Big Data Analysis
(MA60306)

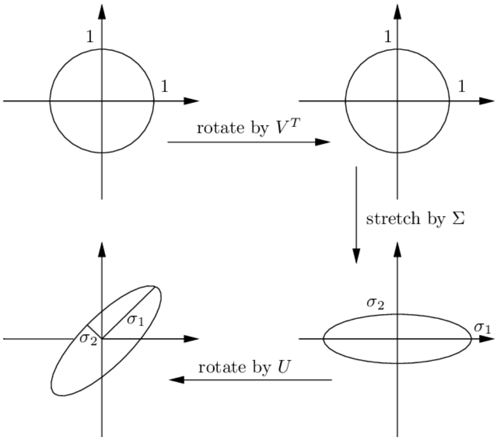
Bibhas Adhikari

Spring 2022-23, IIT Kharagpur

Lecture 10
February 1, 2023

Singular value decomposition

SVD as transformation



Eigenvalue decomposition

$\rightarrow$ Eigenvalue decomposition is defined only for square matrices

Eigenvalue decomposition

$\rightarrow$ Eigenvalue decomposition is defined only for square matrices

$\rightarrow$ Eigenvector and eigenvalue: $Mv = \lambda\, v$, $\|v\| = 1$

Eigenvalue decomposition

- → Eigenvalue decomposition is defined only for square matrices
- → Eigenvector and eigenvalue: $Mv = \lambda\, v$, $\|v\| = 1$
- → To avoid complex eigenvalues, we focus only computing eigenvalues of positive semi-definite matrices.

Eigenvalue decomposition

→ Eigenvalue decomposition is defined only for square matrices

→ Eigenvector and eigenvalue: $Mv = \lambda\, v$, $\|v\| = 1$

→ To avoid complex eigenvalues, we focus only computing eigenvalues of positive semi-definite matrices.

Let $A \in \mathbb{R}^{n\times d}$ be a data matrix. Then

$$M_R = A^T A \in \mathbb{R}^{d\times d} \text{ and } M_L = AA^T \in \mathbb{R}^{n\times n}$$

are positive semi-definite matrices

Eigenvalue decomposition

→ Eigenvalue decomposition is defined only for square matrices

→ Eigenvector and eigenvalue: $Mv = \lambda\, v$, $\|v\| = 1$

→ To avoid complex eigenvalues, we focus only computing eigenvalues of positive semi-definite matrices.
Let $A \in \mathbb{R}^{n\times d}$ be a data matrix. Then

$$M_R = A^T A \in \mathbb{R}^{d\times d} \text{ and } M_L = AA^T \in \mathbb{R}^{n\times n}$$

are positive semi-definite matrices

→ if SVD of A is $A = U\Sigma V^T$ then

$$M_R V = A^T AV = (V\Sigma U^T)(U\Sigma V^T)V = V\Sigma^2$$

Eigenvalue decomposition

→ Eigenvalue decomposition is defined only for square matrices

→ Eigenvector and eigenvalue: $Mv = \lambda\, v$, $\|v\| = 1$

→ To avoid complex eigenvalues, we focus only computing eigenvalues of positive semi-definite matrices.

Let $A \in \mathbb{R}^{n\times d}$ be a data matrix. Then

$$M_R = A^T A \in \mathbb{R}^{d\times d} \text{ and } M_L = AA^T \in \mathbb{R}^{n\times n}$$

are positive semi-definite matrices

→ if SVD of A is $A = U\Sigma V^T$ then

$$M_R V = A^T A V = (V\Sigma U^T)(U\Sigma V^T)V = V\Sigma^2$$

→ Then

$$M_R v_j = v_j \sigma_i^2$$

i.e. the $j$the right singular vector of A is the jth eigenvector of $M_R = A^T A$ corresponding to the eigenvalue $\lambda_j = \sigma_j^2$

Eigenvalue decomposition

→ Similarly,

$$M_L U = AA^T U = (U\Sigma V^T)(V\Sigma U^T)U = U\Sigma^2$$

i.e. the left singular vectors of A are eigenvectors of $M_L = AA^T$ and the eigenvalues are σ_j^2

Eigenvalue decomposition

$\rightarrow$ Similarly,

$$M_L U = AA^T U = (U\Sigma V^T)(V\Sigma U^T)U = U\Sigma^2$$

i.e. the left singular vectors of A are eigenvectors of $M_L = AA^T$ and the eigenvalues are σ_j^2

Power method - a simple iterative algorithm for computing the first eigenvector and eigenvalue

Eigenvalue decomposition

$\rightarrow$ Similarly,

$$M_L U = AA^T U = (U\Sigma V^T)(V\Sigma U^T)U = U\Sigma^2$$

i.e. the left singular vectors of A are eigenvectors of $M_L = AA^T$ and the eigenvalues are σ_j^2

Power method - a simple iterative algorithm for computing the first eigenvector and eigenvalue

PowerMethod($M = A^T A, q$)
Initialize $u^{(0)}$ as random unit vector
for $i = 1$ to q do
  $u^{(i)} := Mu^{(i-1)}$
return $v = u^{(q)}/\|u^{(q)}\|$

Eigenvalue decomposition

→ The output of PowerMethod($M = A^T A, q$) is a unit vector v which is close to the first eigenvector v_1

Eigenvalue decomposition

→ The output of PowerMethod($M = A^T A, q$) is a unit vector v which is close to the first eigenvector v_1

→ recover $\lambda_1 = \|Mv_1\|$

Eigenvalue decomposition

- $\rightarrow$ The output of PowerMethod($M = A^T A, q$) is a unit vector v which is close to the first eigenvector v_1
- $\rightarrow$ recover $\lambda_1 = \|Mv_1\|$
- $\rightarrow$ Define (factor out v_1)

$$A_1 := A - Av_1 v_1^T \text{ (deflation)}, \; M_1 := A_1 A_1^T$$

Then run PowerMethod($M_1 = A_1^T A_1, q$) to recover v_2 and λ_2

Eigenvalue decomposition

→ The output of PowerMethod($M = A^T A, q$) is a unit vector v which is close to the first eigenvector v_1

→ recover $\lambda_1 = \|Mv_1\|$

→ Define (factor out v_1)

$$A_1 := A - Av_1 v_1^T \text{ (deflation)}, \;\; M_1 := A_1 A_1^T$$

Then run PowerMethod($M_1 = A_1^T A_1, q$) to recover v_2 and λ_2

→ then continue the method iteravtively

Eigenvalue decomposition

→ The output of PowerMethod($M = A^T A, q$) is a unit vector v which is close to the first eigenvector v_1

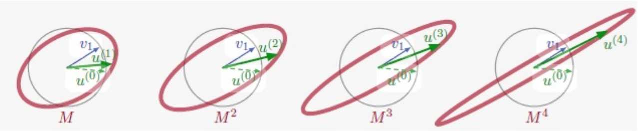
→ recover $\lambda_1 = \|Mv_1\|$

→ Define (factor out v_1)

$$A_1 := A - Av_1 v_1^T \text{ (deflation)}, \;\; M_1 := A_1 A_1^T$$

Then run PowerMethod($M_1 = A_1^T A_1, q$) to recover v_2 and λ_2

→ then continue the method iteravtively



Eigenvalue decomposition

Question How does the Power Method work!

Eigenvalue decomposition

Question How does the Power Method work!

Note Eigenvectors $v_1, \ldots, v_d$ of M forms an orthogonal basis of $\mathbb{R}^d$

Eigenvalue decomposition

Question How does the Power Method work!

Note Eigenvectors $v_1, \ldots, v_d$ of M forms an orthogonal basis of $\mathbb{R}^d$

Then

$$u^{(0)} = \sum_{j=1}^{d} \alpha_j v_j$$

Eigenvalue decomposition

Question How does the Power Method work!

Note Eigenvectors $v_1, \ldots, v_d$ of M forms an orthogonal basis of $\mathbb{R}^d$

Then

$$u^{(0)} = \sum_{j=1}^{d} \alpha_j v_j$$

Recall: $\alpha_j = v_j^T u^{(0)}$, since it is random, it is possible to claim that with probability at least $1/2$ that for any α_j we have $|\alpha_j| \geq \frac{1}{2\sqrt{n}}$

Eigenvalue decomposition

Question How does the Power Method work!

Note Eigenvectors $v_1, \ldots, v_d$ of M forms an orthogonal basis of $\mathbb{R}^d$

Then

$$u^{(0)} = \sum_{j=1}^{d} \alpha_j v_j$$

Recall: $\alpha_j = v_j^T u^{(0)}$, since it is random, it is possible to claim that with probability at least $1/2$ that for any α_j we have $|\alpha_j| \geq \frac{1}{2\sqrt{n}}$

Now

$$M^q v_j = M^{q-1}(\lambda_j v_j) = M^{q-2}(v_j \lambda_j)\lambda_j = \ldots = v_j \lambda_j^q$$

Eigenvalue decomposition

Then

$$v = M^q u^{(0)} = \sum_{j=1}^{d} \frac{\alpha_j \lambda_j^q}{\sqrt{\sum_{j=1}^{d} (\alpha_j \lambda_j^q)^2}} v_j$$

Eigenvalue decomposition

Then

$$v = M^q u^{(0)} = \sum_{j=1}^{d} \frac{\alpha_j \lambda_j^q}{\sqrt{\sum_{j=1}^{d} (\alpha_j \lambda_j^q)^2}} v_j$$

Further

$$\begin{aligned}
|v_1^T M^q u^{(0)}| &= \frac{\alpha_1 \lambda_1^q}{\sqrt{\sum_{j=1}^{d} (\alpha_j \lambda_j^q)^2}} \\
&\geq \frac{\alpha_1 \lambda_1^q}{\sqrt{\alpha_1^2 \lambda_1^{2q} + d\lambda_2^{2q}}} \\
&\geq \frac{\alpha_1 \lambda_1^q}{\alpha_1 \lambda_1^q + \sqrt{d}\lambda_2^q} = 1 - \frac{\lambda_2^q \sqrt{d}}{\alpha_1 \lambda_1^q + \sqrt{d}\lambda_2^q} \\
&\geq 1 - 2d \left(\frac{\lambda_2}{\lambda_1} \right)^q
\end{aligned}$$

Eigenvalue decomposition

Observation If the 'gap' $\frac{\lambda_1}{\lambda_2}$ between two eigenvalues is large then the algorithm converges faster

Eigenvalue decomposition

Observation If the 'gap' $\dfrac{\lambda_1}{\lambda_2}$ between two eigenvalues is large then the algorithm converges faster

Question Is the Power method backward stable?

Eigenvalue decomposition

Observation If the 'gap' $\dfrac{\lambda_1}{\lambda_2}$ between two eigenvalues is large then the algorithm converges faster

Question Is the Power method backward stable?

Recall

→ Given a data point $\boldsymbol{a}_j \in \mathbb{R}^d$ and a vector $\boldsymbol{v} \in \mathbb{R}^d$, the projection of $\boldsymbol{a}_j$ onto $\boldsymbol{v}$ is

$$\pi_{\boldsymbol{v}}(\boldsymbol{a}_j) = \langle \boldsymbol{v}, \boldsymbol{a}_j \rangle \boldsymbol{v} = (\boldsymbol{a}_j^T \boldsymbol{v})\boldsymbol{v}$$

Eigenvalue decomposition

Observation If the 'gap' $\dfrac{\lambda_1}{\lambda_2}$ between two eigenvalues is large then the algorithm converges faster
Question Is the Power method backward stable?
Recall

- → Given a data point $\boldsymbol{a}_j \in \mathbb{R}^d$ and a vector $\boldsymbol{v} \in \mathbb{R}^d$, the projection of $\boldsymbol{a}_j$ onto $\boldsymbol{v}$ is

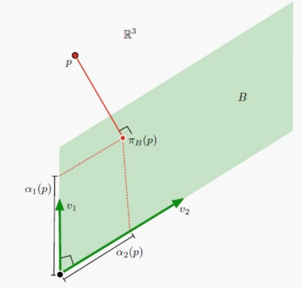
$$\pi_{\boldsymbol{v}}(\boldsymbol{a}_j) = \langle \boldsymbol{v}, \boldsymbol{a}_j \rangle \boldsymbol{v} = (\boldsymbol{a}_j^T \boldsymbol{v})\boldsymbol{v}$$

- → Example: for a data point $\boldsymbol{p} \in \mathbb{R}^3$ and a a two dimensional subspace V with basis $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$,

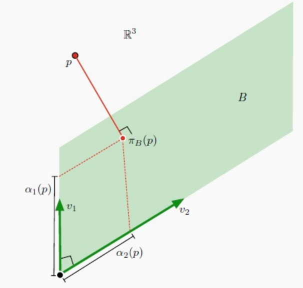
$$\pi_V(\boldsymbol{p}) = \alpha_1(\boldsymbol{p})\boldsymbol{v}_1 + \alpha_2(\boldsymbol{p})\boldsymbol{v}_2$$

where $\alpha_1(\boldsymbol{p}) = \boldsymbol{p}^T \boldsymbol{v}_1$ and $\alpha_2(\boldsymbol{p}) = \boldsymbol{p}^T \boldsymbol{v}_2$

Principal component analysis



Principal component analysis



→ The goal is to find the best fitting subspace V^* such that the sum of squared errors (SSE)

$$SSE(A, V) = \sum_{\mathbf{a}_j \in A} \|\mathbf{a}_j - \pi_V(\mathbf{a}_j)\|^2$$

is minimized and the desired k-dimensional subspace is

$$V^* = \arg\min_V SSE(A, V)$$

Principal component analysis

$\rightarrow$ Using SVD: Let A has rank r and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r$ are right singular vectors of A with singular values $\sigma_j^2 = \|A\boldsymbol{v}_j\|_2^2$. Then setting $B = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$, $k \leq r$, for any data point $\boldsymbol{a}$,

$$\|\boldsymbol{a} - \pi_B(\boldsymbol{a})\|^2 = \left\| \sum_{j=1}^{d} (\boldsymbol{a}^T \boldsymbol{v}_j) v_j - \sum_{j=1}^{k} (\boldsymbol{a}^T \boldsymbol{v}_j) v_j \right\|^2 = \sum_{j=k+1}^{d} (\boldsymbol{a}^T \boldsymbol{v}_j)^2$$

So the projection error of the subspace V_B, spanned by B is that part of A in the last $(d-k)$ right singular vectors.

Principal component analysis

→ Projection error for the given data points:

$$\sum_{i=1}^{n} \|\boldsymbol{a}_i - \pi_B(\boldsymbol{a}_i)\|^2 = \sum_{i=1}^{n} \left(\sum_{j=k+1}^{d} (\boldsymbol{a}_i^T \boldsymbol{v}_j)^2 \right)$$

Principal component analysis

$\rightarrow$ Projection error for the given data points:

$$\sum_{i=1}^{n} \|\boldsymbol{a}_i - \pi_B(\boldsymbol{a}_i)\|^2 = \sum_{i=1}^{n} \left(\sum_{j=k+1}^{d} (\boldsymbol{a}_i^T \boldsymbol{v}_j)^2 \right)$$

$$= \sum_{j=k+1}^{d} \left(\sum_{i=1}^{n} (\boldsymbol{a}_i^T \boldsymbol{v}_j)^2 \right) = \sum_{k+1}^{d} \|A\boldsymbol{v}_j\|^2 = \sum_{j=k+1}^{d} \sigma_j^2$$

$$= SSE(A, B) = \sum_{\boldsymbol{a}_i \in A} \|\boldsymbol{a}_i - \pi_B(\boldsymbol{a}_i)\|^2 = \|A - \pi_B(A)\|_F^2$$