Text as Data

Justin Grimmer

Associate Professor
Department of Political Science
Stanford University

October 14th, 2014

Finding Lower Dimensional Embeddings of Text

1) Task:

Finding Lower Dimensional Embeddings of Text

1) Task:
    - Embed our documents in a lower dimensional space

Finding Lower Dimensional Embeddings of Text

1) Task:
    - Embed our documents in a lower dimensional space
    - Visualize our documents

Finding Lower Dimensional Embeddings of Text

1) Task:
    - Embed our documents in a lower dimensional space
    - Visualize our documents
    - Inference about similarity

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points

Finding Lower Dimensional Embeddings of Text

1) Task:
    - Embed our documents in a lower dimensional space
    - Visualize our documents
    - Inference about similarity
    - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
    - Reconstruction error: measure the distance between the embedded points and the original points $\rightsquigarrow$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points $\rightsquigarrow$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances $\rightsquigarrow$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points $\rightsquigarrow$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances $\rightsquigarrow$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates
3) Optimization

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points $\leadsto$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances $\leadsto$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates
3) Optimization $\leadsto$ eigenvectors, eigenvalues

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points $\rightsquigarrow$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances $\rightsquigarrow$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates
3) Optimization $\rightsquigarrow$ eigenvectors, eigenvalues
   - Reconstruction error $\rightsquigarrow$ Search over components and weights

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points$\leadsto$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances$\leadsto$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates
3) Optimization$\leadsto$ eigenvectors, eigenvalues
   - Reconstruction error$\leadsto$ Search over components and weights
   - Distance preservation$\leadsto$ Search over locations

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points $\rightsquigarrow$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances $\rightsquigarrow$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates
3) Optimization $\rightsquigarrow$ eigenvectors, eigenvalues
   - Reconstruction error $\rightsquigarrow$ Search over components and weights
   - Distance preservation $\rightsquigarrow$ Search over locations
4) Validation

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points $\leadsto$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances $\leadsto$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates
3) Optimization $\leadsto$ eigenvectors, eigenvalues
   - Reconstruction error $\leadsto$ Search over components and weights
   - Distance preservation $\leadsto$ Search over locations
4) Validation
   a) Labeling exercise: What are the dimensions?

Finding Lower Dimensional Embeddings of Text

1) Task:
   - Embed our documents in a lower dimensional space
   - Visualize our documents
   - Inference about similarity
   - Social science inference about behavior
2) Objective Function, $f(\boldsymbol{X}, \boldsymbol{\theta})$
   - Reconstruction error: measure the distance between the embedded points and the original points $\rightsquigarrow$ $\boldsymbol{\theta}$ will contain basis vectors (principal components) and weights to those vectors
   - Distance preservation: measure how well pairwise distances are preserved with distances $\rightsquigarrow$ $\boldsymbol{\theta}$ a set of new lower-dimensional coordinates
3) Optimization $\rightsquigarrow$ eigenvectors, eigenvalues
   - Reconstruction error $\rightsquigarrow$ Search over components and weights
   - Distance preservation $\rightsquigarrow$ Search over locations
4) Validation
   a) Labeling exercise: What are the dimensions?
   b) Are the dimensions interesting semantically?

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose $\boldsymbol{A}$ *is an* $N \times N$ *matrix and* λ *is a scalar.*
If

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$$

Then $\boldsymbol{x}$ *is an eigenvector and* λ *is the associated eigenvalue*

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose $\boldsymbol{A}$ *is an* $N \times N$ *matrix and* λ *is a scalar.*
If

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$$

Then $\boldsymbol{x}$ *is an eigenvector and* λ *is the associated eigenvalue*

- $\boldsymbol{A}$ stretches the eigenvector $\boldsymbol{x}$

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose $\boldsymbol{A}$ *is an* $N \times N$ *matrix and* λ *is a scalar.*
If

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$$

Then $\boldsymbol{x}$ *is an eigenvector and* λ *is the associated eigenvalue*

- $\boldsymbol{A}$ stretches the eigenvector $\boldsymbol{x}$
- $\boldsymbol{A}$ stretches $\boldsymbol{x}$ by λ

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose $\boldsymbol{A}$ *is an* $N \times N$ *matrix and* λ *is a scalar.*
If

$$\boldsymbol{Ax} = \lambda \boldsymbol{x}$$

Then $\boldsymbol{x}$ *is an eigenvector and* λ *is the associated eigenvalue*

- $\boldsymbol{A}$ stretches the eigenvector $\boldsymbol{x}$
- $\boldsymbol{A}$ stretches $\boldsymbol{x}$ by λ
- To find eigenvectors/values: (`eigen` in `R`)

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose $\boldsymbol{A}$ *is an* $N \times N$ *matrix and* λ *is a scalar.*
If

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$$

Then $\boldsymbol{x}$ *is an eigenvector and* λ *is the associated eigenvalue*

- $\boldsymbol{A}$ stretches the eigenvector $\boldsymbol{x}$
- $\boldsymbol{A}$ stretches $\boldsymbol{x}$ by λ
- To find eigenvectors/values: (`eigen` in `R`)
  - Find λ that solves $\det(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0$

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose $\boldsymbol{A}$ is an $N \times N$ matrix and λ is a scalar.
If

$$\boldsymbol{A}\boldsymbol{x} \quad = \quad \lambda \boldsymbol{x}$$

Then $\boldsymbol{x}$ is an eigenvector and λ is the associated eigenvalue

- $\boldsymbol{A}$ stretches the eigenvector $\boldsymbol{x}$
- $\boldsymbol{A}$ stretches $\boldsymbol{x}$ by λ
- To find eigenvectors/values: (`eigen` in `R`)
  - Find λ that solves $\det(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0$
  - Find vectors in null space of:

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose $\boldsymbol{A}$ *is an* $N \times N$ *matrix and* λ *is a scalar.*
If

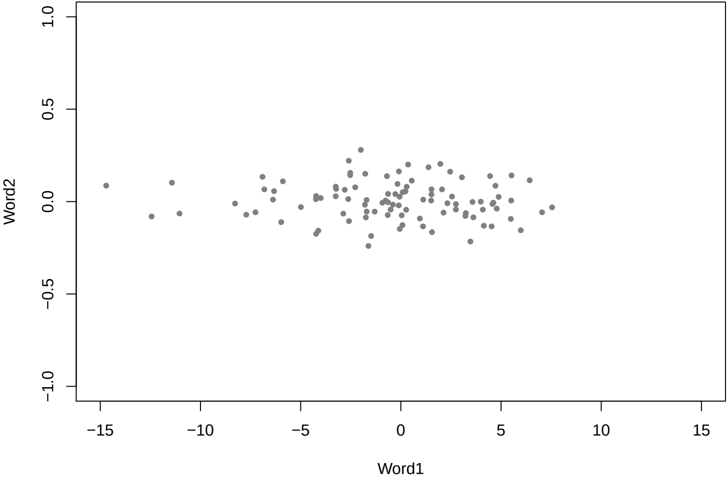
$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$$

Then $\boldsymbol{x}$ *is an eigenvector and* λ *is the associated eigenvalue*

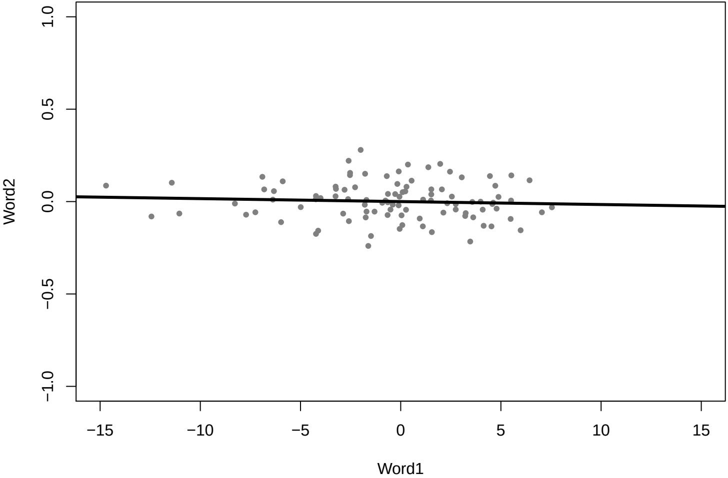
- $\boldsymbol{A}$ stretches the eigenvector $\boldsymbol{x}$
- $\boldsymbol{A}$ stretches $\boldsymbol{x}$ by λ
- To find eigenvectors/values: (`eigen` in `R`)
  - Find λ that solves $\det(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0$
  - Find vectors in null space of:

$$(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0$$

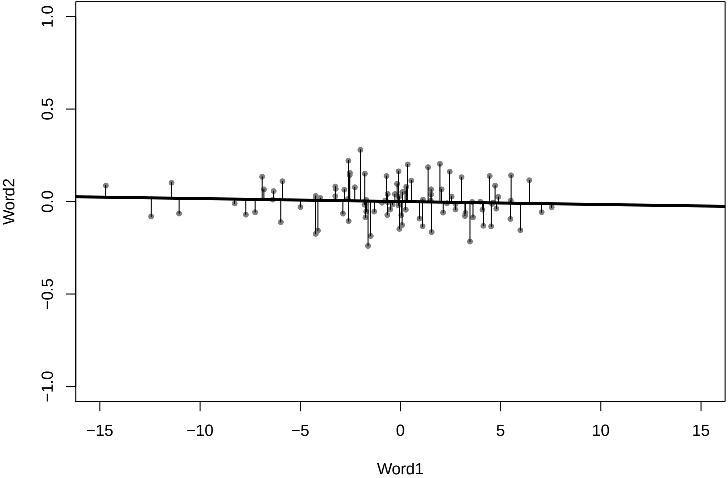
Finding a Lower Dimensional Space (Manifold Learning)



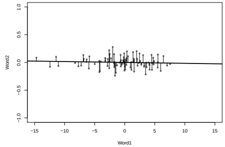
Finding a Lower Dimensional Space (Manifold Learning)



Finding a Lower Dimensional Space (Manifold Learning)

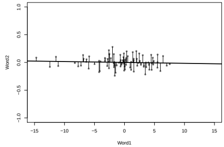


Finding a Lower Dimensional Space (Manifold Learning)



Original data:

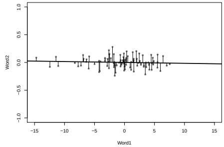
Finding a Lower Dimensional Space (Manifold Learning)



Original data:

$$\boldsymbol{x}_i = (x_{i1}, x_{i2})$$

Finding a Lower Dimensional Space (Manifold Learning)

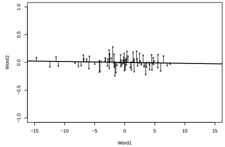


Original data:

$$\mathbf{x}_i = (x_{i1}, x_{i2})$$

Which we approximate with

Finding a Lower Dimensional Space (Manifold Learning)



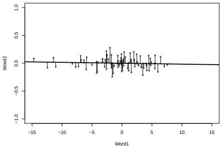
Original data:

$$\boldsymbol{x}_i = (x_{i1}, x_{i2})$$

Which we approximate with

$$\begin{aligned}\tilde{\boldsymbol{x}}_i &= z_i \boldsymbol{w}_1 \\ &= z_i(w_{11}, w_{12})\end{aligned}$$

Finding a Lower Dimensional Space (Manifold Learning)



Original data $\boldsymbol{x}_i \in \Re^J$

$$\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iJ})$$

Which we approximate with $L(< J)$ weights z_{il} and vectors $\boldsymbol{w}_l \in \Re^J$

$$\tilde{\boldsymbol{x}}_i = z_{i1}\boldsymbol{w}_1 + z_{i2}\boldsymbol{w}_2 + \ldots + z_{iL}\boldsymbol{w}_L$$

Define $\boldsymbol{\theta} = (\underbrace{\boldsymbol{Z}}_{N\times L}, \underbrace{\boldsymbol{W}_L}_{L\times J})$

Principal Component Analysis $\rightsquigarrow$ Objective function

Consider 1-dimensional case ($L = 1$), centered data, and $||\boldsymbol{w}_1|| = 1$.

Principal Component Analysis$\rightsquigarrow$ Objective function

Consider 1-dimensional case ($L = 1$), centered data, and $||\boldsymbol{w}_1|| = 1$.

$$f(\boldsymbol{\theta}, \boldsymbol{X}) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1||^2$$

Principal Component Analysis⇝ Objective function

Consider 1-dimensional case ($L = 1$), centered data, and $||\boldsymbol{w}_1|| = 1$.

$$
\begin{aligned}
f(\boldsymbol{\theta}, \boldsymbol{X}) &= \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1||^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1)
\end{aligned}
$$

Principal Component Analysis $\rightsquigarrow$ Objective function

Consider 1-dimensional case ($L = 1$), centered data, and $||\boldsymbol{w}_1|| = 1$.

$$
\begin{aligned}
f(\theta, \boldsymbol{X}) &= \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1||^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1) \\
&= \frac{1}{N}\sum_{i=1}^{N} \left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - 2z_{i1}\boldsymbol{w}_1^{'}\boldsymbol{x}_i + z_{i1}^2\right)
\end{aligned}
$$

Principal Component Analysis⇝ Objective function

Consider 1-dimensional case ($L = 1$), centered data, and $||\boldsymbol{w}_1|| = 1$.

$$
\begin{aligned}
f(\boldsymbol{\theta}, \boldsymbol{X}) &= \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1||^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}\boldsymbol{w}_1) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - 2z_{i1}\boldsymbol{w}_1^{'}\boldsymbol{x}_i + z_{i1}^2\right)
\end{aligned}
$$

$\boldsymbol{w}_1^{'}\boldsymbol{w}_1 = 1$

Principal Component Analysis$\rightsquigarrow$ Optimization

Optimization:

Principal Component Analysis⇝ Optimization

Optimization:

$$\frac{\partial f(\boldsymbol{\theta}, \boldsymbol{X})}{\partial z_{i1}} = -\frac{2\boldsymbol{w}_1^{'}\boldsymbol{x}_i + 2z_{i1}}{N}$$

Principal Component Analysis $\rightsquigarrow$ Optimization

Optimization:

$$
\begin{aligned}
\frac{\partial f(\boldsymbol{\theta}, \boldsymbol{X})}{\partial z_{i1}} &= -\frac{2\boldsymbol{w}_1^{'}\boldsymbol{x}_i + 2z_{i1}}{N} \\
0 &= -\frac{2\boldsymbol{w}_1^{'}\boldsymbol{x}_i + 2z_{i1}^{*}}{N}
\end{aligned}
$$

Principal Component Analysis $\rightsquigarrow$ Optimization

Optimization:

$$
\begin{aligned}
\frac{\partial f(\boldsymbol{\theta}, \boldsymbol{X})}{\partial z_{i1}} &= -\frac{2\boldsymbol{w}_1^{'}\boldsymbol{x}_i + 2z_{i1}}{N} \\
0 &= -\frac{2\boldsymbol{w}_1^{'}\boldsymbol{x}_i + 2z_{i1}^{*}}{N} \\
z_{i1}^{*} &= \boldsymbol{w}_1^{'}\boldsymbol{x}_i
\end{aligned}
$$

Principal Component Analysis $\rightsquigarrow$ Optimization

Substituting in z_{i1}^{*}

Principal Component Analysis$\rightsquigarrow$ Optimization

Substituting in z_{i1}^{*}

$$= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - z_{i1}^{*}\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}^{*}\boldsymbol{w}_1)$$

Principal Component Analysis⇝ Optimization

Substituting in z_{i1}^*

$$
\begin{aligned}
&= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1) \\
&= \frac{1}{N}\sum_{i=1}^{N}(\underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{Constant}} - 2z_{i1}^*\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{x}_i}_{z_{i1}^*} + (z_{i1}^*)^2\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{w}_1}_{1})
\end{aligned}
$$

Principal Component Analysis$\rightsquigarrow$ Optimization

Substituting in z_{i1}^*

$$
\begin{aligned}
&= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1) \\
&= \frac{1}{N}\sum_{i=1}^{N}(\underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{Constant}} - 2z_{i1}^*\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{x}_i}_{z_{i1}^*} + (z_{i1}^*)^2\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{w}_1}_{1}) \\
&= -\frac{1}{N}\sum_{i=1}^{N}(z_{i1}^*)^2 + c
\end{aligned}
$$

Principal Component Analysis⇝ Optimization

Substituting in z_{i1}^*

$$
\begin{aligned}
&= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1) \\
&= \frac{1}{N}\sum_{i=1}^{N}(\underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{Constant}} - 2z_{i1}^*\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{x}_i}_{z_{i1}^*} + (z_{i1}^*)^2\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{w}_1}_{1}) \\
&= -\frac{1}{N}\sum_{i=1}^{N}(z_{i1}^*)^2 + c \\
&= -\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{w}_1^{'}\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_1
\end{aligned}
$$

Principal Component Analysis$\leadsto$ Optimization

Substituting in z_{i1}^*

$$
\begin{aligned}
&= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1)^{'}(\boldsymbol{x}_i - z_{i1}^*\boldsymbol{w}_1) \\
&= \frac{1}{N}\sum_{i=1}^{N}(\underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{Constant}} - 2z_{i1}^*\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{x}_i}_{z_{i1}^*} + (z_{i1}^*)^2\underbrace{\boldsymbol{w}_1^{'}\boldsymbol{w}_1}_{1}) \\
&= -\frac{1}{N}\sum_{i=1}^{N}(z_{i1}^*)^2 + c \\
&= -\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{w}_1^{'}\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_1 \\
&= -\boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1
\end{aligned}
$$

Principal Component Analysis$\rightsquigarrow$ Optimization

$$= \; -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

Principal Component Analysis⇝ Optimization

$$= \; -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

Principal Component Analysis$\rightsquigarrow$ Optimization

$$= \quad -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix$\rightsquigarrow$ $\frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$

Principal Component Analysis $\rightsquigarrow$ Optimization

$$= \quad -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix $\rightsquigarrow \frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$
- Variance of the projected data. Define

Principal Component Analysis $\rightsquigarrow$ Optimization

$$= \quad -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix $\rightsquigarrow \frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$
- Variance of the projected data. Define

$$\boldsymbol{z}_1 \quad = \quad (\boldsymbol{w}_1\boldsymbol{x}_1, \boldsymbol{w}_1\boldsymbol{x}_2, \ldots, \boldsymbol{w}_1\boldsymbol{x}_N)$$

Principal Component Analysis $\rightsquigarrow$ Optimization

$$= -\sum_{i=1}^{N} \boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix $\rightsquigarrow \frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$
- Variance of the projected data. Define

$$\begin{aligned} \boldsymbol{z}_1 &= (\boldsymbol{w}_1\boldsymbol{x}_1, \boldsymbol{w}_1\boldsymbol{x}_2, \ldots, \boldsymbol{w}_1\boldsymbol{x}_N) \\ \text{var}(\boldsymbol{z}_1) &= E[\boldsymbol{z}_1^2] - E[\boldsymbol{z}_1]^2 \end{aligned}$$

Principal Component Analysis⇝ Optimization

$$= \quad -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix⇝ $\frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$
- Variance of the projected data. Define

$$\begin{aligned}
\boldsymbol{z}_1 &= (\boldsymbol{w}_1\boldsymbol{x}_1, \boldsymbol{w}_1\boldsymbol{x}_2, \ldots, \boldsymbol{w}_1\boldsymbol{x}_N) \\
\text{var}(\boldsymbol{z}_1) &= E[\boldsymbol{z}_1^2] - E[\boldsymbol{z}_1]^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} z_{i1}^2 - 0
\end{aligned}$$

Principal Component Analysis$\rightsquigarrow$ Optimization

$$= -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix$\rightsquigarrow$ $\frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$
- Variance of the projected data. Define

$$\begin{aligned}
\boldsymbol{z}_1 &= (\boldsymbol{w}_1\boldsymbol{x}_1, \boldsymbol{w}_1\boldsymbol{x}_2, \ldots, \boldsymbol{w}_1\boldsymbol{x}_N) \\
\text{var}(\boldsymbol{z}_1) &= E[\boldsymbol{z}_1^2] - E[\boldsymbol{z}_1]^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} z_{i1}^2 - 0 \\
&= \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{w}_1^{'}\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_1
\end{aligned}$$

Principal Component Analysis⇝ Optimization

$$= -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix⇝ $\frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$
- Variance of the projected data. Define

$$\begin{aligned}
\boldsymbol{z}_1 &= (\boldsymbol{w}_1\boldsymbol{x}_1, \boldsymbol{w}_1\boldsymbol{x}_2, \ldots, \boldsymbol{w}_1\boldsymbol{x}_N) \\
\text{var}(\boldsymbol{z}_1) &= E[\boldsymbol{z}_1^2] - E[\boldsymbol{z}_1]^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} z_{i1}^2 - 0 \\
&= \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{w}_1^{'}\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_1
\end{aligned}$$

Minimize reconstruction error

Principal Component Analysis$\leadsto$ Optimization

$$= \quad -\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1$$

where $\boldsymbol{\Sigma}$ is the :

- Empirical covariance matrix$\leadsto$ $\frac{1}{N}\boldsymbol{X}^{'}\boldsymbol{X}$
- Variance of the projected data. Define

$$\begin{aligned}
\boldsymbol{z}_1 &= (\boldsymbol{w}_1 \boldsymbol{x}_1, \boldsymbol{w}_1 \boldsymbol{x}_2, \ldots, \boldsymbol{w}_1 \boldsymbol{x}_N) \\
\text{var}(\boldsymbol{z}_1) &= E[\boldsymbol{z}_1^2] - E[\boldsymbol{z}_1]^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} z_{i1}^2 - 0 \\
&= \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{w}_1^{'} \boldsymbol{x}_i \boldsymbol{x}_i^{'} \boldsymbol{w}_1
\end{aligned}$$

Minimize reconstruction error $\leadsto$ maximize variance of projected data

Principal Component Analysis$\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

Principal Component Analysis $\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X}) = \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1 - \lambda_1(\boldsymbol{w}_1^{'} \boldsymbol{w}_1 - 1)$$

Principal Component Analysis $\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$\begin{aligned} g(\boldsymbol{z}^*, \boldsymbol{w}_1, \boldsymbol{X}) &= \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1 - \lambda_1(\boldsymbol{w}_1^{'} \boldsymbol{w}_1 - 1) \\ \frac{\partial g(\boldsymbol{z}^*, \boldsymbol{w}_1, \boldsymbol{X})}{\partial \boldsymbol{w}_1} &= 2\boldsymbol{\Sigma} \boldsymbol{w}_1 - 2\lambda_1 \boldsymbol{w}_1 \end{aligned}$$

Principal Component Analysis $\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$
\begin{aligned}
g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X}) &= \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1 - \lambda_1(\boldsymbol{w}_1^{'} \boldsymbol{w}_1 - 1) \\
\frac{\partial g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X})}{\partial \boldsymbol{w}_1} &= 2\boldsymbol{\Sigma} \boldsymbol{w}_1 - 2\lambda_1 \boldsymbol{w}_1 \\
\boldsymbol{\Sigma} \boldsymbol{w}_1^{*} &= \lambda_1 \boldsymbol{w}_1^{*}
\end{aligned}
$$

Principal Component Analysis $\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$\begin{aligned}
g(\boldsymbol{z}^*, \boldsymbol{w}_1, \boldsymbol{X}) &= \boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1 - \lambda_1(\boldsymbol{w}_1^{'}\boldsymbol{w}_1 - 1) \\
\frac{\partial g(\boldsymbol{z}^*, \boldsymbol{w}_1, \boldsymbol{X})}{\partial \boldsymbol{w}_1} &= 2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\lambda_1\boldsymbol{w}_1 \\
\boldsymbol{\Sigma}\boldsymbol{w}_1^* &= \lambda_1\boldsymbol{w}_1^*
\end{aligned}$$

$\boldsymbol{w}_1^*$ = Eigenvector of $\boldsymbol{\Sigma}$

Principal Component Analysis $\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$\begin{aligned} g(\boldsymbol{z}^*, \boldsymbol{w}_1, \boldsymbol{X}) &= \boldsymbol{w}_1^{'} \boldsymbol{\Sigma} \boldsymbol{w}_1 - \lambda_1 (\boldsymbol{w}_1^{'} \boldsymbol{w}_1 - 1) \\ \frac{\partial g(\boldsymbol{z}^*, \boldsymbol{w}_1, \boldsymbol{X})}{\partial \boldsymbol{w}_1} &= 2\boldsymbol{\Sigma} \boldsymbol{w}_1 - 2\lambda_1 \boldsymbol{w}_1 \\ \boldsymbol{\Sigma} \boldsymbol{w}_1^* &= \lambda_1 \boldsymbol{w}_1^* \end{aligned}$$

$\boldsymbol{w}_1^*$ = Eigenvector of $\boldsymbol{\Sigma}$ (!!!!!!)

Principal Component Analysis$\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$\begin{aligned} g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X}) &= \boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1 - \lambda_1(\boldsymbol{w}_1^{'}\boldsymbol{w}_1 - 1) \\ \frac{\partial g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X})}{\partial \boldsymbol{w}_1} &= 2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\lambda_1\boldsymbol{w}_1 \\ \boldsymbol{\Sigma}\boldsymbol{w}_1^{*} &= \lambda_1\boldsymbol{w}_1^{*} \end{aligned}$$

$\boldsymbol{w}_1^{*}$ = Eigenvector of $\boldsymbol{\Sigma}$ (!!!!!!)
We want $\boldsymbol{w}_1$ to maximize variance and

Principal Component Analysis $\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$
\begin{aligned}
g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X}) &= \boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1 - \lambda_1(\boldsymbol{w}_1^{'}\boldsymbol{w}_1 - 1) \\
\frac{\partial g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X})}{\partial \boldsymbol{w}_1} &= 2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\lambda_1\boldsymbol{w}_1 \\
\boldsymbol{\Sigma}\boldsymbol{w}_1^{*} &= \lambda_1\boldsymbol{w}_1^{*}
\end{aligned}
$$

$\boldsymbol{w}_1^{*}$ = Eigenvector of $\boldsymbol{\Sigma}$ (!!!!!!)
We want $\boldsymbol{w}_1$ to maximize variance and

$$
\boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1 = \lambda_1
$$

Principal Component Analysis$\rightsquigarrow$ Optimization

Maximize variance, subject to constraints

$$\begin{aligned} g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X}) &= \boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1 - \lambda_1(\boldsymbol{w}_1^{'}\boldsymbol{w}_1 - 1) \\ \frac{\partial g(\boldsymbol{z}^{*}, \boldsymbol{w}_1, \boldsymbol{X})}{\partial \boldsymbol{w}_1} &= 2\boldsymbol{\Sigma}\boldsymbol{w}_1 - 2\lambda_1\boldsymbol{w}_1 \\ \boldsymbol{\Sigma}\boldsymbol{w}_1^{*} &= \lambda_1\boldsymbol{w}_1^{*} \end{aligned}$$

$\boldsymbol{w}_1^{*}$ = Eigenvector of $\boldsymbol{\Sigma}$ (!!!!!!)
We want $\boldsymbol{w}_1$ to maximize variance and

$$\boldsymbol{w}_1^{'}\boldsymbol{\Sigma}\boldsymbol{w}_1 = \lambda_1$$

So $\boldsymbol{w}_1$ is eigenvector associated with the largest eigenvalue λ_1

An Introduction to Eigenvectors, Values, and Diagonalization

Theorem

Suppose $\boldsymbol{A}$ is an invertible $N \times N$ matrix. Then $\boldsymbol{A}$ has N distinct eigenvalues and N linearly independent eigenvectors. Further, we can write $\boldsymbol{A}$ as,

$$\boldsymbol{A} = \boldsymbol{W}^{'} \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_N \end{pmatrix} \boldsymbol{W}$$

where $\boldsymbol{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_N)$ is an $N \times N$ matrix with the N eigenvectors as column vectors.

An Introduction to Eigenvectors, Values, and Diagonalization

Definition

Suppose A is a covariance matrix. Then, we can write A as

$$\boldsymbol{A} \;=\; \boldsymbol{W}^{'} \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_N \end{pmatrix} \boldsymbol{W}$$

Where $\lambda_1 > \lambda_2 > \ldots > \lambda_N \geq 0$.
We will call $\boldsymbol{w}_1$ *the first eigenvector,* $\boldsymbol{w}_2$ *the second eigenvector, ...,* $\boldsymbol{w}_j$ *the* j^{th} *eigenvector.*

Back to Principal Components

Theorem

Suppose we want to approximate N observations $\boldsymbol{x}_i \in \Re^J$ with $L < J$ orthogonal-unit length vectors $\boldsymbol{w}_l \in \Re^J$ with associated scores z_{il} to minimize reconstruction error:

Back to Principal Components

Theorem

Suppose we want to approximate N observations $\boldsymbol{x}_i \in \Re^J$ with $L < J$ orthogonal-unit length vectors $\boldsymbol{w}_l \in \Re^J$ with associated scores z_{il} to minimize reconstruction error:

$$f(\boldsymbol{X}, \boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Back to Principal Components

Theorem

Suppose we want to approximate N observations $\boldsymbol{x}_i \in \Re^J$ with $L < J$ orthogonal-unit length vectors $\boldsymbol{w}_l \in \Re^J$ with associated scores z_{il} to minimize reconstruction error:

$$f(\boldsymbol{X}, \boldsymbol{\theta}) \;=\; \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

The optimal solution sets each $\boldsymbol{w}_l$ to be the l^{th} eigenvector of the empirical covariance matrix.

Back to Principal Components

Theorem

Suppose we want to approximate N observations $\boldsymbol{x}_i \in \Re^J$ with $L < J$ orthogonal-unit length vectors $\boldsymbol{w}_l \in \Re^J$ with associated scores z_{il} to minimize reconstruction error:

$$f(\boldsymbol{X}, \boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il} \boldsymbol{w}_l||^2$$

The optimal solution sets each $\boldsymbol{w}_l$ to be the l^{th} eigenvector of the empirical covariance matrix.Further $z_{il}^{*} = \boldsymbol{w}_l^{'} \boldsymbol{x}_i$ so that the L dimensional representation is:

Back to Principal Components

Theorem

Suppose we want to approximate N observations $\boldsymbol{x}_i \in \Re^J$ with $L < J$ orthogonal-unit length vectors $\boldsymbol{w}_l \in \Re^J$ with associated scores z_{il} to minimize reconstruction error:

$$f(\boldsymbol{X}, \boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il} \boldsymbol{w}_l||^2$$

The optimal solution sets each $\boldsymbol{w}_l$ to be the l^{th} eigenvector of the empirical covariance matrix.Further $z_{il}^* = \boldsymbol{w}_l^{'} \boldsymbol{x}_i$ so that the L dimensional representation is:

$$\boldsymbol{x}_i^L = (\boldsymbol{w}_1^{'} \boldsymbol{x}_i, \boldsymbol{w}_2^{'} \boldsymbol{x}_i, \ldots, \boldsymbol{w}_L^{'} \boldsymbol{x}_i)$$

Application of Principal Components in R

Consider press releases from 2005 US Senators

Application of Principal Components in R

Consider press releases from 2005 US Senators
Define $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iJ})$ as the rate senator i uses J words.

Application of Principal Components in R

Consider press releases from 2005 US Senators
Define $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iJ})$ as the rate senator i uses J words.

$$x_{ij} \quad = \quad \frac{\text{No. Times } i \text{ uses word } j}{\text{No. words } i \text{ uses}}$$

Application of Principal Components in R

Consider press releases from 2005 US Senators
Define $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iJ})$ as the rate senator i uses J words.

$$x_{ij} = \frac{\text{No. Times } i \text{ uses word } j}{\text{No. words } i \text{ uses}}$$

`dtm`: 100×2796 matrix containing word rates for senators

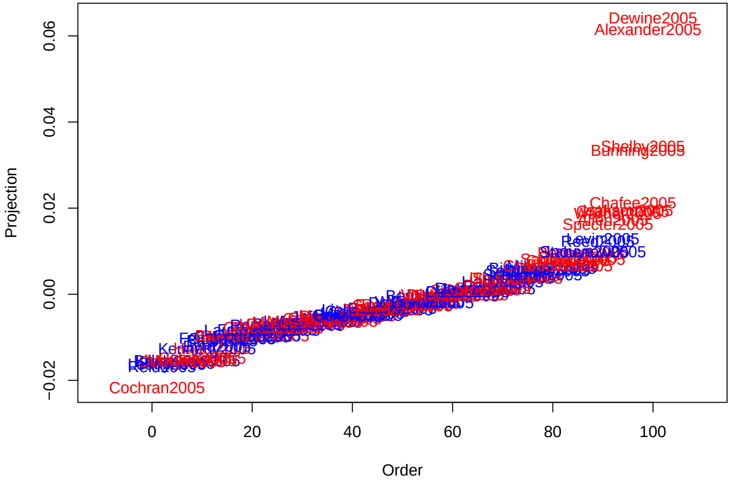
Application of Principal Components in R

Consider press releases from 2005 US Senators
Define $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iJ})$ as the rate senator i uses J words.

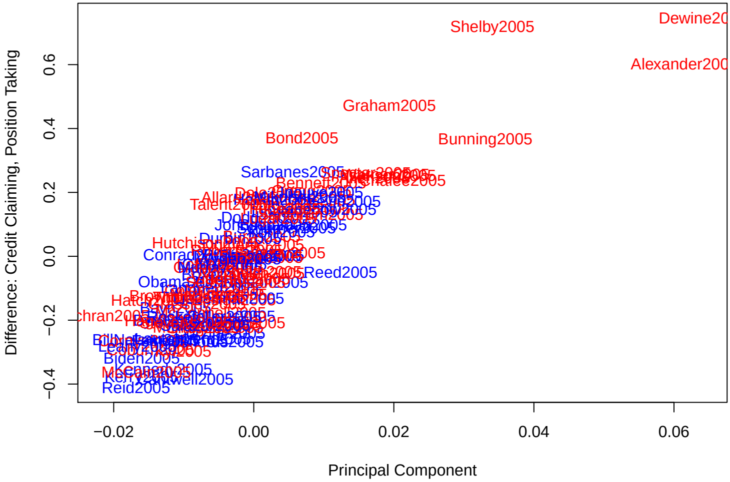
$$x_{ij} = \frac{\text{No. Times } i \text{ uses word } j}{\text{No. words } i \text{ uses}}$$

`dtm`: 100×2796 matrix containing word rates for senators
`prcomp(dtm)` applies principal components

Application of Principal Components in R



Application of Principal Components in R



How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)^{'}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)^{'}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)$$

Four types of terms:

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)^{'}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)$$

Four types of terms:

1) $\boldsymbol{x}_i^{'}\boldsymbol{x}_i$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)^{'}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)$$

Four types of terms:

1) $\boldsymbol{x}_i^{'}\boldsymbol{x}_i$
2) $z_{ij}z_{ik}\boldsymbol{w}_j^{'}\boldsymbol{w}_k = z_{ij}z_{ik}0 = 0$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)^{'}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)$$

Four types of terms:

1) $\boldsymbol{x}_i^{'}\boldsymbol{x}_i$
2) $z_{ij}z_{ik}\boldsymbol{w}_j^{'}\boldsymbol{w}_k = z_{ij}z_{ik}0 = 0$
3) $z_{ij}z_{ij}\boldsymbol{w}_j^{'}\boldsymbol{w}_j = z_{ij}^2$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)^{'}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)$$

Four types of terms:

1) $\boldsymbol{x}_i^{'}\boldsymbol{x}_i$
2) $z_{ij}z_{ik}\boldsymbol{w}_j^{'}\boldsymbol{w}_k = z_{ij}z_{ik}0 = 0$
3) $z_{ij}z_{ij}\boldsymbol{w}_j^{'}\boldsymbol{w}_j = z_{ij}^2$
4) $\boldsymbol{x}_i^{'}\sum_{l=1}^{L} z_{il}\boldsymbol{w}_l = \sum_{l=1}^{L} z_{il}^2$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

We want to minimize reconstruction error $\rightsquigarrow$ how well did we do?

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l||^2$$

Simplifying:

$$\begin{aligned}
\text{error}(L) &= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l)^{'}(\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}\boldsymbol{w}_l) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}^2\right)
\end{aligned}$$

Four types of terms:

1) $\boldsymbol{x}_i^{'}\boldsymbol{x}_i$
2) $z_{ij}z_{ik}\boldsymbol{w}_j^{'}\boldsymbol{w}_k = z_{ij}z_{ik}0 = 0$
3) $z_{ij}z_{ij}\boldsymbol{w}_j^{'}\boldsymbol{w}_j = z_{ij}^2$
4) $\boldsymbol{x}_i^{'}\sum_{l=1}^{L} z_{il}\boldsymbol{w}_l = \sum_{l=1}^{L} z_{il}^2$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$\text{error}(L) = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}^2\right)$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$
\begin{aligned}
\text{error}(L) &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}^2\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} \boldsymbol{w}_l \boldsymbol{x}_i \boldsymbol{x}_i^{'} \boldsymbol{w}_l\right)
\end{aligned}
$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$
\begin{aligned}
\text{error}(L) &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}^2\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} \boldsymbol{w}_l \boldsymbol{x}_i \boldsymbol{x}_i^{'} \boldsymbol{w}_l\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \frac{1}{N}\sum_{l=1}^{L}\sum_{l=1}^{N} \boldsymbol{w}_i^{'} \boldsymbol{x}_i \boldsymbol{x}_i^{'} \boldsymbol{w}_i
\end{aligned}
$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$
\begin{aligned}
\text{error}(L) &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}^2\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} \boldsymbol{w}_l\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_l\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \frac{1}{N}\sum_{l=1}^{L}\sum_{l=1}^{N} \boldsymbol{w}_i^{'}\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_i \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L} \boldsymbol{w}_l^{'}\boldsymbol{\Sigma}\boldsymbol{w}_l
\end{aligned}
$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$
\begin{aligned}
\text{error}(L) &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}^2\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} \boldsymbol{w}_l\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_l\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \frac{1}{N}\sum_{l=1}^{L}\sum_{l=1}^{N} \boldsymbol{w}_i^{'}\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_i \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L} \boldsymbol{w}_l^{'}\boldsymbol{\Sigma}\boldsymbol{w}_l \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L} \lambda_l\boldsymbol{w}_l^{'}\boldsymbol{w}_l
\end{aligned}
$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$
\begin{aligned}
\text{error}(L) &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} z_{il}^2\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i - \sum_{l=1}^{L} \boldsymbol{w}_l\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_l\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \frac{1}{N}\sum_{l=1}^{L}\sum_{l=1}^{N} \boldsymbol{w}_i^{'}\boldsymbol{x}_i\boldsymbol{x}_i^{'}\boldsymbol{w}_i \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L} \boldsymbol{w}_l^{'}\boldsymbol{\Sigma}\boldsymbol{w}_l \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L} \lambda_l\boldsymbol{w}_l^{'}\boldsymbol{w}_l \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L} \lambda_l
\end{aligned}
$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

If $L = J$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

If $L = J$

$$\text{error}(J) = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{J}\lambda_l = 0$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

If $L = J$

$$\text{error}(J) \quad = \quad \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{J}\lambda_l = 0$$

So for $L < J$,

How do we select the number of dimensions L? $\rightsquigarrow$ Model

If $L = J$

$$\text{error}(J) = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{J}\lambda_l = 0$$

So for $L < J$,

$$0 = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - (\sum_{l=1}^{L}\lambda_l + \sum_{j=L+1}^{J}\lambda_l)$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

If $L = J$

$$\text{error}(J) = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{J}\lambda_l = 0$$

So for $L < J$,

$$\begin{aligned} 0 &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - (\sum_{l=1}^{L}\lambda_l + \sum_{j=L+1}^{J}\lambda_l) \\ \sum_{j=L+1}^{J}\lambda_l &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L}\lambda_l \end{aligned}$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

If $L = J$

$$\text{error}(J) = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{J}\lambda_l = 0$$

So for $L < J$,

$$\begin{aligned} 0 &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \left(\sum_{l=1}^{L}\lambda_l + \sum_{j=L+1}^{J}\lambda_l\right) \\ \sum_{j=L+1}^{J}\lambda_l &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i^{'}\boldsymbol{x}_i\right) - \sum_{l=1}^{L}\lambda_l \\ \sum_{j=L+1}^{J}\lambda_l &= \text{error}(L) \end{aligned}$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$\sum_{j=L+1}^{J} \lambda_l = \text{error}(L)$$

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$\sum_{j=L+1}^{J} \lambda_l = \text{error}(L)$$

- Error = Sum of "remaining" eigenvalues

How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$\sum_{j=L+1}^{J} \lambda_l \quad = \quad \text{error}(L)$$

- Error = Sum of "remaining" eigenvalues
- Total variance explained = (sum of included eigenvalues)/(sum of all eigenvalues)

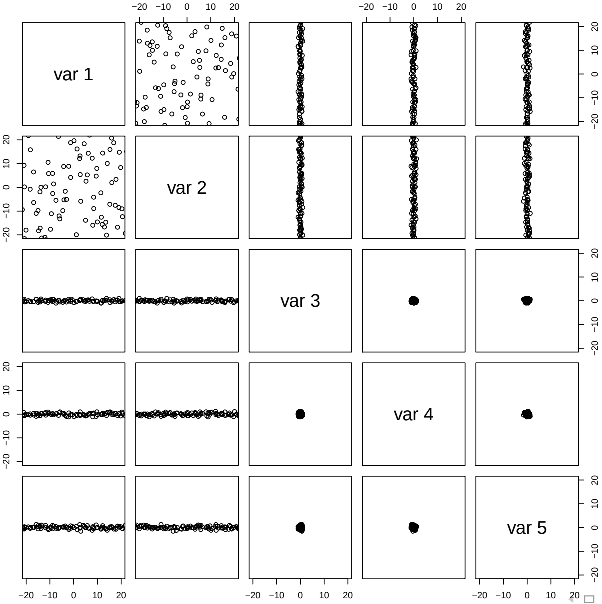
How do we select the number of dimensions L? $\rightsquigarrow$ Model

$$\sum_{j=L+1}^{J} \lambda_l \quad = \quad \text{error}(L)$$

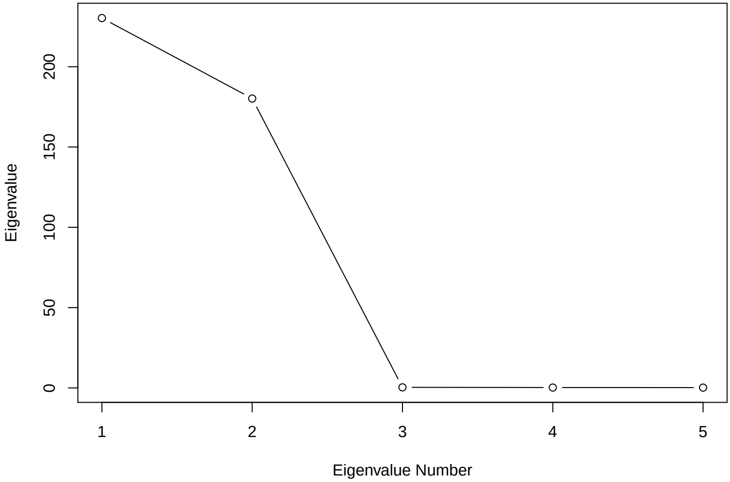
- Error = Sum of "remaining" eigenvalues
- Total variance explained = (sum of included eigenvalues)/(sum of all eigenvalues)

Recommendation $\rightsquigarrow$ look for Elbow

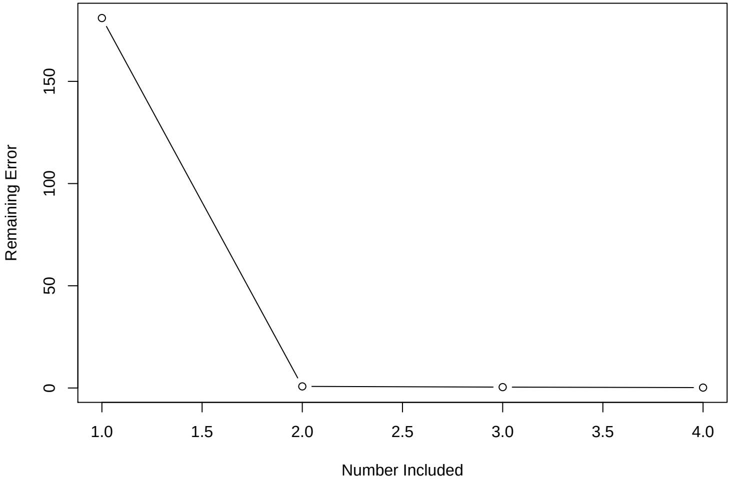
How do we select the number of dimensions L? $\rightsquigarrow$ Model



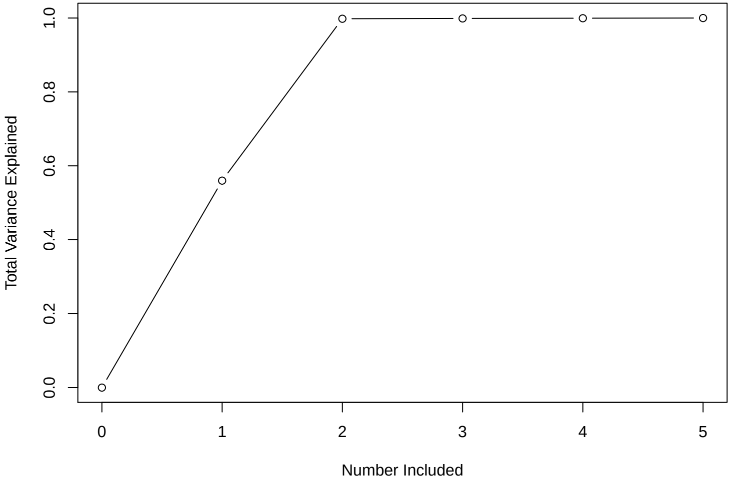
How do we select the number of dimensions L? $\rightsquigarrow$ Model



How do we select the number of dimensions L? $\rightsquigarrow$ Model



How do we select the number of dimensions L? $\rightsquigarrow$ Model



Non-model based evaluations: What's the point?

What is the true underlying dimensionality of $\boldsymbol{X}$?

Non-model based evaluations: What's the point?

What is the true underlying dimensionality of $\boldsymbol{X}$? J

Non-model based evaluations: What's the point?

What is the true underlying dimensionality of $\boldsymbol{X}$? J(!!!!!)

Non-model based evaluations: What's the point?

What is the true underlying dimensionality of $\boldsymbol{X}$? J(!!!!!)

- Attempts to assess dimensionality require a model $\rightsquigarrow$ some way to tradeoff accuracy of reconstruction with simplicity

Non-model based evaluations: What's the point?

What is the true underlying dimensionality of $\boldsymbol{X}$? J(!!!!!)

- Attempts to assess dimensionality require a model $\rightsquigarrow$ some way to tradeoff accuracy of reconstruction with simplicity
- Any answer (no matter how creatively obtained) supposes you have the right function to measure tradeoff

Non-model based evaluations: What's the point?

What is the true underlying dimensionality of $\boldsymbol{X}$? J(!!!!!)

- Attempts to assess dimensionality require a model $\rightsquigarrow$ some way to tradeoff accuracy of reconstruction with simplicity
- Any answer (no matter how creatively obtained) supposes you have the right function to measure tradeoff
- The "right" number of dimensions depends on the task you have in mind

Non-model based evaluations: What's the point?

What is the true underlying dimensionality of $\boldsymbol{X}$? J(!!!!!)

- Attempts to assess dimensionality require a model $\rightsquigarrow$ some way to tradeoff accuracy of reconstruction with simplicity
- Any answer (no matter how creatively obtained) supposes you have the right function to measure tradeoff
- The "right" number of dimensions depends on the task you have in mind

Mathematical model $\rightsquigarrow$ insufficient to make modeling decision

Kernel Principal Component Analysis

Recall from Thursday that we define a Kernel ($N \times N$) matrix as:

$$\boldsymbol{K} = \begin{pmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_1, x_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}$$

Kernel Principal Component Analysis

Recall from Thursday that we define a Kernel ($N \times N$) matrix as:

$$\boldsymbol{K} = \begin{pmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_1, x_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}$$

Where we suppose this matrix emerges from applying $\phi : \Re^J \rightarrow \Re^M$ to the data and then taking the inner product:

Kernel Principal Component Analysis

Recall from Thursday that we define a Kernel ($N \times N$) matrix as:

$$\boldsymbol{K} = \begin{pmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_1, x_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}$$

Where we suppose this matrix emerges from applying $\phi : \Re^J \rightarrow \Re^M$ to the data and then taking the inner product:

$$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'} \text{ (The inner product matrix)}$$

Kernel Principal Component Analysis

Recall from Thursday that we define a Kernel ($N \times N$) matrix as:

$$\boldsymbol{K} = \begin{pmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_1, x_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}$$

Where we suppose this matrix emerges from applying $\phi : \Re^J \rightarrow \Re^M$ to the data and then taking the inner product:

$$\begin{aligned} \boldsymbol{K} &= \boldsymbol{\Phi}\boldsymbol{\Phi}^{'} \text{ (The inner product matrix)} \\ &= \begin{pmatrix} \phi(\boldsymbol{x}_1)^{'}\phi(\boldsymbol{x}_1) & \phi(\boldsymbol{x}_1)^{'}\phi(\boldsymbol{x}_2) & \ldots & \phi(\boldsymbol{x}_1)^{'}\phi(\boldsymbol{x}_N) \\ \phi(\boldsymbol{x}_2)^{'}\phi(\boldsymbol{x}_1) & \phi(\boldsymbol{x}_2)^{'}\phi(\boldsymbol{x}_2) & \ldots & \phi(\boldsymbol{x}_2)^{'}\phi(\boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\boldsymbol{x}_N)^{'}\phi(\boldsymbol{x}_1) & \phi(\boldsymbol{x}_N)^{'}\phi(\boldsymbol{x}_2) & \ldots & \phi(\boldsymbol{x}_N)^{'}\phi(\boldsymbol{x}_N) \end{pmatrix} \end{aligned}$$

Kernel Principal Component Analysis

Recall from Thursday that we define a Kernel ($N \times N$) matrix as:

$$\boldsymbol{K} = \begin{pmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_1, x_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}$$

Where we suppose this matrix emerges from applying $\phi : \Re^J \rightarrow \Re^M$ to the data and then taking the inner product:

$$\begin{aligned} \boldsymbol{K} &= \boldsymbol{\Phi}\boldsymbol{\Phi}^{'} \text{ (The inner product matrix)} \\ &= \begin{pmatrix} \phi(\boldsymbol{x}_1)^{'}\phi(\boldsymbol{x}_1) & \phi(\boldsymbol{x}_1)^{'}\phi(\boldsymbol{x}_2) & \ldots & \phi(\boldsymbol{x}_1)^{'}\phi(\boldsymbol{x}_N) \\ \phi(\boldsymbol{x}_2)^{'}\phi(\boldsymbol{x}_1) & \phi(\boldsymbol{x}_2)^{'}\phi(\boldsymbol{x}_2) & \ldots & \phi(\boldsymbol{x}_2)^{'}\phi(\boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\boldsymbol{x}_N)^{'}\phi(\boldsymbol{x}_1) & \phi(\boldsymbol{x}_N)^{'}\phi(\boldsymbol{x}_2) & \ldots & \phi(\boldsymbol{x}_N)^{'}\phi(\boldsymbol{x}_N) \end{pmatrix} \end{aligned}$$

Compute PCA of $\boldsymbol{\Phi}$ from $\boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$

Kernel PCA

PCA of $\boldsymbol{X}$

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)

Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1.

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)

Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)
Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 \quad = \quad \lambda_1 \boldsymbol{u}_1$$

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)

Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$\begin{aligned}
(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\
(\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1(\boldsymbol{X}^{'}\boldsymbol{u}_1)
\end{aligned}$$

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)

Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$\begin{aligned}
(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\
(\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1 (\boldsymbol{X}^{'}\boldsymbol{u}_1) \\
&= \lambda_1 \boldsymbol{v}_1
\end{aligned}$$

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)
Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$\begin{aligned}(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\ (\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1(\boldsymbol{X}^{'}\boldsymbol{u}_1) \\ &= \lambda_1 \boldsymbol{v}_1\end{aligned}$$

But $\boldsymbol{v}_1$ needs unit length, and

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)

Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$\begin{aligned}
(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\
(\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1(\boldsymbol{X}^{'}\boldsymbol{u}_1) \\
&= \lambda_1 \boldsymbol{v}_1
\end{aligned}$$

But $\boldsymbol{v}_1$ needs unit length, and

$$||\boldsymbol{v}_1||^2 = \boldsymbol{v}_1^{'}\boldsymbol{v}_1$$

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)
Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$\begin{aligned}
(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\
(\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1(\boldsymbol{X}^{'}\boldsymbol{u}_1) \\
&= \lambda_1 \boldsymbol{v}_1
\end{aligned}$$

But $\boldsymbol{v}_1$ needs unit length, and

$$||\boldsymbol{v}_1||^2 = \boldsymbol{v}_1^{'}\boldsymbol{v}_1$$

$$= \boldsymbol{u}_1^{'}\boldsymbol{X}\boldsymbol{X}^{'}\boldsymbol{u}_1$$

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)
Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$\begin{aligned}(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\ (\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1(\boldsymbol{X}^{'}\boldsymbol{u}_1) \\ &= \lambda_1 \boldsymbol{v}_1\end{aligned}$$

But $\boldsymbol{v}_1$ needs unit length, and

$$\begin{aligned}||\boldsymbol{v}_1||^2 = \boldsymbol{v}_1^{'}\boldsymbol{v}_1 & \\ &= \boldsymbol{u}_1^{'}\boldsymbol{X}\boldsymbol{X}^{'}\boldsymbol{u}_1 \\ &= \lambda_1 \boldsymbol{u}_1^{'}\boldsymbol{u}_1 = \lambda_1\end{aligned}$$

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)

Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$
\begin{aligned}
(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\
(\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1 (\boldsymbol{X}^{'}\boldsymbol{u}_1) \\
&= \lambda_1 \boldsymbol{v}_1
\end{aligned}
$$

But $\boldsymbol{v}_1$ needs unit length, and

$$
||\boldsymbol{v}_1||^2 = \boldsymbol{v}_1^{'}\boldsymbol{v}_1
$$

$$
\begin{aligned}
&= \boldsymbol{u}_1^{'}\boldsymbol{X}\boldsymbol{X}^{'}\boldsymbol{u}_1 \\
&= \lambda_1 \boldsymbol{u}_1^{'}\boldsymbol{u}_1 = \lambda_1
\end{aligned}
$$

So first eigenvector of $\boldsymbol{X}^{'}\boldsymbol{X}$ is

Kernel PCA

PCA of $\boldsymbol{X}$ Eigenvectors of $\boldsymbol{X}^{'}\boldsymbol{X}$ ($\frac{1}{N}$ doesn't affect eigenvectors)
Suppose $\boldsymbol{u}_1$ is an eigenvector for $\boldsymbol{X}\boldsymbol{X}^{'}$, with value λ_1. Then

$$
\begin{aligned}
(\boldsymbol{X}\boldsymbol{X}^{'})\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1 \\
(\boldsymbol{X}^{'}\boldsymbol{X})(\boldsymbol{X}^{'}\boldsymbol{u}_1) &= \lambda_1(\boldsymbol{X}^{'}\boldsymbol{u}_1) \\
&= \lambda_1 \boldsymbol{v}_1
\end{aligned}
$$

But $\boldsymbol{v}_1$ needs unit length, and

$$
\begin{aligned}
||\boldsymbol{v}_1||^2 = \boldsymbol{v}_1^{'}\boldsymbol{v}_1 & \\
&= \boldsymbol{u}_1^{'}\boldsymbol{X}\boldsymbol{X}^{'}\boldsymbol{u}_1 \\
&= \lambda_1 \boldsymbol{u}_1^{'}\boldsymbol{u}_1 = \lambda_1
\end{aligned}
$$

So first eigenvector of $\boldsymbol{X}^{'}\boldsymbol{X}$ is

$$
\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}}\boldsymbol{X}^{'}\boldsymbol{u}_1
$$

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)

We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$.

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)

We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)

We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)
We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

But suppose we want to project a point $\phi(\boldsymbol{x}_i)$, then

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)
We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

But suppose we want to project a point $\phi(\boldsymbol{x}_i)$, then

$$\phi(\boldsymbol{x}_i)^{'}\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}}\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'}\boldsymbol{u}_1$$

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)
We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

But suppose we want to project a point $\phi(\boldsymbol{x}_i)$, then

$$\begin{aligned}
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{w}_1 &= \frac{1}{\sqrt{\lambda_1}}\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'}\boldsymbol{u}_1 \\
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'} &= \left[\phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_N)\right]
\end{aligned}$$

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)

We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

But suppose we want to project a point $\phi(\boldsymbol{x}_i)$, then

$$\begin{aligned}
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{w}_1 &= \frac{1}{\sqrt{\lambda_1}}\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'}\boldsymbol{u}_1 \\
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'} &= \left[\phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_N)\right] \\
&= [k(\boldsymbol{x}_i, \boldsymbol{x}_1), k(\boldsymbol{x}_i, \boldsymbol{x}_2), \ldots, k(\boldsymbol{x}_i, \boldsymbol{x}_N)]
\end{aligned}$$

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)
We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

But suppose we want to project a point $\phi(\boldsymbol{x}_i)$, then

$$\begin{aligned}
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{w}_1 &= \frac{1}{\sqrt{\lambda_1}}\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'}\boldsymbol{u}_1 \\
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'} &= \left[\phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_N)\right] \\
&= \left[k(\boldsymbol{x}_i, \boldsymbol{x}_1), k(\boldsymbol{x}_i, \boldsymbol{x}_2), \ldots, k(\boldsymbol{x}_i, \boldsymbol{x}_N)\right] = \boldsymbol{k}(\boldsymbol{x}_i, *)
\end{aligned}$$

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)
We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

But suppose we want to project a point $\phi(\boldsymbol{x}_i)$, then

$$\begin{aligned}
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{w}_1 &= \frac{1}{\sqrt{\lambda_1}}\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'}\boldsymbol{u}_1 \\
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'} &= \left[\phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_N)\right] \\
&= \left[k(\boldsymbol{x}_i, \boldsymbol{x}_1), k(\boldsymbol{x}_i, \boldsymbol{x}_2), \ldots, k(\boldsymbol{x}_i, \boldsymbol{x}_N)\right] = \boldsymbol{k}(\boldsymbol{x}_i, *)
\end{aligned}$$

Then, we can obtain projection for observation i using Kernel with

Kernel PCA

$\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{'}$ (assume $\boldsymbol{\Phi}$ is mean-centered, for now)
We can obtain $\boldsymbol{u}_1$ and λ_1 from $\boldsymbol{K}$. We know that

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}} \underbrace{\boldsymbol{\Phi}^{'}}_{\text{Unknown}} \boldsymbol{u}_1$$

But suppose we want to project a point $\phi(\boldsymbol{x}_i)$, then

$$\begin{aligned}
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{w}_1 &= \frac{1}{\sqrt{\lambda_1}}\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'}\boldsymbol{u}_1 \\
\phi(\boldsymbol{x}_i)^{'}\boldsymbol{\Phi}^{'} &= \left[\phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_i)^{'}\phi(\boldsymbol{x}_N)\right] \\
&= [k(\boldsymbol{x}_i, \boldsymbol{x}_1), k(\boldsymbol{x}_i, \boldsymbol{x}_2), \ldots, k(\boldsymbol{x}_i, \boldsymbol{x}_N)] = \boldsymbol{k}(\boldsymbol{x}_i, *)
\end{aligned}$$

Then, we can obtain projection for observation i using Kernel with

$$\phi(\boldsymbol{x}_i)^{'}\boldsymbol{w}_1 = \frac{1}{\sqrt{\lambda_1}}\boldsymbol{k}(\boldsymbol{x}_i, *)\boldsymbol{u}_1$$

Kernel PCA

Center $\boldsymbol{K}$?
Use centering matrix $\boldsymbol{H}$

$$\begin{aligned}\boldsymbol{H} &= \boldsymbol{I}_N - \frac{(\mathbf{1}_N\mathbf{1}_N^{'})}{N}\\ \boldsymbol{K}_{\text{center}} &= \boldsymbol{HKH}\end{aligned}$$

Spirling and Indian Treaties

Spirling (2013): model Treaties between US and Native Americans

Spirling and Indian Treaties

Spirling (2013): model Treaties between US and Native Americans
Why?

Spirling and Indian Treaties

Spirling (2013): model Treaties between US and Native Americans

Why?

- American political development

Spirling and Indian Treaties

Spirling (2013): model Treaties between US and Native Americans
Why?

- American political development
- IR Theories of Treaties and Treaty Violations

Spirling and Indian Treaties

Spirling (2013): model Treaties between US and Native Americans
Why?

- American political development
- IR Theories of Treaties and Treaty Violations
- Comparative studies of indigenous/colonialist interaction

Spirling and Indian Treaties

Spirling (2013): model Treaties between US and Native Americans

Why?

- American political development
- IR Theories of Treaties and Treaty Violations
- Comparative studies of indigenous/colonialist interaction
- Political Science question: how did Native Americans lose land so quickly?

Spirling and Indian Treaties

How do we preserve word order and semantic language?

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us
```

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

- `Peace Between Us`
- `No Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.
Spirling uses complicated representation of texts to preserve word order

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

- `Peace Between Us`
- `No Peace Between Us`

are identical.
Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

Spirling and Indian Treaties

How do we preserve word order and semantic language?

After stemming, stopping, bag of wording:

```
- Peace Between Us

- No Peace Between Us
```

are identical.

Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?

After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.

Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.
Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.
Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.
Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.
Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?

After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.

Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?

After stemming, stopping, bag of wording:

```
- Peace Between Us

- No Peace Between Us
```

are identical.

Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

```
- Peace Between Us

- No Peace Between Us
```

are identical.
Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?

After stemming, stopping, bag of wording:

- `Peace Between Us`
- `No Peace Between Us`

are identical.

Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?
After stemming, stopping, bag of wording:

- `Peace Between Us`
- `No Peace Between Us`

are identical.
Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?

After stemming, stopping, bag of wording:

```
- Peace Between Us
- No Peace Between Us
```

are identical.

Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

How do we preserve word order and semantic language?

After stemming, stopping, bag of wording:

```
- Peace Between Us

- No Peace Between Us
```

are identical.

Spirling uses complicated representation of texts to preserve word order $\rightsquigarrow$ broad application

`Peace Between Us`

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.

We say $\boldsymbol{x}_i \in \mathcal{X}$

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.

We say $\boldsymbol{x}_i \in \mathcal{X}$

Spirling examines 5-character strings, $s \in \mathcal{A}$

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.
We say $\boldsymbol{x}_i \in \mathcal{X}$
Spirling examines 5-character strings, $s \in \mathcal{A}$
Define:

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.

We say $\boldsymbol{x}_i \in \mathcal{X}$

Spirling examines 5-character strings, $s \in \mathcal{A}$

Define:

$\phi_s : \mathcal{X} \rightarrow \Re$ as a function that counts the number of times string s occurs in document $\boldsymbol{x}$.

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.

We say $\boldsymbol{x}_i \in \mathcal{X}$

Spirling examines 5-character strings, $s \in \mathcal{A}$

Define:

$\phi_s : \mathcal{X} \rightarrow \Re$ as a function that counts the number of times string s occurs in document $\boldsymbol{x}$.

Define string kernel to be,

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.
We say $\boldsymbol{x}_i \in \mathcal{X}$
Spirling examines 5-character strings, $s \in \mathcal{A}$
Define:
$\phi_s : \mathcal{X} \rightarrow \Re$ as a function that counts the number of times string s occurs in document $\boldsymbol{x}$.
Define string kernel to be,

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{s \in \mathcal{A}} w_s \phi_s(\boldsymbol{x}_i) \phi_s(\boldsymbol{x}_j)$$

Spirling and Indian Treaties

Consider documents $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where we have preserved order, punctuation, and all else.

We say $\boldsymbol{x}_i \in \mathcal{X}$

Spirling examines 5-character strings, $s \in \mathcal{A}$

Define:

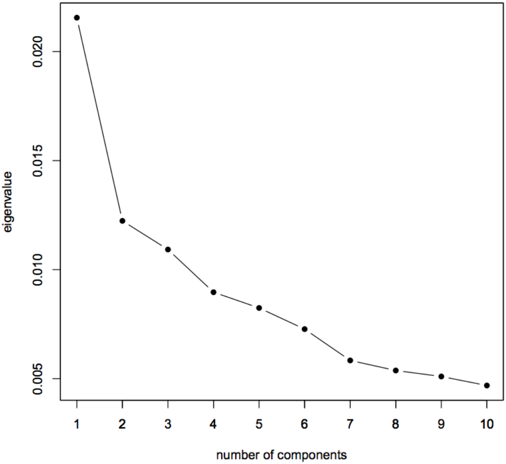
$\phi_s : \mathcal{X} \rightarrow \Re$ as a function that counts the number of times string s occurs in document $\boldsymbol{x}$.

Define string kernel to be,

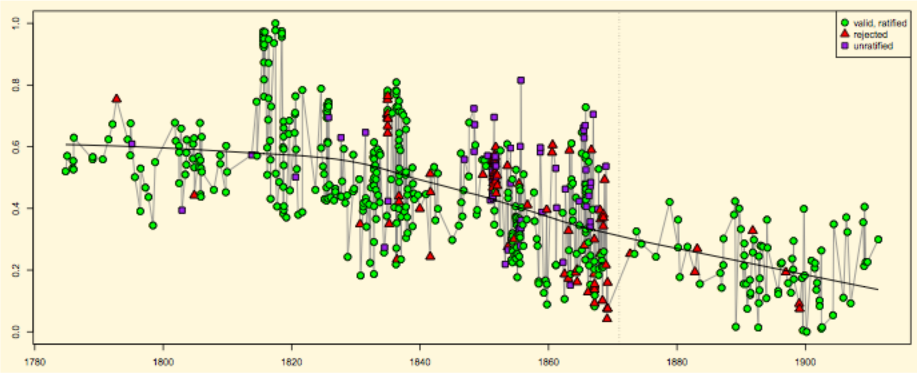
$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{s \in \mathcal{A}} w_s \phi_s(\boldsymbol{x}_i) \phi_s(\boldsymbol{x}_j)$$

$\boldsymbol{\phi}(\boldsymbol{x}_i) \approx \binom{32}{5}$ element long count vector

Spirling and Indian Treaties



Spirling and Indian Treaties



Classic Multidimensional Scaling $\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D}$ $\rightsquigarrow$

Classic Multidimensional Scaling $\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D} \rightsquigarrow$ matrix of squared Euclidean distances between documents $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Re^J$.

Classic Multidimensional Scaling $\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D} \rightsquigarrow$ matrix of squared Euclidean distances between documents $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Re^J$.
Typical distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

Classic Multidimensional Scaling$\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D} \rightsquigarrow$ matrix of squared Euclidean distances between documents $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Re^J$.
Typical distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

$$d_{ij}^2 = \sum_{k=1}^{J} (x_{ik} - x_{jk})^2$$

Classic Multidimensional Scaling $\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D} \rightsquigarrow$ matrix of squared Euclidean distances between documents $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Re^J$.
Typical distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

$$d_{ij}^2 = \sum_{k=1}^{J} (x_{ik} - x_{jk})^2$$

For all N documents we look for points $\boldsymbol{z}_i \in \Re^L$ (with $L < J$) to minimize

Classic Multidimensional Scaling $\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D} \rightsquigarrow$ matrix of squared Euclidean distances between documents $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Re^J$.
Typical distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

$$d_{ij}^2 = \sum_{k=1}^{J} (x_{ik} - x_{jk})^2$$

For all N documents we look for points $\boldsymbol{z}_i \in \Re^L$ (with $L < J$) to minimize

$$f(\boldsymbol{X}, \boldsymbol{\theta}) = f(\boldsymbol{X}, \boldsymbol{Z})$$

Classic Multidimensional Scaling $\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D} \rightsquigarrow$ matrix of squared Euclidean distances between documents $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Re^J$.
Typical distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

$$d_{ij}^2 = \sum_{k=1}^{J} (x_{ik} - x_{jk})^2$$

For all N documents we look for points $\boldsymbol{z}_i \in \Re^L$ (with $L < J$) to minimize

$$\begin{aligned} f(\boldsymbol{X}, \boldsymbol{\theta}) &= f(\boldsymbol{X}, \boldsymbol{Z}) \\ &= \sum_{i \neq j} (d_{ij} - ||z_i - z_j||)^2 \end{aligned}$$

Classic Multidimensional Scaling $\rightsquigarrow$ Objective Function

Suppose we have an $N \times N$ matrix $\boldsymbol{D} \rightsquigarrow$ matrix of squared Euclidean distances between documents $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Re^J$.
Typical distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

$$d_{ij}^2 = \sum_{k=1}^{J} (x_{ik} - x_{jk})^2$$

For all N documents we look for points $\boldsymbol{z}_i \in \Re^L$ (with $L < J$) to minimize

$$\begin{aligned} f(\boldsymbol{X}, \boldsymbol{\theta}) &= f(\boldsymbol{X}, \boldsymbol{Z}) \\ &= \sum_{i \neq j} (d_{ij} - ||z_i - z_j||)^2 \end{aligned}$$

Recover best low-dimensional representation $\rightsquigarrow$ Eigenvector decomposition of $\boldsymbol{D}$

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

1) Start with $\boldsymbol{D}$
2) "Center" $\boldsymbol{D}$ to obtain $\boldsymbol{X}\boldsymbol{X}^{'}$
3) Find largest eigenvectors of $\boldsymbol{X}\boldsymbol{X}^{'}$

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

Center the distance matrix, obtain inner product

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

Center the distance matrix, obtain inner product

$$d_{ij}^2 = \sum_{k=1}^{J} (x_{ik} - x_{jk})^2$$

Classic Multidimensional Scaling$\rightsquigarrow$ Optimization

Center the distance matrix, obtain inner product

$$
\begin{aligned}
d_{ij}^2 &= \sum_{k=1}^{J} (x_{ik} - x_{jk})^2 \\
&= \sum_{k=1}^{J} (x_{ik}^2 - 2x_{ik}x_{jk} + x_{jk}^2)
\end{aligned}
$$

Classic Multidimensional Scaling$\leadsto$ Optimization

Center the distance matrix, obtain inner product

$$\begin{aligned}
d_{ij}^2 &= \sum_{k=1}^{J} (x_{ik} - x_{jk})^2 \\
&= \sum_{k=1}^{J} (x_{ik}^2 - 2x_{ik}x_{jk} + x_{jk}^2) \\
&= \underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{rows}} - 2\boldsymbol{x}_i\boldsymbol{x}_j + \underbrace{\boldsymbol{x}_j^{'}\boldsymbol{x}_j}_{\text{columns}}
\end{aligned}$$

Classic Multidimensional Scaling$\rightsquigarrow$ Optimization

Center the distance matrix, obtain inner product

$$
\begin{aligned}
d_{ij}^2 &= \sum_{k=1}^{J} (x_{ik} - x_{jk})^2 \\
&= \sum_{k=1}^{J} (x_{ik}^2 - 2x_{ik}x_{jk} + x_{jk}^2) \\
&= \underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{rows}} - 2\boldsymbol{x}_i\boldsymbol{x}_j + \underbrace{\boldsymbol{x}_j^{'}\boldsymbol{x}_j}_{\text{columns}}
\end{aligned}
$$

- Subtract off row means

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

Center the distance matrix, obtain inner product

$$
\begin{aligned}
d_{ij}^2 &= \sum_{k=1}^{J} (x_{ik} - x_{jk})^2 \\
&= \sum_{k=1}^{J} (x_{ik}^2 - 2x_{ik}x_{jk} + x_{jk}^2) \\
&= \underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{rows}} - 2\boldsymbol{x}_i\boldsymbol{x}_j + \underbrace{\boldsymbol{x}_j^{'}\boldsymbol{x}_j}_{\text{columns}}
\end{aligned}
$$

- Subtract off row means
- Subtract off column means

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

Center the distance matrix, obtain inner product

$$
\begin{aligned}
d_{ij}^2 &= \sum_{k=1}^{J}(x_{ik} - x_{jk})^2 \\
&= \sum_{k=1}^{J}(x_{ik}^2 - 2x_{ik}x_{jk} + x_{jk}^2) \\
&= \underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{rows}} - 2\boldsymbol{x}_i\boldsymbol{x}_j + \underbrace{\boldsymbol{x}_j^{'}\boldsymbol{x}_j}_{\text{columns}}
\end{aligned}
$$

- Subtract off row means
- Subtract off column means
- Divide by -2

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

Center the distance matrix, obtain inner product

$$
\begin{aligned}
d_{ij}^2 &= \sum_{k=1}^{J}(x_{ik} - x_{jk})^2 \\
&= \sum_{k=1}^{J}(x_{ik}^2 - 2x_{ik}x_{jk} + x_{jk}^2) \\
&= \underbrace{\boldsymbol{x}_i^{'}\boldsymbol{x}_i}_{\text{rows}} - 2\boldsymbol{x}_i\boldsymbol{x}_j + \underbrace{\boldsymbol{x}_j^{'}\boldsymbol{x}_j}_{\text{columns}}
\end{aligned}
$$

- Subtract off row means
- Subtract off column means
- Divide by -2

$$
\boldsymbol{X}\boldsymbol{X}^{'} = \frac{-\boldsymbol{HDH}}{2}
$$

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

We can write $\boldsymbol{X}\boldsymbol{X}^{'}$ as

$$
\begin{aligned}
\boldsymbol{X}\boldsymbol{X}^{'} &= \boldsymbol{W}^{'} \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_N \end{pmatrix} \boldsymbol{W} \\
&= \underbrace{\boldsymbol{W}^{'} \begin{pmatrix} \sqrt{\lambda_1} & 0 & \ldots & 0 \\ 0 & \sqrt{\lambda_2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \sqrt{\lambda_N} \end{pmatrix}}_{\boldsymbol{X}} \underbrace{\begin{pmatrix} \sqrt{\lambda_1} & 0 & \ldots & 0 \\ 0 & \sqrt{\lambda_2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \sqrt{\lambda_N} \end{pmatrix} \boldsymbol{W}}_{\boldsymbol{X}^{'}}
\end{aligned}
$$

Approximate $\boldsymbol{X}$ with first L eigenvectors

Classic Multidimensional Scaling$\rightsquigarrow$ Optimization

Define $\boldsymbol{z}_i \in \Re^L$ as,

$$\boldsymbol{z}_i^* = \left(\sqrt{\lambda_1}w_{1i}, \sqrt{\lambda_2}w_{2i}, \ldots, \sqrt{\lambda_L}w_{Li}\right)$$

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

Define $\boldsymbol{z}_i \in \Re^L$ as,

$$\boldsymbol{z}_i^* \quad = \quad \left(\sqrt{\lambda_1} w_{1i}, \sqrt{\lambda_2} w_{2i}, \ldots, \sqrt{\lambda_L} w_{Li}\right)$$

- Same diagnostics as before: eigenvalues

Classic Multidimensional Scaling $\rightsquigarrow$ Optimization

Define $\boldsymbol{z}_i \in \Re^L$ as,

$$\boldsymbol{z}_i^* = \left(\sqrt{\lambda_1} w_{1i}, \sqrt{\lambda_2} w_{2i}, \ldots, \sqrt{\lambda_L} w_{Li}\right)$$

- Same diagnostics as before: eigenvalues
- Same issues in dimension reduction $\rightsquigarrow$ what are the dimensions for your task

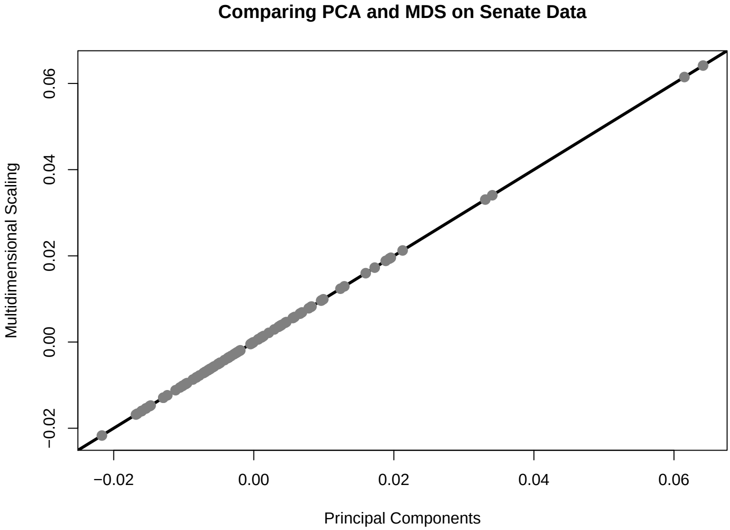
Classic Multidimensional Scaling$\leadsto$ Optimization

Define $\boldsymbol{z}_i \in \Re^L$ as,

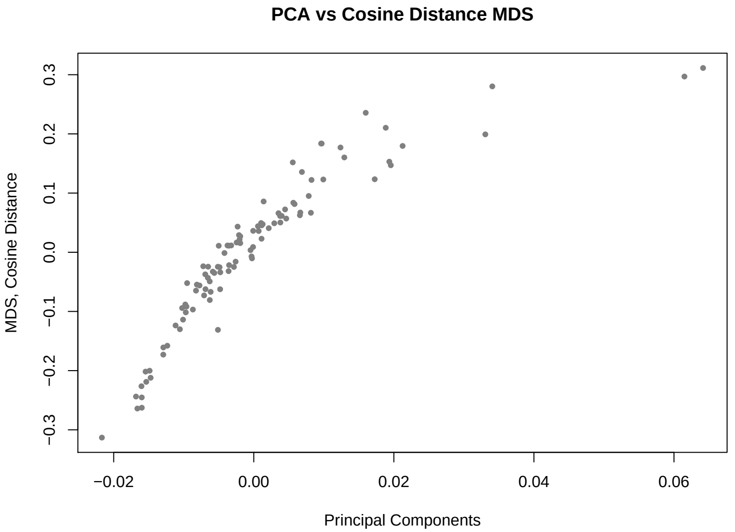
$$\boldsymbol{z}_i^* = \left(\sqrt{\lambda_1} w_{1i}, \sqrt{\lambda_2} w_{2i}, \ldots, \sqrt{\lambda_L} w_{Li}\right)$$

- Same diagnostics as before: eigenvalues
- Same issues in dimension reduction$\leadsto$ what are the dimensions for your task
- Information up to rotation and arbitrary center (same result with any orthogonal matrix $\boldsymbol{M}$)

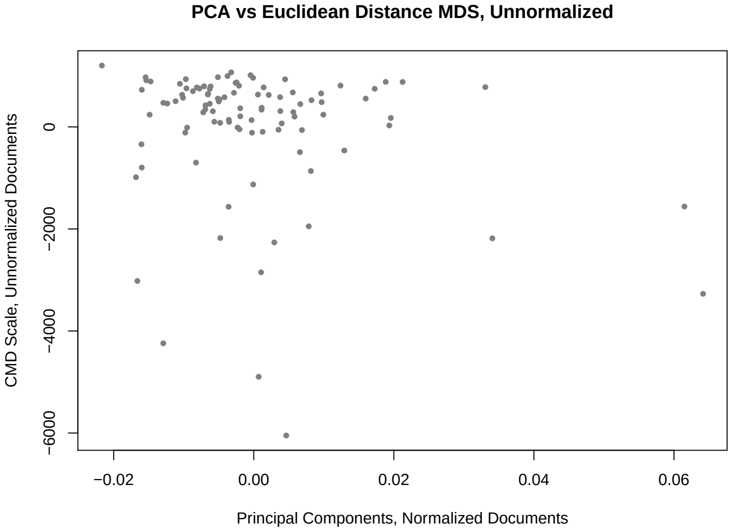
Comparing MDS and PCA



Comparing MDS and PCA



Comparing MDS and PCA



Sammon Multidimensional Scaling $\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

Sammon Multidimensional Scaling $\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling $\rightsquigarrow$ prioritizes small differences

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\rightsquigarrow$ for large scale embeddings

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\rightsquigarrow$ for large scale embeddings
- ISOMap$\rightsquigarrow$ Non-linear embedding, emphasizing local patterns

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\rightsquigarrow$ for large scale embeddings
- ISOMap$\rightsquigarrow$ Non-linear embedding, emphasizing local patterns

1) Calculate distance matrix

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\rightsquigarrow$ for large scale embeddings
- ISOMap$\rightsquigarrow$ Non-linear embedding, emphasizing local patterns

1) Calculate distance matrix
2) Create K nearest neighbor graph: find K nearest neighbors for each point, with edge weight $=$ Euclidean distance

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\rightsquigarrow$ for large scale embeddings
- ISOMap$\rightsquigarrow$ Non-linear embedding, emphasizing local patterns

1) Calculate distance matrix
2) Create K nearest neighbor graph: find K nearest neighbors for each point, with edge weight = Euclidean distance
3) Compute shortest weighted path between each node (distance matrix)

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\rightsquigarrow$ for large scale embeddings
- ISOMap$\rightsquigarrow$ Non-linear embedding, emphasizing local patterns

1) Calculate distance matrix
2) Create K nearest neighbor graph: find K nearest neighbors for each point, with edge weight = Euclidean distance
3) Compute shortest weighted path between each node (distance matrix)
4) Apply classic MDS

Sammon Multidimensional Scaling$\leadsto$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\leadsto$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\leadsto$ for large scale embeddings
- ISOMap$\leadsto$ Non-linear embedding, emphasizing local patterns

1) Calculate distance matrix
2) Create K nearest neighbor graph: find K nearest neighbors for each point, with edge weight = Euclidean distance
3) Compute shortest weighted path between each node (distance matrix)
4) Apply classic MDS

Common theme$\leadsto$ Manifold Learning

Sammon Multidimensional Scaling$\rightsquigarrow$ Manifold Learning

Many other methods we won't cover:

- Sammon Multidimensional Scaling$\rightsquigarrow$ prioritizes small differences

$$f(\boldsymbol{X}, \boldsymbol{Z}) = \sum_{i \neq j} \frac{(d_{ij} - ||z_i - z_j||)^2}{d_{ij}}$$

- Landmark MDS$\rightsquigarrow$ for large scale embeddings
- ISOMap$\rightsquigarrow$ Non-linear embedding, emphasizing local patterns

1) Calculate distance matrix
2) Create K nearest neighbor graph: find K nearest neighbors for each point, with edge weight $=$ Euclidean distance
3) Compute shortest weighted path between each node (distance matrix)
4) Apply classic MDS

Common theme$\rightsquigarrow$ Manifold Learning

Low Dimensional Embedding

1) Find lower dimensional space to summarize documents $\rightsquigarrow$ Eigenvectors
2) Thursday: basic language models and the Dirichlet distribution