Data Preparation: *Reduction and Transformation*

Mining Massive Datasets

Materials provided by Prof. Carlos Castillo — https://chato.cl/teach

Instructor: Dr. Teodora Sandra Buda — https://tbuda.github.io/

Universitat Pompeu Fabra Barcelona

Main Sources

- Data Mining, The Textbook (2015) by Charu Aggarwal (Chapter 2) + slides by Lijun Zhang
- Introduction to Data Mining 2nd edition (2019) by Tan et al. (Chapter 2)
- Data Mining Concepts and Techniques, 3rd edition (2011) by Han et al. (Chapter 3)

Data reduction and transformation

- Sampling
  - ≃ “Less rows”
- Dimensionality Reduction or Feature Selection
  - ≃ “Less columns”

Why reduce/transform data?

- Advantages
  - Reduce space complexity
  - Reduce time complexity
  - Reduce noise
  - Reveal hidden structures (e.g., manifold learning)
- Disadvantages
  - Information loss

Sampling for static data

- **Uniform** random sampling
  - with/without replacement
- **Biased** sampling
  - e.g., emphasize recent items
- **Stratified** sampling
  - Partition data in strata, sample in each stratum

Sampling example

- There are 10000 people which contain 100 millionaires
- Uniform random sample of 100 people
  - In expectation, one millionaire will be sampled
  - There is ≃ 37% chance no millionaires are sampled, why?
- Stratified Sampling
  - Unbiased Sampling 1 from 100 millionaires
  - Unbiased Sampling 99 from remaining

Sampling from data streams

- Suppose you want to give away for free 10 VIP passes at a concert
  - You want everybody to have exactly the same chance of getting the VIP pass, independently on when they arrived, as long they arrive before the concert starts
  - Once people leave the entrance area they become impossible to find, so if you win, you should receive the VIP pass at the door
  - People arrive in sequence, and you do not know how many people will arrive
- **Reservoir sampling algorithm** … seen in the stream processing part

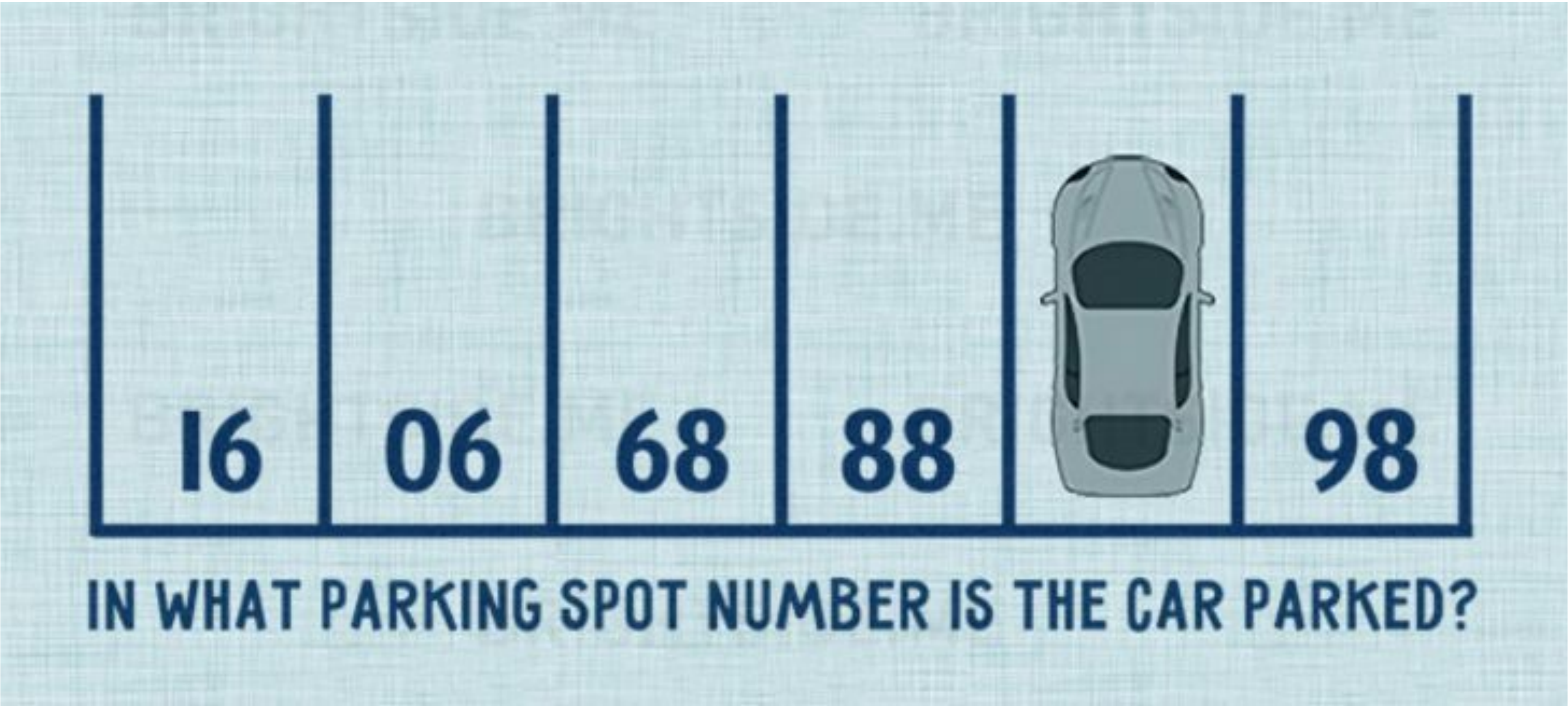
Reducing data dimensionality

Note: PCA/SVD covered well in other courses, won’t be part of our exam

Feature selection

- **Unsupervised** Feature Selection
  - Using the performance of unsupervised learning (e.g, clustering) to guide the selection
- **Supervised** Feature Selection
  - Using the performance of supervised learning (e.g., classification) to guide the selection

An axis rotation may help :-)



Source: [Centauro Blog (2017)]

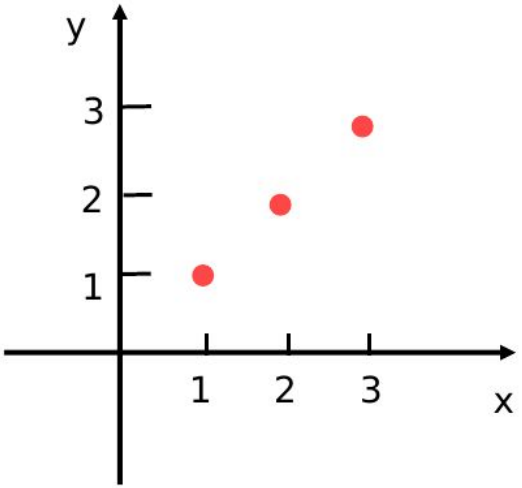
Dimensionality reduction with axis rotation (perfect case)

- Motivation: three points in a line in two-dimensional space

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\mathbf{x}_2 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

$$\mathbf{x}_3 = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$



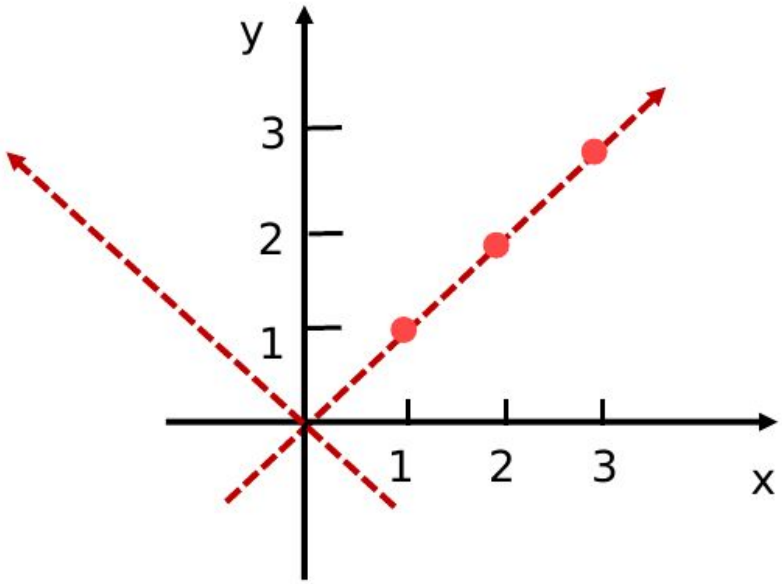
Dimensionality reduction with axis rotation (perfect case, cont.)

- Coordinates after axes rotation

$$\mathbf{x}_1 = \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix}$$

$$\mathbf{x}_2 = \begin{bmatrix} 2\sqrt{2} \\ 0 \end{bmatrix}$$

$$\mathbf{x}_3 = \begin{bmatrix} 3\sqrt{2} \\ 0 \end{bmatrix}$$



Dimensionality reduction with axis rotation (perfect case, cont.)

- Coordinates after axes rotation

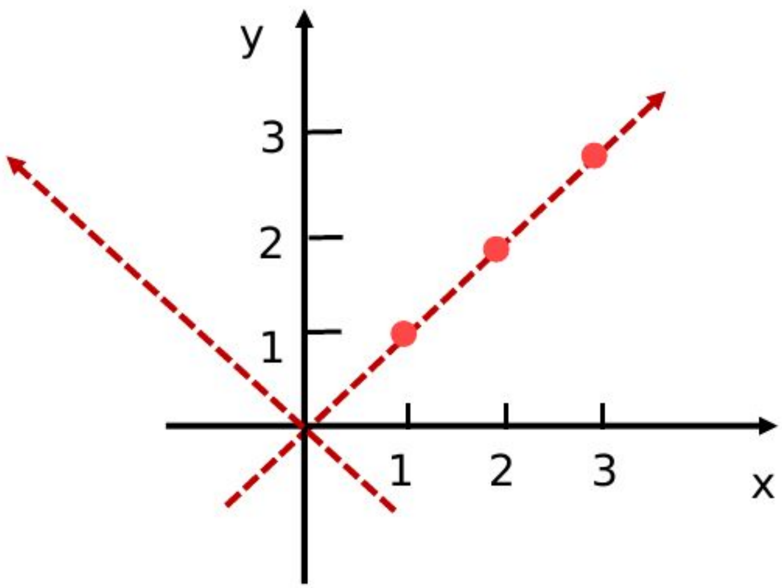
Drop second coordinate, **no** information is lost.

2D data reduced to 1D data

$$\mathbf{x}_1 = \begin{bmatrix} \sqrt{2} \\ \cancel{0} \end{bmatrix}$$

$$\mathbf{x}_2 = \begin{bmatrix} 2\sqrt{2} \\ \cancel{0} \end{bmatrix}$$

$$\mathbf{x}_3 = \begin{bmatrix} 3\sqrt{2} \\ \cancel{0} \end{bmatrix}$$



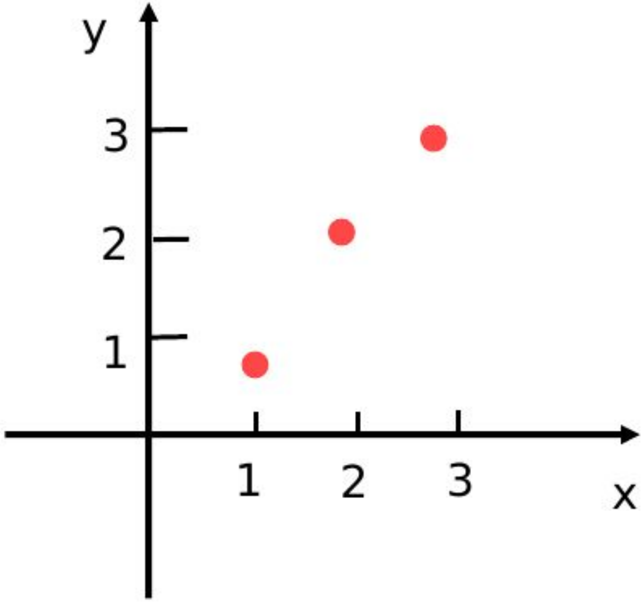
Dimensionality reduction with axis rotation (noisy case)

- Suppose points don't lie exactly on a line

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 0.9 \end{bmatrix}$$

$$\mathbf{x}_2 = \begin{bmatrix} 2.1 \\ 2 \end{bmatrix}$$

$$\mathbf{x}_3 = \begin{bmatrix} 2.9 \\ 3.1 \end{bmatrix}$$



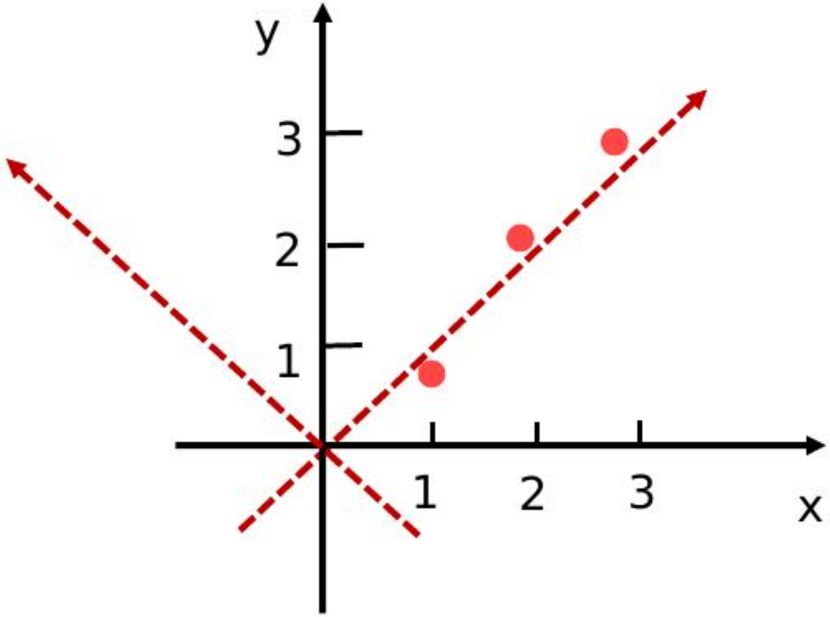
Dimensionality reduction with axis rotation (noisy case, cont.)

- Suppose points don’t lie exactly on a line

$$\mathbf{x}_1 = \begin{bmatrix} 1.34 \\ 0.07 \end{bmatrix}$$

$$\mathbf{x}_2 = \begin{bmatrix} 2.89 \\ 0.07 \end{bmatrix}$$

$$\mathbf{x}_3 = \begin{bmatrix} 4.24 \\ -0.14 \end{bmatrix}$$



Dimensionality reduction with axis rotation (noisy case, cont.)

- Suppose points don't lie exactly on a line

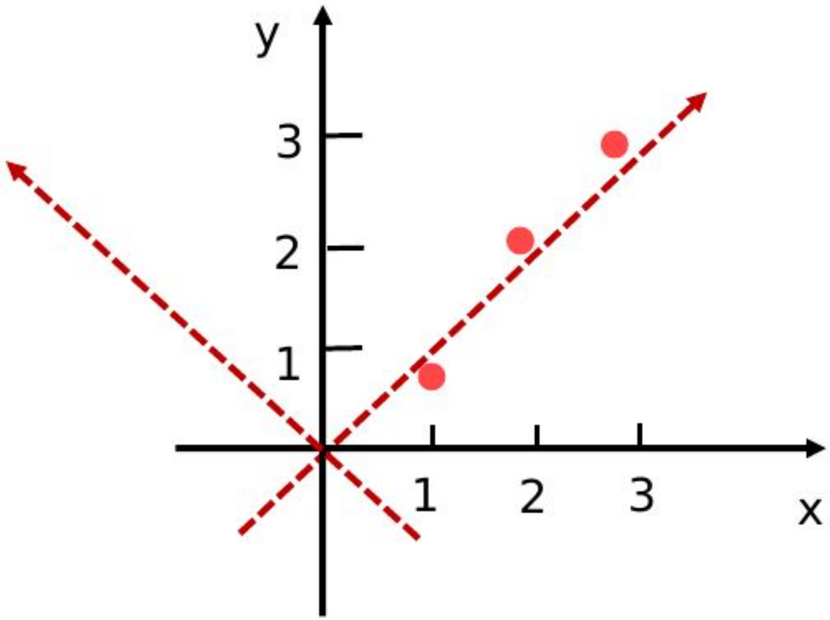
Drop second coordinate, **some** information is lost.

2D data reduced to 1D data

$$\mathbf{x}_1 = \begin{bmatrix} 1.34 \\ \cancel{0.07} \end{bmatrix}$$

$$\mathbf{x}_2 = \begin{bmatrix} 2.89 \\ \cancel{0.07} \end{bmatrix}$$

$$\mathbf{x}_3 = \begin{bmatrix} 4.24 \\ \cancel{-0.14} \end{bmatrix}$$



How does this work in reality?

- Change of axes removes correlations and reduces dimensionality
- Techniques
  - Principal Component Analysis (PCA)
  - Singular-Value Decomposition (SVD)
  -

Summary

Things to remember

- Data sampling methods
- Why would we want to reduce dimensionality?
- What are the main techniques for doing so

Exercises for TT03-TT05

- Exercises 3.7 of Data Mining Concepts and Techniques, 3rd edition (2011) by Han et al.
- Exercises 2.6 of Introduction to Data Mining, Second Edition (2019) by Tan et al.
  - Mostly the first exercises, say 1-6

Additional contents
(not included in exams)

Axis rotation - formulation

- Points are usually described with respect to the standard basis

$$\mathbf{x} = \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^d \end{bmatrix} \in \mathbb{R}^d \Longleftrightarrow \mathbf{x} = x^1\mathbf{e}_1 + x^2\mathbf{e}_2 + \cdots + x^d\mathbf{e}_d$$

Axis rotation – formulation (cont.)

We will determine **new coordinates** under basis W:

$W = [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_d]$ is a orthonormal matrix

$$\mathbf{x} = WW^\top \mathbf{x} = \left(\sum_{i=1}^{d} \mathbf{w}_i \mathbf{w}_i^\top \right) \mathbf{x} = \sum_{i=1}^{d} \mathbf{w}_i (\mathbf{w}_i^\top \mathbf{x})$$

$$= (\mathbf{w}_1^\top \mathbf{x}) \mathbf{w}_1 + (\mathbf{w}_2^\top \mathbf{x}) \mathbf{w}_2 + \cdots + (\mathbf{w}_d^\top \mathbf{x}) \mathbf{w}_d$$

Thus, the new coordinates are

$$\mathbf{y} = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x} \\ \mathbf{w}_2^\top \mathbf{x} \\ \vdots \\ \mathbf{w}_d^\top \mathbf{x} \end{bmatrix} \in \mathbb{R}^d$$

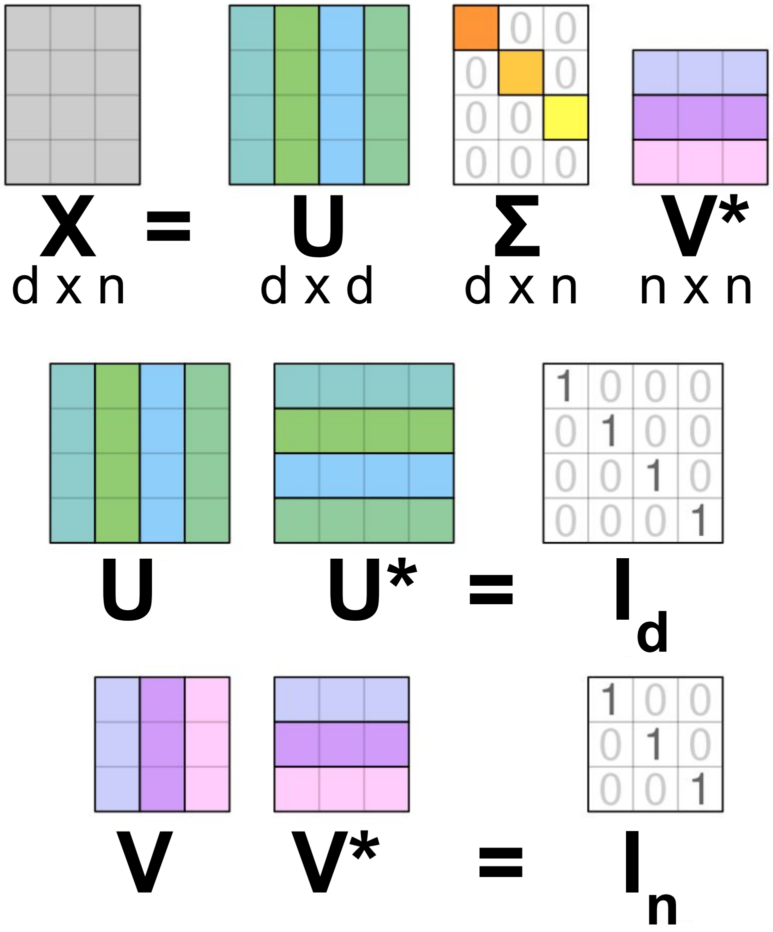
Vector *x* has *n* dimensions, but vector *y* has *d*≤*n* dimensions

PCA formulation: optimization

- Find new basis { w_1, w_2, …, w_k }, with k ≤ d such that the variance of this set is maximized:

$$\left\{ \mathbf{y}_1 = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x}_1 \\ \mathbf{w}_2^\top \mathbf{x}_1 \\ \vdots \\ \mathbf{w}_k^\top \mathbf{x}_1 \end{bmatrix}, \mathbf{y}_2 = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x}_2 \\ \mathbf{w}_2^\top \mathbf{x}_2 \\ \vdots \\ \mathbf{w}_k^\top \mathbf{x}_2 \end{bmatrix}, \cdots, \mathbf{y}_n = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x}_n \\ \mathbf{w}_2^\top \mathbf{x}_n \\ \vdots \\ \mathbf{w}_k^\top \mathbf{x}_n \end{bmatrix} \right\}$$

SVD formulation



- U and V are rotation matrices; Σ is a scaling matrix
- The rotated data is obtained by multiplying $U^T X$

Algorithms for PCA and SVD

- PCA
  1. Calculate the mean vector $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$
  2. Calculate the covariance matrix $C = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top$
  3. Calculate the k-largest eigenvectors of C

- SVD
  1. Calculate the mean vector $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$
  2. Calculate the k largest left singular vectors of $\bar{X} = [\mathbf{x}_1 - \bar{\mathbf{x}}, \dots, \mathbf{x}_n - \bar{\mathbf{x}}]$