COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco

University of Massachusetts Amherst. Spring 2020.
Lecture 16

SUMMARY

Last Class: Low-Rank Approximation, Eigendecomposition, and PCA

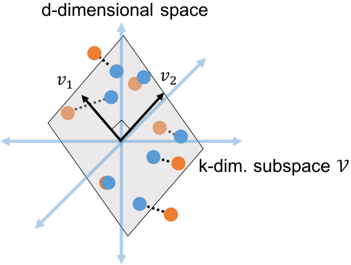
- Can approximate data lying close to in a k-dimensional subspace by projecting data points into that space.
- Finding the best k-dimensional subspace via eigendecomposition (PCA).
- Measuring error in terms of the eigenvalue spectrum.

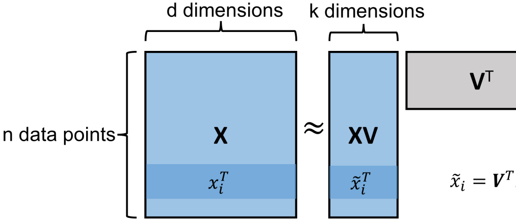
This Class: Finish Low-Rank Approximation and Connection to the singular value decomposition (SVD)

- Finish up PCA – runtime considerations and picking k.
- View of optimal low-rank approximation using the SVD.
- Applications of low-rank approximation beyond compression.

BASIC SET UP

Set Up: Assume that data points $\vec{x}_1, \ldots, \vec{x}_n$ lie close to any k-dimensional subspace $\mathcal{V}$ of $\mathbb{R}^d$. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the data matrix.





Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for $\mathcal{V}$ and $\mathbf{V} \in \mathbb{R}^{d \times k}$ be the matrix with these vectors as its columns.

- $\mathbf{V}\mathbf{V}^T \in \mathbb{R}^{d \times d}$ is the projection matrix onto $\mathcal{V}$.
- $\mathbf{X} \approx \mathbf{X}(\mathbf{V}\mathbf{V}^T)$. Gives the closest approximation to $\mathbf{X}$ with rows in $\mathcal{V}$.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

LOW-RANK APPROXIMATION VIA EIGENDECOMPOSITION

$\mathbf{V}$ minimizing $\|\mathbf{X} - \mathbf{X}\mathbf{V}\mathbf{V}^T\|_F^2$ is given by:

$$\underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\|_F^2 = \sum_{j=1}^{k} \|\mathbf{X}\vec{v}_j\|_2^2$$

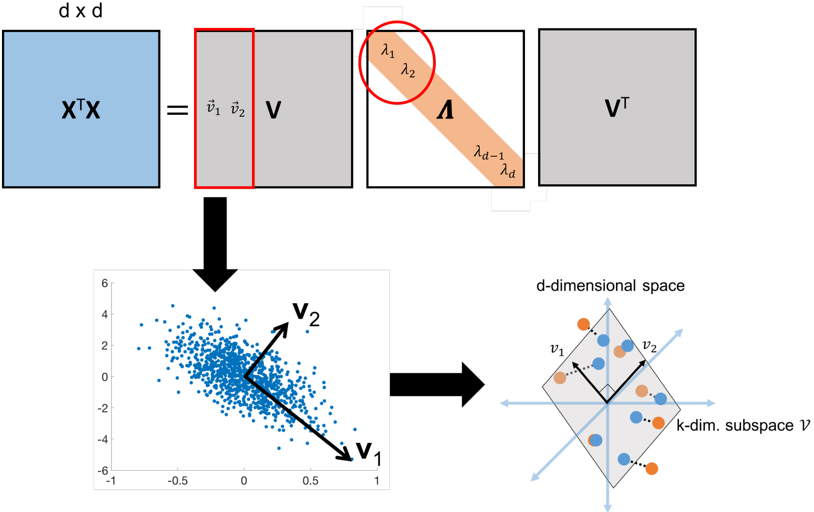
Solution via eigendecomposition: Letting $\mathbf{V}_k$ have columns $\vec{v}_1, \ldots, \vec{v}_k$ corresponding to the top k eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$,

$$\mathbf{V}_k = \underset{\text{orthonormal } \mathbf{V} \in \mathbb{R}^{d \times k}}{\arg\max} \|\mathbf{X}\mathbf{V}\|_F^2$$

- Proof via Courant-Fischer and greedy maximization.
- Approximation error is $\|\mathbf{X}\|_F^2 - \|\mathbf{X}\mathbf{V}_k\|_F^2 = \sum_{i=k+1}^{d} \lambda_i(\mathbf{X}^T\mathbf{X})$.

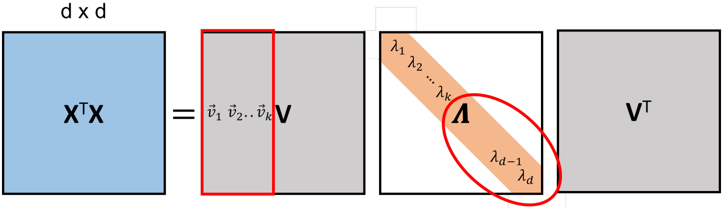
$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: orthogonal basis for subspace $\mathcal{V}$. $\mathbf{V} \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

LOW-RANK APPROXIMATION VIA EIGENDECOMPOSITION



SPECTRUM ANALYSIS

Plotting the spectrum of the covariance matrix $\mathbf{X}^T\mathbf{X}$ (its eigenvalues) shows how compressible $\mathbf{X}$ is using low-rank approximation (i.e., how close $\vec{x}_1, \ldots, \vec{x}_n$ are to a low-dimensional subspace).

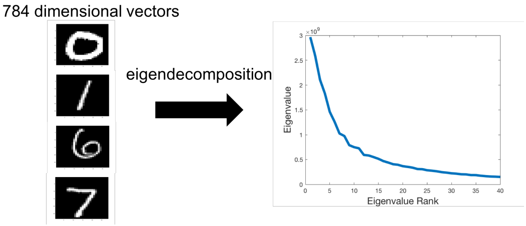


error of optimal low rank approximation

- Choose k to balance accuracy and compression.
- Often at an ‘elbow’.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: top eigenvectors of $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

SPECTRUM ANALYSIS



784 dimensional vectors

eigendecomposition

Exercise: Show that the eigenvalues of $\mathbf{X}^T\mathbf{X}$ are always positive.

Hint: Use that $\lambda_j = \vec{v}_j^T \mathbf{X}^T \mathbf{X} \vec{v}_j$.

INTERPRETATION IN TERMS OF CORRELATION

Recall: Low-rank approximation is possible when our data features are correlated.

10000* bathrooms+ 10* (sq. ft.) ≈ list price

	bedrooms	bathrooms	sq.ft.	floors	list price	sale price
home 1	2	2	1800	2	200,000	195,000
home 2	4	2.5	2700	1	300,000	310,000
⋮	⋮	⋮	⋮	⋮	⋮	⋮
home n	5	3.5	3600	3	450,000	450,000

Our compressed dataset is $\mathbf{C} = \mathbf{X}\mathbf{V}_k$ where the columns of $\mathbf{V}_k$ are the top k eigenvectors of $\mathbf{X}^T\mathbf{X}$.

What is the covariance of $\mathbf{C}$? $\mathbf{C}^T\mathbf{C} = \mathbf{V}_k^T\mathbf{X}^T\mathbf{X}\mathbf{V}_k = \mathbf{V}_k^T\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T\mathbf{V}_k = \boldsymbol{\Lambda}_k$

Covariance becomes diagonal. I.e., all correlations have been removed. Maximal compression.

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: top eigenvectors of $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

ALGORITHMIC CONSIDERATIONS

What is the runtime to compute an optimal low-rank approximation?

- Computing the covariance matrix $\mathbf{X}^T\mathbf{X}$ requires $O(nd^2)$ time.
- Computing its full eigendecomposition to obtain $\vec{v}_1, \ldots, \vec{v}_k$ requires $O(d^3)$ time (similar to the inverse $(\mathbf{X}^T\mathbf{X})^{-1}$).

Many faster iterative and randomized methods. Runtime is roughly $\tilde{O}(ndk)$ to output just to top k eigenvectors $\vec{v}_1, \ldots, \vec{v}_k$.

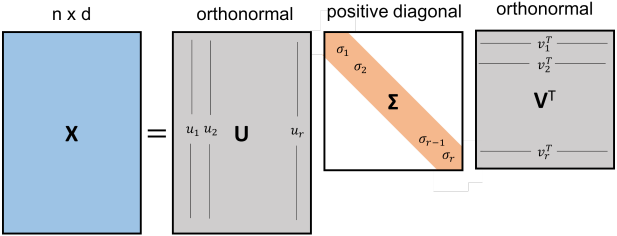
- Will see in a few classes (power method, Krylov methods).

$\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$: data points, $\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$: top eigenvectors of $\mathbf{X}^T\mathbf{X}$, $\mathbf{V}_k \in \mathbb{R}^{d \times k}$: matrix with columns $\vec{v}_1, \ldots, \vec{v}_k$.

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 $\text{rank}(\mathbf{X}) = r$ can be written as $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$.

- $\mathbf{U}$ 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).
- $\boldsymbol{\Sigma}$ is diagonal with elements $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ (singular values).



The ‘swiss army knife’ of modern linear algebra.

CONNECTION OF THE SVD TO EIGENDECOMPOSITION

Writing $\mathbf{X} \in \mathbb{R}^{n \times d}$ in its singular value decomposition $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$:

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T \text{ (the eigendecomposition)}$$

Similarly: $\mathbf{X}\mathbf{X}^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^T = \mathbf{U}\boldsymbol{\Sigma}^2\mathbf{U}^T$.

The left and right singular vectors are the eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$ and the gram matrix $\mathbf{X}\mathbf{X}^T$ respectively.

So, letting $\mathbf{V}_k \in \mathbb{R}^{d \times k}$ have columns equal to $\vec{v}_1, \ldots, \vec{v}_k$, we know that $\mathbf{X}\mathbf{V}_k\mathbf{V}_k^T$ is the best rank-k approximation to $\mathbf{X}$ (given by PCA).

What about $\mathbf{U}_k\mathbf{U}_k^T\mathbf{X}$ where $\mathbf{U}_k \in \mathbb{R}^{n \times k}$ has columns equal to $\vec{u}_1, \ldots, \vec{u}_k$? Gives exactly the same approximation!

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \operatorname{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \operatorname{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\operatorname{rank}(\mathbf{X}) \times \operatorname{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

THE SVD AND OPTIMAL LOW-RANK APPROXIMATION

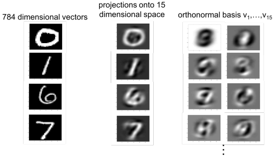
The best low-rank approximation to $\mathbf{X}$:

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

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

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

Row (data point) compression



Column (feature) compression

10000* bathrooms+ 10* (sq. ft.) ≈ list price

	bedrooms	bathrooms	sq.ft.	floors	list price	sale price
home 1	2	2	1800	2	200,000	195,000
home 2	4	2.5	2700	1	300,000	310,000
⋮	⋮	⋮	⋮	⋮	⋮	⋮
home n	5	3.5	3600	3	450,000	450,000

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:

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

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

THE SVD AND OPTIMAL LOW-RANK APPROXIMATION

$\mathbf{X} \in \mathbb{R}^{n \times d}$: data matrix, $\mathbf{U} \in \mathbb{R}^{n \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{u}_1, \vec{u}_2, \ldots$ (left singular vectors), $\mathbf{V} \in \mathbb{R}^{d \times \text{rank}(\mathbf{X})}$: matrix with orthonormal columns $\vec{v}_1, \vec{v}_2, \ldots$ (right singular vectors), $\boldsymbol{\Sigma} \in \mathbb{R}^{\text{rank}(\mathbf{X}) \times \text{rank}(\mathbf{X})}$: positive diagonal matrix containing singular values of $\mathbf{X}$.

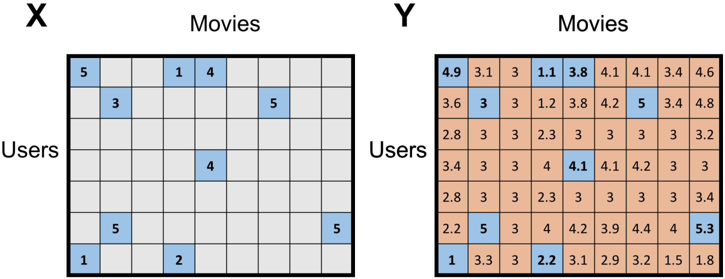
APPLICATIONS OF LOW-RANK APPROXIMATION

Rest of Class: Examples of how low-rank approximation is applied in a variety of data science applications.

- Used for many reasons other than dimensionality reduction/data 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). Classic example: the Netflix prize problem.



5			1	4				
	3					5		
				4				
	5							5
1			2					

4.9	3.1	3	**1.1**	**3.8**	4.1	4.1	3.4	4.6
3.6	**3**	3	1.2	3.8	4.2	**5**	3.4	4.8
2.8	3	3	2.3	3	3	3	3	3.2
3.4	3	3	4	**4.1**	4.1	4.2	3	3
2.8	3	3	2.3	3	3	3	3	3.4
2.2	**5**	3	4	4.2	3.9	4.4	4	**5.3**
1	3.3	3	**2.2**	3.1	2.9	3.2	1.5	1.8

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

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

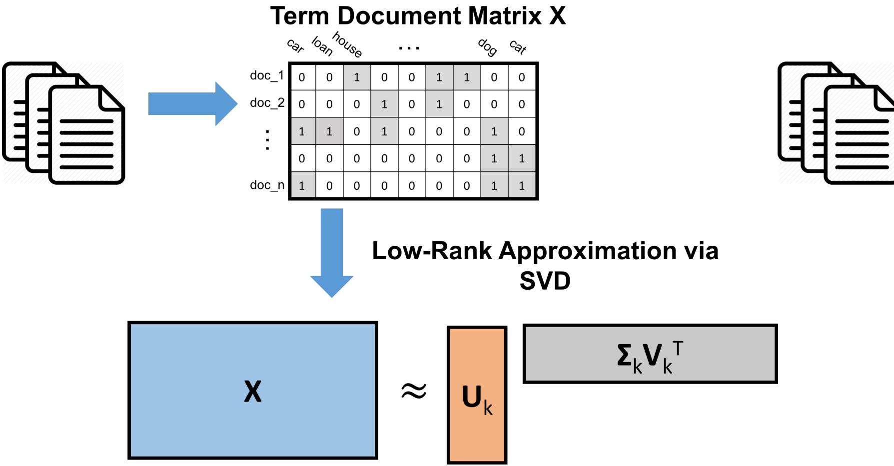
ENTITY EMBEDDINGS

Dimensionality reduction embeds d-dimensional vectors into d' 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

Usual Approach: Convert each item into a high-dimensional feature vector and then apply low-rank approximation.

EXAMPLE: LATENT SEMANTIC ANALYSIS



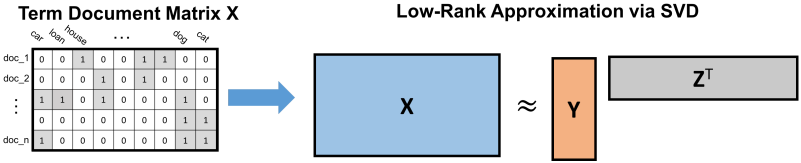
Term Document Matrix X

	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

$$X \approx U_k \; \Sigma_k V_k^T$$

EXAMPLE: LATENT SEMANTIC ANALYSIS



	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

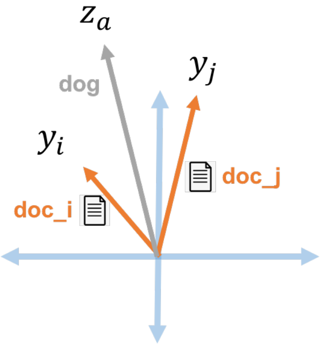
- If the error $\|X - YZ^T\|_F$ is small, then on average,

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

- 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 $word_a$, $\langle \vec{y}_i, \vec{z}_a \rangle \approx \langle \vec{y}_j, \vec{z}_a \rangle = 1$.

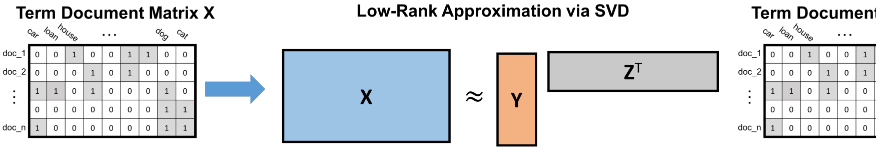
EXAMPLE: LATENT SEMANTIC ANALYSIS

If *doc*$_i$ and *doc*$_j$ both contain *word*$_a$, $\langle \vec{y}_i, \vec{z}_a \rangle \approx \langle \vec{y}_j, \vec{z}_a \rangle = 1$



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



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

Term Document

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

- Just like with documents, $\vec{z}_a$ and $\vec{z}_b$ will tend to have high dot product if *word*$_i$ and *word*$_j$ appear in many of the same documents.
- In an SVD decomposition we set $\mathbf{Z} = \boldsymbol{\Sigma}_k \mathbf{V}_K^T$.
- The columns of $\mathbf{V}_k$ are equivalently: the top k eigenvectors of $\mathbf{X}^T\mathbf{X}$. The eigendecomposition of $\mathbf{X}^T\mathbf{X}$ is $\mathbf{X}^T\mathbf{X} = \mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T$.
- What is the best rank-k approximation of $\mathbf{X}^T\mathbf{X}$? I.e. $\arg\min_{\text{rank}-k\ \mathbf{B}} \|\mathbf{X}^T\mathbf{X} - \mathbf{B}\|_F$
- $\mathbf{X}^T\mathbf{X} = \mathbf{V}_k\boldsymbol{\Sigma}_k^2\mathbf{V}_k^T = \mathbf{Z}\mathbf{Z}^T$.

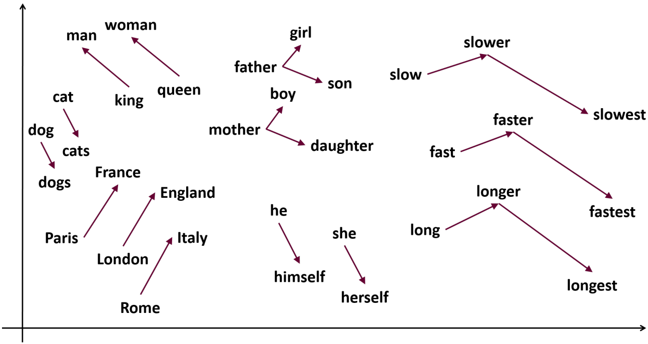
EXAMPLE: WORD EMBEDDING

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, fastTest, etc.

EXAMPLE: WORD EMBEDDING



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

SUMMARY

Summary:

- Can use the SVD to understand optimal low-rank approximation in terms of the dual row/column projection view: $\mathbf{X}\mathbf{V}_k\mathbf{V}_k^T = \mathbf{U}_k\mathbf{U}_k^T\mathbf{X} = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T$.
- A generalization of eigendecomposition: singular vectors are eigenvectors of $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$.
- Applications to low-rank approximation to matrix completion and entity embeddings.

Next Time: Low-rank representations of graphs and networks. Beginning of spectral graph theory.