CS456: Machine Learning

Unsupervised Dimensionality Reduction

Jakramate Bootkrajang

Department of Computer Science
Chiang Mai University

March 5, 2020

Objectives

- To understand issues found in high-dimensional space
- To understand what principal component in PCA is
- Be able to apply PCA for realworld problem

Outlines

- Curse of Dimensionality
- Principal Component Analysis (PCA)

High dimensional data

- The working of machine learning algorithms depends in some way on the geometry of data
  - lengths of vectors, distances, angles
- High dimensional geometry is different from low dimensional geometry

What happens in HD? [1/2]

Concentration of norms: generate points in m-dimensional space and measure their lengths

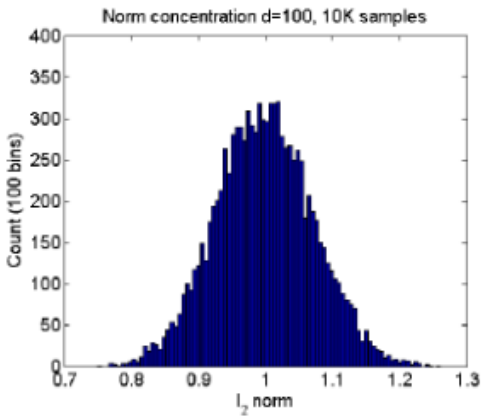


Figure: $d = 100$ norm concentration

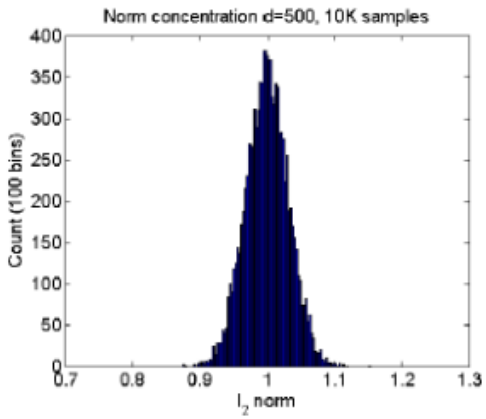


Figure: $d = 500$ norm concentration

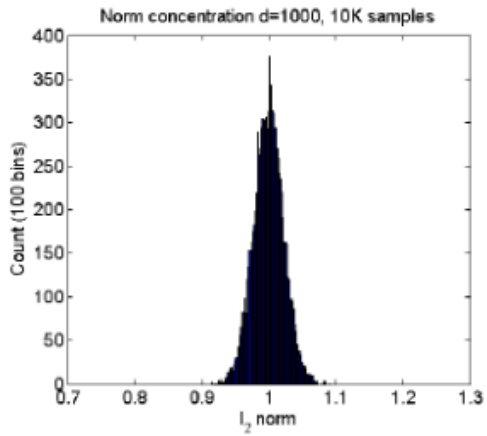


Figure: $d = 1000$ norm concentration

Figure: Credit: A.Kaban, CS-Bham

What happens in HD? [2/2]

Near-orthogonality: generate points in m-dimensional space and measure their angles

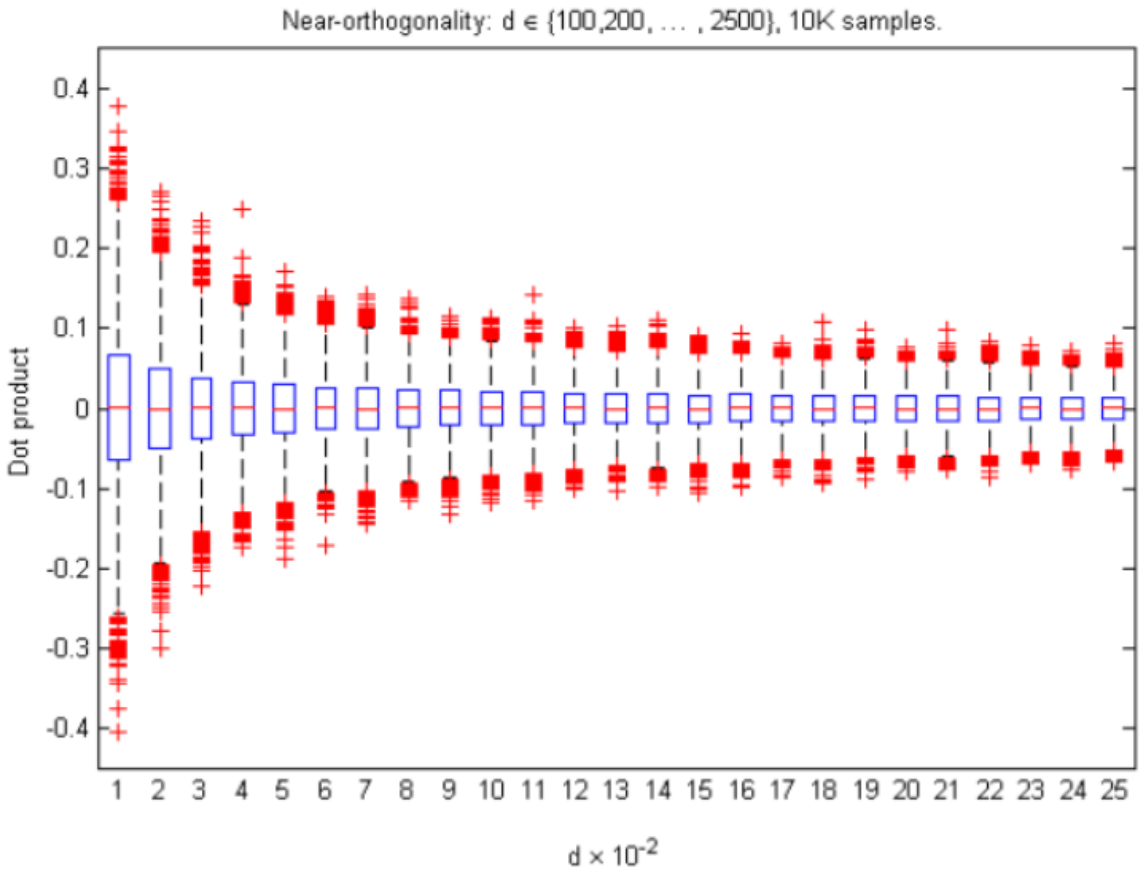


Figure: Credit: A.Kaban, CS-Bham

Curse of Dimensionality

- We can see from the plots that
  - As m increases, any two random vectors end up being orthogonal to each other
  - As m increases, any random vectors ends up having about the same length
- We also need a lot (exponentially) more data to cover the space as m increases
- Training time is also increased significantly as m grows

Bless of dimensionality

- Surprisingly, high dimensionality makes data more linearly separable
- Think of kernel method, or feature learnt by convolutional neural networks
- It is easier for algorithm to find separating hyperplane
  - providing that there's no noise in data

Dimensionality reduction approaches

- Feature selection
  - Find subset of features
- Feature projection (feature extraction)
  - learn a function $\phi(\cdot)$ which transforms data from HD to lower dimensional space
  - In general, we aim at minimising reconstruction error

$$\epsilon_{recon} = ||X - \phi^{-1}(\phi(X))||^2$$

Principal Component Analysis (PCA)

Introduction

- An unsupervised algorithm for dimensionality reduction
- Reduce dimensionality of the data while trying to preserve data structure

Intuition

Find low-dimensional projection with largest spread

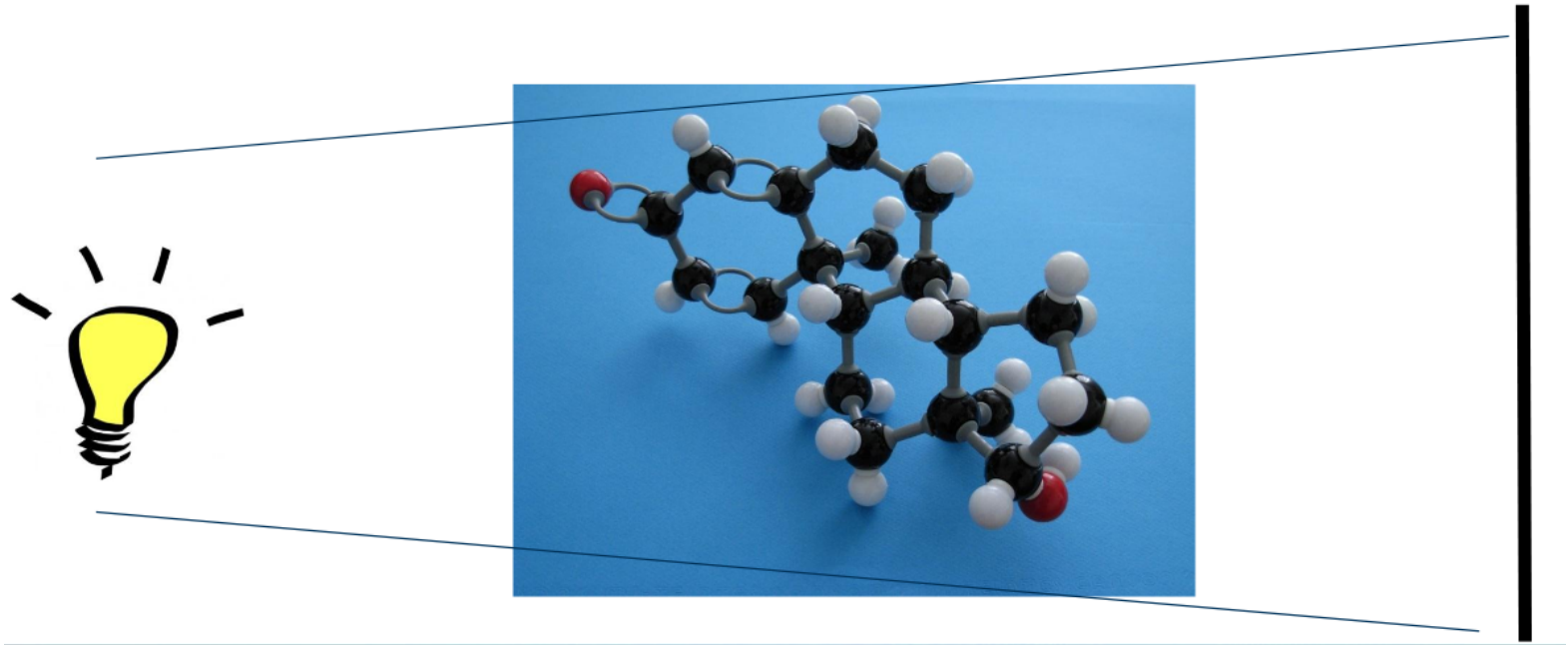


Figure: Applied Multivariate Statistics: ETZ

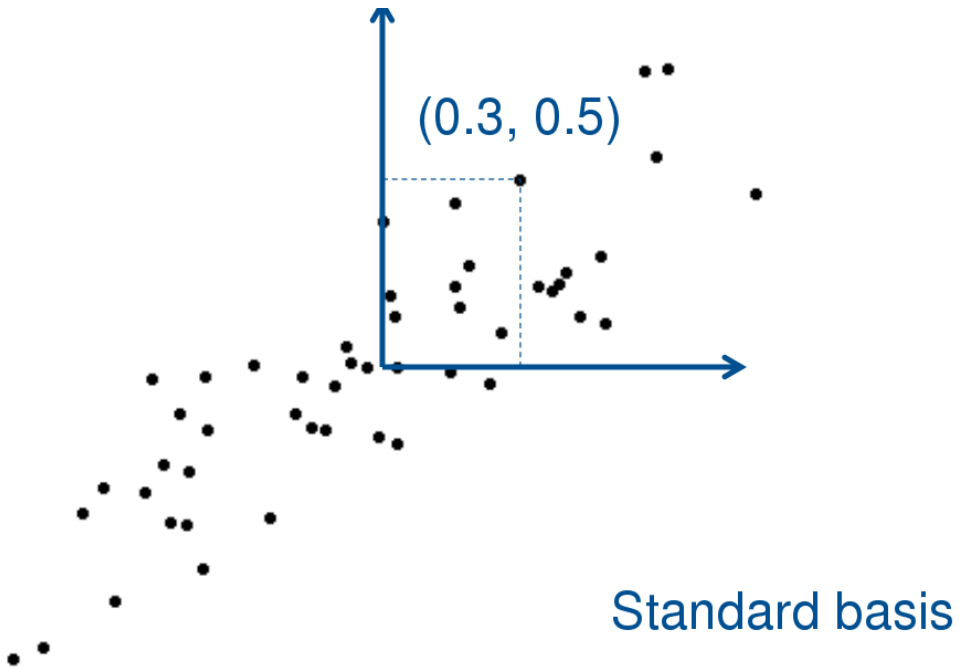
Toy data

Assuming data is in 2D space

Figure: Applied Multivariate Statistics: ETZ

Standard Basis

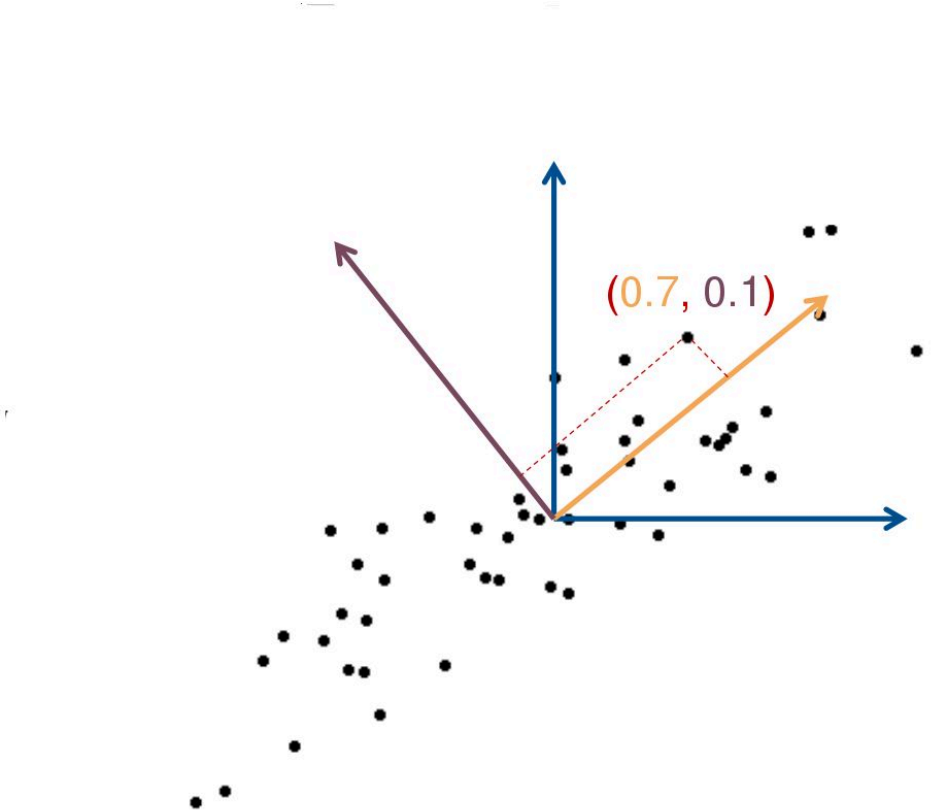
- Normally, data lives In the standard basis defined by two basis vectors $\{[0 \quad 1], [1 \quad 0]\}$ [1]



[1] A basis is a set of linearly independent vectors that can represent any vector in a given vector space

Other basis systems

- Imagine using another valid basis system



- Observe: data is more spread along one of the new basis directions (orange vector)

Ideas

- Find new basis system that maximises variance in all directions
- These directions are known as Principal Components (PC)
- Hopefully we can select a few of these PCs and project the data onto this new basis

Projection onto one direction

- Projection of a vector $\mathbf{x} = \begin{bmatrix} x^1 \\ x^2 \\ \dots \\ x^m \end{bmatrix}$ on to a vector $\mathbf{a} = \begin{bmatrix} a^1 \\ a^2 \\ \dots \\ a^m \end{bmatrix}$ is the linear combination

$$\mathbf{a}^T\mathbf{x} = \sum_{i=1}^{m} a^i x^i$$

- Usually, $\mathbf{a}$ is kept as a unit vector

Projection onto multiple directions

- To generalise one direction projection, the projection of $\mathbf{x}$ onto a set of linearly independent vectors (*basis*) A is

$$A^T\mathbf{x} = \begin{bmatrix} a_1^1 & a_2^1 & a_3^1 \\ a_1^2 & a_2^2 & a_3^2 \\ \vdots & \vdots & \vdots \\ a_1^m & a_2^m & a_3^m \end{bmatrix}^T \begin{bmatrix} x^1 \\ \vdots \\ x^m \end{bmatrix} = \begin{bmatrix} p^1 \\ p^2 \\ p^3 \end{bmatrix}$$

- $\mathbf{p}$ is a vector in 3D space , compared to the original $\mathbf{x}$ in mD space

Data spread after a projection

- Assumed data matrix X and its projection $\mathbf{a}^T X$ are mean-centred
- The spread of the projection is

$$\begin{aligned}\sigma_{\mathbf{a}}^2 &= (\mathbf{a}^T X)(\mathbf{a}^T X)^T \\ &= \mathbf{a}^T X X^T \mathbf{a} \\ &= \mathbf{a}^T V \mathbf{a}\end{aligned}$$

- We see that spread is a function of projection direction $\mathbf{a}$ and $m \times m$ covariance matrix V

Objective function of projection direction

- Maximising $\mathbf{a}^T V\mathbf{a}$ makes no sense, because we can increase the spread by multiplying $\mathbf{a}$ by some large number
- We have to impose size constraint on $\mathbf{a}$, e.g., $\mathbf{a}^T\mathbf{a} = 1$
- We then arrive at an objective function

$$u = \mathbf{a}^T V\mathbf{a} - \lambda(\mathbf{a}^T\mathbf{a} - 1)$$

- $\lambda > 0$ is a parameter imposing the size constraint [2]
  - Think of λ like the C parameter in SVM

[2] Its name is Lagrange multiplier often employed in constrained optimisation

Maximising the spread

- Objective function is convex, calculus helps finding stationary point
- Differentiating the objective function w.r.t **a** and equating to zero

$$\frac{\partial u}{\partial \mathbf{a}} = 2V\mathbf{a} - 2\lambda\mathbf{a} = 0 \tag{1}$$

$$V\mathbf{a} = \lambda\mathbf{a} \tag{2}$$

- Eq.(2) is one type of Linear system of equations called the Characteristic Equations
- If we can solve the system for **a** we will have maximum spread direction

Characteristic Equations

$$V\mathbf{a} = \lambda\mathbf{a}$$

- For an $m \times m$, real and symmetric matrix V, there are m possible solution vectors
- For symmetric matrices, eigenvectors for distinct eigenvalues are orthogonal
- Each of the solutions $\mathbf{a}_i$ is known as eigenvector of V
- Each eigenvector is associated with an eigenvalue λ_i

Eigenvector problem refresher

Eigenvector and Eigenvalue

Definition

A nonzero vector $\mathbf{x}$ is an eigenvector of a square matrix A if there exists a scalar λ such that

$$A\mathbf{x} = \lambda\mathbf{x}$$

- λ is an eigenvalue of A associated with eigenvector $\mathbf{x}$
- The zero vector can not be an eigenvector even though

$$A0 = \lambda 0$$

- But $\lambda = 0$ can be an eigenvalue

Example, solving for eigenvectors

- Given $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$, find its eigenvectors and eigenvalues
- Solution:
  1. from definition $Ax = \lambda x$
  2. $(A - \lambda I)x = 0$
  3. Since x is nonzero, we know that $(A - \lambda I)$ is not invertible
  4. So determinant of $(A - \lambda I)$ must be zero
  5. $|A - \lambda I| = 0$

Summarising

- The m eigenvectors form to a new basis system
- The eigenvector **a** with largest eigenvalue is the projection direction with maximum spread (most important)
- PCA selects k most important from m eigenvectors where $k < m$
- PCA projects dataset X onto the new basis formed by k eigenvectors
- $$X_r = \begin{bmatrix} a_1^1 & a_2^1 & \dots & a_k^1 \\ a_1^2 & a_2^2 & \dots & a_k^2 \\ \vdots & \vdots & \vdots & \vdots \\ a_1^m & a_2^m & \dots & a_k^m \end{bmatrix}^T X$$

Practical usage

- Standardising X
- Calculate a covariance matrix $V = XX^T$
- Find all the eigenvectors of V
- Select k most important principal components according to eigenvalues and put it in a matrix A
- Project X onto A by calculating $X_r = A^T X$
- The reducted data is in X_r

How to choose k?

- The structure of data can be defined as sum of spreads in all direction
- From $V\mathbf{a}_i = \lambda_i \mathbf{a}_i$, we see that λ_i quantifies the spread of data after projecting on $\mathbf{a}_i$
- The loss in structure information by selecting only k PCs is

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

- Usually, we stop throwing PCs away when the loss exceeds the predefined threshold

Computational Issue

- Computing V takes $O(nm^2)$
- PCA complexity is then $O(nm^2)$ + Complexity of solving eigenvector
- PCA can be applied to large dataset (scale well with n) but it does not scale well with dimensionality m
  - Slow for high-dimensional data

Objectives: revisited

- To understand issues found in high-dimensional space
- To understand what principal component in PCA is
- Be able to apply PCA for realworld problem