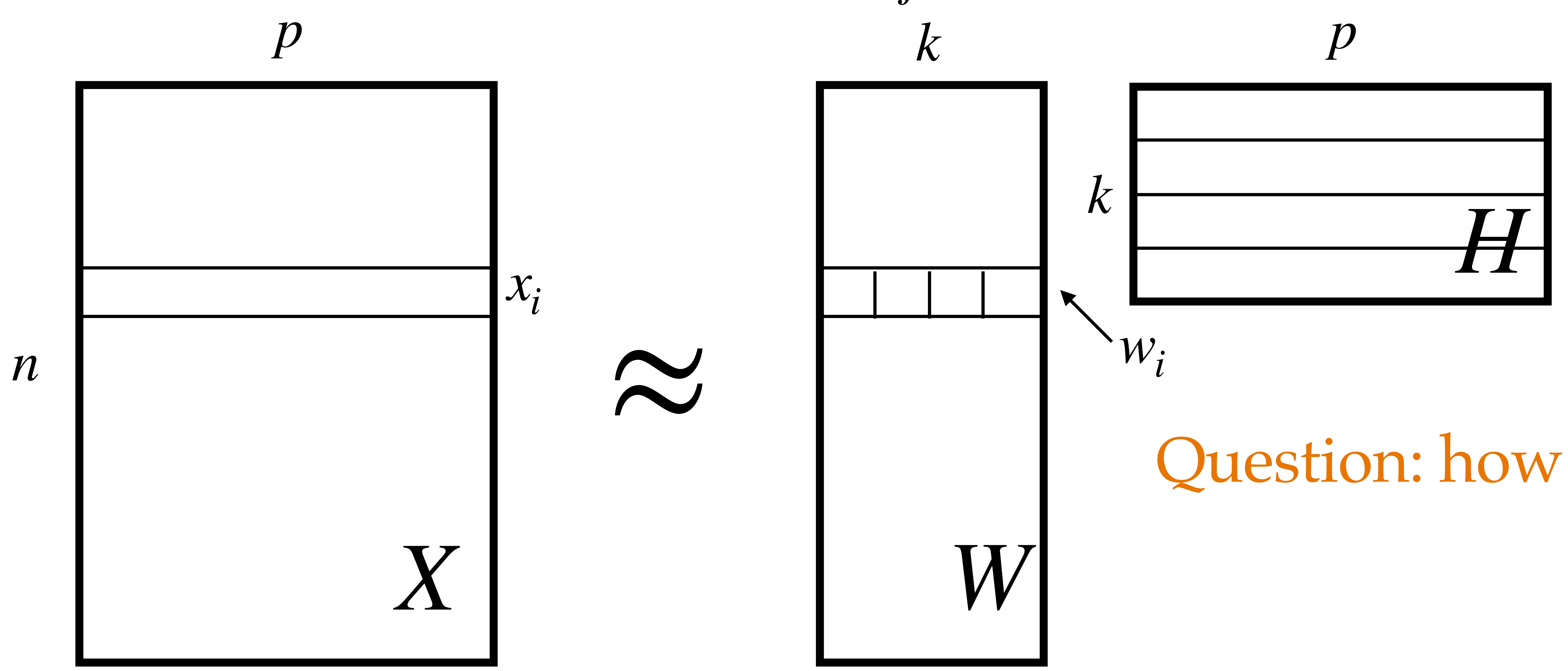
Non-negative matrix factorization (NMF)

- How do we pick k?
  - Use SVD to determine how low rank the matrix is
  - Domain knowledge (number of topics, number of sources)
  - Cross-validation
- Are W and H unique?
  - Not unique if we are just minimizing $\|X - WH\|_F^2 : WBB^{-1}H = WH, \forall B$
- More terms can be added to the optimization objective (L1 penalty, L2 penalty…)

$$\frac{1}{2}\|X - WH\|_F^2 + \alpha\|\text{vec}(W)\|_1 + \alpha\|\text{vec}(H)\|_1 + \beta\|W\|_F^2 + \beta\|H\|_F^2 \quad \text{s.t.} W \geq 0, H \geq 0$$

NMF in HW3

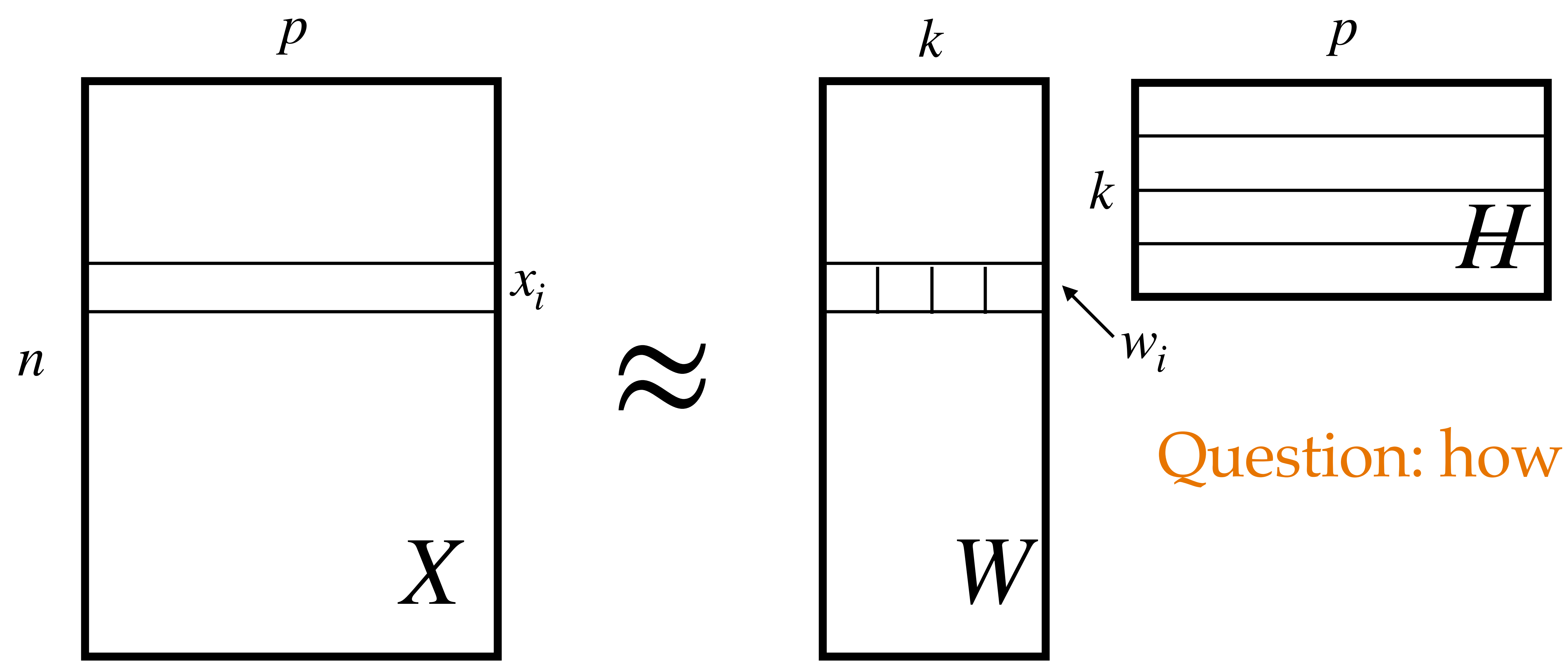
- X is the police-compliant data matrix
  - n police officers $\times$ p type of complaints: x_{ij} is the count of complaints of type j for police officer i



Question: how to interpret W and H?

NMF in HW3

- X is the police-compliant data matrix
  - n police officers $\times$ p type of complaints: x_{ij} is the count of complaints of type j for police officer i



Question: how to interpret W and H?

- k type of officers, each row of H corresponds to complaint pattern for each type of officers
- w_i: weights for the different types

NMF in HW3

- What can I do with the latent representations W and H?

NMF in HW3

- What can I do with the latent representations W and H? Just some ideas:
  - Use W as low-dimensional features of the police officers and apply learning methods on top of it.
    - Clustering? Prediction of certain features in the police officer database?

NMF in HW3

- What can I do with the latent representations W and H? Just some ideas:
  - Use W as low-dimensional features of the police officers and apply learning methods on top of it.
    - Clustering? Prediction of certain features in the police officer database?
  - H could give some interpretations about the patterns in police officers.
    - Does it correspond to certain type of police officers? Connections with the police officer database?

Outline

Why dimension reduction?

Principle component analysis (PCA)

Singular value decomposition (SVD)

Non-negative matrix factorization (NMF)

Dealing with huge data and missing data

Dealing with huge data

- Use sparse matrix format (supported by scikit learn)
  - Reduces computation time and uses less memory
- Use approximate but faster methods:
  - Stochastic gradient descent (SGD) as an approximate method to gradient descent
  - MiniBatchKMeans, a faster approximate variant of KMeans
- Dimension reduction
  - Represent data in a low-dimensional subspace
  - Perform learning on the low-dimensional representations

Matrix factorization with missing data

- Suppose the objective we are minimizing is:

$$\arg\min_{W,H} \frac{1}{2}\|X - WH\|_F^2$$

- Define a binary mask matrix M over the labeled ratings:

$$\hat{W}, \hat{H} = \arg\min_{W,H} \frac{1}{2}\|M \odot (X - WH)\|_F^2$$

- Fill up missing values with $\hat{X} = \hat{W}\hat{H}$

Netflix data matrix

The figure is due to Mahdi Soltanolkotabi

Related python package

- The sklearn.decomposition module for matrix factorization
  - PCA, TruncatedSVD, NMF and many more
  - Hyperparamters: n_components, regularization terms for NMF
- The scipy.sparse module for sparse matrices

Start thinking about HW3!

1. Come up with an ML task of your interest about the dataset
   - Look at the data, what data are there?
   - What patterns are you interested in finding out?
   - Ask questions that help inform better policies
2. Formulate the task mathematically

Start thinking about HW3!

3. What ML methods do you plan to use and motivate the methods

- Dimension reduction (PCA, NMF, LDA)
- Clustering
- Community detection (stochastic block model)
- Graph analysis
- Time series analysis (Poisson/Hawkes processes) …

Start thinking about HW3!

4. Explain your results
    - Visualization,
    - Interpretation
    - Prediction…
5. In the end, it is all up to you to explore! We look forward to your interesting findings (;

Some more resources

- Ryan Adams's COS 302 Lectures on matrix factorization and SVD
  - https://www.youtube.com/watch?v=67a8CIukcPA&ab_channel=IntelligentSystemsLab
  - https://www.youtube.com/watch?v=JUYGohQY41U&ab_channel=IntelligentSystemsLab
- Eigenfaces
  - https://towardsdatascience.com/eigenfaces-recovering-humans-from-ghosts-17606c328184