# CS540 Introduction to Artificial Intelligence
## Lecture 12

Young Wu
Based on lecture slides by Jerry Zhu, Yingyu Liang, and Charles Dyer

June 28, 2020

# High Dimensional Data

## Motivation

- High dimensional data are training set with a lot of features.

1. Document classification.
2. MEG brain imaging.
3. Handwritten digits (or images in general).

$\begin{pmatrix}0.1\\0.2\\0.3\end{pmatrix} \leftarrow \begin{pmatrix}1\\0\\0\\0\end{pmatrix}$ vocab

# Low Dimension Representation

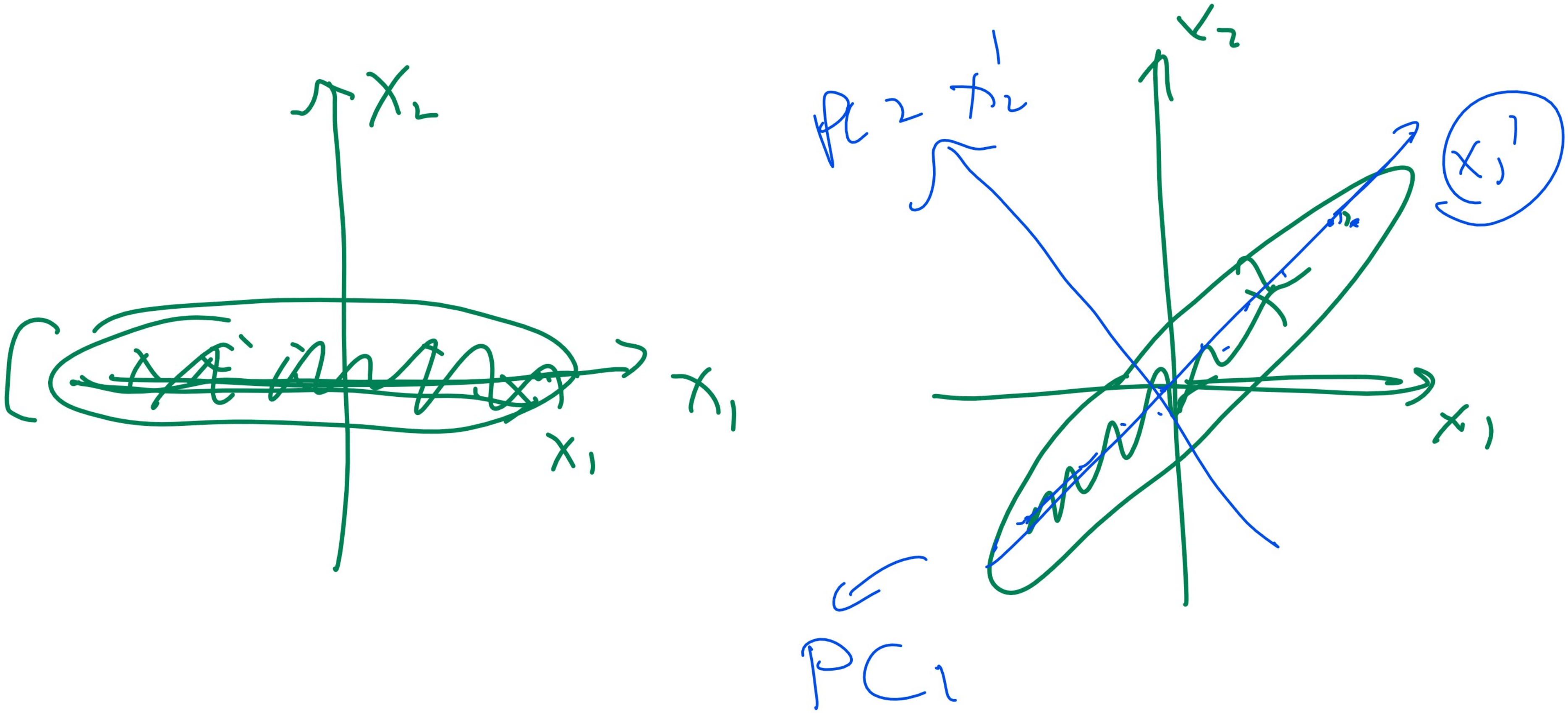
## Motivation

- Unsupervised learning techniques are used to find low dimensional representation.

1. Visualization. 2D 3D
2. Efficient storage.
3. Better generalization.
4. Noise removal.

# Dimension Reduction Diagram

## Motivation



# Dimension Reduction

## Description

- Rotate the axes so that they capture the directions of the greatest variability of data.
- The new axes (orthogonal directions) are principal components.

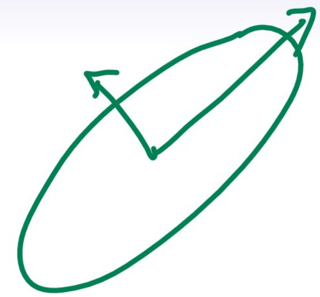
# Principal Component Analysis

## Description

- Find the direction of the greatest variability in data, call it $u_1$.
- Find the next direction orthogonal to $u_1$ of the greatest variability, call it $u_2$.
- Repeat until there are $u_1, u_2, \ldots, u_K$.

# Orthogonal Directions

## Definition

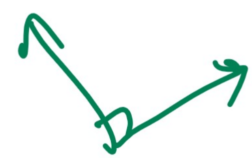


- In Euclidean space ($L_2$ norm), a unit vector $u_k$ has length 1.

$$\|u_k\|_2^2 = \|u_k\|_2 = u_k^T u_k = 1$$

- Two vectors $u_k, u_{k'}$ are orthogonal (or uncorrelated) if the dot product is 0.

$$u_k \cdot u_{k'} = \boxed{u_k^T u_{k'} = 0}$$



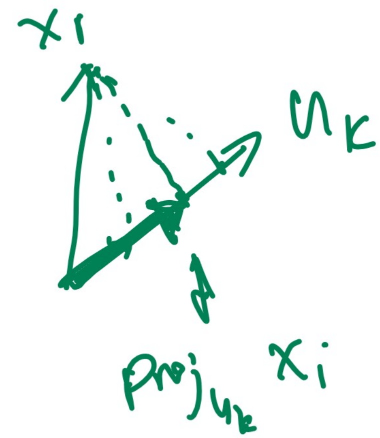
# Projection

## Definition

- The projection of $x_i$ onto a unit vector $u_k$ is the vector in the direction of $u_k$ that is the closest to $x_i$.


$$\text{proj}_{u_k} x_i \overset{\text{def}}{=} \left( \frac{u_k^T x_i}{u_k^T u_k} \right) u_k = u_k^T x_i u_k$$

$u_k^T u_k = 1$ unit




- The length of the projection of $x_i$ onto a unit vector $u_k$ is $u_k^T x_i$.

$$\left\| \text{proj}_{u_k} x_i \right\|_2 = u_k^T x_i$$

$\| u_k^T x_i u_k \|_2$

# Variance

## Definition

- The sample variance of a data set $\{x_1, x_2, ..., x_n\}$ is the sum of the squared distance from the mean.

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \dots \\ x_n^T \end{bmatrix}$$

$x_{i1} \; x_{i2} \; \cdots \; x_{im}$

row = 1 instance

# features ,

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

mean

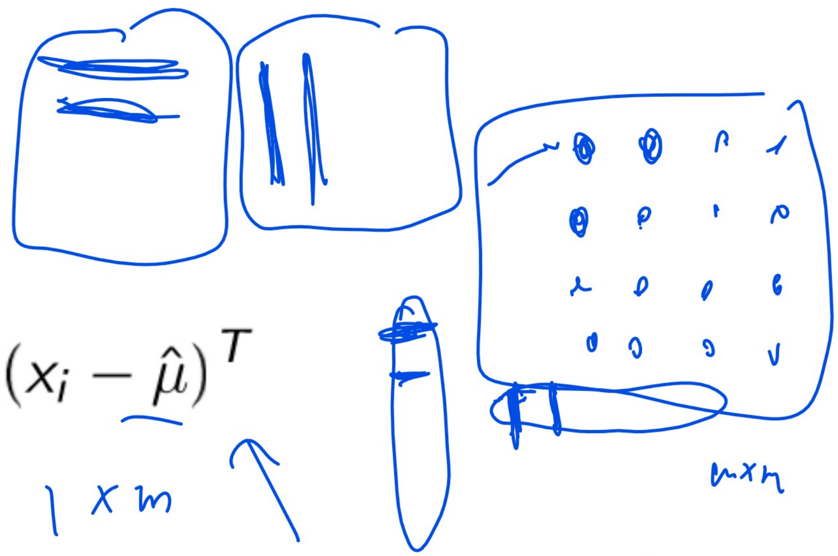
m x 1 m x 1

$$\hat{\Sigma} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \hat{\mu})(x_i - \hat{\mu})^T$$

variance

m x m m x 1 1 x m

m x m



m x m

not dot prod

# Normalization

## Definition

- Normalize the data by subtracting the mean, then the variance expression can be simplified.

$$x_i = x_i - \mu$$

$$\hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n} x_i x_i^T = \frac{1}{n-1} X^T X$$

# Covariance Matrix

## Definition

- $\hat{\Sigma}$ is an $m \times m$ matrix and it is usually called the sample covariance matrix. The diagonal elements are variances in each dimension.

$$\hat{\sigma}_j^2 = \hat{\Sigma}_{jj} = \frac{1}{n-1} \sum_{i=1}^{n} x_{ij}^2$$

# Projected Variance

## Definition

- Note that $x_{ij} = e_j^T x_i$, where $e_j$ is the vector of 0 except it is 1 in coordinate $j$.

$$\hat{\sigma}_j^2 = e_j^T \hat{\Sigma} e_j = \frac{1}{n-1} e_j^T X^T X e_j$$
$$= \frac{1}{n-1} \sum_{i=1}^{n} \left( e_j^T x_i \right)^2$$

- The variance of the normalized $x_i$ projected onto direction $u_k$ has a similar expression.

$$u_k^T \hat{\Sigma} u_k = \frac{1}{n-1} u_k^T X^T X u_k$$
$$= \frac{1}{n-1} \sum_{i=1}^{n} \left( u_k^T x_i \right)^2$$

# Maximum Variance Directions

## Definition

- The goal is to find the direction that maximizes the projected variance.

$$\max_{u_k} u_k^T \hat{\Sigma} u_k \text{ such that } u_k^T u_k = 1$$

$$\Rightarrow \max_{u_k} u_k^T \hat{\Sigma} u_k - \lambda u_k^T u_k$$

$$\Rightarrow \hat{\Sigma} u_k = \lambda u_k$$

# Eigenvalue

## Definition

- The $\lambda$ represents the projected variance.

$$u_k^T \hat{\Sigma} u_k = u_k^T \lambda u_k = \lambda$$

- The larger the variance, the larger the variability in direction $u_k$. There are $m$ eigenvalues for a symmetric positive semidefinite matrix (for example, $X^T X$ is always symmetric PSD). Order the eigenvectors $u_k$ by the size of their corresponding eigenvalues $\lambda_k$.

$$\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_m$$

# Eigenvalue Algorithm

## Definition

- Solving eigenvalue using the definition (characteristic polynomial) is computationally inefficient.

$$\left(\hat{\Sigma} - \lambda_k I\right) u_k = 0 \Rightarrow \det \left(\hat{\Sigma} - \lambda_k I\right) = 0$$

- There are many fast eigenvalue algorithms that computes the spectral (eigen) decomposition for real symmetric matrices. Columns of $Q$ are unit eigenvectors and diagonal elements of $D$ are eigenvalues.

$$\begin{aligned} \hat{\Sigma} &= PDP^{-1}, D \text{ is diagonal} \\ &= QDQ^T, \text{ if } Q \text{ is orthogonal, i.e. } Q^T Q = I \end{aligned}$$

# Principal Component Analysis

## Algorithm

- Input: instances: $\{x_i\}_{i=1}^{n}$, the number of dimensions after reduction $K < m$.
- Output: $K$ principal components.
- Find the largest $K$ eigenvalues $\lambda_1 \geqslant \lambda_2 \geqslant ... \geqslant \lambda_K$ .
- Return the corresponding unit orthogonal eigenvectors $u_1, u_2 ... u_K$ .

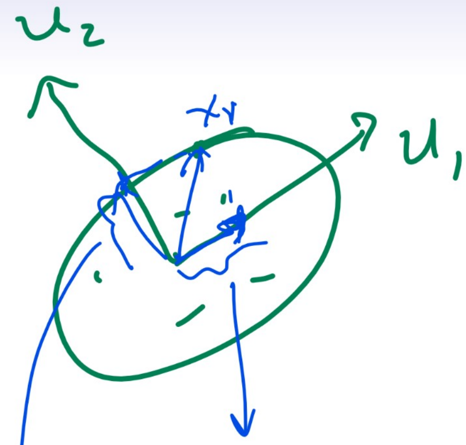
# Number of Dimensions

## Discussion

- There are a few ways to choose the number of principal components $K$.
- $K$ can be selected given prior knowledge or requirement.
- $K$ can be the number of non-zero eigenvalues.
- $K$ can be the number of eigenvalues that are large (larger than some threshold).

# Reduced Feature Space

## Discussion




- The original feature space is $m$ dimensional.

$$(x_{i1}, x_{i2}, \ldots, x_{im})^T$$

$$\begin{pmatrix} u_1^T x_i \\ u_2^T x_i \\ \vdots \end{pmatrix}$$

- The new feature space is $K$ dimensional.

$$\left(u_1^T x_i, u_2^T x_i, \ldots, u_K^T x_i\right)^T$$

- Other supervised learning algorithms can be applied on the new features.

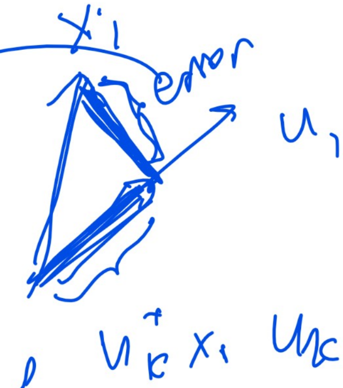
SVM

# Reconstruction Error

## Discussion

- Reconstruction error is the squared error (distance) between the original data and its projection onto $u_k$.

$$\left\| x_i - \left( u_k^T x_i \right) u_k \right\|^2$$




- Finding the variance maximizing directions is the same as finding the reconstruction error minimizing directions.

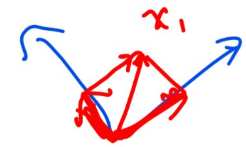
$$\min \quad \frac{1}{n} \sum_{i=1}^{n} \left\| x_i - \left( u_k^T x_i \right) u_k \right\|^2 \quad = \quad \max \; u_k^T \Sigma u_k$$

# Reconstruction Error Diagram

## Discussion

# Eigenface

## Discussion




PC

- Eigenfaces are eigenvectors of face images (pixel intensities or HOG features).
- Every face can be written as a linear combination of eigenfaces. The coefficients determine specific faces.

$$x_i = \sum_{k=1}^{m} \left(u_k^T x_i\right) u_k \approx \sum_{k=1}^{K < m} \left(\boxed{u_k^T x_i}\right) u_k$$

coeff PC

K < m

Smaller input

- Eigenfaces and SVM can be combined to detect or recognize faces.

# Autoencoder

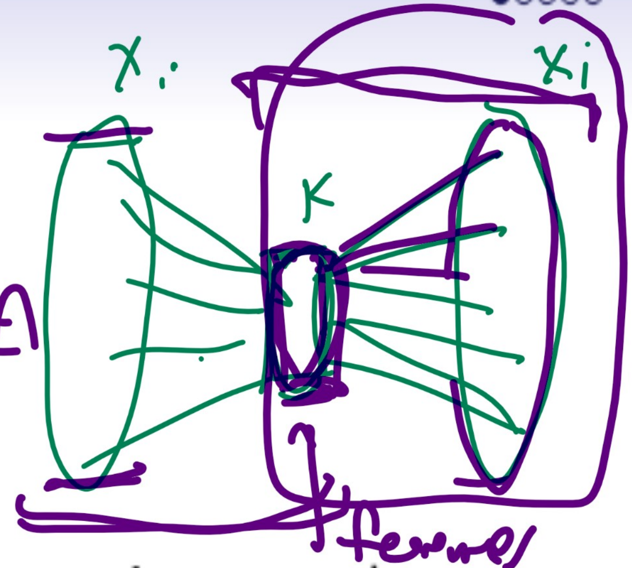
## Discussion

$$a_i = g(wx + b)$$

linear

no activation

≈ PCA




- A multi-layer neural network with the same input and output $y_i = x_i$ is called an autoencoder.
- The hidden layers have fewer units than the dimension of the input $m$.
- The hidden units form an encoding of the input with reduced dimensionality.

# Autoencoder Diagram

## Discussion

# Kernel PCA

## Discussion

SVM

- A kernel can be applied before finding the principal components.

feature map

$$\hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n}\varphi(x_i)\,\varphi(x_i)^T$$

Kernel matrix

- The principal components can be found without explicitly computing $\varphi(x_i)$, similar to the kernel trick for support vector machines.
- Kernel PCA is a non-linear dimensionality reduction method.

# T-Distributed Stochastic Neighbor Embedding

## Discussion

- t-distributed stochastic neighbor embedding is another non-linear dimensionality reduction method used mainly for visualization.
- Points in high dimensional spaces are embedded in 2 or 3-dimensional spaces to preserve the distance (neighbor) relationship between points.

# Embedding Diagram

## Discussion