

CS6510
Applied Machine Learning

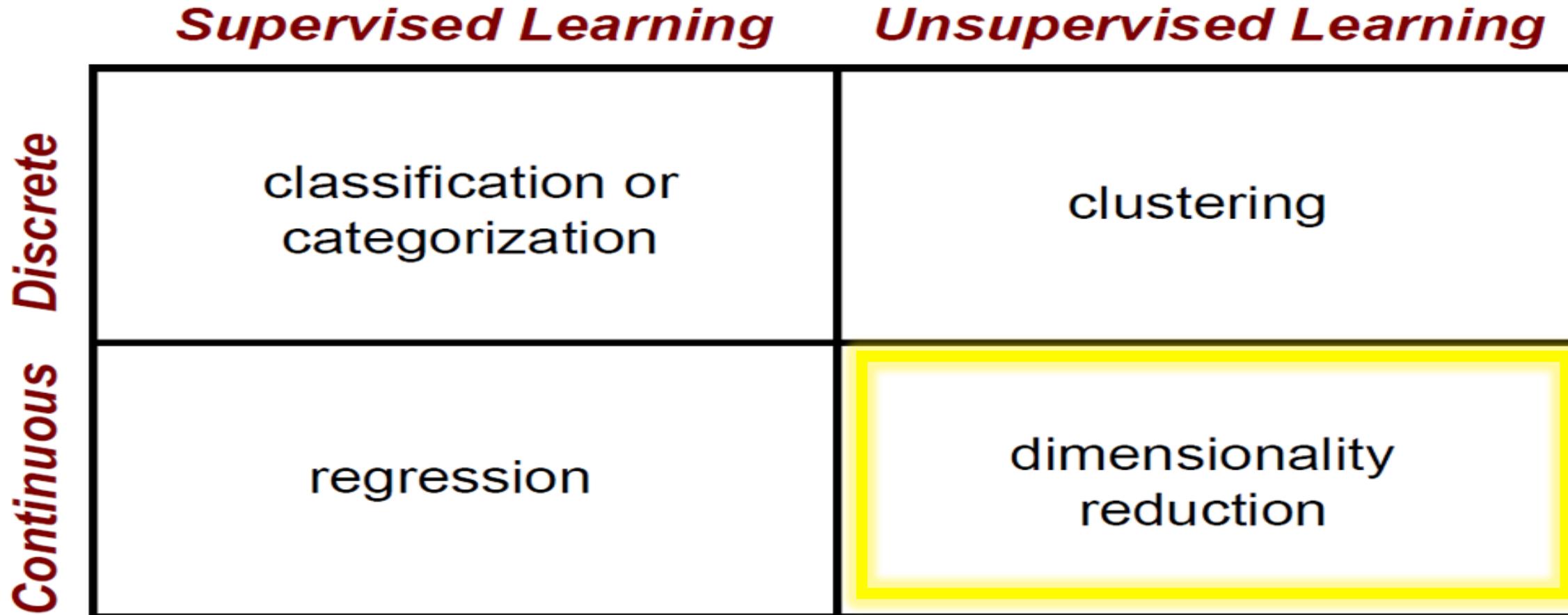
Dimensionality Reduction

4 Nov 2017

Vineeth N Balasubramanian



ML Problems



Outline

- Linear Methods
 - Principal Component Analysis (PCA)
 - Fisher (Linear) Discriminant Analysis (LDA)
 - Multi-Dimensionality Scaling (MDS)
- Non-linear Methods
 - Kernel PCA
 - Manifold Learning methods: ISOMAP, LLE, Laplacian Eigenmaps
 - t-Stochastic Neighborhood Embedding (t-SNE)

How does the brain store these images?



What is Dimensionality Reduction

- Refers to the mapping of the original high-dimensional data onto a lower-dimensional space.
- Criterion for feature reduction can be different based on different problem settings.
 - Unsupervised setting: minimize the information loss
 - Supervised setting: maximize the class discrimination
- Given a set of data points of p variables $\{x_1, x_2, \dots, x_n\}$
Compute the linear transformation (projection)

$$G \in \mathbb{R}^{p \times d} : x \in \mathbb{R}^p \rightarrow y = G^T x \in \mathbb{R}^d \quad (d \ll p)$$

Linear DR

Dimensionality Reduction vs Feature Selection

- **Feature reduction**
 - All original features are used
 - The transformed features are linear combinations of the original features.
- **Feature selection**
 - Only a subset of the original features are used.
- **Continuous versus discrete**

Why DR?

- Most machine learning techniques may not be effective for high-dimensional data
 - **Curse of Dimensionality**
 - Query accuracy and efficiency degrade rapidly as the dimension increases
 - Lower space and time complexity
 - Visualization, Data compression, Noise/irrelevant feature removal
- The **intrinsic** dimension may be small
 - For example, the number of genes responsible for a certain type of disease may be small

High-dimensional data are strange

- Consider the hypersphere of radius r on a space of dimension d

$$\mathcal{S} = \left\{ \mathbf{x} \mid \sum_{i=1}^d x_i^2 \leq r^2 \right\}$$

- Its volume is

$$V_d(r) = \frac{r^d \pi^{\frac{d}{2}}}{\Gamma\left(\frac{d}{2} + 1\right)}$$

- Where $\Gamma(n)$ is the Gamma function

$$\Gamma(n) = \int_0^\infty e^{-x} x^{n-1} dx$$

Source: N Vasconcelos

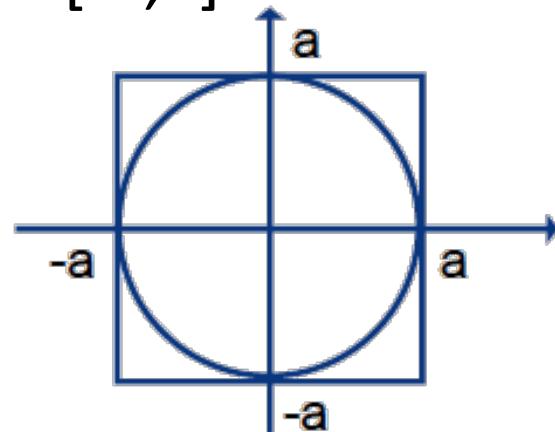


4-Nov-17

CS6510 - Applied Machine Learning

Hypercube vs Hypersphere

- Consider the hyper-cube $[-a, a]^d$ and the inscribed hyper-sphere, i.e.



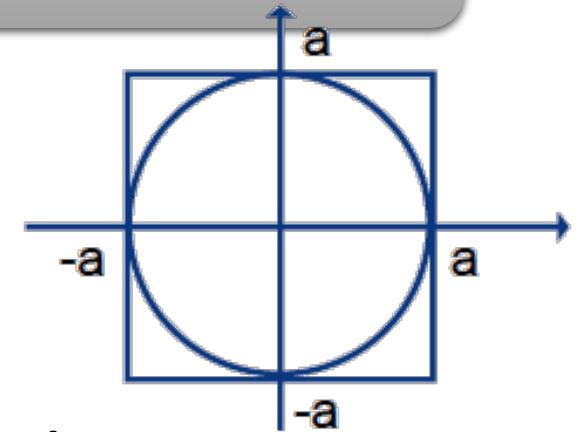
- What does your intuition tell you about the relative sizes of these two objects?
 - Volume of sphere \approx volume of cube
 - Volume of sphere \gg volume of cube
 - Volume of sphere \ll volume of cube

Source: N Vasconcelos

Hypercube vs Hypersphere

- Let's compute the answer

$$f_d = \frac{Vol(sphere)}{Vol(cube)} = \frac{\frac{a^d \pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2}+1)}}{(2a)^d} = \frac{\pi^{\frac{d}{2}}}{2^d \Gamma(\frac{d}{2} + 1)}$$



- Sequence that does not depend on a , just on the dimension d !

d	1	2	3	4	5	6	7
f_d	1	.785	.524	.308	.164	.08	.037

- It goes to zero, and goes to zero fast!

Source: N Vasconcelos

Hypercube vs Hypersphere

- Let's compute the answer

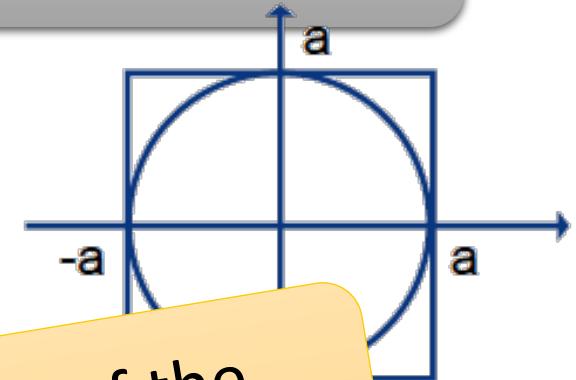
$$f_d = \frac{Vol(sphere)}{Vol(cube)} = \frac{\frac{a^d \pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2}+1)}}{(2a)^d} = \frac{\pi^{\frac{d}{2}}}{2^d \Gamma(\frac{d}{2} + 1)}$$

- Sequence that does not depend on d!

As the dimension of the space increases, the volume of the sphere is much smaller (infinitesimal) than that of the cube!

.785	.524	.308	.164	.08	.037
------	------	------	------	-----	------

- It goes to zero, and goes to zero fast!



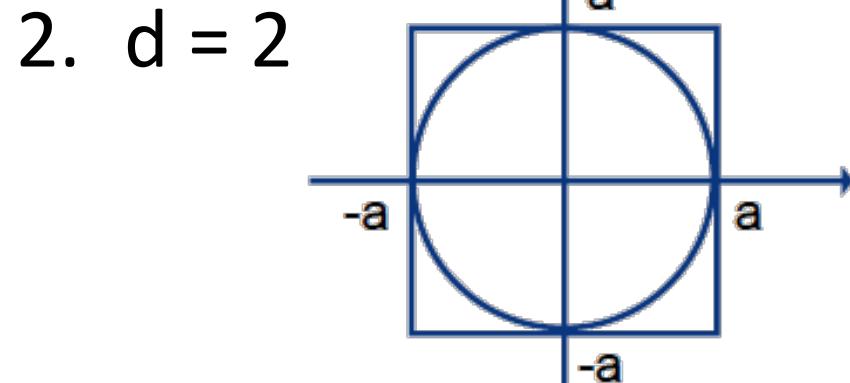
Source: N Vasconcelos

Hypercube vs Hypersphere

Actually not very surprising

1. $d = 1$ 

Volume is the same



Volume of sphere is already smaller

Source: N Vasconcelos



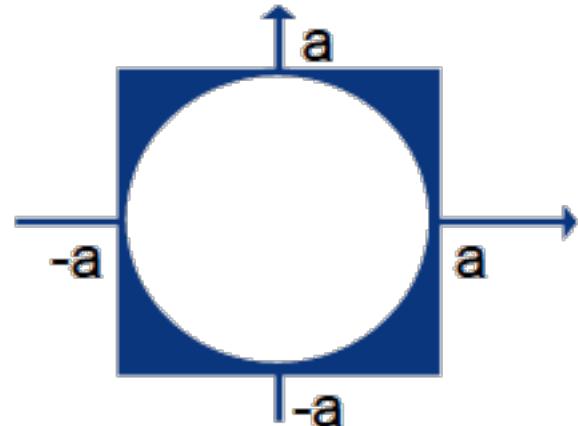
4-Nov-17

CS6510 - Applied Machine Learning

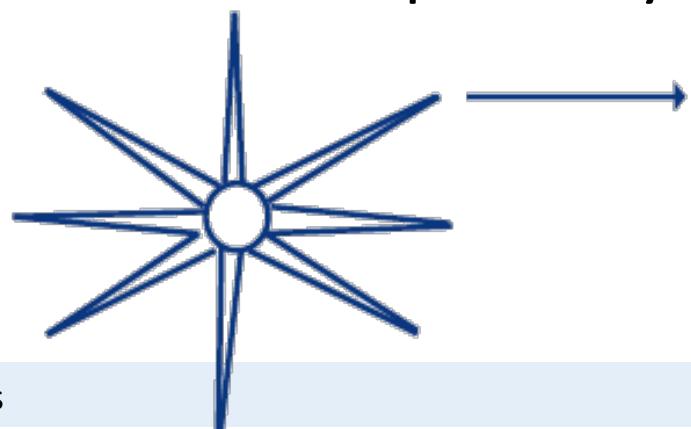
12

Hypercube vs Hypersphere

- As the dimension increases the volume of the shaded corners becomes larger



- In high dimensions the picture you should have in mind is

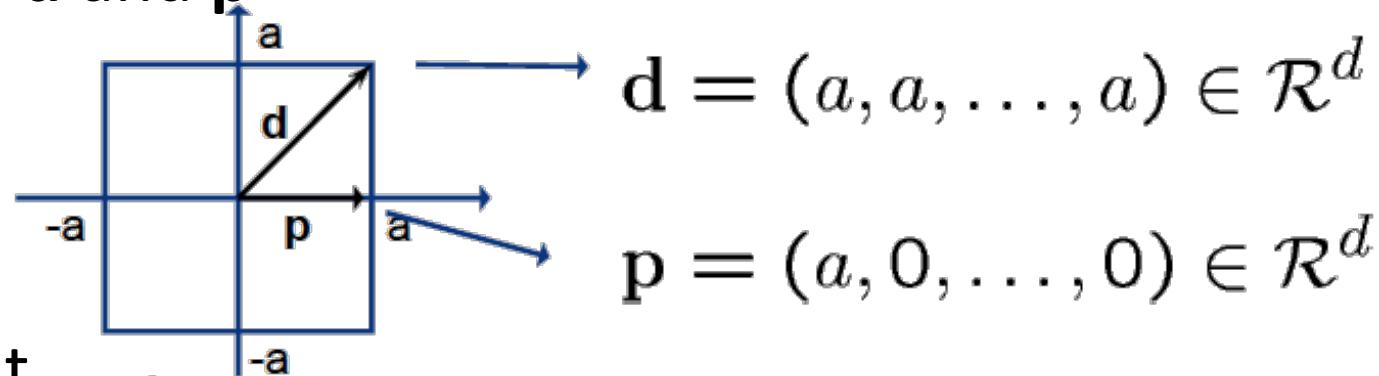


all the volume of the cube
is in these spikes!

Source: N Vasconcelos

We can check mathematically

- Consider \mathbf{d} and \mathbf{p}



- Note that

$$\frac{\|\mathbf{d}\|^2}{\|\mathbf{p}\|^2} = \frac{da^2}{a^2} = d \rightarrow \infty \quad \cos\theta = \frac{\mathbf{d}^T \mathbf{p}}{\sqrt{\|\mathbf{d}\|^2 \|\mathbf{p}\|^2}} = \frac{a^2}{\sqrt{da^2 a^2}} = \frac{1}{\sqrt{d}} \rightarrow 0$$

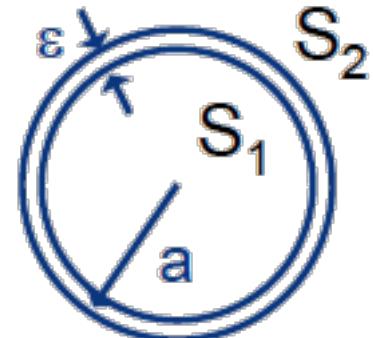
- \mathbf{d} orthogonal to \mathbf{p} as \mathbf{d} increases and infinitely larger!!!

Source: N Vasconcelos

Wait, there is more!

- Consider the crust of unit hypersphere of thickness ϵ
- Let's compute the ratio of volumes

$$\frac{Vol(S_1)}{Vol(S_2)} = \frac{\frac{(a-\epsilon)^d \pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2}+1)}}{\frac{a^d \pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2}+1)}} = \frac{a^d \left(1 - \frac{\epsilon}{a}\right)^d}{a^d} = \left(1 - \frac{\epsilon}{a}\right)^d$$



- No matter how small ϵ is, ratio goes to zero as d increases
- i. e. "**all the volume is in the crust!**"

The Curse of Dimensionality

DR Methods: Categorization

- **Unsupervised**

- Latent Semantic Indexing (LSI): truncated SVD
- Independent Component Analysis (ICA)
- Principal Component Analysis (PCA)
- Canonical Correlation Analysis (CCA)

- **Supervised**

- Linear Discriminant Analysis (LDA)

- **Semi-supervised**

- Research topic

DR Methods

- **Linear**

- Latent Semantic Indexing (LSI): truncated SVD
- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Canonical Correlation Analysis (CCA)

- **Non-linear**

- Nonlinear feature reduction using kernels
- Manifold learning

Outline

- Linear Methods
 - Principal Component Analysis (PCA)
 - Fisher (Linear) Discriminant Analysis (LDA)
 - Multi-Dimensionality Scaling (MDS)
- Non-linear Methods
 - Kernel PCA
 - Manifold Learning methods: ISOMAP, LLE, Laplacian Eigenmaps
 - t-Stochastic Neighborhood Embedding (t-SNE)

Principal Component Analysis (PCA)

- Find a low-dimensional space such that when \mathbf{x} is projected there, information loss is minimized.
- The projection of \mathbf{x} on the direction of \mathbf{w} is: $z = \mathbf{w}^T \mathbf{x}$
- Find \mathbf{w} such that $\text{Var}(z)$ is maximized

$$\begin{aligned}\text{Var}(z) &= \text{Var}(\mathbf{w}^T \mathbf{x}) = E[(\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \boldsymbol{\mu})^2] \\ &= E[(\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \boldsymbol{\mu})(\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \boldsymbol{\mu})] \\ &= E[\mathbf{w}^T (\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{w}] \\ &= \mathbf{w}^T E[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T] \mathbf{w} = \mathbf{w}^T \Sigma \mathbf{w}\end{aligned}$$

where $\text{Var}(\mathbf{x}) = E[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T] = \Sigma$

PCA

- Maximize $\text{Var}(z)$ subject to $\|\mathbf{w}\|=1$

$$\max_{\mathbf{w}_1} \mathbf{w}_1^T \Sigma \mathbf{w}_1 - \alpha(\mathbf{w}_1^T \mathbf{w}_1 - 1)$$

Using Lagrange multipliers, a la SVM

$\Sigma \mathbf{w}_1 = \alpha \mathbf{w}_1$ that is, \mathbf{w}_1 is an eigenvector of Σ

Choose the one with the largest eigenvalue for $\text{Var}(z)$ to be max

- Second principal component: Max $\text{Var}(z_2)$, s.t., $\|\mathbf{w}_2\|=1$ and orthogonal to \mathbf{w}_1

$$\max_{\mathbf{w}_2} \mathbf{w}_2^T \Sigma \mathbf{w}_2 - \alpha(\mathbf{w}_2^T \mathbf{w}_2 - 1) - \beta(\mathbf{w}_2^T \mathbf{w}_1 - 0)$$

How?

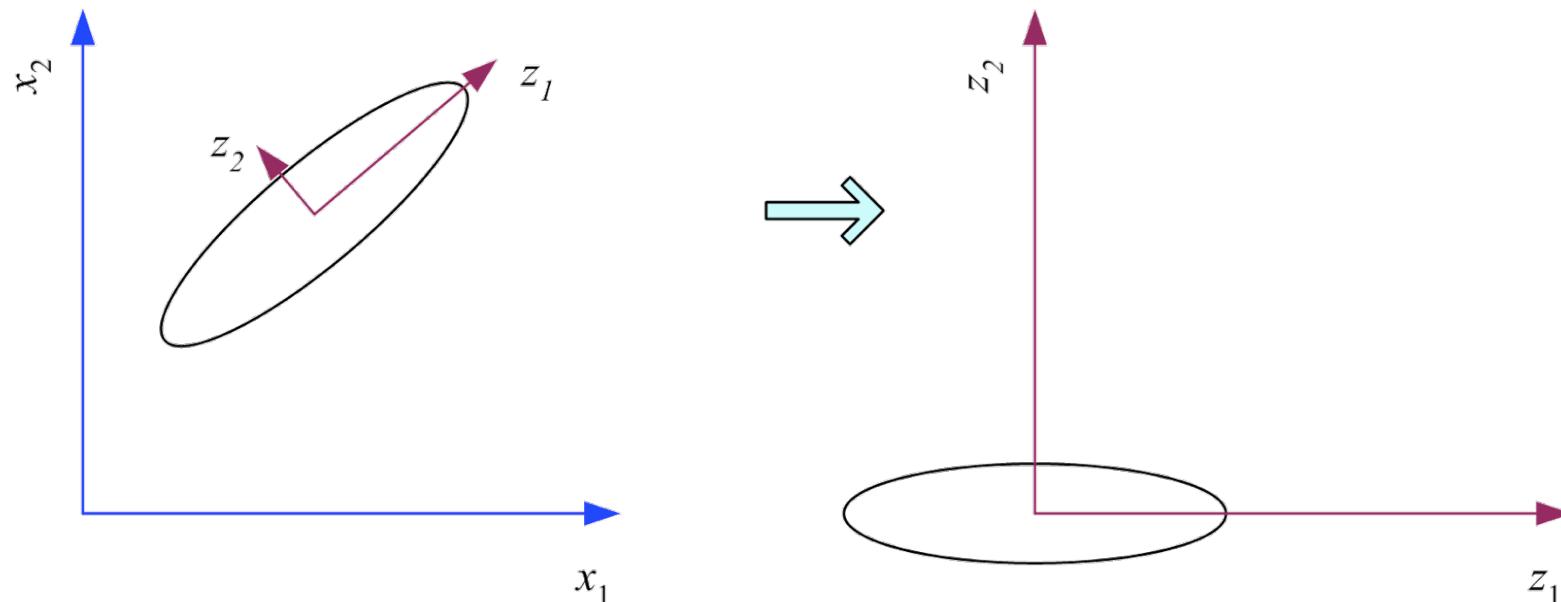
$\Sigma \mathbf{w}_2 = \alpha' \mathbf{w}_2$ that is, \mathbf{w}_2 is another eigenvector of Σ

and so on.

PCA

$$\mathbf{z} = \mathbf{W}^T(\mathbf{x} - \mathbf{m})$$

where the columns of \mathbf{W} are the eigenvectors of Σ , and \mathbf{m} is sample mean; Centers the data at the origin and rotates the axes



How to choose k

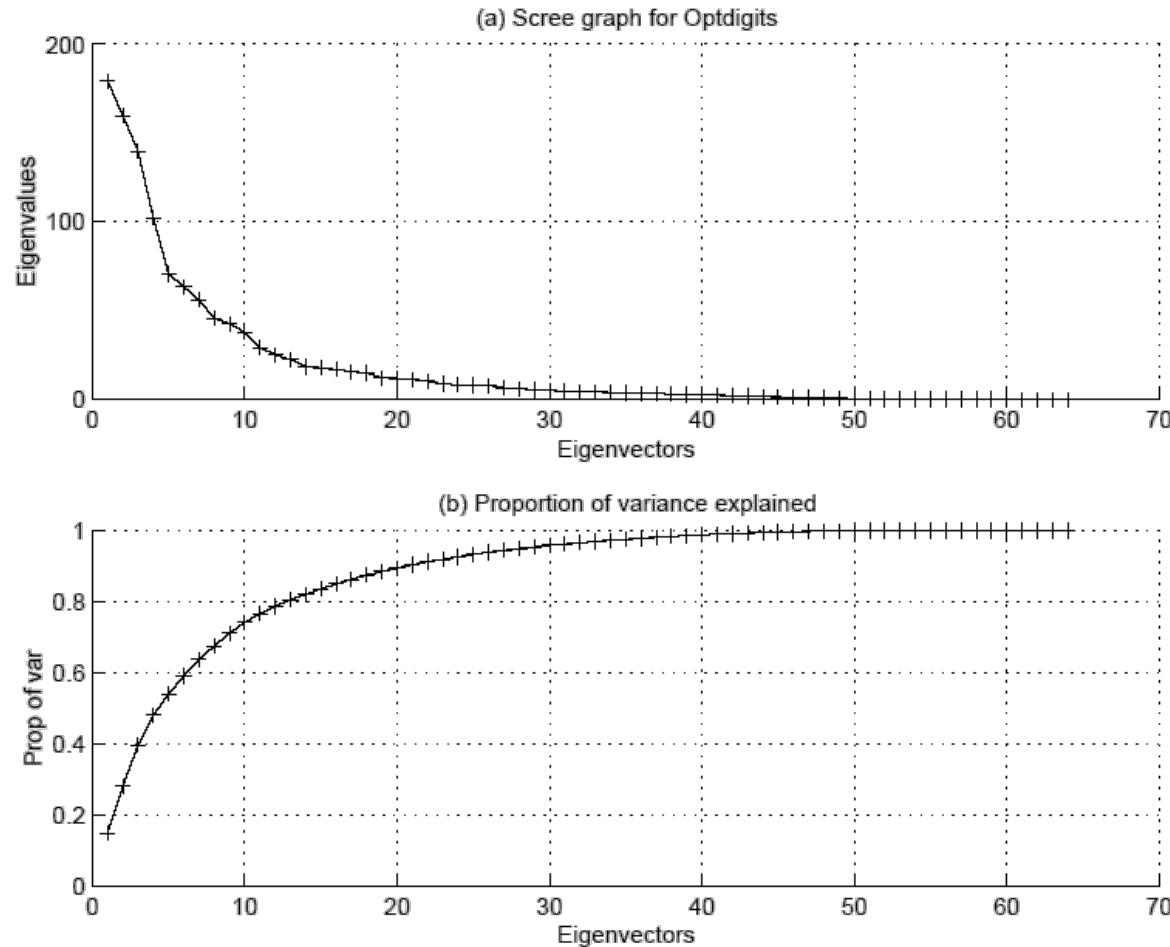
- Proportion of Variance (PoV) explained

$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

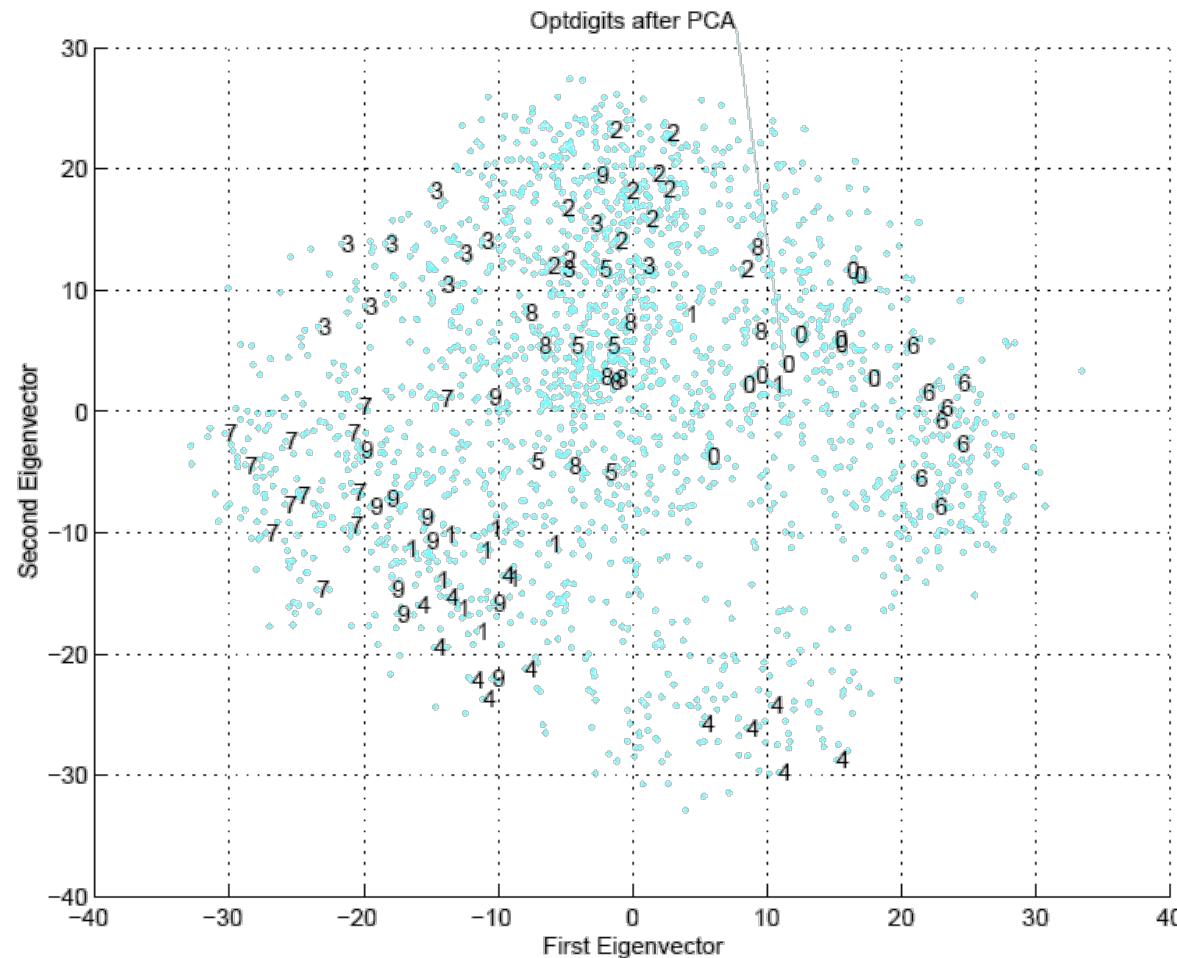
when λ_i are sorted in descending order

- Typically, stop at PoV > 0.9
- See graph plots of PoV vs k, stop at “elbow”

Illustration



Example



Outline

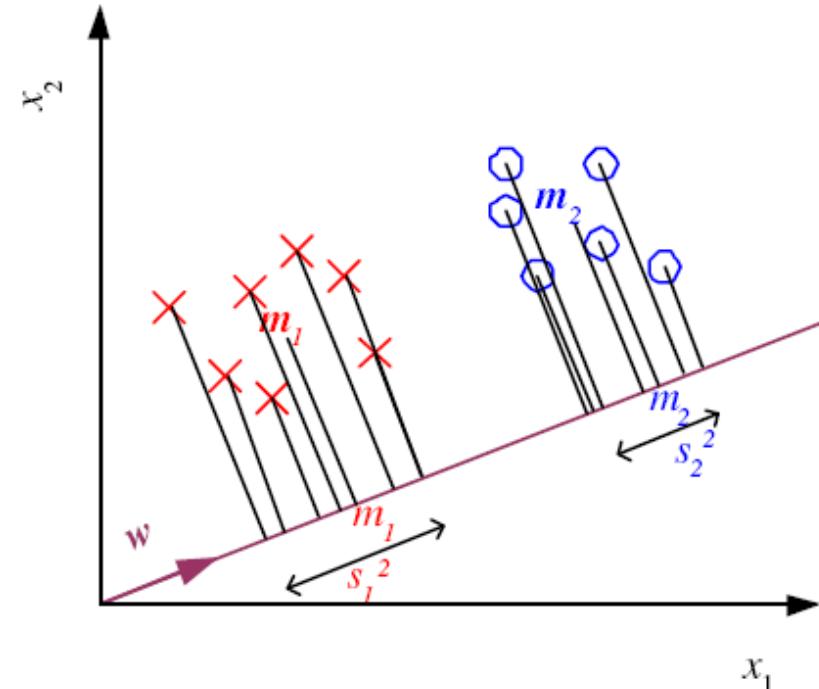
- Linear Methods
 - Principal Component Analysis (PCA)
 - Fisher (Linear) Discriminant Analysis (LDA)
 - Multi-Dimensionality Scaling (MDS)
- Non-linear Methods
 - Kernel PCA
 - Manifold Learning methods: ISOMAP, LLE, Laplacian Eigenmaps
 - t-Stochastic Neighborhood Embedding (t-SNE)

Linear Discriminant Analysis

- Supervised linear DR method
- Find a low-dimensional space such that when \mathbf{x} is projected, classes are well-separated.
- Find \mathbf{w} that maximizes

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2}$$

$$m_1 = \frac{\sum_t \mathbf{w}^T \mathbf{x}^t r^t}{\sum_t r^t} \quad s_1^2 = \sum_t (\mathbf{w}^T \mathbf{x}^t - m_1)^2 r^t$$



Linear Discriminant Analysis

- Between-class scatter:

$$\begin{aligned}(m_1 - m_2)^2 &= (\mathbf{w}^T \mathbf{m}_1 - \mathbf{w}^T \mathbf{m}_2)^2 \\&= \mathbf{w}^T (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T \mathbf{w} \\&= \mathbf{w}^T \mathbf{S}_B \mathbf{w} \text{ where } \mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T\end{aligned}$$

- Within-class scatter:

$$\begin{aligned}s_1^2 &= \sum_t (\mathbf{w}^T \mathbf{x}^t - m_1)^2 r^t \\&= \sum_t \mathbf{w}^T (\mathbf{x}^t - \mathbf{m}_1)(\mathbf{x}^t - \mathbf{m}_1)^T \mathbf{w} r^t = \mathbf{w}^T \mathbf{S}_1 \mathbf{w} \\&\text{where } \mathbf{S}_1 = \sum_t (\mathbf{x}^t - \mathbf{m}_1)(\mathbf{x}^t - \mathbf{m}_1)^T r^t \\s_1^2 + s_2^2 &= \mathbf{w}^T \mathbf{S}_W \mathbf{w} \text{ where } \mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2\end{aligned}$$

Fisher's Linear Discriminant

- Find \mathbf{w} that max

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} = \frac{\left| \mathbf{w}^T (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T \mathbf{w} \right|}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

- LDA soln:

$$\mathbf{w} = c \cdot \mathbf{S}_W^{-1} (\mathbf{m}_1 - \mathbf{m}_2)$$

- Alternative parametric soln:

$$\mathbf{w} = \Sigma^{-1} (\mu_1 - \mu_2)$$

when $p(\mathbf{x} | C_i) \sim \mathcal{N}(\mu_i, \Sigma)$

Fisher's Linear Discriminant

- Within-class scatter:

$$\mathbf{S}_w = \sum_{i=1}^K \mathbf{S}_i \quad \mathbf{S}_i = \sum_t r_i^t (\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T$$

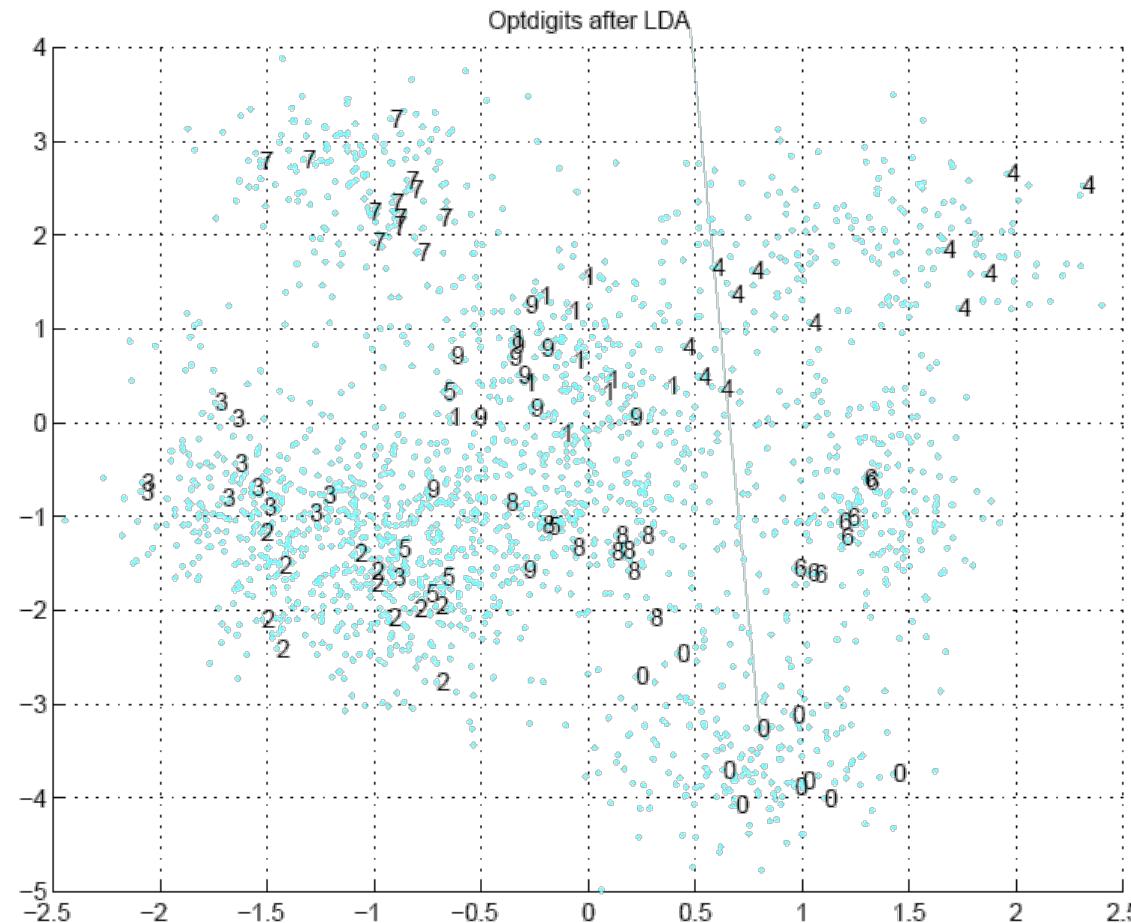
- Between-class scatter:

$$\mathbf{S}_B = \sum_{i=1}^K N_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T \quad \mathbf{m} = \frac{1}{K} \sum_{i=1}^K \mathbf{m}_i$$

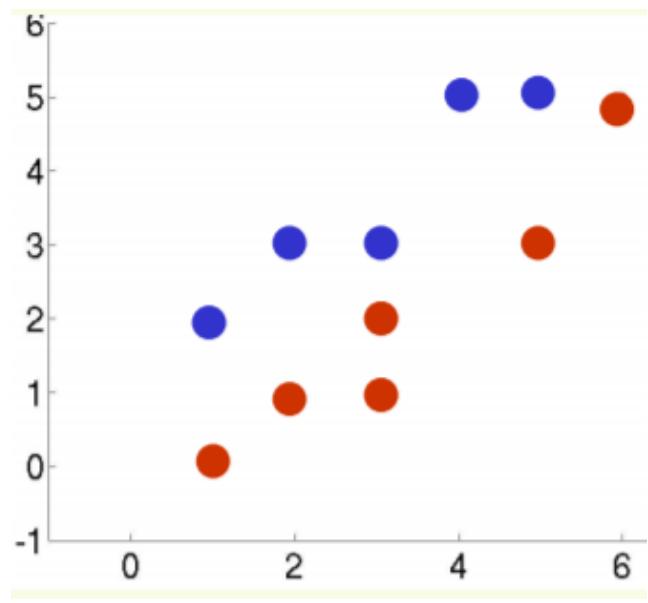
- Find \mathbf{W} that max

$$J(\mathbf{W}) = \frac{|\mathbf{W}^T \mathbf{S}_B \mathbf{W}|}{|\mathbf{W}^T \mathbf{S}_w \mathbf{W}|} \quad \text{The largest eigenvectors of } \mathbf{S}_w^{-1} \mathbf{S}_B$$

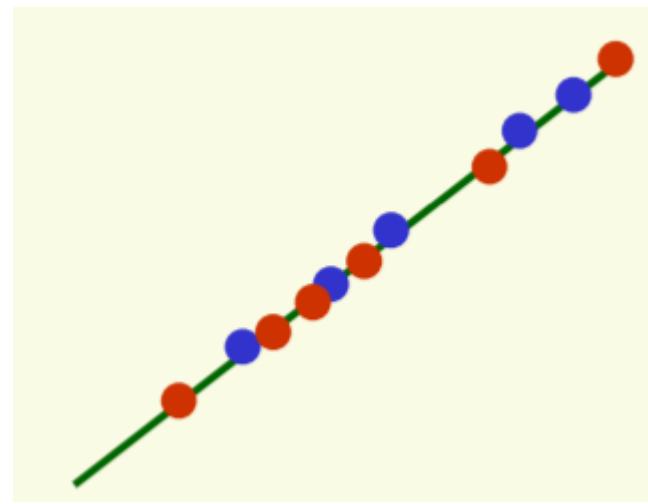
LDA: Illustration



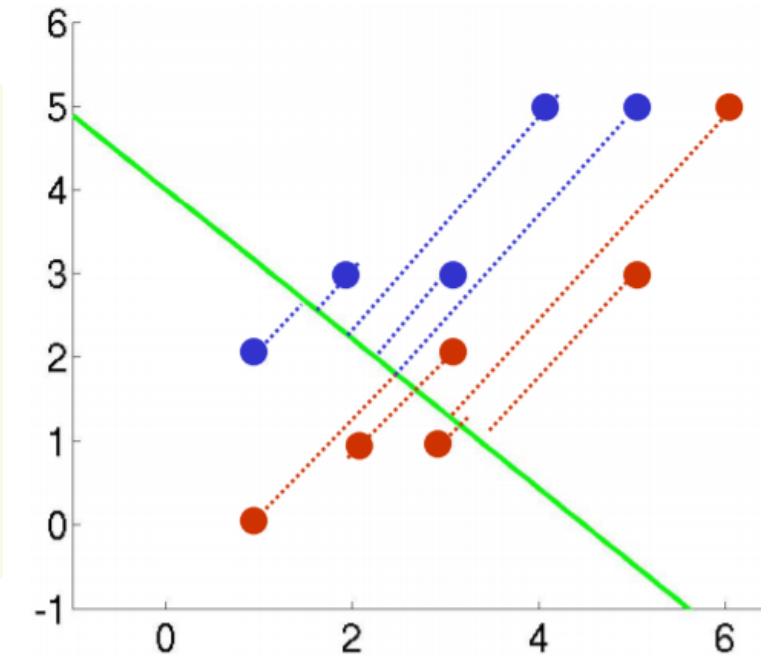
LDA: Illustration



Data



PCA projection



LDA projection

Outline

- Linear Methods
 - Principal Component Analysis (PCA)
 - Fisher (Linear) Discriminant Analysis (LDA)
 - Multi-Dimensionality Scaling (MDS)
- Non-linear Methods
 - Kernel PCA
 - Manifold Learning methods: ISOMAP, LLE, Laplacian Eigenmaps
 - t-Stochastic Neighborhood Embedding (t-SNE)

Multi-Dimensional Scaling

- History: T. Cox and M. Cox, 2001
- Attempts to preserve pairwise distances
- Different formulation of PCA, but yields similar result form

$$\min_Y \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^{(X)} - d_{ij}^{(Y)})^2 \quad \text{where } d_{ij}^{(X)} = \|x_i - x_j\|^2 \text{ and } d_{ij}^{(Y)} = \|y_i - y_j\|^2.$$

- Transformation

$$X^\top X = -\frac{1}{2} H D^{(X)} H \quad \text{where } H = I - \frac{1}{N} \mathbf{1}\mathbf{1}^\top.$$

$$\min_Y \sum_{i=1}^N \sum_{j=1}^N (x_i^\top x_j - y_i^\top y_j)^2$$

Source: Haiqin Yang

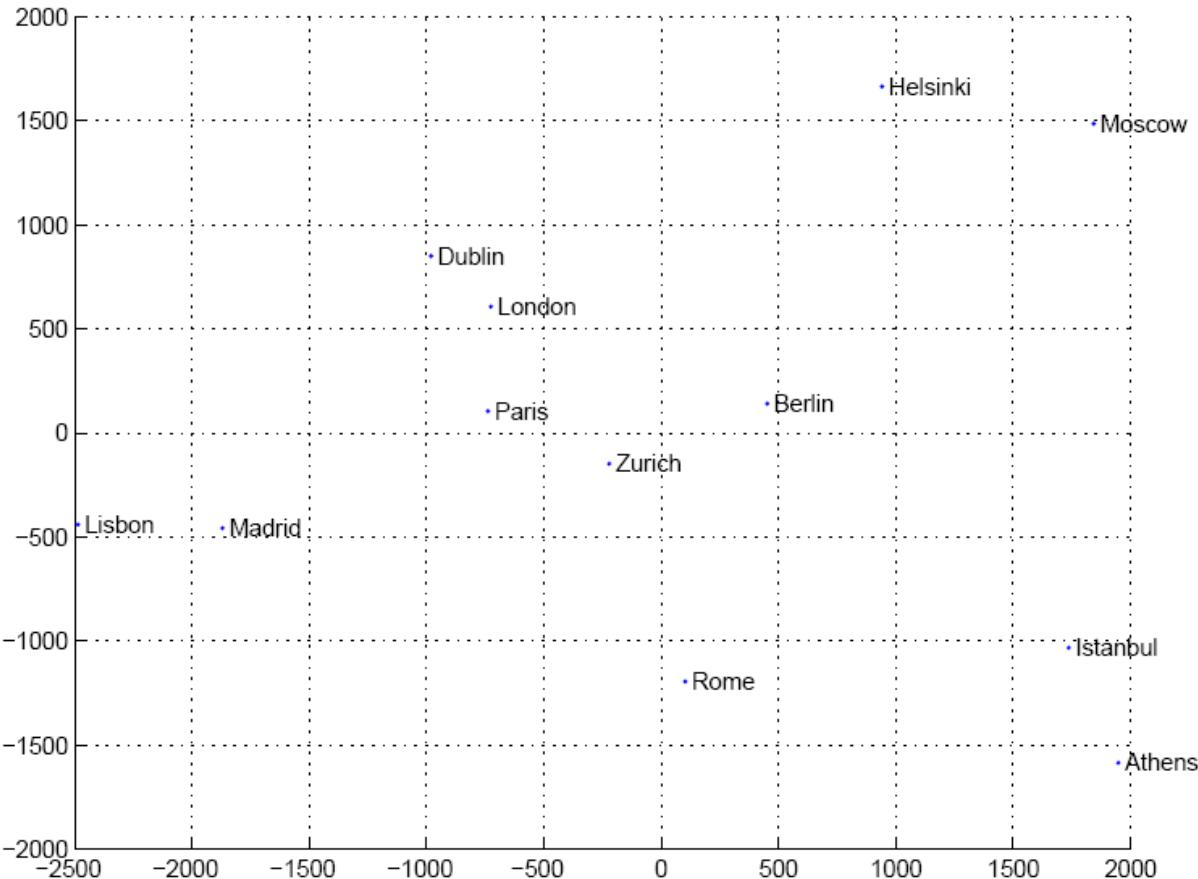


4-Nov-17

CS6510 - Applied Machine Learning

33

MDS: Illustration



Outline

- Linear Methods
 - Principal Component Analysis (PCA)
 - Fisher (Linear) Discriminant Analysis (LDA)
 - Multi-Dimensionality Scaling (MDS)
- Non-linear Methods
 - Kernel PCA
 - Manifold Learning methods: ISOMAP, LLE, Laplacian Eigenmaps
 - t-Stochastic Neighborhood Embedding (t-SNE)

Kernel PCA

- History: S. Mika et al, NIPS, 1999
- Data may lie on or near a nonlinear manifold, not a linear subspace
- Find principal components that are nonlinearly to the input space via nonlinear mapping $x \mapsto \Phi(x)$
- Objective $\min_{U_k} \sum_{i=1}^N \|\Phi(\mathbf{x}_i) - U_k U_k^T \Phi(\mathbf{x}_i)\|^2$
- Solution found by SVD: $\Phi(X) = U \Sigma V^T$
 U contains the eigenvectors of $\Phi(X)\Phi(X)^T$

Source: Haiqin Yang



4-Nov-17

CS6510 - Applied Machine Learning

36

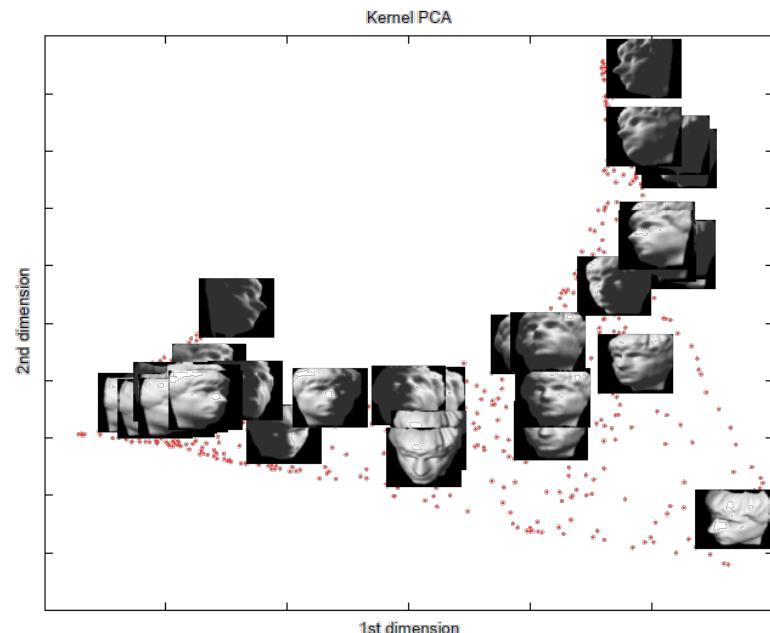
Kernel PCA

- Centering

$$\tilde{\Phi}(X) = \Phi(X) - E_x[\Phi(X)] \quad \tilde{K}(x, y) = \tilde{\Phi}(x)\tilde{\Phi}(y)$$

$$\begin{aligned}\tilde{K}(x, y) &= (\Phi(x) - E_x[\Phi(x)]).(\Phi(y) - E_y[\Phi(y)]) \\ &= K(x, y) - E_x[K(x, y)] - E_y[K(x, y)] + E_x[E_y[K(x, y)]]\end{aligned}$$

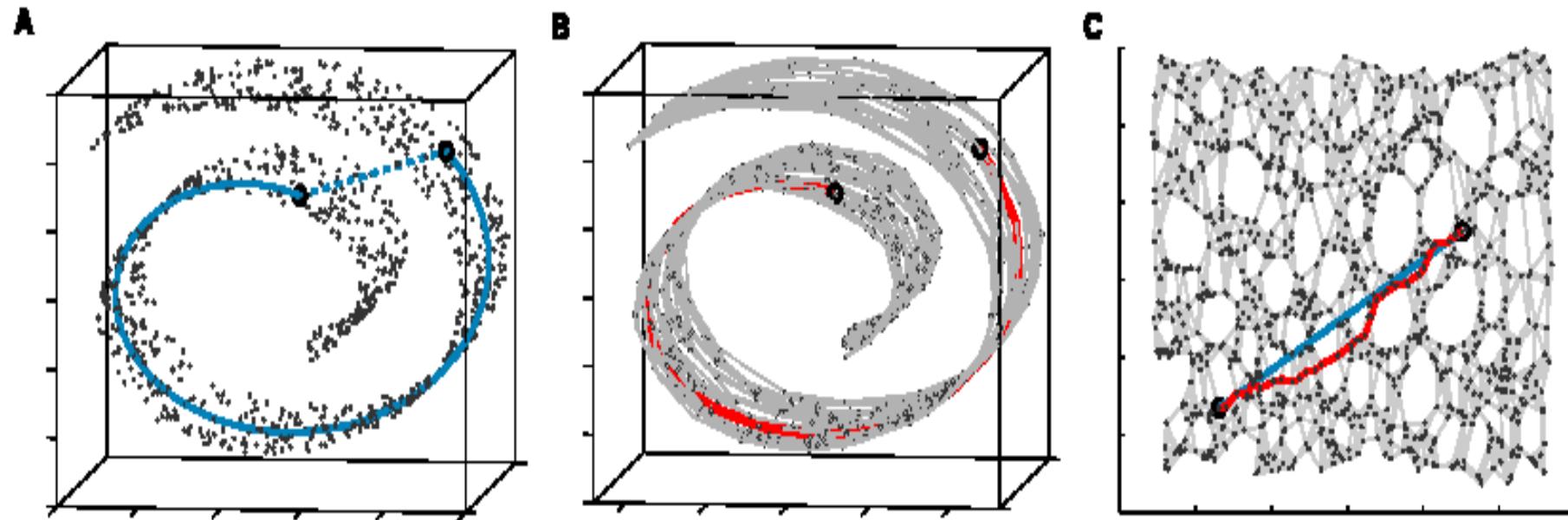
- Issue: Difficult to reconstruct



Outline

- Linear Methods
 - Principal Component Analysis (PCA)
 - Fisher (Linear) Discriminant Analysis (LDA)
 - Multi-Dimensionality Scaling (MDS)
- Non-linear Methods
 - Kernel PCA
 - Manifold Learning methods: ISOMAP, LLE, Laplacian Eigenmaps
 - t-Stochastic Neighborhood Embedding (t-SNE)

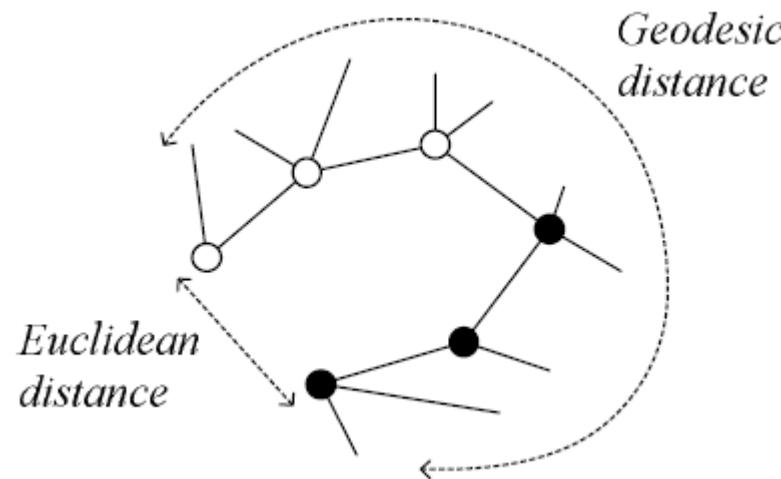
Manifold Learning



A manifold is a topological space which is locally Euclidean.

Isomap

- Geodesic distance is the distance along the manifold that the data lies in, as opposed to the Euclidean distance in the input space

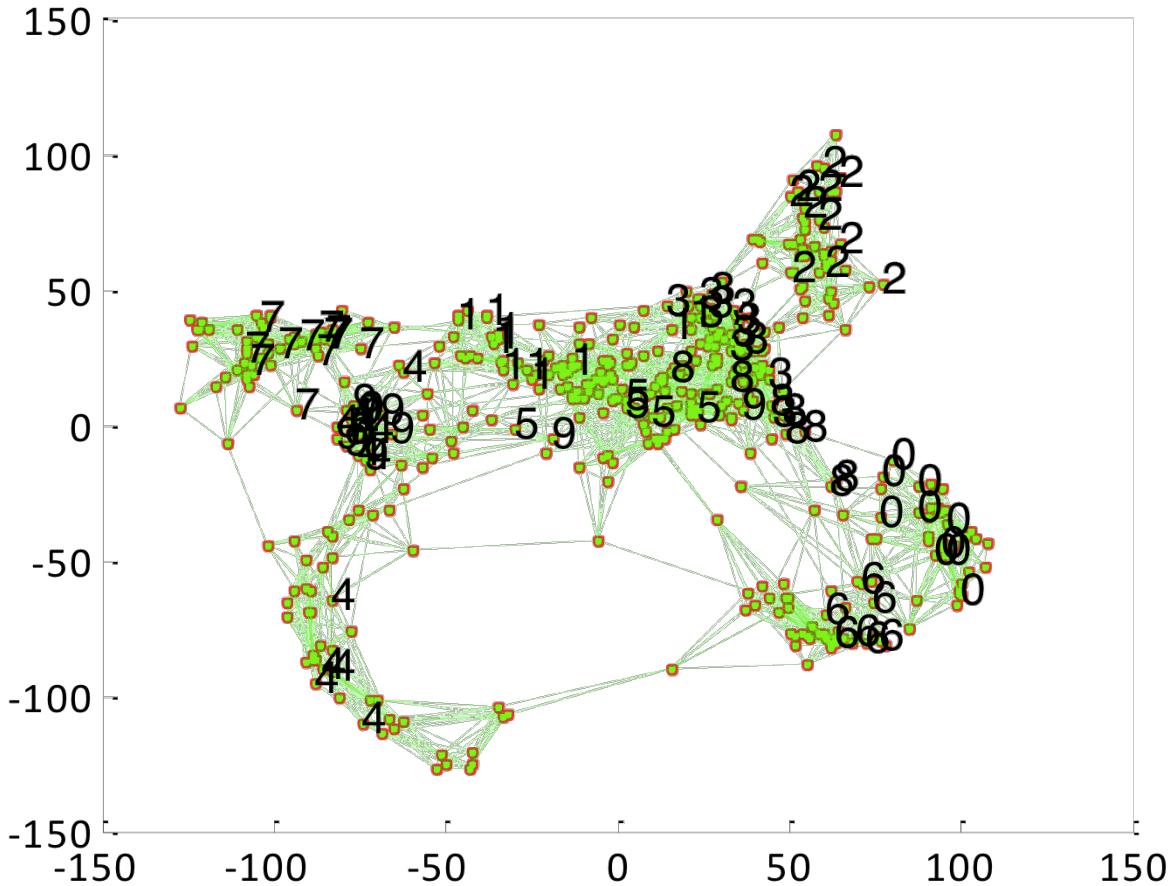


Isomap

- Instances r and s are connected in the graph if $\|x^r - x^s\| < e$ or if x^s is one of the k neighbors of x^r . The edge length is $\|x^r - x^s\|$
- For two nodes r and s not connected, the distance is equal to the shortest path between them
- Once the $N \times N$ distance matrix is thus formed, use MDS to find a lower-dimensional mapping

Isomap: Illustration

Optdigits after Isomap (with neighborhood graph).



Locally Linear Embedding (LLE)

- History: S. Roweis and L. Saul, Science, 2000
- Procedure
 1. Identify the neighbors of each data point
 2. Compute weights that best linearly reconstruct the point from its neighbors

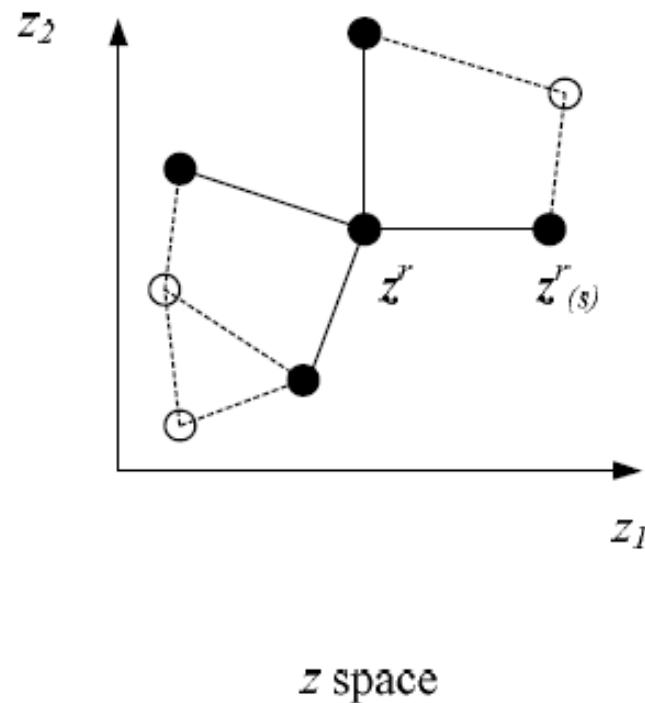
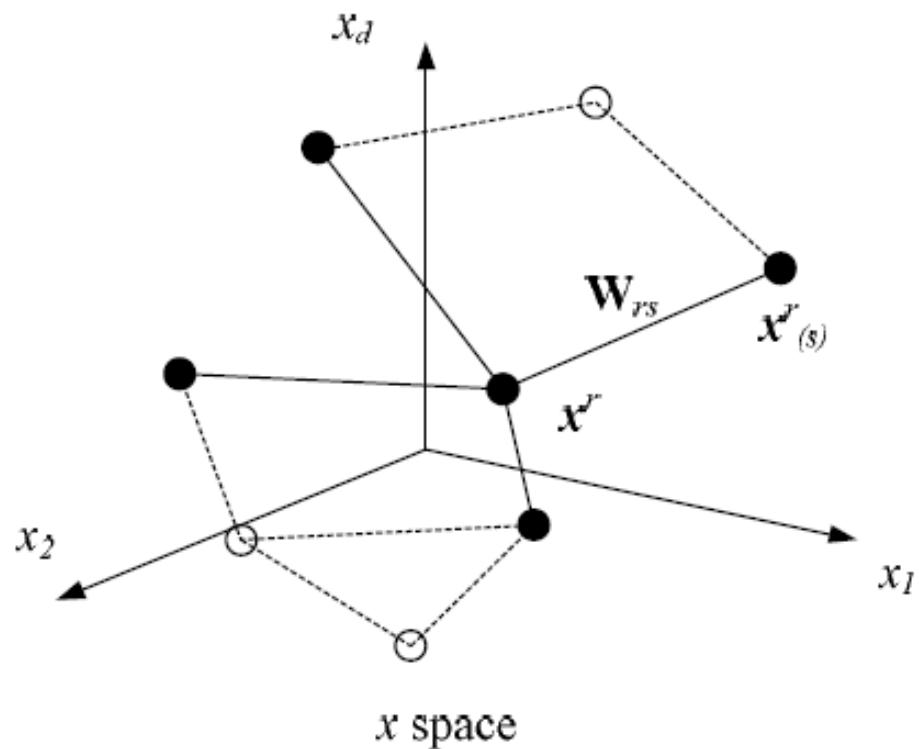
$$\min_{\mathbf{w}} \sum_{i=1}^N \left\| \mathbf{x}_i - \sum_{j=1}^k w_{ij} \mathbf{x}_{N_i(j)} \right\|^2$$

- 3. Find the low-dimensional embedding vector which is best reconstructed by the weights determined in Step 2

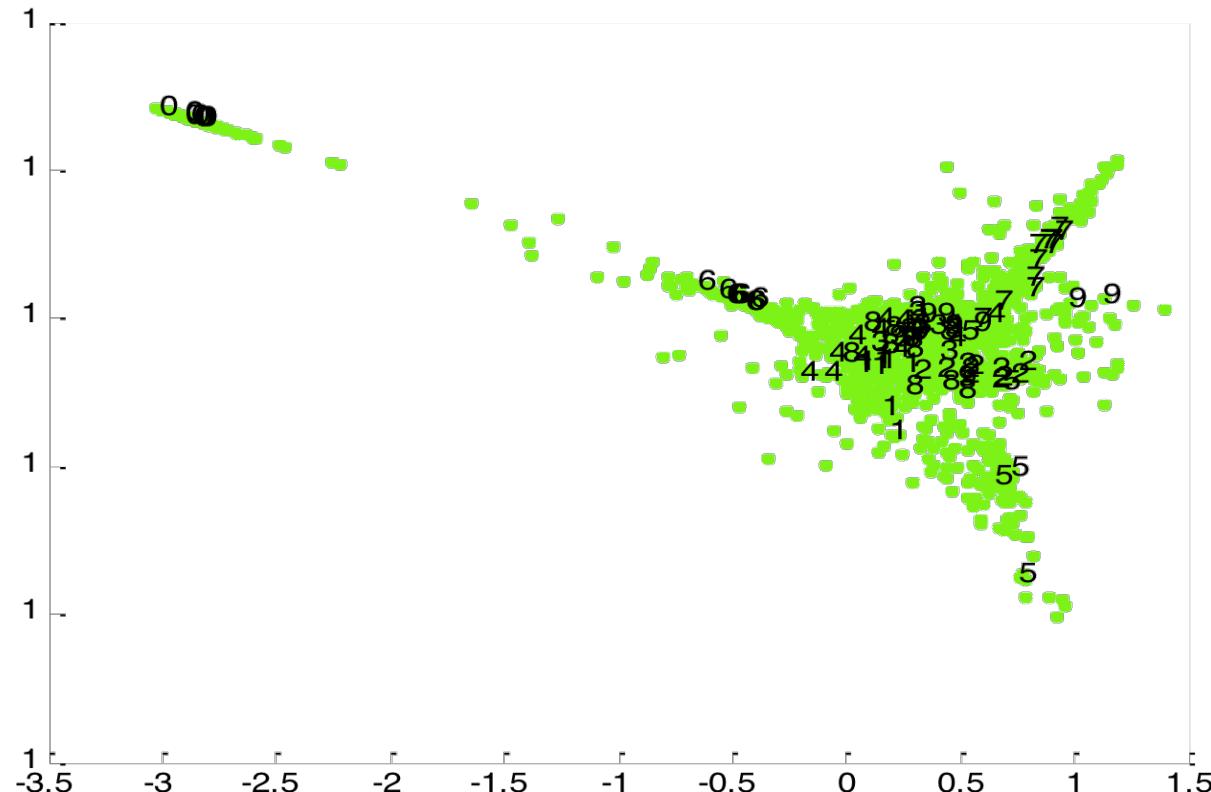
$$\min_Y \sum_{i=1}^N \left\| \mathbf{y}_i - \sum_{j=1}^k w_{ij} \mathbf{y}_{N_i(j)} \right\|^2 \iff \min_Y \text{tr}(Y^\top Y L)$$

Centering \mathbf{Y} with unit variance

LLE



LLE: Illustration



Laplacian Eigenmaps

- History: M. Belkin and P. Niyogi, 2003
- Similar to locally linear embedding
- Different in weights setting and objective function

- Weights

$$W_{ij} = \begin{cases} 1 & i, j \text{ are connected} \\ \exp\left(\frac{-\|x_i - x_j\|^2}{s}\right) & \text{otherwise} \end{cases}$$

- Objective

$$\min_Y \sum_{i=1}^N \sum_{j=1}^N (\mathbf{y}_i - \mathbf{y}_j) W_{ij} \iff \min_Y \text{tr}(YLY^\top)$$

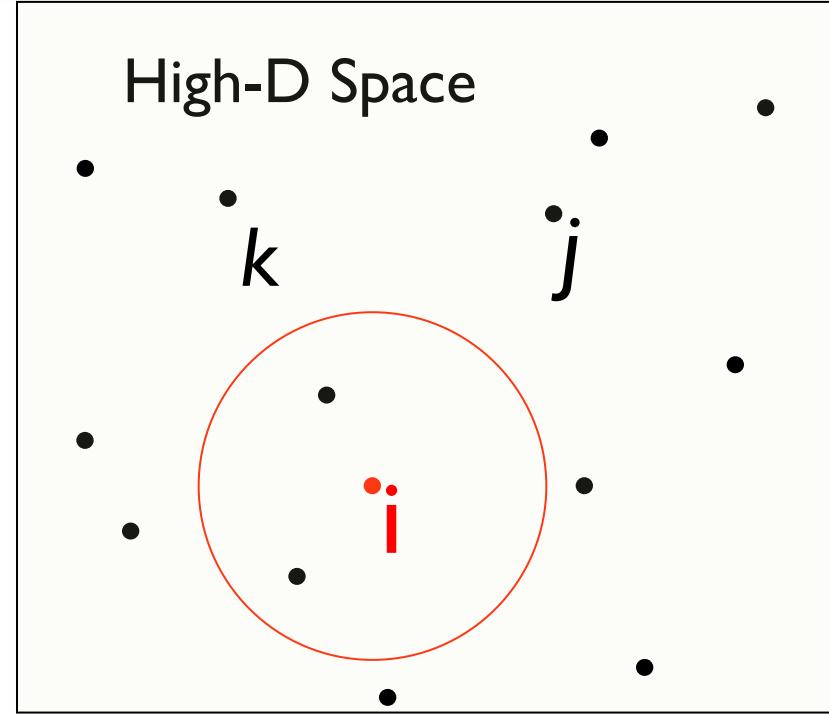
where $L = R - W$, R is diagonal and $R_{ii} = \sum_{j=1}^N W_{ij}$.

Outline

- Linear Methods
 - Principal Component Analysis (PCA)
 - Fisher (Linear) Discriminant Analysis (LDA)
 - Multi-Dimensionality Scaling (MDS)
- Non-linear Methods
 - Kernel PCA
 - Manifold Learning methods: ISOMAP, LLE, Laplacian Eigenmaps
 - t-Stochastic Neighborhood Embedding (t-SNE)

Stochastic Neighborhood Embedding

- First convert each high-dimensional similarity into the probability that one data point will pick the other data point as its neighbor.
- To evaluate a map:
 - Use the pairwise distances in the low-dimensional map to define the probability that a map point will pick another map point as its neighbor.
 - Compute the Kullback-Leibler divergence between the probabilities in the high-dimensional and low-dimensional spaces.



$$p_{j|i} = \frac{e^{-d_{ij}^2/2\sigma_i^2}}{\sum_k e^{-d_{ik}^2/2\sigma_i^2}}$$

probability of picking j
given that you start at i

SNE

$$Cost = \sum_i KL(P_i \| Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

Q is the probability of picking j
when you start at i in low-D space

- For points where p_{ij} is large and q_{ij} is small we lose a lot.
 - Nearby points in high-D really want to be nearby in low-D
- For points where q_{ij} is large and p_{ij} is small we lose a little because we waste some of the probability mass in the Q_i distribution.
 - Widely separated points in high-D have a mild preference for being widely separated in low-D.

Readings

- “Introduction to Machine Learning” by Ethem Alpaydin, Chapter 6
- Additional
 - More about t-SNE: <https://lvdmaaten.github.io/tsne/>
 - Dimensionality Reduction:A Comparative Review (a very good read)
 - A toolbox for 34 dimensionality reduction methods:
<https://lvdmaaten.github.io/drtoolbox/> (Matlab)