

1. Introduction and challenges of high dimensionality
2. Feature Selection
- 3. Feature Reduction and Metric Learning**
4. Clustering in High-Dimensional Data

Idea: Instead of removing features, try to find a *low dimensional feature space* generating the original space as accurate as possible:

- Redundant features are summarized
 - Irrelevant features are weighted by small values or are “erased”
- => actually, the new feature space should contain no “irrelevant” features anymore

Some sample methods (among lots of others):

- Reference point embedding
- Principal component analysis (PCA)
- Singular value decomposition (SVD)
- Fischer-Faces (FF) and Relevant Component Analysis(RCA)
- Large Margin Nearest Neighbor (LMNN)

Idea: Describe the position of each object by their *distances* to a set of *reference points*.

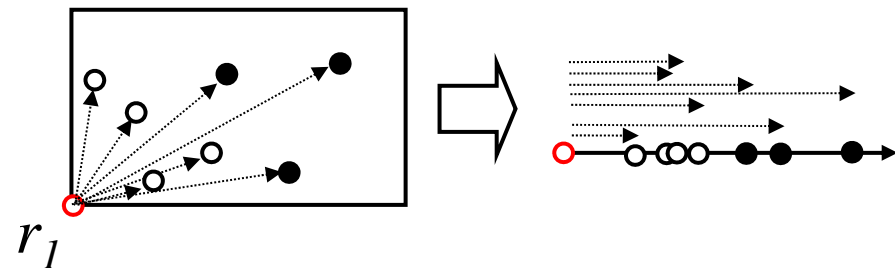
Given: Vector space $F = D_1 \times \dots \times D_n$ where $D = \{D_1, \dots, D_n\}$.

Target: A k -dimensional space R which yields optimal solutions for a given data mining task.

Method: For each reference point $R = \{r_1, \dots, r_k\}$ and a distance measure $d(\bullet, \bullet)$:

Transform vector $x \in F$:

$$r_R(x) = \begin{pmatrix} d(r_1, x) \\ \vdots \\ d(r_k, x) \end{pmatrix}$$



- Distance measure is usually determined by the application.
- Selection of reference points:
 - use centroids of the classes or cluster-centroids
 - using points on the margin of the data space
 - use random sample

Advantages :

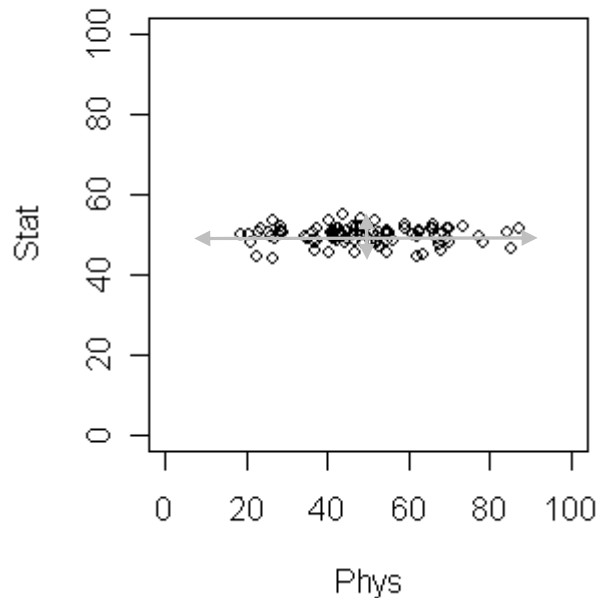
- Simple approach which is easy to implement
- The transformed vectors yields lower and upper bounds of the exact distances
(What the hell is that good for???)

Disadvantages:

- Even using d reference points does not reproduce a d -dimensional feature space
- Selecting good reference points is important but very difficult

Principal Component Analysis (PCA): A simple example 1/3

- Consider the grades of students in Physics and Statistics.
- If we want to compare among the students, which grade should be more discriminative? Statistics or Physics?



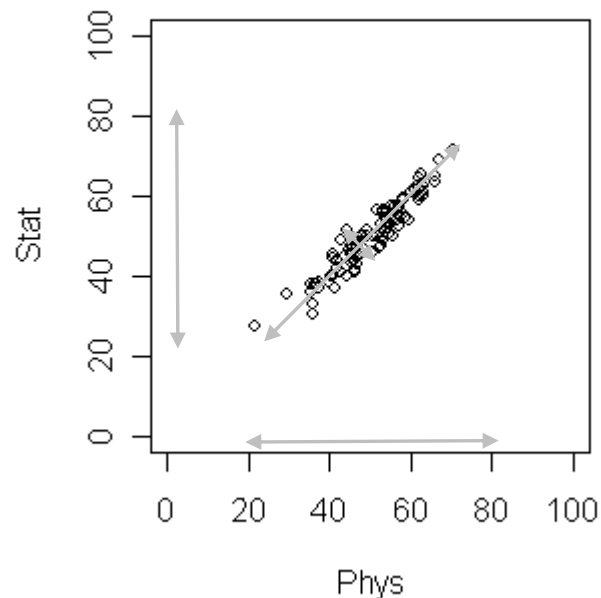
Physics since the variation along that axis is larger.

Based on:

<http://astrostatistics.psu.edu/su09/lecturenotes/pca.html>

Principal Component Analysis (PCA): A simple example 2/3

- Suppose now the plot looks as below.
- What is the best way to compare students now?



We should take a linear combination of the two grades to get the best results.

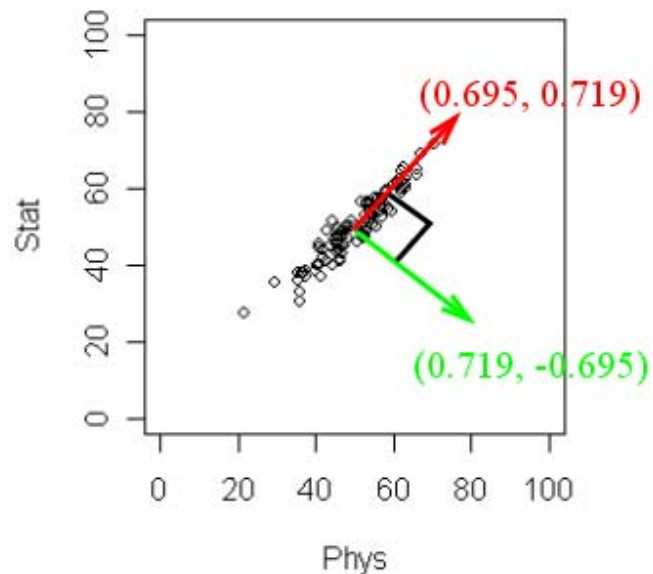
Here the direction of maximum variance is clear.

In general → PCA

Based on:
<http://astrostatistics.psu.edu/su09/lecturenotes/pca.html>

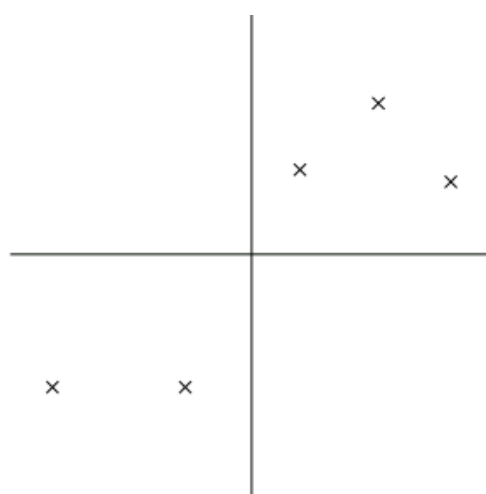
Principal Component Analysis (PCA): A simple example 3/3

- PCA returns two principal components
 - The first gives the direction of the maximum spread of the data.
 - The second gives the direction of maximum spread perpendicular to the first

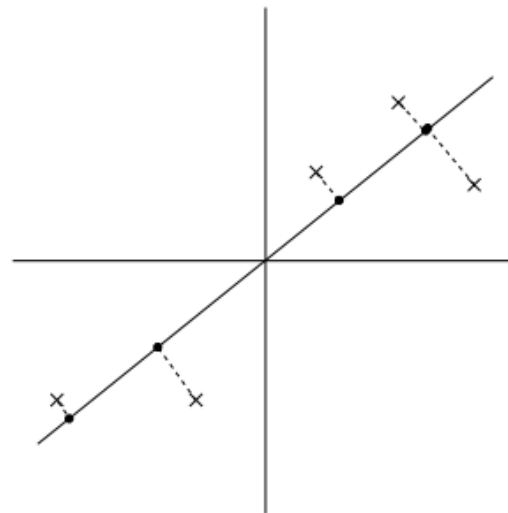


Based on:
<http://astrostatistics.psu.edu/su09/lecturenotes/pca.html>

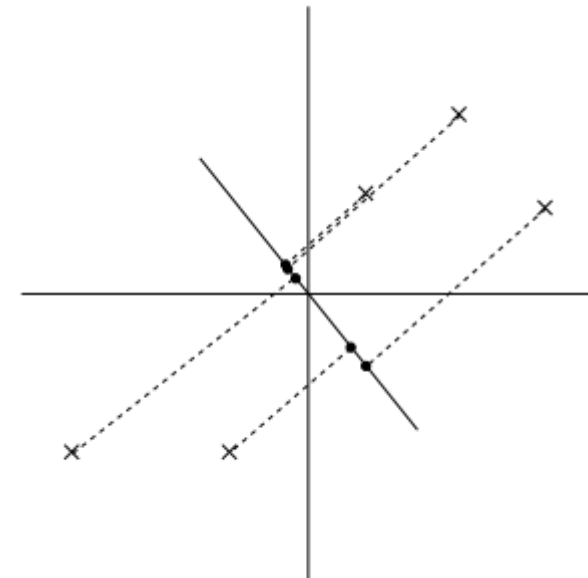
- The data starts off with some amount of variance/information in it. We would like to choose a direction u so that if we were to approximate the data as lying in the direction/subspace corresponding to u , as much as possible of this variance is still retained.



Initial data



Direction 1



Direction 2

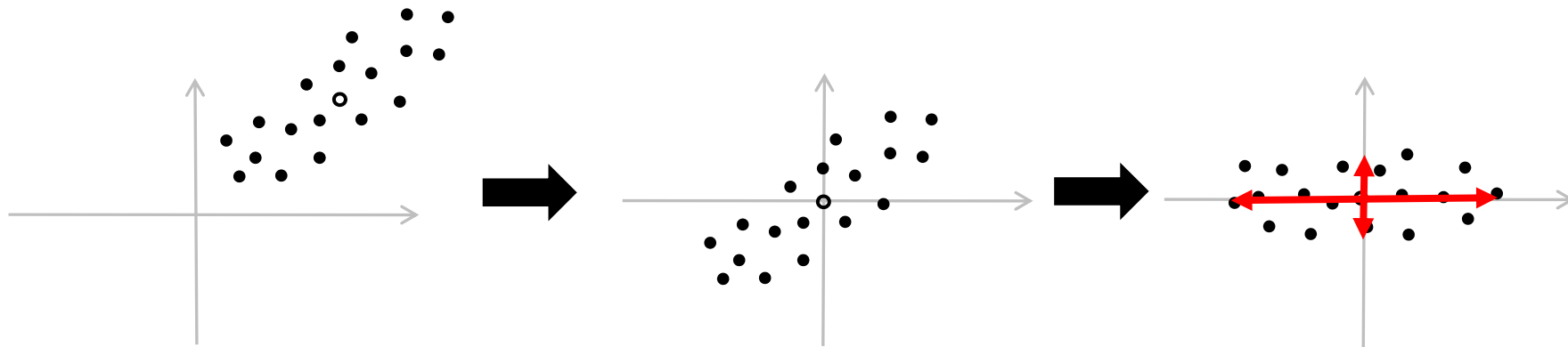
Idea: Choose the direction that maximizes the variance of the projected data (here: Dir. 1)

- PCA computes the most meaningful basis to re-express a noisy, garbled data set.
- Think of PCA as choosing a new coordinate system for the data, the principal components being the unit vectors along the axes
- PCA asks: *Is there another basis, which is a linear combination of the original basis, that best expresses our dataset?*
- General form: $PX=Y$

where P is a linear transformation, X is the original dataset and Y the new representation of this dataset.

 - P is a matrix that transforms X into Y
 - Geometrically, P is a *rotation* and a *stretch* of X to get Y
 - The eigenvectors are the rotations to the new axes
 - The eigenvalues are the amount of stretching that needs to be done
- The p 's are the principal components
 - Directions with the largest variance ... those are the most important, most *principal*.

Idea: Rotate the data space in a way that the principal components are placed along the main axis of the data space
=> Variance analysis based on principal components



- Rotate the data space in a way that the direction with the largest variance is placed on an axis of the data space
- Rotation is equivalent to a basis transformation by an orthonormal basis
 - Mapping is equal of angle and preserves distances:

$$x \cdot B = x \cdot (b_{*,1}, \dots, b_{*,d}) = (\langle x, b_{*,1} \rangle, \dots, \langle x, b_{*,d} \rangle) \text{ mit } \forall_{i \neq j} \langle b_i, b_j \rangle = 0 \wedge \forall_{1 \leq i \leq d} \|b_i\| = 1$$

- B is built from the largest variant direction which is orthogonal to all previously selected vectors in B.

- Basics of statistical measures:
 - variance
 - covariance
- Basics of linear algebra:
 - Matrices
 - Vector space
 - Basis
 - Eigenvectors, eigenvalues

- A measure of the spread of the data

$$VAR(X) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

- Variance refers to a single dimension, e.g., height

- A measure of how much two random variables vary together

$$COV(X, Y) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)$$

- What the values mean
 - Positive values: both dimensions move together (increase or decrease)
 - Negative values: while one dimension increases the other decreases
 - Zero value: the dimensions are independent of each other.

- Describes the variance of all features and the pairwise correlations between them (given the n data points)

$$\Sigma_D = \begin{pmatrix} VAR(X_1) & \cdots & COV(X_1, X_d) \\ \vdots & \ddots & \vdots \\ COV(X_d, X_1) & \cdots & VAR(X_d) \end{pmatrix}$$

$$VAR(X) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

$$COV(X, Y) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)$$

- Properties:
 - For d -dimensional data, $d \times d$ covariance matrix
 - symmetric matrix as $COV(X, Y) = COV(Y, X)$

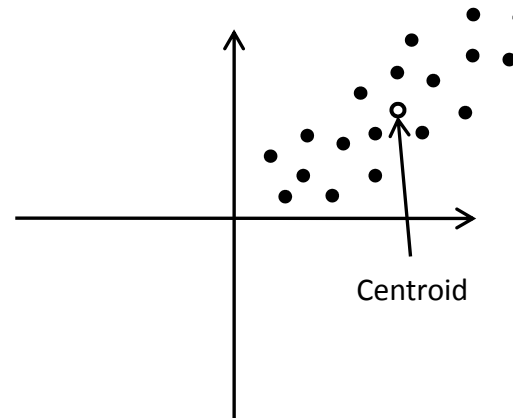
- Given n vectors $v_i \in \mathbb{R}^d$, the $n \times d$ matrix

$$D = \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} v_{1,1} & \cdots & v_{1,d} \\ \vdots & \ddots & \vdots \\ v_{n,1} & \cdots & v_{n,d} \end{pmatrix}$$

is called data matrix

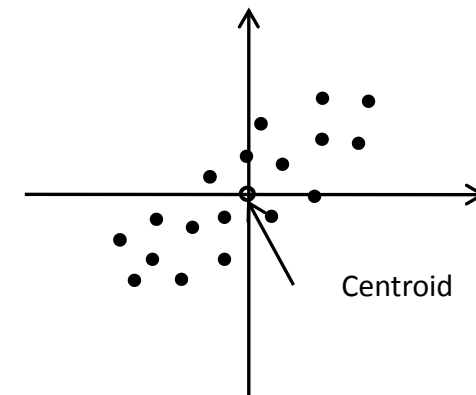
- Centroid/mean vector of D :

$$\vec{\mu} = \frac{1}{n} \sum_{i=1}^n v_i$$



- Centered data matrix:

$$D_{cent} = \begin{pmatrix} v_1 - \vec{\mu} \\ \vdots \\ v_n - \vec{\mu} \end{pmatrix}$$



- The covariance matrix can be expressed in terms of the centered data matrix as follows:

$$\Sigma_D = \begin{pmatrix} \text{VAR}(X_1) & \cdots & \text{COV}(X_1, X_d) \\ \vdots & \ddots & \vdots \\ \text{COV}(X_d, X_1) & \cdots & \text{VAR}(X_d) \end{pmatrix} = \frac{1}{n} D_{cent}^T D_{cent}$$

- Inner (dot) product of vectors x, y :

$$x \cdot y = x^T \cdot y = (x_1 \cdots x_d) \cdot \begin{pmatrix} y_1 \\ \vdots \\ y_d \end{pmatrix} = \langle x, y \rangle = \sum_{i=1}^d x_i \cdot y_i$$

- Outer product of vectors x, y :

$$x \otimes y = x \cdot y^T = \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix} \cdot (y_1 \cdots y_d) = \begin{pmatrix} x_1 y_1 & \cdots & x_1 y_d \\ \vdots & \ddots & \vdots \\ x_d y_1 & \cdots & x_d y_d \end{pmatrix}$$

- Matrix multiplication:

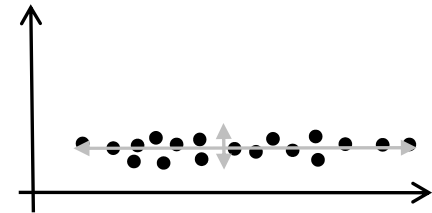
$$A = [a_{ij}]_{m \times p} \quad \text{and} \quad B = [b_{ij}]_{p \times n} \quad \text{then}$$

$$AB = C = [c_{ij}]_{m \times n} \quad \text{where} \quad c_{ij} = \text{row}_i(A) \cdot \text{col}_j(B)$$

- Length of a vector

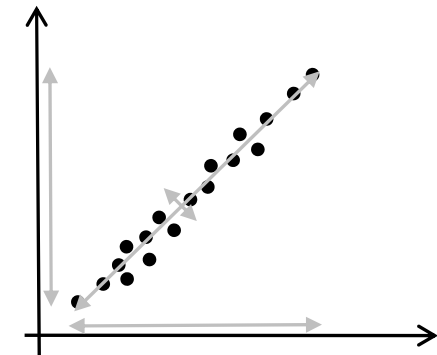
- Unit vector: if $\|a\| = 1$ where $\|a\| = \sqrt{a^T \cdot a} = \sqrt{\sum_{i=1}^n a_i^2}$

- Which attributes are the most important to the distance ?
=> **attributes with strongly varying value differences $|x_i - y_i|$**
=> distance to the mean value is large $|x_i - \mu_i|$
=> variance is large: $\frac{1}{n} \sum_{i=1}^n (x_i - \mu_i)^2$



Idea: Variance Analysis (= unsupervised feature selection)

- Attributes with large variance allow strong distinction between objects
- Attributes with small variance: difference between objects are negligible
- Method:
 - Determine the variance between the values in each dimension
 - Sort all features w.r.t. to the variance
 - Select k features having the strongest variance



Beware: Even linear correlation can distribute one strong feature over arbitrarily many other dimension!!!

- Let D be $d \times d$ square matrix.
- A non zero vector v_i is called an *eigenvector* of D if and only if there exists a scalar λ_i such that: $Dv_i = \lambda_i v_i$.
 - λ_i is called an *eigenvalue* of D .
- How to find the eigenvalues/eigenvectors of D ?
 - By solving the equation: $\det(D - \lambda I_{d \times d}) = 0$ we get the eigenvalues
 - $I_{d \times d}$ is the identity matrix
 - For each eigenvalue λ_i , we find its eigenvector by solving $(D - \lambda_i)v_i = 0$

- Let D be $d \times d$ square matrix.
- Eigenvalue decomposition of the data matrix in two $d \times d$ matrices

$$D = VE V^T$$

such that:

- $V = (v_1 \cdots v_d)$ such that $\forall_{i \neq j} \langle v_i, v_j \rangle = 0$ and $\forall_{1 \leq i \leq d} \|v_i\| = 1$

- $E = \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_d \end{pmatrix}$

Every eigenvector is a unit vector

The eigenvectors are linearly independent

The corresponding eigenvalues

i.e.:

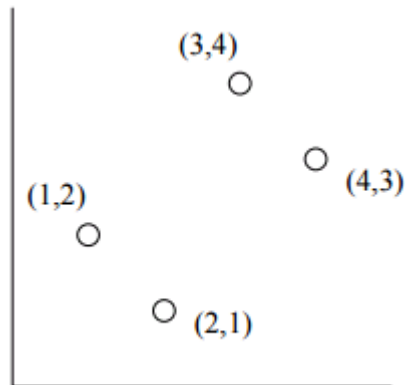
- The columns of V are the eigenvectors of D
- The diagonal elements of E are the eigenvalues of D

Feature reduction using PCA

1. Compute the covariance matrix Σ
2. Compute the eigenvalues and the corresponding eigenvectors of Σ
3. Select the k biggest eigenvalues and their eigenvectors (V')
4. The k selected eigenvectors represent an orthogonal basis
5. Transform the original $n \times d$ data matrix D with the $d \times k$ basis V' :

$$D \cdot V' = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} (v'_1, \dots, v'_k) = \begin{pmatrix} \langle x_1, v'_1 \rangle & \cdots & \langle x_1, v'_k \rangle \\ \vdots & \ddots & \vdots \\ \langle x_n, v'_1 \rangle & \cdots & \langle x_n, v'_k \rangle \end{pmatrix}$$

- Original



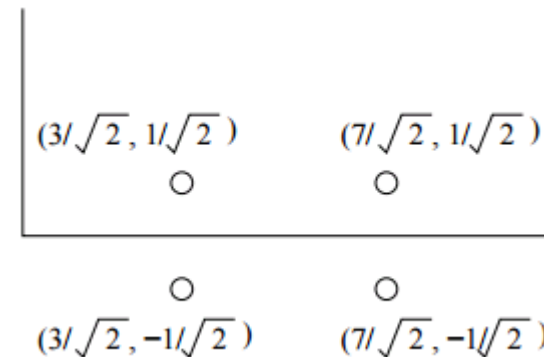
Eigenvectors

$$\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} \quad \begin{bmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

- Transformed data

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 3/\sqrt{2} & 1/\sqrt{2} \\ 3/\sqrt{2} & -1/\sqrt{2} \\ 7/\sqrt{2} & 1/\sqrt{2} \\ 7/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

In the rotated coordinate system



Source: <http://infolab.stanford.edu/~ullman/mmds/ch11.pdf>

- Let k be the number of top eigenvalues out of d (d is the number of dimensions in our dataset)
- The percentage of variance in the dataset explained by the k selected eigenvalues is:

$$\frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^d \lambda_i}$$

- Similarly, you can find the variance explained by each principal component
- Rule of thumb: keep enough to explain (at least) **85%** of the variation

- Example: iris dataset ($d=4$), results from R
- 4 principal components

```

                PC1          PC2          PC3          PC4
Sepal.Length  0.5038236 -0.45499872  0.7088547  0.19147575
Sepal.Width   -0.3023682 -0.88914419 -0.3311628 -0.09125405
Petal.Length  0.5767881 -0.03378802 -0.2192793 -0.78618732
Petal.Width   0.5674952 -0.03545628 -0.5829003  0.58044745

```

Importance of components:

```

                PC1          PC2          PC3          PC4
Proportion of Variance 0.7331 0.2268 0.03325 0.00686
Cumulative Proportion 0.7331 0.9599 0.99314 1.00000

```

=> Choose PC1 and PC2 explaining appr. 96% of the total variance

Generalization of the eigenvalue decomposition

Let $D_{n \times n}$ be the data matrix and let k be its rank (max number of independent rows/ columns).

We can decompose D into matrices O, S, A as follows $D = OSA^T$ with

$$\begin{pmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,d} \end{pmatrix} = \begin{pmatrix} o_{1,1} & \cdots & o_{1,k} \\ \vdots & \ddots & \vdots \\ o_{n,1} & \cdots & o_{n,k} \end{pmatrix} \cdot \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_k \end{pmatrix} \cdot \begin{pmatrix} a_{1,1} & \cdots & a_{1,d} \\ \vdots & \ddots & \vdots \\ a_{k,1} & \cdots & a_{k,d} \end{pmatrix}$$

such that:

O is an $n \times k$ column-orthonormal matrix ; that is, each of its columns is a unit vector and the dot product of any two columns is 0.

S is a diagonal $k \times k$ matrix; that is, all elements not on the main diagonal are 0. The elements of S are called the *singular values* of D .

A is a $k \times d$ column-orthonormal matrix. Note that we always use A in its transposed form, so it is the rows of A^T that are orthonormal.

The decomposition can be found e.g. based on numerical algorithms.

Example 1

- D: ratings of movies by users
- The corresponding SVD

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix} \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}$$

D
O
S
A^T

	Matrix	Star Wars	Alien	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

Ratings of movies by users

- Interpretation of SVD
 - O shows two concepts “science fiction” and “romance”
 - S shows the strength of these concepts
 - A relates movies to concepts

Source: <http://infolab.stanford.edu/~ullman/mmds/ch11.pdf>

Example 2

- A slightly different D
- The corresponding SVD

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} .13 & .02 & -.01 \\ .41 & .07 & -.03 \\ .55 & .09 & -.04 \\ .68 & .11 & -.05 \\ .15 & -.59 & .65 \\ .07 & -.73 & -.67 \\ .07 & -.29 & .32 \end{bmatrix} \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \\ .40 & -.80 & .40 & .09 & .09 \end{bmatrix}$$

D
O
S
A^T

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	2	0	4	4
Jenny	0	0	0	5	5
Jane	0	1	0	2	2

- Interpretation of SVD
 - O shows three concepts “science fiction” and “romance” and “”?
 - S shows the strength of these concepts
 - A relates movies to concepts

Source: <http://infolab.stanford.edu/~ullman/mmds/ch11.pdf>

- To reduce dimensionality, we can set the smallest singular values to 0 in S and eliminate the corresponding column in O and row in A^T
 - Check previous example
- How Many Singular Values Should We Retain?
 - Rule of thumb: retain enough singular values to make up 90% of the energy in S
 - Energy defined in terms of the singular values (matrix S)
 - In previous example, total energy is: $(12.4)^2 + (9.5)^2 + (1.3)^2 = 245.70$
 - The retained energy is: $(12.4)^2 + (9.5)^2 = 244.01 > 99\%$

Apply SVD to the covariance data:

- $\Sigma_D = \frac{1}{n} D_{cent}^T D_{cent}$
- $D_{cent} = OSA^T$

Recall O is orthonormal matrix, so $O^T O$ is the identity matrix

Recall S is a diagonal matrix, transposing has no effect

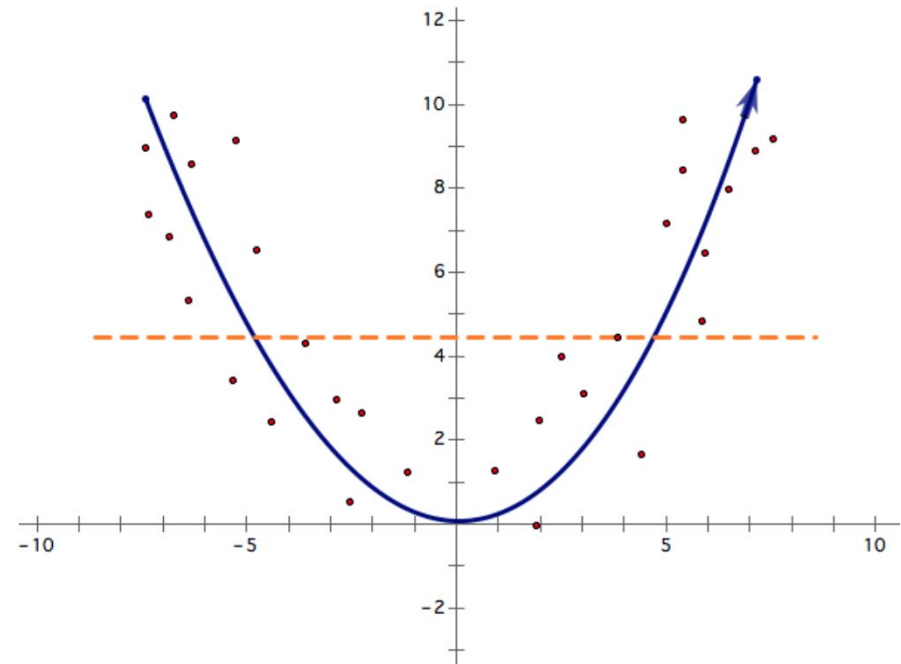
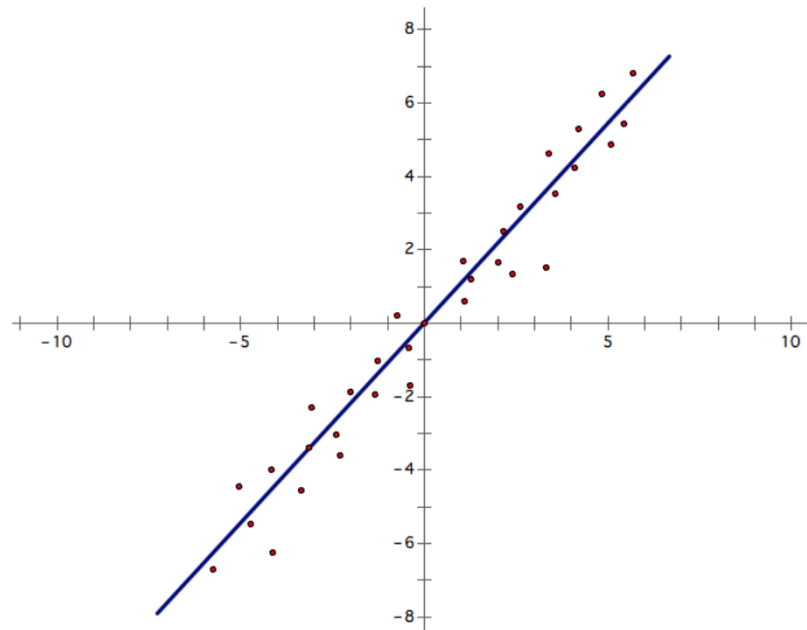
- $\Sigma_D = (OSA^T)^T OSA^T = AS^T(O^T O)SA^T = A(S^T S)A^T = A \begin{pmatrix} \lambda_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_k^2 \end{pmatrix} A^T$

- Here: A is a matrix of eigenvectors
- Eigenvalues of the covariance matrix = squared singular values of D

Conclusion: Eigenvalues and eigenvectors of the covariance matrix Σ can be determined by the SVD of the data matrix D.

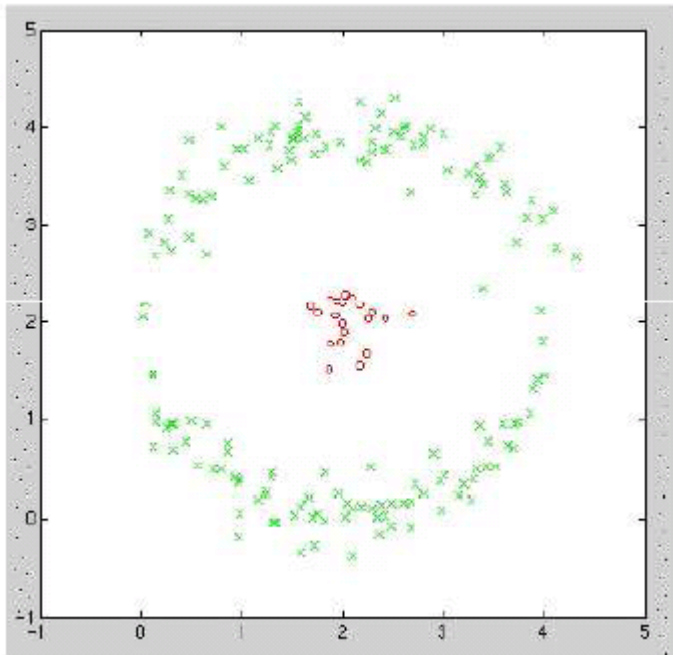
- ⇒ SVD is sometimes a better way to perform PCA (Large dimensionalities e.g., text data)
- ⇒ SVD can cope with dependent dimensions ($k < d$ is an ordinary case in SVD)

An extension of PCA using techniques of kernel methods.

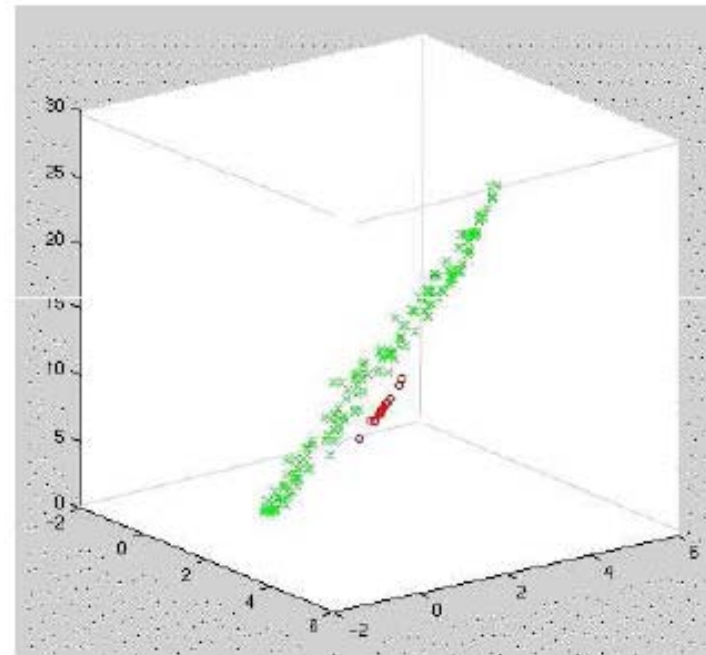


Left figure displays a 2D example in which PCA is effective because data lie near a linear subspace. In the right figure though, PCA is ineffective, because the data lie near a parabola. In this case, the PCA compression of the data might project all points onto the orange line, which is far from ideal.

- Project the data into a higher dimensional space



These classes are linearly inseparable in the input space



We can make the problem linearly separable by a simple mapping

$$\Phi: \mathbf{R}^2 \rightarrow \mathbf{R}^3$$

$$(x_1, x_2) \mapsto (x_1, x_2, x_1^2 + x_2^2)$$

- Wait a minute! Seriously? You suggest to pump up the feature space to get a better discriminability of points?

And how does that compare to the curse of dimensionality?

- Well: look at all that stuff we did a little closer.
- Results on (un)stability of distances and neighborhoods are based on the assumption that you add features that are
 - Independent
 - Randomly distributed
- Using a Kernel, you do a (completely) different thing
 - You add „relevant“ features that are combinations of others (i.e. not independent and probably not random)
 - ***In fact, there is a curse AND a blessing in high dimensions***

$$\Phi: \mathbf{R}^2 \rightarrow \mathbf{R}^3$$

$$(x_1, x_2) \mapsto (x_1, x_2, x_1^2 + x_2^2)$$

- But: high-dimensional mapping can seriously increase computation time.
- Can we get around this problem and still get the benefit of high dimensions?
- Yes! Kernel Trick

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

- Different types of kernels
 - Polynomial
 - Gaussian
 - ...

Example: Polynomial kernel

- For degree- d polynomials, the polynomial kernel is defined as

$$K(x, y) = (x^T y + c)^d$$

- Example:

$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

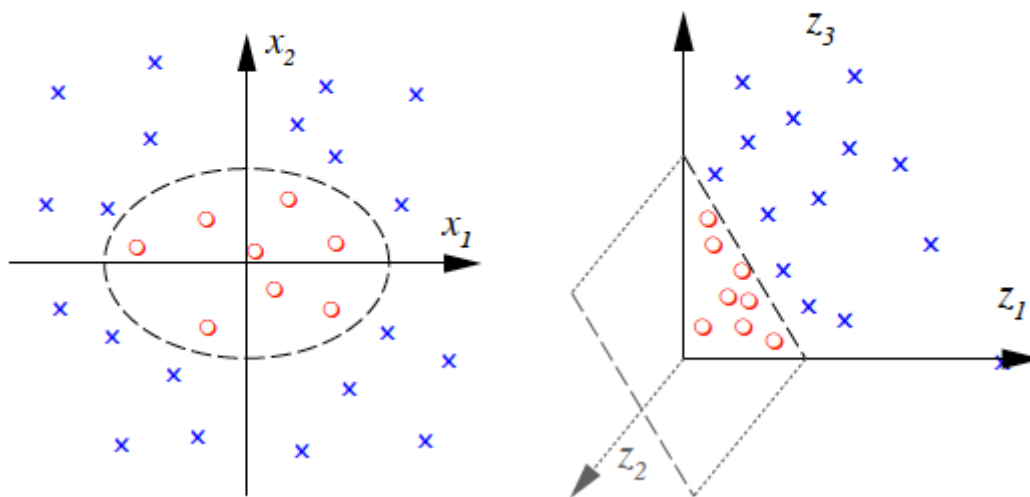


Image from: <http://i.stack.imgur.com/qZV3s.png>

Connection between the orthonormal basis O und A : $D = OSA^T$

- A is a k -dimensional basis of eigenvectors of $D^T \cdot D$
(cf. previous slide)
- Analogously: O is a k -dimension basis of Eigenvectors $D \cdot D^T$
 - $D \cdot D^T$ is a kernel matrix for the linear kernell $\langle x, y \rangle$ (cf. SVMs in KDD I)
 - The vectors of A and O are connected in the following way:

$$D_{cent} = OSA^T \Rightarrow O^T D_{cent} = O^T OSA^T = SA^T \Rightarrow S^{-1} O^T D_{cent} = A^T$$

$$\Rightarrow a_j = \sum_{i=1}^n o_{i,j} x_i$$

The j^{th} d -dimensional eigenvector in A is a linear combination of the vectors in D based on k -dimensional j^{th} eigenvectors as weighting vector (the i^{th} values is the weight for vector d_i)

\Rightarrow A basis in vector space corresponds to a basis in the kernel space

\Rightarrow A PCA can be computed for any kernel space based on the kernel matrix
(Kernel PCA allows PCA in a non-linear transformation of the original data)

Let $K(x, y) = \langle \Phi(x), \Phi(y) \rangle$ be a kernel for the non-linear transformation $\Phi(x)$.

Assume: $K(x, y)$ is known, but $\Phi(x)$ is not explicitly given.

- Let K be the kernel matrix of D w.r.t. $K(x, y)$: $K = \begin{pmatrix} K(x_1, x_1) & \cdots & K(x_1, x_n) \\ \vdots & \ddots & \vdots \\ K(x_n, x_1) & \cdots & K(x_n, x_n) \end{pmatrix}$
- The eigenvalue decomposition of K : $K = VSV^T$
where V is a n -dimensional basis from eigenvectors of K
- To map D w.r.t. V the principal components in the target space the vectors x_i in D *must be transformed using the kernel $K(x, y)$ as follows:*

$$y' = \begin{pmatrix} \langle \Phi(y), \sum_{i=1}^n v_{i,1} \Phi(x_i) \rangle \\ \vdots \\ \langle \Phi(y), \sum_{i=1}^n v_{i,k} \Phi(x_i) \rangle \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n v_{i,1} \langle \Phi(y), \Phi(x_i) \rangle \\ \vdots \\ \sum_{i=1}^n v_{i,k} \langle \Phi(y), \Phi(x_i) \rangle \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n v_{i,1} K(y, x_i) \\ \vdots \\ \sum_{i=1}^n v_{i,k} K(y, x_i) \end{pmatrix}$$

SVD and PCA are standard problems in Algebra.

- Matrix decomposition can be formulated as an optimization task.
- This allows a computation via numerical optimization algorithms
- In this formulation the diagonal matrix is often distributed to both basis matrixes

$$D = ASB^T = \left(A \begin{pmatrix} \sqrt{\lambda_1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sqrt{\lambda_k} \end{pmatrix} \right) \left(\begin{pmatrix} \sqrt{\lambda_1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sqrt{\lambda_k} \end{pmatrix} B^T \right) = UV^T$$

- As an optimization problem: $L(U, V) = \|D - UV^T\|_f^2$
 (squared Frobenius Norm of a matrix) $\|M\|_f^2 = \sum_{i=1}^n \sum_{j=1}^m |m_{i,j}|^2$

subject to: $\forall_{i \neq j} : \langle v_i, v_j \rangle = 0 \wedge \langle u_i, u_j \rangle = 0$

Idea: Use examples to increase the discriminative power of the target space.

Target:

- Minimize the similarity between objects from different classes.

(between class scatter matrix: Σ_b)

Σ_b : Covariance matrix of the class centroids

- Maximize similarity between objects belonging to the same class

(within class scatter matrix Σ_w)

Σ_w : Average covariance matrix of all classes.

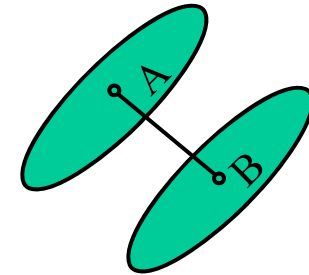
Solution:

- Determine basis x_i in a way that
$$S = \frac{x_i^T \cdot \Sigma_b \cdot x_i}{x_i^T \cdot \Sigma_w \cdot x_i}$$

is maximized subject to $i \neq j : \langle x_i, x_j \rangle = 0$

$$\Sigma_b = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

$$\Sigma_w = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$



Remark: The vector having the largest eigenvalue corresponds to the normal vector of the separating hyper plane in linear discriminant analysis or Fisher's discriminant analysis. (cf. KDD I)

Discussion: Fischer Faces are limited due to the assumption of mono-modal classes: each class is assumed to follow a multivariate

Conclusion: Multi-modal or non-Gaussian distributions are not modeled well

Relevant Component Analysis (RCA):

- Remove linear dependent features (e.g. with SVD).
- Given: chunks of data which are known to consist of similar objects.
=> replace Σ_w with an within-chunk matrix:
- The covariance of all data objects is dominated by dissimilarity
=> replace Σ_b with the covariance matrix of D

Large Margin Nearest Neighbor (LMNN):

- Objects in a class might vary rather strongly.
- Idea: Define an optimization problem only considering the distances of the most similar objects from the same and other classes.

If you want to know the details ...

Define: $y_{i,j}=1$ if x_i and x_j are from the same class else $y_{i,j}=0$

- *Target:* $L: \mathbb{R}^d \rightarrow \mathbb{R}^d$ linear transformation of the vector space: $D(x, y) = \|L(x) - L(y)\|^2$
- *Target neighbors:* T_x k-nearest neighbors from the same class
 $\eta_{i,j} = 1$: x_j is a target neighbor of x_i else $\eta_{i,j} = 0$
- Training by minimizing the following error function:

$$E(L) = \sum_{i=1}^n \sum_{j=1}^n \eta_{i,j} \|L(x_i) - L(x_j)\|^2 + c \sum_{i=1}^n \sum_{j=1}^n \sum_{l=1}^n \eta_{i,j} (1 - y_{i,l}) \left[1 + \|L(x_i) - L(x_j)\|^2 - \|L(x_i) - L(x_l)\|^2 \right]_+$$

where $[z]_+ = \max(z, 0)$

- Problem is a **semi-definite program**
 \Rightarrow Standard optimization problem where the optimization parameters must form a semi-definite matrix. Here the matrix is the basis transformation $L(x)$.

- Linear basis transformation yield a rich framework to optimize feature spaces
- Unsupervised methods delete low variant dimensions (PCA und SVD)
- Kernel PCA allows to compute PCA in non-linear kernel spaces
- Supervised methods try to minimize the within class distances while maximizing between class distances
- Fischer Faces extend linear discriminant analysis based on the assumption that all classes follow Gaussian distributions
- Relevant Component Analysis(RCA) generalize this notion and only minimize the distances between chunks of similar objects
- Large Margin Nearest Neighbor(LMNN) minimizes the distances to the nearest target neighbors and punish small distances to non-target neighbors in other classes

- S. Deerwester, S. Dumais, R. Harshman: *Indexing by Latent Semantic Analysis*, Journal of the American Society of Information Science, Vol. 41, 1990
- L. Yang and R. Jin. Distance metric learning: A comprehensive survey. Technical report, Department of Computer Science and Engineering, Michigan State University, 2006.
- K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. Journal of Machine Learning Research, 10:207,244, 2009.
- P. Comon. Independent component analysis, a new concept? Signal Processing, 36(3):287{314, 1994.
- J. Davis, B. Kulis, S. Sra, and I. Dhillon. Information theoretic metric learning. In in NIPS 2006 Workshop on Learning to Compare Examples, 2007.
- A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In Proceedings of the 20th International Conference on Machine Learning (ICML), Washington, DC, USA, pages 11-18, 2003.