COMP 7180
Quantitative Methods for Data Analytics and Artificial Intelligence

Lecture 4: Dimensionality Reduction (Feature Extraction) – Part I

Dimensionality

- An object can be described by a set of characters
- Mathematically, an object can be defined as one point in the vector space
  - Each dimension of the vector space is used to describe one character of the object
  - Example: a pixel in an image/video

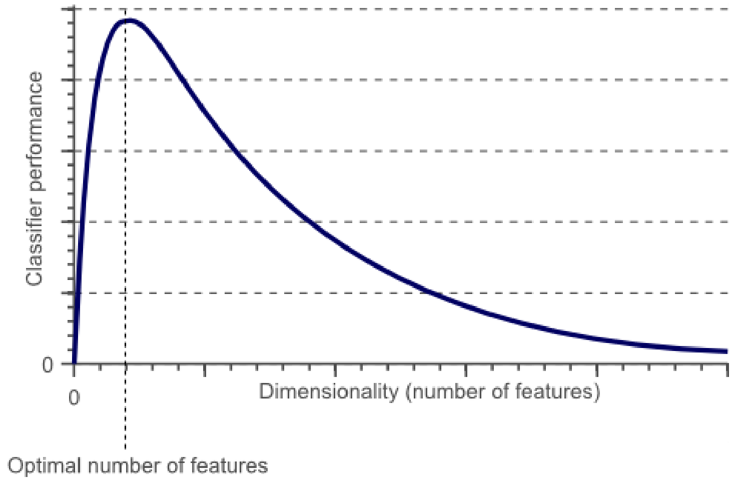
How High the dimensionality could be?

- A small gray image with the resolution 100×100 is represented as a 10,000-dimensional vector in the pixel space

- The movie “Kung Fu Panda 3”: consider each pixel value as a dimension, the total dimension of this data will be 1280×720×25×60×120×3 = **500,000,000,000 !!!**

Curse of Dimensionality

- From a theoretical point of view, increasing the number of features should lead to better performance. However …

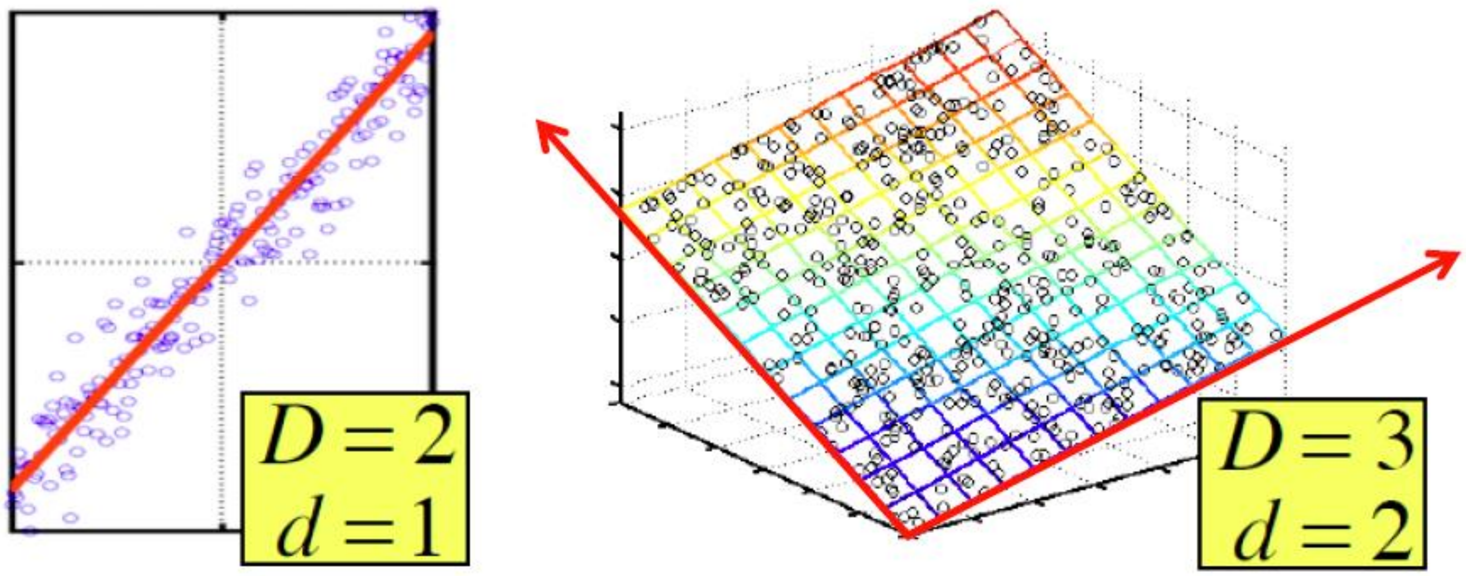


- In practice, the inclusion of more features leads to worse performance (i.e., curse of dimensionality)
  – High computational cost
  – Redundant information

Dimensionality Reduction

- Motivation
  - Overcome the curse of dimensionality
  - The intrinsic dimension may be small
  - Visualization: projection of high-dimensional data onto 2D or 3D
  - Data compression: efficient storage and retrieval
  - Noise removal: positive effect on query accuracy
- Definition
  - Generate a lower dimensional equivalence to the original high-dimensional feature space while capturing essentials of original data according to some criteria
- Applications
  - Face recognition, handwritten digit recognition, text summarization, image retrieval, movie editing, protein classification, …

Dimensionality Reduction



- **Assumption:** Data lies on or near a low *d*-dimensional subspace
- **Axes of this subspace are effective representation of the data**

Dimensionality Reduction

- **Compress / reduce dimensionality:**

day / customer	We 7/10/96	Th 7/11/96	Fr 7/12/96	Sa 7/13/96	Su 7/14/96
ABC Inc.	1	1	1	0	0
DEF Ltd.	2	2	2	0	0
GHI Inc.	1	1	1	0	0
KLM Co.	5	5	5	0	0
Smith	0	0	0	2	2
Johnson	0	0	0	3	3
Thompson	0	0	0	1	1

The above matrix is really "2-dimensional." All rows can be reconstructed by scaling [1 1 1 0 0] or [0 0 0 1 1]

Rank of a Matrix

- **Q:** What is **rank** of a matrix **A**?
- **A:** Number of **linearly independent** columns of **A**
- **For example:**
  - Matrix **A =** $\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$ has rank **r=2**

    ➢ **Why?** The first two rows are linearly independent, but all three rows are linearly dependent.
- **Why do we care about low rank?**
  - We can write **A** as two "basis" vectors: [1 2 1] [-2 -3 1]
  - And new coordinates of : [1 0] [0 1] [1 -1]

Mathematatic Definition of Dimensionality Reduction

- Given the high-dimensional data point

$$\mathbf{x} = (x_1, x_2, \cdots, x_D)^T$$

- Find a compact representation

$$\mathbf{y} = (y_1, y_2, \cdots, y_d)^T \qquad d \le D$$

- Construct the transformation function to capture essentials in the original

$$\Phi : \mathbf{x} \rightarrow \mathbf{y}$$

$$\rightarrow [32 \;\; 79 \;\; 54 \;\; \ldots \ldots]^T$$

Objectives of Dimensionality Reduction

- Generate a lower dimensional equivalence to the original high-dimensional feature space while capturing essentials of original data according to some criteria
- Information preserving (unsupervised)
  - We would like to retain as much information (data variance/distance) as possible
  - Principal component analysis (PCA)
- Classification (supervised)
  - We would like to maximize the separation among classes
  - Linear discriminant analysis (LDA)

Principal Component Analysis (PCA)

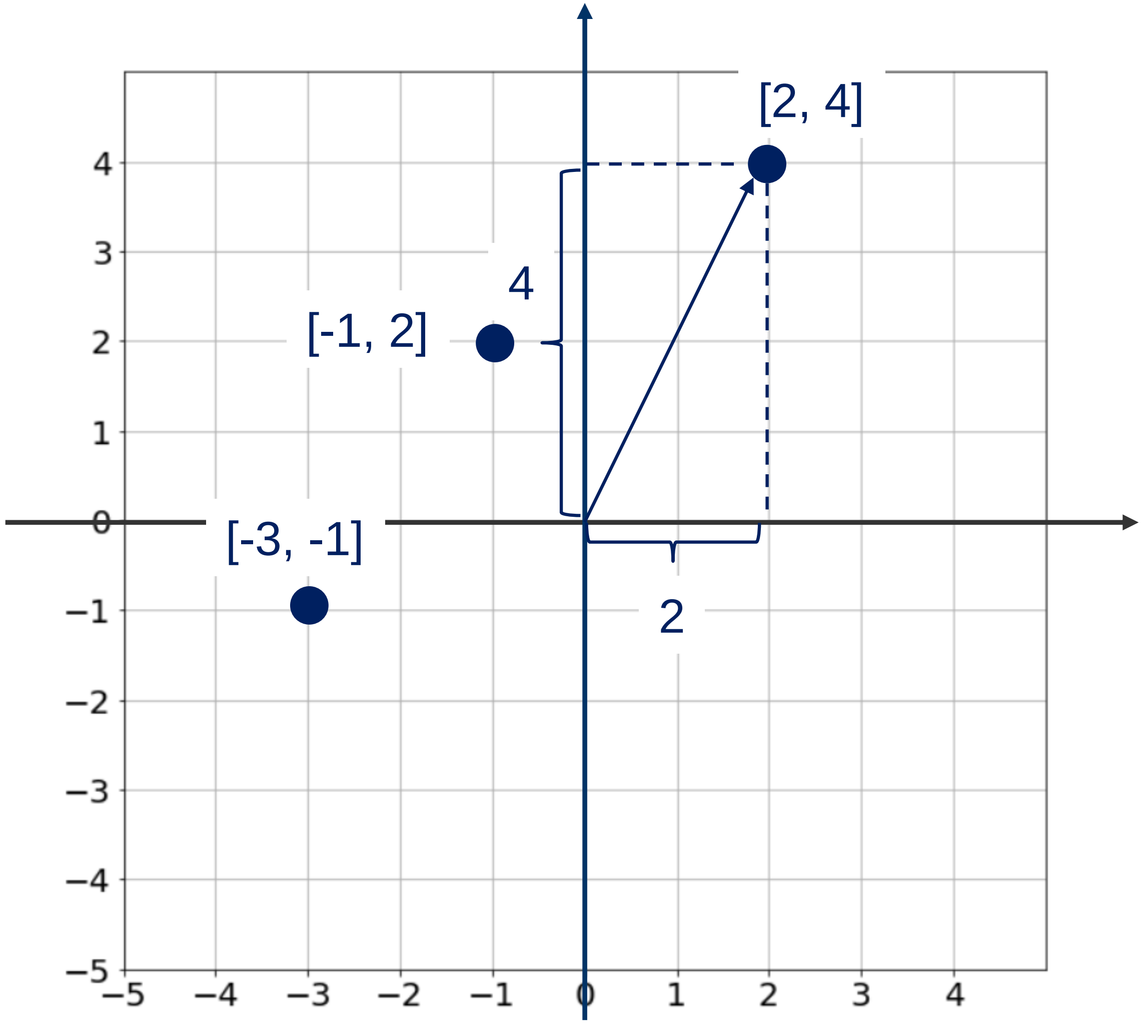
What is PCA

- Principal component analysis (PCA)
  - A classic linear dimensionality reduction method (Pearson, 1901; Hotelling, 1930)
  - Reduce the dimensionality of a data set by finding a new set of projection directions (coordinates), smaller than the original set of directions (coordinates)
  - Preserve most of the samples' information
    - Directions that capture maximum variance in data

Projection

- Vector projection
  - Dot/inner product of two vectors
  - $\mathbf{a} = [a_1, a_2]^T$, $\mathbf{b} = [b_1, b_2]$
  - $\mathbf{a}^\mathbf{T}\mathbf{b} = a_1 b_1 + a_2 b_2 = \|\mathbf{a}\|\|\mathbf{b}\| \cos\theta$

- Projection on “standard coordinate system”
  - Vector $[2, 4]^T$ projection on the x-axis is the dot production between [2, 4] and [1, 0]: 2*1 + 4*0 = 2
  - Vector $[2, 4]^T$ projection on the y-axis is the dot production between [2, 4] and [0, 1]: 2*0 + 4*1 = 4



Projection on other directions

- Project on the direction $\mathbf{u} = [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$
- Project $[2, 4]^T$ on direction **u**:

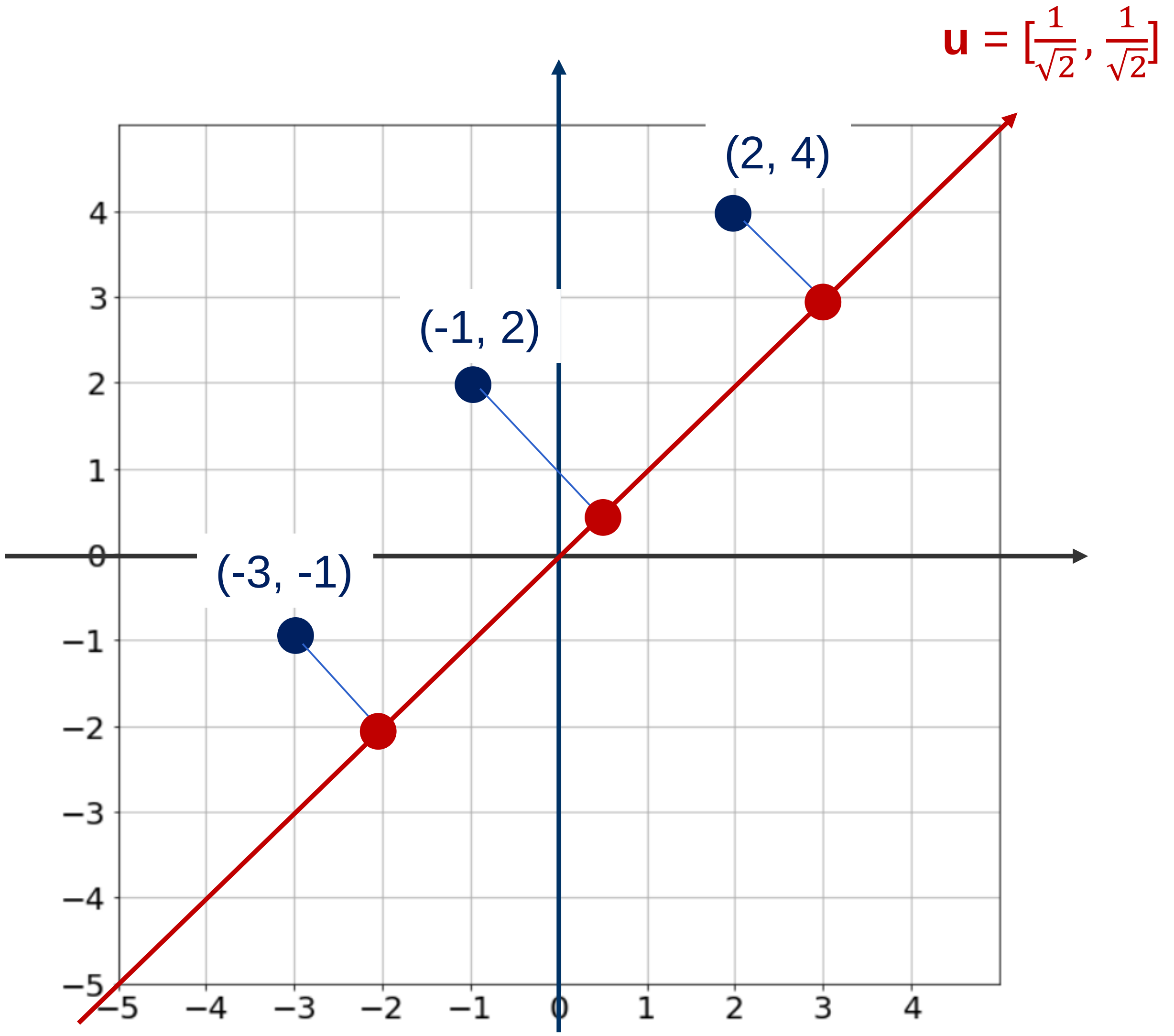
$$2\frac{1}{\sqrt{2}} + 4\frac{1}{\sqrt{2}} = \frac{6}{\sqrt{2}}$$

- Project $[-1, 2]^T$ on direction **u**:

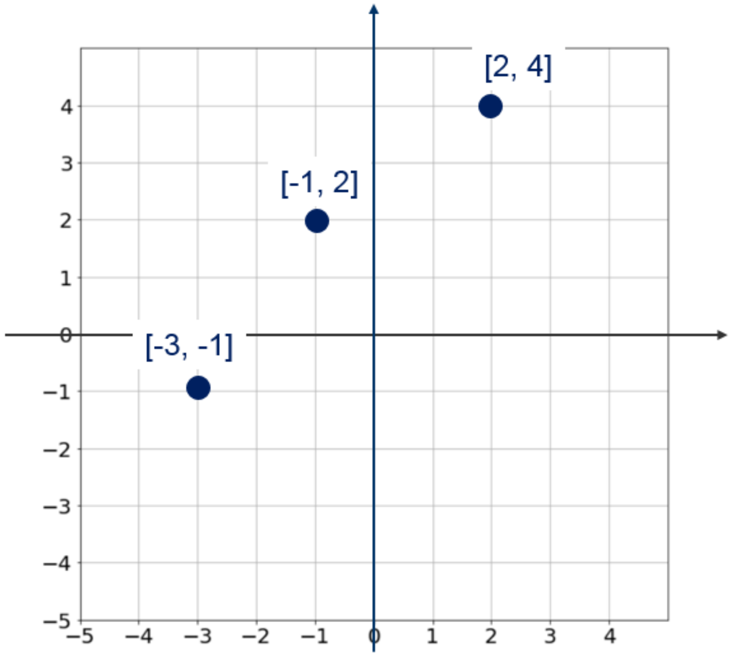
$$-1\frac{1}{\sqrt{2}} + 2\frac{1}{\sqrt{2}} = \frac{1}{\sqrt{2}}$$

- Project $[-3, -1]^T$ on direction **u**:

$$-3\frac{1}{\sqrt{2}} + (-1)\frac{1}{\sqrt{2}} = -\frac{4}{\sqrt{2}}$$

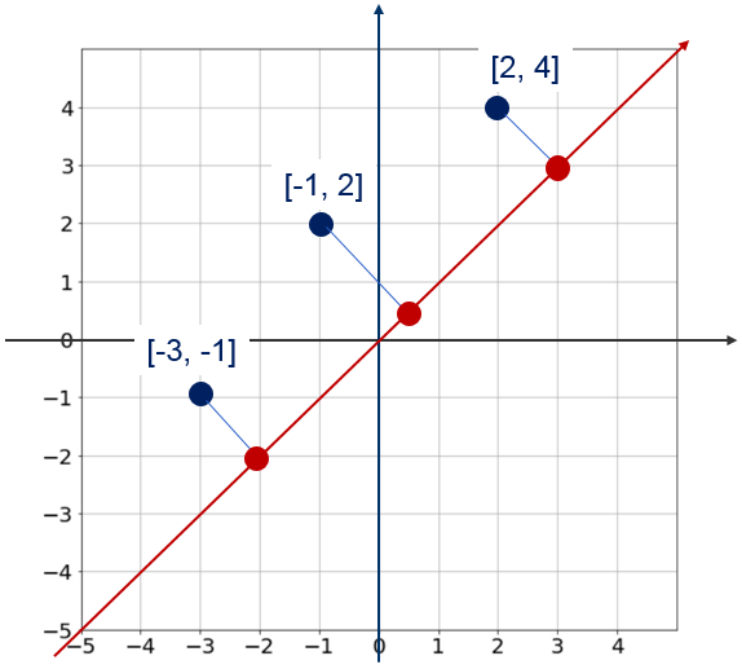


Projection for Dimensionality Reduction



Data Points in 2D

$$\mathbf{X} = \begin{bmatrix} 2 & -1 & -3 \\ 4 & 2 & -1 \end{bmatrix}$$



Projection onto 1D

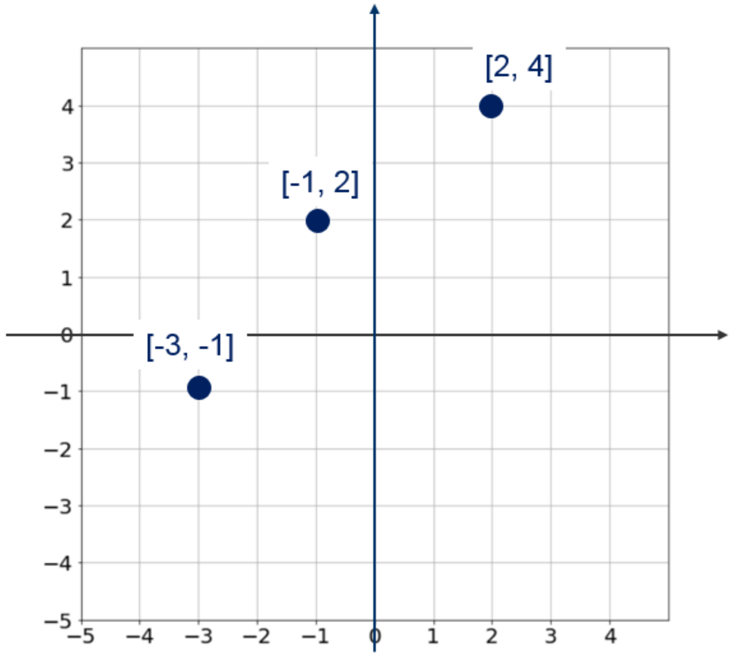
$$\mathbf{u}^{\mathrm{T}}\mathbf{X} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 2 & -1 & -3 \\ 4 & 2 & -1 \end{bmatrix} = \begin{bmatrix} \frac{6}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{4}{\sqrt{2}} \end{bmatrix}$$

Data Points in 1D

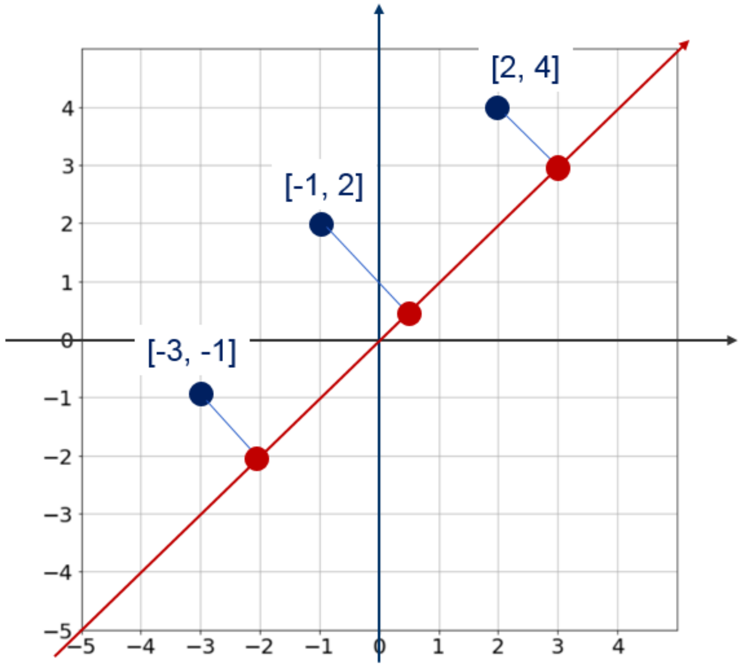
$$\mathbf{Z} = \begin{bmatrix} \frac{6}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{4}{\sqrt{2}} \end{bmatrix}$$

Projection for Dimensionality Reduction

This process projects 2 dimensional data to 1 dimensional data (i.e., dimensionality reduction).



Data Points in 2D



Projection onto 1D

Data Points in 1D

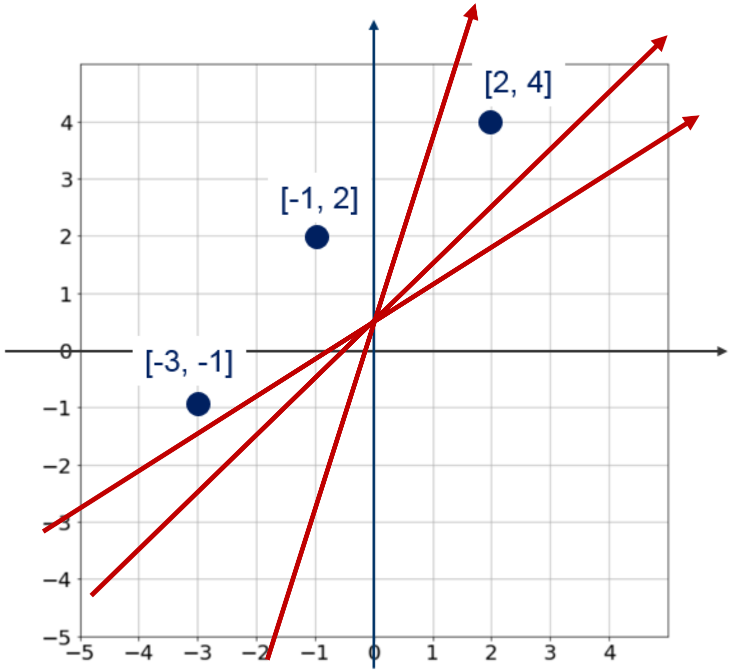
$$\mathbf{X} = \begin{bmatrix} 2 & -1 & -3 \\ 4 & 2 & -1 \end{bmatrix} \quad \Rightarrow \quad \mathbf{u}^{\mathrm{T}}\mathbf{X} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 2 & -1 & -3 \\ 4 & 2 & -1 \end{bmatrix} = \begin{bmatrix} \frac{6}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{4}{\sqrt{2}} \end{bmatrix} \quad \Rightarrow \quad \mathbf{Z} = \begin{bmatrix} \frac{6}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{4}{\sqrt{2}} \end{bmatrix}$$

Linear Dimensionality Reduction

- A projection matrix $\mathbf{U} = [\mathbf{u}_1\ \mathbf{u}_2\ \dots\ \mathbf{u}_k]$ of size $d*k$ defines k linear projection directions.
- Each column $\mathbf{u}_k$ in $\mathbf{U}$ denotes a linear project direction for d dimensional data (assume $k < d$)
- Then projection matrix $\mathbf{U}$ can be used to transform a high dimensional sample $\mathbf{x}$ into a low dimensional sample $\mathbf{z}$ by:

$$\underset{k*1}{\mathbf{z}} = \underset{k*d}{\mathbf{U}^T}\,\underset{d*1}{\mathbf{x}}$$

Linear Dimensionality Reduction

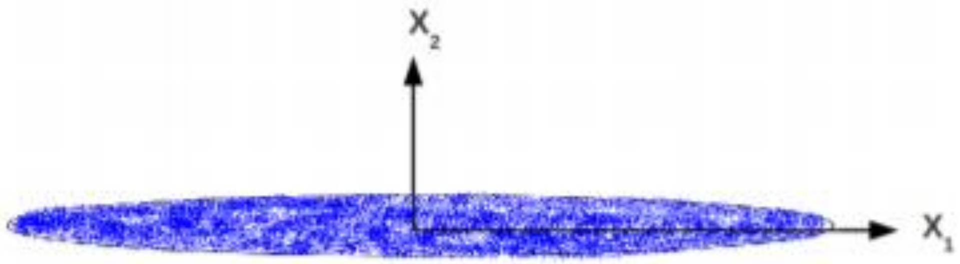


There are infinite ways to project the data **X**.

- How do we learn the “best” projection matrix **U**?
- What criteria should we optimize for learning **U**?
- Principle Component Analysis (PCA) is an algorithm for doing this.

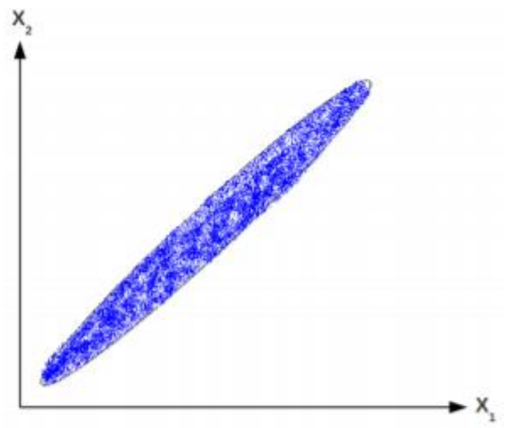
PCA as Maximizing Variance

PCA as Maximizing Variance: A Simple Illustration



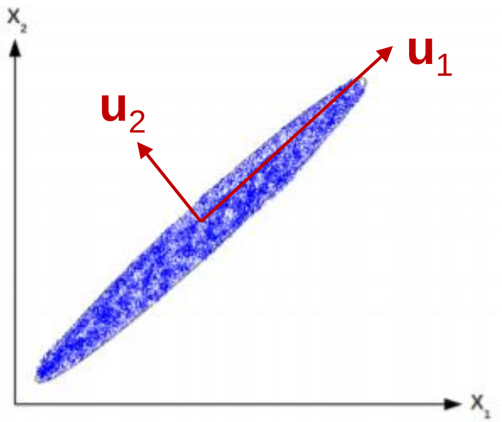
- Consider this two dimensional data
- Each data sample **x** is represented by 2 features $[x_1, x_2]^T$
- Considering ignoring the feature x_2 for each data sample
- Each 2-dimensional data sample **x** now becomes one-dimensional $[x_1]$
- Are we losing much information by simply removing x_2 ?
  - **No.** Most of the data spread is along x_1 (very little variance along x_2)

PCA as Maximizing Variance: A Simple Illustration



- Consider this two dimensional data
- Each data sample **x** is represented by 2 features $[x_1, x_2]^T$
- Considering ignoring the feature x_2 for each data sample
- Each 2-dimensional data sample **x** now becomes one-dimensional $[x_1]$
- Are we losing much information by simply removing x_2 ?
  - **Yes.** This data has substantial variance along both features.

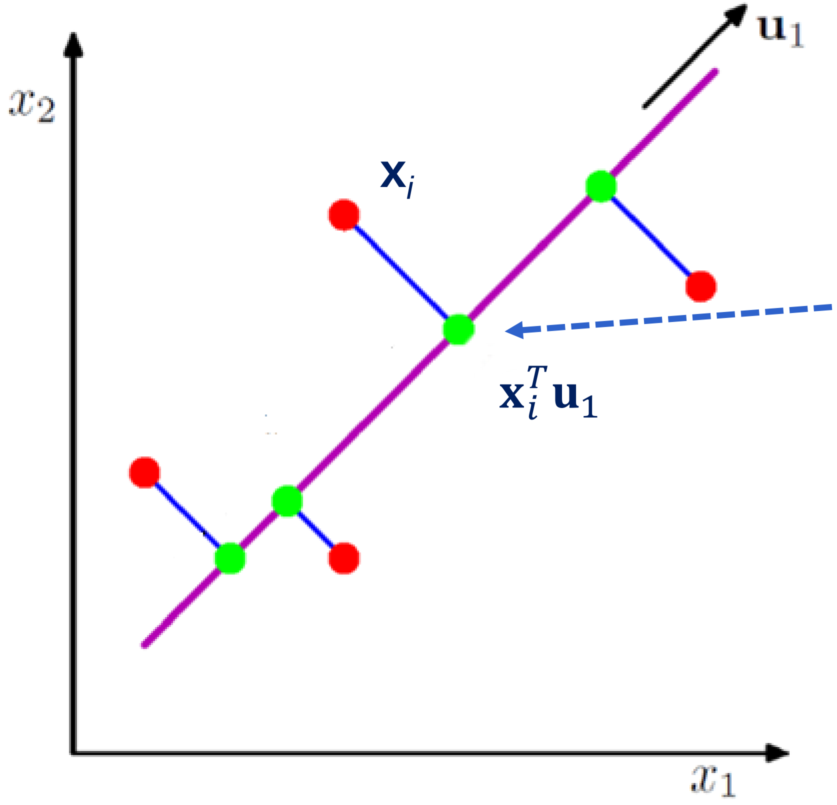
PCA as Maximizing Variance: A Simple Illustration



- Now consider we project the data into another two directions $\mathbf{u}_1$, $\mathbf{u}_2$
- Each data sample $\mathbf{x}$ is represented by 2 features $[z_1, z_2]^T$
- Considering ignoring the feature z_2 for each data sample
- Each 2-dimensional data sample $\mathbf{x}$ now becomes one-dimensional $[z_1]$
- Are we losing much information by simply removing z_2 ?
  - **No.** Most of the data spread is along z_1 (very little variance along z_2)

PCA as Maximizing Variance

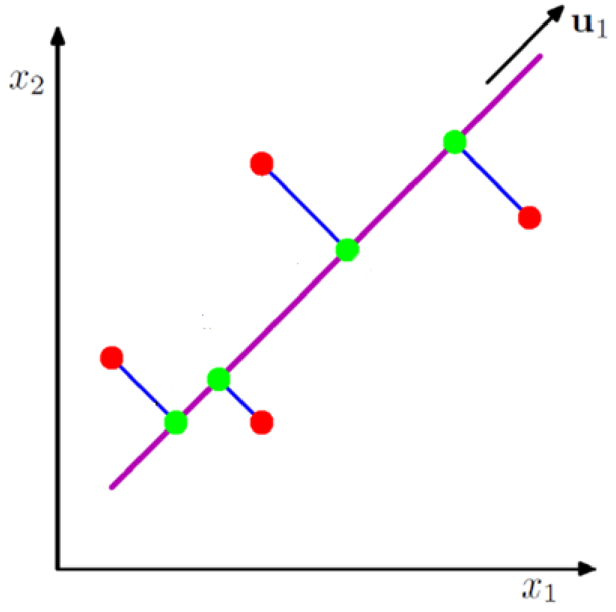
- Projecting $\mathbf{x}_i$ (a d-dimensional feature vector) to a one-dimensional vector z_i by $\mathbf{u}_1$: $z_i = \mathbf{u}_1^T \mathbf{x}_i = \mathbf{x}_i^T \mathbf{u}_1$



$\mathbf{x}_i^T \mathbf{u}_1$ denotes the location of the green point along the purple line representing $\mathbf{u}_1$

PCA as Maximizing Variance

- Projecting $\mathbf{x}_i$ (a d-dimensional feature vector) to a one-dimensional vector z_i by $\mathbf{u}_1$: $z_i = \mathbf{u}_1^T\mathbf{x}_i = \mathbf{x}_i^T\mathbf{u}_1$



- Therefore, the mean of projections of all data (i.e., "center" of the green points) can be computed as

$$\frac{\sum_{i=1}^{n}\mathbf{x}_i^T\mathbf{u}_1}{n} = \frac{\sum_{i=1}^{n}\mathbf{x}_i^T}{n}\mathbf{u}_1 = \bar{\mathbf{x}}^T\mathbf{u}_1$$

$\bar{\mathbf{x}}$ is the mean feature vector $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$

- Variance of the projected data (i.e., "spread" of the green points)

$$\frac{\sum_{i=1}^{n}(\mathbf{x}_i^T\mathbf{u}_1 - \bar{\mathbf{x}}^T\mathbf{u}_1)^2}{n} = \frac{\sum_{i=1}^{n}((\mathbf{x}_i^T - \bar{\mathbf{x}}^T)\mathbf{u}_1)^2}{n}$$

PCA as Maximizing Variance

- Variance of the projected data

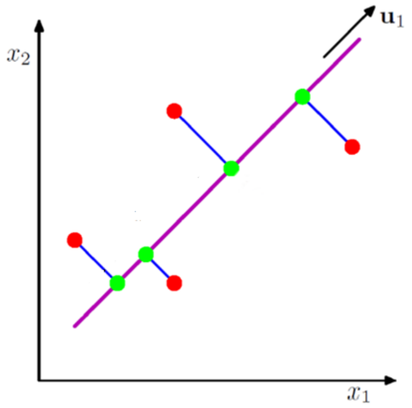
$$\frac{\sum_{i=1}^{n}((\mathbf{x}_i^T - \bar{\mathbf{x}}^T)\mathbf{u}_1)^2}{n} = \mathbf{u}_1^T \frac{\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i^T - \bar{\mathbf{x}}^T)}{n} \mathbf{u}_1$$

- Let $\boldsymbol{S} = \frac{\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i^T - \bar{\mathbf{x}}^T)}{n}$, the variance of the projected data is

$$\mathbf{u}_1^T \mathbf{S} \mathbf{u}_1$$

- **S** is the *d***d* data covariance matrix. If data is already centered (i.e., $\bar{\mathbf{x}}$ = 0), then $\boldsymbol{S} = \frac{\sum_{i=1}^{n}(\mathbf{x}_i)(\mathbf{x}_i^T)}{n} = \frac{1}{n}\mathbf{X}\mathbf{X}^T$

Direction of Maximum Variance



Variance of the projected data is:

$$\mathbf{u}_1^T \mathbf{S} \mathbf{u}_1$$

- Objective: We want $\mathbf{u}_1$ that the variance of the project data is maximized

$$\max_{\mathbf{u}_1} \mathbf{u}_1^T \mathbf{S} \mathbf{u}_1$$

- To prevent trivial solution (max variance = infinite), assume $\|\mathbf{u}_1\|_2 = \sqrt{\mathbf{u}_1^T \mathbf{u}_1} = 1$. Therefore $\mathbf{u}_1^T \mathbf{u}_1 = 1$
- Therefore, $\mathbf{u}_1$ can be obtained by solving the following optimization problem

$$\max_{\mathbf{u}_1} \mathbf{u}_1^T \mathbf{S} \mathbf{u}_1 + \lambda_1 (1 - \mathbf{u}_1^T \mathbf{u}_1)$$

λ_1 is a Lagrange multiplier

Direction of Maximum Variance

- The objective: $\max_{\mathbf{u}_1} \mathbf{u}_1^T \mathbf{S} \mathbf{u}_1 + \lambda_1 (1 - \mathbf{u}_1^T \mathbf{u}_1)$

- Obtaining the optimal solution by taking the derivative with respect to $\mathbf{u}_1$ and setting to zero

$$\mathbf{S}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1$$

- Thus $\mathbf{u}_1$ is an eigenvector of **S** (with corresponding eigenvalue λ_1)

- **S** is a *d***d* matrix, there are *d* possible eigenvectors, which ones to take?

Direction of Maximum Variance

- Note that the constraint $\mathbf{u}_1^T\mathbf{u}_1 = 1$, the variance of the projected data is

$$\mathbf{u}_1^T\mathbf{S}\mathbf{u}_1 = \mathbf{u}_1^T\lambda_1\mathbf{u}_1 = \lambda_1\mathbf{u}_1^T\mathbf{u}_1 = \lambda_1$$

- Therefore, variance is maximized when $\mathbf{u}_1$ is the (top) eigenvector with largest eigenvalue.

- **Other directions can also be found similarly (with each being orthogonal to all previous ones)**

Direction of Maximum Variance

- Question: What is $\mathbf{u}_2$?

$$\max_{\mathbf{u}_2} \mathbf{u}_2^T \mathbf{S} \mathbf{u}_2$$
$$s.t.\ \mathbf{u}_2^T \mathbf{u}_2 = 1, \mathbf{u}_2^T \mathbf{u}_1 = 0$$

$$\Downarrow$$

$$\max_{\mathbf{u}_2}\ \mathbf{u}_2^T \mathbf{S} \mathbf{u}_2 - \lambda(\mathbf{u}_2^T \mathbf{u}_2 - 1) - \phi \mathbf{u}_2^T \mathbf{u}_1$$

$$\Downarrow$$

$$\frac{\partial}{\partial \mathbf{u}_2}(\mathbf{u}_2^T \mathbf{S} \mathbf{u}_2 - \lambda(\mathbf{u}_2^T \mathbf{u}_2 - 1) - \phi \mathbf{u}_2^T \mathbf{u}_1) = 0$$

Direction of Maximum Variance

- Question: What is $\mathbf{u}_2$?

$$\frac{\partial}{\partial \mathbf{u}_2}(\mathbf{u}_2^T\mathbf{S}\mathbf{u}_2 - \lambda(\mathbf{u}_2^T\mathbf{u}_2 - 1) - \phi\mathbf{u}_2^T\mathbf{u}_1) = 0$$

$$\Downarrow$$

$$2\mathbf{S}\mathbf{u}_2 - 2\lambda\mathbf{u}_2 - \phi\mathbf{u}_1 = 0$$

$$\phi = 0\ ? \quad \Downarrow$$

$$\mathbf{S}\mathbf{u}_2 = \lambda\mathbf{u}_2$$

$\mathbf{u}_2$ is the eigenvector with the second largest eigenvalue.

To show that $\phi = 0$, we multiply $\mathbf{u}_1^T$ on both side, and we have:

$$\mathbf{u}_1^T(2\mathbf{S}\mathbf{u}_2 - 2\lambda\mathbf{u}_2 - \phi\mathbf{u}_1) = \mathbf{u}_1^T 0 = 0$$

$$\Downarrow$$

$$2\mathbf{u}_1^T\mathbf{S}\mathbf{u}_2 - 2\lambda\mathbf{u}_1^T\mathbf{u}_2 - \phi\mathbf{u}_1^T\mathbf{u}_1 = 0$$

$$\Downarrow$$

$$2\mathbf{u}_1^T\mathbf{S}^T\mathbf{u}_2 - 0 - \phi\mathbf{u}_1^T\mathbf{u}_1 = 0$$

$$\Downarrow$$

$$(\mathbf{S}\mathbf{u}_1)^T\mathbf{u}_2 - \phi\mathbf{u}_1^T\mathbf{u}_1 = 0$$

$$\Downarrow$$

$$(\boldsymbol{\lambda}_1\mathbf{u}_1)^T\mathbf{u}_2 - \phi\mathbf{u}_1^T\mathbf{u}_1 = 0$$

$$\Downarrow$$

$$\boldsymbol{\lambda}_1\mathbf{u}_1^T\mathbf{u}_2 - \phi\mathbf{u}_1^T\mathbf{u}_1 = 0$$

$$\Downarrow$$

$$0 - \phi\mathbf{u}_1^T\mathbf{u}_1 = 0 \ \Rightarrow\ \phi = 0$$

Steps of Principle Component Analysis

- Center the data (subtract the mean $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ from each data point) to get $\mathbf{X}_c$
- Compute the covariance matrix **S** using the centered data as

$$\mathbf{S} = \frac{1}{n}\mathbf{X}_c\mathbf{X}_c^T$$

- Do an eigen-decomposition of the covariance matrix **S**
- Take first *k* leading eigenvectors $\{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ with *k* largest eigenvalue $\{\lambda_1, \ldots, \lambda_k\}$
- The final *k* dimensional representation of data is obtained by

$$\mathbf{Z} = \mathbf{U}^{\mathrm{T}}\mathbf{X}_c$$

How many Principal Components to Use?

- Eigenvalue λ_i measures the variance captured by the corresponding projection direction $\mathbf{u}_i$

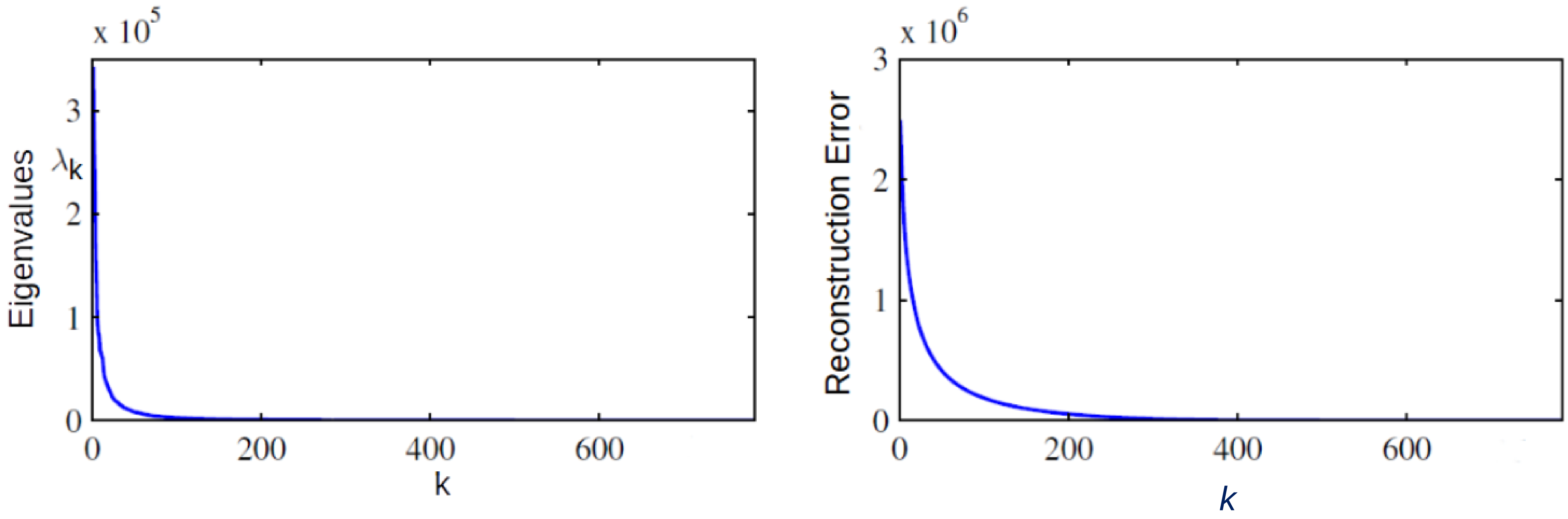
$$\mathbf{u}_i^T \mathbf{S} \mathbf{u}_i = \mathbf{u}_i^T \lambda_i \mathbf{u}_i = \lambda_i \mathbf{u}_i^T \mathbf{u}_i = \lambda_i$$

- The “left-over" variance will therefore be $\sum_{i=k+1}^{d} \lambda_i$

- Can choose *k* by looking at what fraction of variance is captured by the first k projection directions: $\mathbf{u}_1, \mathbf{u}_2, \ldots \mathbf{u}_k$

- Another direct way is to look at the spectrum of the eigenvalues plot, or the plot of reconstruction error vs *k*

How many Principal Components to Use?



PCA for image compression




d=1 **d=2** **d=4** **d=8**

d=16 **d=32** **d=64** **d=100**




$$z = U^T x$$
$$\overline{x} = Uz$$
$$\overline{x} = UU^T x$$

Original Image

64*64

References and Acknowledgement

- Korn F, Jagadish H V, Faloutsos C. Efficiently supporting ad hoc queries in large datasets of time sequences[J]. Acm Sigmod Record, 1997, 26(2): 289-300.

- Zhang D, Zou L, Zhou X, et al. Integrating feature selection and feature extraction methods with deep learning to predict clinical outcome of breast cancer[J]. Ieee Access, 2018, 6: 28936-28944.

- Slide from Maria-Florina Balcan, Carnegie Mellon University, Advanced Introduction to Machine Learning. https://www.cs.cmu.edu/~10715-f18/lectures/pca-2018.pdf

- Slide from Piyush Rai, Indian Institute of Technology Kanpur, Probabilistic Machine Learning (CS772A). https://www.cse.iitk.ac.in/users/piyush/courses/pml_winter16/slides_lec10.pdf

- https://slideplayer.com/slide/14520684/

- https://builtin.com/data-science/curse-dimensionality