

Introduction to Information Retrieval

Lecture 19 LSI

Thanks to Thomas Hofmann for some slides.

Today's topic

- Latent Semantic Indexing
 - Term-document matrices are very large
 - But the number of topics that people talk about is small (in some sense)
 - Clothes, movies, politics, ...
 - Can we represent the term-document space by a lower dimensional latent space?

Linear Algebra Background

Eigenvalues & Eigenvectors

- Eigenvectors** (for a square $m \times m$ matrix S)

$$Sv = \lambda v$$

(right) eigenvector $v \in \mathbb{R}^m \neq 0$ eigenvalue $\lambda \in \mathbb{R}$

Example
 $\begin{pmatrix} 1 & -2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix} = -1 \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

- How many eigenvalues are there at most?

$$Sv = \lambda v \iff (S - \lambda I)v = 0$$

only has a non-zero solution if $|S - \lambda I| = 0$

this is a m -th order equation in λ which can have

at most m distinct solutions (roots of the characteristic polynomial) - can be complex even though S is

Matrix-vector multiplication

$S = \begin{bmatrix} 30 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ has eigenvalues 30, 20, 1 with corresponding eigenvectors

$$v_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad v_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad v_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

On each eigenvector, S acts as a multiple of the identity matrix: but as a different multiple on each.

Any vector (say $x = \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$) can be viewed as a combination of the eigenvectors:
 $x = 2v_1 + 4v_2 + 6v_3$

Matrix vector multiplication

- Thus a matrix-vector multiplication such as Sx (S , x as in the previous slide) can be rewritten in terms of the eigenvalues/vectors:

$$Sx = S(2v_1 + 4v_2 + 6v_3)$$

$$Sx = 2Sv_1 + 4Sv_2 + 6Sv_3 = 2\lambda_1v_1 + 4\lambda_2v_2 + 6\lambda_3v_3$$

$$Sx = 60v_1 + 80v_2 + 6v_3$$

- Even though x is an arbitrary vector, the action of S on x is determined by the eigenvalues/vectors.

Matrix vector multiplication

- Suggestion: the effect of "small" eigenvalues is small.
- If we ignored the smallest eigenvalue (1), then instead of

$$\begin{pmatrix} 60 \\ 80 \\ 6 \end{pmatrix} \quad \text{we would get} \quad \begin{pmatrix} 60 \\ 80 \\ 0 \end{pmatrix}$$

- These vectors are similar (in cosine similarity, etc.)

Eigenvalues & Eigenvectors

For symmetric matrices, eigenvectors for distinct eigenvalues are **orthogonal**

$$Sv_{\{1,2\}} = \lambda_{\{1,2\}}v_{\{1,2\}}, \text{ and } \lambda_1 \neq \lambda_2 \Rightarrow v_1 \bullet v_2 = 0$$

All eigenvalues of a real symmetric matrix are **real**.

for complex λ , if $|S - \lambda I| = 0$ and $S = S^T \Rightarrow \lambda \in \mathfrak{R}$

All eigenvalues of a **positive semidefinite** matrix are **non-negative**

$$\forall w \in \mathfrak{R}^n, w^T S w \geq 0, \text{ then if } S v = \lambda v \Rightarrow \lambda \geq 0$$

Example

- Let $S = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$ ← **Real, symmetric.**
- Then $S - \lambda I = \begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix} \Rightarrow (2-\lambda)^2 - 1 = 0.$
- The eigenvalues are 1 and 3 (nonnegative, real).
- The eigenvectors are orthogonal (and real):

$$\begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Plug in these values and solve for eigenvectors.

Eigen/diagonal Decomposition

- Let $S \in \mathbb{R}^{m \times m}$ be a **square** matrix with **m linearly independent eigenvectors** (a "non-defective" matrix)

- **Theorem:** Exists an **eigen decomposition**

$$S = U \Lambda U^{-1} \quad \text{diagonal}$$

Unique for distinct eigenvalues

- (cf. matrix diagonalization theorem)
- Columns of U are **eigenvectors** of S
- Diagonal elements of Λ are **eigenvalues** of S

$$\Lambda = \text{diag}(\lambda_1, \dots, \lambda_m), \quad \lambda_i \geq \lambda_{i+1}$$

Diagonal decomposition: why/how

Let U have the eigenvectors as columns: $U = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix}$

Then, SU can be written

$$SU = S \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix} = \begin{bmatrix} \lambda_1 v_1 & \dots & \lambda_n v_n \end{bmatrix} = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \dots & \\ & & \lambda_n \end{bmatrix}$$

Thus $SU = U\Lambda$, or $U^{-1}SU = \Lambda$

And $S = U\Lambda U^{-1}$.

Diagonal decomposition - example

Recall $S = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}; \lambda_1 = 1, \lambda_2 = 3.$

The eigenvectors $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ form $U = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$

Inverting, we have $U^{-1} = \begin{bmatrix} 1/2 & -1/2 \\ 1/2 & 1/2 \end{bmatrix}$ ← Recall $UU^{-1} = I$.

Then, $S = U\Lambda U^{-1} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 1/2 & -1/2 \\ 1/2 & 1/2 \end{bmatrix}$

Example continued

Let's divide U (and multiply U^{-1}) by $\sqrt{2}$

$$\text{Then, } S = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

$Q \quad A \quad (Q^{-1} = Q^T)$

Why? Stay tuned ...

Symmetric Eigen Decomposition

- If $S \in \mathbb{R}^{m \times m}$ is a **symmetric** matrix:
- **Theorem:** There exists a (unique) **eigen decomposition** $S = Q\Lambda Q^T$
- where Q is **orthogonal**:
 - $Q^{-1} = Q^T$
 - Columns of Q are normalized eigenvectors
 - Columns are orthogonal.
 - (everything is real)

Exercise

- Examine the symmetric eigen decomposition, if any, for each of the following matrices:

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

Time out!

- I came to this class to learn about text retrieval and mining, not have my linear algebra past dredged up again ...
 - *But if you want to dredge, Strang's Applied Mathematics is a good place to start.*
- What do these matrices have to do with text?
- Recall $M \times N$ term-document matrices ...
- But everything so far needs square matrices – so ...

Singular Value Decomposition

For an $M \times N$ matrix A of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

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

$M \times M \quad M \times N \quad N \times N$

The columns of U are orthogonal eigenvectors of AA^T .

The columns of V are orthogonal eigenvectors of $A^T A$.

Eigenvalues $\lambda_1 \dots \lambda_r$ of AA^T are the eigenvalues of $A^T A$.

$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma = \text{diag}(\sigma_1 \dots \sigma_r) \leftarrow \text{Singular values.}$$

Singular Value Decomposition

- Illustration of SVD dimensions and sparseness

SVD example

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

Thus $M=3, N=2$. Its SVD is

$$\begin{bmatrix} 0 & 2/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

Typically, the singular values arranged in decreasing order.

Low-rank Approximation

- SVD can be used to compute optimal **low-rank approximations**.
- Approximation problem: Find A_k of rank k such that

$$A_k = \min_{X: \text{rank}(X)=k} \|A - X\|_F \leftarrow \text{Frobenius norm}$$

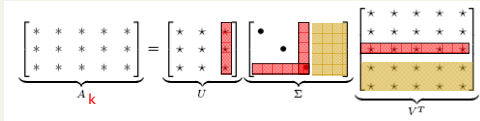
$$\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}$$

A_k and X are both $m \times n$ matrices.
Typically, want $k \ll r$.

Low-rank Approximation

- Solution via SVD

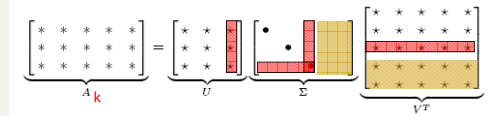
$$A_k = U \text{diag}(\sigma_1, \dots, \sigma_k, \underbrace{0, \dots, 0}_{\text{set smallest } r-k \text{ singular values to zero}}) V^T$$



$$A_k = \sum_{i=1}^k \sigma_i u_i v_i^T \leftarrow \text{column notation: sum of rank 1 matrices}$$

Reduced SVD

- If we retain only k singular values, and set the rest to 0, then we don't need the matrix parts in red
- Then Σ is $k \times k$, U is $M \times k$, V^T is $k \times N$, and A_k is $M \times N$
- This is referred to as the reduced SVD
- It is the convenient (space-saving) and usual form for computational applications
- It's what Matlab gives you



Approximation error

- How good (bad) is this approximation?
- It's the best possible, measured by the Frobenius norm of the error:

$$\min_{X: \text{rank}(X)=k} \|A - X\|_F = \|A - A_k\|_F = \sigma_{k+1}$$

where the σ_i are ordered such that $\sigma_i \geq \sigma_{i+1}$.
Suggests why Frobenius error drops as k increased.

SVD Low-rank approximation

- Whereas the term-doc matrix A may have $M=50000, N=10$ million (and rank close to 50000)
- We can construct an approximation A_{100} with rank 100.
 - Of all rank 100 matrices, it would have the lowest Frobenius error.
- Great ... but why would we??
- Answer: *Latent Semantic Indexing*

Latent Semantic Indexing via the SVD

What it is

- From term-doc matrix A , we compute the approximation A_k .
- There is a row for each term and a column for each doc in A_k
- Thus docs live in a space of $k \ll r$ dimensions
 - These dimensions are not the original axes
- But why?

Vector Space Model: Pros

- **Automatic** selection of index terms
- **Partial matching** of queries and documents (dealing with the case where no document contains all search terms)
- **Ranking** according to **similarity score** (dealing with large result sets)
- **Term weighting** schemes (improves retrieval performance)
- Various extensions
 - Document clustering
 - Relevance feedback (modifying query vector)
- Geometric foundation

Problems with Lexical Semantics

- Ambiguity and association in natural language
 - **Polysemy**: Words often have a **multitude of meanings** and different types of usage (more severe in very heterogeneous collections).
- The vector space model is unable to discriminate between different meanings of the same word.

$$\text{sim}_{\text{true}}(d, q) < \cos(\angle(\vec{d}, \vec{q}))$$

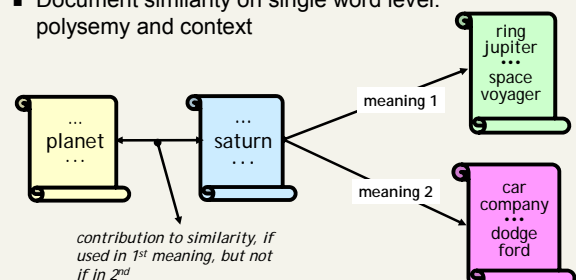
Problems with Lexical Semantics

- **Synonymy**: Different terms may have an **identical or a similar meaning** (weaker: words indicating the same topic).
- No associations between words are made in the vector space representation.

$$\text{sim}_{\text{true}}(d, q) > \cos(\angle(\vec{d}, \vec{q}))$$

Polysemy and Context

- Document similarity on single word level: polysemy and context



Latent Semantic Indexing (LSI)

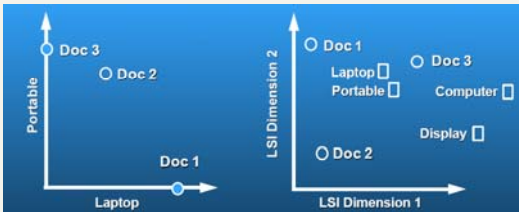
- Perform a **low-rank approximation** of **document-term matrix** (typical rank **100-300**)
- General idea
 - Map documents (*and terms*) to a **low-dimensional** representation.
 - Design a mapping such that the low-dimensional space reflects **semantic associations** (latent semantic space).
 - Compute document similarity based on the **inner product** in this **latent semantic space**

Goals of LSI

- Similar terms map to similar location in low dimensional space
- Noise reduction by dimension reduction

Latent Semantic Analysis

- **Latent semantic space**: illustrating example



courtesy of Susan Dumais

Performing the maps

- Each row and column of A gets mapped into the k -dimensional LSI space, by the SVD.
- Claim – this is not only the mapping with the best (Frobenius error) approximation to A , but in fact *improves* retrieval.
- A query q is also mapped into this space, by

$$q_k = q^T U_k \Sigma_k^{-1}$$
 - Query NOT a sparse vector.

Empirical evidence

- Experiments on TREC 1/2/3 – Dumais
- Lanczos SVD code (available on netlib) due to Berry used in these expts
 - Running times of ~ one day on tens of thousands of docs **[still an obstacle to use]**
- Dimensions – various values 250-350 reported. Reducing k improves recall.
 - (Under 200 reported unsatisfactory)
- Generally expect recall to improve – what about precision?

Empirical evidence

- Precision at or above median TREC precision
 - Top scorer on almost 20% of TREC topics
- Slightly better on average than straight vector spaces
- Effect of dimensionality:

Dimensions	Precision
250	0.367
300	0.371
346	0.374

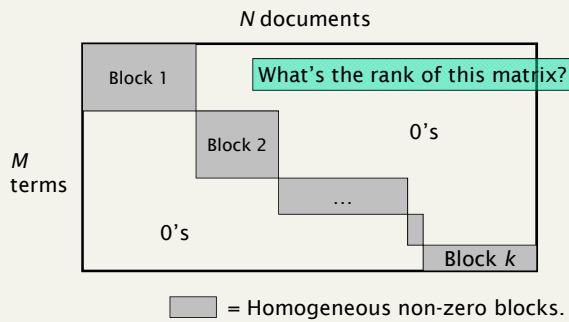
Failure modes

- Negated phrases
 - TREC topics sometimes negate certain query/terms phrases – automatic conversion of topics to
- Boolean queries
 - As usual, freetext/vector space syntax of LSI queries precludes (say) “Find any doc having to do with the following 5 companies”
- See Dumais for more.

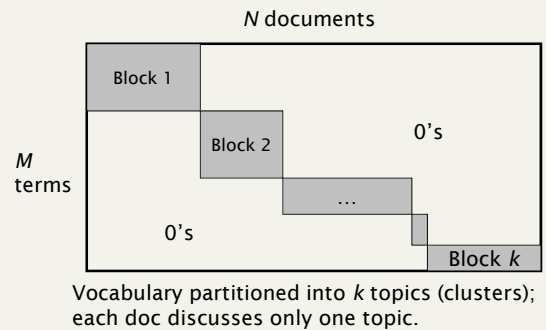
But why is this clustering?

- We’ve talked about docs, queries, retrieval and precision here.
- What does this have to do with clustering?
- Intuition: Dimension reduction through LSI brings together “related” axes in the vector space.

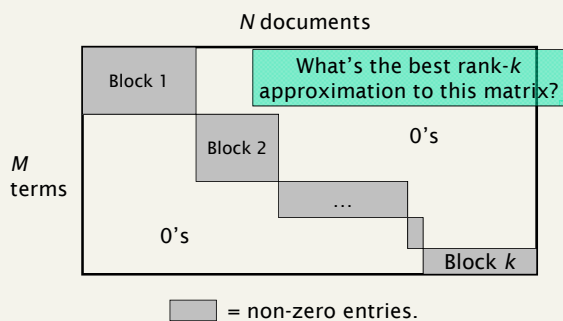
Intuition from block matrices



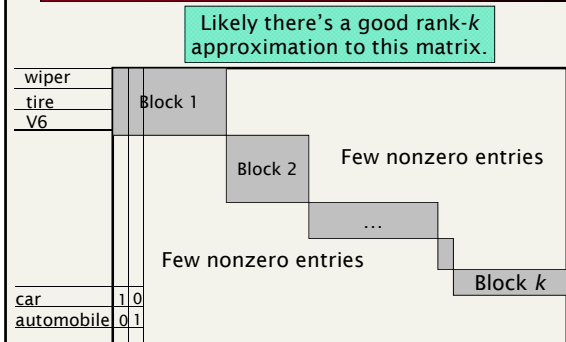
Intuition from block matrices



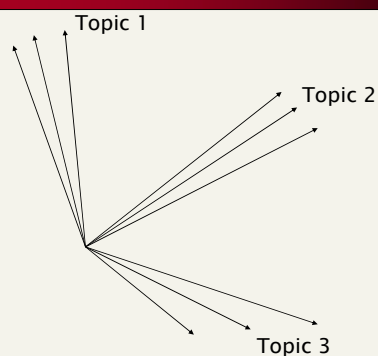
Intuition from block matrices



Intuition from block matrices



Simplistic picture



Some wild extrapolation

- The “dimensionality” of a corpus is the number of distinct topics represented in it.
- More mathematical wild extrapolation:
 - if A has a rank k approximation of low Frobenius error, then there are no more than k distinct topics in the corpus.

LSI has many other applications

- In many settings in pattern recognition and retrieval, we have a feature-object matrix.
 - For text, the terms are features and the docs are objects.
 - Could be opinions and users ...
 - This matrix may be redundant in dimensionality.
 - Can work with low-rank approximation.
 - If entries are missing (e.g., users' opinions), can recover if dimensionality is low.
- Powerful general analytical technique
 - Close, principled analog to clustering methods.

Resources

- IIR 18