INFO411 Lecture 4 - Dimension Reduction

Jeremiah Deng

University of Otago

1/8/2017

Outline

Motivations & Benefits

- Reduce the dimension of the data in a linear or non-linear fashion
  - Remove redundancy and noisy information
  - Improve algorithm efficiency and learning outcome
- Identify abstract variables which have generated the inter-instance similarity
  - Better understanding of the data
- Reproduce non-linear higher-dimensional structures on a lower-dimensional display for visualization

Related Topics

- Clustering
- Classification
- Regression
- Data compression
- Feature extraction
- Data visualization

Maths warm-up

- Matrix multiplication $\mathbf{X}_{NM} \times \mathbf{Y}_{MK} = \mathbf{Z}_{NK}$

$$Z_{ij} = \sum_m X_{im} Y_{mk}$$

- $\mathbf{AB} \neq \mathbf{BA}$
- $(\mathbf{A}^T)^T = \mathbf{A}$
- $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$
- $(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$
- $(\mathbf{A}+\mathbf{B})\mathbf{C} = \mathbf{AC}+\mathbf{BC}$

Multidimensional Scaling

- Data are usually of high-dimensional space.
- Data points are similar or dissimilar to each other.
- We assess closeness mainly on a 2-D or 3-D “mental” space.
- MDS: produce projection into lower display space while keeping similar/distance between data points.

Metric MDS

- Projection: $X \to X'$
- Distances between data items are given, a configuration of points which gives rise to those distances is sought
- Can be used for non-linear projection
- Tries to maintain dissimilarities (distances) between data points
  - Original distance: $d(k,l)$
  - In projected space: $d'(k,l)$
- Objective function to minimize: e.g.

$$E_M = \sum_{k \neq l} [d(k,l) - d'(k,l)]^2$$

Sammon's Mapping

- Closely related to metric MDS
- Tries to *preserve* pairwise distances in the projected space
- Errors in distance preservation are normalized
- Objective function (aka *‘stress’*): $E_M = \sum_{k \neq l} \frac{[d(k,l) - d'(k,l)]^2}{d(k,l)}$
- Minimization can be done by gradient descent.

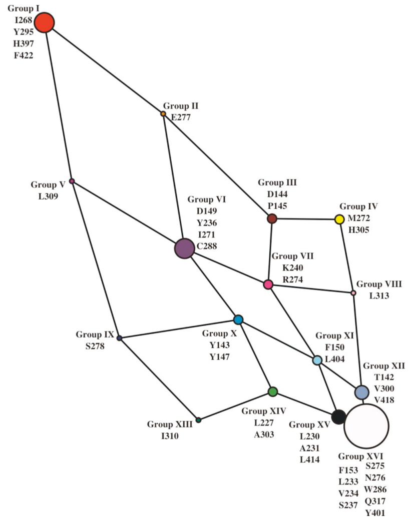
  Implications?

  Local minima!

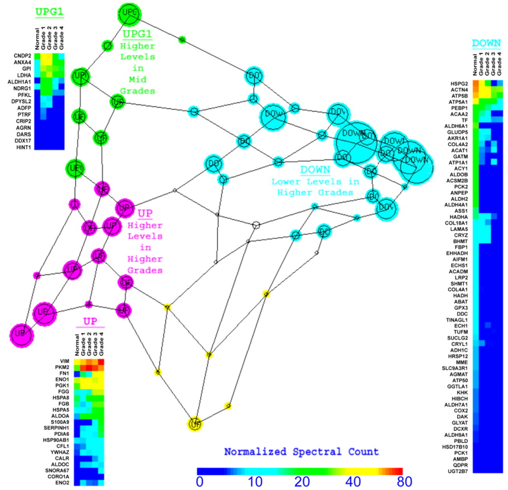
Self-Organizing Maps

- An algorithm that performs clustering and non-linear projection onto lower dimension at the same time
- Finds and orders a set of reference vectors located on a discrete lattice
- Learning rule:
  $\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \gamma(t) h_{ci}(t)(\mathbf{x} - \mathbf{m}_i)$
  $h_{ci}()$: neighbour function centred at BMU c
- Nice properties:
  - Low-dimensional grids ready for display
  - Topology preservation
  - Probability density matching

SOM with Sammon's projection



doi: 10.1074/jbc.M513609200



doi: 10.1074/mcp.M800252-MCP200

Principal Component Analysis

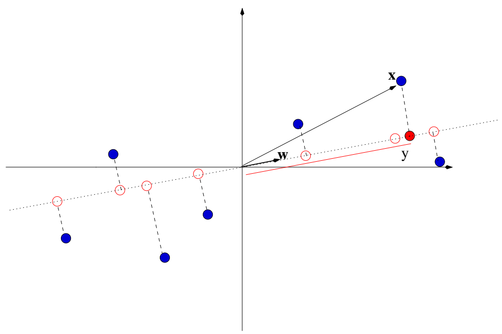
- A standard statistical method.
- Applied in data compression, feature extraction & visualization.
- Also known as Karhunen-Loeve transform in signal processing, or the Hotelling transform in image processing.

PCA explained - a 2-D example

- Given a set of points $\{\mathbf{x}\}$ on a 2-D plane.
- Assume it's zero-meaned.
- Use orthogonal transform so reconstruction is easy.
- Goal: Find an optimal projection $y = \mathbf{w}^T\mathbf{x}$, subject to $\|\mathbf{w}\| = 1$.
- Reconstruction: $\mathbf{x}' = y\mathbf{w}$
- Criterion: For best reconstruction with minimum reconstruction error
- Solution: y should take on variance as large as possible.

Best Representation in Reduced Form

- A set of zero-centered data points
- With a vector $\mathbf{w}$, 1-D projection of data points in $\{\mathbf{x}\}$: $y = \mathbf{w}^T\mathbf{x}$
- Use y to *represent* $\mathbf{x}$
- Question: What is the best projection vector, subject to $\|\mathbf{w}\| = 1$: best keeping variation, with least distortion?



The Optimization Process

- Our goal is to minimize the reconstruction error:

$$\begin{aligned} J &= E\{\|\mathbf{x} - y\mathbf{w}\|^2\} = E\{(\mathbf{x} - y\mathbf{w})^T(\mathbf{x} - y\mathbf{w})\} \\ &= E\{\mathbf{x}^T\mathbf{x}\} - E\{y\mathbf{w}^T\mathbf{x}\} - E\{y\mathbf{x}^T\mathbf{w}\} + E\{y^2\mathbf{w}^T\mathbf{w}\} \\ &= E\{\|\mathbf{x}\|^2\} - E(y^2). \end{aligned}$$

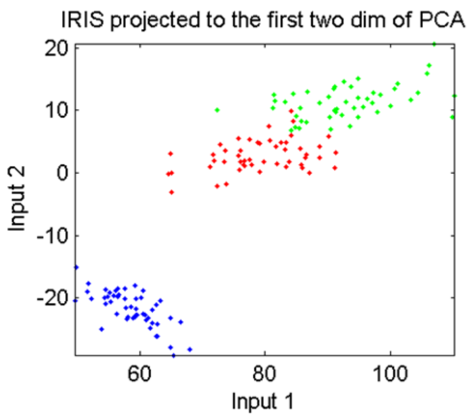
  Indeed, minimization of reconstruction error is equivalent to maximization of the projection variance.
- Use a Lagrange to maximize $J' = E(y^2) - \lambda(\|\mathbf{w}\|^2 - 1)$: i.e., $J' = E\{\mathbf{w}^T\mathbf{x}\mathbf{x}^T\mathbf{w}\} - \lambda(\mathbf{w}^T\mathbf{w} - 1)$
- To find the optimal $\mathbf{w}$:

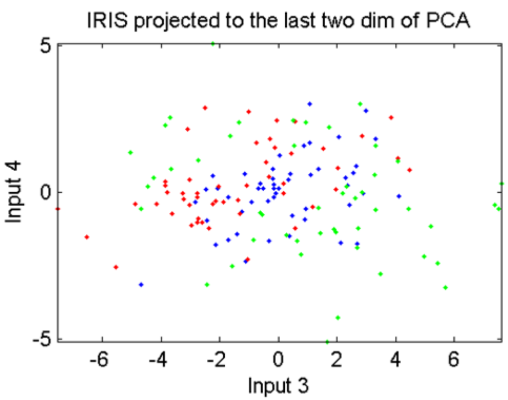
$$\frac{\delta J'}{\delta \mathbf{w}} = 0 \Rightarrow E\{\mathbf{x}\mathbf{x}^T\mathbf{w}\} - \lambda\mathbf{w} = 0 \Rightarrow \mathbf{R}\mathbf{w} = \lambda\mathbf{w}$$

Eigenvectors and eigenvalues

- $\mathbf{R} = E\{\mathbf{x}\mathbf{x}^T\}$ is the covariance matrix of data $X \in \mathbb{R}^N$.
- $\mathbf{R}\mathbf{w} = \lambda\mathbf{w}$ suggests that the optimal $\mathbf{w}$ is the eigenvector of $\mathbf{R}$, with λ as the relevant eigenvalue.
- The projection onto the eigenvector, $y = \mathbf{w}^T\mathbf{x}$, is called the principal component.
- Preserved variance: $E(y^2) = \lambda$
- Matrix $\mathbf{R}$ is positive semi-definite, and there usually exist N eigenvectors with positive eigenvalues.
- If we pick the first k principal components (with the largest eigenvalues), the proportion of variance kept is $\frac{\lambda_1+\lambda_2+\cdots+\lambda_k}{\lambda_1+\lambda_2+\cdots+\lambda_N}$.

An Example - Iris data





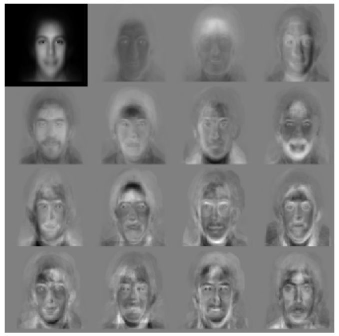
- As the principal components $y_i = \mathbf{w}_i^T\mathbf{x}$ (i=1,2,...) preserve major variances, they can normally *explain* the data better than *minor* components.

PCA: How To

- Given a data set $\mathbf{X}_{n\times d}$, make its columns zero-meaned.
- Compute the covariance matrix - $\mathbf{R}_{d\times d}$
- Conduct eigenanalysis of $\mathbf{R}$ and get first m eigenvectors $\mathbf{e_1}, \mathbf{e_2}, \cdots, \mathbf{e_m}$ (with eigenvalues sorted in decreasing order) as column vectors to form the transform matrix $\mathbf{E}$
- New data becomes $\mathbf{y} = \mathbf{XE}$.
- Question: How to decide m?
  - Look for elbow point on the eigenvalue scree plot
  - Or include principal components so a large extent e.g. 95% of the variance is kept.

"Eigenface" for face recognition

- MIT photobook experiment
- The eigenfaces for this database were approximated using a PCA on a representative sample of 128 faces.
- Recognition and matching was subsequently performed using the first 20 eigenvectors.
- Tests conducted on a database of 7,562 images of about 3,000 people.



Standard Eigenfaces used in photobook

Power Method

- Power Method: multiply a (column) weight vector $\mathbf{w}$ with the covariance matrix ($\mathbf{R}$) until convergence
  - $\mathbf{w} \leftarrow \mathbf{R}\mathbf{w}$, $\mathbf{w} \leftarrow \dfrac{\mathbf{w}}{\|\mathbf{w}\|}$
  - $\mathbf{w}$ converges to the first eigenvector $\mathbf{e}_1$
  - Other eigenvectors can be obtained progressively by deflating R:

$$\mathbf{R} \leftarrow \mathbf{R} - \lambda_1 \mathbf{e}_1^T \mathbf{e}_1$$

$$\mathbf{R} \leftarrow \mathbf{R} - \lambda_2 \mathbf{e}_2^T \mathbf{e}_2$$

$$\cdots$$

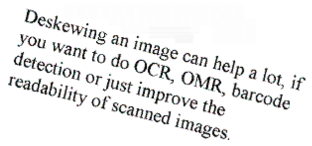
☺ Online implementation of PCA is possible.

Neural Networks for PCA

- A family of linear neural network models: GHA, APEX etc.
- All based on the Oja's rule using Hebbian learning: $\mathbf{w} \leftarrow \mathbf{w} + \gamma y(\mathbf{x} - y\mathbf{w})$, where $y = \mathbf{w}^T\mathbf{x}$.
- Network weights converge to eigenvectors asymptotically.
- Enable online eigen-analysis: no need to calculate R or conduct matrix eigen-analysis

There Is A Problem

- Maximizing variations on the p.c. does not necessarily increase the discriminant between classes!
- Some times minor components separate classes better than principal components!
- Example: OCR deskewing vs. line segmentation

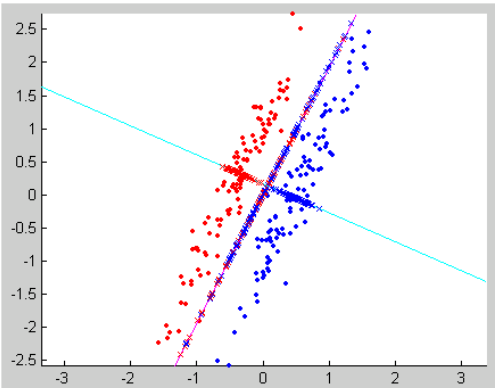


Fisher's Linear Discriminant Analysis

- Discriminant analysis seeks directions that are efficient for discrimination of patterns of different classes
- For a data set with two 'modes': $D = D_1 \cup D_2$, define 'scatter':
  - Within Class i: $S_i = \sum_{x \in D_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T$
  - Within class (total): $S_W = S_1 + S_2$
  - Between-class: $S_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$
- Linear projection: $y = \mathbf{w}^T \mathbf{x}$
- Forms two projection sets Y_1 and Y_2
  - Goal: maximize *new* S_B and minimize *new* S_W
  - Criterion function: $J(w) = \dfrac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}}$
  - Solution: $\mathbf{w} = S_B^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$

LDA vs PCA

- Red line: 1st eigenvector of PCA
- Cyan line: vector found by LDA
- Dots on the lines: projections
- Which projection gives a better classification potential?



Kernel Methods

- ☹ Nonlinearity: it is usually quite challenging to work with data in high-dimensional space.
- Kernel tricks: (nonlinearly) project data into a 'kernel space', where a kernel function k can be used to calculate dot product:
  $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$
- The kernel space is usually of high-dimensionality, but linear relations may exist among data
- Kernel transform can be carried out in the original feature space using the kernel function $k(.)$.
  - i.e., we don't even need to know $\phi()$!
- Popular kernels: Gaussian, polynomial, RBF ...

Kernel PCA

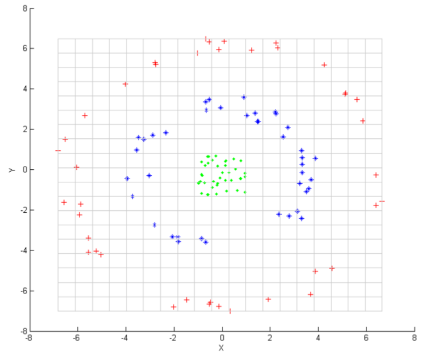
- KPCA: a nonlinear PCA that works linearly in the kernel space.

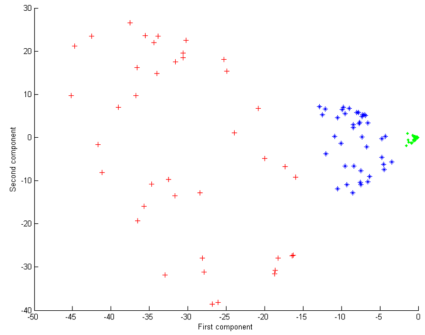
Summary

1. Compute the $N \times N$ kernel matrix $\mathbf{K}$: $K(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j)$
2. Solve the eigenproblem: $\mathbf{K}\boldsymbol{\alpha} = \lambda\boldsymbol{\alpha}$
3. Normalize eigenvectors: $\boldsymbol{\alpha}_k^T \boldsymbol{\alpha}_k = \frac{1}{\lambda_k}$
4. Compute the projection for test point $\mathbf{x}$: $a_k = \sum_j \boldsymbol{\alpha}_{k,j} K(\mathbf{x}_j, \mathbf{x})$

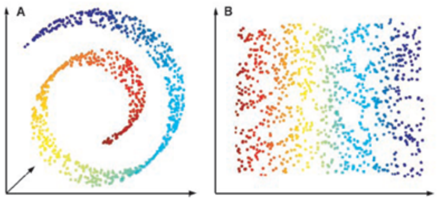
KPCA - An Example

- Left: original data; note the linear inseparability
- Right: Output after kernel PCA. The three groups are now distinguishable using just the 1st component.



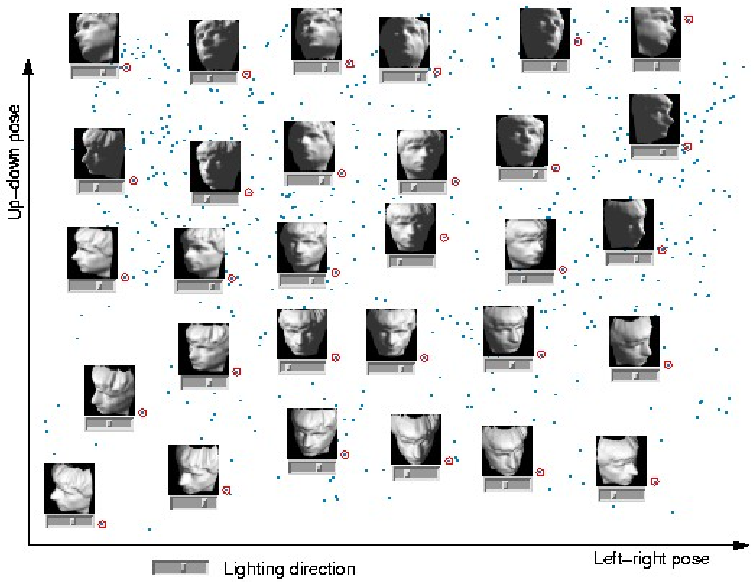


Isomap



- Tenenbaum et al., Science, v290(5500), 2000
- A nonlinear MDS
- Connect each point to its k nearest neighbors to form a graph
- Approximate pairwise geodesic distances using Dijkstra's algorithm on this graph
- Apply Metric MDS to recover a low dimensional isometric embedding

Isomap: An Example



Local Linear Embedding (LLE)

- Roweis & Saul, Science, v290(5500), 2000.
- Compute the k nearest neighbors;
- Solve for the weight matrix W necessary to reconstruct each point using a linear combination of its neighbors:

$$\epsilon = \sum_i \|X_i - \sum_j W_{ij} X_j\|^2,$$

$W_{ij} = 0$ if X_i and X_j are not neighbours, $\sum_j W_{ij} = 1$.

- Find a low dimensional embedding which minimizes reconstruction loss: $\sum_i |Y_i - \sum_j W_{ij} Y_j|^2$
  - Equivalent to find $d+1$ eigenvector of matrix $(I - W)^T (I - W)$ with the *smallest* eigenvalues;
  - Discard the unit vector with zero eigenvalue and fetch d eigenvectors.

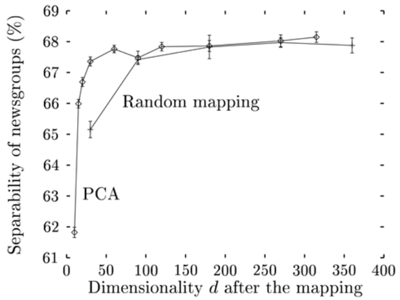
Random Projection

- Use a random linear transform into a space of reduced dimensionality.
- In high-dimensional space there exist a much larger number of almost orthogonal than orthogonal directions.
- So even random vectors may be sufficiently close to orthogonal to provide an approximation of a basis.
- Very attractive with low computing complexity.
- Scales well to high dimensional data.
- Applied e.g. in document clustering with thousands of dimensionality.

Random Projection - how to

- Construct a random matrix $\mathbf{R}_{k\times d}$
- Data projection:
  $\mathbf{X}'_{k\times N} = \mathbf{R}_{k\times d}\mathbf{X}_{d\times N}$, $k \ll N$
- Elements r_{ij} of $\mathbf{R}$ are often Gaussian distributed
- E.g., Achlioptas (2001):

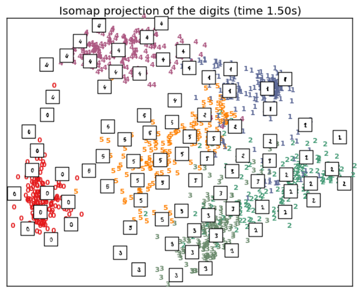
$$r_{ij} = \sqrt{3} \times \begin{cases} +1 & Prob. = 1/6 \\ 0 & Prob. = 2/3 \\ -1 & Prob. = 1/6 \end{cases}$$

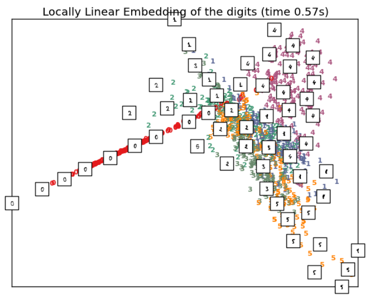


Kaski (1998)

Manifolds of handwritten digits

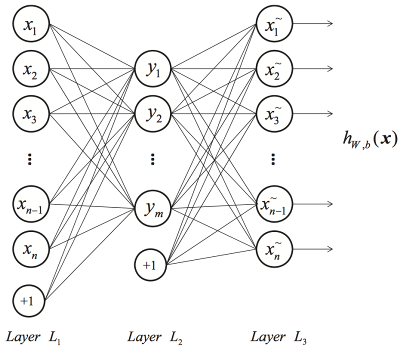
http://scikit-learn.org/dev/auto_examples/manifold/plot_lle_digits.html





Auto-encoder

- Auto-encoder, aka auto-associator, is a 3-layer NN
- Attempts to learn a function $h_{W,b}(x) \approx y$ through error back-propagation.
- Usually with a small hidden-layer, similar to PCA; other constraints e.g. sparsity
- Can be extended to deep structures, e.g. Hinton's restricted Boltzmann machines (RBM)



Wang, Yao, & Zhao (2016)

Independent Components Analysis

- Signals/data we have usually come from a number of sources as a mixture
- How to separate them - blind source separation
- Seek components that are most independent from each other
- The algorithm: independent component analysis (ICA)
- Extremely useful in audio signal processing and medical applications (e.g., EEG)

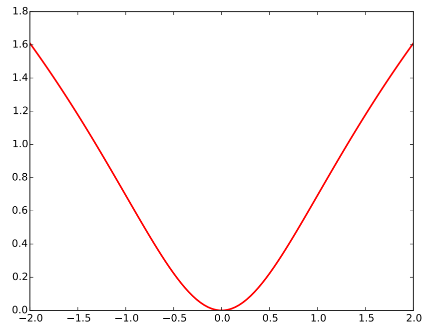
Sparse Coding

- Reconsider the linear approximation problem $\mathbf{x} = \sum_{i=1}^{k} a_i \phi_i$, $\mathbf{x} \in \mathbf{R}^n$.
- Rather than having $k \leq n$, assume $k > n$ (overcomplete); however, most of a_i coefficients are zero, hence with *sparsity*
- Optimization goal is to find

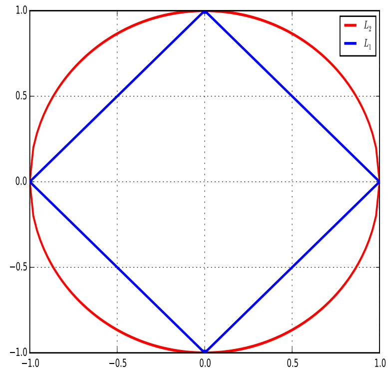
$$\min_{a_i, \phi_i} \sum_j \|\mathbf{x}^{(j)} - \sum_i a_i^{(j)} \phi_i\|^2 + \lambda \sum_i S(a_i^{(j)})$$

- Usual setting is to use L_1 norm: $S(a_i) = |a_i|_1$
- Or use a penalty function: $S(a_i) = \log(1 + a_i^2)$

$S(a_i) = \log(1 + a_i^2)$



L_2 vs. L_1



References

- Alpaydin, Chapter 6.
- Hertz et al., *Introduction to the Theory of Neural Computation*, 1991. Chapter 8.
- Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd Ed., 1999, Chapter 8.
- UFLDL Tutorial: PCA, Sparse Coding, `http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial`

Recap

Review Questions

- SOM already maps high-dimensional data onto low-dimensional grids. Why do we still use MDS on the maps?
- What is the goal PCA tries to achieve?
- Can we use PCA to help build classifiers? Why, and why not?
- Give two dimension reduction methods that deal with linear inseparability of data and explain how they work.