GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

LECTURE 13: DIMENSIONALITY REDUCTION

January 21, 2015

DAT11-SF

LAST TIME:

- CLUSTER ANALYSIS
- K-MEANS CLUSTERING
- HIERARCHICAL CLUSTERING

AGENDA

I. DIMENSIONALITY REDUCTION
II. PRINCIPAL COMPONENTS ANALYSIS
III. SINGULAR VALUE DECOMPOSITION

EXERCISE:
IV. DIMENSIONALITY REDUCTION IN SCIKIT-LEARN

I. DIMENSIONALITY REDUCTION

Q: What is dimensionality reduction?
A: A set of techniques for reducing the size (in terms of features, records, and/or bytes) of the dataset under examination.

In general, the idea is to regard the dataset is a matrix and to decompose the matrix into simpler, meaningful pieces.

Dimensionality reduction is frequently performed as a pre-processing step before another learning algorithm is applied.

Q: What are the motivations for dimensionality reduction?

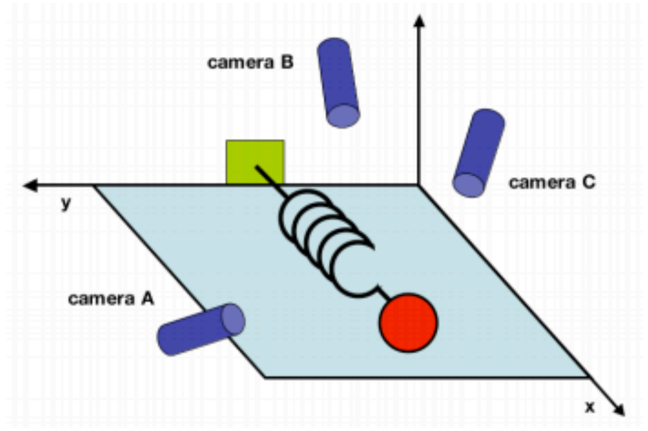
The number of features in our dataset can be difficult to manage, or even misleading (eg, if the relationships are actually simpler than they appear).

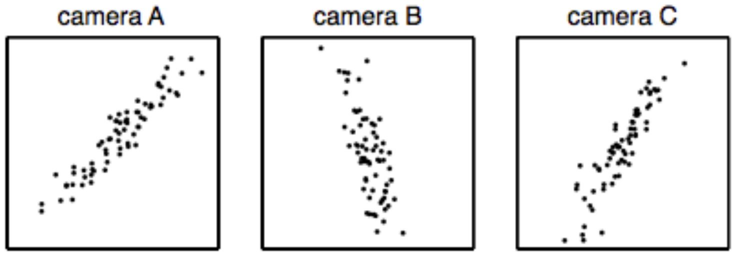
For example, suppose we have a dataset with some features that are related to each other.

Ideally, we would like to eliminate this redundancy and consolidate the number of variables we're looking at.

If these relationships are *linear*, then we can use well-established techniques like PCA/SVD.

EXAMPLE: 1D HARMONIC OSCILLATOR





source: http://www.snl.salk.edu/~shlens/pca.pdf

Q: What is the goal of dimensionality reduction?

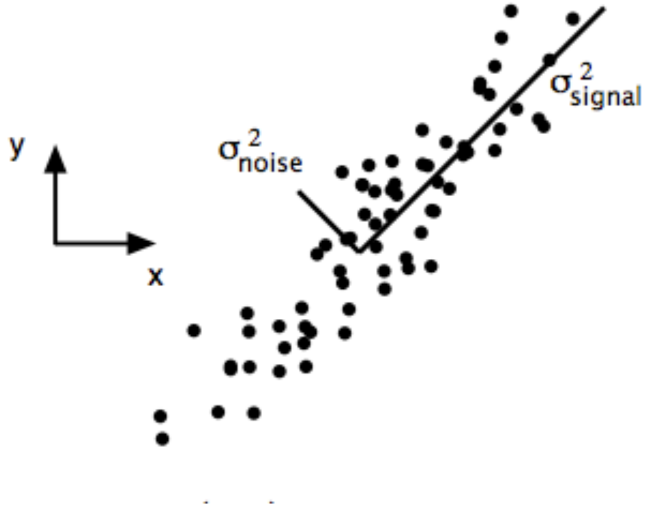
We'd like to analyze the data using the most meaningful basis (or **coordinates**) possible.

More precisely: given an m x n matrix A (encoding m observations of a n-dimensional random variable), we want to find a k-dimensional representation of A ($k < n$) that captures most of the information in the original data, according to some criterion.

Q: What is the goal of dimensionality reduction?

- reduce computational expense
- reduce susceptibility to overfitting
- reduce noise in the dataset
- enhance our intuition

DENOISING



source: http://www.snl.salk.edu/~shlens/pca.pdf

Some applications of dimensionality reduction:

- topic models (document clustering)
- image recognition/computer vision
- bioinformatics (microarray analysis)
- speech recognition
- astronomy (spectral data analysis)
- recommender systems

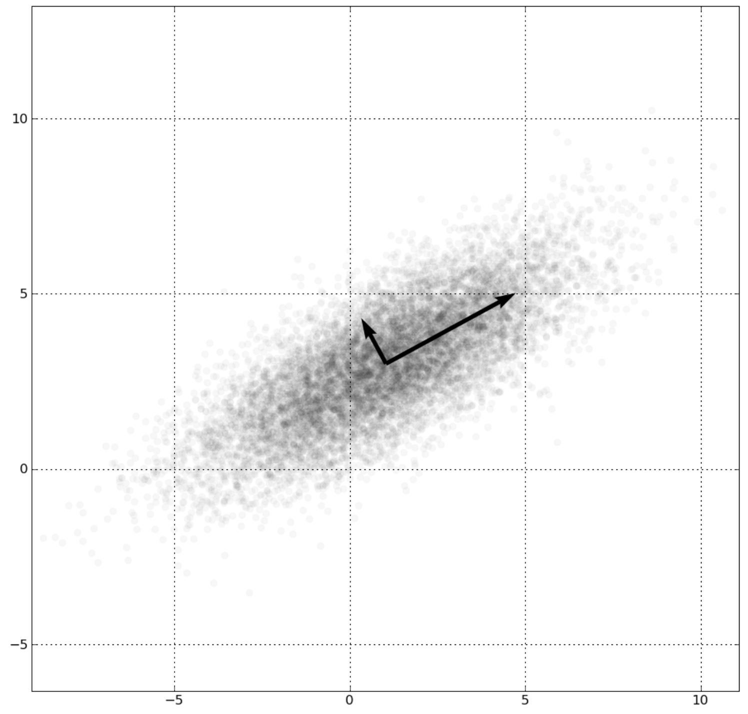
II. PRINCIPAL COMPONENT ANALYSIS

Principal component analysis is an:

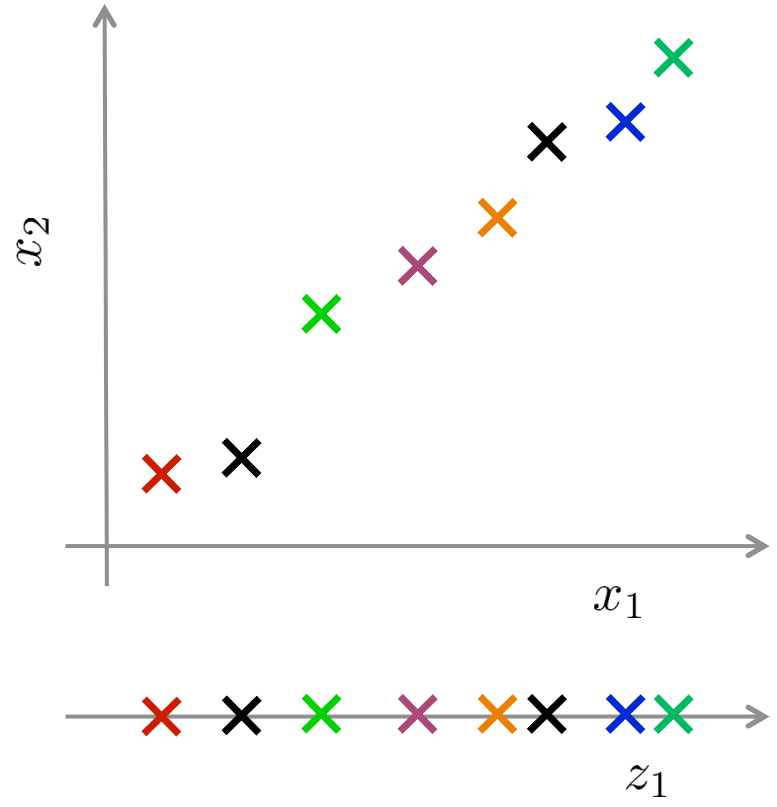
- orthogonal projection of the data onto a lower dimensional space such that the variance of the projection is maximized
- orthogonal projection onto the lower dimensional subspace that minimizes average reconstruction error

The PCA of a matrix A boils down to the **eigenvalue decomposition** of the **covariance matrix** of A.

PRINCIPAL COMPONENT ANALYSIS



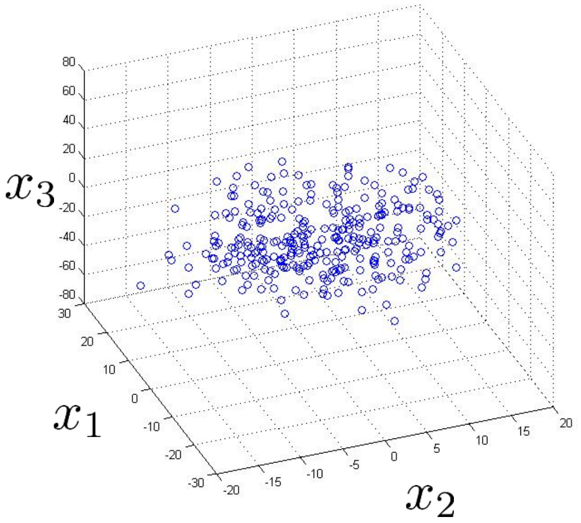
EXAMPLE

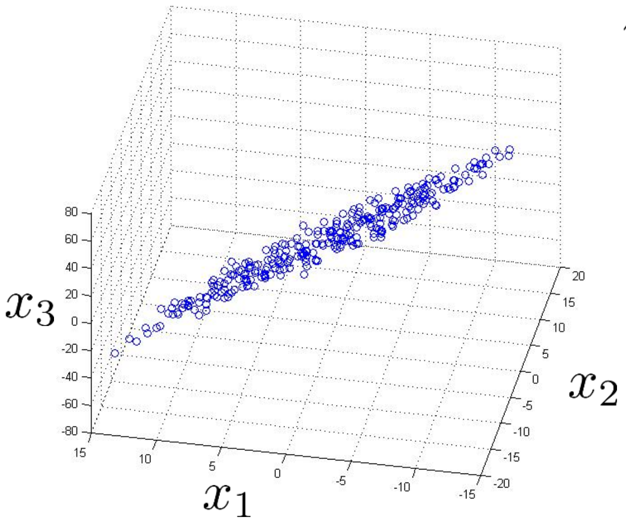


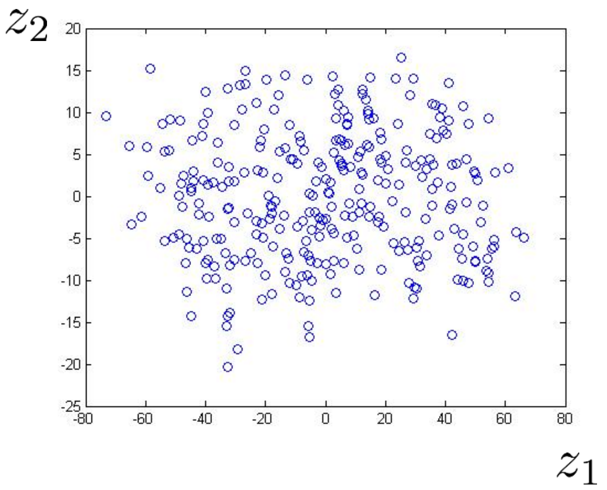
Reduce data from 2D to 1D

$$x^{(1)} \rightarrow z^{(1)}$$
$$x^{(2)} \rightarrow z^{(2)}$$
$$\vdots$$
$$x^{(m)} \rightarrow z^{(m)}$$

EXAMPLE

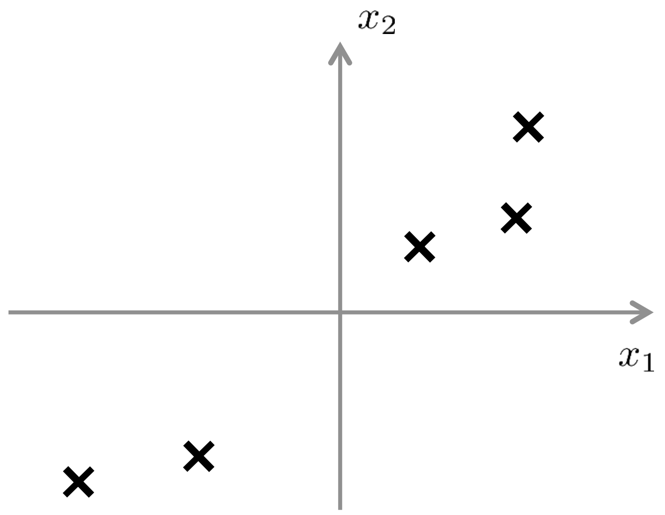




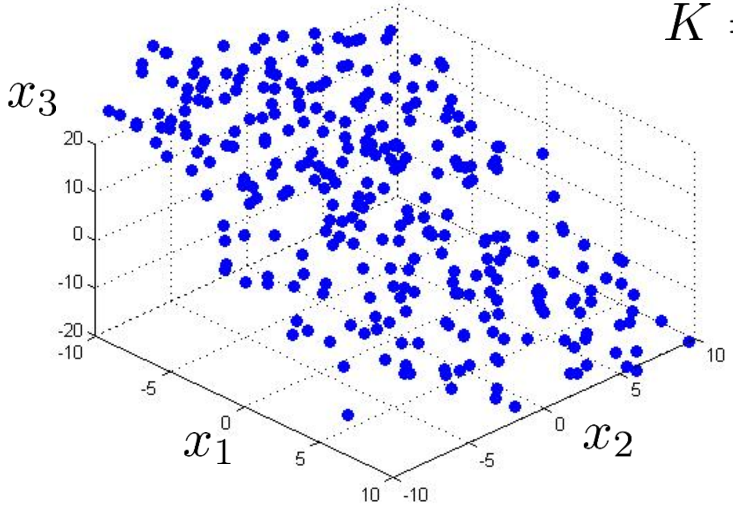


Reduce data from 3D to 2D

PCA

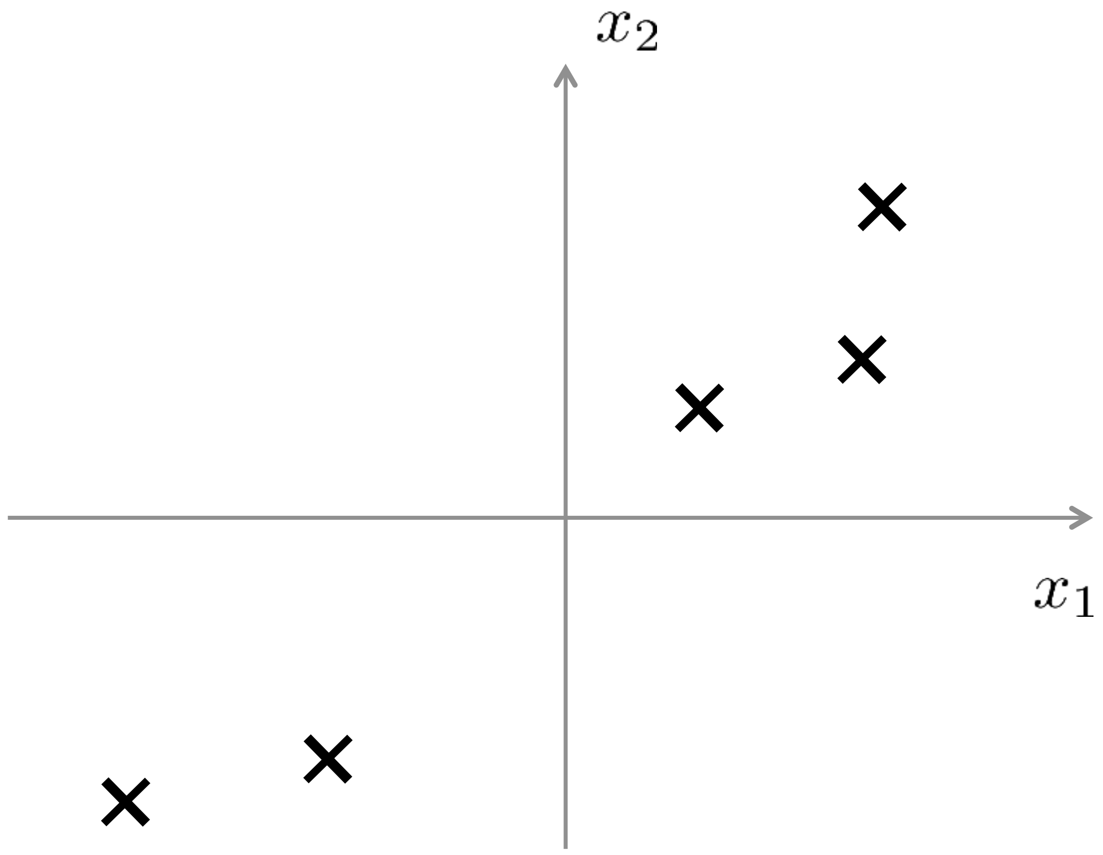


$3D \to 2D$
$K = 2$



Find a direction (a vector) $u^{(1)} \in \mathbb{R}^n$
onto which to project the data so as to minimize the projection error.

Find k vectors $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$
onto which to project the data, so as to minimize the projection error.


x_2
x_1

The covariance matrix C of a matrix A is always square:

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

off-diagonal elements C_{ij} give the *covariance* between X_i, X_j $(i \neq j)$
diagonal elements C_{ii} give the *variance* of X_i

ASIDE: EIGENVALUE DECOMPOSITION

The *eigenvalue decomposition* of a square matrix A is given by:

$$A = Q\Lambda Q^{-1}$$

The columns of Q are the **eigenvectors** of A, and the values in Λ are the associated **eigenvalues** of A.

For an eigenvector v of A and its eigenvalue λ, we have the important relation:

$$Av = \lambda v$$

SINGULAR VALUE DECOMPOSITION

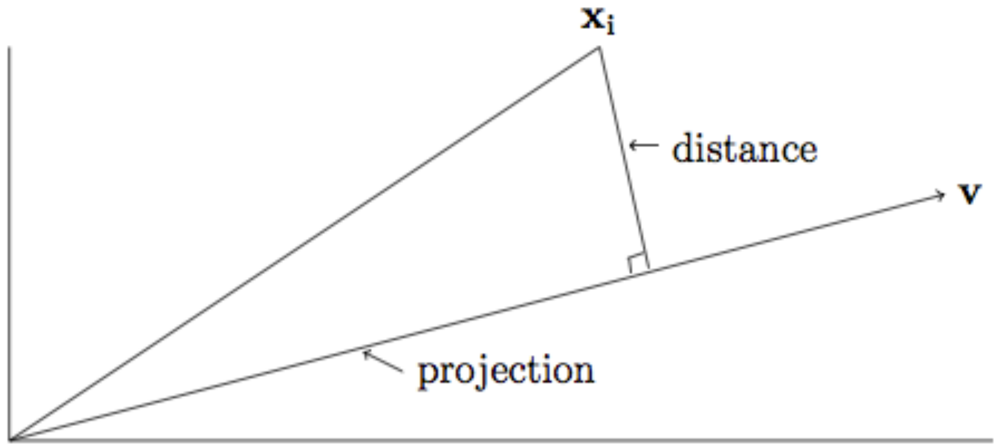


Figure 4.1: The projection of the point $\mathbf{x_i}$ onto the line through the origin in the direction of $\mathbf{v}$

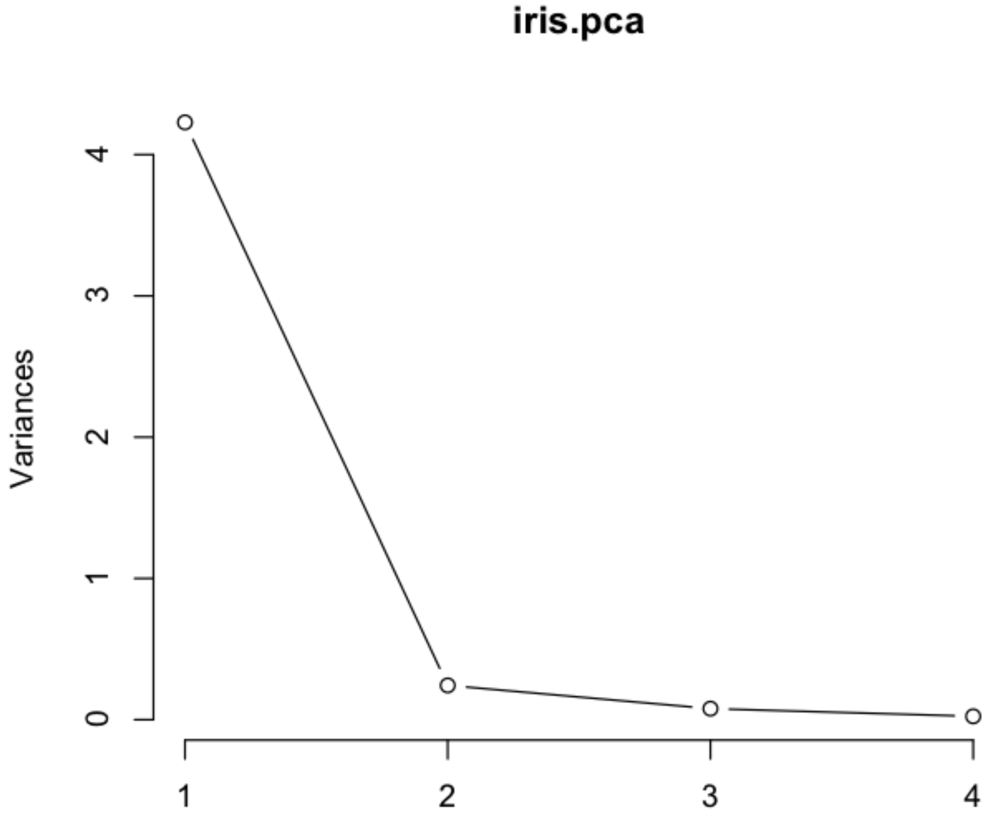
source: http://www.cs.princeton.edu/courses/archive/spring12/cos598C/svdchapter.pdf

The eigenvectors form a basis of the vector space on which A acts (eg, they are orthogonal).

Furthermore the basis elements are ordered by their eigenvalues (from largest to smallest), and these eigenvalues represent the amount of variance explained by each basis element.

This can be visualized in a **scree plot**, which shows the amount of variance explained by each basis vector.

PRINCIPAL COMPONENT ANALYSIS



III. SINGULAR VALUE DECOMPOSITION

SINGULAR VALUE DECOMPOSITION

Consider a matrix A with m rows and n features.

The **singular value decomposition** of A is given by:

$$\underset{(\mathrm{m\ x\ n})}{A} = \underset{(\mathrm{m\ x\ n})}{U}\ \underset{(\mathrm{n\ x\ n})}{\Sigma}\ \underset{(\mathrm{n\ x\ n})}{V^T}$$

st. U, V are **orthogonal** matrices and Σ is a **diagonal** matrix.

→ $UU^T = I_m,\ VV^T = I_n$ → $\Sigma_{ij} = 0\ \ (i \neq j)$

The **singular value decomposition** of A is given by:

$$A = U \; \Sigma \, V^T$$

)

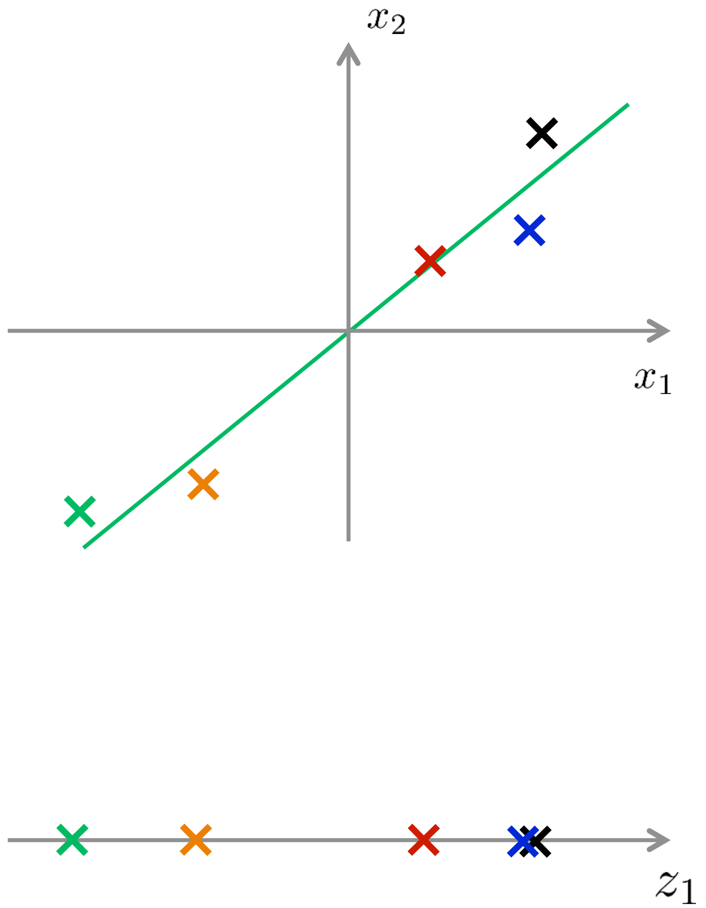
The nonzero entries of Σ are the **singular values** of A. These are real, nonnegative, and *rank-ordered* (decreasing from left to right).

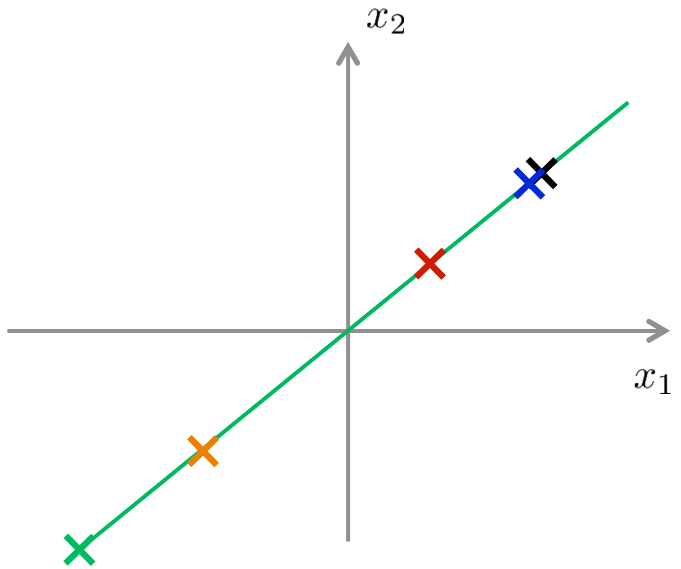
SINGULAR VALUE DECOMPOSITION

For a general SVD, the columns of U are the eigenvectors of AA^T, and the columns of V are the eigenvectors of A^TA.

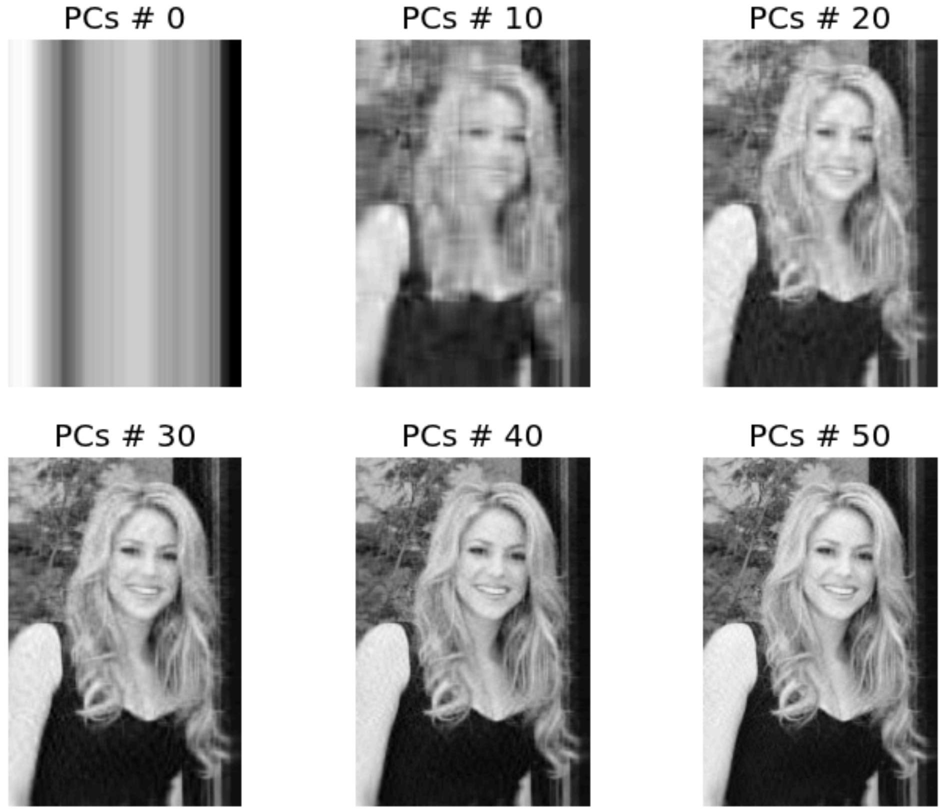
Also, the singular values of A are the square roots of the eigenvalues of AA^T and A^TA.

RECONSTRUCTION





DIMENSIONALITY REDUCTION



source: http://glowingpython.blogspot.it/2011/07/pca-and-image-compression-with-numpy.html

III. OTHER METHODS

SVD, PCA, and factor analysis are all linear techniques (eg, we use a linear transformation to embed the in a lower-dimensional space).

But as we saw with SVM's, sometimes linear techniques are not sufficient.

NONLINEAR METHODS

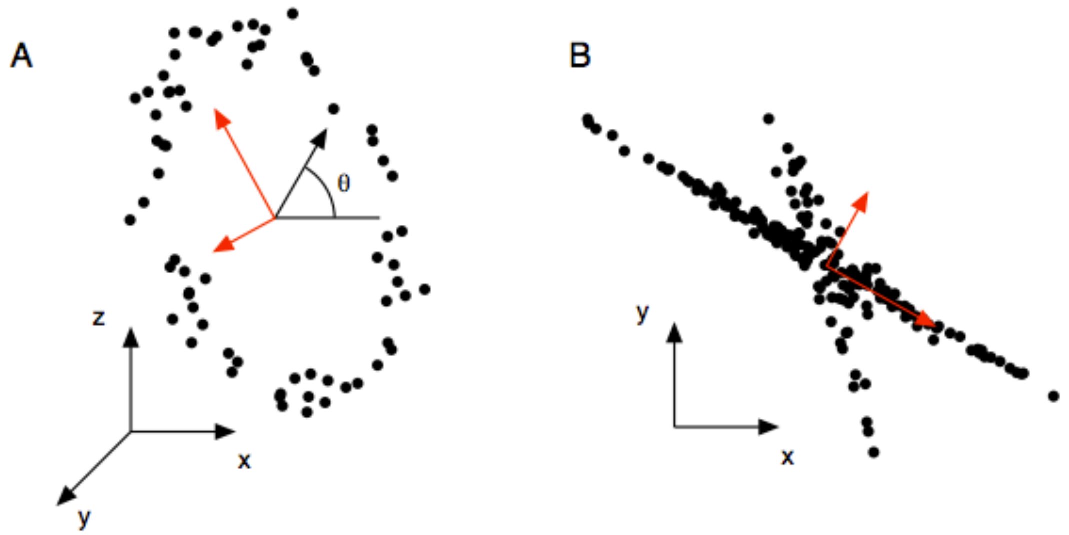
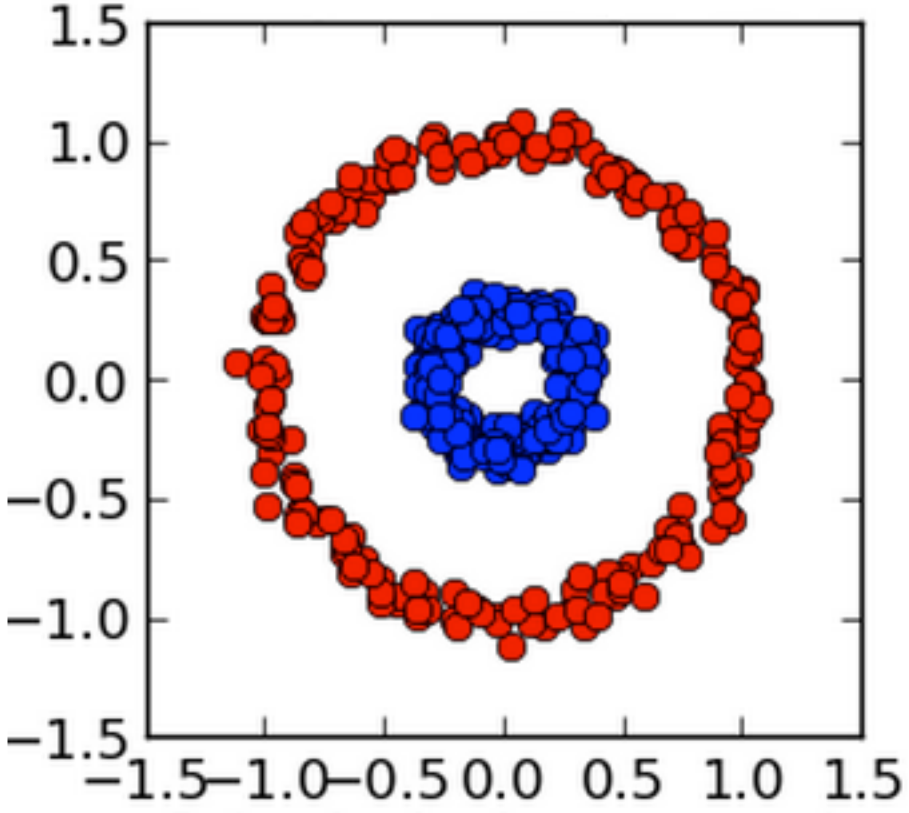


FIG. 6 Example of when PCA fails (red lines). (a) Tracking a person on a ferris wheel (black dots). All dynamics can be described by the phase of the wheel θ, a non-linear combination of the naive basis. (b) In this example data set, non-Gaussian distributed data and non-orthogonal axes causes PCA to fail. The axes with the largest variance do not correspond to the appropriate answer.

NONLINEAR METHODS



Some methods for nonlinear dimensional reduction (or *manifold learning*) include:

multidimensional scaling: low-dim embedding that preserves pairwise distances

locally linear embedding: approximates local structure of data (nbd preserving embedding)

Some methods for nonlinear dimensional reduction (or *manifold learning*) include:

kernel PCA: exploits PCA dependence on inner product (same logic as SVM)

isomap: nonlinear dim reduction via MDS using geodesic (surface-bound) distances