Linear Dimensionality Reduction

Practical Machine Learning (CS294-34)

September 24, 2009

Percy Liang

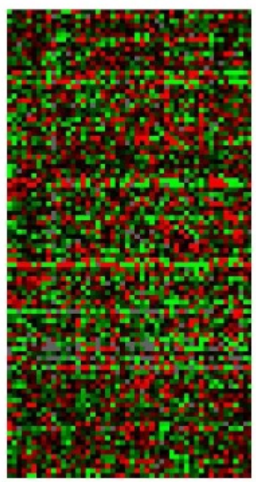
Lots of high-dimensional data...

face images

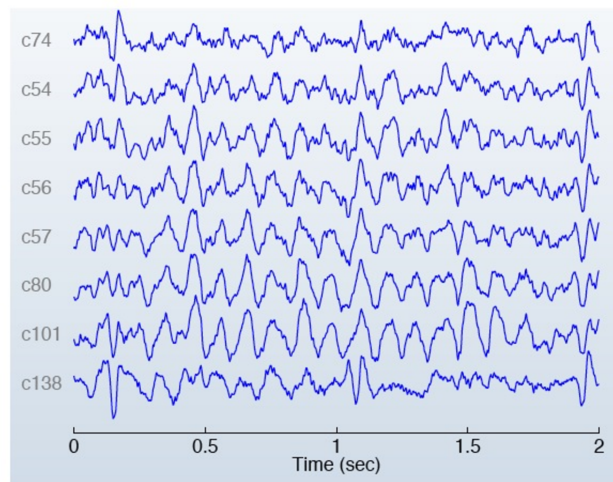
Zambian President Levy Mwanawasa has won a second term in office in an election his challenger Michael Sata accused him of rigging, official results showed on Monday.

According to media reports, a pair of hackers said on Saturday that the Firefox Web browser, commonly perceived as the safer and more customizable alternative to market leader Internet Explorer, is critically flawed. A presentation on the flaw was shown during the ToorCon hacker conference in San Diego.

documents



gene expression data



MEG readings

Motivation and context

Why do dimensionality reduction?

- Computational: compress data $\Rightarrow$ time/space efficiency

Motivation and context

Why do dimensionality reduction?

- Computational: compress data $\Rightarrow$ time/space efficiency
- Statistical: fewer dimensions $\Rightarrow$ better generalization

Motivation and context

Why do dimensionality reduction?

- Computational: compress data $\Rightarrow$ time/space efficiency
- Statistical: fewer dimensions $\Rightarrow$ better generalization
- Visualization: understand structure of data

Motivation and context

Why do dimensionality reduction?

- Computational: compress data $\Rightarrow$ time/space efficiency
- Statistical: fewer dimensions $\Rightarrow$ better generalization
- Visualization: understand structure of data
- Anomaly detection: describe normal data, detect outliers

Motivation and context

Why do dimensionality reduction?

- Computational: compress data $\Rightarrow$ time/space efficiency
- Statistical: fewer dimensions $\Rightarrow$ better generalization
- Visualization: understand structure of data
- Anomaly detection: describe normal data, detect outliers

Dimensionality reduction in this course:

- Linear methods (this week)
- Clustering (last week)
- Feature selection (next week)
- Nonlinear methods (later)

Types of problems

- Prediction $\mathbf{x} \longrightarrow \mathbf{y}$: classification, regression

Types of problems

- Prediction $\mathbf{x} \rightarrow \mathbf{y}$: classification, regression
  Applications: face recognition, gene expression prediction
  Techniques: kNN, SVM, least squares (+ dimensionality reduction preprocessing)

Types of problems

- Prediction $\mathbf{x} \rightarrow \mathbf{y}$: classification, regression
  Applications: face recognition, gene expression prediction
  Techniques: kNN, SVM, least squares (+ dimensionality reduction preprocessing)

- Structure discovery $\mathbf{x} \rightarrow \mathbf{z}$: find an alternative representation $\mathbf{z}$ of data $\mathbf{x}$

Types of problems

- Prediction $\mathbf{x} \rightarrow \mathbf{y}$: classification, regression
  Applications: face recognition, gene expression prediction
  Techniques: kNN, SVM, least squares (+ dimensionality reduction preprocessing)

- Structure discovery $\mathbf{x} \rightarrow \mathbf{z}$: find an alternative representation $\mathbf{z}$ of data $\mathbf{x}$
  Applications: visualization
  Techniques: clustering, linear dimensionality reduction

Types of problems

- Prediction $\mathbf{x} \longrightarrow \mathbf{y}$: classification, regression
  Applications: face recognition, gene expression prediction
  Techniques: kNN, SVM, least squares ($+$ dimensionality reduction preprocessing)

- Structure discovery $\mathbf{x} \longrightarrow \mathbf{z}$: find an alternative representation $\mathbf{z}$ of data $\mathbf{x}$
  Applications: visualization
  Techniques: clustering, linear dimensionality reduction

- Density estimation $p(\mathbf{x})$: model the data

Types of problems

- Prediction $\mathbf{x} \rightarrow \mathbf{y}$: classification, regression
  Applications: face recognition, gene expression prediction
  Techniques: kNN, SVM, least squares (+ dimensionality reduction preprocessing)

- Structure discovery $\mathbf{x} \rightarrow \mathbf{z}$: find an alternative representation $\mathbf{z}$ of data $\mathbf{x}$
  Applications: visualization
  Techniques: clustering, linear dimensionality reduction

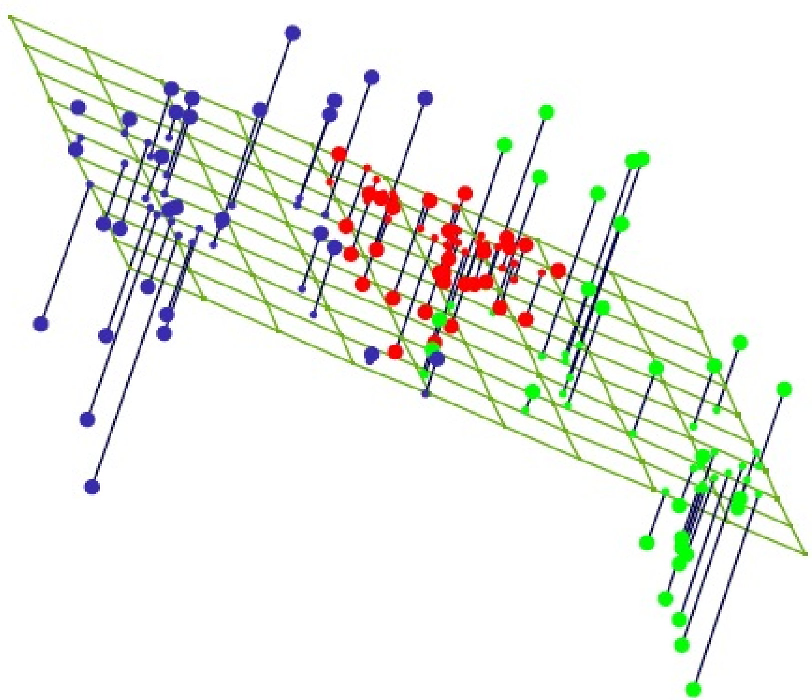
- Density estimation $p(\mathbf{x})$: model the data
  Applications: anomaly detection, language modeling
  Techniques: clustering, linear dimensionality reduction

Basic idea of linear dimensionality reduction

Represent each face as a high-dimensional vector $\mathbf{x} \in \mathbb{R}^{361}$

Basic idea of linear dimensionality reduction

Represent each face as a high-dimensional vector $\mathbf{x} \in \mathbb{R}^{361}$



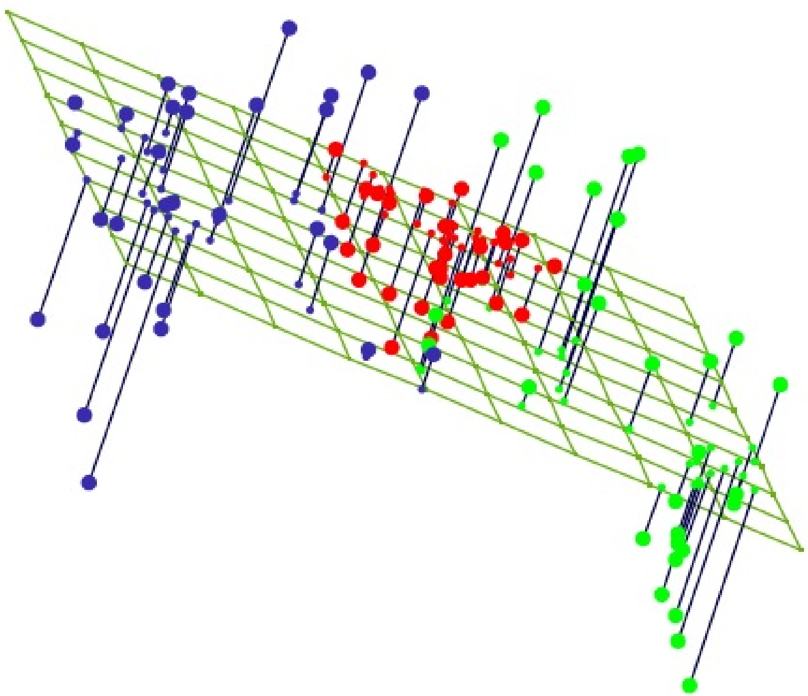
$$\mathbf{x} \in \mathbb{R}^{361}$$

$$\mathbf{z} = \mathbf{U}^\top \mathbf{x}$$

$$\mathbf{z} \in \mathbb{R}^{10}$$

Basic idea of linear dimensionality reduction

Represent each face as a high-dimensional vector $\mathbf{x} \in \mathbb{R}^{361}$



$$\mathbf{x} \in \mathbb{R}^{361}$$

$$\mathbf{z} = \mathbf{U}^\top \mathbf{x}$$

$$\mathbf{z} \in \mathbb{R}^{10}$$

How do we choose $\mathbf{U}$?

Outline

- Principal component analysis (PCA)
  - Basic principles
  - Case studies
  - Kernel PCA
  - Probabilistic PCA
- Canonical correlation analysis (CCA)
- Fisher discriminant analysis (FDA)
- Summary

Roadmap

- Principal component analysis (PCA)
  - Basic principles
  - Case studies
  - Kernel PCA
  - Probabilistic PCA
- Canonical correlation analysis (CCA)
- Fisher discriminant analysis (FDA)
- Summary

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Want to reduce dimensionality from d to k

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Want to reduce dimensionality from d to k

Choose k directions $\mathbf{u}_1, \ldots, \mathbf{u}_k$

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Want to reduce dimensionality from d to k

Choose k directions $\mathbf{u}_1, \dots, \mathbf{u}_k$

$$\mathbf{U} = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \cdot\cdot & \mathbf{u}_k \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times k}$$

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Want to reduce dimensionality from d to k

Choose k directions $\mathbf{u}_1, \ldots, \mathbf{u}_k$

$$\mathbf{U} = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \cdot\cdot & \mathbf{u}_k \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times k}$$

For each $\mathbf{u}_j$, compute "similarity" $z_j = \mathbf{u}_j^\top \mathbf{x}$

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Want to reduce dimensionality from d to k

Choose k directions $\mathbf{u}_1, \ldots, \mathbf{u}_k$

$$\mathbf{U} = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \cdot\cdot & \mathbf{u}_k \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times k}$$

For each $\mathbf{u}_j$, compute "similarity" $z_j = \mathbf{u}_j^\top \mathbf{x}$

Project $\mathbf{x}$ down to $\mathbf{z} = (z_1, \ldots, z_k)^\top = \mathbf{U}^\top \mathbf{x}$

Dimensionality reduction setup

Given n data points in d dimensions: $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Want to reduce dimensionality from d to k

Choose k directions $\mathbf{u}_1, \dots, \mathbf{u}_k$

$$\mathbf{U} = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \cdot\cdot & \mathbf{u}_k \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times k}$$

For each $\mathbf{u}_j$, compute "similarity" $z_j = \mathbf{u}_j^\top \mathbf{x}$

Project $\mathbf{x}$ down to $\mathbf{z} = (z_1, \dots, z_k)^\top = \mathbf{U}^\top \mathbf{x}$

How to choose $\mathbf{U}$?

PCA objective 1: reconstruction error

$\mathbf{U}$ serves two functions:

- Encode: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}, \quad z_j = \mathbf{u}_j^\top \mathbf{x}$

PCA objective 1: reconstruction error

$\mathbf{U}$ serves two functions:

- Encode: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}$, $\quad z_j = \mathbf{u}_j^\top \mathbf{x}$
- Decode: $\tilde{\mathbf{x}} = \mathbf{U}\mathbf{z} = \sum_{j=1}^{k} z_j \mathbf{u}_j$

PCA objective 1: reconstruction error

$\mathbf{U}$ serves two functions:

- Encode: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}$, $\quad z_j = \mathbf{u}_j^\top \mathbf{x}$
- Decode: $\tilde{\mathbf{x}} = \mathbf{U}\mathbf{z} = \sum_{j=1}^{k} z_j \mathbf{u}_j$

Want reconstruction error $\|\mathbf{x} - \tilde{\mathbf{x}}\|$ to be small

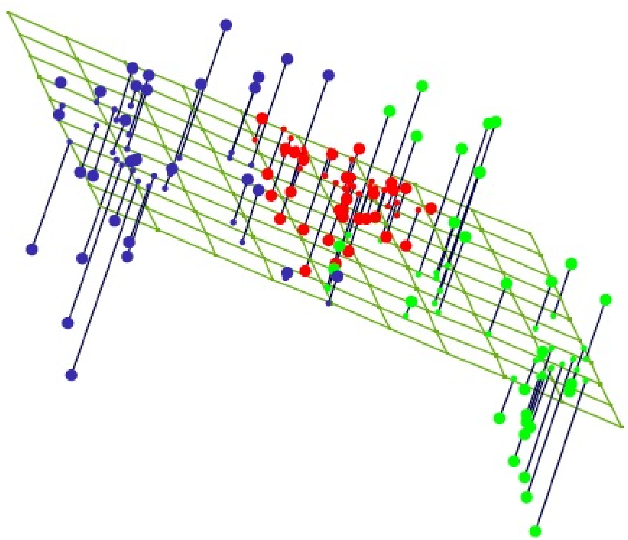
PCA objective 1: reconstruction error

$\mathbf{U}$ serves two functions:

- Encode: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}$, $\quad z_j = \mathbf{u}_j^\top \mathbf{x}$
- Decode: $\tilde{\mathbf{x}} = \mathbf{U}\mathbf{z} = \sum_{j=1}^{k} z_j \mathbf{u}_j$

Want reconstruction error $\|\mathbf{x} - \tilde{\mathbf{x}}\|$ to be small

Objective: minimize total squared reconstruction error



$$\min_{\mathbf{U} \in \mathbb{R}^{d \times k}} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{U}\mathbf{U}^\top \mathbf{x}_i\|^2$$

PCA objective 2: projected variance

Empirical distribution: uniform over $\mathbf{x}_1, \ldots, \mathbf{x}_n$

PCA objective 2: projected variance

Empirical distribution: uniform over $\mathbf{x}_1, \dots, \mathbf{x}_n$

Expectation (think sum over data points):

$$\hat{\mathbb{E}}[f(\mathbf{x})] = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_i)$$

PCA objective 2: projected variance

Empirical distribution: uniform over $\mathbf{x}_1, \dots, \mathbf{x}_n$

Expectation (think sum over data points):

$$\hat{\mathbb{E}}[f(\mathbf{x})] = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_i)$$

Variance (think sum of squares if centered):

$$\widehat{\text{var}}[f(\mathbf{x})] + (\hat{\mathbb{E}}[f(\mathbf{x})])^2 = \hat{\mathbb{E}}[f(\mathbf{x})^2] = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_i)^2$$

PCA objective 2: projected variance

Empirical distribution: uniform over $\mathbf{x}_1, \dots, \mathbf{x}_n$

Expectation (think sum over data points):

$$\hat{\mathbb{E}}[f(\mathbf{x})] = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_i)$$

Variance (think sum of squares if centered):

$$\widehat{\mathsf{var}}[f(\mathbf{x})] + (\hat{\mathbb{E}}[f(\mathbf{x})])^2 = \hat{\mathbb{E}}[f(\mathbf{x})^2] = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_i)^2$$

Assume data is centered: $\hat{\mathbb{E}}[\mathbf{x}] = 0$

PCA objective 2: projected variance

Empirical distribution: uniform over $\mathbf{x}_1, \ldots, \mathbf{x}_n$

Expectation (think sum over data points):

$$\hat{\mathbb{E}}[f(\mathbf{x})] = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i)$$

Variance (think sum of squares if centered):

$$\widehat{\text{var}}[f(\mathbf{x})] + (\hat{\mathbb{E}}[f(\mathbf{x})])^2 = \hat{\mathbb{E}}[f(\mathbf{x})^2] = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i)^2$$

Assume data is centered: $\hat{\mathbb{E}}[\mathbf{x}] = 0$ (what's $\hat{\mathbb{E}}[\mathbf{U}^\top \mathbf{x}]$?)

PCA objective 2: projected variance

Empirical distribution: uniform over $\mathbf{x}_1, \ldots, \mathbf{x}_n$

Expectation (think sum over data points):

$$\hat{\mathbb{E}}[f(\mathbf{x})] = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i)$$

Variance (think sum of squares if centered):

$$\widehat{\mathrm{var}}[f(\mathbf{x})] + (\hat{\mathbb{E}}[f(\mathbf{x})])^2 = \hat{\mathbb{E}}[f(\mathbf{x})^2] = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i)^2$$

Assume data is centered: $\hat{\mathbb{E}}[\mathbf{x}] = 0$ (what's $\hat{\mathbb{E}}[\mathbf{U}^\top \mathbf{x}]$?)

Objective: maximize variance of projected data

$$\max_{\mathbf{U} \in \mathbb{R}^{d \times k}, \mathbf{U}^\top \mathbf{U} = I} \hat{\mathbb{E}}[\|\mathbf{U}^\top \mathbf{x}\|^2]$$

Equivalence in two objectives

Key intuition:

$$\underbrace{\text{variance of data}}_{\text{fixed}} = \underbrace{\text{captured variance}}_{\text{want large}} + \underbrace{\text{reconstruction error}}_{\text{want small}}$$

Equivalence in two objectives

Key intuition:

$$\underbrace{\text{variance of data}}_{\text{fixed}} = \underbrace{\text{captured variance}}_{\text{want large}} + \underbrace{\text{reconstruction error}}_{\text{want small}}$$

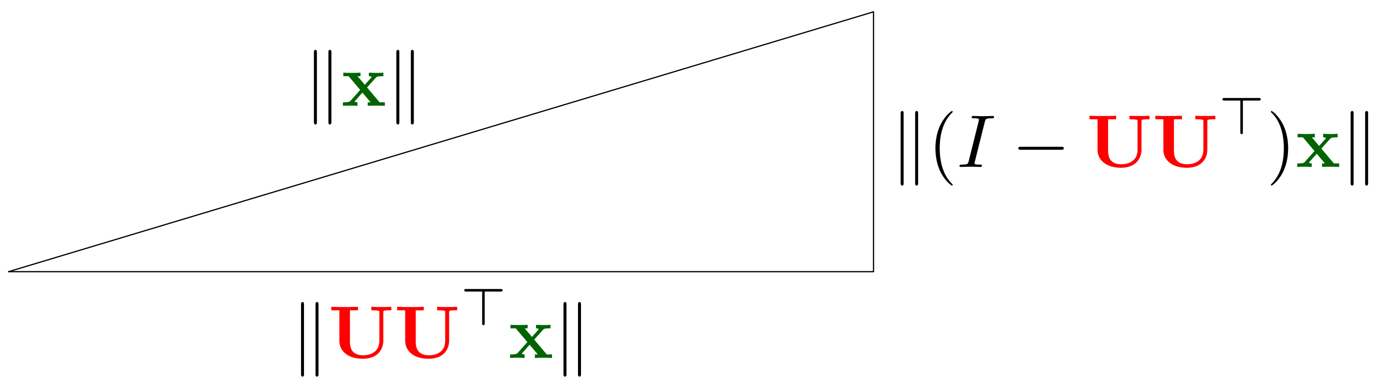
Pythagorean decomposition: $\mathbf{x} = \mathbf{U}\mathbf{U}^\top \mathbf{x} + (I - \mathbf{U}\mathbf{U}^\top)\mathbf{x}$

Equivalence in two objectives

Key intuition:

$$\underbrace{\text{variance of data}}_{\text{fixed}} = \underbrace{\text{captured variance}}_{\text{want large}} + \underbrace{\text{reconstruction error}}_{\text{want small}}$$

Pythagorean decomposition: $\mathbf{x} = \mathbf{U}\mathbf{U}^\top \mathbf{x} + (I - \mathbf{U}\mathbf{U}^\top)\mathbf{x}$

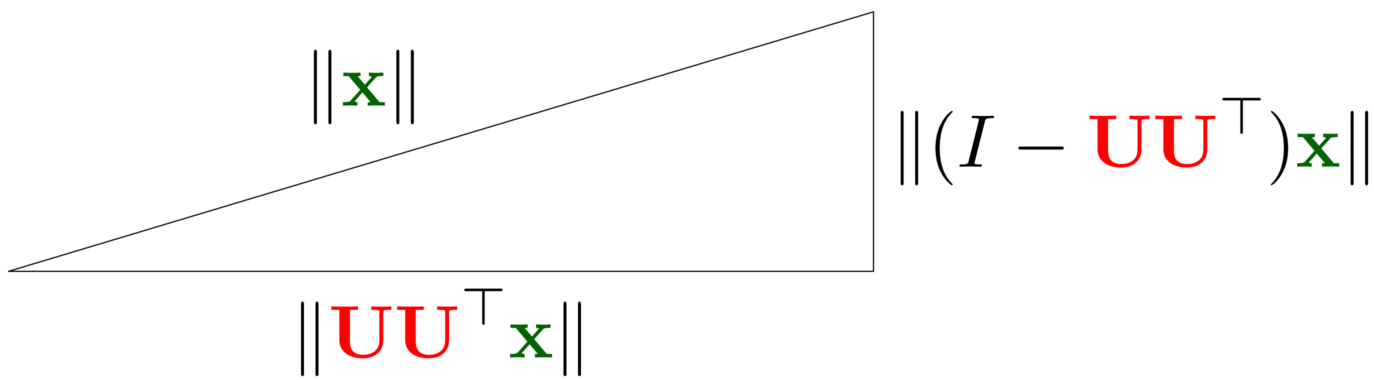


Equivalence in two objectives

Key intuition:

$$\underbrace{\text{variance of data}}_{\text{fixed}} = \underbrace{\text{captured variance}}_{\text{want large}} + \underbrace{\text{reconstruction error}}_{\text{want small}}$$

Pythagorean decomposition: $\mathbf{x} = \mathbf{U}\mathbf{U}^\top\mathbf{x} + (I - \mathbf{U}\mathbf{U}^\top)\mathbf{x}$



Take expectations; note rotation $\mathbf{U}$ doesn't affect length:

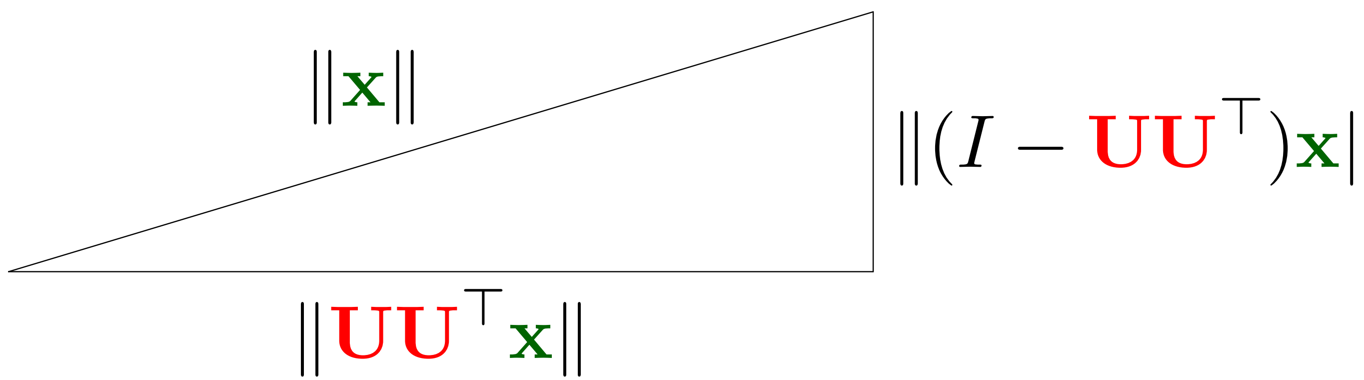
$$\hat{\mathbb{E}}[\|\mathbf{x}\|^2] = \hat{\mathbb{E}}[\|\mathbf{U}^\top\mathbf{x}\|^2] + \hat{\mathbb{E}}[\|\mathbf{x} - \mathbf{U}\mathbf{U}^\top\mathbf{x}\|^2]$$

Equivalence in two objectives

Key intuition:

$$\underbrace{\text{variance of data}}_{\text{fixed}} = \underbrace{\text{captured variance}}_{\text{want large}} + \underbrace{\text{reconstruction error}}_{\text{want small}}$$

Pythagorean decomposition: $\mathbf{x} = \mathbf{U}\mathbf{U}^\top \mathbf{x} + (I - \mathbf{U}\mathbf{U}^\top)\mathbf{x}$

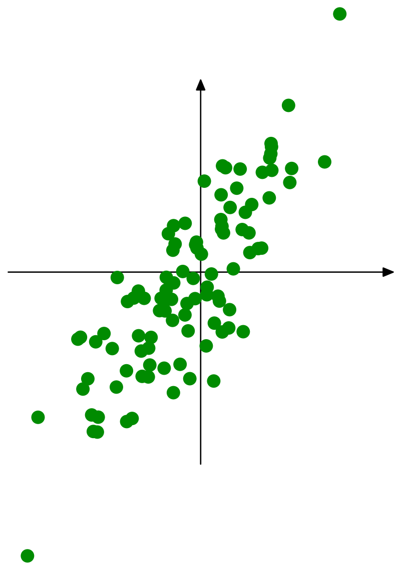


Take expectations; note rotation $\mathbf{U}$ doesn't affect length:

$$\hat{\mathbb{E}}[\|\mathbf{x}\|^2] = \hat{\mathbb{E}}[\|\mathbf{U}^\top \mathbf{x}\|^2] + \hat{\mathbb{E}}[\|\mathbf{x} - \mathbf{U}\mathbf{U}^\top \mathbf{x}\|^2]$$

Minimize reconstruction error $\leftrightarrow$ Maximize captured variance

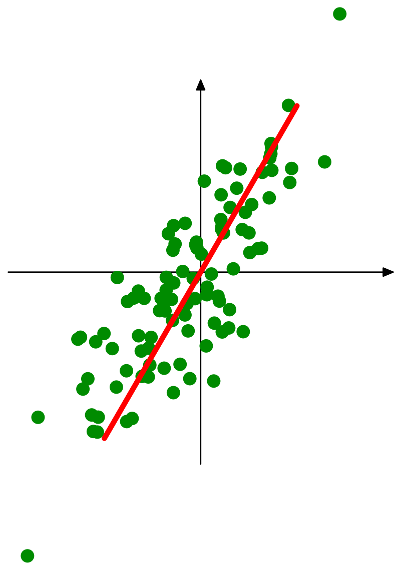
Finding one principal component



Input data:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \ldots & \mathbf{x}_n \\ | & & | \end{pmatrix}$$

Finding one principal component

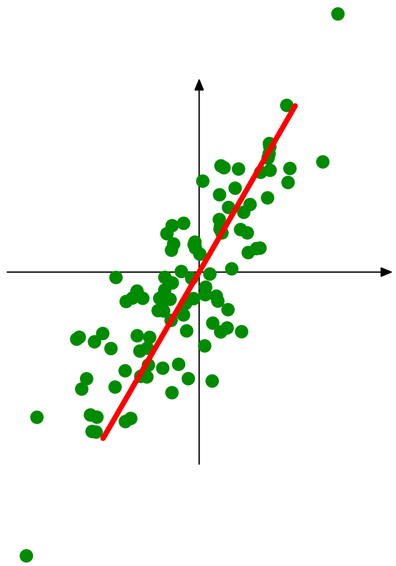


Objective: maximize variance of projected data

Input data:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \dots & \mathbf{x}_n \\ | & & | \end{pmatrix}$$

Finding one principal component



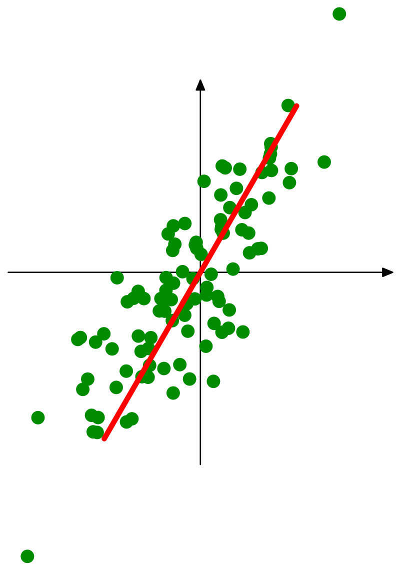
Objective: maximize variance of projected data

$$= \max_{\|\mathbf{u}\|=1} \hat{\mathbb{E}}[(\mathbf{u}^\top \mathbf{x})^2]$$

Input data:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \ldots & \mathbf{x}_n \\ | & & | \end{pmatrix}$$

Finding one principal component



Objective: maximize variance of projected data

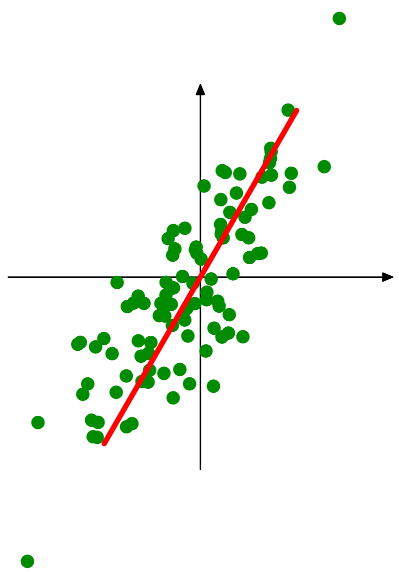
$$= \max_{\|\mathbf{u}\|=1} \hat{\mathbb{E}}[(\mathbf{u}^\top \mathbf{x})^2]$$

$$= \max_{\|\mathbf{u}\|=1} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{u}^\top \mathbf{x}_i)^2$$

Input data:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \ldots & \mathbf{x}_n \\ | & & | \end{pmatrix}$$

Finding one principal component



Objective: maximize variance of projected data

$$= \max_{\|\mathbf{u}\|=1} \hat{\mathbb{E}}[(\mathbf{u}^\top \mathbf{x})^2]$$

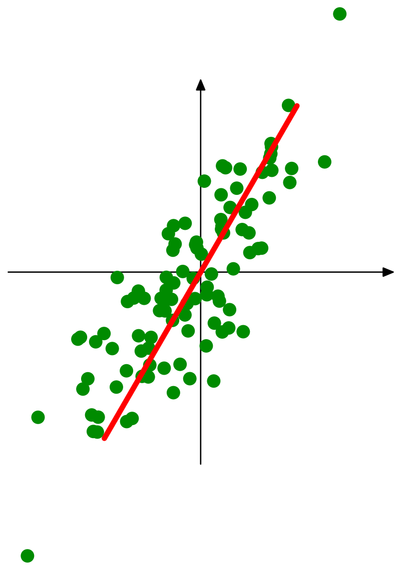
$$= \max_{\|\mathbf{u}\|=1} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{u}^\top \mathbf{x}_i)^2$$

$$= \max_{\|\mathbf{u}\|=1} \frac{1}{n} \|\mathbf{u}^\top \mathbf{X}\|^2$$

Input data:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \dots & \mathbf{x}_n \\ | & & | \end{pmatrix}$$

Finding one principal component



Objective: maximize variance of projected data

$$= \max_{\|\mathbf{u}\|=1} \hat{\mathbb{E}}[(\mathbf{u}^\top \mathbf{x})^2]$$

$$= \max_{\|\mathbf{u}\|=1} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{u}^\top \mathbf{x}_i)^2$$

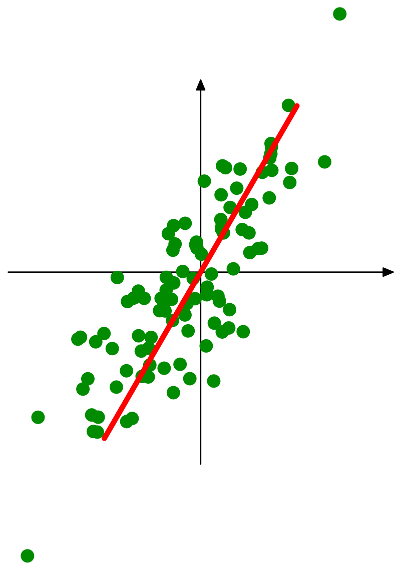
$$= \max_{\|\mathbf{u}\|=1} \frac{1}{n} \|\mathbf{u}^\top \mathbf{X}\|^2$$

$$= \max_{\|\mathbf{u}\|=1} \mathbf{u}^\top \left(\frac{1}{n} \mathbf{X}\mathbf{X}^\top \right) \mathbf{u}$$

Input data:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots & \mathbf{x}_n \\ | & & | \end{pmatrix}$$

Finding one principal component



Input data:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \dots & \mathbf{x}_n \\ | & & | \end{pmatrix}$$

Objective: maximize variance of projected data

$$= \max_{\|\mathbf{u}\|=1} \hat{\mathbb{E}}[(\mathbf{u}^\top \mathbf{x})^2]$$

$$= \max_{\|\mathbf{u}\|=1} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{u}^\top \mathbf{x}_i)^2$$

$$= \max_{\|\mathbf{u}\|=1} \frac{1}{n} \|\mathbf{u}^\top \mathbf{X}\|^2$$

$$= \max_{\|\mathbf{u}\|=1} \mathbf{u}^\top \left(\frac{1}{n} \mathbf{X}\mathbf{X}^\top \right) \mathbf{u}$$

$=$ largest eigenvalue of $C \stackrel{\text{def}}{=} \frac{1}{n}\mathbf{X}\mathbf{X}^\top$

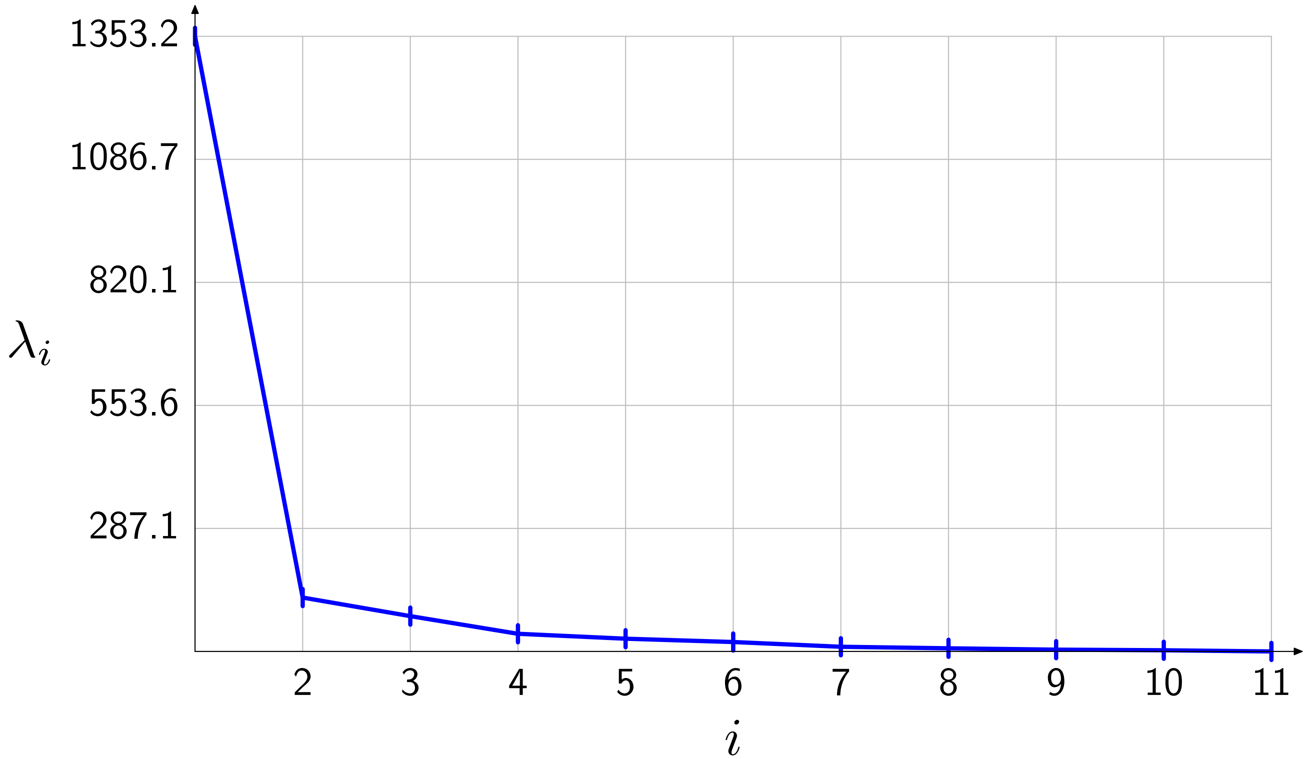
(C is covariance matrix of data)

How many principal components?

- Similar to question of "How many clusters?"
- Magnitude of eigenvalues indicate fraction of variance captured.

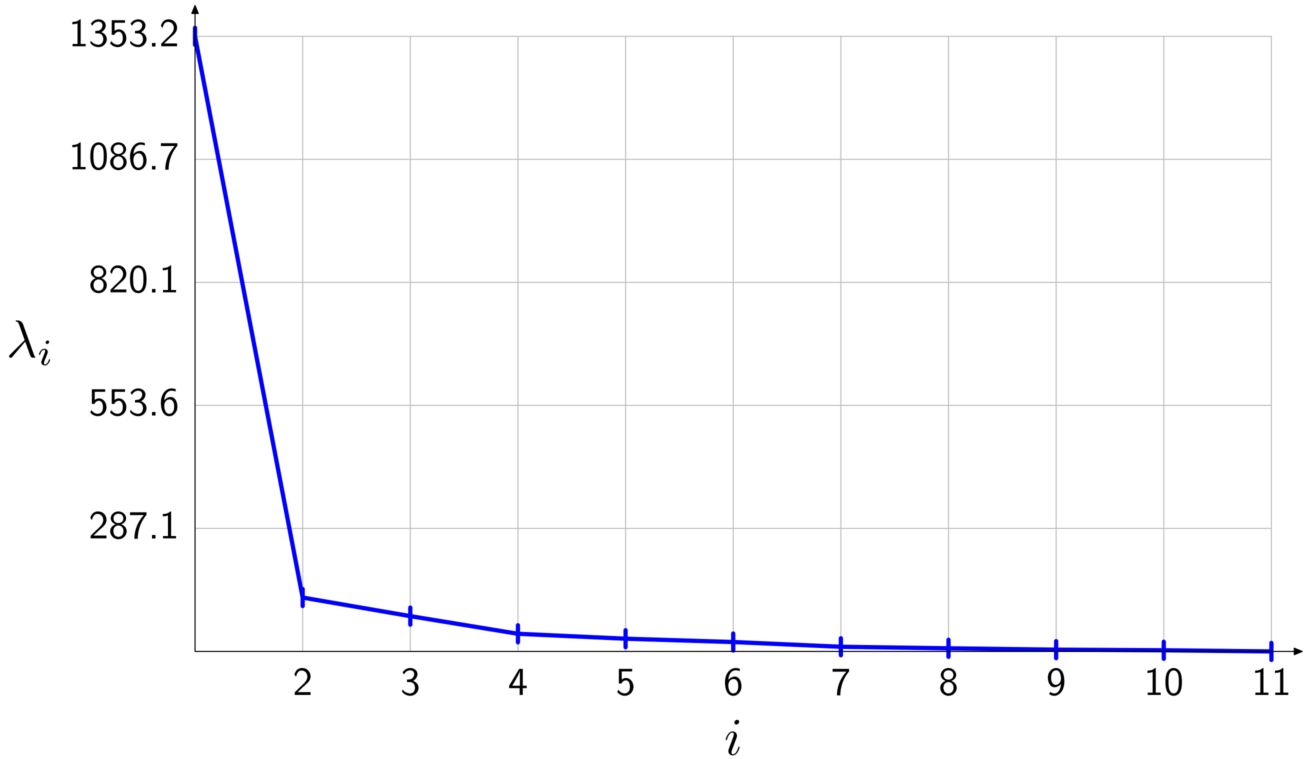
How many principal components?

- Similar to question of "How many clusters?"
- Magnitude of eigenvalues indicate fraction of variance captured.
- Eigenvalues on a face image dataset:



How many principal components?

- Similar to question of "How many clusters?"
- Magnitude of eigenvalues indicate fraction of variance captured.
- Eigenvalues on a face image dataset:



- Eigenvalues typically drop off sharply, so don't need that many.
- Of course variance isn't everything...

Computing PCA

Method 1: eigendecomposition

$\mathbf{U}$ are eigenvectors of covariance matrix $C = \frac{1}{n}\mathbf{X}\mathbf{X}^\top$

Computing C already takes $O(nd^2)$ time (very expensive)

Computing PCA

Method 1: eigendecomposition

$\mathbf{U}$ are eigenvectors of covariance matrix $C = \frac{1}{n}\mathbf{X}\mathbf{X}^\top$

Computing C already takes $O(nd^2)$ time (very expensive)

Method 2: singular value decomposition (SVD)

Find $\mathbf{X} = \mathbf{U}_{d\times d}\Sigma_{d\times n}\mathbf{V}_{n\times n}^\top$

where $\mathbf{U}^\top\mathbf{U} = I_{d\times d}$, $\mathbf{V}^\top\mathbf{V} = I_{n\times n}$, Σ is diagonal

Computing top k singular vectors takes only $O(ndk)$

Computing PCA

Method 1: eigendecomposition

$\mathbf{U}$ are eigenvectors of covariance matrix $C = \frac{1}{n}\mathbf{X}\mathbf{X}^\top$

Computing C already takes $O(nd^2)$ time (very expensive)

Method 2: singular value decomposition (SVD)

Find $\mathbf{X} = \mathbf{U}_{d\times d}\Sigma_{d\times n}\mathbf{V}_{n\times n}^\top$

where $\mathbf{U}^\top\mathbf{U} = I_{d\times d}$, $\mathbf{V}^\top\mathbf{V} = I_{n\times n}$, Σ is diagonal

Computing top k singular vectors takes only $O(ndk)$

Relationship between eigendecomposition and SVD:

Left singular vectors are principal components ($C = \mathbf{U}\Sigma^2\mathbf{U}^\top$)

Roadmap

- Principal component analysis (PCA)
  - Basic principles
  - Case studies
  - Kernel PCA
  - Probabilistic PCA
- Canonical correlation analysis (CCA)
- Fisher discriminant analysis (FDA)
- Summary

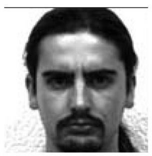
Eigen-faces [Turk and Pentland, 1991]

- d = number of pixels
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a face image
- $\mathbf{x}_{ji}$ = intensity of the j-th pixel in image i

Eigen-faces [Turk and Pentland, 1991]

- d = number of pixels
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a face image
- $\mathbf{x}_{ji}$ = intensity of the j-th pixel in image i

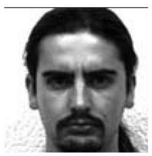
$$\mathbf{X}_{d\times n} \approx \mathbf{U}_{d\times k} \quad \mathbf{Z}_{k\times n}$$

( $\cdots$ ) $\approx$ () $\begin{pmatrix} | & & | \\ \mathbf{z}_1 & \cdots & \mathbf{z}_n \\ | & & | \end{pmatrix}$

Eigen-faces [Turk and Pentland, 1991]

- d = number of pixels
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a face image
- $\mathbf{x}_{ji}$ = intensity of the j-th pixel in image i

$$\mathbf{X}_{d \times n} \quad \cong \quad \mathbf{U}_{d \times k} \quad \mathbf{Z}_{k \times n}$$

( $\cdots$ ) $\cong$ () $\begin{pmatrix} | & & | \\ \mathbf{z}_1 & \cdots & \mathbf{z}_n \\ | & & | \end{pmatrix}$

Idea: $\mathbf{z}_i$ more "meaningful" representation of i-th face than $\mathbf{x}_i$

Can use $\mathbf{z}_i$ for nearest-neighbor classification

Much faster: $O(dk + nk)$ time instead of $O(dn)$ when $n, d \gg k$

Why no time savings for linear classifier?

Latent Semantic Analysis [Deerwater, 1990]

- d = number of words in the vocabulary
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a vector of word counts
- $\mathbf{x}_{ji}$ = frequency of word j in document i

$$\mathbf{X}_{d\times n} \approxeq \mathbf{U}_{d\times k} \quad \mathbf{Z}_{k\times n}$$

$$\begin{pmatrix} \text{stocks: } 2 & \cdots & 0 \\ \text{chairman: } 4 & \cdots & 1 \\ \text{the: } 8 & \cdots & 7 \\ \cdots\ \vdots & \cdots & \vdots \\ \text{wins: } 0 & \cdots & 2 \\ \text{game: } 1 & \cdots & 3 \end{pmatrix} \approxeq \begin{pmatrix} 0.4 & \cdots & \text{-}0.001 \\ 0.8 & \cdots & 0.03 \\ 0.01 & \cdots & 0.04 \\ \vdots & \cdots & \vdots \\ 0.002 & \cdots & 2.3 \\ 0.003 & \cdots & 1.9 \end{pmatrix} \begin{pmatrix} | & & | \\ \mathbf{z}_1 & \ldots & \mathbf{z}_n \\ | & & | \end{pmatrix}$$

Latent Semantic Analysis [Deerwater, 1990]

- d = number of words in the vocabulary
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a vector of word counts
- $\mathbf{x}_{ji}$ = frequency of word j in document i

$$\underset{\mathbf{X}_{d\times n}}{\begin{pmatrix} \text{stocks: } 2 & \cdots\cdots & 0 \\ \text{chairman: } 4 & \cdots\cdots & 1 \\ \text{the: } 8 & \cdots\cdots & 7 \\ \cdots\ \vdots & \cdots\cdots & \vdots \\ \text{wins: } 0 & \cdots\cdots & 2 \\ \text{game: } 1 & \cdots\cdots & 3 \end{pmatrix}} \approxeq \underset{\mathbf{U}_{d\times k}}{\begin{pmatrix} 0.4 & \cdot\cdot & \text{-}0.001 \\ 0.8 & \cdot\cdot & 0.03 \\ 0.01 & \cdot\cdot & 0.04 \\ \vdots & \cdot\cdot & \vdots \\ 0.002 & \cdot\cdot & 2.3 \\ 0.003 & \cdot\cdot & 1.9 \end{pmatrix}} \underset{\mathbf{Z}_{k\times n}}{\begin{pmatrix} | & & | \\ \mathbf{z}_1 & \ldots & \mathbf{z}_n \\ | & & | \end{pmatrix}}$$

How to measure similarity between two documents?

$\mathbf{z}_1^\top \mathbf{z}_2$ is probably better than $\mathbf{x}_1^\top \mathbf{x}_2$

Latent Semantic Analysis [Deerwater, 1990]

- d = number of words in the vocabulary
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a vector of word counts
- $\mathbf{x}_{ji}$ = frequency of word j in document i

$$\mathbf{X}_{d\times n} \quad \approxeq \quad \mathbf{U}_{d\times k} \quad \mathbf{Z}_{k\times n}$$

$$\left(\begin{array}{rlr} \text{stocks:} & 2 \cdots\cdots\cdots & 0 \\ \text{chairman:} & 4 \cdots\cdots\cdots & 1 \\ \text{the:} & 8 \cdots\cdots\cdots & 7 \\ \cdots & \vdots \cdots\cdots\cdots & \vdots \\ \text{wins:} & 0 \cdots\cdots\cdots & 2 \\ \text{game:} & 1 \cdots\cdots\cdots & 3 \end{array}\right) \approxeq \left(\begin{array}{rcc} 0.4 & \cdots & -0.001 \\ 0.8 & \cdots & 0.03 \\ 0.01 & \cdots & 0.04 \\ \vdots & \cdots & \vdots \\ 0.002 & \cdots & 2.3 \\ 0.003 & \cdots & 1.9 \end{array}\right) \left(\begin{array}{ccc} | & & | \\ \mathbf{z}_1 & \ldots & \mathbf{z}_n \\ | & & | \end{array}\right)$$

How to measure similarity between two documents?

$$\mathbf{z}_1^\top \mathbf{z}_2 \text{ is probably better than } \mathbf{x}_1^\top \mathbf{x}_2$$

Applications: information retrieval

Latent Semantic Analysis [Deerwater, 1990]

- d = number of words in the vocabulary
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a vector of word counts
- $\mathbf{x}_{ji}$ = frequency of word j in document i

$$\mathbf{X}_{d\times n} \approx \mathbf{U}_{d\times k} \quad \mathbf{Z}_{k\times n}$$

$$\begin{pmatrix} \text{stocks: } 2 & \cdots & 0 \\ \text{chairman: } 4 & \cdots & 1 \\ \text{the: } 8 & \cdots & 7 \\ \cdots\ \vdots & \cdots & \vdots \\ \text{wins: } 0 & \cdots & 2 \\ \text{game: } 1 & \cdots & 3 \end{pmatrix} \approx \begin{pmatrix} 0.4 & \cdots & -0.001 \\ 0.8 & \cdots & 0.03 \\ 0.01 & \cdots & 0.04 \\ \vdots & \cdots & \vdots \\ 0.002 & \cdots & 2.3 \\ 0.003 & \cdots & 1.9 \end{pmatrix} \begin{pmatrix} | & & | \\ \mathbf{z}_1 & \ldots & \mathbf{z}_n \\ | & & | \end{pmatrix}$$

How to measure similarity between two documents?

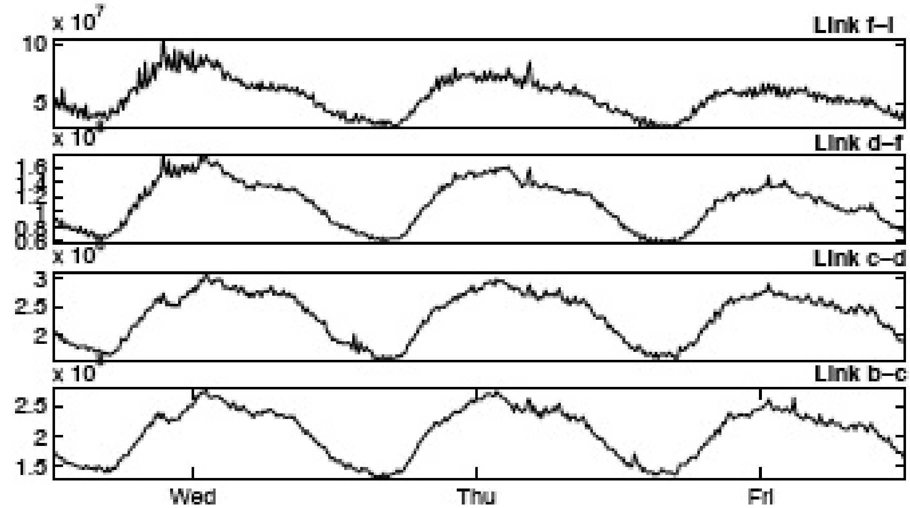
$\mathbf{z}_1^\top \mathbf{z}_2$ is probably better than $\mathbf{x}_1^\top \mathbf{x}_2$

Applications: information retrieval

Note: no computational savings; original $\mathbf{x}$ is already sparse

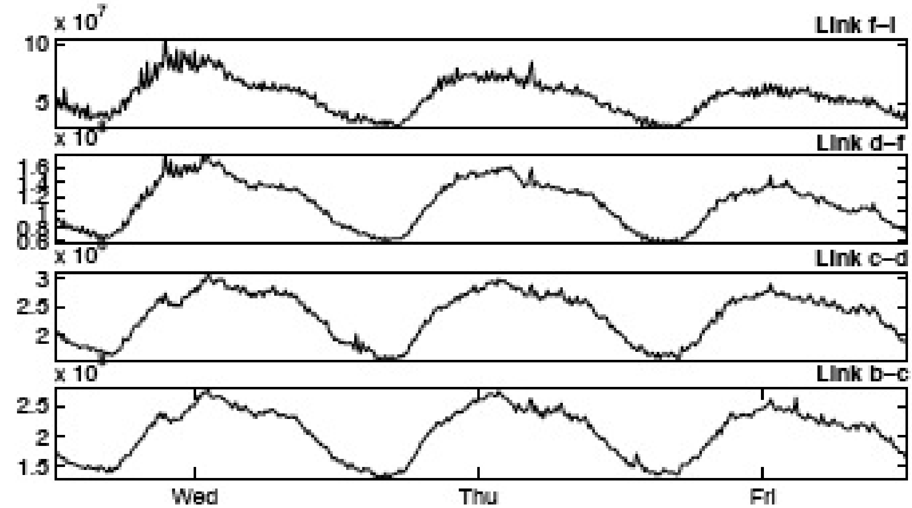
Network anomaly detection [Lakhina, '05]

$\mathbf{x}_{ji}$ = amount of traffic on link j in the network during each time interval i



Network anomaly detection [Lakhina, '05]

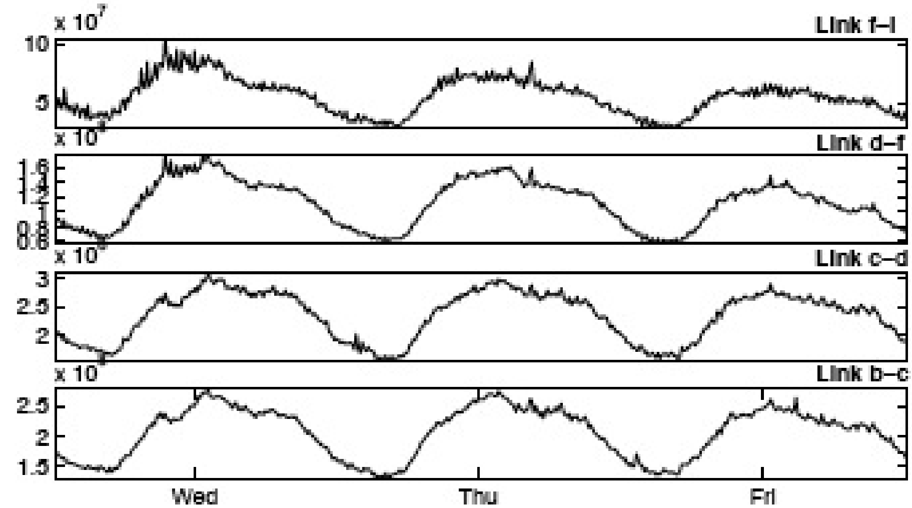
$\mathbf{x}_{ji}$ = amount of traffic on link j in the network during each time interval i



Model assumption: total traffic is sum of flows along a few "paths"

Network anomaly detection [Lakhina, '05]

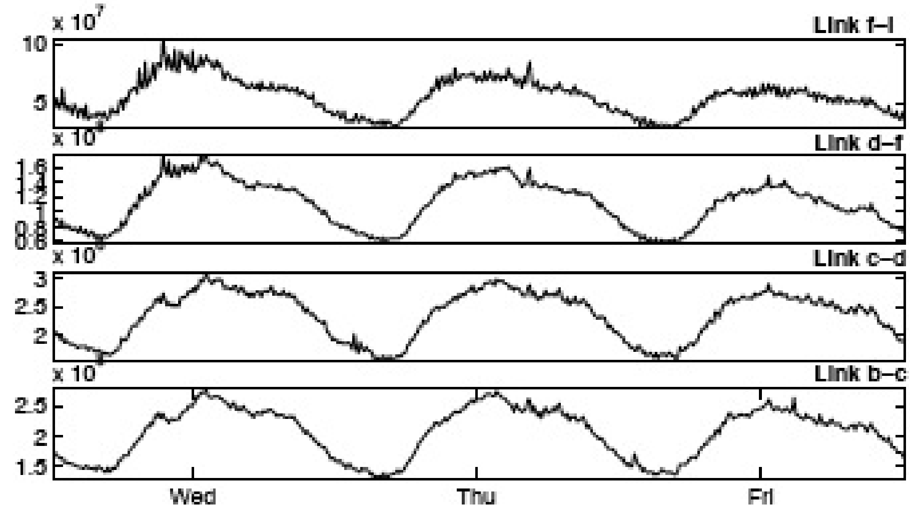
$\mathbf{x}_{ji}$ = amount of traffic on link j in the network during each time interval i



Model assumption: total traffic is sum of flows along a few "paths"
Apply PCA: each principal component intuitively represents a "path"

Network anomaly detection [Lakhina, '05]

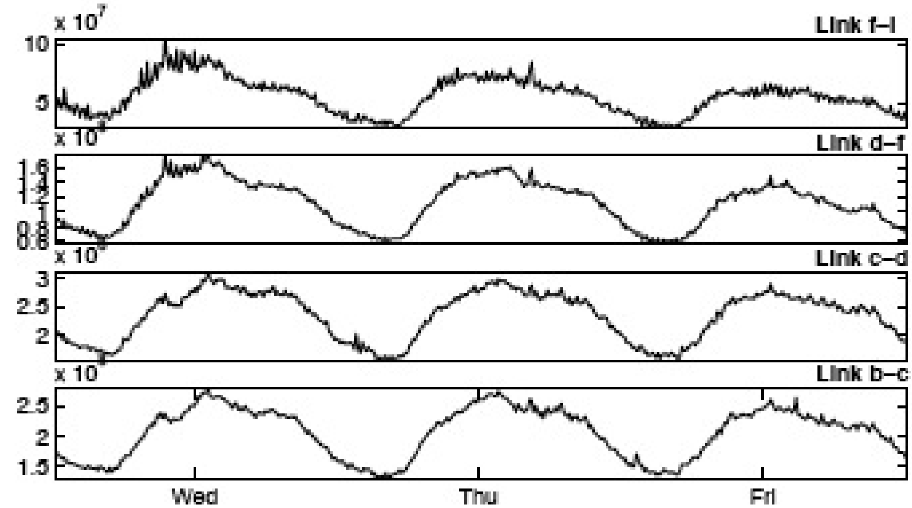
$\mathbf{x}_{ji}$ = amount of traffic on link j in the network during each time interval i



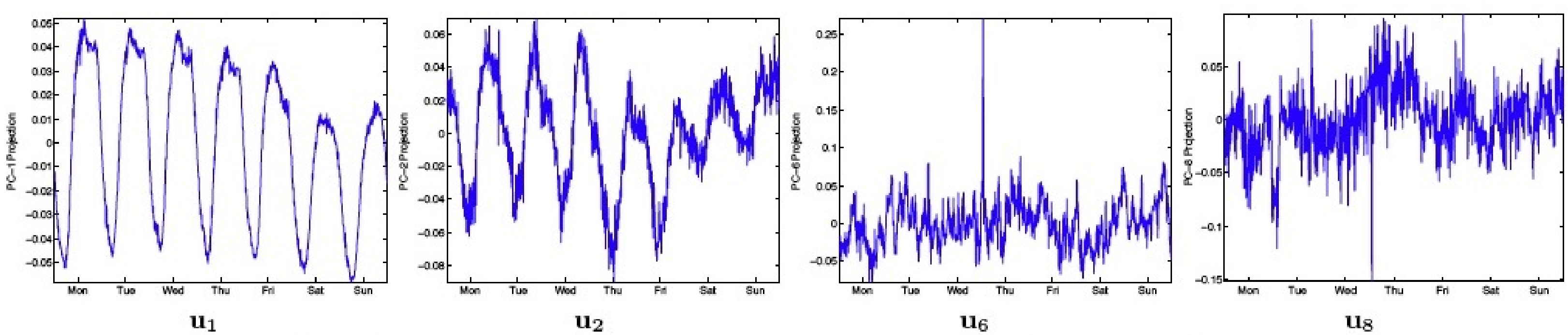
Model assumption: total traffic is sum of flows along a few "paths"
Apply PCA: each principal component intuitively represents a "path"
Anomaly when traffic deviates from first few principal components

Network anomaly detection [Lakhina, '05]

$\mathbf{x}_{ji}$ = amount of traffic on link j in the network during each time interval i



Model assumption: total traffic is sum of flows along a few "paths"
Apply PCA: each principal component intuitively represents a "path"
Anomaly when traffic deviates from first few principal components



(a) Normal Behavior (b) Anomalous Behavior

Unsupervised POS tagging [Schütze, '95]

Part-of-speech (POS) tagging task:

Input:	I	like	reducing	the	dimensionality	of	data	.
Output:	NOUN	VERB	VERB(-ING)	DET	NOUN	PREP	NOUN	.

Unsupervised POS tagging [Schütze, '95]

Part-of-speech (POS) tagging task:

Input:	I	like	reducing	the	dimensionality	of	data	.
Output:	NOUN	VERB	VERB(-ING)	DET	NOUN	PREP	NOUN	.

Each $\mathbf{x}_i$ is (the context distribution of) a word.

$\mathbf{x}_{ji}$ is number of times word i appeared in context j

Key idea: words appearing in similar contexts
tend to have the same POS tags;
so cluster using the contexts of each word type

Problem: contexts are too sparse

Unsupervised POS tagging [Schütze, '95]

Part-of-speech (POS) tagging task:

Input:	I	like	reducing	the	dimensionality	of	data	.
Output:	NOUN	VERB	VERB(-ING)	DET	NOUN	PREP	NOUN	.

Each $\mathbf{x}_i$ is (the context distribution of) a word.

$\mathbf{x}_{ji}$ is number of times word i appeared in context j

Key idea: words appearing in similar contexts
tend to have the same POS tags;
so cluster using the contexts of each word type

Problem: contexts are too sparse

Solution: run PCA first,
then cluster using new representation

Multi-task learning [Ando & Zhang, '05]

- Have n related tasks (classify documents for various users)
- Each task has a linear classifier with weights $\mathbf{x}_i$
- Want to share structure between classifiers

Multi-task learning [Ando & Zhang, '05]

- Have n related tasks (classify documents for various users)
- Each task has a linear classifier with weights $\mathbf{x}_i$
- Want to share structure between classifiers

One step of their procedure:
given n linear classifiers $\mathbf{x}_1, \ldots, \mathbf{x}_n$,
run PCA to identify shared structure:

Multi-task learning [Ando & Zhang, '05]

- Have n related tasks (classify documents for various users)
- Each task has a linear classifier with weights $\mathbf{x}_i$
- Want to share structure between classifiers

One step of their procedure:
given n linear classifiers $\mathbf{x}_1, \ldots, \mathbf{x}_n$,
run PCA to identify shared structure:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \ldots & \mathbf{x}_n \\ | & & | \end{pmatrix} \cong \mathbf{U}\mathbf{Z}$$

Multi-task learning [Ando & Zhang, '05]

- Have n related tasks (classify documents for various users)
- Each task has a linear classifier with weights $\mathbf{x}_i$
- Want to share structure between classifiers

One step of their procedure:
given n linear classifiers $\mathbf{x}_1, \dots, \mathbf{x}_n$,
run PCA to identify shared structure:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \dots & \mathbf{x}_n \\ | & & | \end{pmatrix} \approxeq \mathbf{U}\mathbf{Z}$$

Each principal component is a eigen-classifier

Multi-task learning [Ando & Zhang, '05]

- Have n related tasks (classify documents for various users)
- Each task has a linear classifier with weights $\mathbf{x}_i$
- Want to share structure between classifiers

One step of their procedure:
given n linear classifiers $\mathbf{x}_1, \dots, \mathbf{x}_n$,
run PCA to identify shared structure:

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \dots & \mathbf{x}_n \\ | & & | \end{pmatrix} \approxeq \mathbf{U}\mathbf{Z}$$

Each principal component is a eigen-classifier

Other step of their procedure:
Retrain classifiers, regularizing towards subspace $\mathbf{U}$

PCA summary

- Intuition: capture variance of data or minimize reconstruction error

PCA summary

- Intuition: capture variance of data or minimize reconstruction error
- Algorithm: find eigendecomposition of covariance matrix or SVD

PCA summary

- Intuition: capture variance of data or minimize reconstruction error
- Algorithm: find eigendecomposition of covariance matrix or SVD
- Impact: reduce storage (from $O(nd)$ to $O(nk)$), reduce time complexity

PCA summary

- Intuition: capture variance of data or minimize reconstruction error
- Algorithm: find eigendecomposition of covariance matrix or SVD
- Impact: reduce storage (from $O(nd)$ to $O(nk)$), reduce time complexity
- Advantages: simple, fast

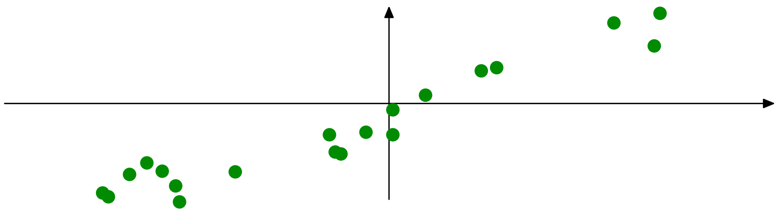
PCA summary

- **Intuition**: capture variance of data or minimize reconstruction error
- **Algorithm**: find eigendecomposition of covariance matrix or SVD
- **Impact**: reduce storage (from $O(nd)$ to $O(nk)$), reduce time complexity
- **Advantages**: simple, fast
- **Applications**: eigen-faces, eigen-documents, network anomaly detection, etc.

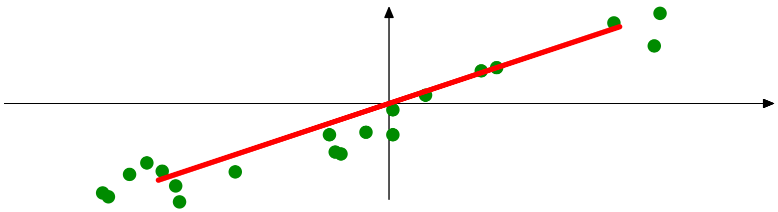
Roadmap

- Principal component analysis (PCA)
  - Basic principles
  - Case studies
  - Kernel PCA
  - Probabilistic PCA
- Canonical correlation analysis (CCA)
- Fisher discriminant analysis (FDA)
- Summary

Limitations of linearity

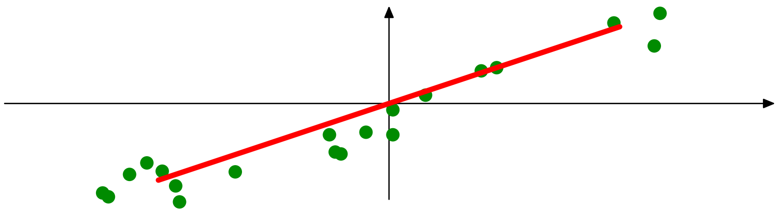


Limitations of linearity

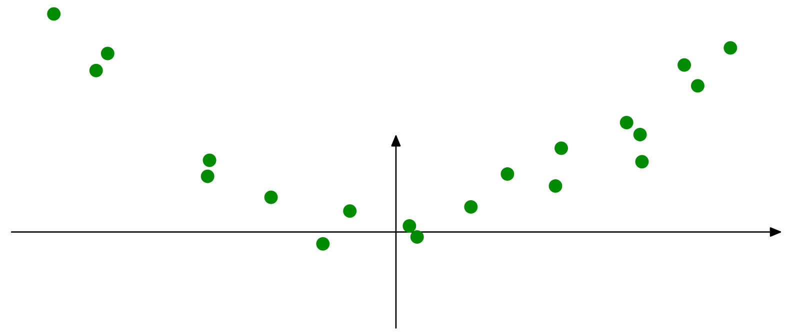


PCA is effective

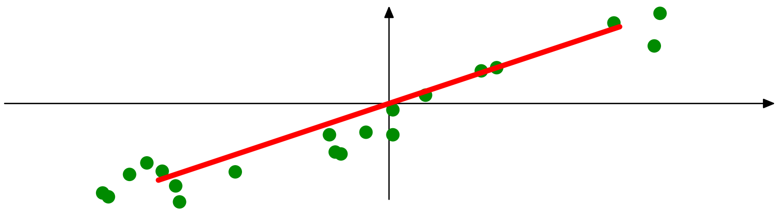
Limitations of linearity



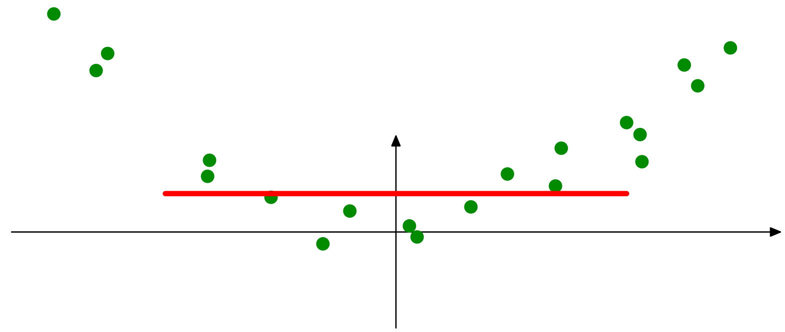
PCA is effective



Limitations of linearity

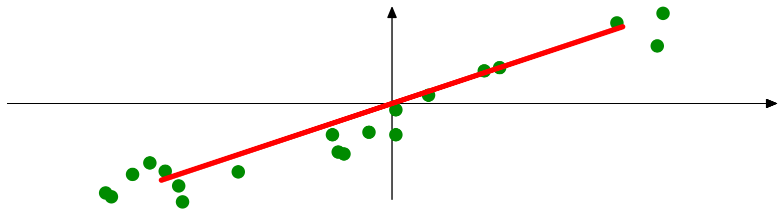


PCA is effective

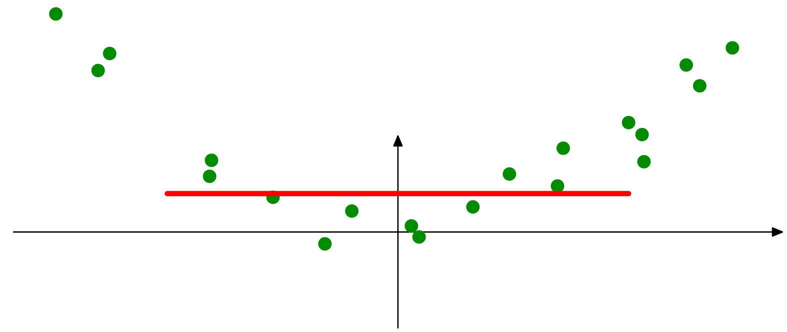


PCA is ineffective

Limitations of linearity



PCA is effective

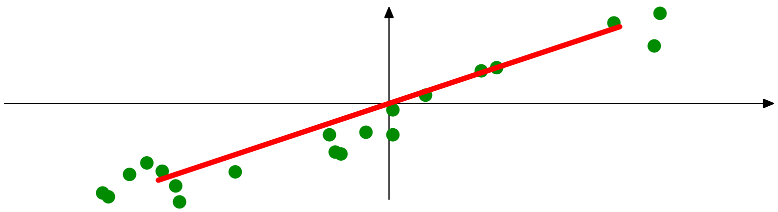


PCA is ineffective

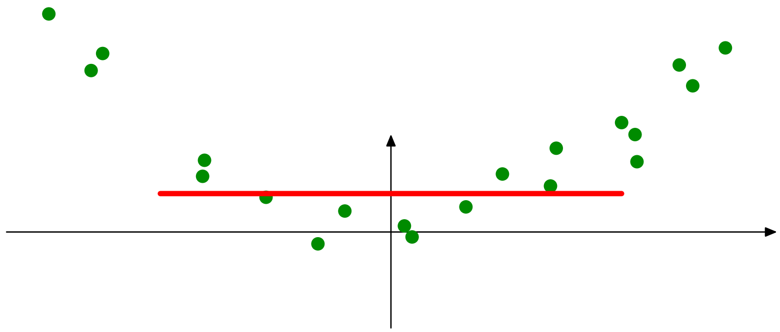
Problem is that PCA subspace is linear:

$$S = \{\mathbf{x} = \mathbf{U}\mathbf{z} : \mathbf{z} \in \mathbb{R}^k\}$$

Limitations of linearity



PCA is effective



PCA is ineffective

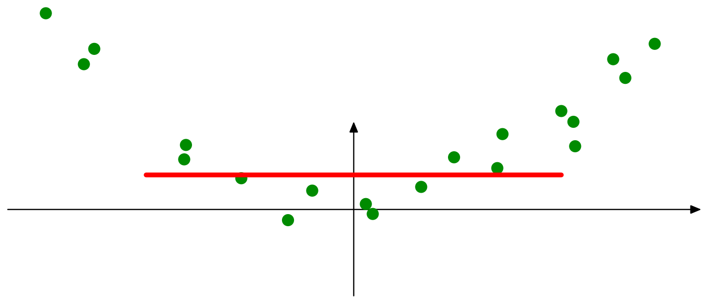
Problem is that PCA subspace is linear:

$$S = \{\mathbf{x} = \mathbf{U}\mathbf{z} : \mathbf{z} \in \mathbb{R}^k\}$$

In this example:

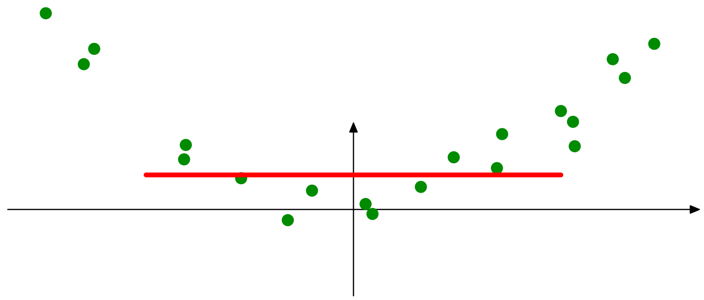
$$S = \{(x_1, x_2) : x_2 = \tfrac{u_2}{u_1} x_1\}$$

Going beyond linearity: quick solution

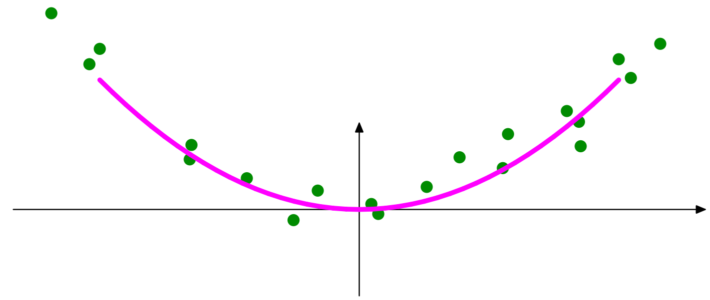


Broken solution

Going beyond linearity: quick solution



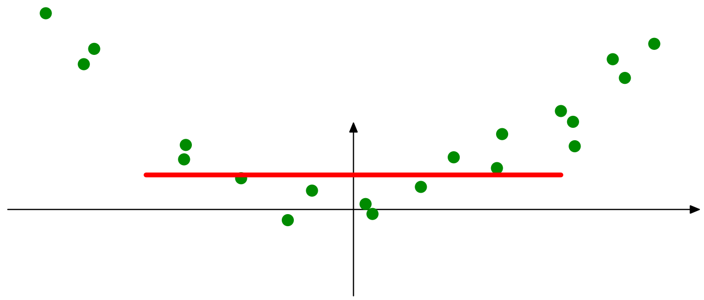
Broken solution



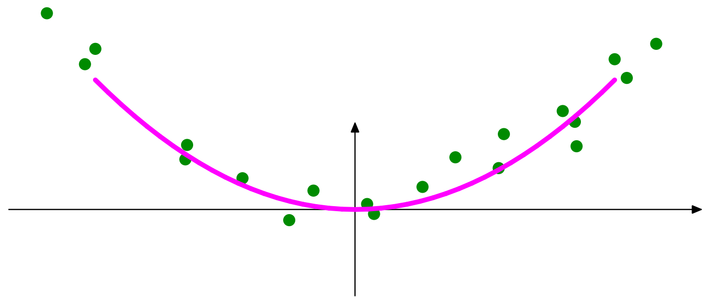
Desired solution

We want desired solution: $S = \{(x_1, x_2) : x_2 = \frac{u_2}{u_1} x_1^2\}$

Going beyond linearity: quick solution



Broken solution

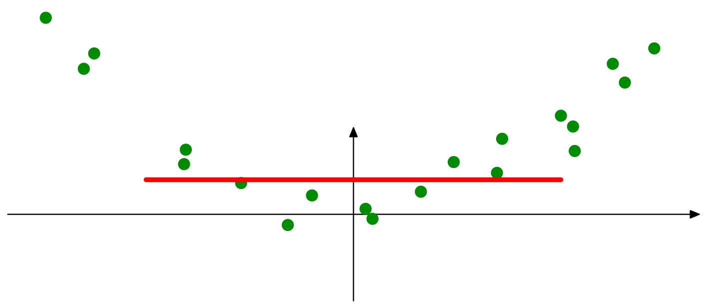


Desired solution

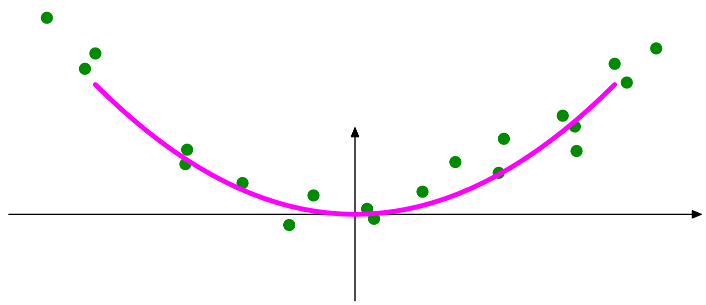
We want desired solution: $S = \{(x_1, x_2) : x_2 = \frac{u_2}{u_1} x_1^2\}$

We can get this: $S = \{\phi(\mathbf{x}) = \mathbf{U}\mathbf{z}\}$ with $\phi(\mathbf{x}) = (x_1^2, x_2)^\top$

Going beyond linearity: quick solution



Broken solution



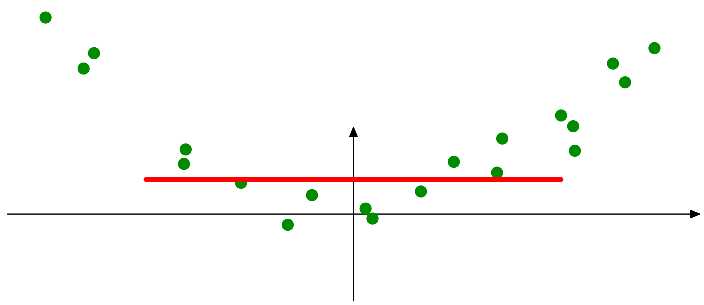
Desired solution

We want desired solution: $S = \{(x_1, x_2) : x_2 = \frac{u_2}{u_1} x_1^2\}$

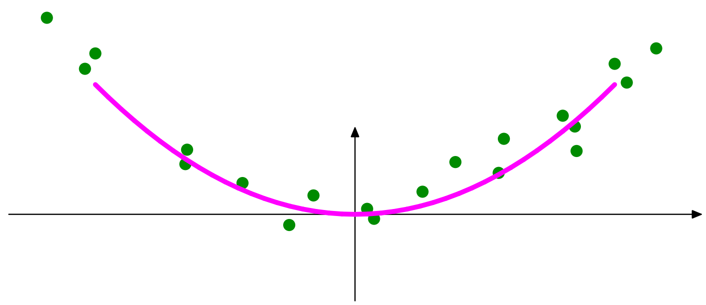
We can get this: $S = \{\phi(\mathbf{x}) = \mathbf{U}\mathbf{z}\}$ with $\phi(\mathbf{x}) = (x_1^2, x_2)^\top$

Linear dimensionality reduction in $\phi(\mathbf{x})$ space $\Updownarrow$ Nonlinear dimensionality reduction in $\mathbf{x}$ space

Going beyond linearity: quick solution



Broken solution



Desired solution

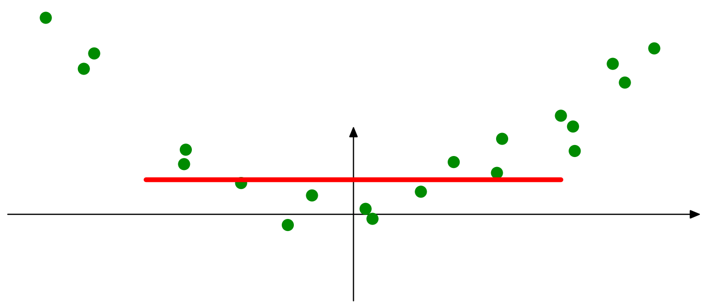
We want desired solution: $S = \{(x_1, x_2) : x_2 = \frac{u_2}{u_1} x_1^2\}$

We can get this: $S = \{\phi(\mathbf{x}) = \mathbf{U}\mathbf{z}\}$ with $\phi(\mathbf{x}) = (x_1^2, x_2)^\top$

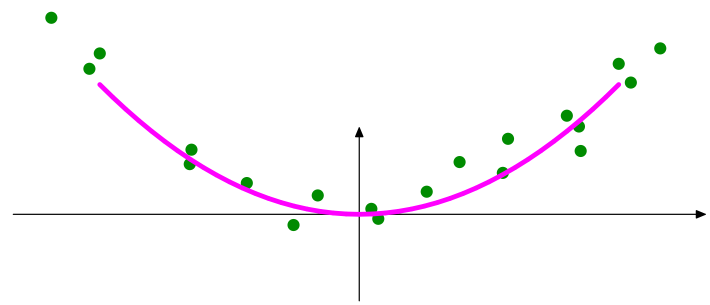
Linear dimensionality reduction in $\phi(\mathbf{x})$ space
$\Updownarrow$
Nonlinear dimensionality reduction in $\mathbf{x}$ space

In general, can set $\phi(\mathbf{x}) = (x_1, x_1^2, x_1 x_2, \sin(x_1), \dots)^\top$

Going beyond linearity: quick solution



Broken solution



Desired solution

We want desired solution: $S = \{(x_1, x_2) : x_2 = \frac{u_2}{u_1} x_1^2\}$

We can get this: $S = \{\phi(\mathbf{x}) = \mathbf{U}\mathbf{z}\}$ with $\phi(\mathbf{x}) = (x_1^2, x_2)^\top$

Linear dimensionality reduction in $\phi(\mathbf{x})$ space
$\Updownarrow$
Nonlinear dimensionality reduction in $\mathbf{x}$ space

In general, can set $\phi(\mathbf{x}) = (x_1, x_1^2, x_1 x_2, \sin(x_1), \dots)^\top$

Problems: (1) ad-hoc and tedious
(2) $\phi(\mathbf{x})$ large, computationally expensive

Towards kernels

Representer theorem:

PCA solution is linear combination of $\mathbf{x}_i$s

Towards kernels

Representer theorem:

PCA solution is linear combination of $\mathbf{x}_i$s

Why?

Recall PCA eigenvalue problem: $\mathbf{X}\mathbf{X}^\top \mathbf{u} = \lambda \mathbf{u}$

Towards kernels

Representer theorem:

PCA solution is linear combination of $\mathbf{x}_i$s

Why?

Recall PCA eigenvalue problem: $\mathbf{X}\mathbf{X}^\top \mathbf{u} = \lambda \mathbf{u}$

Notice that $\mathbf{u} = \mathbf{X}\boldsymbol{\alpha} = \sum_{i=1}^n \alpha_i \mathbf{x}_i$ for some weights $\boldsymbol{\alpha}$

Towards kernels

Representer theorem:

PCA solution is linear combination of $\mathbf{x}_i$s

Why?

Recall PCA eigenvalue problem: $\mathbf{X}\mathbf{X}^\top \mathbf{u} = \lambda \mathbf{u}$

Notice that $\mathbf{u} = \mathbf{X}\boldsymbol{\alpha} = \sum_{i=1}^n \alpha_i \mathbf{x}_i$ for some weights $\boldsymbol{\alpha}$

Analogy with SVMs: weight vector $\mathbf{w} = \mathbf{X}\alpha$

Towards kernels

Representer theorem:

PCA solution is linear combination of $\mathbf{x}_i$s

Why?

Recall PCA eigenvalue problem: $\mathbf{X}\mathbf{X}^\top \mathbf{u} = \lambda \mathbf{u}$
Notice that $\mathbf{u} = \mathbf{X}\boldsymbol{\alpha} = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i$ for some weights $\boldsymbol{\alpha}$
Analogy with SVMs: weight vector $\mathbf{w} = \mathbf{X}\alpha$

Key fact:

PCA only needs inner products $K = \mathbf{X}^\top \mathbf{X}$

Towards kernels

Representer theorem:

PCA solution is linear combination of $\mathbf{x}_i$s

Why?

Recall PCA eigenvalue problem: $\mathbf{X}\mathbf{X}^\top \mathbf{u} = \lambda \mathbf{u}$

Notice that $\mathbf{u} = \mathbf{X}\boldsymbol{\alpha} = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i$ for some weights $\boldsymbol{\alpha}$

Analogy with SVMs: weight vector $\mathbf{w} = \mathbf{X}\alpha$

Key fact:

PCA only needs inner products $K = \mathbf{X}^\top \mathbf{X}$

Why?

Use representer theorem on PCA objective:

$$\max_{\|\mathbf{u}\|=1} \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u} = \max_{\boldsymbol{\alpha}^\top \mathbf{X}^\top \mathbf{X}\boldsymbol{\alpha}=1} \boldsymbol{\alpha}^\top (\mathbf{X}^\top \mathbf{X})(\mathbf{X}^\top \mathbf{X})\boldsymbol{\alpha}$$

Kernel PCA

Kernel function: $k(\mathbf{x}_1, \mathbf{x}_2)$ such that

K, the kernel matrix formed by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, is positive semi-definite

Kernel PCA

Kernel function: $k(\mathbf{x}_1, \mathbf{x}_2)$ such that

K, the kernel matrix formed by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, is positive semi-definite

Examples:

Linear kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$

Kernel PCA

Kernel function: $k(\mathbf{x}_1, \mathbf{x}_2)$ such that

K, the kernel matrix formed by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, is positive semi-definite

Examples:

Linear kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$

Polynomial kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = (1 + \mathbf{x}_1^\top \mathbf{x}_2)^2$

Kernel PCA

Kernel function: $k(\mathbf{x}_1, \mathbf{x}_2)$ such that

K, the kernel matrix formed by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, is positive semi-definite

Examples:

Linear kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$

Polynomial kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = (1 + \mathbf{x}_1^\top \mathbf{x}_2)^2$

Gaussian (RBF) kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = e^{-\|\mathbf{x}_1 - \mathbf{x}_2\|^2}$

Kernel PCA

Kernel function: $k(\mathbf{x}_1, \mathbf{x}_2)$ such that

K, the kernel matrix formed by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$,

is positive semi-definite

Examples:

Linear kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$

Polynomial kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = (1 + \mathbf{x}_1^\top \mathbf{x}_2)^2$

Gaussian (RBF) kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = e^{-\|\mathbf{x}_1 - \mathbf{x}_2\|^2}$

Treat data points $\mathbf{x}$ as black boxes, only access via k

Kernel PCA

Kernel function: $k(\mathbf{x}_1, \mathbf{x}_2)$ such that

K, the kernel matrix formed by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, is positive semi-definite

Examples:

Linear kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$

Polynomial kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = (1 + \mathbf{x}_1^\top \mathbf{x}_2)^2$

Gaussian (RBF) kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = e^{-\|\mathbf{x}_1 - \mathbf{x}_2\|^2}$

Treat data points $\mathbf{x}$ as black boxes, only access via k

k intuitively measures "similarity" between two inputs

Kernel PCA

Kernel function: $k(\mathbf{x}_1, \mathbf{x}_2)$ such that

K, the kernel matrix formed by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, is positive semi-definite

Examples:

Linear kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$

Polynomial kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = (1 + \mathbf{x}_1^\top \mathbf{x}_2)^2$

Gaussian (RBF) kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = e^{-\|\mathbf{x}_1 - \mathbf{x}_2\|^2}$

Treat data points $\mathbf{x}$ as black boxes, only access via k

k intuitively measures "similarity" between two inputs

Mercer's theorem (using kernels is sensible)

Exists high-dimensional feature space ϕ such that

$k(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^\top \phi(\mathbf{x}_2)$ (like quick solution earlier!)

Solving kernel PCA

Direct method:

Kernel PCA objective:

$$\max_{\boldsymbol{\alpha}^\top K \boldsymbol{\alpha} = 1} \boldsymbol{\alpha}^\top K^2 \boldsymbol{\alpha}$$

Solving kernel PCA

Direct method:

Kernel PCA objective:

$$\max_{\boldsymbol{\alpha}^\top K \boldsymbol{\alpha}=1} \boldsymbol{\alpha}^\top K^2 \boldsymbol{\alpha}$$

$\Rightarrow$ kernel PCA eigenvalue problem: $\mathbf{X}^\top \mathbf{X} \boldsymbol{\alpha} = \lambda' \boldsymbol{\alpha}$

Solving kernel PCA

Direct method:

Kernel PCA objective:

$$\max_{\boldsymbol{\alpha}^\top K \boldsymbol{\alpha} = 1} \boldsymbol{\alpha}^\top K^2 \boldsymbol{\alpha}$$

$\Rightarrow$ kernel PCA eigenvalue problem: $\mathbf{X}^\top \mathbf{X} \boldsymbol{\alpha} = \lambda' \boldsymbol{\alpha}$

Modular method (if you don't want to think about kernels):

Find vectors $\mathbf{x}'_1, \dots, \mathbf{x}'_n$ such that

$$\mathbf{x}'^\top_i \mathbf{x}'_j = K_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$$

Solving kernel PCA

Direct method:

Kernel PCA objective:

$$\max_{\boldsymbol{\alpha}^\top K \boldsymbol{\alpha} = 1} \boldsymbol{\alpha}^\top K^2 \boldsymbol{\alpha}$$

$\Rightarrow$ kernel PCA eigenvalue problem: $\mathbf{X}^\top \mathbf{X} \boldsymbol{\alpha} = \lambda' \boldsymbol{\alpha}$

Modular method (if you don't want to think about kernels):

Find vectors $\mathbf{x}'_1, \dots, \mathbf{x}'_n$ such that

$$\mathbf{x}'^\top_i \mathbf{x}'_j = K_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$$

Key: use any vectors that preserve inner products

Solving kernel PCA

Direct method:

Kernel PCA objective:

$$\max_{\boldsymbol{\alpha}^\top K \boldsymbol{\alpha} = 1} \boldsymbol{\alpha}^\top K^2 \boldsymbol{\alpha}$$

$\Rightarrow$ kernel PCA eigenvalue problem: $\mathbf{X}^\top \mathbf{X} \boldsymbol{\alpha} = \lambda' \boldsymbol{\alpha}$

Modular method (if you don't want to think about kernels):

Find vectors $\mathbf{x}'_1, \dots, \mathbf{x}'_n$ such that

$$\mathbf{x}'^\top_i \mathbf{x}'_j = K_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$$

Key: use any vectors that preserve inner products

One possibility is Cholesky decomposition $K = \mathbf{X}^\top \mathbf{X}$

Roadmap

- Principal component analysis (PCA)
  - Basic principles
  - Case studies
  - Kernel PCA
  - Probabilistic PCA

- Canonical correlation analysis (CCA)

- Fisher discriminant analysis (FDA)

- Summary

Probabilistic modeling

So far, deal with objective functions:

$$\min_{\mathbf{U}} f(\mathbf{X}, \mathbf{U})$$

Probabilistic modeling

So far, deal with objective functions:

$$\min_{\mathbf{U}} f(\mathbf{X}, \mathbf{U})$$

Probabilistic modeling:

$$\max_{\mathbf{U}} p(\mathbf{X} \mid \mathbf{U})$$

Probabilistic modeling

So far, deal with objective functions:

$$\min_{\mathbf{U}} f(\mathbf{X}, \mathbf{U})$$

Probabilistic modeling:

$$\max_{\mathbf{U}} p(\mathbf{X} \mid \mathbf{U})$$

- Invent a generative story of how data $\mathbf{X}$ arose
- Play detective: infer parameters $\mathbf{U}$ that produced $\mathbf{X}$

Probabilistic modeling

So far, deal with objective functions:

$$\min_{\mathbf{U}} f(\mathbf{X}, \mathbf{U})$$

Probabilistic modeling:

$$\max_{\mathbf{U}} p(\mathbf{X} \mid \mathbf{U})$$

Invent a generative story of how data $\mathbf{X}$ arose
Play detective: infer parameters $\mathbf{U}$ that produced $\mathbf{X}$

Advantages:

- Model reports estimates of uncertainty
- Natural way to handle missing data

Probabilistic modeling

So far, deal with objective functions:

$$\min_{\mathbf{U}} f(\mathbf{X}, \mathbf{U})$$

Probabilistic modeling:

$$\max_{\mathbf{U}} p(\mathbf{X} \mid \mathbf{U})$$

Invent a generative story of how data $\mathbf{X}$ arose
Play detective: infer parameters $\mathbf{U}$ that produced $\mathbf{X}$

Advantages:

- Model reports estimates of uncertainty
- Natural way to handle missing data
- Natural way to introduce prior knowledge
- Natural way to incorporate in a larger model

Probabilistic modeling

So far, deal with objective functions:

$$\min_{\mathbf{U}} f(\mathbf{X}, \mathbf{U})$$

Probabilistic modeling:

$$\max_{\mathbf{U}} p(\mathbf{X} \mid \mathbf{U})$$

Invent a generative story of how data $\mathbf{X}$ arose
Play detective: infer parameters $\mathbf{U}$ that produced $\mathbf{X}$

Advantages:

- Model reports estimates of uncertainty
- Natural way to handle missing data
- Natural way to introduce prior knowledge
- Natural way to incorporate in a larger model

Example from last lecture: k-means $\Rightarrow$ GMMs

Probabilistic PCA

Generative story [Tipping and Bishop, 1999]:

For each data point $i = 1, \ldots, n$:

Probabilistic PCA

Generative story [Tipping and Bishop, 1999]:

For each data point $i = 1, \ldots, n$:
Draw the latent vector: $\mathbf{z}_i \sim \mathcal{N}(0, I_{k \times k})$

Probabilistic PCA

Generative story [Tipping and Bishop, 1999]:

For each data point $i = 1, \ldots, n$:
- Draw the latent vector: $\mathbf{z}_i \sim \mathcal{N}(0, I_{k \times k})$
- Create the data point: $\mathbf{x}_i \sim \mathcal{N}(\mathbf{U}\mathbf{z}_i, \sigma^2 I_{d \times d})$

Probabilistic PCA

Generative story [Tipping and Bishop, 1999]:

> For each data point $i = 1, \dots, n$:
> Draw the latent vector: $\mathbf{z}_i \sim \mathcal{N}(0, I_{k \times k})$
> Create the data point: $\mathbf{x}_i \sim \mathcal{N}(\mathbf{U}\mathbf{z}_i, \sigma^2 I_{d \times d})$

PCA finds the $\mathbf{U}$ that maximizes the likelihood of the data

Probabilistic PCA

Generative story [Tipping and Bishop, 1999]:

> For each data point $i = 1, \ldots, n$:
> Draw the latent vector: $\mathbf{z}_i \sim \mathcal{N}(0, I_{k \times k})$
> Create the data point: $\mathbf{x}_i \sim \mathcal{N}(\mathbf{U}\mathbf{z}_i, \sigma^2 I_{d \times d})$

PCA finds the $\mathbf{U}$ that maximizes the likelihood of the data

Advantages:

- Handles missing data (important for collaborative filtering)

Probabilistic PCA

Generative story [Tipping and Bishop, 1999]:

For each data point $i = 1, \ldots, n$:
Draw the latent vector: $\mathbf{z}_i \sim \mathcal{N}(0, I_{k \times k})$
Create the data point: $\mathbf{x}_i \sim \mathcal{N}(\mathbf{U}\mathbf{z}_i, \sigma^2 I_{d \times d})$

PCA finds the $\mathbf{U}$ that maximizes the likelihood of the data

Advantages:

- Handles missing data (important for collaborative filtering)
- Extension to factor analysis: allow non-isotropic noise (replace $\sigma^2 I_{d \times d}$ with arbitrary diagonal matrix)

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Generative story for pLSA [Hofmann, 1999]:

For each document $i = 1, \ldots, n$:

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Generative story for pLSA [Hofmann, 1999]:

For each document $i = 1, \ldots, n$:
  Repeat M times (number of word tokens in document):

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Generative story for pLSA [Hofmann, 1999]:

For each document $i = 1, \ldots, n$:
  Repeat M times (number of word tokens in document):
    Draw a latent topic: $\mathbf{z} \sim p(\mathbf{z} \mid i)$

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Generative story for pLSA [Hofmann, 1999]:

> For each document $i = 1, \dots, n$:
>
> Repeat M times (number of word tokens in document):
>
> Draw a latent topic: $\mathbf{z} \sim p(\mathbf{z} \mid i)$
>
> Choose the word token: $\mathbf{x} \sim p(\mathbf{x} \mid \mathbf{z})$

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Generative story for pLSA [Hofmann, 1999]:

For each document $i = 1, \ldots, n$:
  Repeat M times (number of word tokens in document):
    Draw a latent topic: $\mathbf{z} \sim p(\mathbf{z} \mid i)$
    Choose the word token: $\mathbf{x} \sim p(\mathbf{x} \mid \mathbf{z})$
  Set $\mathbf{x}_{ji}$ to be the number of times word j was chosen

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Generative story for pLSA [Hofmann, 1999]:

For each document $i = 1, \dots, n$:
  Repeat M times (number of word tokens in document):
    Draw a latent topic: $\mathbf{z} \sim p(\mathbf{z} \mid i)$
    Choose the word token: $\mathbf{x} \sim p(\mathbf{x} \mid \mathbf{z})$
  Set $\mathbf{x}_{ji}$ to be the number of times word j was chosen

Learning using Hard EM (analog of k-means):

E-step: fix parameters, choose best topics

M-step: fix topics, optimize parameters

More sophisticated methods: EM, Latent Dirichlet Allocation

Probabilistic latent semantic analysis (pLSA)

Motivation: in text analysis, $\mathbf{X}$ contains word counts; PCA (LSA) is bad model as it allows negative counts; pLSA fixes this

Generative story for pLSA [Hofmann, 1999]:

For each document $i = 1, \ldots, n$:
  Repeat M times (number of word tokens in document):
    Draw a latent topic: $\mathbf{z} \sim p(\mathbf{z} \mid i)$
    Choose the word token: $\mathbf{x} \sim p(\mathbf{x} \mid \mathbf{z})$
  Set $\mathbf{x}_{ji}$ to be the number of times word j was chosen

Learning using Hard EM (analog of k-means):
  E-step: fix parameters, choose best topics
  M-step: fix topics, optimize parameters

More sophisticated methods: EM, Latent Dirichlet Allocation

Comparison to a mixture model for clustering:
  Mixture model: assume a single topic for entire document
  pLSA: allow multiple topics per document

Roadmap

- Principal component analysis (PCA)
  - Basic principles
  - Case studies
  - Kernel PCA
  - Probabilistic PCA
- Canonical correlation analysis (CCA)
- Fisher discriminant analysis (FDA)
- Summary

Motivation for CCA [Hotelling, 1936]

Often, each data point consists of two views:

- Image retrieval: for each image, have the following:
  - $\mathbf{x}$: Pixels (or other visual features)
  - $\mathbf{y}$: Text around the image

Motivation for CCA [Hotelling, 1936]

Often, each data point consists of two views:

- Image retrieval: for each image, have the following:
  - $\mathbf{x}$: Pixels (or other visual features)
  - $\mathbf{y}$: Text around the image
- Time series:
  - $\mathbf{x}$: Signal at time t
  - $\mathbf{y}$: Signal at time $t+1$

Motivation for CCA [Hotelling, 1936]

Often, each data point consists of two views:

- Image retrieval: for each image, have the following:
  - $\mathbf{x}$: Pixels (or other visual features)
  - $\mathbf{y}$: Text around the image
- Time series:
  - $\mathbf{x}$: Signal at time t
  - $\mathbf{y}$: Signal at time $t+1$
- Two-view learning: divide features into two sets
  - $\mathbf{x}$: Features of a word/object, etc.
  - $\mathbf{y}$: Features of the context in which it appears

Motivation for CCA [Hotelling, 1936]

Often, each data point consists of two views:

- Image retrieval: for each image, have the following:
  - $\mathbf{x}$: Pixels (or other visual features)
  - $\mathbf{y}$: Text around the image
- Time series:
  - $\mathbf{x}$: Signal at time t
  - $\mathbf{y}$: Signal at time $t + 1$
- Two-view learning: divide features into two sets
  - $\mathbf{x}$: Features of a word/object, etc.
  - $\mathbf{y}$: Features of the context in which it appears

Goal: reduce the dimensionality of the two views jointly

An example

Setup:

Input data: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$ (matrices $\mathbf{X}$, $\mathbf{Y}$)

Goal: find pair of projections $(\mathbf{u}, \mathbf{v})$

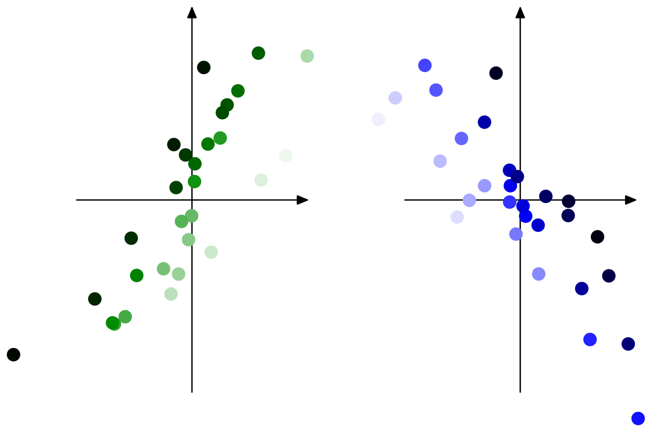
An example

Setup:

Input data: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$ (matrices $\mathbf{X}, \mathbf{Y}$)

Goal: find pair of projections $(\mathbf{u}, \mathbf{v})$

In figure, $\mathbf{x}$ and $\mathbf{y}$ are paired by brightness



An example

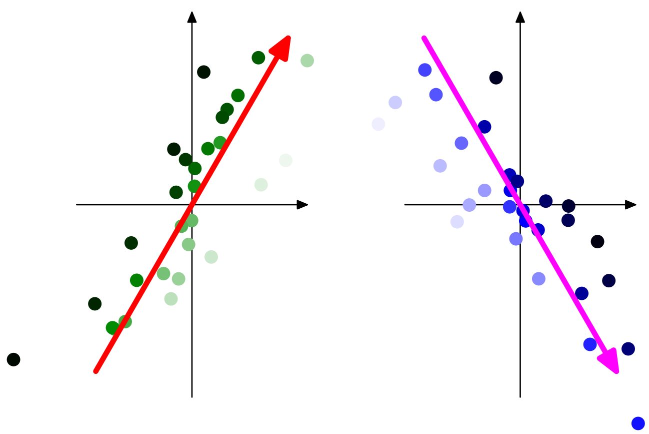
Setup:

Input data: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$ (matrices $\mathbf{X}, \mathbf{Y}$)

Goal: find pair of projections $(\mathbf{u}, \mathbf{v})$

In figure, $\mathbf{x}$ and $\mathbf{y}$ are paired by brightness

Dimensionality reduction solutions:



Independent

An example

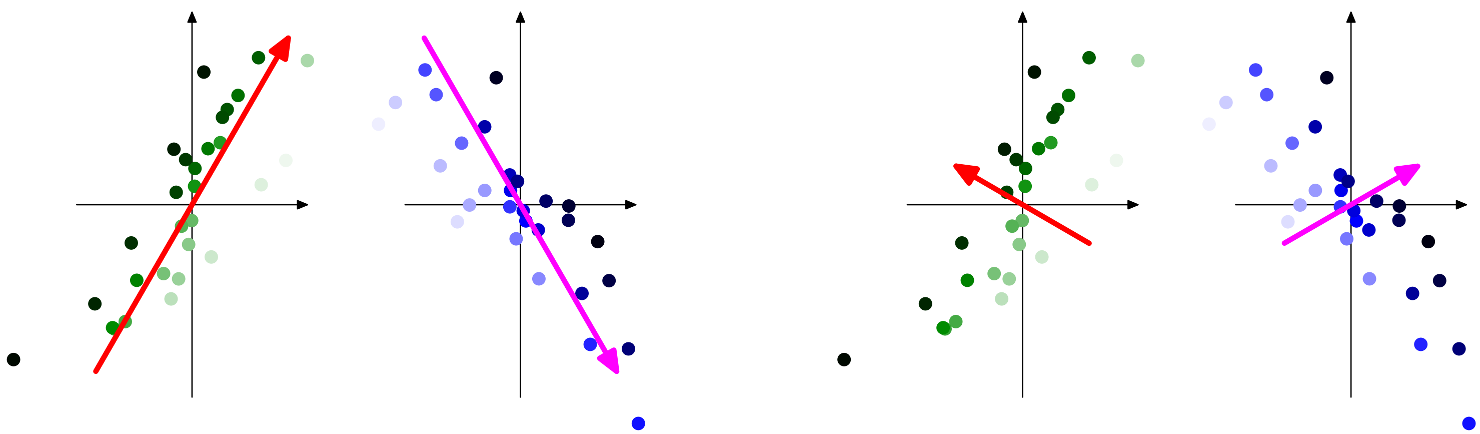
Setup:

Input data: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$ (matrices $\mathbf{X}, \mathbf{Y}$)

Goal: find pair of projections $(\mathbf{u}, \mathbf{v})$

In figure, $\mathbf{x}$ and $\mathbf{y}$ are paired by brightness

Dimensionality reduction solutions:



Independent　　　　Joint

From PCA to CCA

PCA on views separately: no covariance term

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}}{\mathbf{u}^\top \mathbf{u}} + \frac{\mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}{\mathbf{v}^\top \mathbf{v}}$$

From PCA to CCA

PCA on views separately: no covariance term

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}}{\mathbf{u}^\top \mathbf{u}} + \frac{\mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}{\mathbf{v}^\top \mathbf{v}}$$

PCA on concatenation $(\mathbf{X}^\top, \mathbf{Y}^\top)^\top$: includes covariance term

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u} + 2\mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v} + \mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}{\mathbf{u}^\top \mathbf{u} + \mathbf{v}^\top \mathbf{v}}$$

From PCA to CCA

PCA on views separately: no covariance term

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}}{\mathbf{u}^\top \mathbf{u}} + \frac{\mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}{\mathbf{v}^\top \mathbf{v}}$$

PCA on concatenation $(\mathbf{X}^\top, \mathbf{Y}^\top)^\top$: includes covariance term

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u} + 2\mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v} + \mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}{\mathbf{u}^\top \mathbf{u} + \mathbf{v}^\top \mathbf{v}}$$

Maximum covariance: drop variance terms

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}}{\sqrt{\mathbf{u}^\top \mathbf{u}}\sqrt{\mathbf{v}^\top \mathbf{v}}}$$

From PCA to CCA

PCA on views separately: no covariance term

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}}{\mathbf{u}^\top \mathbf{u}} + \frac{\mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}{\mathbf{v}^\top \mathbf{v}}$$

PCA on concatenation $(\mathbf{X}^\top, \mathbf{Y}^\top)^\top$: includes covariance term

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u} + 2\mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v} + \mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}{\mathbf{u}^\top \mathbf{u} + \mathbf{v}^\top \mathbf{v}}$$

Maximum covariance: drop variance terms

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}}{\sqrt{\mathbf{u}^\top \mathbf{u}}\sqrt{\mathbf{v}^\top \mathbf{v}}}$$

Maximum correlation (CCA): divide out variance terms

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}}{\sqrt{\mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}}\sqrt{\mathbf{v}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{v}}}$$

Canonical correlation analysis (CCA)

Definitions:

Variance: $\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x}) = \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}$

Canonical correlation analysis (CCA)

Definitions:

Variance: $\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x}) = \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}$

Covariance: $\widehat{\text{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y}) = \mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}$

Canonical correlation analysis (CCA)

Definitions:

Variance: $\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x}) = \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}$

Covariance: $\widehat{\text{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y}) = \mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}$

Correlation: $\frac{\widehat{\text{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})}{\sqrt{\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x})}\sqrt{\widehat{\text{var}}(\mathbf{v}^\top \mathbf{y})}}$

Canonical correlation analysis (CCA)

Definitions:

Variance: $\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x}) = \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}$

Covariance: $\widehat{\text{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y}) = \mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}$

Correlation: $\frac{\widehat{\text{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})}{\sqrt{\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x})}\sqrt{\widehat{\text{var}}(\mathbf{v}^\top \mathbf{y})}}$

Objective: maximize correlation between projected views

$$\max_{\mathbf{u},\mathbf{v}} \widehat{\text{corr}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})$$

Canonical correlation analysis (CCA)

Definitions:

Variance: $\widehat{\mathrm{var}}(\mathbf{u}^\top \mathbf{x}) = \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}$

Covariance: $\widehat{\mathrm{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y}) = \mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}$

Correlation: $\frac{\widehat{\mathrm{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})}{\sqrt{\widehat{\mathrm{var}}(\mathbf{u}^\top \mathbf{x})}\sqrt{\widehat{\mathrm{var}}(\mathbf{v}^\top \mathbf{y})}}$

Objective: maximize correlation between projected views

$$\max_{\mathbf{u},\mathbf{v}} \widehat{\mathrm{corr}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})$$

Properties:

- Focus on how variables are related, not how much they vary

Canonical correlation analysis (CCA)

Definitions:

Variance: $\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x}) = \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}$

Covariance: $\widehat{\text{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y}) = \mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}$

Correlation: $\frac{\widehat{\text{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})}{\sqrt{\widehat{\text{var}}(\mathbf{u}^\top \mathbf{x})}\sqrt{\widehat{\text{var}}(\mathbf{v}^\top \mathbf{y})}}$

Objective: maximize correlation between projected views

$$\max_{\mathbf{u},\mathbf{v}} \widehat{\text{corr}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})$$

Properties:

- Focus on how variables are related, not how much they vary
- Invariant to any rotation and scaling of data

Canonical correlation analysis (CCA)

Definitions:

Variance: $\widehat{\mathrm{var}}(\mathbf{u}^\top \mathbf{x}) = \mathbf{u}^\top \mathbf{X}\mathbf{X}^\top \mathbf{u}$

Covariance: $\widehat{\mathrm{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y}) = \mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}$

Correlation: $\frac{\widehat{\mathrm{cov}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})}{\sqrt{\widehat{\mathrm{var}}(\mathbf{u}^\top \mathbf{x})}\sqrt{\widehat{\mathrm{var}}(\mathbf{v}^\top \mathbf{y})}}$

Objective: maximize correlation between projected views

$$\max_{\mathbf{u},\mathbf{v}} \widehat{\mathrm{corr}}(\mathbf{u}^\top \mathbf{x}, \mathbf{v}^\top \mathbf{y})$$

Properties:

- Focus on how variables are related, not how much they vary
- Invariant to any rotation and scaling of data

Solved via a generalized eigenvalue problem ($A\mathbf{w} = \lambda B\mathbf{w}$)

Regularization is important

Extreme examples of degeneracy:

- If $\mathbf{x} = A\mathbf{y}$, then any $(\mathbf{u}, \mathbf{v})$ with $\mathbf{u} = A\mathbf{v}$ is optimal (correlation 1)

Regularization is important

Extreme examples of degeneracy:

- If $\mathbf{x} = A\mathbf{y}$, then any $(\mathbf{u}, \mathbf{v})$ with $\mathbf{u} = A\mathbf{v}$ is optimal (correlation 1)
- If $\mathbf{x}$ and $\mathbf{y}$ are independent, then any $(\mathbf{u}, \mathbf{v})$ is optimal (correlation 0)

Regularization is important

Extreme examples of degeneracy:

- If $\mathbf{x} = A\mathbf{y}$, then any $(\mathbf{u}, \mathbf{v})$ with $\mathbf{u} = A\mathbf{v}$ is optimal (correlation 1)
- If $\mathbf{x}$ and $\mathbf{y}$ are independent, then any $(\mathbf{u}, \mathbf{v})$ is optimal (correlation 0)

Problem: if $\mathbf{X}$ or $\mathbf{Y}$ has rank n, then any $(\mathbf{u}, \mathbf{v})$ is optimal (correlation 1) with $\mathbf{u} = \mathbf{X}^{\dagger\top}\mathbf{Y}\mathbf{v} \Rightarrow$ CCA is meaningless!

Regularization is important

Extreme examples of degeneracy:

- If $\mathbf{x} = A\mathbf{y}$, then any $(\mathbf{u}, \mathbf{v})$ with $\mathbf{u} = A\mathbf{v}$ is optimal (correlation 1)
- If $\mathbf{x}$ and $\mathbf{y}$ are independent, then any $(\mathbf{u}, \mathbf{v})$ is optimal (correlation 0)

Problem: if $\mathbf{X}$ or $\mathbf{Y}$ has rank n, then any $(\mathbf{u}, \mathbf{v})$ is optimal (correlation 1) with $\mathbf{u} = \mathbf{X}^{\dagger\top}\mathbf{Y}\mathbf{v}$ $\Rightarrow$ CCA is meaningless!

Solution: regularization (interpolate between maximum covariance and maximum correlation)

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\top \mathbf{X}\mathbf{Y}^\top \mathbf{v}}{\sqrt{\mathbf{u}^\top(\mathbf{X}\mathbf{X}^\top + \lambda I)\mathbf{u}}\sqrt{\mathbf{v}^\top(\mathbf{Y}\mathbf{Y}^\top + \lambda I)\mathbf{v}}}$$

Kernel CCA

Two kernels: k_x and k_y

Kernel CCA

Two kernels: k_x and k_y

Direct method:

(some math)

Kernel CCA

Two kernels: k_x and k_y

Direct method:

(some math)

Modular method:

1. Transform $\mathbf{x}_i$ into $\mathbf{x}'_i \in \mathbb{R}^n$ satisfying $k(\mathbf{x}_i, \mathbf{x}_j) = {\mathbf{x}'_i}^\top \mathbf{x}'_j$ (do same for $\mathbf{y}$)

Kernel CCA

Two kernels: k_x and k_y

Direct method:

(some math)

Modular method:

1. Transform $\mathbf{x}_i$ into $\mathbf{x}'_i \in \mathbb{R}^n$ satisfying $k(\mathbf{x}_i, \mathbf{x}_j) = {\mathbf{x}'_i}^\top \mathbf{x}'_j$ (do same for $\mathbf{y}$)
2. Perform regular CCA

Kernel CCA

Two kernels: k_x and k_y

Direct method:

(some math)

Modular method:

1. Transform $\mathbf{x}_i$ into $\mathbf{x}'_i \in \mathbb{R}^n$ satisfying $k(\mathbf{x}_i, \mathbf{x}_j) = {\mathbf{x}'_i}^\top \mathbf{x}'_j$ (do same for $\mathbf{y}$)
2. Perform regular CCA

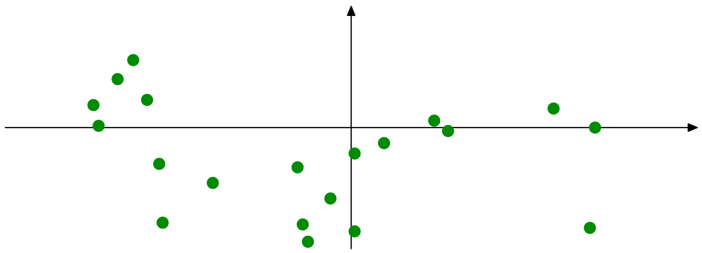
Regularization is especially important for kernel CCA!

Roadmap

- Principal component analysis (PCA)
  - Basic principles
  - Case studies
  - Kernel PCA
  - Probabilistic PCA
- Canonical correlation analysis (CCA)
- Fisher discriminant analysis (FDA)
- Summary

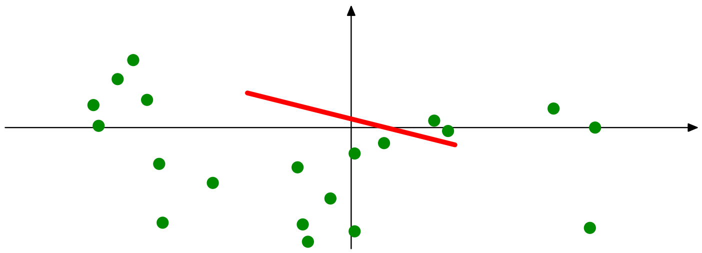
Motivation for FDA [Fisher, 1936]

What is the best linear projection?



Motivation for FDA [Fisher, 1936]

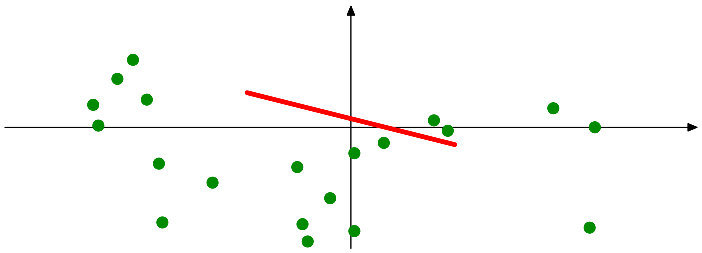
What is the best linear projection?

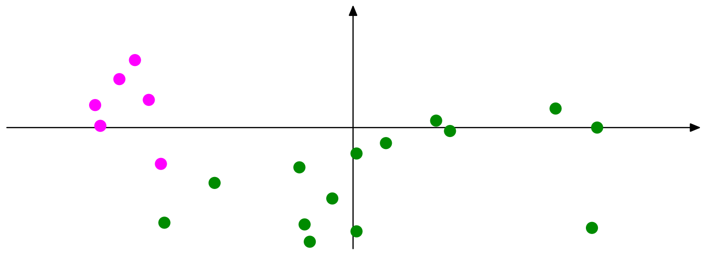


PCA solution

Motivation for FDA [Fisher, 1936]

What is the best linear projection with these labels?

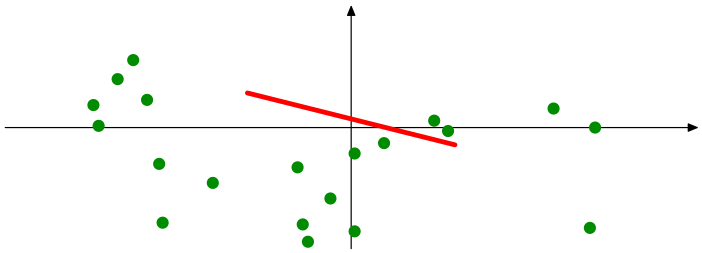




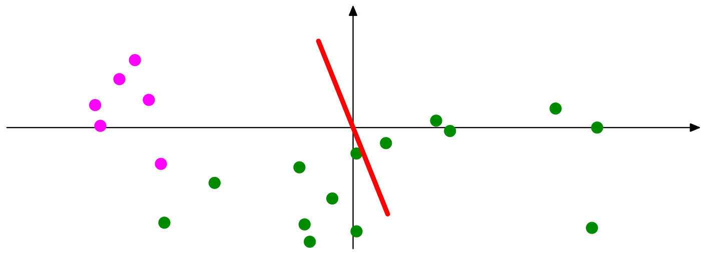
PCA solution

Motivation for FDA [Fisher, 1936]

What is the best linear projection with these labels?



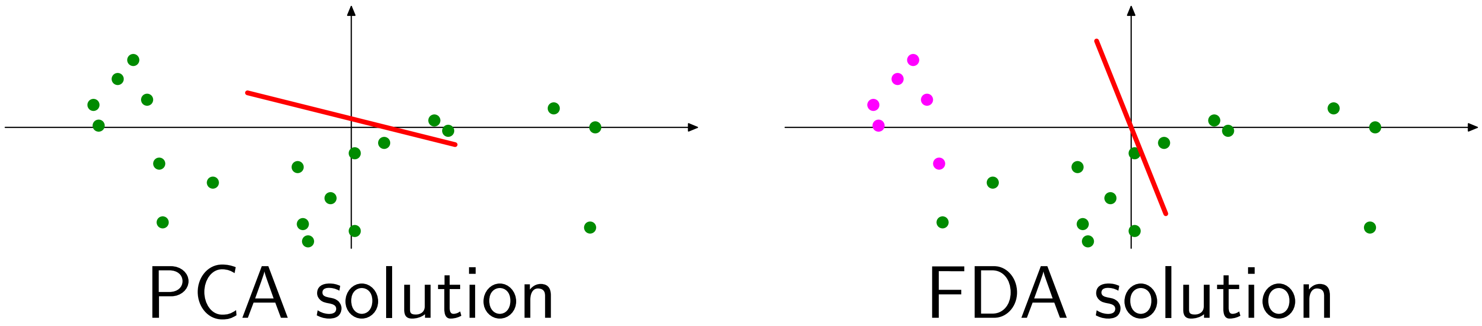
PCA solution



FDA solution

Motivation for FDA [Fisher, 1936]

What is the best linear projection with these labels?

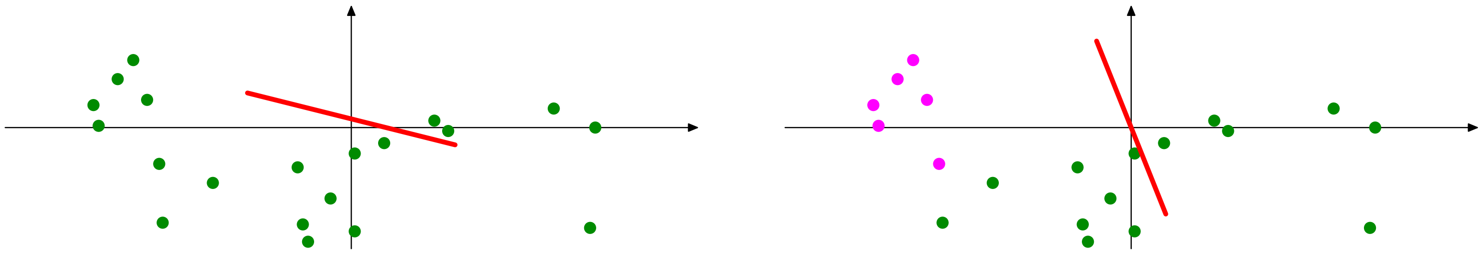


Goal: reduce the dimensionality given labels

Idea: want projection to maximize overall interclass variance relative to intraclass variance

Motivation for FDA [Fisher, 1936]

What is the best linear projection with these labels?



PCA solution FDA solution

Goal: reduce the dimensionality given labels

Idea: want projection to maximize overall interclass variance relative to intraclass variance

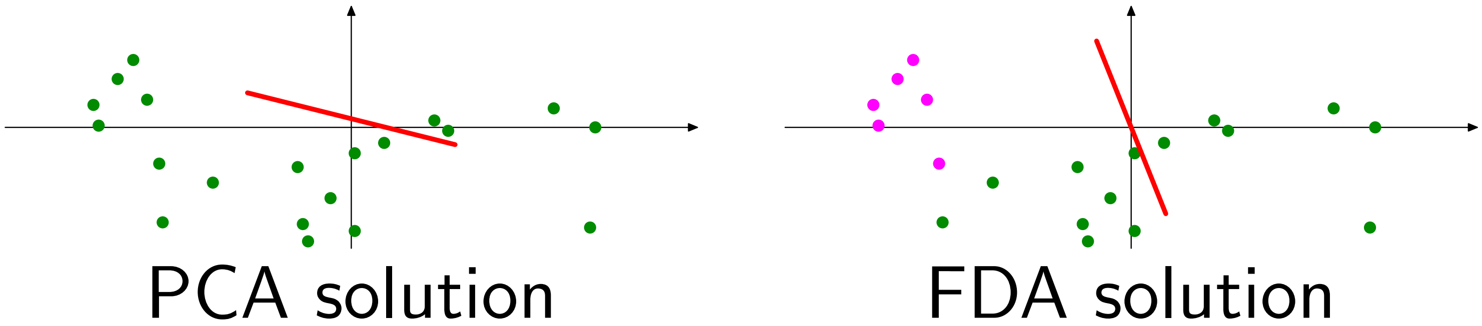
Linear classifiers (logistic regression, SVMs) have similar feel:

Find one-dimensional subspace $\mathbf{w}$,

e.g., to maximize margin between different classes

Motivation for FDA [Fisher, 1936]

What is the best linear projection with these labels?



Goal: reduce the dimensionality given labels

Idea: want projection to maximize overall interclass variance relative to intraclass variance

Linear classifiers (logistic regression, SVMs) have similar feel:

Find one-dimensional subspace $\mathbf{w}$,

e.g., to maximize margin between different classes

FDA handles multiple classes, allows multiple dimensions

FDA objective function

Setup: $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{1, \ldots, m\}$, for $i = 1, \ldots, n$

FDA objective function

Setup: $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{1, \dots, m\}$, for $i = 1, \dots, n$

Objective: maximize $\frac{\text{interclass variance}}{\text{intraclass variance}} = \frac{\text{total variance}}{\text{intraclass variance}} - 1$

FDA objective function

Setup: $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{1, \ldots, m\}$, for $i = 1, \ldots, n$

Objective: maximize $\frac{\text{interclass variance}}{\text{intraclass variance}} = \frac{\text{total variance}}{\text{intraclass variance}} - 1$

Total variance: $\frac{1}{n} \sum_i (\mathbf{u}^\top (\mathbf{x}_i - \mu))^2$

Mean of all points: $\mu = \frac{1}{n} \sum_i \mathbf{x}_i$

FDA objective function

Setup: $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{1, \ldots, m\}$, for $i = 1, \ldots, n$

Objective: maximize $\frac{\text{interclass variance}}{\text{intraclass variance}} = \frac{\text{total variance}}{\text{intraclass variance}} - 1$

Total variance: $\frac{1}{n} \sum_i (\mathbf{u}^\top (\mathbf{x}_i - \mu))^2$

Mean of all points: $\mu = \frac{1}{n} \sum_i \mathbf{x}_i$

Intraclass variance: $\frac{1}{n} \sum_i (\mathbf{u}^\top (\mathbf{x}_i - \mu_{\mathbf{y}_i}))^2$

Mean of points in class y: $\mu_y = \frac{1}{|\{i : y_i = y\}|} \sum_{i : y_i = y} \mathbf{x}_i$

FDA objective function

Setup: $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{1, \dots, m\}$, for $i = 1, \dots, n$

Objective: maximize $\frac{\text{interclass variance}}{\text{intraclass variance}} = \frac{\text{total variance}}{\text{intraclass variance}} - 1$

Total variance: $\frac{1}{n}\sum_i (\mathbf{u}^\top(\mathbf{x}_i - \mu))^2$

Mean of all points: $\mu = \frac{1}{n}\sum_i \mathbf{x}_i$

Intraclass variance: $\frac{1}{n}\sum_i (\mathbf{u}^\top(\mathbf{x}_i - \mu_{\mathbf{y}_i}))^2$

Mean of points in class y: $\mu_y = \frac{1}{|\{i : y_i = y\}|}\sum_{i : y_i = y} \mathbf{x}_i$

Reduces to a generalized eigenvalue problem.

FDA objective function

Setup: $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{1, \dots, m\}$, for $i = 1, \dots, n$

Objective: maximize $\frac{\text{interclass variance}}{\text{intraclass variance}} = \frac{\text{total variance}}{\text{intraclass variance}} - 1$

Total variance: $\frac{1}{n} \sum_i (\mathbf{u}^\top (\mathbf{x}_i - \mu))^2$

Mean of all points: $\mu = \frac{1}{n} \sum_i \mathbf{x}_i$

Intraclass variance: $\frac{1}{n} \sum_i (\mathbf{u}^\top (\mathbf{x}_i - \mu_{\mathbf{y}_i}))^2$

Mean of points in class y: $\mu_y = \frac{1}{|\{i : y_i = y\}|} \sum_{i : y_i = y} \mathbf{x}_i$

Reduces to a generalized eigenvalue problem.

Kernel FDA: use modular method

Other linear methods

Random projections:

Randomly project data onto $k = O(\log n)$ dimensions

Other linear methods

Random projections:

Randomly project data onto $k = O(\log n)$ dimensions

All pairwise distances preserved with high probability

$$\|\mathbf{U}^\top \mathbf{x}_i - \mathbf{U}^\top \mathbf{x}_j\|^2 \approxeq \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for all } i, j$$

Other linear methods

Random projections:

Randomly project data onto $k = O(\log n)$ dimensions

All pairwise distances preserved with high probability

$$\|\mathbf{U}^\top \mathbf{x}_i - \mathbf{U}^\top \mathbf{x}_j\|^2 \approxeq \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for all } i, j$$

Trivial to implement

Other linear methods

Random projections:

Randomly project data onto $k = O(\log n)$ dimensions

All pairwise distances preserved with high probability

$$\|\mathbf{U}^\top \mathbf{x}_i - \mathbf{U}^\top \mathbf{x}_j\|^2 \approx \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for all } i, j$$

Trivial to implement

Kernel dimensionality reduction:

One type of sufficient dimensionality reduction

Find subspace that contains all <u>information</u> about labels

Other linear methods

Random projections:

Randomly project data onto $k = O(\log n)$ dimensions

All pairwise distances preserved with high probability

$$\|\mathbf{U}^\top \mathbf{x}_i - \mathbf{U}^\top \mathbf{x}_j\|^2 \approxeq \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for all } i, j$$

Trivial to implement

Kernel dimensionality reduction:

One type of sufficient dimensionality reduction

Find subspace that contains all information about labels

$$\mathbf{y} \perp\!\!\!\perp \mathbf{x} \mid \mathbf{U}^\top \mathbf{x}$$

Other linear methods

Random projections:

Randomly project data onto $k = O(\log n)$ dimensions

All pairwise distances preserved with high probability

$$\|\mathbf{U}^\top \mathbf{x}_i - \mathbf{U}^\top \mathbf{x}_j\|^2 \approxeq \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for all } i, j$$

Trivial to implement

Kernel dimensionality reduction:

One type of sufficient dimensionality reduction

Find subspace that contains all information about labels

$$\mathbf{y} \perp\!\!\!\perp \mathbf{x} \mid \mathbf{U}^\top \mathbf{x}$$

Capturing information is stronger than capturing variance

Other linear methods

Random projections:

Randomly project data onto $k = O(\log n)$ dimensions

All pairwise distances preserved with high probability

$$\|\mathbf{U}^\top \mathbf{x}_i - \mathbf{U}^\top \mathbf{x}_j\|^2 \approxeq \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for all } i, j$$

Trivial to implement

Kernel dimensionality reduction:

One type of sufficient dimensionality reduction

Find subspace that contains all information about labels

$$\mathbf{y} \perp\!\!\!\perp \mathbf{x} \mid \mathbf{U}^\top \mathbf{x}$$

Capturing information is stronger than capturing variance

Hard nonconvex optimization problem

Summary

Framework: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}$, $\mathbf{x} \approxeq \mathbf{U}\mathbf{z}$

Summary

Framework: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}$, $\mathbf{x} \approxeq \mathbf{U}\mathbf{z}$

Criteria for choosing $\mathbf{U}$:

- PCA: maximize projected variance
- CCA: maximize projected correlation
- FDA: maximize projected $\frac{\text{interclass variance}}{\text{intraclass variance}}$

Summary

Framework: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}$, $\mathbf{x} \approxeq \mathbf{U}\mathbf{z}$

Criteria for choosing $\mathbf{U}$:

- PCA: maximize projected variance
- CCA: maximize projected correlation
- FDA: maximize projected $\frac{\text{interclass variance}}{\text{intraclass variance}}$

Algorithm: generalized eigenvalue problem

Summary

Framework: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}$, $\mathbf{x} \approxeq \mathbf{U}\mathbf{z}$

Criteria for choosing $\mathbf{U}$:

- PCA: maximize projected variance
- CCA: maximize projected correlation
- FDA: maximize projected $\frac{\text{interclass variance}}{\text{intraclass variance}}$

Algorithm: generalized eigenvalue problem

Extensions:

non-linear using kernels (using same linear framework)

probabilistic, sparse, robust (hard optimization)