Dimesionality Reduction and Feature Selection

M. VAZIRGIANNIS

November 2015

Outline

Motivation

Feature Ranking and Selection

Dimensionality Reduction

Feature selection

Select the “best” features (subset of the original one)

Filter methods:

rank the features individually according to some criteria (information gain, χ^2, etc.) and take the top-k or eliminate redundant features (correlation)

Wrapper methods:

evaluate each subset using some data mining algorithm; use heuristics for the exploration of the subset space (forward/backward search, etc.)

Embedded methods:

feature selection is part of the data mining algorithm

Filter methods - Information Gain (IG)

For a random variable X (class) its entropy

$$H = -\sum_{i=1}^{c} P(x_i) \times \log(P(x_i)) \quad \text{, c classes}$$

"High Entropy": X is from a uniform distribution – lack on information

"Low Entropy": X is from varied (peaks and valleys) distribution – rich in information content

Let variable A (feature), IG(X, A) represents the reduction in entropy (~ gain in Information) of X achieved by learning the state of A:

IG(X,A)=H(X)–H(X|A)

Filter methods - Chi-squared test ($\chi 2$)

Test of independence between a class X and a feature A

$$\chi^2(A) = \sum_{i=1}^{v}\sum_{j=1}^{c}\frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$, v values, c classes

O_{ij}: observed frequency of class j for feature A (value i)

E_{ij}: the expected frequency

$$E_{ij} = \frac{\text{(\# of samples with value i) x (\# of samples with class j)}}{\text{\# of samples in total}}$$

Finding the k best variables

Find the subset of k variables that predicts best:

This is a generic problem when p is large
(arises with all types of models, not just linear regression)

Models with different complexity..

- *p* models with a single variable
- *p(p-1)/2* models with 2 variables, etc
- …
- 2^p possible models in total

Best k set is not the same as the *best k individual* variables

What does “best” mean here?

Search Problem

How can we search over all 2^p possible models?

exhaustive search is clearly infeasible

Heuristic search is used to search over model space:

- Forward search (greedy)
- Backward search (greedy)
- Branch and bound techniques

Variable selection problem in several data mining algorithms

- Outer loop that searches over variable combinations
- Inner loop that evaluates each combination

Forward model selection

- Start with the variable the lowest p-value (i.e. value with the highest evidence for rejecting the null hypothesis)
- add in each repetition the variable with the *highest* F-test value:

$$F = \frac{\frac{RSS_1 - RSS_2}{\rho_2 - \rho_1}}{\frac{RSS_2}{n - \rho_2}}$$

- *Assume two models* p_2, p_1 with $|p_2| > |p_1|$
- Repeat until F-value < $threshold_f$ (or p-value > $threshold_p$)
- *RSSi* the residual sum of squares - the error induced by the model:

$$F = \sum_1^n (y_i - f(x_i))^2$$

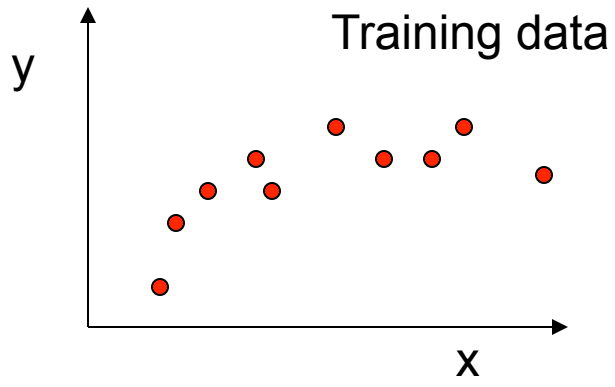
with y_i real value and $f(x_i)$ predicted by models containing p_i .

Backward Elimination

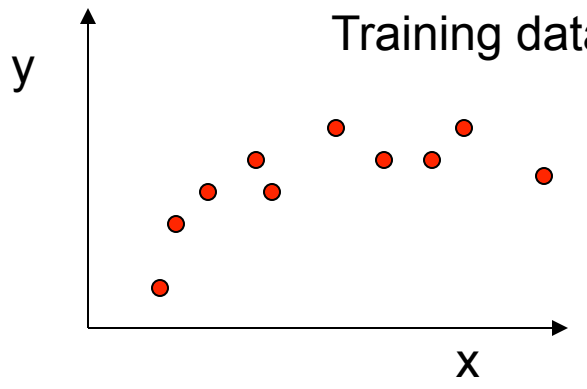
- start with the full model
- drop the predictor that produces the smallest F value (or highest p-value)
- Continue until F-value < threshold_f

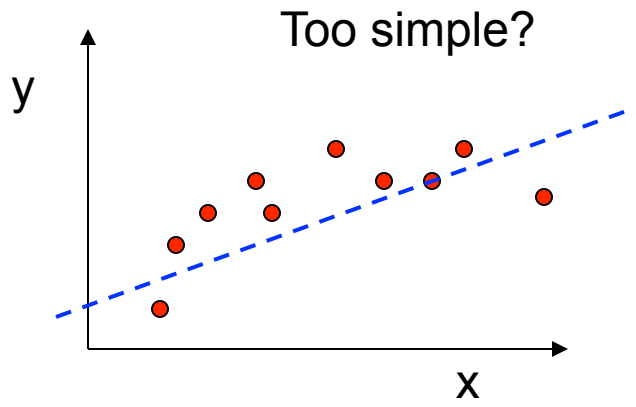
  (or p-value > threshold_p)
- Sometimes constraint N>p

Complexity versus Goodness of Fit

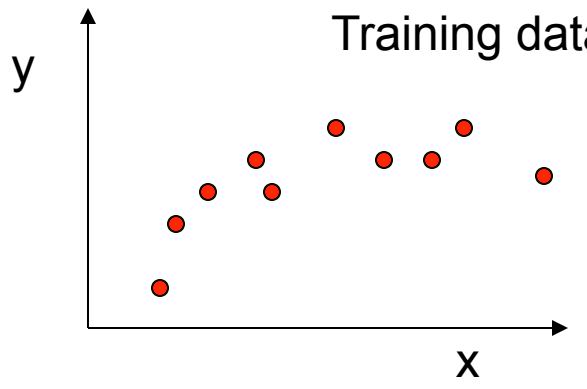


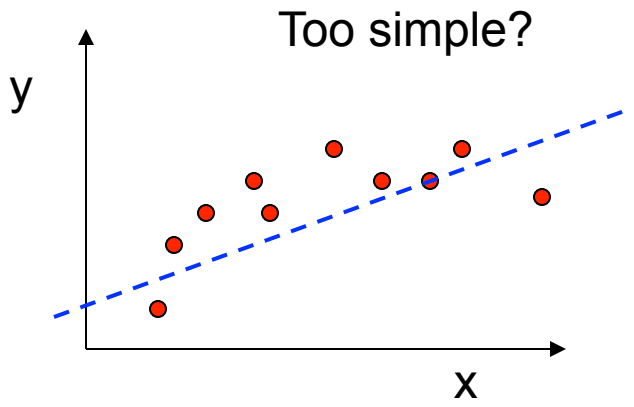
Complexity versus Goodness of Fit

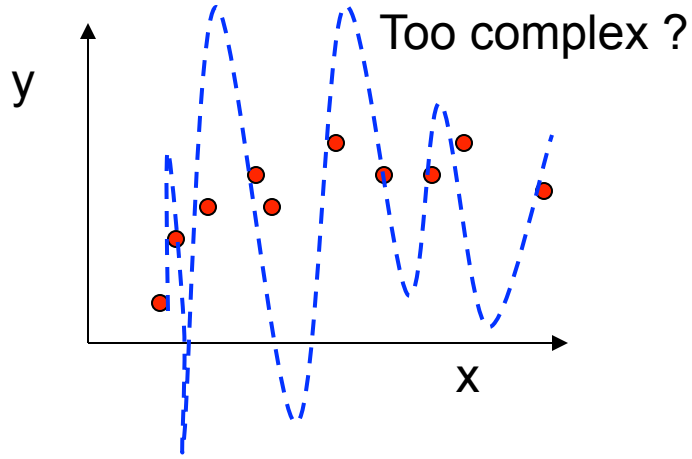




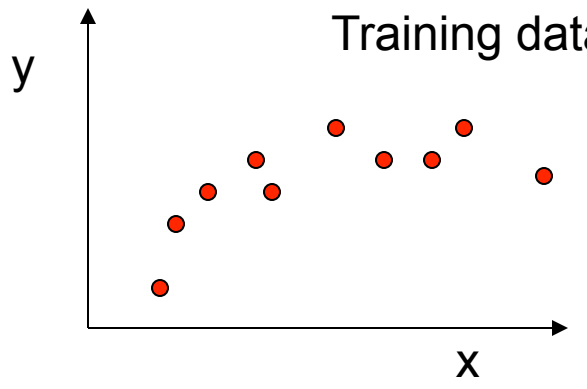
Complexity versus Goodness of Fit

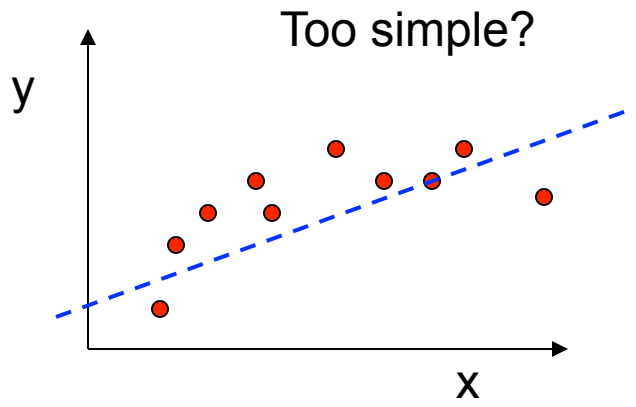


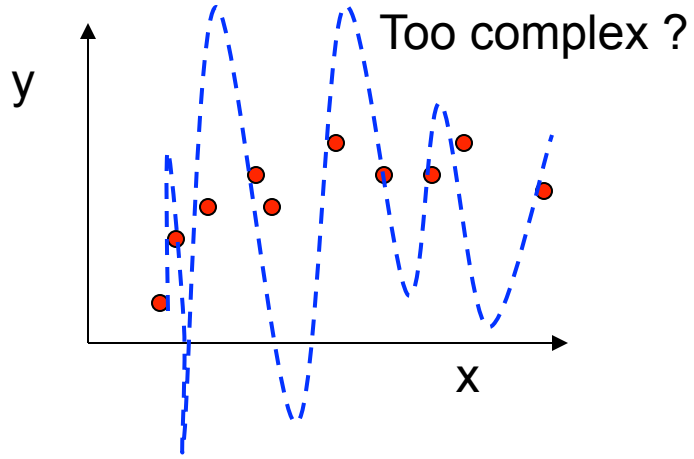


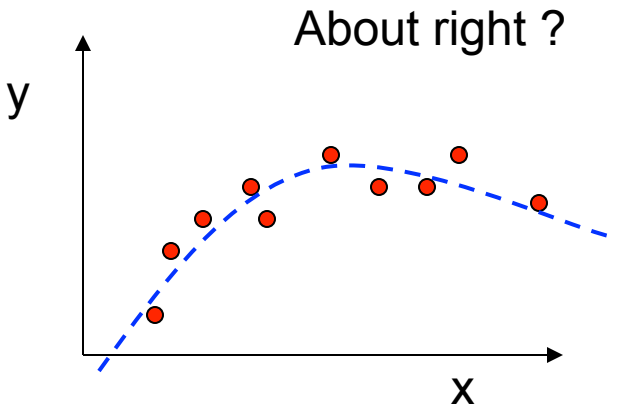


Complexity versus Goodness of Fit

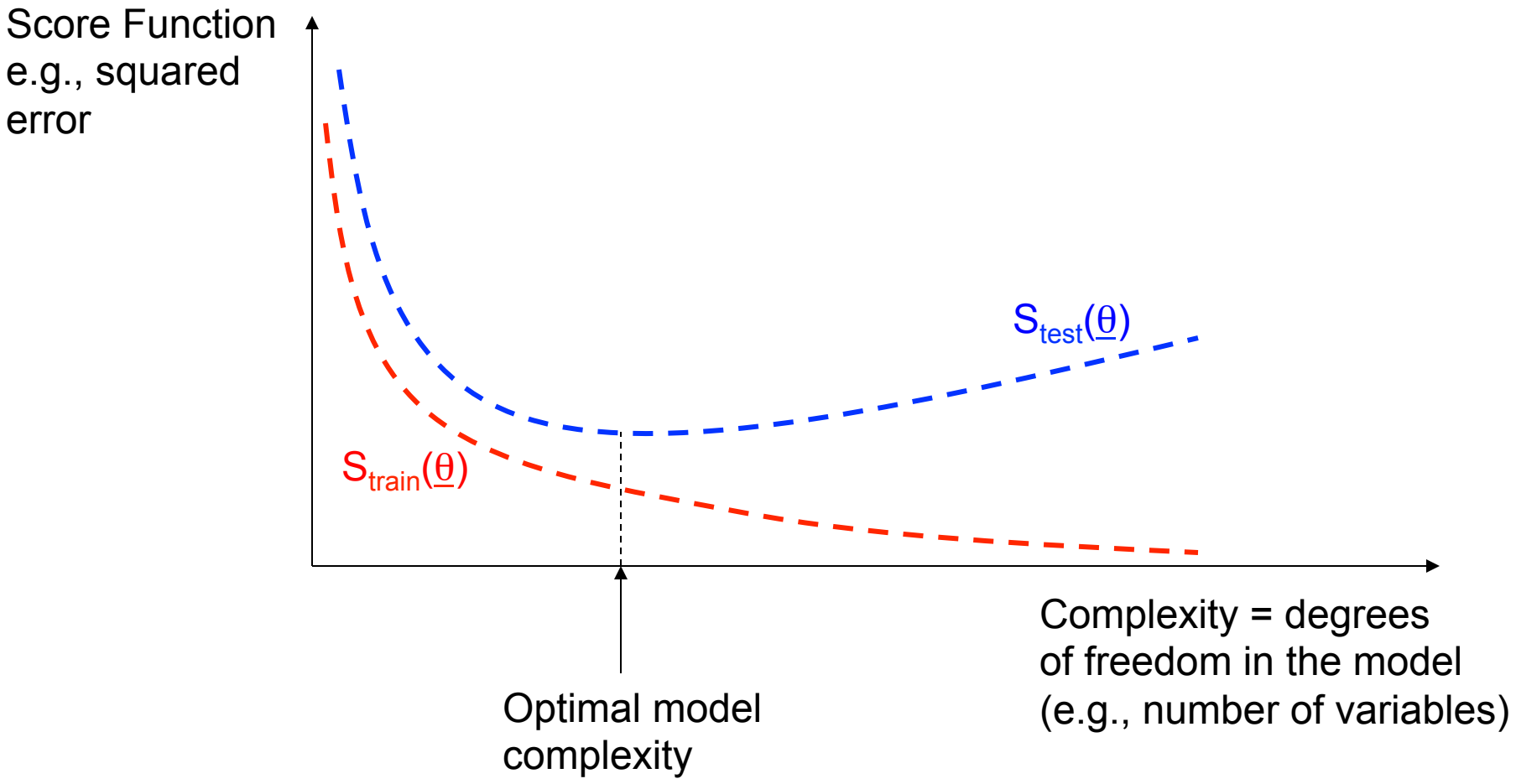








Complexity and Generalization



Useful References

- **Principles of Data Mining, David J. Hand, Heikki Mannila and Padhraic Smyth** MIT Press 2001
- **T. Hastie, R. Tibshirani, and J. Friedman, Elements of Statistical Learning,** Springer Verlag, 2001
- **Dash, Manoranjan, and Huan Liu**. "Feature selection for classification." *Intelligent data analysis* 1.1-4 (1997): 131-156.
- **N. R. Draper and H. Smith, Applied Regression Analysis, 2nd edition, Wiley, 1981** (the "bible" for classical regression methods in statistics
- **An introduction to variable and feature selection, Isabelle Guyon, André Elisseeff**, The Journal of Machine Learning Research archive Volume 3, 3/1/2003, pp. 1157-1182
- **Mohammed J. Zaki, course notes, High Dimesional Notes**
  http://www.cs.rpi.edu/~zaki/www-new/uploads/Dmcourse/Main/chap6.pdf

Dimensionality Reduction

Linear

SVD, MDS, PCA, NMF, LDA

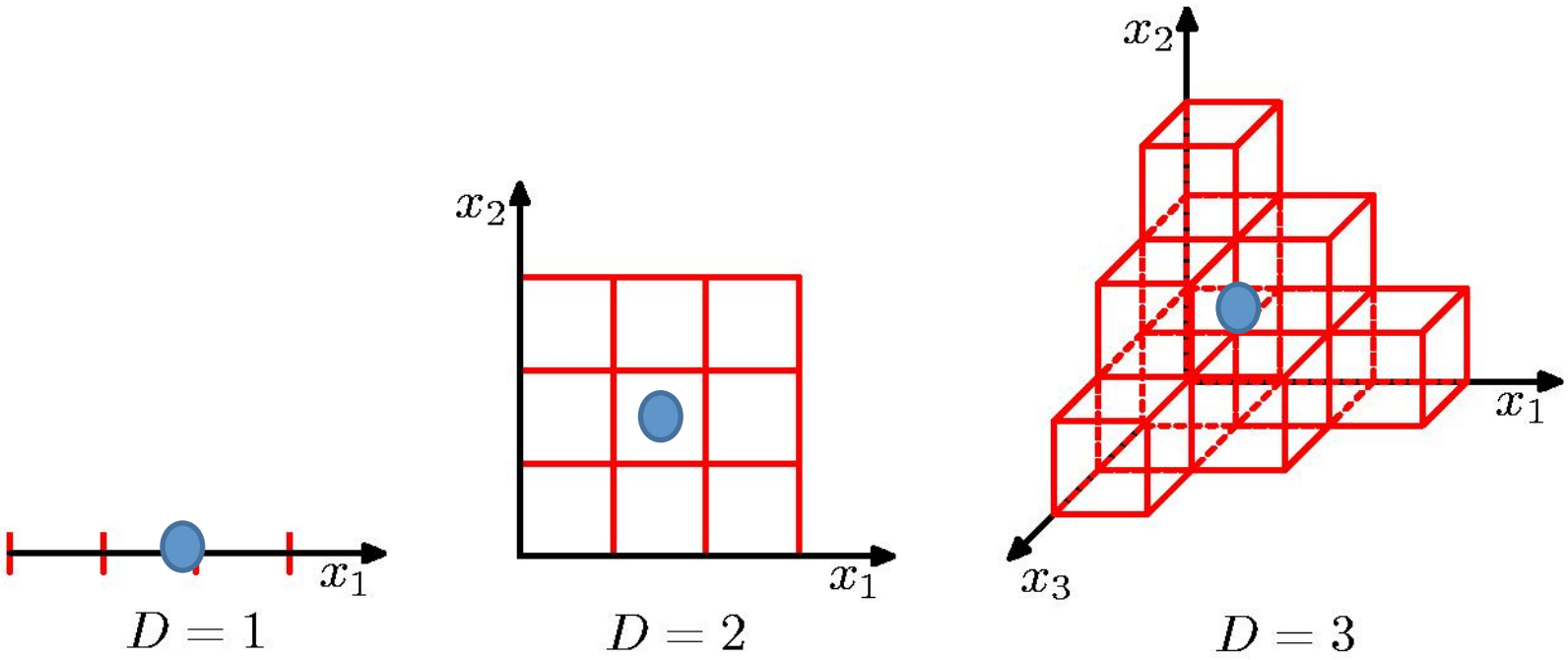
Data features

- Huge volume/ Dimensionality
- Heterogeneity
- Dynamism
  - Motion
  - Availability?
  - Frequent Updates
- Huge query loads
- Examples: Web, P2P systems, Image data

Curse of Dimensionality

- Some coordinates do not contribute to the data representation.
- Subsets of the dimensions may be highly correlated.
- Nearest neighbor is distorted in a high dimensional space
  Low dimension intuitions do not apply to high dimensions
- Empty space phenomenon

Curse of Dimensionality – k-NN



Assuming k-nn

- **2dk neighbors are needed for a d dimensional space**
- **Distance computations are increasingly complex**

Empty space phenomenon

Hyper sphere within a hyper rectangle

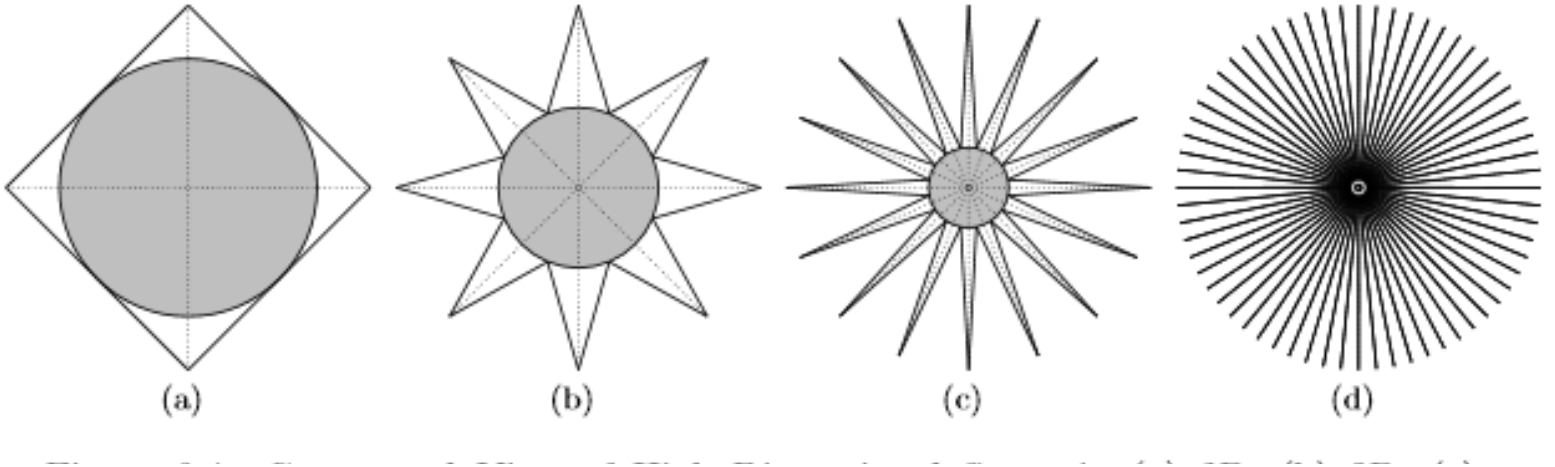
Respective Volumes $\mathbf{V(S)} = {2r^d\pi^{\frac{d}{2}}}/{d\Gamma(d/2)}, \boldsymbol{V(R)} = \boldsymbol{(2r)^d}$

The fraction of the sphere within the rectangle becomes insignificant with d increasing

$$\lim_{d\to\infty}\left({\pi^{\frac{d}{2}}}/{d2^{d-1}\Gamma(d/2)}\right) = 0$$

- normal distribution in high dimensions
- longest/shortest distances converge.
- clustering becomes infeasible

Inscription of hyper sphere in a hypercube



The radius of the inscribed circle accurately reflects the difference between the volume of the hypercube and the inscribed hypersphere in d-dimensions.

http://www.cs.rpi.edu/~zaki/www-new/uploads/Dmcourse/Main/chap6.pdf

Dim. Reduction – Linear Algorithms

Matrix Factorization methods

- Principal Components Analysis (PCA)
- Singular Value Decomposition (SVD)
- Multidimensional Scaling (MDS)
- Non negative Matrix Factorization (NMF)
- Latent Semantic Indexing (LSI)

Low Rank Approximation

Data: $\boldsymbol{X} = \{\boldsymbol{x}_i \in \boldsymbol{R}^{mxn} | \boldsymbol{x}_i$ **columns of** $\boldsymbol{X}\}$

Goal: approximate $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T$,

$\boldsymbol{U} \in \boldsymbol{R}^{mxr}$, $\boldsymbol{V} \in \boldsymbol{R}^{nxr}$, , $r << n$

- each data vector $\boldsymbol{x}_i$: $\boldsymbol{x}_i \sim \mathrm{U}\boldsymbol{v}_i^T$, $\boldsymbol{v}_i$ is the i-th column of $\boldsymbol{V}$.

Geometric interpretation:

- each data vector $\boldsymbol{x}_i \in \boldsymbol{R}^m$, $_i \sim \mathrm{U}\boldsymbol{v}_i^T$, is approximated by its projection to an r-dimesional space spanned by the column vectors of U
- $\boldsymbol{Y} = \boldsymbol{U}\,\boldsymbol{V}^T$ the approximation matrix, max rank r

Frobenius distance

$$||X - Y||^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}(X_{ij} - Y_{ij})^2$$

- Minimizing the Frobenius distance can be considered as *maximum likelihood estimation*

SOME CONTRIBUTIONS TO DIMENSIONALITY REDUCTION, Wei Tong, Ph.D. thesis, 2010, Michigan State University
http://www.ece.uprm.edu/~domingo/teaching/ciic8996/SOME%20CONTRIBUTIONS%20TO%20DIMENSIONALITY%20REDUCTION.pdf

Dim. Reduction–Eigenvectors

A nxn matrix

- eigenvalues λ: $|A-\lambda I|=0$
- Eigenvectors x : $Ax=\lambda x$
- Matrix rank: # linearly independent rows or columns
- A real symmetric table A nxn can be expressed as: $A=U\Lambda U^T$
- U's columns are A's eigenvectors
- Λ' s diagonal contains A's eigenvalues
- $A=U\Lambda U^T=\lambda_1 x_1 x^T_1+\lambda_2 x_2 x^T_2+\ldots+\lambda_n x_n x^T_n$
- $x_1 x^T_1$ represents projection via x_1 (λ_i eigenvalue, x_i eigenvector)

xx^T vs. x^Tx

Singular Value Decomposition (SVD)

Eigen values and eigenvectors decomposition is applied to square matrices. For non square matrices we apply ***Singular Value Decomposition.***

Let **X** a **mxn table**, $X = U\Sigma V^T$

U : orthogonal mxm, its columns are the eigenvectors of XX^T.

U,V define orthogonal basis: $U^T U = VV^T = 1$

Σ: mxn contains A's singular values (square roots of XX^T eigenvalues)

V : nxn, its columns are the eigenvectors of $X^T X$

Singular Value Decomposition (SVD) - I

PROOF:

$X = U\Sigma V^T$, $X^T = V\Sigma^T U^T$ -> $XX^T = U\Sigma(V^T V)\Sigma^T U^T$ -> $XX^T = U\Sigma\Sigma^T U^T$

Similarly: -> $X^T X = U\Sigma(V^T V)\Sigma^T U^T$ therefore $X^T X = V\Sigma^T \Sigma U^T$

Hence: U: eigenvectors of XX^T, V: *eigenvectors of* $X^T X$ and Σ sqrt of MM^T (or $M^T M$) eigenvalues

Let **X** a **mxn table**, $\boldsymbol{X = U\Sigma V^T}$ then an r rank approximation of X is:

$$\boldsymbol{Y = U_{mxr}\,\mathrm{diag}(\sigma_1, \ldots \sigma_r)V_{nxr}{}^T}$$

Singular Value Decomposition (SVD) - II

Matrix approximation

The best rank r approximation Y' of a matrix *X*. (minimizing the Frobenius norm)

$$\|A\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2 = \operatorname{trace}(AA^H) = \sum_{i=1}^{\min\{m,n\}}\sigma_i^2$$

where A^H transpose of *A*, σ_i are the singular values of *A*, and the trace function is used.

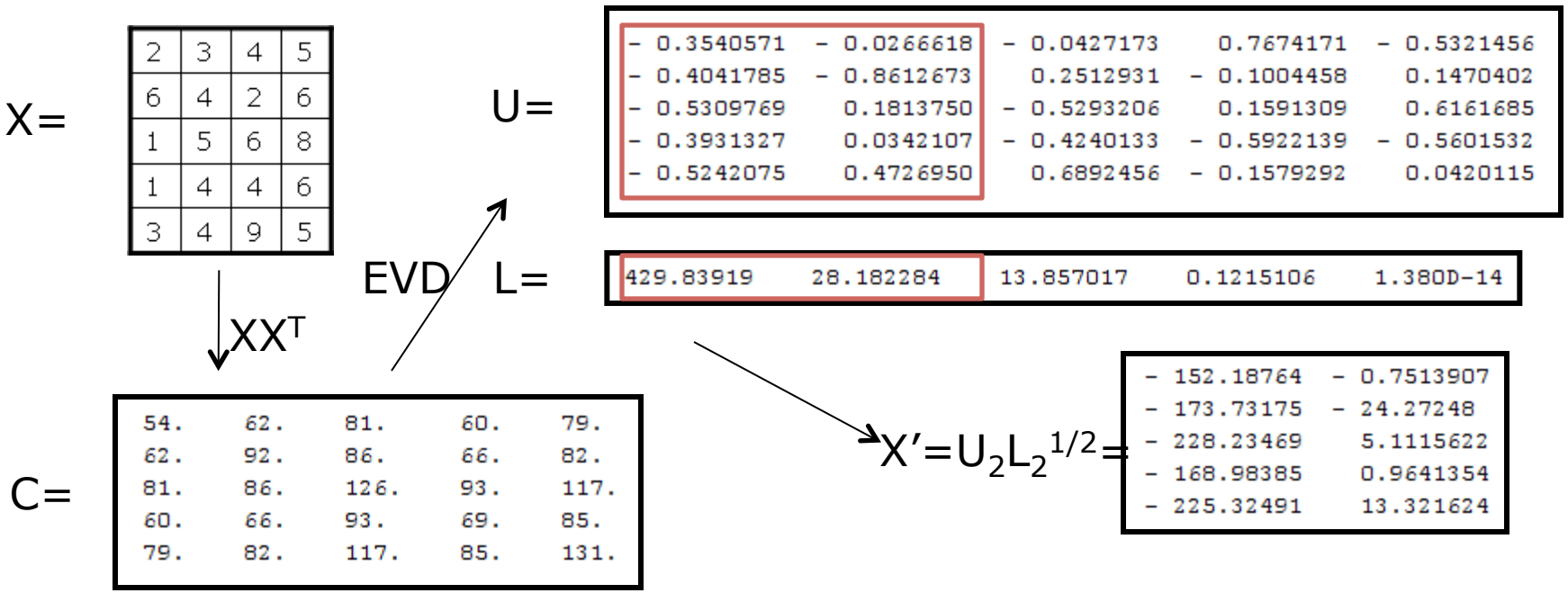
The Frobenius norm is sub-multiplicative and is very useful for numerical linear algebra. This norm is often easier to compute than induced norms.

Multidimensional Scaling (MDS)

- Initially we depict vectors in random places
- Iteratively reposition them in order to minimize Stress.
  - Stress = $\sum(d_{ij}-d_{ij}')^2/\sum d_{ij}^2$
  - Complexity $O(N^3)$ (N:number of vectors)
- Result:
  - A new depiction of the data in a lower dimensional space.
- Implement usually by:
  - Eigen decomposition of the inner product matrix and projection on the k eigenvectors that correspond to the k largest eigenvalues.

Multidimensional Scaling

- **Data is given as rows in X**
  - $C=XX^T$ (inner product of x_i with x_j)
  - Eigen decomposition of $C' = ULU^{-1}$
  - Eventually $X' = U_k L_k^{1/2}$, where k is the projection dimension



Principal Components Analysis

The main concept behind *Principal Components Analysis* is dimensionality reduction, maintaining as much as possible data's variance.

variance: $V(X)=\sigma^2=E[(X-\mu)^2]$

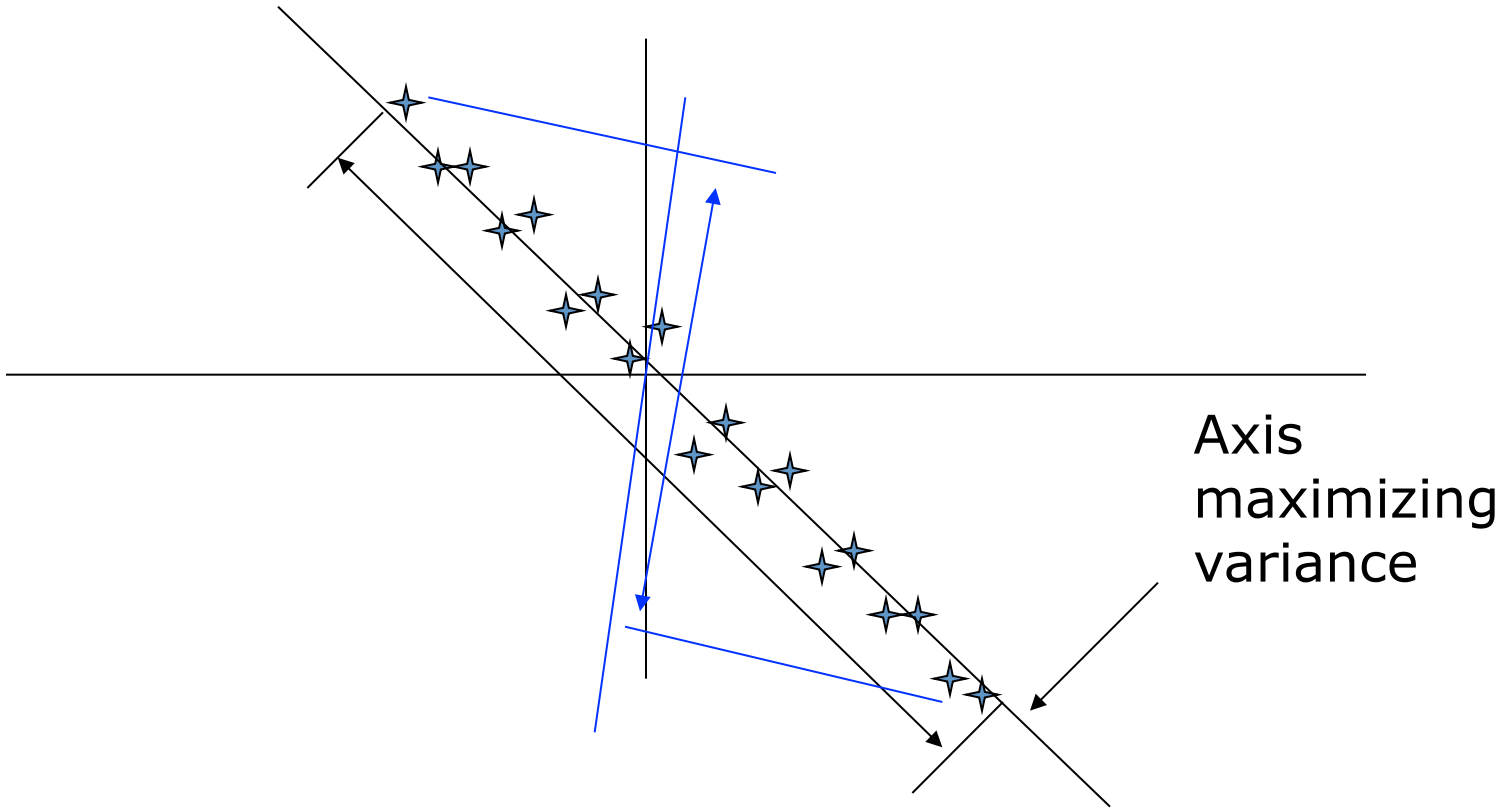
Let *N* objects, with mean value, *m*, it is approximated as:

$$\frac{1}{N}\sum_{i=1}^{N}(x_i - m)^2,$$

Sample of *N* objects with unknown mean value:

$$\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2,$$

Dimensionality reduction based on variance maintenance



Principal Components Analysis

«A linear transformation that chooses a new coordinate system for the data set such that the greatest variance by any projection of the data set comes to lie on the first axis (then called the first principal component), the second greatest variance on the second axis, and so on ...» (wikipedia)

Let n dimensional data, with dimensions: $x_1,\ldots,x_n$

The objective is to project the data to k dimensions via some linear decomposition:

$y_1=a_1*x_1+\ldots+a_n*x_n$

……

$y_k=b_1*x_1+\ldots+b_n*x_n$

should maintain the variance of the original data

Covariance Matrıx

Let Matrix $X = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}$ where Xi vectors

covariance matrix Σ is the matrix whose (*i*, *j*) entry is the covariance

$$\Sigma = \begin{bmatrix} \mathrm{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \mathrm{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \mathrm{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \mathrm{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \mathrm{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$

Also: cov(X) = X’$^{\mathsf{T}}$X’, where X’= X-M

Principal Components Analysis (PCA)

- **The basic idea of PCA is the maximization of the covariance.**
  - Variance: Depicts the maximum deviation of a random variable from the mean.
  - $\sigma^2=\sum_{i=1}^{n} ((x_i - \mu_i)^2/n)$

- **Method:**
  - Assumption: Data is described by p variables and contained as rows in matrix X_{pxn}
  - We subtract mean values from columns. X'=(X-M)
  - Calculate covariance matrix $W = X'^T X'$

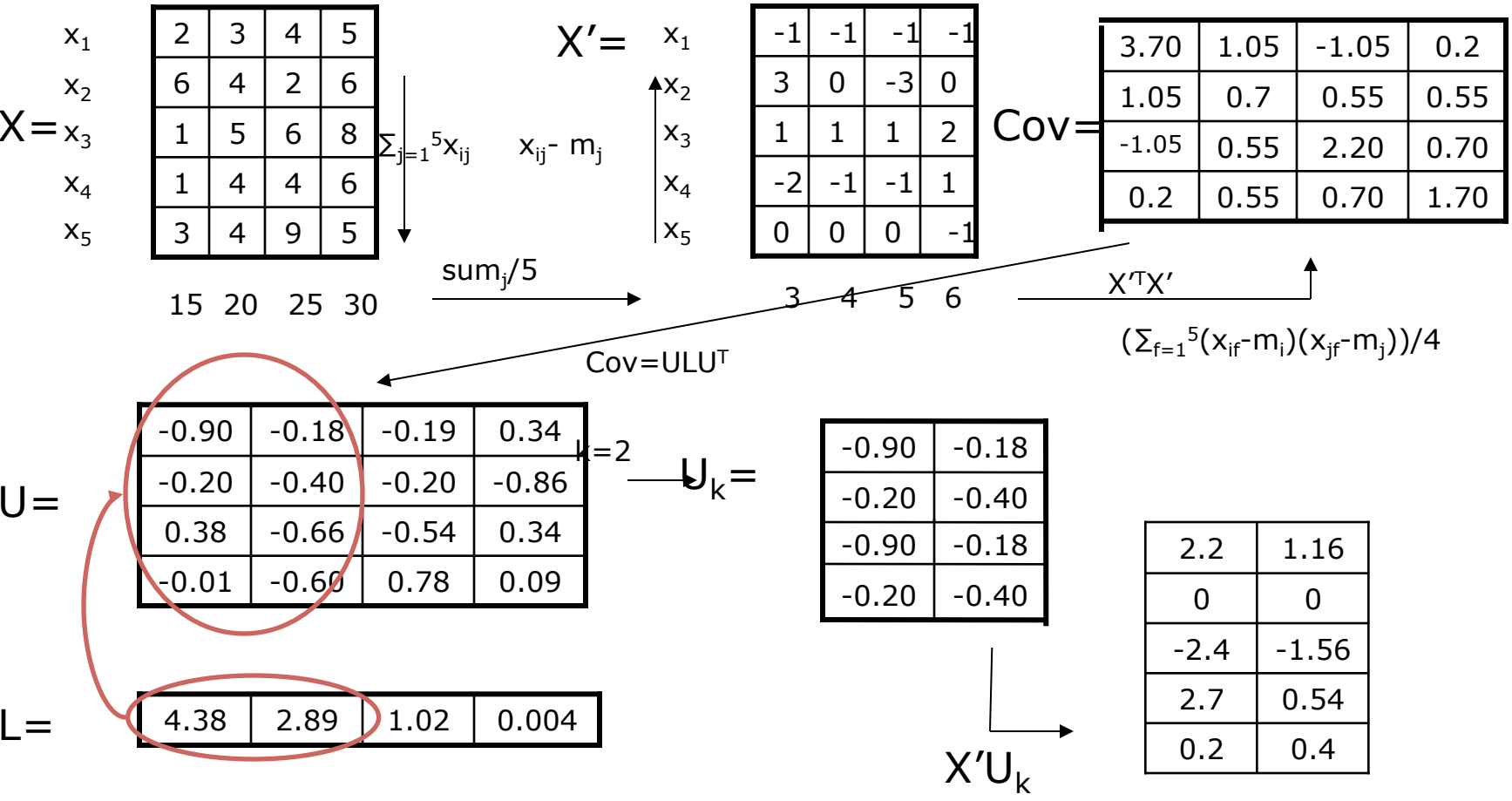
Principal Components Analysis (PCA) – (2)

- **Calculation of covariance matrix W**
  - A matrix nxn, in each cell of W(i,j) we have the covariance of X_i,X_j.
- **Calculate eigenvalues and eigenvectors of W (X,D) = UAU^T**
- **Retain k largest eigenvalues and corresponding eigenvectors**
  - k is an input parameter
  - There is an input parameter and k is calculate by

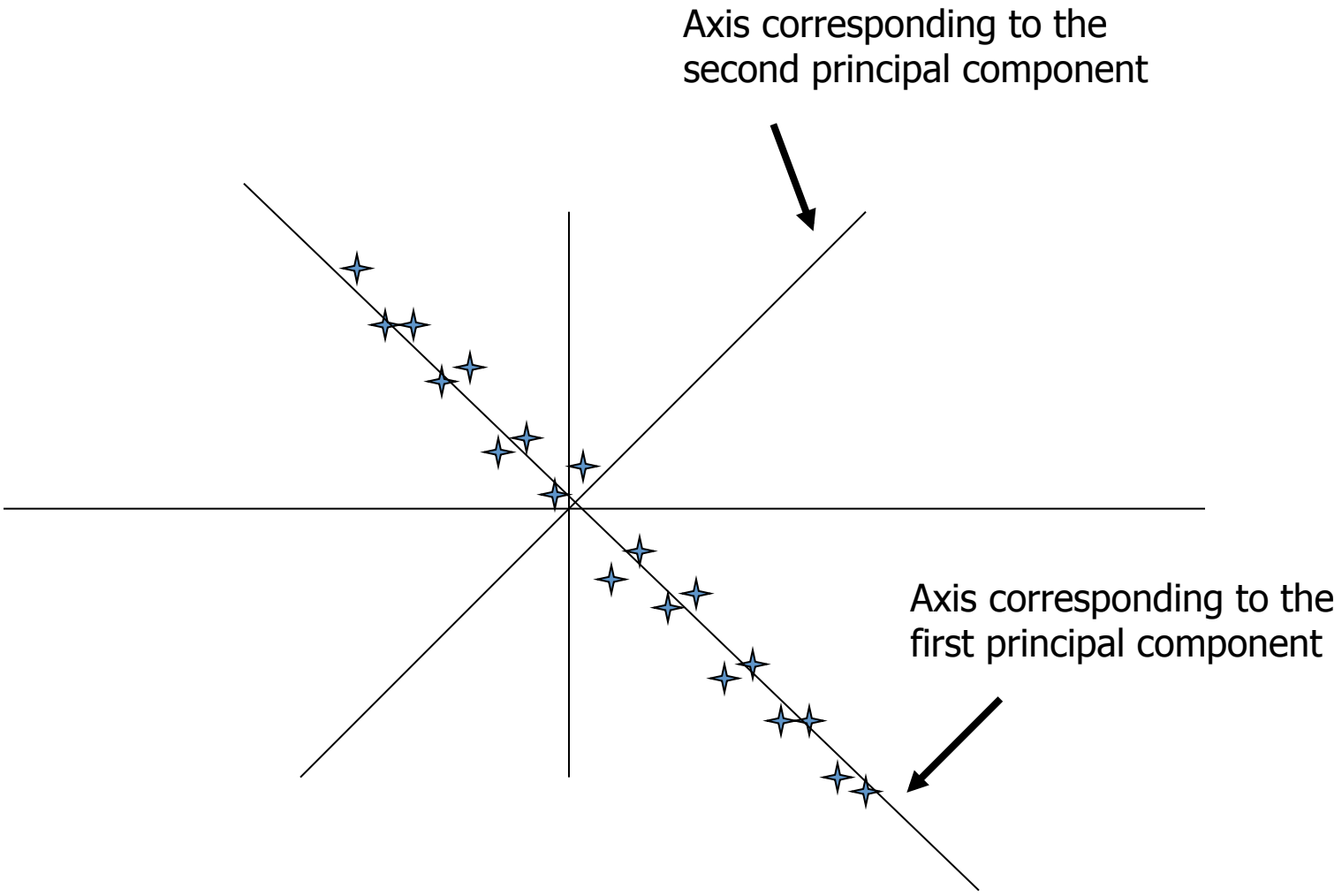
$$\sum^{p}_{j=k+1}\lambda_j / \sum^{p}_{j=1}\lambda_j > 85\%$$

- **Projection : $A'X_k$**

Principal Components Analysis



PCA, example


Axis corresponding to the
second principal component
Axis corresponding to the
first principal component

PCA Synopsis & Applications

- Preprocessing step preceding the application of data mining algorithms (such as clustering).
- Data Visualization & Noise reduction.
- Nominal complexity O(np^2+p^3)

  n: number of data points

  p: number of initial space dimensions

- The new space maintains sufficiently the data variance.

Non Negative Matrix factorization (NMF)

- Applying SVD results in factorized matrices with positive and negative elements may contradict the physical meaning of the result.

Example:

- X gray-scale image intensities, Y its SVD approximation

- difficult to interpret the reconstructed matrix Y for a gray-scale image with negative elements.

- *Nonnegative matrix factorization (NMF)*

find the reduced rank *nonnegative factors* to approximate a given nonnegative data matrix.

Non Negative Matrix factorization (NMF)

Assume X mxn data matrix $(X_{ij} \geq 0), r \ll \min(m,n)$

Then NMF finds non negative matrices

$$U \in R^{mxr}, V \in R^{nxr} : X \approx UV^T$$

To find U, V is to minimize Euclidian Distance $X - UV^T$:

$$\min_{U,V} \quad f(U,V) = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(X_{ij}\log\frac{X_{ij}}{(UV^\top)_{ij}} - X_{ij} + (UV^\top)_{ij}\right)$$
$$\text{s. t.} \quad U_{ia} \geq 0, V_{jb} \geq 0, \forall i, a, b, j.$$

NMF Algorithms

Multiplicative: updating solutions U and V

$$V_{bj}^{\top} \leftarrow V_{bj}^{\top} \frac{\left(U^{\top} X\right)_{bj}}{\left(U^{\top} U V^{\top}\right)_{bj}} \qquad U_{ia} \leftarrow U_{ia} \frac{(XV)_{ia}}{(UV^{\top} V)_{ia}}$$

Gradient descent algorithms

$$V_{bj}^{\top} \leftarrow V_{bj}^{\top} - \epsilon_V \frac{\partial f}{\partial V_{bj}^{\top}} \qquad U_{ia} \leftarrow U_{ia} - \epsilon_U \frac{\partial f}{\partial U_{ia}}$$

ε_v and ε_u are the step sizes.

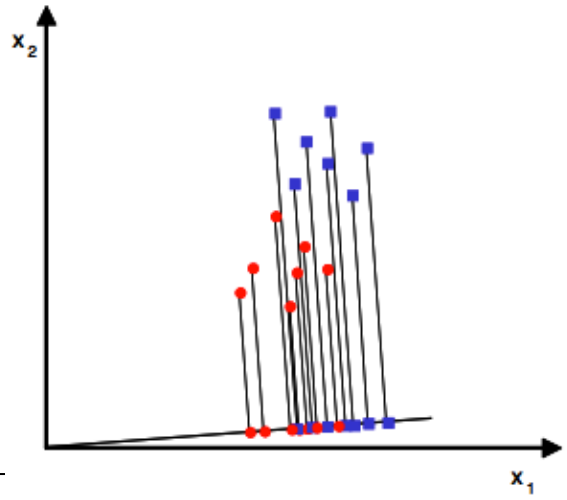
Linear discriminants analysis

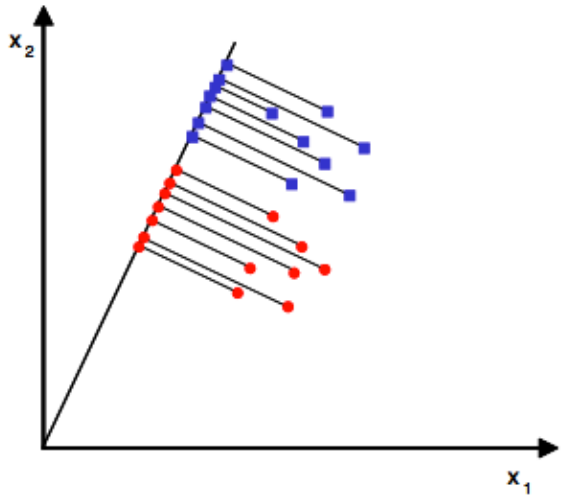
Linear discriminant analysis, two classes

Linear discriminant analysis, C classes

LDA – two classes

- LDA aims to project data in a lower dimensional space that preserves as much of the class discriminatory information as possible
- Assume $X = (x_1, x_2, ...x_n)$ d-dimensional that belong to either of the classes C1 and C2.
- We search for $y = w^T X$ that best separates the data of C1 and C2.

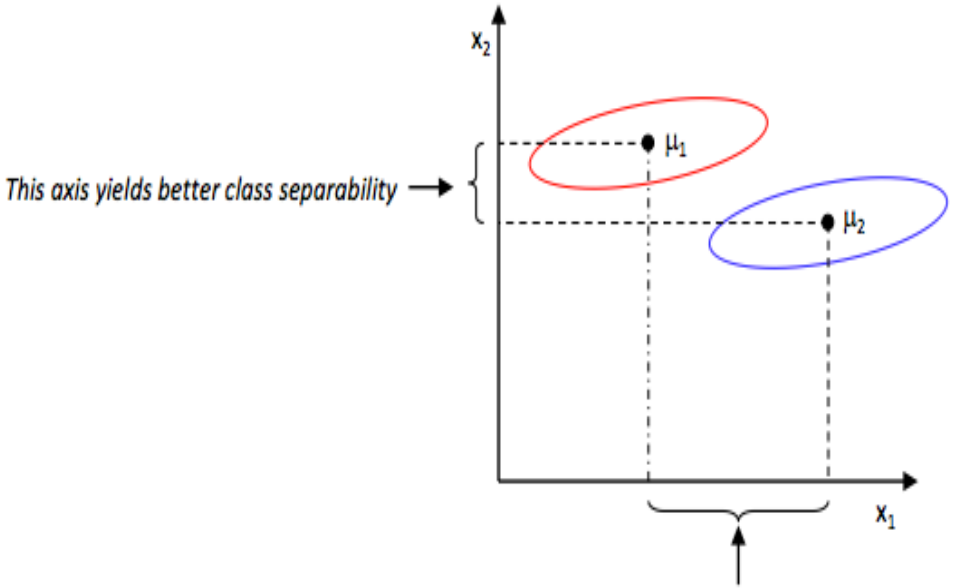


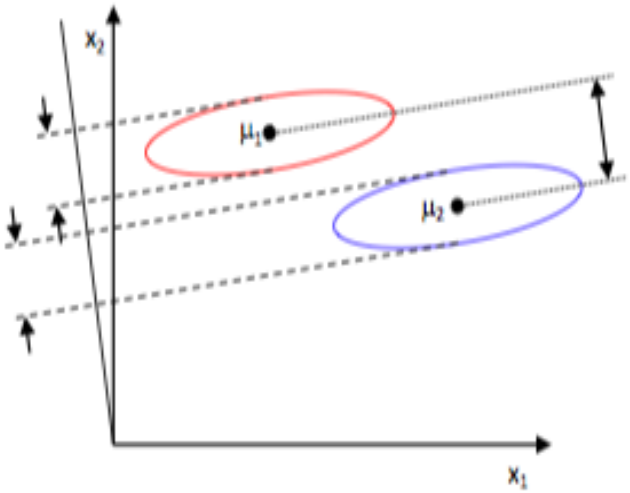


LDA – two classes

The projection should

- maximize the distance of the class centers
- minimize the in class variance





LDA two classes – Fischer's criterion

Distance between projected classes centers $\hat{\mu}_1 \hat{\mu}_2$:

$$|\hat{\mu}_1 - \hat{\mu}_2| = \frac{1}{|C_1|}\sum_{C_1} y_i^{C_1} - \frac{1}{|C_2|}\sum_{C_2} y_i^{C_2} = \frac{1}{|C_1|}\sum_{C_1} w^T x_i^{C_1} - \frac{1}{|C_2|}\sum_{C_2} w^T x_i^{C_2}$$

Within class variance $\hat{S}_i = \sum_{y \in C_i} (y - \hat{\mu}_i)^2$

Total *within class scatter* for the projected data

$$\hat{S}_1 + \hat{S}_2$$

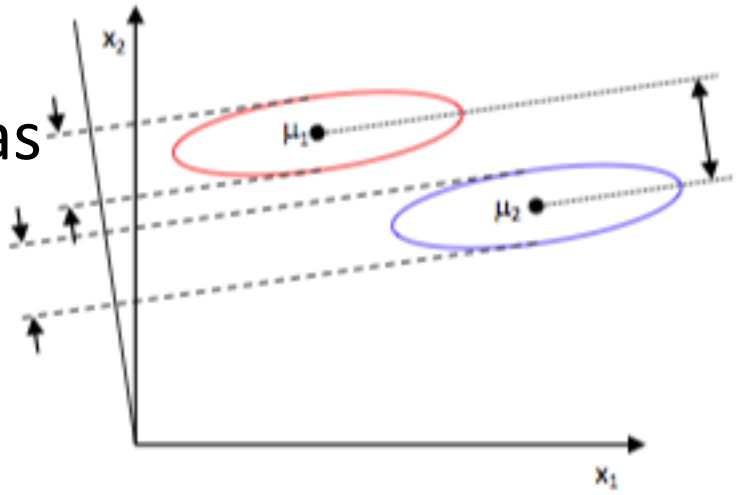
Fischer's linear discriminant

linear function of $y = w^T X$

maximizing: $J(w) = \frac{|\hat{\mu}_1 - \hat{\mu}_2|^2}{\hat{S}_1 + \hat{S}_2}$

Therefore, searching for a projection where same class points are

- projected very close to each
- projected means are as farther as possible



Optimization – within class scatter

Within class scatter matrix in feature space x

$$S_w = S_1 + S_2$$

Where $S_i = \sum_{x \in C_i} (x - \mu_i)(x - \mu_i)^T$

Within class scatter matrix in projected space y

$$\hat{S}_1^{\,2} + \hat{S}_2^{\,2} = w^T S_w w$$

Where $\hat{S}_i = w^T S_i w$

between class scatter in projected space y

$$|\hat{\mu_1} - \hat{\mu_2}|^2 = w^T (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T w = w^T S_B w$$

Fishers Criterion optimization

$$J(W) = \frac{w^T S_B w}{w^T S_W w}$$

To find maximum

$$\frac{d}{dw} J(W) = 0 \implies S_w^{-1} S_B w - Jw = 0$$

Solving the generalized eigenvalue problem:

$$S_w^{-1} S_B w = Jw$$

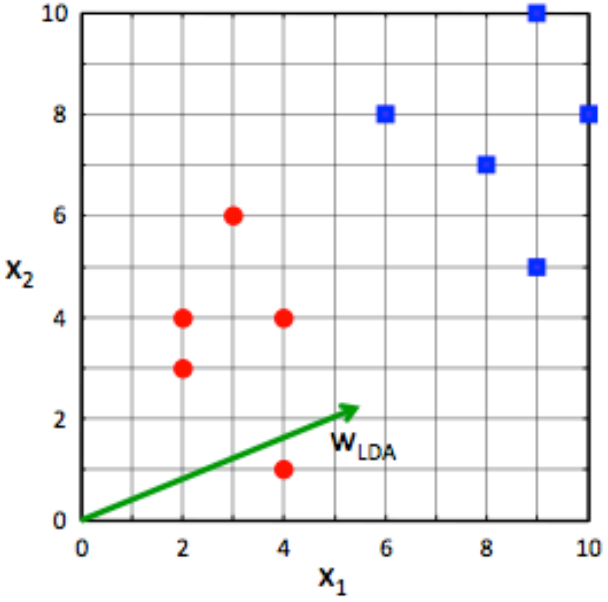
We find:

$$w^* = argmax[\frac{w^T S_B w}{w^T S_W w}] = S_w^{-1}(\mu_1 - \mu_2)$$

An example[1]

- **Compute the LDA projection for the following 2D dataset**

$$X1 = \{(4,1), (2,4), (2,3), (3,6), (4,4)\}$$
$$X2 = \{(9,10), (6,8), (9,5), (8,7), (10,8)\}$$



SOLUTION (by hand)

- The class statistics are

$$S_1 = \begin{bmatrix} .8 & -.4 \\ & 2.64 \end{bmatrix} \quad S_2 = \begin{bmatrix} 1.84 & -.04 \\ & 2.64 \end{bmatrix}$$

$$\mu_1 = [3.0\ 3.6]^T; \quad \mu_2 = [8.4\ 7.6]^T$$

- The within- and between-class scatter are

$$S_B = \begin{bmatrix} 29.16 & 21.6 \\ & 16.0 \end{bmatrix} \quad S_W = \begin{bmatrix} 2.64 & -.44 \\ & 5.28 \end{bmatrix}$$

- The LDA projection is then obtained as the solution of the generalized eigenvalue problem

$$S_W^{-1} S_B v = \lambda v \Rightarrow \left| S_W^{-1} S_B - \lambda I \right| = 0 \Rightarrow \begin{vmatrix} 11.89 - \lambda & 8.81 \\ 5.08 & 3.76 - \lambda \end{vmatrix} = 0 \Rightarrow \lambda = 15.65$$

$$\begin{bmatrix} 11.89 & 8.81 \\ 5.08 & 3.76 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 15.65 \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \Rightarrow \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} .91 \\ .39 \end{bmatrix}$$

- Or directly by

$$w^* = S_W^{-1}(\mu_1 - \mu_2) = [-.91\ -.39]^T$$

LDA C classes [1]

Fisher's LDA generalizes gracefully for C-class problems

- Instead of one projection y, we will now seek $(C-1)$ projections $[y_1, y_2, \dots y_{C-1}]$ by means of $(C-1)$ projection vectors w_i arranged by columns into a projection matrix $W = [w_1|w_2|\dots|w_{C-1}]$:

$$y_i = w_i^T x \Rightarrow y = W^T x$$

Derivation

- The within-class scatter generalizes as

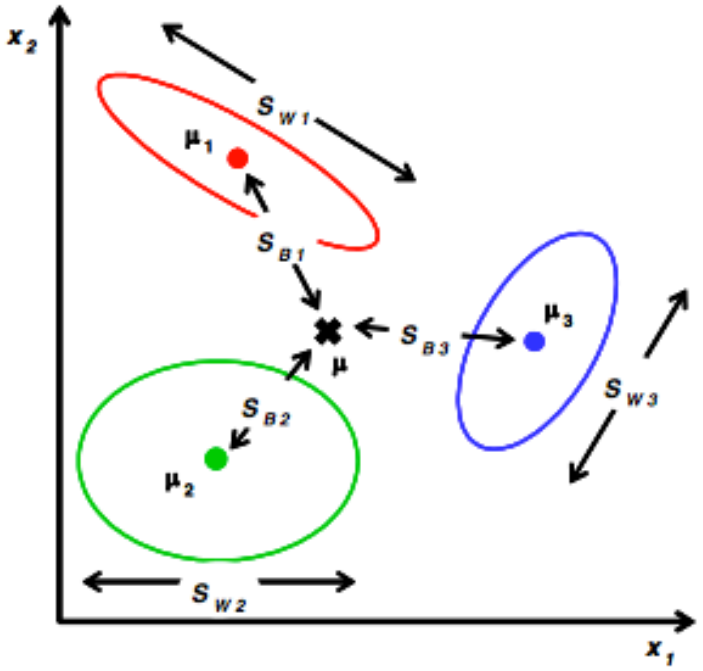
$$S_W = \sum_{i=1}^{C} S_i$$

  - where $S_i = \sum_{x \in \omega_i} (x - \mu_i)(x - \mu_i)^T$ and $\mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$

- And the between-class scatter becomes

$$S_B = \sum_{i=1}^{C} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

  - where $\mu = \frac{1}{N} \sum_{\forall x} x = \frac{1}{N} \sum_{i=1}^{C} N_i \mu_i$



- Matrix $S_T = S_B + S_W$ is called the <u>total scatter</u>

LDA C classes

- mean vector and scatter matrices for the projected samples

$$\tilde{\mu}_i = \frac{1}{N_i}\sum_{y\in\omega_i} y \qquad \tilde{S}_W = \sum_{i=1}^{C}\sum_{y\in\omega_i}(y-\tilde{\mu}_i)(y-\tilde{\mu}_i)^T$$

$$\tilde{\mu} = \frac{1}{N}\sum_{\forall y} y \qquad \tilde{S}_B = \sum_{i=1}^{C} N_i(\tilde{\mu}_i-\tilde{\mu})(\tilde{\mu}_i-\tilde{\mu})^T$$

- Thus generalizing the two class problem: $\tilde{S}_W = W^T S_W W$, $\tilde{S}_B = W^T S_B W$

- Searching for a projection maximizing the ratio of *between-class/within-class scatter*.

- projection is no longer a scalar ($C-1$ dimensions), use the determinant of the scatter matrices to obtain a scalar objective function

$$J(W) = \frac{|\tilde{S}_B|}{|\tilde{S}_W|} = \frac{|W^T S_B W|}{|W^T S_W W|}$$

LDA C classes

- seek the projection matrix $W*$ maximizing this ratio
- optimal projection matrix $W*$
  - columns are eigenvectors resp. largest eigenvalues of generalized eigenvalue problem:

$$W^* = [w_1^* | w_2^* | \dots w_{C-1}^*] = \arg\max \frac{|W^T S_B W|}{|W^T S_W W|} \Rightarrow (S_B - \lambda_i S_W) w_i^* = 0$$

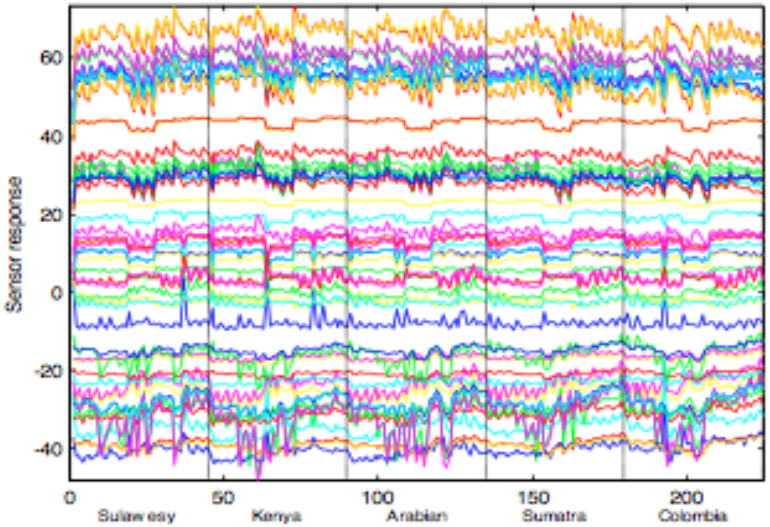
LDA vs. PCA

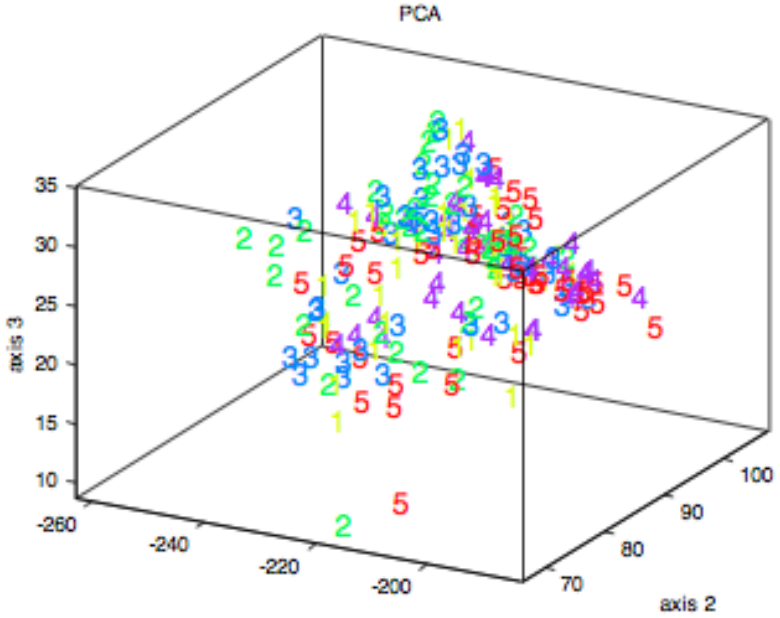
This example illustrates the performance of PCA and LDA on an odor recognition problem

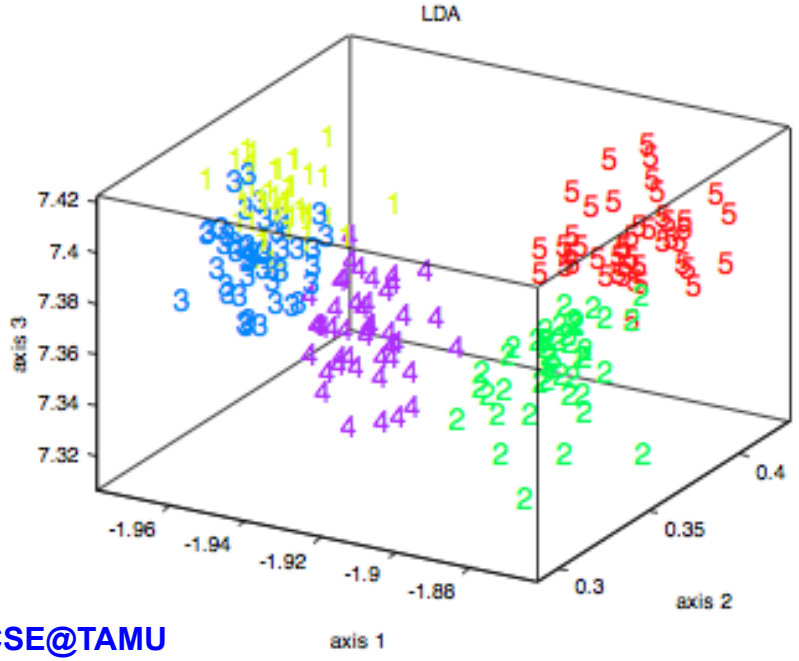
- Five types of coffee beans were presented to an array of gas sensors
- For each coffee type, 45 "sniffs" were performed and the response of the gas sensor array was processed in order to obtain a 60-dimensional feature vector

Results

- From the 3D scatter plots it is clear that LDA outperforms PCA in terms of class discrimination
- This is one example where the discriminatory information is not aligned with the direction of maximum variance







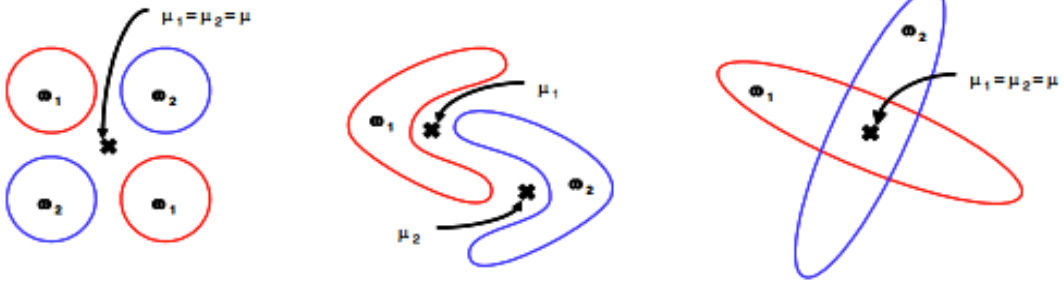
Limitations of LDA

LDA produces at most $C - 1$ feature projections

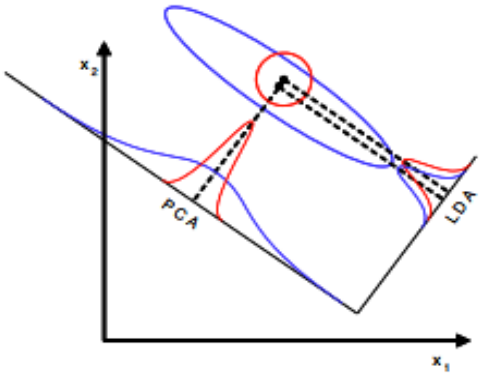
- If the classification error estimates establish that more features are needed, some other method must be employed to provide those additional features

LDA is a parametric method (it assumes unimodal Gaussian likelihoods)

- If the distributions are significantly non-Gaussian, the LDA projections may not preserve complex structure in the data needed for classification



LDA will also fail if discriminatory information is not in the mean but in the variance of the data



Regression evaluation

variance explained by our model

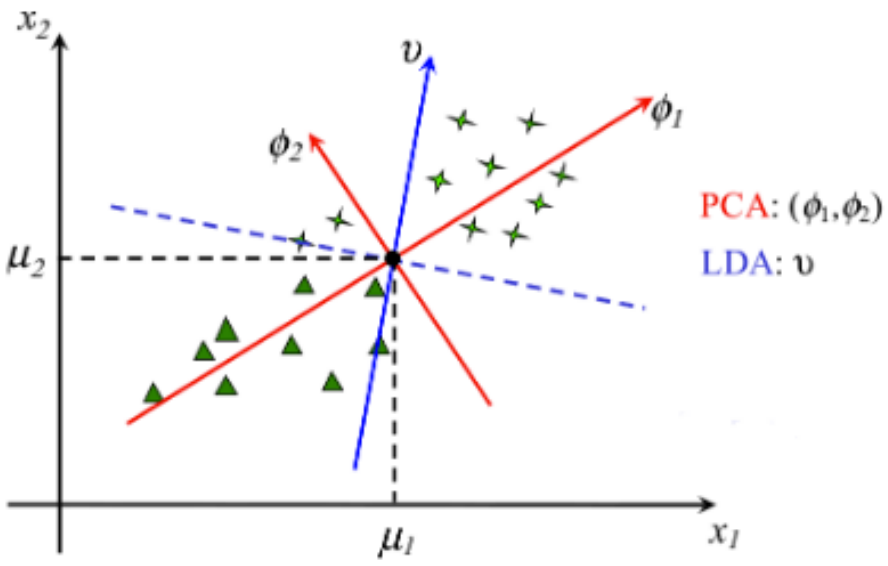
$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y}_i)^2}$$

: mean squared error (residual) divided by total error: unexplained variance proportion.

LDA vs. PCA

- PCA considers the data as a whole
- Axes optimal for representing the data indicating the maximum variation actually lies.
- LDA axis is optimal for distinguishing between the different classes.

- compare ϕ_1 and u axes…



http://www.doc.ic.ac.uk/~dfg/ProbabilisticInference/

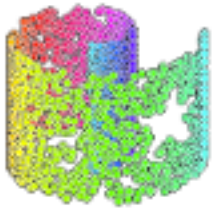
Nonlinear Dimensionality reduction

- ISOMAP
- LLE

ISOMAP

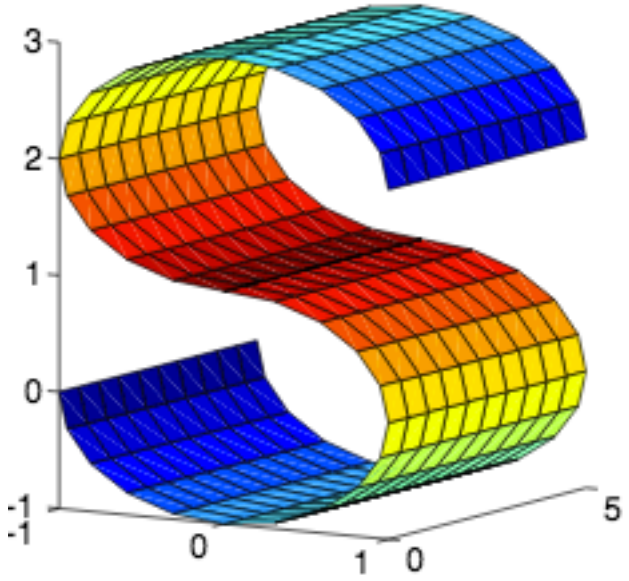
Swiss Role:

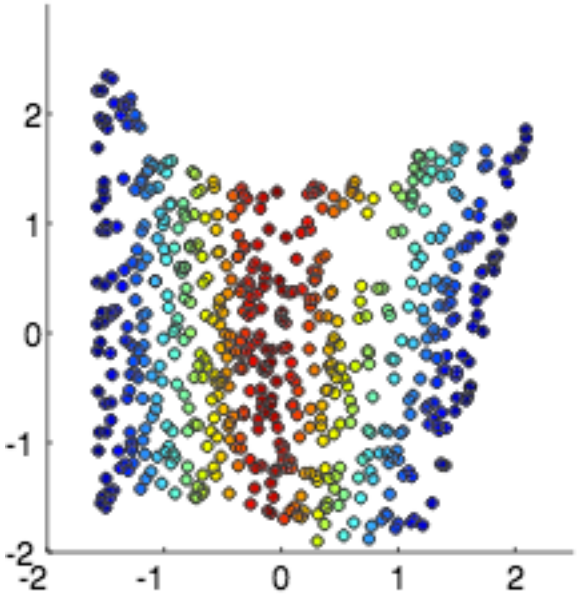
- geometric distance is not a good approach
- Assume Geodesic Distance
  - Graph based distance



- Compute X: pair-wise geodesic distances between all of the points (Floyd–Warshall algorithm)
- Apply Multidimensional Scaling:
  - $XX^T = U\Lambda U^T$
  - $X_k = U_k \Lambda^{1/2}$

Locally-Linear Embedding (LLE)





LLE objective:
maintain the Topological proximity of the 3D space in the 2D one

Locally-Linear Embedding (LLE)

1. Compute the Neighbors of each data point:
   k-NN or all within distance th

2. Reconstruction error: $\mathcal{E}(W) = \sum_i \left| \vec{X}_i - \sum_j W_{ij} \vec{X}_j \right|^2$

   invariance to rotation, scaling

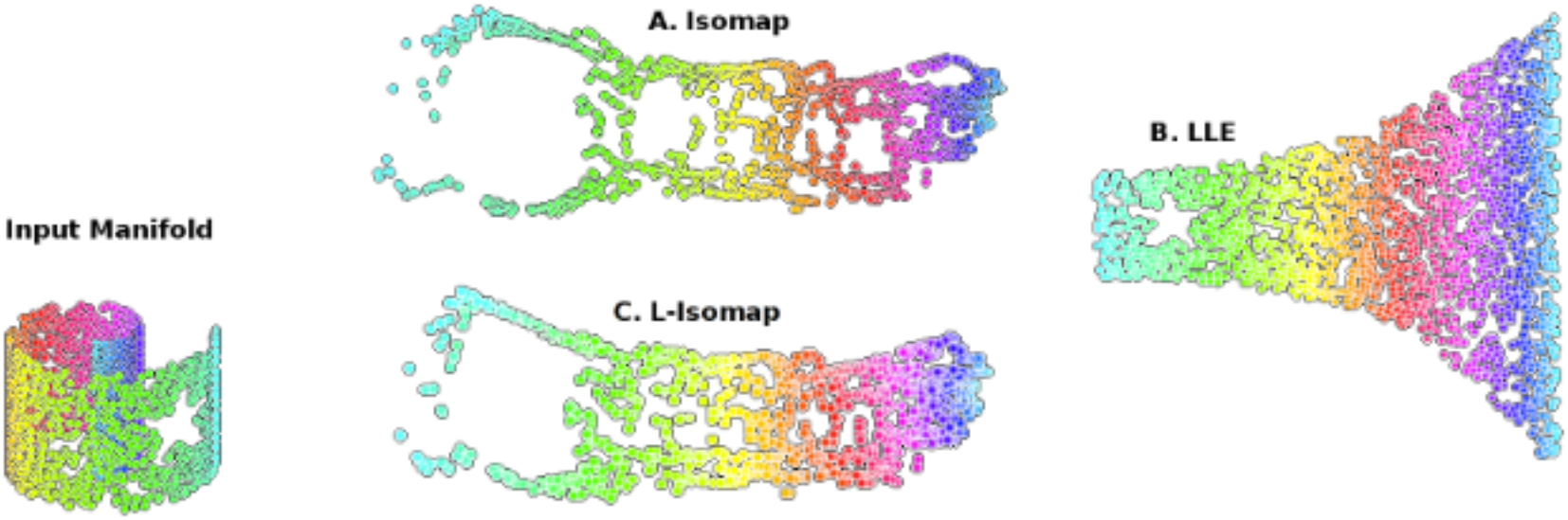
3. Compute the Weights that Best Reconstruct Each Data point from its neighbors, minimizing the cost above

4. Compute the low-dim vector Y best reconstructed by the weights minimizing

$$\Phi(Y) = \sum_i \left| \vec{Y}_i - \sum_j W_{ij} \vec{Y}_j \right|^2$$

- Solve s sparse NxN eigenvector problem
- Bottom nonzero Eigenvectors.

Nonlinear dimensionality reduction by locally linear embedding, Sam T. Roweis, Lawrence K. Saul, Science, 2000


A. Isomap
B. LLE
Input Manifold
C. L-Isomap