

Lecture 8

Singular Value Decomposition in Machine Learning

SWCON253, Machine Learning

Won Hee Lee, PhD

Learning Goals

- Understand the fundamental concepts of **singular value decomposition (SVD)** in machine learning

$X \in \mathbb{R}^{n \times p}$ has SVD, $U \Sigma V^T$, where

$U \in \mathbb{R}^{n \times n}$ is orthogonal $U^T U = U U^T = I$ (columns of U = left singular vectors)

$V \in \mathbb{R}^{p \times p}$ is orthogonal $V^T V = V V^T = I$ (columns of V = right singular vectors)

$\Sigma \in \mathbb{R}^{n \times p}$ is diagonal with diagonal elements (singular values),

$$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(n,p)} \geq 0 \quad \#\sigma_i > 0 = \text{rank}(X)$$

$n = p$

$$\Sigma = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}$$

$n > p$

$$\left[\begin{array}{ccc} \sigma_1 & & \\ & \ddots & \\ & & \sigma_p \\ & & & \end{array} \right] \begin{array}{l} \left. \begin{array}{c} \text{ } \end{array} \right\} p \times p \\ \left. \begin{array}{c} \text{ } \end{array} \right\} (n-p) \times p \end{array}$$

$n < p$

$$\left[\begin{array}{ccc} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{array} \right] \begin{array}{l} \left. \begin{array}{c} \text{ } \end{array} \right\} n \times n \\ \left. \begin{array}{c} \text{ } \end{array} \right\} n \times (p-n) \end{array}$$

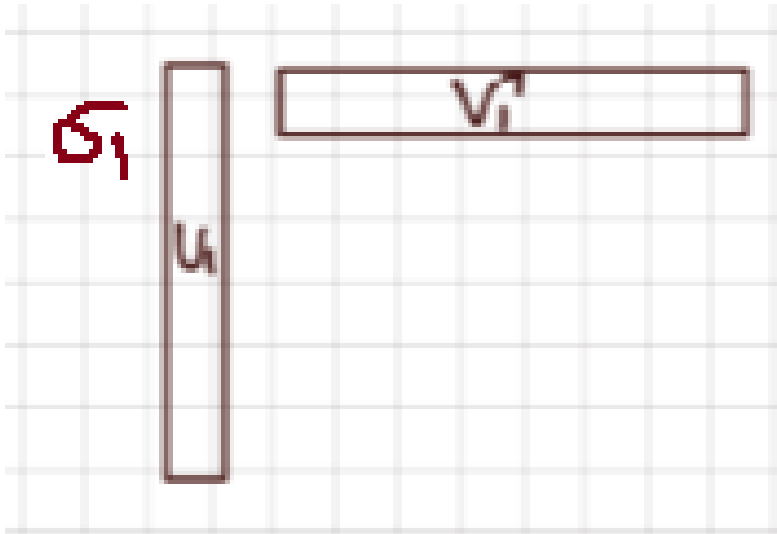
If we have a matrix X , we can find matrices U, V^T that satisfy these criteria, and then we compute the SVD.

Ex.

$$X = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix} = I \begin{bmatrix} 10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix} I = U \Sigma V^T$$

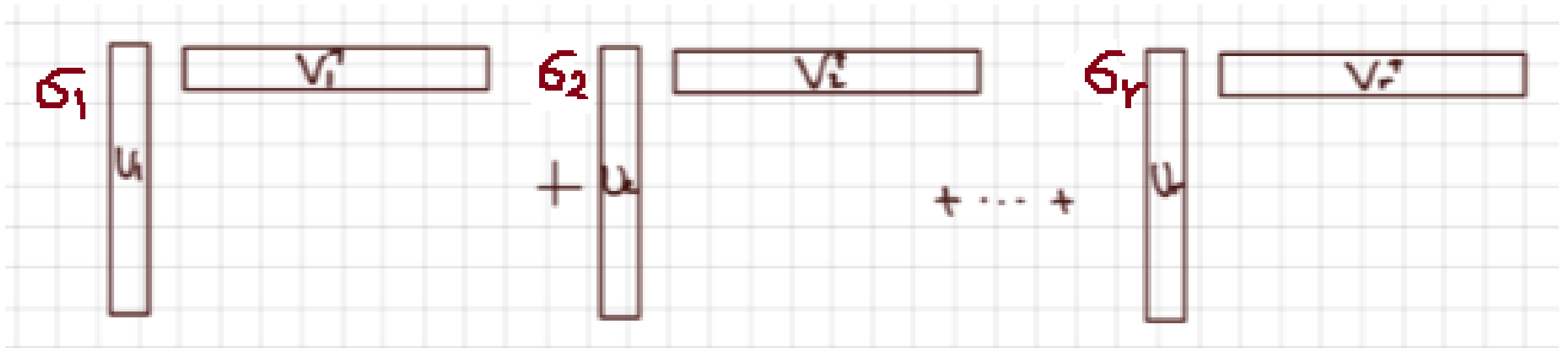
Equivalently, we can write X as sum of rank-1 matrices

$$X = \sum_{i=1}^{r=\min(n,p)} \sigma_i \underset{\text{(i}^{\text{th}} \text{ column of } U)}{u_i} \overset{\text{(i}^{\text{th}} \text{ column of } V)}{v_i^T}$$



Equivalently, we can write X as sum of rank-1 matrices

$$X = \sum_{i=1}^{r=\min(n,p)} \sigma_i \underset{\text{(i}^{\text{th}} \text{ column of } U)}{u_i} \overset{\text{(i}^{\text{th}} \text{ column of } V)}{v_i^T}$$



The “Economy” SVD

Assume $X \in \mathbb{R}^{n \times p}$ has $\text{rank}(X) = r \ll \min(n, p)$

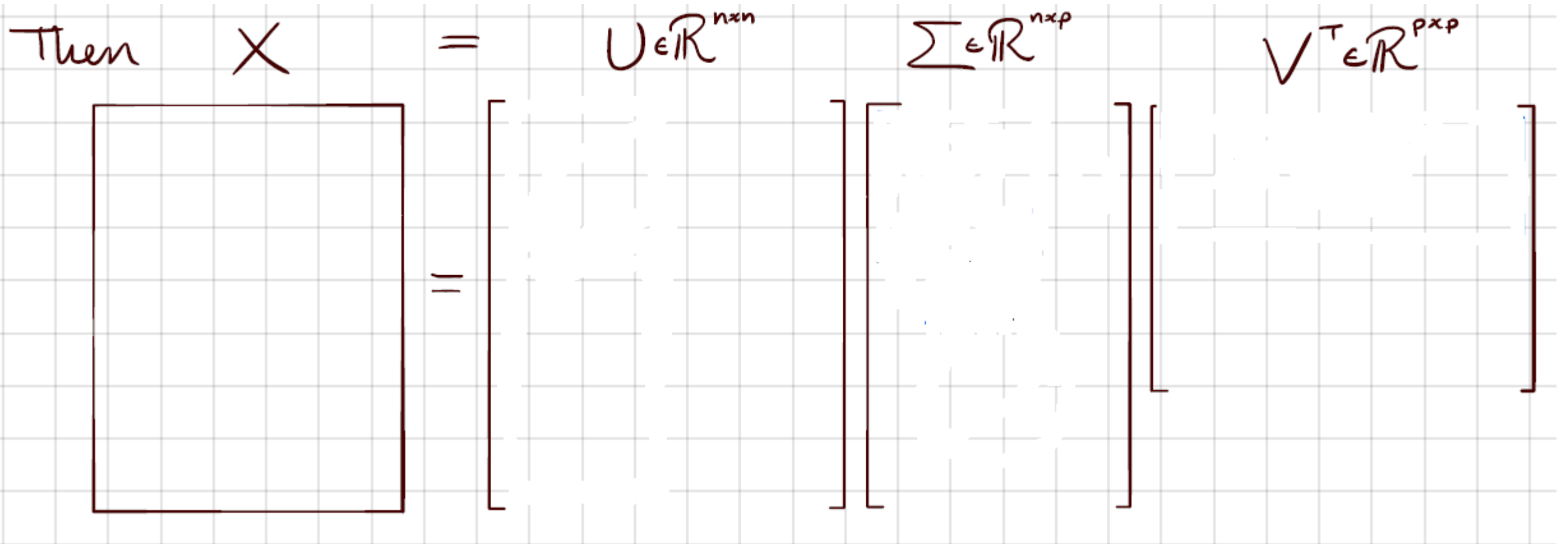
What if you have a matrix (million by million), but only rank-10?

What we want to do with **the economy SVD** is to represent all the information about the SVD of X **without storing huge matrices**.

The "Economy" SVD

Assume $X \in \mathbb{R}^{n \times p}$ has $\text{rank}(X) = r \ll \min(n, p)$

Then $X = U \in \mathbb{R}^{n \times n} \Sigma \in \mathbb{R}^{n \times p} V^T \in \mathbb{R}^{p \times p}$



Assume $X \in \mathbb{R}^{n \times p}$ has $\text{rank}(X) = r \ll \min(n, p)$

Then $X = U \in \mathbb{R}^{n \times n} \quad \Sigma \in \mathbb{R}^{n \times p} \quad V^T \in \mathbb{R}^{p \times p}$

The diagram illustrates the SVD decomposition $X = U \Sigma V^T$. Matrix X is a rectangle. Matrix U is a tall rectangle. Matrix Σ is a large rectangle containing a blue-outlined square of size $r \times r$. Inside this square are circles labeled $\sigma_1, \sigma_2, \dots, \sigma_r$. Below the square is a blue label $\Sigma \in \mathbb{R}^{r \times r}$. To the right of the square is a circle. Below the entire Σ matrix is a large circle. Matrix V^T is a wide rectangle.

when rank r is small

Then $X = U \in \mathbb{R}^{n \times n} \Sigma \in \mathbb{R}^{n \times p} V^T \in \mathbb{R}^{p \times p}$

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix X . The equation is written as $X = U \Sigma V^T$, with dimensions specified for each matrix: $U \in \mathbb{R}^{n \times n}$, $\Sigma \in \mathbb{R}^{n \times p}$, and $V^T \in \mathbb{R}^{p \times p}$.

The matrix U is represented by a blue rectangle labeled $\tilde{U} \in \mathbb{R}^{n \times r}$. The matrix Σ is represented by a blue rectangle containing singular values $\sigma_1, \sigma_2, \dots, \sigma_r$ on a diagonal, labeled $\Sigma \in \mathbb{R}^{r \times r}$. The matrix V^T is represented by a large rectangle.

Then $X = U \in \mathbb{R}^{n \times n} \Sigma \in \mathbb{R}^{n \times p} V^T \in \mathbb{R}^{p \times p}$

Diagram illustrating the Singular Value Decomposition (SVD) of matrix X :

- X is represented by a large empty rectangle.
- U is represented by a blue rectangle containing the text $\tilde{U} \in \mathbb{R}^{n \times r}$.
- Σ is represented by a blue rectangle containing a dashed box with diagonal elements $\sigma_1, \sigma_2, \dots, \sigma_r$ and a blue label $\Sigma \in \mathbb{R}^{r \times r}$ with an arrow pointing to the dashed box.
- V^T is represented by a blue rectangle containing the text $V^T \in \mathbb{R}^{p \times p}$.

Then $X = U \in \mathbb{R}^{n \times n} \quad \Sigma \in \mathbb{R}^{n \times p} \quad V^T \in \mathbb{R}^{p \times p}$

The diagram illustrates the SVD decomposition $X = U \Sigma V^T$. The matrix X is represented by a large rectangle. It is equal to the product of three matrices: U , Σ , and V^T . The matrix U is shown as a rectangle with the label $\tilde{U} \in \mathbb{R}^{n \times r}$ inside. The matrix Σ is shown as a rectangle containing a smaller rectangle with diagonal elements $\sigma_1, \sigma_2, \dots, \sigma_r$ and a circle below it, with the label $\Sigma \in \mathbb{R}^{n \times r}$ below it. The matrix V^T is shown as a rectangle with the label $V^T \in \mathbb{R}^{r \times p}$ inside.

$$X = U \Sigma V^T = \underline{\tilde{U} \tilde{\Sigma} \tilde{V}^T}$$

this is the economy SVD

Properties of the economy properties

$\tilde{\Sigma} \in \mathbb{R}^{r \times r}$ diagonal with diagonal elements (singular values),
 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$

$\tilde{U} \in \mathbb{R}^{n \times r}$ $\tilde{U}^T \tilde{U} = I$, but $\tilde{U} \tilde{U}^T \neq I$

$\tilde{V} \in \mathbb{R}^{p \times r}$ $\tilde{V}^T \tilde{V} = I$, but $\tilde{V} \tilde{V}^T \neq I$

Netflix example

n = number of movies $\approx 5k = 5000$

p = number of customers ≈ 100 million

storage = $5k \times 100 \text{ million} \times 4 \text{ bytes} = 2 \text{ TB}$

okay to store, difficult to use in learning algorithms

If X is rank-10, then we can compute the economy SVD

$\tilde{U} = 5k \text{ (row)} \times 10 \text{ (cols)} \times 4 \text{ bytes}$

$\tilde{\Sigma} = 10 \text{ (singular values)} \times 4 \text{ bytes}$

$\tilde{V} = 100 \text{ m} \times 10 \times 4 \text{ bytes}$

Storage = 4 GB

Dimensionality Reduction

We have n points $\underline{x}_1, \underline{x}_2, \dots, \underline{x}_n \in \mathbb{R}^p$

Goal: Find $\underline{z}_1, \underline{z}_2, \dots, \underline{z}_n \in \mathbb{R}^k$ for **$k < p$**

k implies some lower dimensional space than p .

\underline{z}_i is reduced dimensionality vectors.

We want to find for each one of these points (we want to map it) to smaller or fewer dimensional vectors.

We can represent essentially the same information but with fewer or smaller vectors.

Dimensionality reduction means defining new points $\underline{z}_i \in \mathbb{R}^k$ for $i=1, \dots, n$ for $k < p$ that preserve important properties of the \underline{x}_i 's.

Dimensionality Reduction

So that \underline{z}_i 's share useful properties with \underline{x}_i 's.

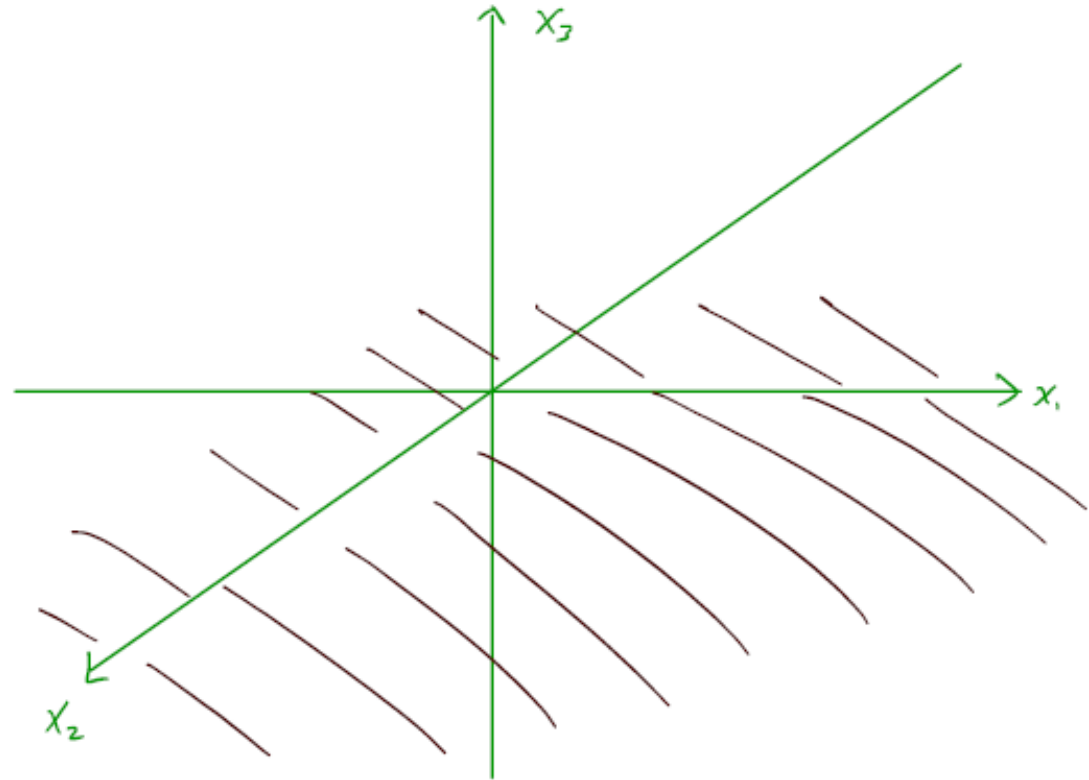
(e.g., $\|\underline{x}_i - \underline{x}_j\| \approx \|\underline{z}_i - \underline{z}_j\|$)

How?

Let's use the SVD.

Ex. $n=3$, $S=\{\underline{x} \in \mathbb{R}^3: x_3 = 0\}$

Horizontal plane



Instead of representing each x_i using 3D coordinates, we only need to represent when it is in the x_1 - x_2 plane.

Ex.

$$X = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 0 & 0 \end{bmatrix} \quad \text{e.g., } x_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \in \mathbb{R}^4$$

We want to find some z_i , reduced dimensional representation of x_i .

Ex.

$$X = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 0 & 0 \end{bmatrix} \quad \text{e.g., } x_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \in \mathbb{R}^4$$

$$X = \begin{bmatrix} 1 & 1 \\ -1 & 1 \\ 1 & 1 \\ -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

This is not SVD. Why? These matrices are not orthogonal.
What is rank of X? $\text{rank}(X)=2$

We will compute the SVD using an example:

$$X = \begin{bmatrix} 1 & 1 \\ -1 & 1 \\ 1 & 1 \\ -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

$$X = \begin{bmatrix} 1/\sqrt{5} & 1/2 \\ -1/\sqrt{5} & 1/2 \\ 1/\sqrt{5} & 1/2 \\ -1/\sqrt{5} & 1/2 \\ 1/\sqrt{5} & 0 \end{bmatrix} \begin{bmatrix} \sqrt{5}\sqrt{2} & 0 \\ 0 & 2\sqrt{2} \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 & 0 \\ 0 & 0 & -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

\tilde{U}
 $\tilde{\Sigma}$
 \tilde{V}^T

$$X = \begin{bmatrix} 1/\sqrt{5} & 1/2 \\ -1/\sqrt{5} & 1/2 \\ 1/\sqrt{5} & 1/2 \\ -1/\sqrt{5} & 1/2 \\ 1/\sqrt{5} & 0 \end{bmatrix} \begin{bmatrix} \sqrt{5}\sqrt{2} & 0 \\ 0 & 2\sqrt{2} \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 & 0 \\ 0 & 0 & -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

As the rank of X is 2, all of the columns of X lie in a 2D subspace and also all of the rows of X lie in a 2D subspace.

So, what they suggest is that we should be able to do some dimensionality reduction because everything lies in some sort of a 2D plane and

we don't necessarily need to all 4 dimensions to represent where these points are.

So, we try to accomplish dimensionality reduction using SVD.

Let's consider X^T . We want to make new versions that we want to reduce their dimension.

$$X^T = \begin{bmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ \underline{x_1} & \underline{x_2} & \underline{x_3} & \underline{x_4} & \underline{x_5} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} = \tilde{V} \tilde{\Sigma} \tilde{U}^T$$

$$= \underset{\tilde{V}}{1/\sqrt{2} \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & -1 \\ 0 & 1 \end{bmatrix}} \underset{\tilde{\Sigma} \tilde{U}^T}{\sqrt{2} \begin{bmatrix} 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}}$$

Each of the x_i 's that we want to reduce the dimensionality is the rank-4 vectors.

Each of them can be written as a **weighted sum of the columns of \tilde{V}** . So they all are in a 2D subspace.

$\tilde{\Sigma} \tilde{U}^T$ tells us where they lie in that subspace.

So we can say that

\underline{x}_i = **weighted sum of columns of \tilde{V} .**

\underline{z}_i = reduced dimensional vectors = **weights**

For example,

$$\underline{z}_1 = \sqrt{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \underline{z}_2 = \sqrt{2} \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \quad \dots$$

We started with 4D points x_i and then we used SVD to compute 2D representations that still preserve these distances, e.g., $\|\underline{x}_i - \underline{x}_j\| \approx \|\underline{z}_i - \underline{z}_j\|$.

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \rightarrow \underline{z}_1 = \sqrt{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

We are only using half of many features to represent \underline{z}_i .

Given (\underline{x}_i, y_i) for $i=1, \dots, n$

Labels $\underline{y} \in \mathbb{R}^p$ for n training samples

Features $X \in \mathbb{R}^{n \times p}$ (p features)

$$y_i \approx \hat{y}_i = \underline{x}^T \underline{\hat{w}}$$

$$y_i \approx \hat{y}_i = X \underline{\hat{w}}$$

If $n \geq p$, $\text{rank}(X) = p$ (X has p linearly independent columns), then

$$\underline{\hat{w}} = (X^T X)^{-1} X^T \underline{y}$$

In the context of the SVD, $X = U\Sigma V^T$

$$\begin{aligned} & (X^T X)^{-1} X^T \\ &= (V \Sigma^T U^T U \Sigma V^T)^{-1} V \Sigma^T U^T \\ &= (V \Sigma^T \Sigma V^T)^{-1} V \Sigma^T U^T \\ &= (\textcolor{red}{V} \Sigma^T \textcolor{red}{\Sigma} \textcolor{blue}{V}^T)^{-1} V \Sigma^T U^T \\ &= (\textcolor{blue}{V}^T)^{-1} (\textcolor{red}{V} \Sigma^T \textcolor{red}{\Sigma})^{-1} V \Sigma^T U^T \\ &= (V^T)^{-1} (\textcolor{green}{V} \Sigma^T \textcolor{orange}{\Sigma})^{-1} V \Sigma^T U^T \\ &= (V^{-1})^{-1} (\textcolor{green}{V} \Sigma^T \textcolor{orange}{\Sigma})^{-1} V \Sigma^T U^T \\ &= (V^{-1})^{-1} (\Sigma^T \Sigma)^{-1} V^{-1} V \Sigma^T U^T \\ &= V (\Sigma^T \Sigma)^{-1} V^T V \Sigma^T U^T \\ &= V (\Sigma^T \Sigma)^{-1} \Sigma^T U^T \end{aligned}$$

$$\hat{w} = (X^T X)^{-1} X^T \underline{y}$$

$$U^T U = I$$

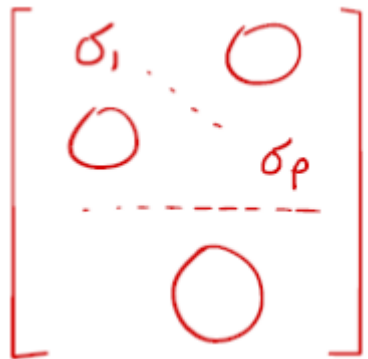
If A, B are square and invertible, then
 $(AB)^{-1} = B^{-1}A^{-1}$

$$V^T V = V V^T = I = V^{-1} V \rightarrow V^T = V^{-1}$$

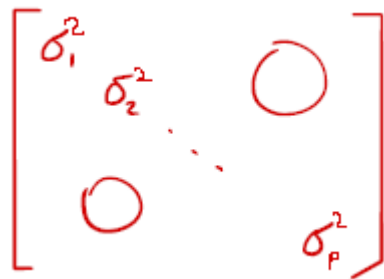
Let's think about $(\Sigma^T \Sigma)^{-1} \Sigma^T$

$n \geq p$

$\Sigma (n \times p)$



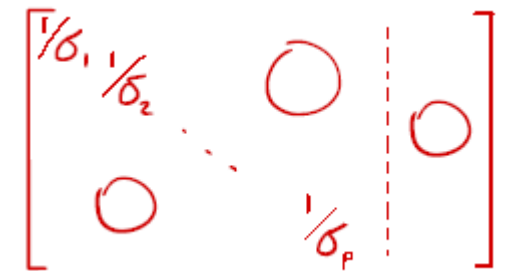
$\Sigma^T \Sigma (p \times p)$



$(\Sigma^T \Sigma)^{-1}$



$(\Sigma^T \Sigma)^{-1} \Sigma^T (p \times n)$



$= \Sigma^+$
(pseudo-inverse)

We are interested in $(X^T X)^{-1} X^T = V \Sigma^+ U^T$

We said our model

$$\hat{y} = U \Sigma V^T \underline{\hat{w}}$$

Our estimator

$$\underline{\hat{w}} = V \Sigma^+ U^T \underline{y}$$

$$y \approx \hat{y} = U \Sigma V^T \underline{\hat{w}}$$

$$y \approx \hat{y} = U \Sigma V^T \underline{\hat{w}}$$

Multiplying both sides by U^T

$$U^T y \approx U^T U \Sigma V^T \underline{\hat{w}}$$

$$U^T U = I$$

Multiplying both sides by Σ^+

$$\Sigma^+ U^T y \approx \Sigma^+ \Sigma V^T \underline{\hat{w}}$$

$$\Sigma^+ \Sigma = I$$

Multiplying both sides by V

$$V \Sigma^+ U^T y \approx V V^T \underline{\hat{w}}$$

$$V^T V = I$$

$$\underline{\hat{w}} \approx V \Sigma^+ U^T y$$

- Basically what we've shown here is that this **basic least squares model** $(X^T X)^{-1} X^T$ that involve these big matrices and inverses.
- We start thinking about using SVD as an elegant, simple and interpretable form.
- So, ideally what we'd like to do is to just invert this matrix $X=U\Sigma V^T$ (rectangular and generally non-invertible) and so
- What we do instead is we use this SVD where U's and V's are invertible and instead of inverting sigma Σ which is not exactly invertible.
- We use the notion of **pseudo-inverse** and this notion gives us exactly the least squares estimator and
- We like to do a matrix inverse (the matrix is not invertible), suddenly becomes very clean and elegant using SVD.

Announcements

- Homework #2
 - 스스로 복습한 내용을 A4용지에 손글씨로 작성/스캔하여 제출
 - 1-page 이상 per a lecture (lectures 6-8)
 - Any forms (pdf, jpeg, etc.) should be fine!
 - **Due April 1st Thursday at 11:59 pm**

SW Installation

1. Installing Python Packages
2. Installing Jupyter Notebook

SW Installation

- ◆ Python Machine Learning (PyML) Chapter 1.5 & Appendix
- ◆ Python (>3.7)
 - <https://www.python.org>
- ◆ Anaconda
 - A highly recommended alternative Python distribution for scientific computing
 - install: <https://docs.anaconda.com/anaconda/install/>
 - guide: <https://docs.anaconda.com/anaconda/user-guide/getting-started/>
- ◆ Python Core Packages
 - [NumPy](#) ≥ 1.17.4 : to use multi-dimensional arrays
 - [SciPy](#) ≥ 1.3.1 : to use scientific functions
 - [scikit-learn](#) ≥ 0.22.0 : to use ML algorithms
 - [matplotlib](#) ≥ 3.1.0 : to visualize quantitative data
 - [pandas](#) ≥ 0.25.3 : to make working with tabular data more convenient