COMPSCI 514: Algorithms for Data Science

Cameron Musco

University of Massachusetts Amherst. Spring 2026.

Lecture 15

Logistics

- Problem Set 3 is due this Friday at 11:59pm.
- office hours after class

  Le SVD, other applications of SVD.

Summary

Last Class

- Review of optimal low-rank approximation via eigendecomposition.
- Linear algebra exervises and practice proofs.
- The Singular Value Decomposition (SVD) and its connection to eigendecomposition and low-rank approximation

This Class:

- Applications of SVD beyond low-rank approximation.
- Applications of low-rank approximation beyond compression
- Low-rank matrix completion (predicting missing measurements using low-rank structure).
- Entity embeddings (e.g., word embeddings, node embeddings).

Low-Rank Approximation Review

True or False? $X \in \mathbb{R}^{n \times d}$

$$\min_{V \in \mathbb{R}^{d \times k} : V^T V = I} \|X - XVV^T\|_F^2 = \min_{B : \text{rank}(B) \leq k} \|X - B\|_F^2.$$

V: $d \times k$

TRUE

Proof by contradiction:

$\min \|X - XVV^T\|_F < \min \|X - B\|_F$ ✗

↑ this has rank $\leq k$

$\min \|X - XVV^T\|_F > \min \|X - B\|_F$ ✗

$\bar{V}$ be a basis for B's rows $\bar{V} \in \mathbb{R}^{d \times k}$

$\min \|X - XVV^T\|_F < \|X - X\bar{V}\bar{V}^T\|_F \leq \|X - B\|_F$

Low-Rank Approximation Review

True or False?

$$\min_{V \in \mathbb{R}^{d \times k} : V^T V = I} \|X - XVV^T\|_F^2 = \min_{B : \mathrm{rank}(B) \le k} \|X - B\|_F^2.$$

TRUE

How about:

$$\min_{U \in \mathbb{R}^{n \times k} : U^T U = I} \|X - UU^T X\|_F^2 = \min_{B : \mathrm{rank}(B) \le k} \|X - B\|_F^2.$$

TRUE

Low-Rank Approximation Review

What is the value of

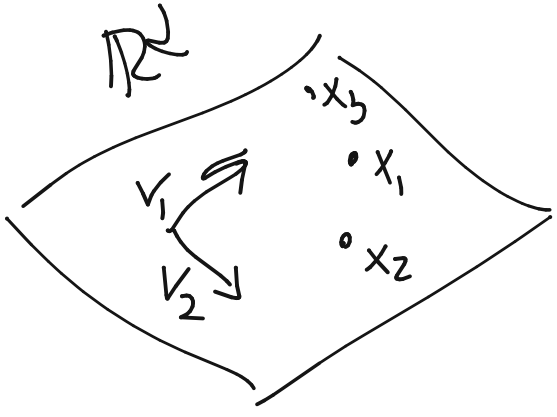
$$\min_{B:\text{rank}(B)\leq k} \|X - B\|_F^2?$$

$$\sum_{i=k+1}^{d} \lambda_i(X^TX) = \sum_{i=k+1}^{r} \sigma_i(X)^2$$

$$\sum_{i=k+1}^{n} \lambda_i(XX^T)$$

Column and Row Spans

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ have it's ~~columns~~ rows spanned by the ~~rows~~ columns of an orthonormal matrix $\mathbf{V} \in \mathbb{R}^{d \times k}$. Show that the columns of $\mathbf{X}$ are spanned by the columns of $\mathbf{XV} \in \mathbb{R}^{n \times k}$.



$$X = \begin{bmatrix} \\ \\ \\ \\ \end{bmatrix}$$

$X_i = VC_i$

$X_i^T = C_i^T V^T$

$$X = \begin{bmatrix} C_1^T V^T \\ C_2^T V^T \\ \vdots \\ \end{bmatrix}$$

$$= \begin{bmatrix} & & \\ C & & \end{bmatrix}_k \begin{bmatrix} & V^T & \end{bmatrix}_k$$

n

XVV^T:
X but with rows projected to subspace spanned by V's columns $\text{col}(V)$

$C = XV$

$CV^T = XVV^T = X$

$$\begin{bmatrix} XV \end{bmatrix} \begin{bmatrix} v_1 & v_2 & \cdots & v_k \end{bmatrix}^T$$

XVv_i

Matrix Completion

Assume that **A** is a rank-1 matrix, and that you have access to a subset of its entries, shown below:

$$\mathbf{A} = \begin{bmatrix} 6 & 9 & 4 \\ x & 2 & y \\ 3 & z & 2 \end{bmatrix}$$

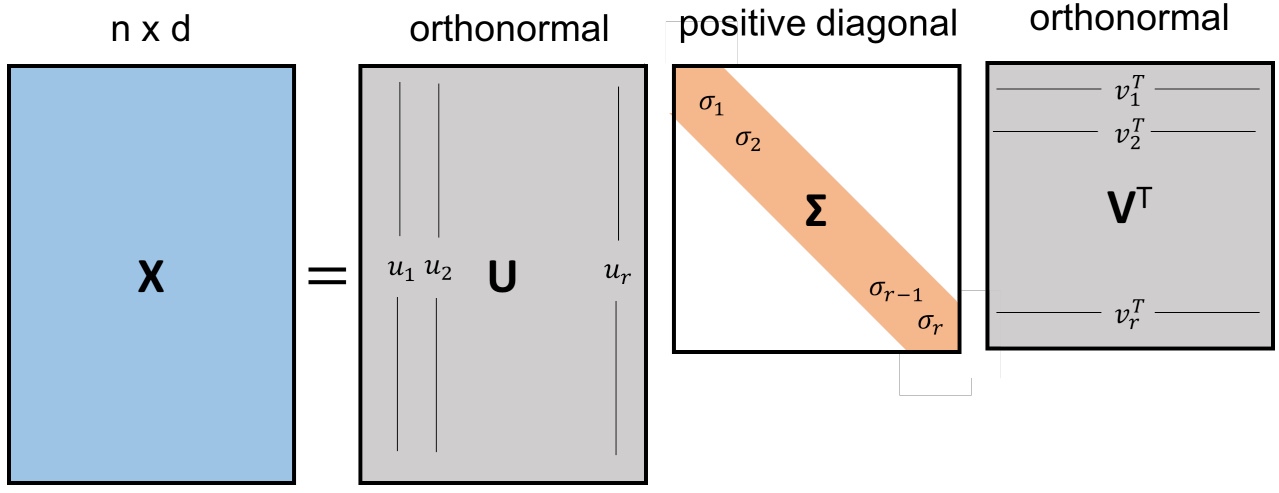
What are the values of x, y, and z?

Singular Value Decomposition

The Singular Value Decomposition (SVD) generalizes the eigendecomposition to asymmetric (even rectangular) matrices. Any matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with $\mathbf{rank(X)} = r$ can be written as $\mathbf{X} = \mathbf{U\Sigma V}^T$.

- $\mathbf{U}$ has orthonormal columns $\vec{u}_1, \ldots, \vec{u}_r \in \mathbb{R}^n$ (left singular vectors).
- $\mathbf{V}$ has orthonormal columns $\vec{v}_1, \ldots, \vec{v}_r \in \mathbb{R}^d$ (right singular vectors).
- $\mathbf{\Sigma}$ is diagonal with elements $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ (singular values).

$A = V \Lambda V^T$

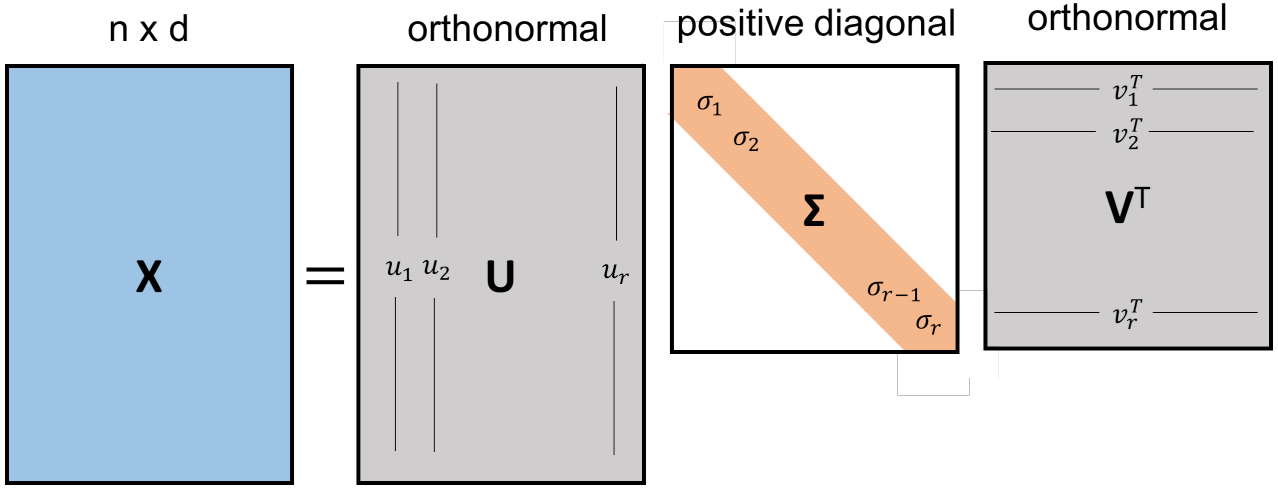


The SVD and Optimal Low-Rank Approximation

The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B}\in\mathbb{R}^{n\times d}} \|\mathbf{X}-\mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$$

Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$

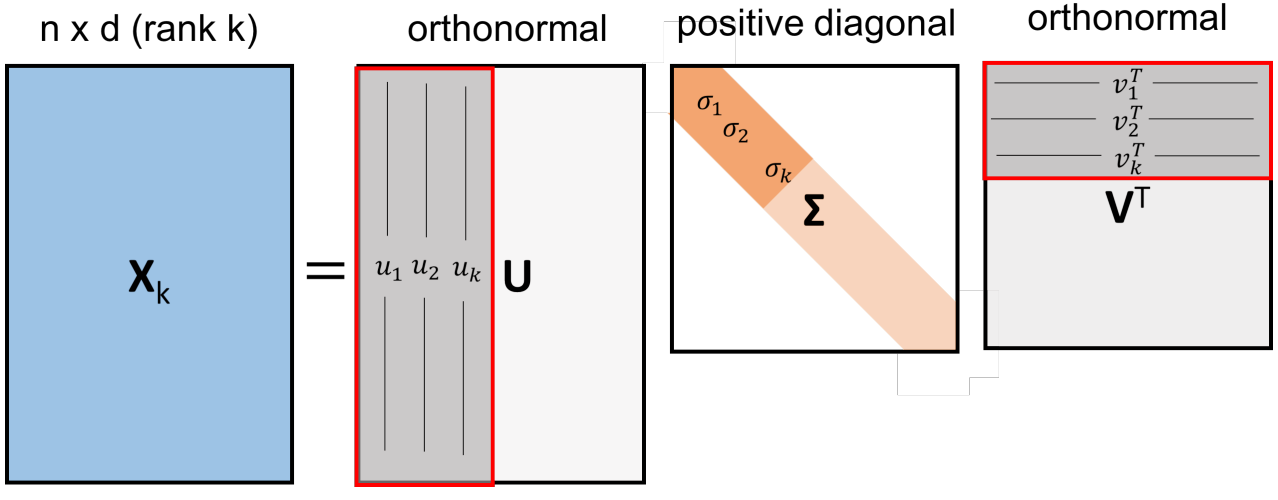


The SVD and Optimal Low-Rank Approximation

The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B}\in\mathbb{R}^{n\times d}} \|\mathbf{X}-\mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$$

Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$

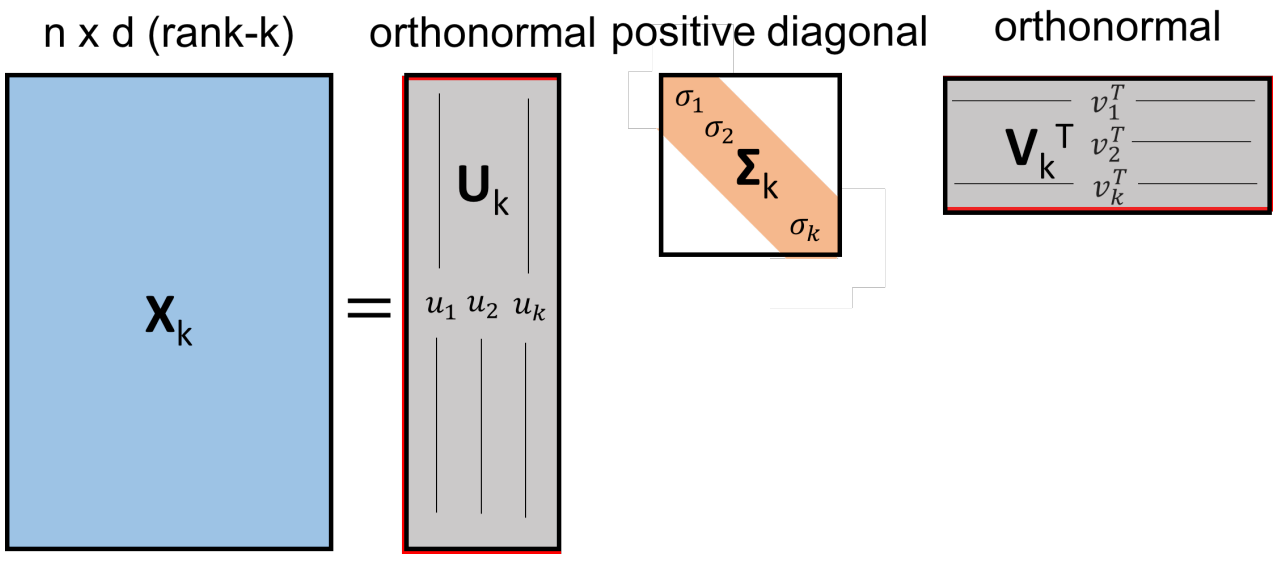


The SVD and Optimal Low-Rank Approximation

The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B}\in\mathbb{R}^{n\times d}} \|\mathbf{X}-\mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$$

Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$

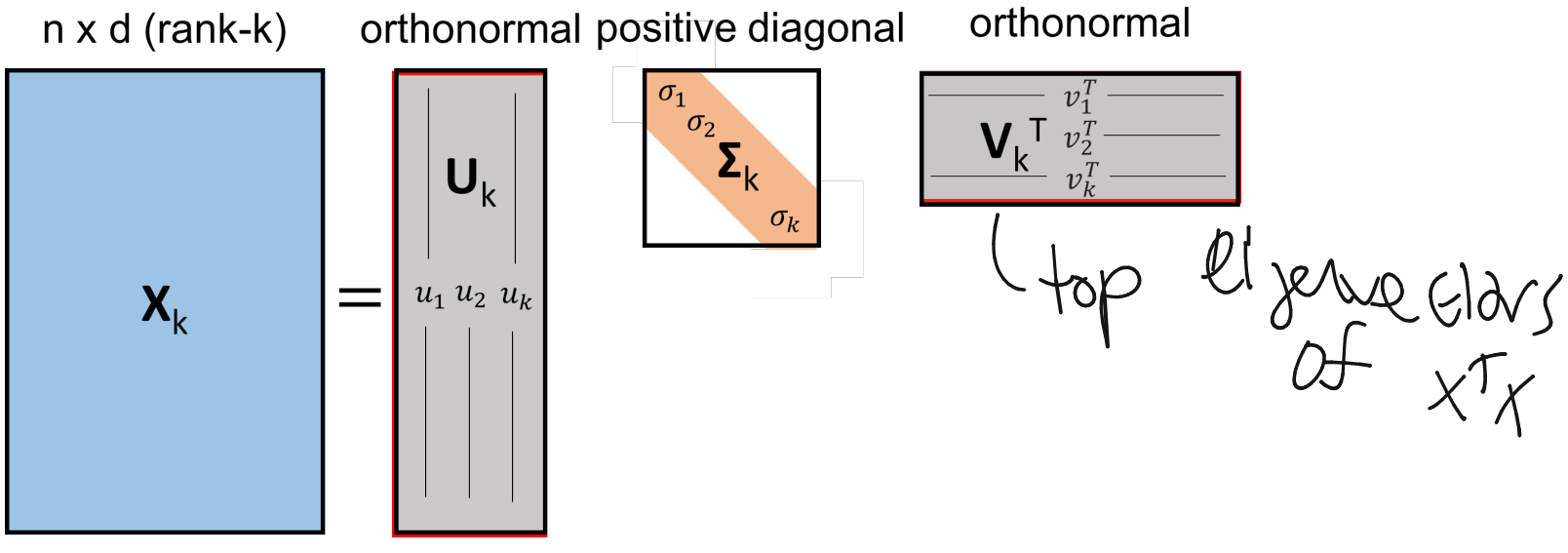


The SVD and Optimal Low-Rank Approximation

The best low-rank approximation to $\mathbf{X}$:
$\mathbf{X}_k = \arg\min_{\text{rank}-k\ \mathbf{B}\in\mathbb{R}^{n\times d}} \|\mathbf{X}-\mathbf{B}\|_F$ is given by:

$$\mathbf{X}_k = \mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X} = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T$$

Correspond to projecting the rows (data points) onto the span of $\mathbf{V}_k$ or the columns (features) onto the span of $\mathbf{U}_k$



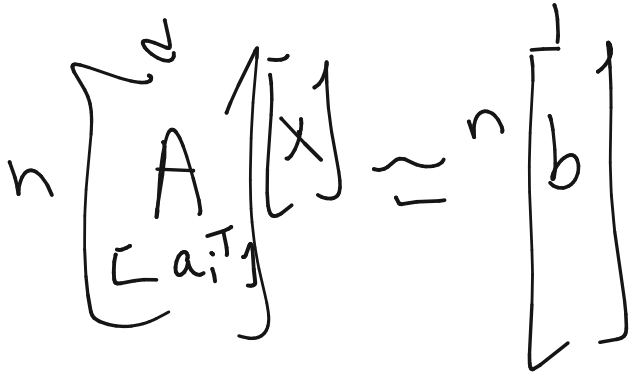
Another Application of SVD

SVD is the ‘Swiss army knife’ of linear algebra. Lots of applications beyond low-rank approximation.

Another Application of SVD

SVD is the 'Swiss army knife' of linear algebra. Lots of applications beyond low-rank approximation.

Consider the following optimization problem: given $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{b} \in \mathbb{R}^n$, compute $\arg\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2^2$.



What is this problem known as?

↳ least squares

linear regression

$$\sum_{i=1}^{n} \left([Ax]_i - b_i\right)^2 = \sum_{i=1}^{n} \left(\underbrace{a_i^T}_{\text{data point}} x - b_i\right)^2$$

↓ ↓ label

inner function

Another Application of SVD

Equivalent Formulation: Letting $\mathbf{y} = \mathbf{Ax}$, computing $\arg\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2^2$ is equivalent to finding:

$$\arg\min_{\mathbf{y} \in colspan(\mathbf{A})} \|\mathbf{y} - \mathbf{b}\|_2^2.$$

Another Application of SVD

Equivalent Formulation: Letting $\mathbf{y} = \mathbf{A}\mathbf{x}$, computing $\arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$ is equivalent to finding:

$$\arg\min_{\mathbf{y} \in colspan(\mathbf{A})} \|\mathbf{y} - \mathbf{b}\|_2^2.$$

What is the solution to this problem?

Another Application of SVD

Equivalent Formulation: Letting $\mathbf{y} = \mathbf{Ax}$, computing $\arg\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2^2$ is equivalent to finding:

$$\arg\min_{\mathbf{y} \in colspan(\mathbf{A})} \|\mathbf{y} - \mathbf{b}\|_2^2.$$

What is the solution to this problem? $\mathbf{y} = \mathbf{U}\mathbf{U}^T\mathbf{b}$ where $\mathbf{U}$ is an orthonormal basis for the columns of $\mathbf{A}$

projection onto the column span

Taking $\mathbf{U}$ to be the left singular vectors of $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ and letting $\mathbf{x}^* = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T\mathbf{b}$ we have:

$$\mathbf{y} = \mathbf{A}\mathbf{x}^* = \underbrace{\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T}_{\mathbf{A}} \underbrace{\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T\mathbf{b}}_{\mathbf{x}^*} = \mathbf{U}\mathbf{U}^T\mathbf{b}$$

Another Application of SVD

Equivalent Formulation: Letting $\mathbf{y} = \mathbf{Ax}$, computing $\arg\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2^2$ is equivalent to finding:

$$\arg\min_{\mathbf{y} \in colspan(\mathbf{A})} \|\mathbf{y} - \mathbf{b}\|_2^2.$$

What is the solution to this problem? $\mathbf{y} = \mathbf{U}\mathbf{U}^T\mathbf{b}$ where $\mathbf{U}$ is an orthonormal basis for the columns of $\mathbf{A}$

Taking $\mathbf{U}$ to be the left singular vectors of $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ and letting $\mathbf{x}^* = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T\mathbf{b}$ we have:

$$\mathbf{y} = \mathbf{Ax} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T = \mathbf{U}\mathbf{U}^T\mathbf{b}.$$

So $\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2^2$.

Another Application of SVD

Equivalent Formulation: Letting $\mathbf{y} = \mathbf{A}\mathbf{x}$, computing $\arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$ is equivalent to finding:

$$\arg\min_{\mathbf{y} \in colspan(\mathbf{A})} \|\mathbf{y} - \mathbf{b}\|_2^2.$$

What is the solution to this problem? $\mathbf{y} = \mathbf{U}\mathbf{U}^T\mathbf{b}$ where $\mathbf{U}$ is an orthonormal basis for the columns of $\mathbf{A}$

Taking $\mathbf{U}$ to be the left singular vectors of $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ and letting $\mathbf{x}^* = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T\mathbf{b}$ we have:

$$\mathbf{y} = \mathbf{A}\mathbf{x} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T = \mathbf{U}\mathbf{U}^T\mathbf{b}.$$

So $\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$.

$\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T$ is the Moore–Penrose pseudoinverse of $\mathbf{A}$. Can be read off from the SVD.

Applications of Low-Rank Approximation Beyond Compression

Matrix Completion

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix).

Matrix Completion

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.

X

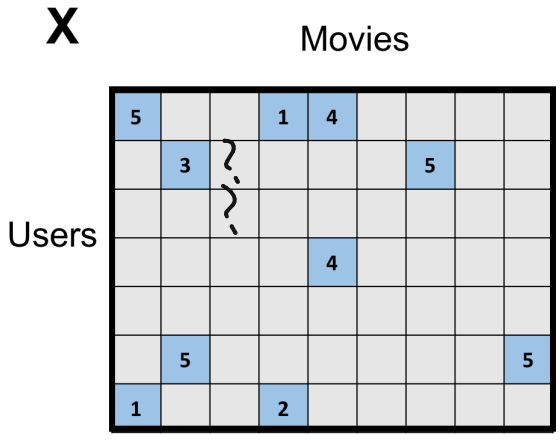
Movies

Users

5	3	3	1	4	4	4	3	5
4	3	3	1	4	4	5	3	5
3	3	3	2	3	3	3	3	3
4	3	3	4	4	4	4	3	3
3	3	3	2	3	3	3	3	3
2	5	3	4	4	4	4	4	5
1	3	3	2	3	3	3	1	2

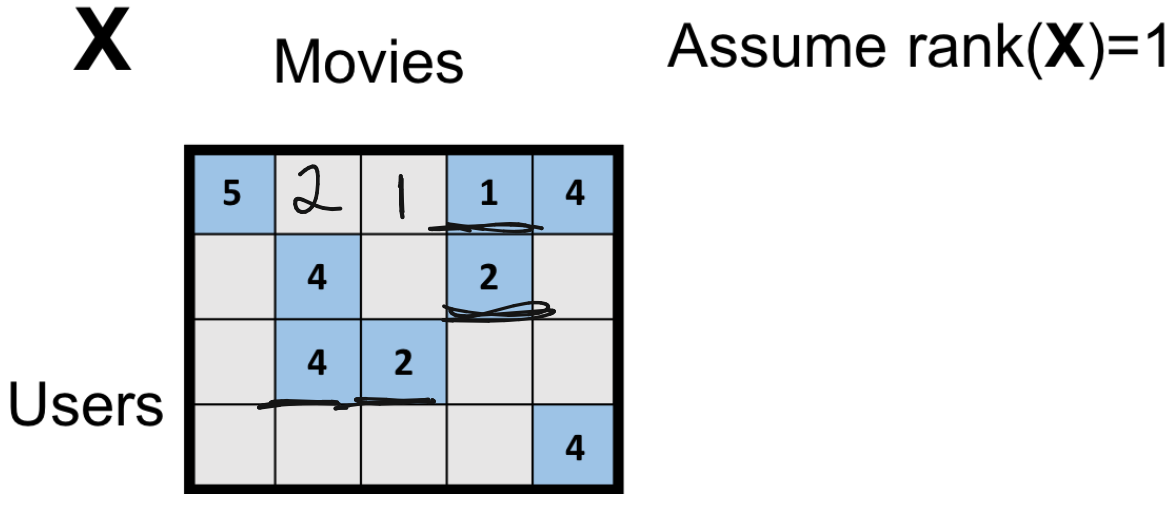
Matrix Completion

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.



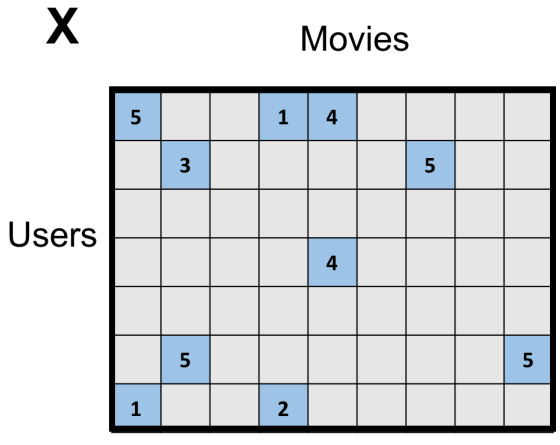
Matrix Completion

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.



Matrix Completion

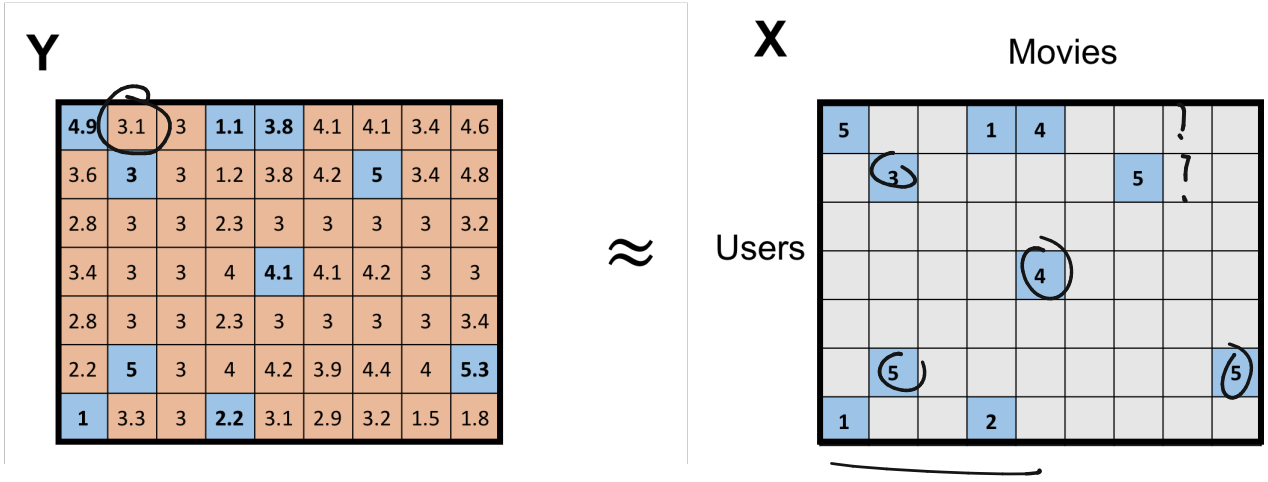
Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.



Solve: $Y = \underset{\mathbf{B} \text{ s.t. } \mathbf{rank}(\mathbf{B}) \leq k}{\arg\min} \sum_{\text{observed } (j,k)} \left[\mathbf{X}_{j,k} - \mathbf{B}_{j,k}\right]^2$

Matrix Completion

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.

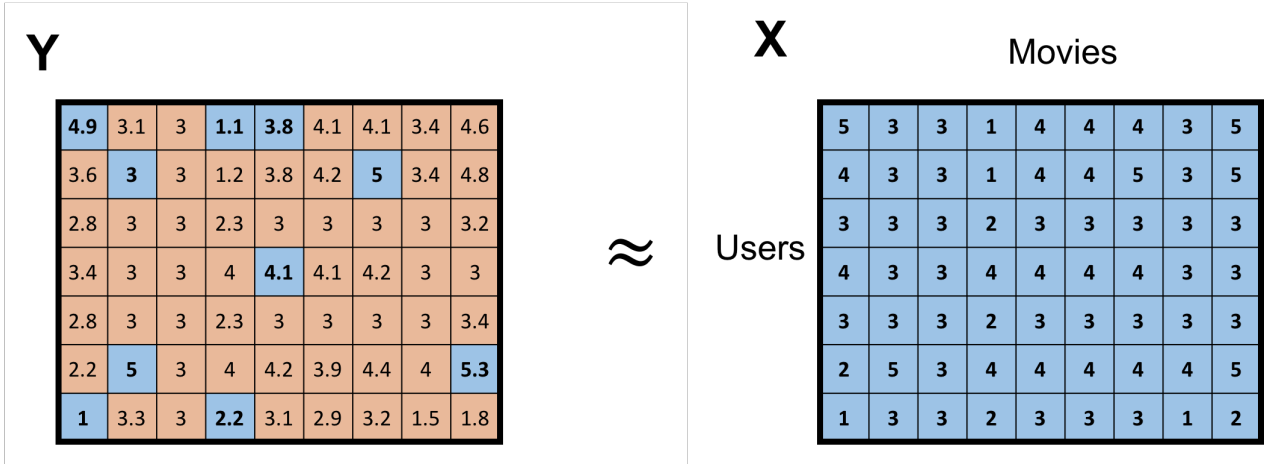


Solve: $Y = \underset{\mathbf{B} \text{ s.t. } \mathrm{rank}(\mathbf{B}) \leq k}{\arg\min} \sum_{\text{observed } (j,k)} \left[\mathbf{X}_{j,k} - \mathbf{B}_{j,k}\right]^2$

$\|X - B\|_F$

Matrix Completion

Consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ which we cannot fully observe but believe is close to rank-k (i.e., well approximated by a rank k matrix). Classic example: the Netflix prize problem.



Solve: $Y = \underset{\mathbf{B} \text{ s.t. } \mathbf{rank}(\mathbf{B}) \leq k}{\arg\min} \sum_{\text{observed } (j,k)} \left[\mathbf{X}_{j,k} - \mathbf{B}_{j,k}\right]^2$

Under certain assumptions, can show that $\mathbf{Y}$ well approximates $\mathbf{X}$ on both the observed and (most importantly) unobserved entries.

Entity Embeddings

Dimensionality reduction embeds d-dimensional vectors into k dimensions. But what about when you want to embed objects other than vectors?

- Documents (for topic-based search and classification)
- Words (to identify synonyms, translations, etc.)
- Nodes in a social network

Entity Embeddings

Dimensionality reduction embeds d-dimensional vectors into k dimensions. But what about when you want to embed objects other than vectors?

- Documents (for topic-based search and classification)
- Words (to identify synonyms, translations, etc.)
- Nodes in a social network

Classic Approach: Convert each item into a (very) high-dimensional feature vector and then apply low-rank approximation.

Example: Latent Semantic Analysis

Term Document Matrix X

	car	loan	house	…	dog	cat
doc_1	0	0	1	0	0	1	1	0	0
doc_2	0	0	0	1	0	1	0	0	0
⋮	1	1	0	1	0	0	0	1	0
	0	0	0	0	0	0	0	1	1
doc_n	1	0	0	0	0	0	0	1	1

Low-Rank Approximation via SVD

$X \approx U_k \Sigma_k V_k^T$

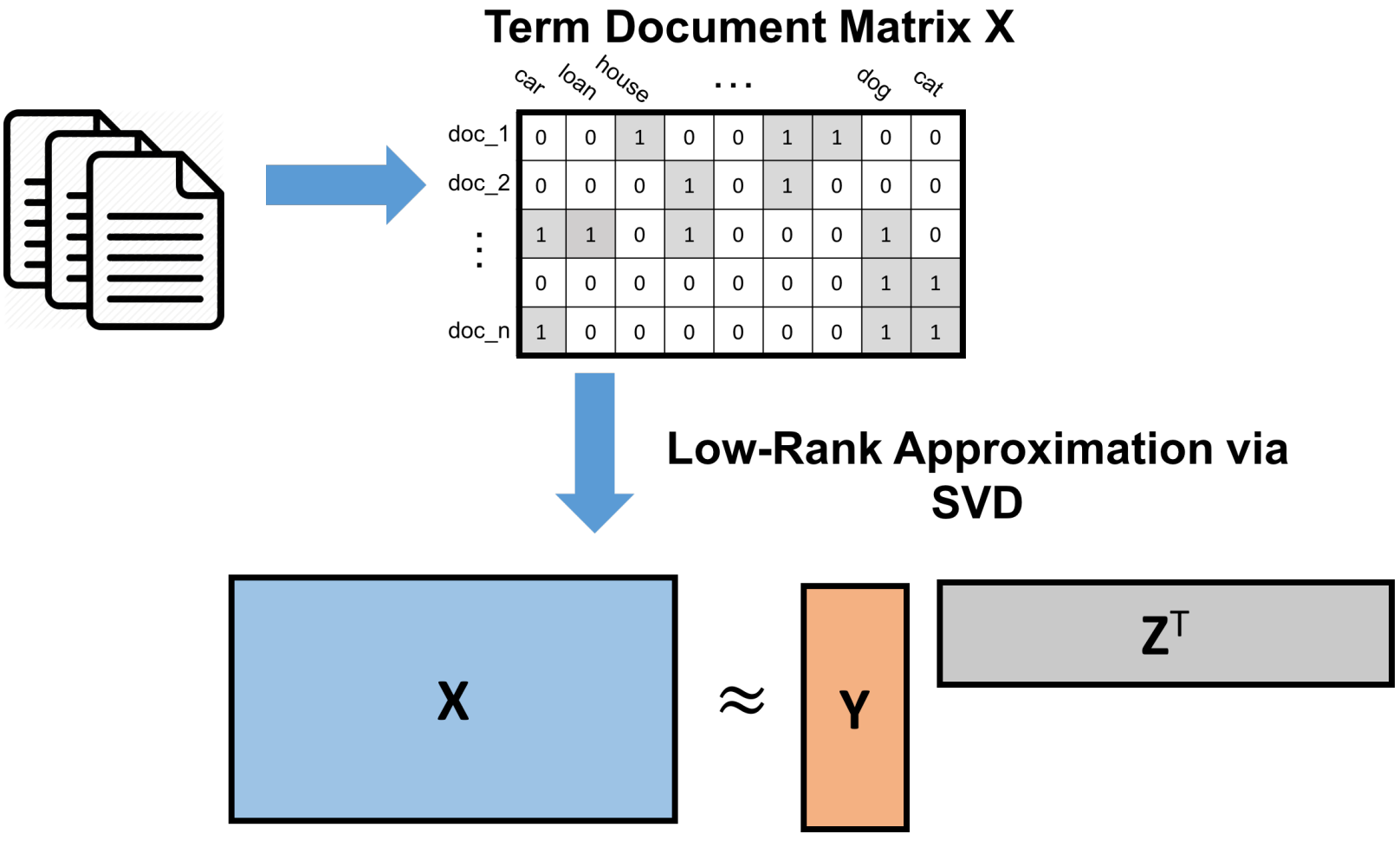
k

k-dim embedding word j

k-dimension embedding of doc i

$(U_k \Sigma_k^{1/2})(\Sigma_k^{1/2} V_k^T)$

Example: Latent Semantic Analysis

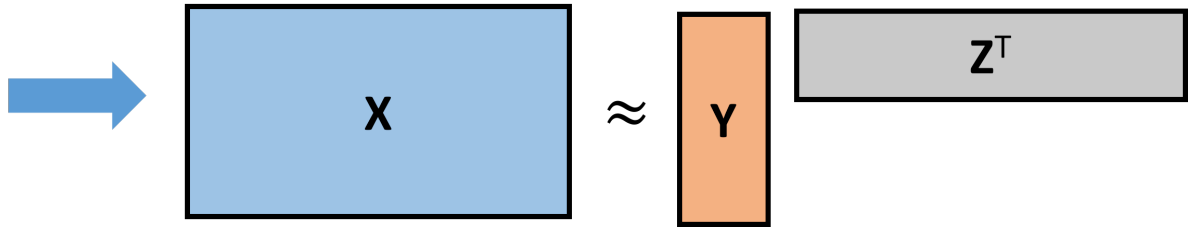


Example: Latent Semantic Analysis

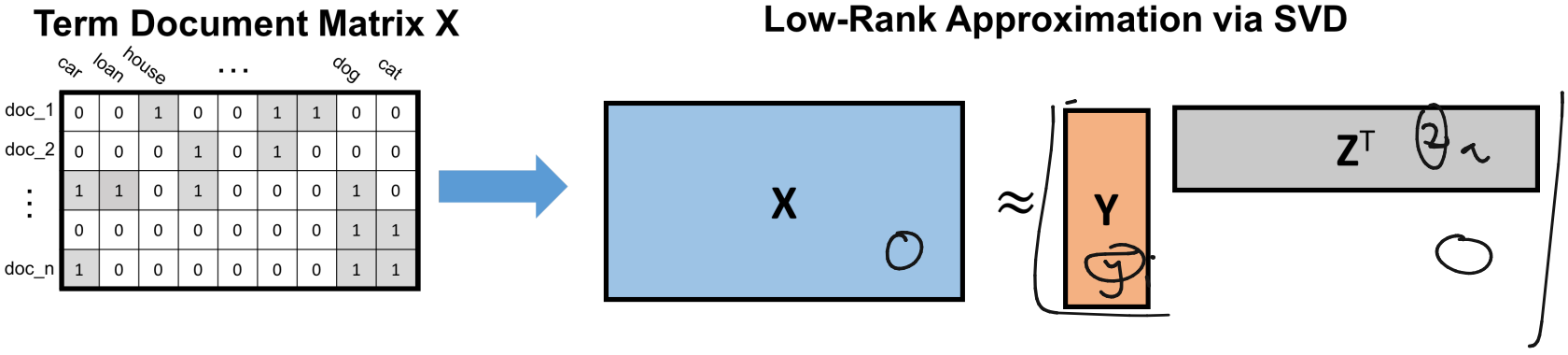
Term Document Matrix X

	car	loan	house	...	dog	cat
doc_1	0	0	1	0	0	1	1	0	0
doc_2	0	0	0	1	0	1	0	0	0
⋮	1	1	0	1	0	0	0	1	0
	0	0	0	0	0	0	0	1	1
doc_n	1	0	0	0	0	0	0	1	1

Low-Rank Approximation via SVD



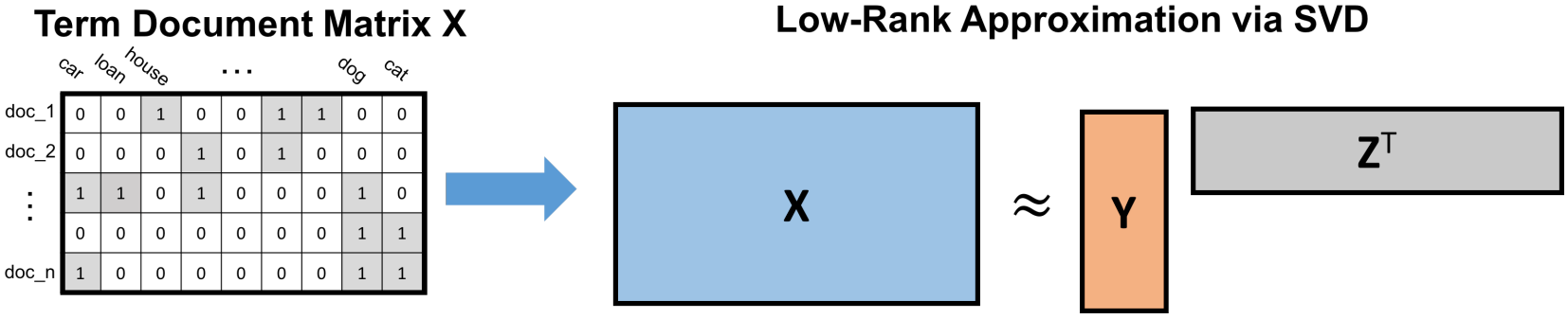
Example: Latent Semantic Analysis



- If the error $\|X - YZ^T\|_F$ is small, then on average,

$$X_{i,a} \approx (YZ^T)_{i,a} = \langle \vec{y}_i, \vec{z}_a \rangle.$$

Example: Latent Semantic Analysis

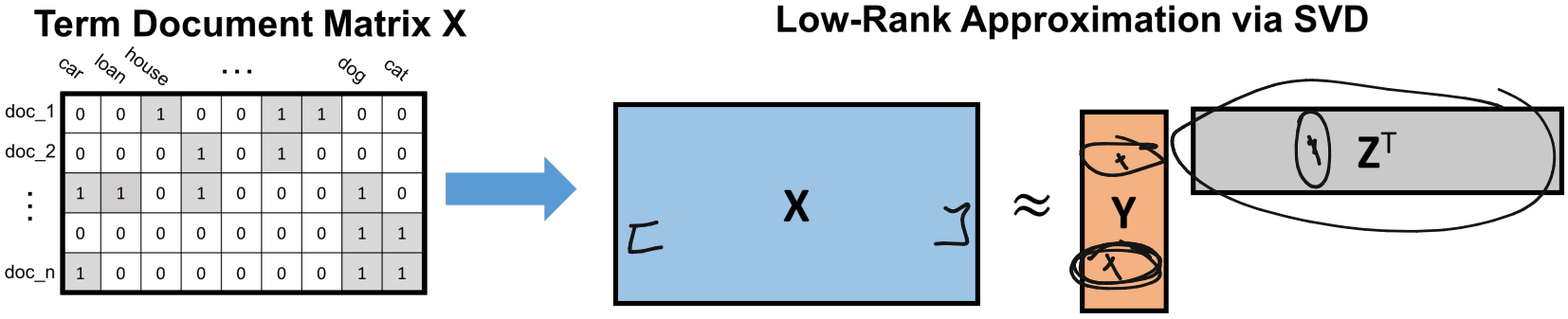


- If the error $\|\mathbf{X} - \mathbf{Y}\mathbf{Z}^T\|_F$ is small, then on average,

$$\mathbf{X}_{i,a} \approx (\mathbf{Y}\mathbf{Z}^T)_{i,a} = \langle \vec{y}_i, \vec{z}_a \rangle.$$

- I.e., $\langle \vec{y}_i, \vec{z}_a \rangle \approx 1$ when doc_i contains $word_a$.

Example: Latent Semantic Analysis



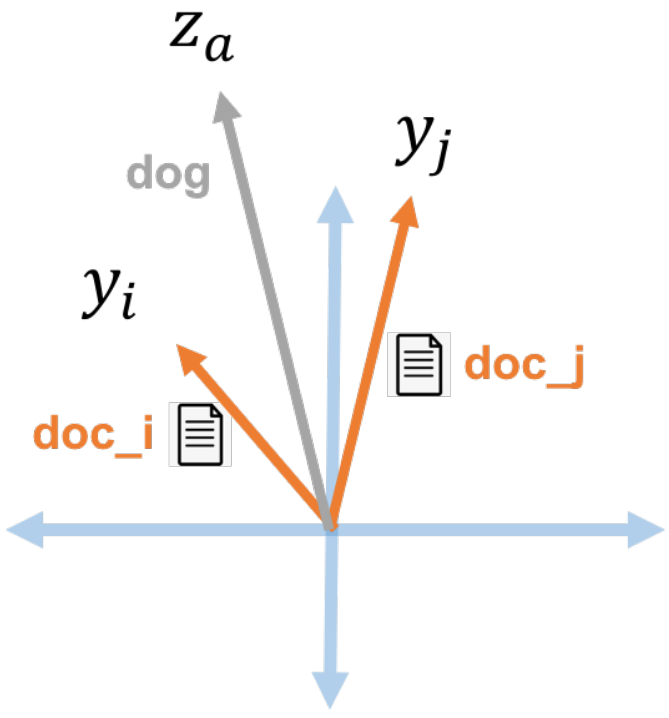
- If the error $\|\mathbf{X} - \mathbf{Y}\mathbf{Z}^T\|_F$ is small, then on average,

$$\mathbf{X}_{i,a} \approx (\mathbf{Y}\mathbf{Z}^T)_{i,a} = \langle \vec{y}_i, \vec{z}_a \rangle.$$

- I.e., $\langle \vec{y}_i, \vec{z}_a \rangle \approx 1$ when doc_i contains $word_a$.
- If doc_i and doc_j both contain $word_a$, $\langle \vec{y}_i, \vec{z}_a \rangle \approx \langle \vec{y}_j, \vec{z}_a \rangle \approx 1$.

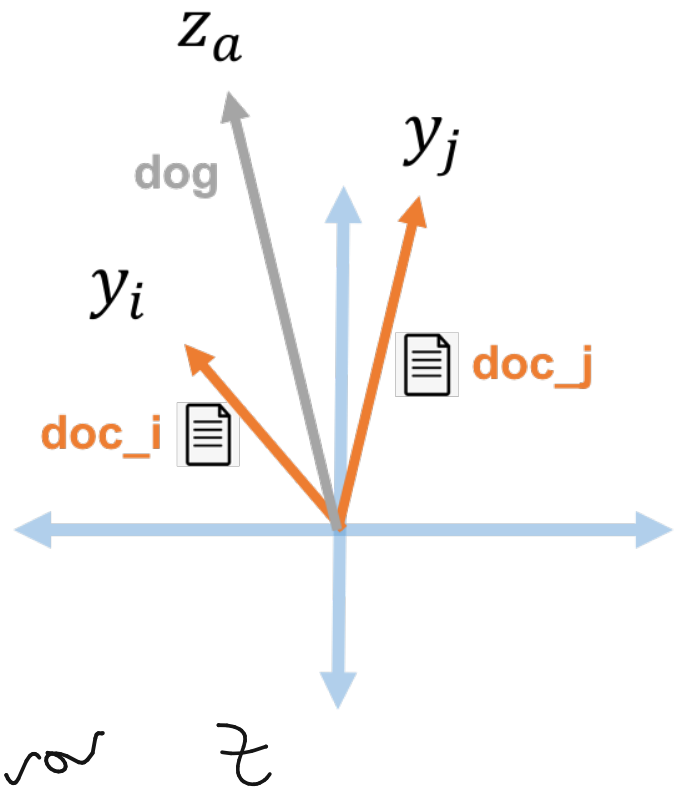
Example: Latent Semantic Analysis

If doc_i and doc_j both contain $word_a$, $\langle \vec{y}_i, \vec{z}_a \rangle \approx \langle \vec{y}_j, \vec{z}_a \rangle \approx 1$



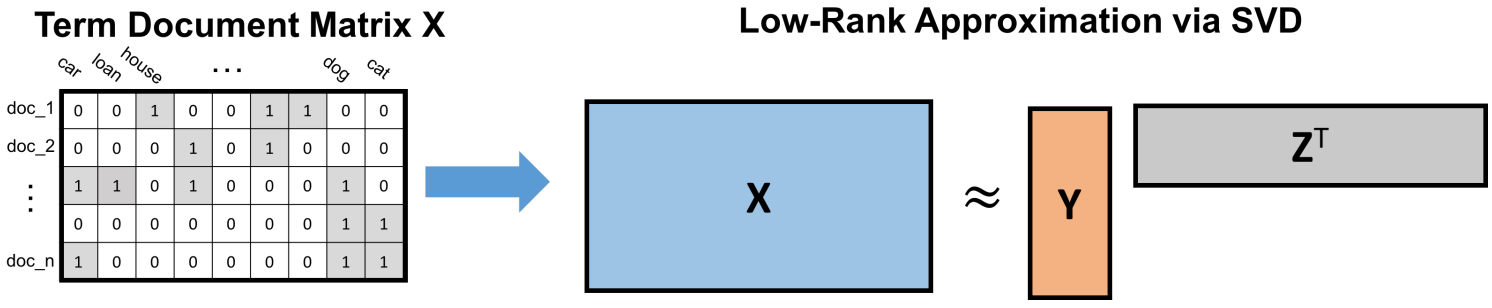
Example: Latent Semantic Analysis

If doc_i and doc_j both contain $word_a$, $\langle \vec{y}_i, \vec{z}_a \rangle \approx \langle \vec{y}_j, \vec{z}_a \rangle \approx 1$



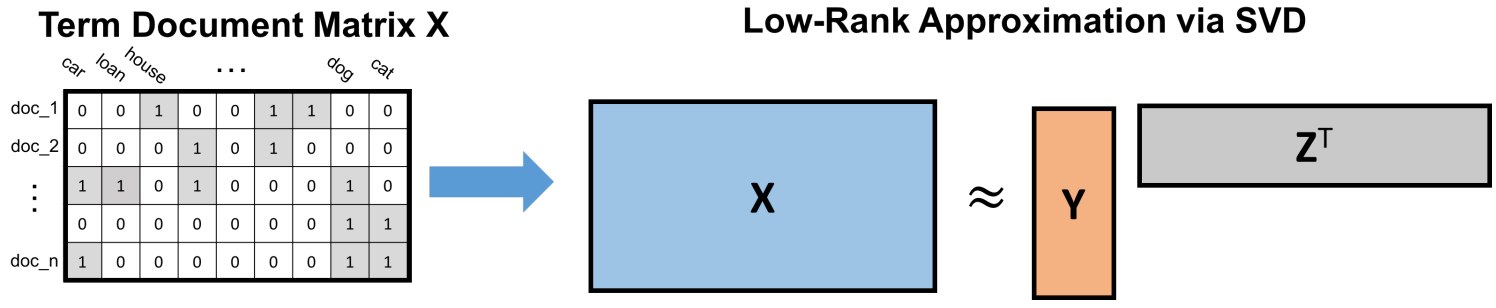
Another View: Each column of Y represents a 'topic'. $\vec{y}_i(j)$ indicates how much doc_i belongs to topic j. $\vec{z}_a(j)$ indicates how much $word_a$ associates with that topic.

Example: Latent Semantic Analysis



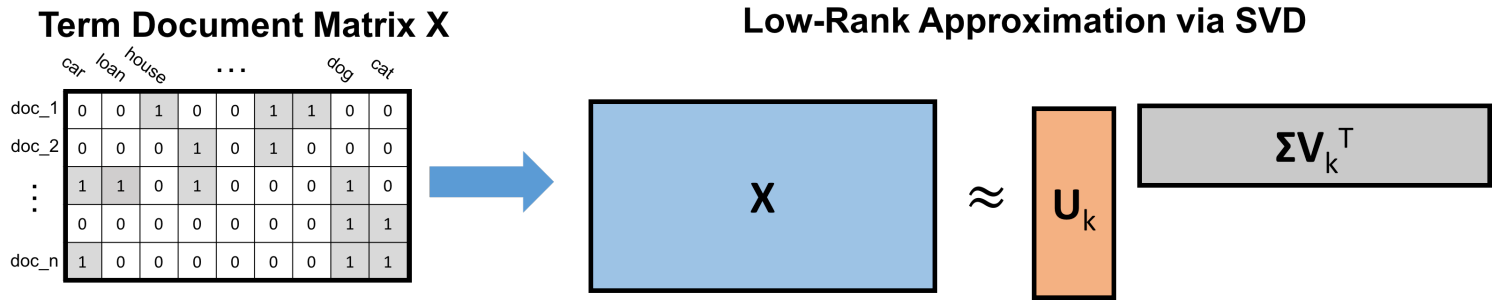
- Just like with documents, $\vec{z}_a$ and $\vec{z}_b$ will tend to have high dot product if $word_a$ and $word_b$ appear in many of the same documents.

Example: Latent Semantic Analysis



- Just like with documents, $\vec{z}_a$ and $\vec{z}_b$ will tend to have high dot product if $word_a$ and $word_b$ appear in many of the same documents.
- In an SVD decomposition we set $\mathbf{Z}^T = \mathbf{\Sigma}_k \mathbf{V}_K^T$.
- The columns of $\mathbf{V}_k$ are equivalently: the top k eigenvectors of $\mathbf{X}^T\mathbf{X}$.

Example: Latent Semantic Analysis

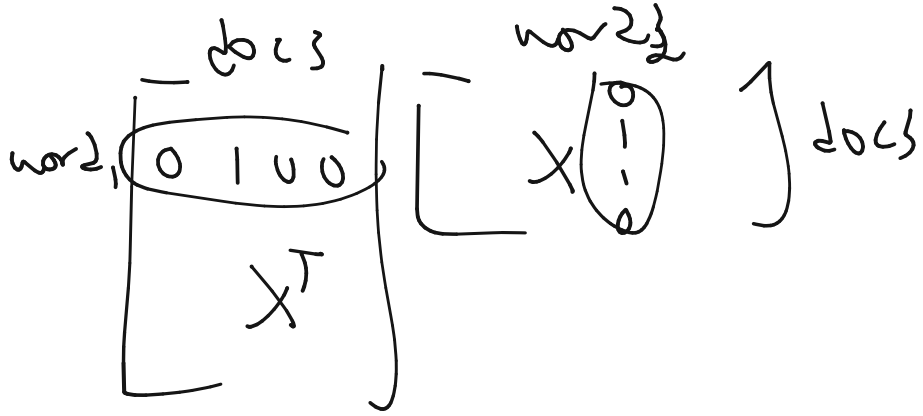


- Just like with documents, $\vec{z}_a$ and $\vec{z}_b$ will tend to have high dot product if $word_a$ and $word_b$ appear in many of the same documents.
- In an SVD decomposition we set $Z^T = \mathbf{\Sigma}_k V_K^T$.
- The columns of V_k are equivalently: the top k eigenvectors of $X^T X$.
- **Claim:** ZZ^T is the best rank-k approximation of $X^T X$. I.e., $\arg\min_{\text{rank}-k\ B} \|X^T X - B\|_F$

Example: Word Embedding

LSA gives a way of embedding words into k-dimensional space.

- Embedding is via low-rank approximation of $\mathbf{X}^T\mathbf{X}$: where $(\mathbf{X}^T\mathbf{X})_{a,b}$ is the number of documents that both $word_a$ and $word_b$ appear in.



Example: Word Embedding

LSA gives a way of embedding words into k-dimensional space.

- Embedding is via low-rank approximation of $\mathbf{X}^T\mathbf{X}$: where $(\mathbf{X}^T\mathbf{X})_{a,b}$ is the number of documents that both $word_a$ and $word_b$ appear in.
- Think about $\mathbf{X}^T\mathbf{X}$ as a similarity matrix (gram matrix, kernel matrix) with entry (a, b) being the similarity between $word_a$ and $word_b$.

Example: Word Embedding

LSA gives a way of embedding words into k-dimensional space.

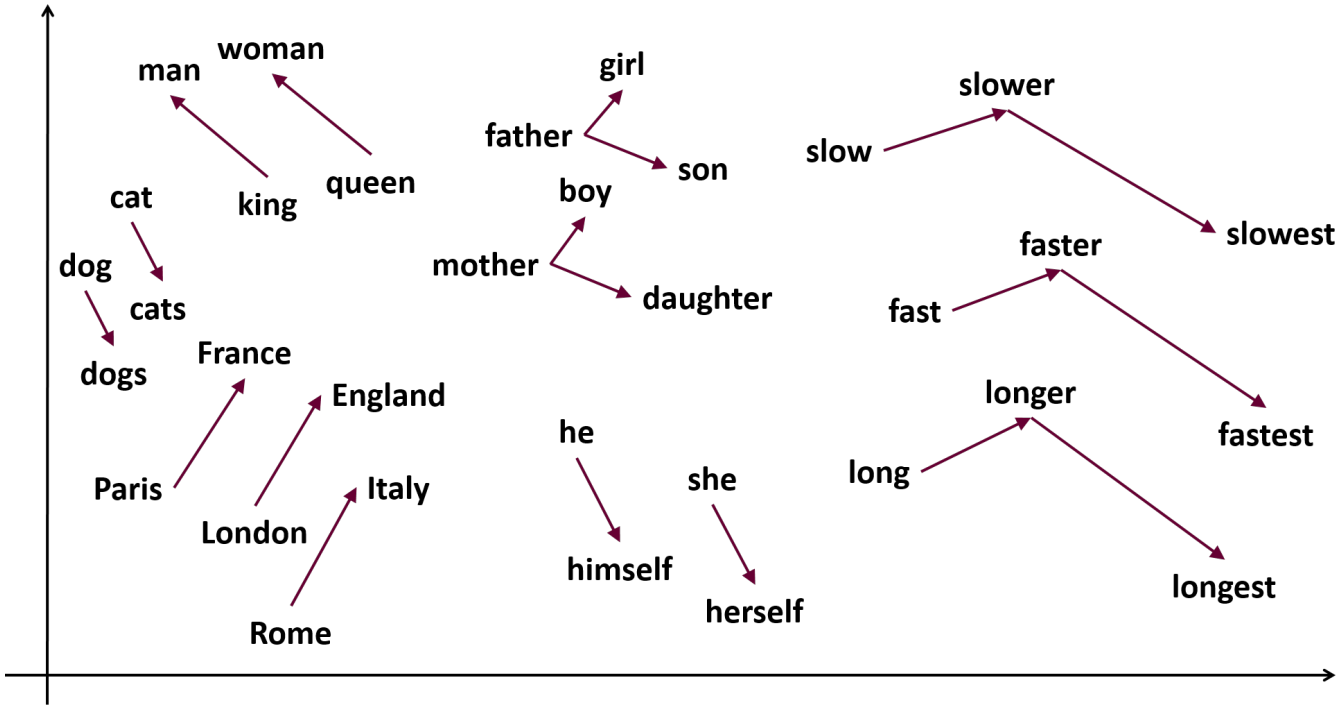
- Embedding is via low-rank approximation of $\mathbf{X}^T\mathbf{X}$: where $(\mathbf{X}^T\mathbf{X})_{a,b}$ is the number of documents that both $word_a$ and $word_b$ appear in.
- Think about $\mathbf{X}^T\mathbf{X}$ as a similarity matrix (gram matrix, kernel matrix) with entry (a, b) being the similarity between $word_a$ and $word_b$.
- Many ways to measure similarity: number of sentences both occur in, number of times both appear in the same window of w words, in similar positions of documents in different languages, etc.

Example: Word Embedding

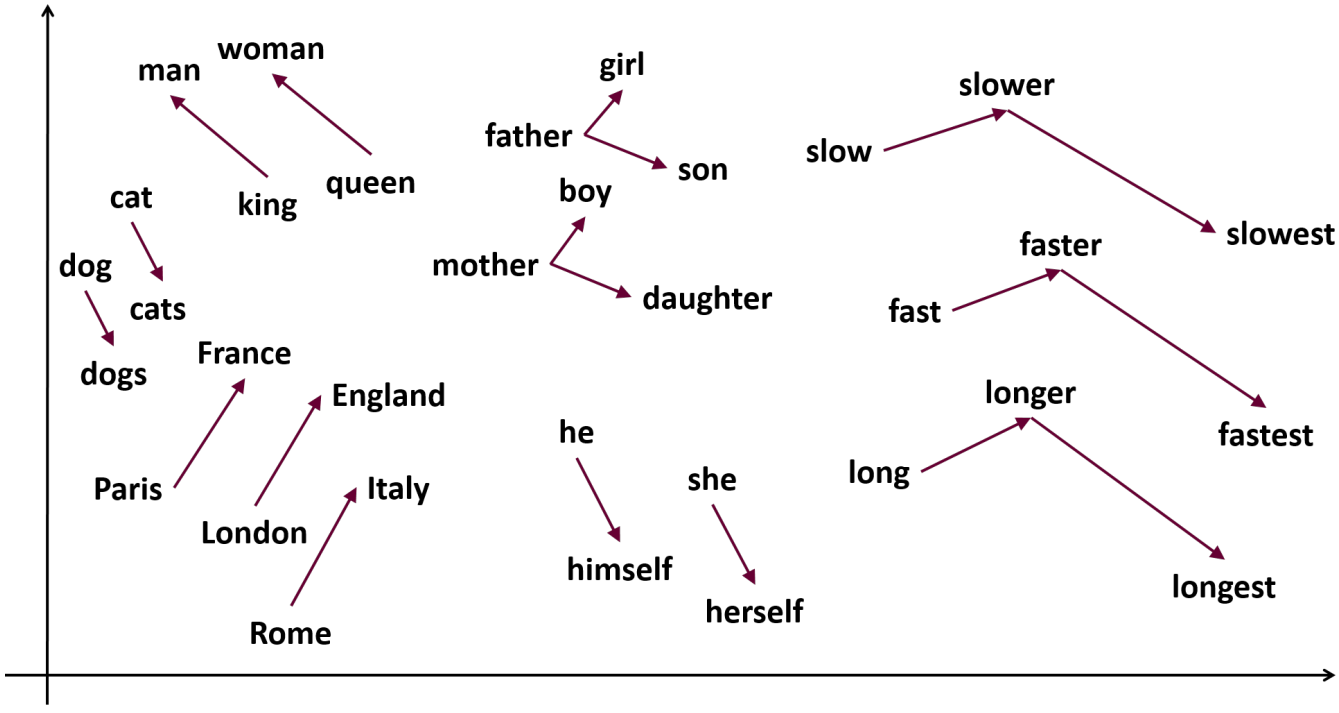
LSA gives a way of embedding words into k-dimensional space.

- Embedding is via low-rank approximation of $\mathbf{X}^T\mathbf{X}$: where $(\mathbf{X}^T\mathbf{X})_{a,b}$ is the number of documents that both $word_a$ and $word_b$ appear in.
- Think about $\mathbf{X}^T\mathbf{X}$ as a similarity matrix (gram matrix, kernel matrix) with entry (a, b) being the similarity between $word_a$ and $word_b$.
- Many ways to measure similarity: number of sentences both occur in, number of times both appear in the same window of w words, in similar positions of documents in different languages, etc.
- Replacing $\mathbf{X}^T\mathbf{X}$ with these different metrics (sometimes appropriately transformed) leads to popular word embedding algorithms: word2vec, GloVe, fastText, etc.

Example: Word Embedding



Example: Word Embedding



Note: word2vec is typically described as a neural-network method, but can be viewed as just a low-rank approximation of a specific similarity matrix. *Neural word embedding as implicit matrix factorization*, Levy and Goldberg.

Questions?