GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

LECTURE 7: DIMENSIONALITY REDUCTION (REPRISE)

Francesco Mosconi
DAT10 SF // October 27, 2014

DATA SCIENCE IN THE NEWS

Predicting the NFL Using Twitter

Shiladitya Sinha[1], Chris Dyer[1], Kevin Gimpel[2], and Noah A. Smith[1]

[1] Carnegie Mellon University, Pittsburgh PA 15213, USA
[2] Toyota Technological Institute at Chicago, Chicago IL 60637, USA

Abstract. We study the relationship between social media output and National Football League (NFL) games, using a dataset containing messages from Twitter and NFL game statistics. Specifically, we consider tweets pertaining to specific teams and games in the NFL season and use them alongside statistical game data to build predictive models for future game outcomes (which team will win?) and sports betting outcomes (which team will win with the point spread? will the total points be over/under the line?). We experiment with several feature sets and find that simple features using large volumes of tweets can match or exceed the performance of more traditional features that use game statistics.

source: http://arxiv.org/pdf/1310.6998v1.pdf

Daily Stress Recognition from Mobile Phone Data, Weather Conditions and Individual Traits

Andrey Bogomolov
University of Trento,
SKIL Telecom Italia Lab
Via Sommarive, 5
I-38123 Povo - Trento, Italy
andrey.bogomolov@unitn.it

Bruno Lepri
Fondazione Bruno Kessler
Via Sommarive, 18
I-38123 Povo - Trento, Italy
lepri@fbk.eu

Michela Ferron
Fondazione Bruno Kessler
via Sommarive, 18
I-38123 Povo - Trento, Italy
ferron@fbk.eu

Fabio Pianesi
Fondazione Bruno Kessler
Via Sommarive, 18
I-38123 Povo - Trento, Italy
pianesi@fbk.eu

Alex (Sandy) Pentland
MIT Media Lab
20 Ames Street
Cambridge, MA, USA
pentland@mit.edu

ABSTRACT

Research has proven that stress reduces quality of life and causes many diseases. For this reason, several researchers devised stress detection systems based on physiological parameters. However, these systems require that obtrusive sensors are continuously carried by the user. In our paper, we propose an alternative approach providing evidence that daily stress can be reliably recognized based on behavioral metrics, derived from the user's mobile phone activity and from additional indicators, such as the weather conditions (data pertaining to transitory properties of the environment) and the personality traits (data concerning permanent dispositions of individuals). Our multifactorial statistical model, which is person-independent, obtains the accuracy score of 72.28% for a 2-class daily stress recognition problem. The model is efficient to implement for most of multimedia applications due to highly reduced low-dimensional feature space (32d). Moreover, we identify and discuss the indicators which have strong predictive power.

Keywords

stress recognition; mobile sensing; pervasive computing

1. INTRODUCTION

Nowadays, the number of mobile phones in use worldwide is about 5 billion, with millions of new subscribers everyday [1]. Mobile phones allow for unobtrusive and cost-efficient access to huge streams of previously inaccessible data related to daily social behavior [29]. These devices are able to sense a wealth of behavioral data such as (i) location, (ii) other devices in physical proximity through Bluetooth scanning, (iii) communication data, including both metadata (logs of who, when, and duration) of phone calls and text messages (sms), etc. Correspondingly, the availability is continuously growing of huge streams of *personal data* related to activities, routines and social interactions [11, 29] which represent a novel opportunity to address fundamental problems of our societies in different fields, such as mobility and urban planning [19], finance [46], healthy living and subjective

source: http://arxiv.org/pdf/1410.5816v1.pdf

LAST TIME:

- DIMENSIONALITY REDUCTION
- PRINCIPAL COMPONENTS ANALYSIS

QUESTIONS?

I. RECAP: DIMENSIONALITY REDUCTION AND PCA
II. SINGULAR VALUE DECOMPOSITION
III. OTHER METHODS

EXERCISE:
IV. DIMENSIONALITY REDUCTION USING SCIKIT-LEARN
V. EXERCISE IN PAIRS

I. RECAP: DIMENSIONALITY REDUCTION

	Continuous	**Categorical**
Supervised	regression	classification
Unsupervised	dimension reduction	clustering

Q: What are the motivations for dimensionality reduction?

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

EXAMPLE: 1D HARMONIC OSCILLATOR

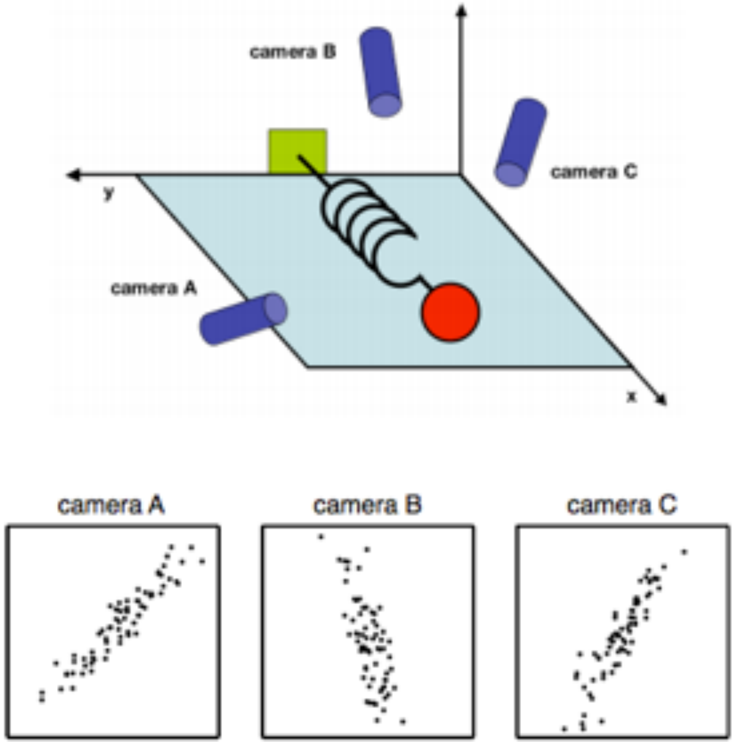


FIG. 1 A toy example. The position of a ball attached to an oscillating spring is recorded using three cameras A, B and C. The position of the ball tracked by each camera is depicted in each panel below.

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

Q: What is the goal of dimensionality reduction?

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

Q: How is dimensionality reduction performed?
A: There are two approaches: feature selection and feature extraction.

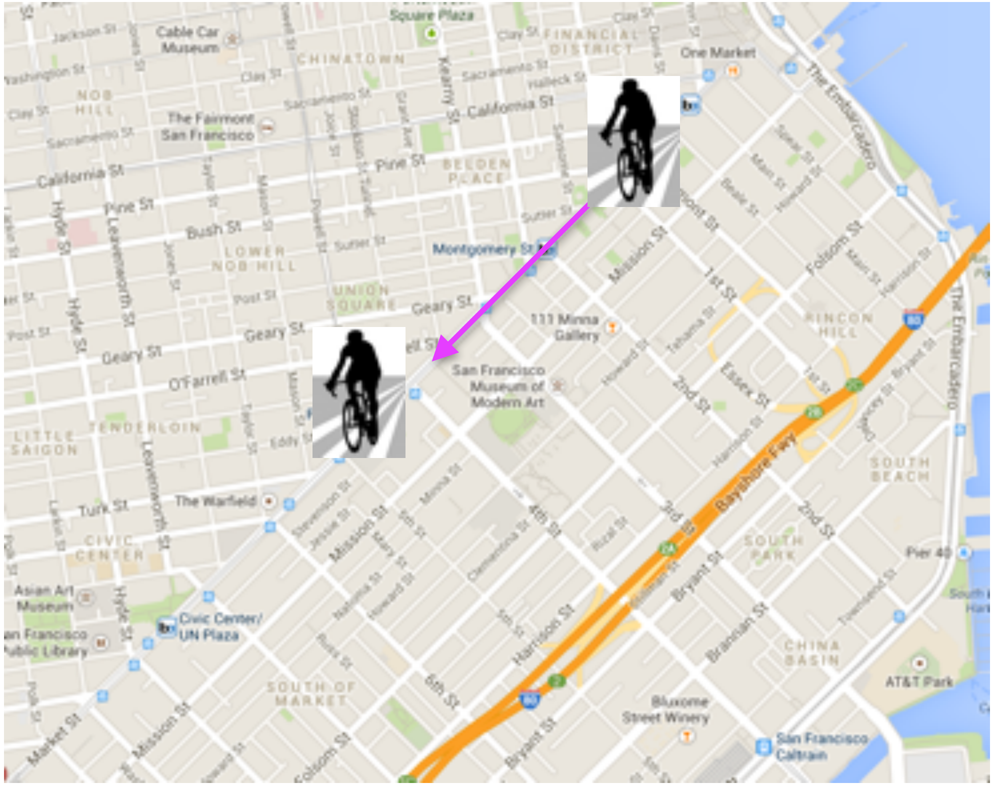
feature selection – selecting a subset of features using an external criterion (*filter*) or the learning algo accuracy itself (*wrapper*)

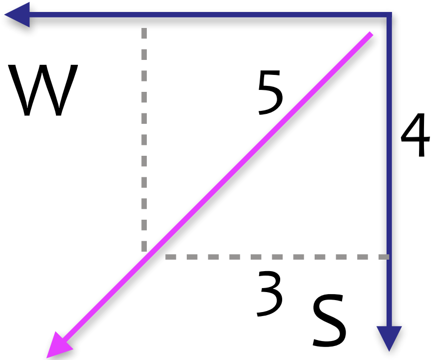
feature extraction – mapping the features to a lower dimensional space

Feature selection is important, but typically when people say dimensionality reduction, they are referring to *feature extraction*.

The goal of feature extraction is to create a new set of coordinates that *simplify the representation* of the data.

INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET





Of course, we can always map back to the original coordinate system!

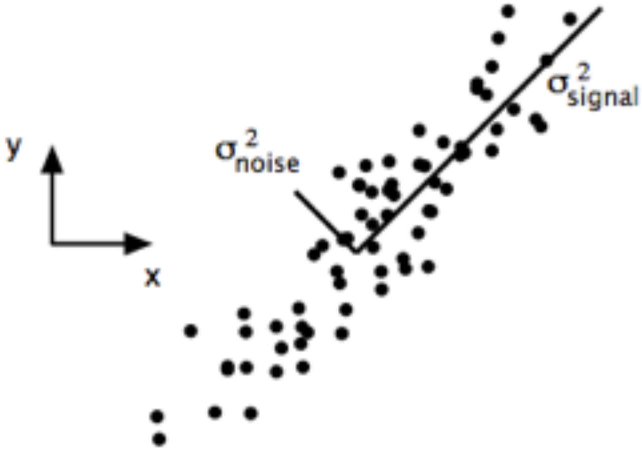


FIG. 2 Simulated data of (x, y) for camera A. The signal and noise variances σ^2_{signal} and σ^2_{noise} are graphically represented by the two lines subtending the cloud of data. Note that the largest direction of variance does not lie along the basis of the recording (x_A, y_A) but rather along the best-fit line.

$$SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}}.$$

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

I. RECAP: PCA

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

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

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

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

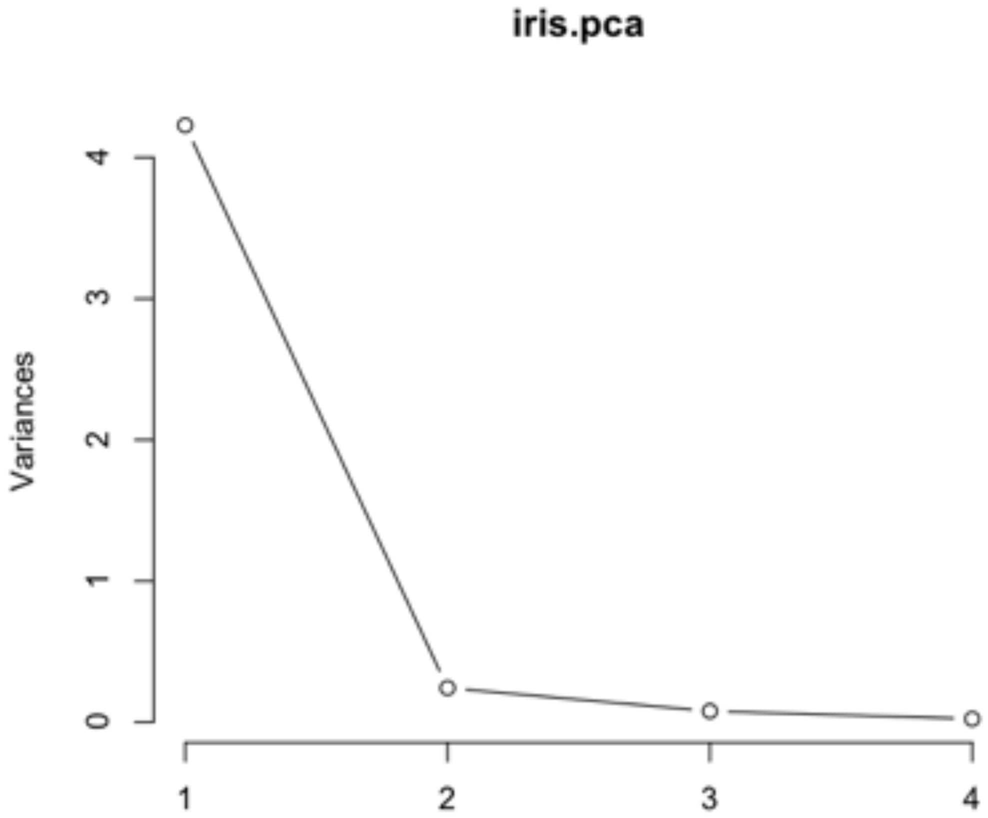
$$A = Q\Lambda Q^{-1}$$

The columns of Q are the eigenvectors of A, and the
are the associated eigenvalues of A.

NOTE
This relationship *defines* what it means to be an eigenvector of *A*.

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

$$Av = \lambda v$$



NOTE

Looking at this plot also gives you an idea of how many principal components to keep.

Apply the *elbow test*: keep only those pc's that appear to the left of the elbow in the graph.

1. **Linearity** – The change in basis is a linear projection

2. **Large variances have important structure** – e.g. large signal-to-noise ratio. In other words, we assume that principal components with larger associated variances are signal, while those with lower variances represent noise. NOTE: this is a strong (and not always correct) assumption!

3. **The principal components are orthogonal** – A simplification that makes PCA soluble with linear algebra matrix decomposition techniques

II. SINGULAR VALUE DECOMPOSITION

Notice: Lots of math / linear algebra notation ahead!

It's okay if it does not all immediately make sense.

Take a deep breath…

Notice: Lots of math / linear algebra notation ahead!

It's okay if it does not all immediately make sense.

Take a deep breath…

That's better! Okay, then…

Consider a matrix A with n rows and d features.

Consider a matrix A with n rows and d features.

The singular value decomposition of A is given by:

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

Consider a matrix A with n rows and d features.

The singular value decomposition of A is given by:

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

Consider a matrix A with n rows and d features.

The singular value decomposition of A is given by:

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

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

Consider a matrix A with n rows and d features.

The singular value decomposition of A is given by:

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

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

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

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

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

The columns of *U* & *V* are the (left- and right-) singular vectors of *A*.

The singular value decomposition of A is given by:

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

$$\underset{(n \times d)}{A} \qquad \underset{(n \times n)}{U} \quad \underset{(n \times d)}{\Sigma} \quad \underset{(d \times d)}{V^T}$$

The columns of U & V are the (left- and right-) singular vectors of A.

These singular vectors provide orthonormal bases for the spaces K_n & K_d (columns of U & V, respectively).

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

$$\underset{(n \times d)}{A} = \underset{(n \times n)}{U} \; \underset{(n \times d)}{\Sigma} \; \underset{(d \times d)}{V^T}$$

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

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.

Q: How do you interpret the SVD?

Q: How do you interpret the SVD?

A: Recall that given a set of n points in d-dimensional space (e.g., a matrix A), we want to find the best $k < d$ dimensional subspace to represent the data.

Q: How do you interpret the SVD?

A: Recall that given a set of n points in d-dimensional space (eg, a matrix A), we want to find the best $k < d$ dimensional subspace to represent the data.

For $k = 1$, this subspace is a line passing through the origin.

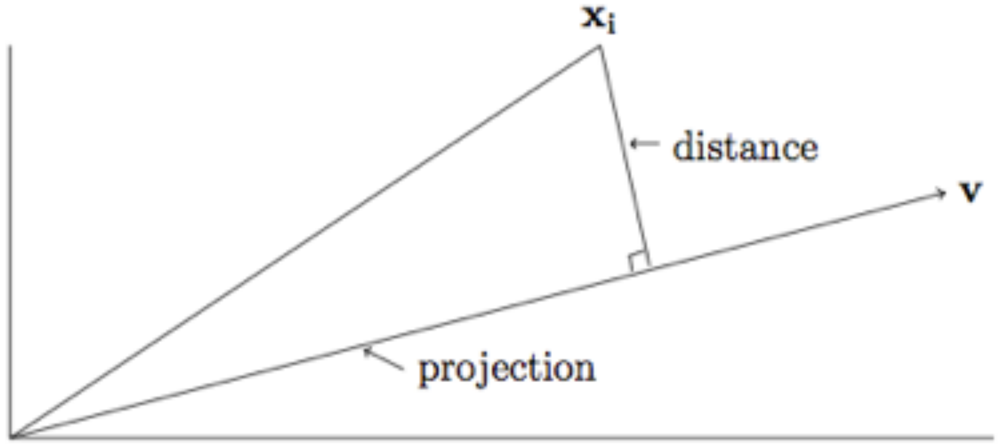


Figure 4.1: The projection of the point $\mathbf{x_i}$ onto the line through the origin in the direction of $\mathbf{v}$

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

For a geometric interpretation of the singular values, consider a unit sphere in R_n and a linear map T (eg, a rotation and a stretch) that sends this sphere to an ellipsoid in R_d.

For a geometric interpretation of the singular values, consider a unit sphere in R_n and a linear map T (eg, a rotation and a stretch) that sends this sphere to an ellipsoid in R_d.

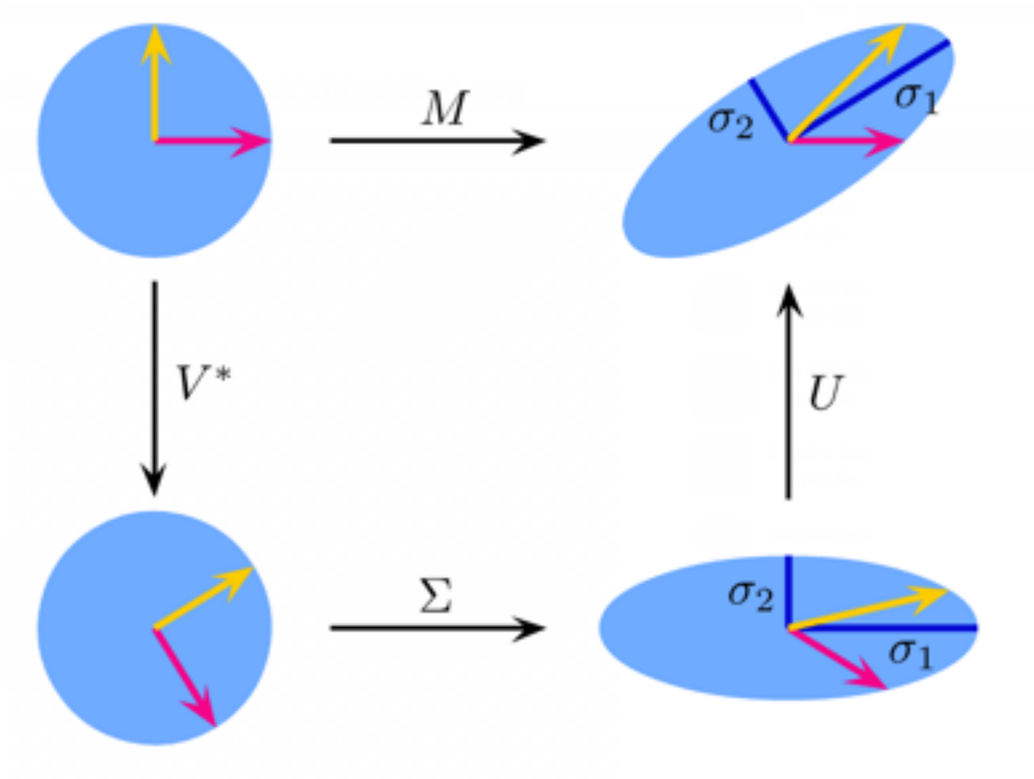
The singular vectors of T correspond to the lengths of the axes of the d-dimensional ellipsoid.

For a geometric interpretation of the singular values, consider a unit sphere in R_n and a linear map T (eg, a rotation and a stretch) that sends this sphere to an ellipsoid in R_d.

The singular vectors of T correspond to the lengths of the axes of the d-dimensional ellipsoid.

The singular values give the magnitudes of the projection of each column of the original dataset on the elements of the new basis.

SINGULAR VALUE DECOMPOSITION



$$M = U \cdot \Sigma \cdot V^*$$

source: http://en.wikipedia.org/wiki/Singular_value_decomposition

III. OTHER METHODS

Whereas PCA and SVD create new coordinates by transform the old coordinates, factor analysis requires new coordinates to be specified externally.

Whereas PCA and SVD create new coordinates by transform the old coordinates, factor analysis requires new coordinates to be specified externally.

These new coordinates are associated with *hidden* or *latent* features that we think our data depends on.

Whereas PCA and SVD create new coordinates by transform the old coordinates, factor analysis requires new coordinates to be specified externally.

These new coordinates are associated with *hidden* or *latent* features that we think our data depends on.

The old coordinates are then modeled as linear combinations of the latent features.

For example, consider a dataset that represents the results of a decathalon (rows = participants, columns = events, entries = times).

For example, consider a dataset that represents the results of a decathalon (rows = participants, columns = events, entries = times).

Though this dataset contains 10 features X_i, we may be interested in modeling these features as functions of *latent variables* such as the speed and strength of the participants:

$$X_i = \lambda_1 f_1 + \lambda_2 f_2 + \varepsilon$$

For example, consider a dataset that represents the results of a decathalon (rows = participants, columns = events, entries = times).

Though this dataset contains 10 features X_i, we may be interested in modeling these features as functions of *latent variables* such as the speed and strength of the participants:

$$X_i = \lambda_1 f_1 + \lambda_2 f_2 + \varepsilon$$

This would allow us to analyze the data in a more fundamental way.

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

However, sometimes linear techniques are not sufficient.

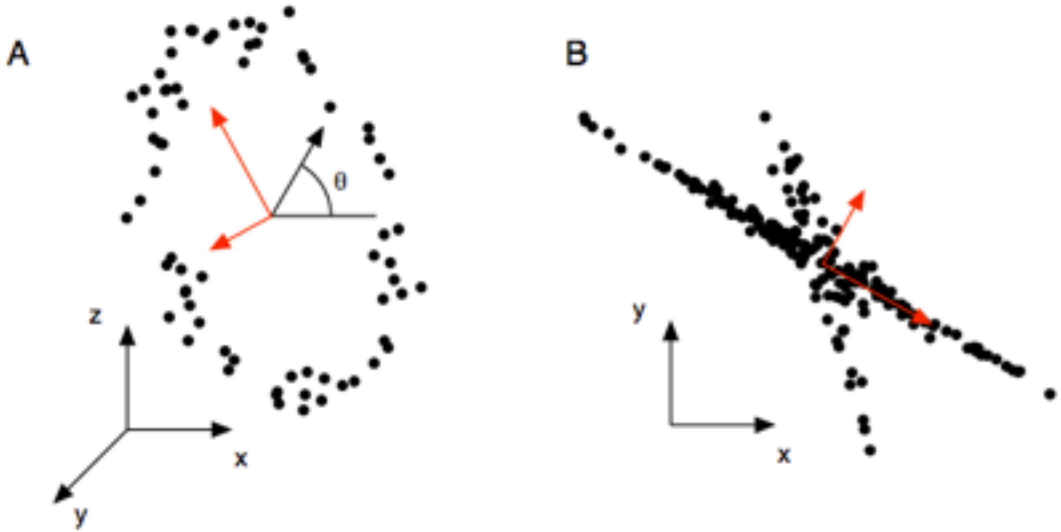


FIG. 6 Example of when PCA fails (red lines). (a) Tracking a person on a ferris wheel (black dots). All dynamics can be described by the phase of the wheel θ, a non-linear combination of the naive basis. (b) In this example data set, non-Gaussian distributed data and non-orthogonal axes causes PCA to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

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

In any case, the key difficulties with dimensionality reduction are time/space complexity, randomness (eg different results for different runs), and selecting the number of dimensions in the lower-dim subspace.

In any case, the key difficulties with dimensionality reduction are time/space complexity, randomness (eg different results for different runs), and selecting the number of dimensions in the lower-dim subspace.

Furthermore, there's an obvious (bias/variance) tradeoff between the number of subspace dimensions and the size of approximation error.

IV. LAB

V. EXERCISE IN PAIRS

http://scikit-learn.org/stable/auto_examples/decomposition/plot_faces_decomposition.html