Machine Learning Department
School of Computer Science
Carnegie Mellon University

Principal Component Analysis (PCA)

Matt Gormley, Henry Chai, Hoda Heidari

Lecture 23
Apr. 10, 2024

Reminders

- **Homework 8: Deep RL**
  - Out: Mon, Apr. 8
  - Due: Fri, Apr. 19 at 11:59pm

DIMENSIONALITY REDUCTION

High Dimension Data

Examples of high dimensional data:

– High resolution images (millions of pixels)

High Dimension Data

Examples of high dimensional data:

- Multilingual News Stories
  (vocabulary of hundreds of thousands of words)

Hilo Tribune.

KILAUEA CRATER CONTINUES IN ERUPTION

FOUR DAYS OF ACTIVITY

ANOTHER OUTBREAK OCCURS

HONOLULU GETS APPROPRIATIONS

FOR HARBOR AND LIGHTHOUSE

Македония

PRAVA BRAĆA

Истински братя

DEKLARACIJA.

Декларация

குடி அரசு

"KUDI-ARASU"

High Dimension Data

Examples of high dimensional data:

– Brain Imaging Data (100s of MBs per scan)

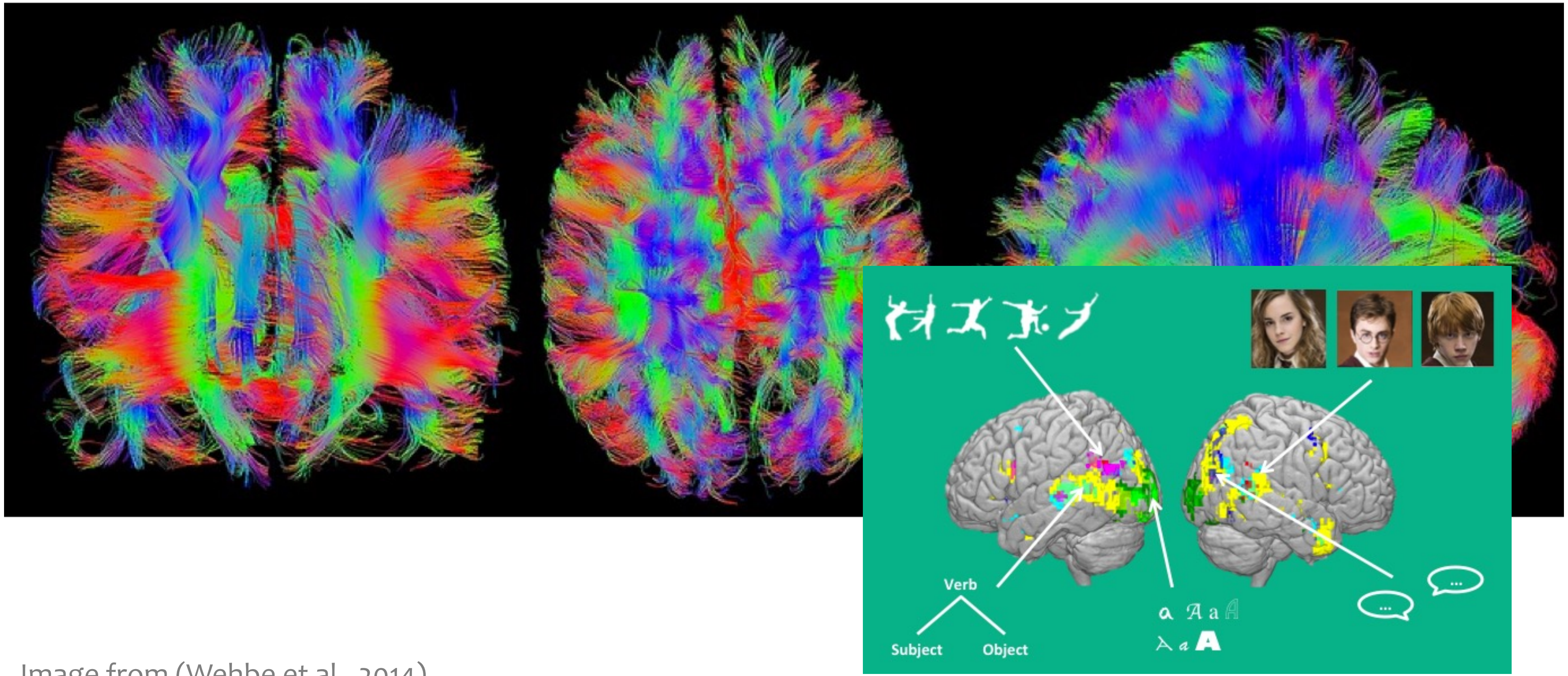


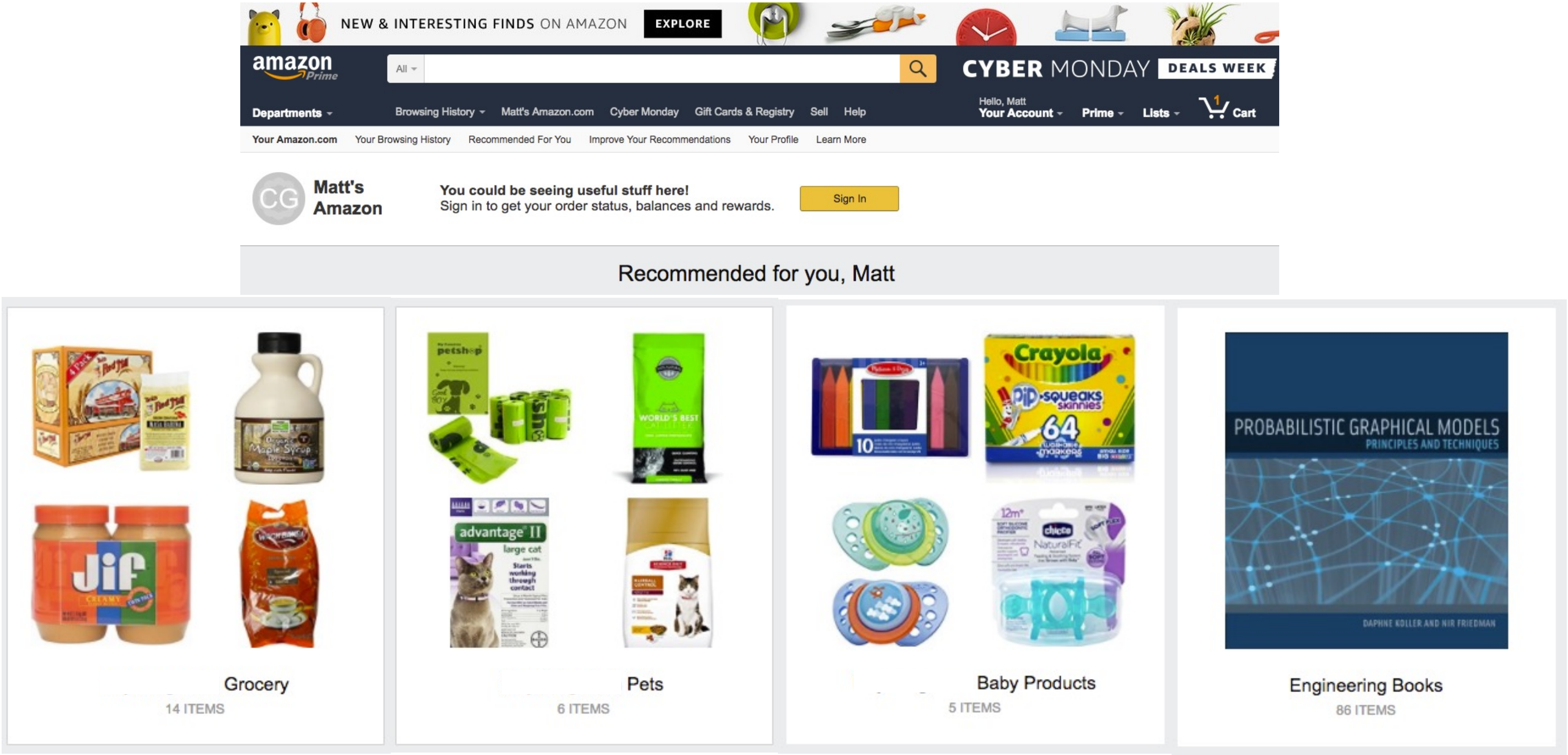
Image from (Wehbe et al., 2014)

Image from https://pixabay.com/en/brain-mrt-magnetic-resonance-imaging-1728449/

High Dimension Data

Examples of high dimensional data:

– Customer Purchase Data



Learning Representations

Dimensionality Reduction Algorithms:

Powerful (often unsupervised) learning techniques for extracting hidden (potentially lower dimensional) structure from high dimensional datasets.

Examples:

PCA, Kernel PCA, ICA, CCA, t-SNE, Autoencoders, Matrix Factorization

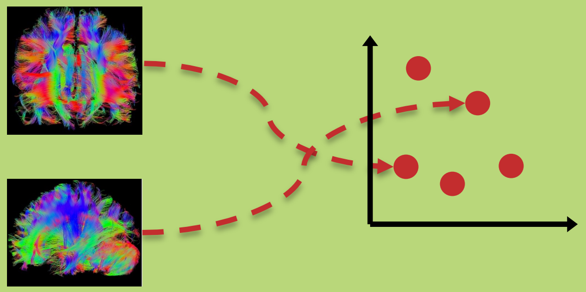
Useful for:

- Visualization
- More efficient use of resources (e.g., time, memory, communication)
- Statistical: fewer dimensions → better generalization
- Noise removal (improving data quality)

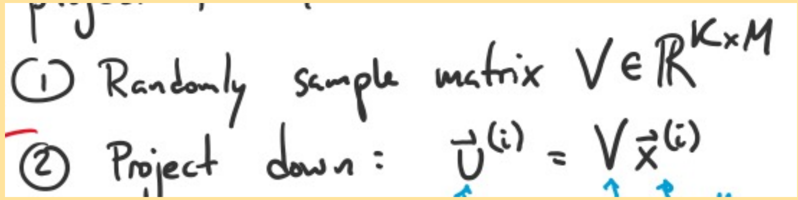
Slide adapted from Nina Balcan

This section in one slide…

1. Dimensionality reduction:



2. Random Projection:



3. Definition of PCA:

Choose the matrix V that either…

1. minimizes reconstruction error
2. consists of the K eigenvectors with largest eigenvalue

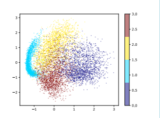
The above are equivalent definitions.

4. Algorithm for PCA:

The option we'll focus on:

Run Singular Value Decomposition (SVD) to obtain all the eigenvectors. Keep just the top-K to form V. Play some tricks to keep things efficient.

5. An Example



DIMENSIONALITY REDUCTION BY RANDOM PROJECTION

Random Projection

Goal: project from M-dimensions down to K-dimensions

Data:

$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ where $\mathbf{x}^{(i)} \in \mathbb{R}^M$

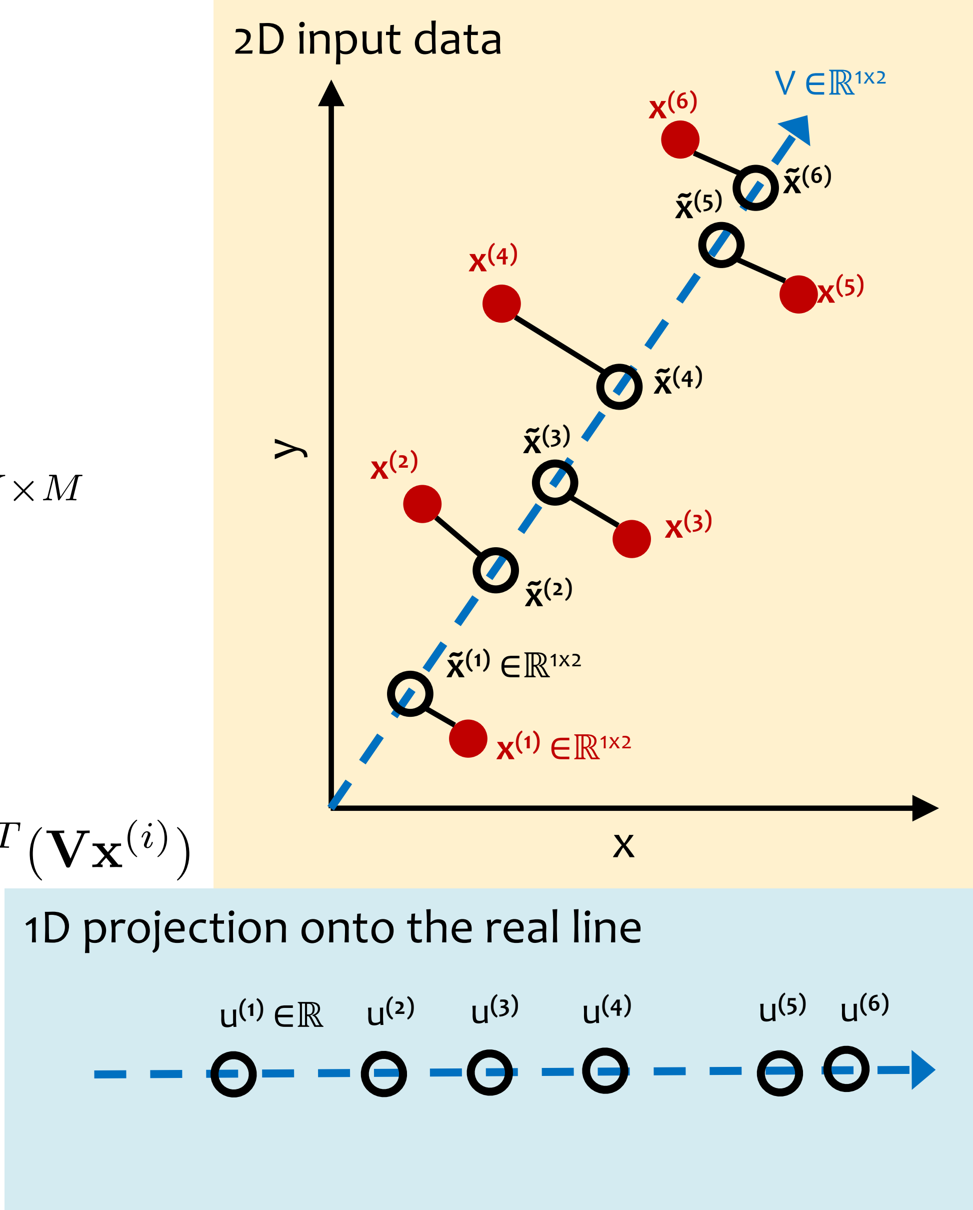
Algorithm:

1. Randomly sample matrix: $\mathbf{V} \in \mathbb{R}^{K \times M}$
   $V_{km} \sim \text{Gaussian}(0, 1)$

2. Project down: $\underbrace{\mathbf{u}^{(i)}}_{K \times 1} = \underbrace{\mathbf{V}}_{K \times M} \underbrace{\mathbf{x}^{(i)}}_{M \times 1}$

3. Project up: $\underbrace{\tilde{\mathbf{x}}^{(i)}}_{M \times 1} = \underbrace{\mathbf{V}^T}_{M \times K} \underbrace{\mathbf{u}^{(i)}}_{K \times 1} = \mathbf{V}^T(\mathbf{V}\mathbf{x}^{(i)})$

Example: 2D to 1D



Random Projection

<u>Goal</u>: project from M-dimensions down to K-dimensions

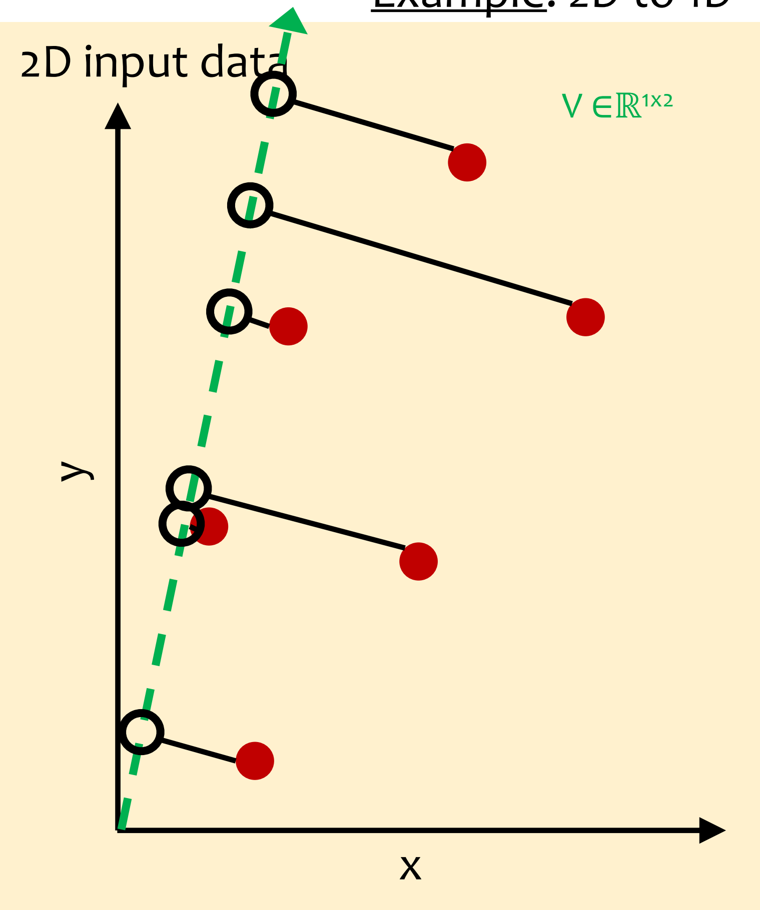
<u>Data</u>:

$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ where $\mathbf{x}^{(i)} \in \mathbb{R}^M$

<u>Algorithm</u>:

1. Randomly sample matrix: $\mathbf{V} \in \mathbb{R}^{K \times M}$
   $V_{km} \sim \text{Gaussian}(0, 1)$
2. Project down: $\underbrace{\mathbf{u}^{(i)}}_{K \times 1} = \underbrace{\mathbf{V}}_{K \times M} \underbrace{\mathbf{x}^{(i)}}_{M \times 1}$
3. Project up: $\underbrace{\mathbf{x}^{(i)}}_{M \times 1} = \underbrace{\mathbf{V}^T}_{M \times K} \underbrace{\mathbf{u}^{(i)}}_{K \times 1} = \mathbf{V}^T(\mathbf{V}\mathbf{x}^{(i)})$

<u>Example</u>: 2D to 1D



Problem: a random projection might give us a poor low dimensional representation of the data

Johnson-Lindenstrauss Lemma

Q: But how could we ever hope to preserve any useful information by randomly projecting into a low-dimensional space?

A: Even random projection enjoys some surprisingly impressive properties. In fact, a standard of the J-L lemma starts by assuming we have a random linear projection obtained by sampling each matrix entry from a Gaussian(0,1).

An Elementary Proof of a Theorem of Johnson and Lindenstrauss

Sanjoy Dasgupta,[1] Anupam Gupta[2]

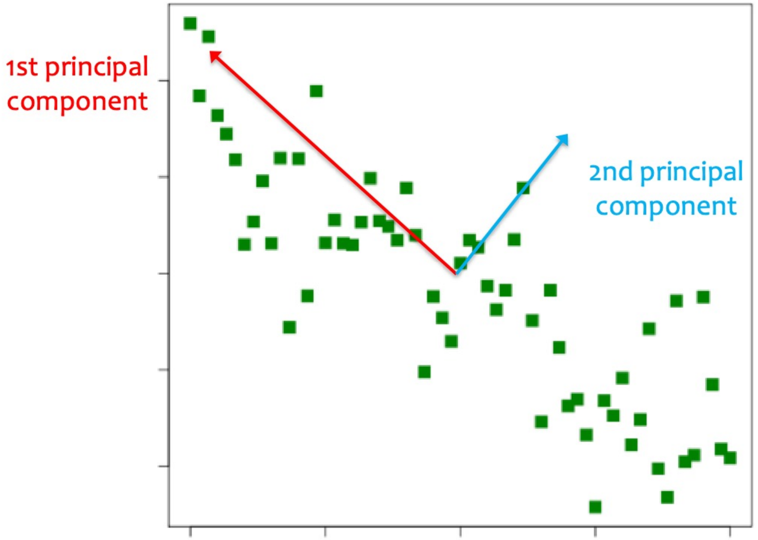
ABSTRACT: A result of Johnson and Lindenstrauss [13] shows that a set of n points in high dimensional Euclidean space can be mapped into an $O(\log n/\epsilon^2)$-dimensional Euclidean space such that the distance between any two points changes by only a factor of $(1 \pm \epsilon)$. In this note, we prove this theorem using elementary probabilistic techniques. © 2003 Wiley Periodicals, Inc. Random Struct. Alg., 22: 60–65, 2002

http://www.cs.cmu.edu/~anupamg/papers/jl.pdf

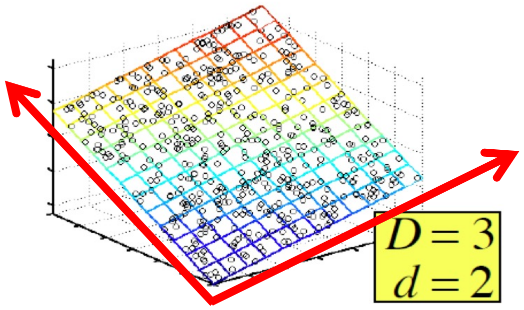
DEFINITION OF PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis (PCA)

- **Assumption:** the data lies on a low K-dimensional linear subspace
- **Goal:** identify the axes of that subspace, and project each point onto hyperplane
- **Algorithm:** find the K eigenvectors with largest eigenvalue using classic matrix decomposition tools



PCA Example: 2D Gaussian Data



https://commons.wikimedia.org/wiki/File:Scatter_diagram_for_quality_characteristic_XXX.svg

Data for PCA

$$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$$

$$\mathbf{x}^{(i)} \in \mathbb{R}^M$$

$$\mathbf{X} = \begin{bmatrix} (\mathbf{x}^{(1)})^T \\ (\mathbf{x}^{(2)})^T \\ \vdots \\ (\mathbf{x}^{(N)})^T \end{bmatrix}$$

We assume the data is **centered**, i.e. the **sample mean** is zero

$$\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}^{(i)} = \mathbf{0}$$

Q: What if your data is **not** centered?

A: Subtract off the sample mean

$$\tilde{\mathbf{x}}^{(i)} = \mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}, \forall i$$

Background: Sample Variance

Suppose we have a sequence of random samples $\{x^{(1)}, \ldots, x^{(N)}\}$ from a random variable X.

The (biased) **sample variance** $\hat{\sigma}^2$ is given by:

$$\hat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}(x^{(i)} - \hat{\mu})^2$$

where $\hat{\mu}$ is the sample mean.

Sample Covariance Matrix

The **sample covariance matrix** $\boldsymbol{\Sigma} \in \mathbb{R}^{M \times M}$ is given by:

$$\Sigma_{jk} = \frac{1}{N} \sum_{i=1}^{N} (x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k)$$

Since the data matrix is centered, we rewrite as:

$$\boldsymbol{\Sigma} = \frac{1}{N} \mathbf{X}^T \mathbf{X}$$

$$\mathbf{X} = \begin{bmatrix} (\mathbf{x}^{(1)})^T \\ (\mathbf{x}^{(2)})^T \\ \vdots \\ (\mathbf{x}^{(N)})^T \end{bmatrix}$$

Principal Component Analysis (PCA)

Linear Projection:
Given KxM matrix **V,** and Mx1 vector $\mathbf{x}^{(i)}$ we obtain the Kx1 projection $\mathbf{u}^{(i)}$ by:

$$\mathbf{u}^{(i)} = \mathbf{V}\,\mathbf{x}^{(i)}$$

$$\mathbf{V} = \begin{bmatrix} - & \mathbf{v}_1^T & - \\ - & \mathbf{v}_2^T & - \\ & \vdots & \\ - & \mathbf{v}_K^T & - \end{bmatrix}$$

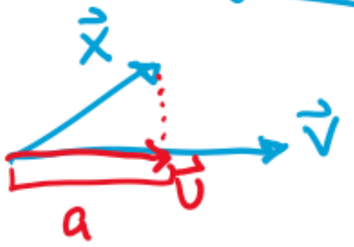
Definition of PCA:
PCA repeatedly chooses a next vector $\mathbf{v}_j$ that minimizes the reconstruction error s.t. $\mathbf{v}_j$ is orthogonal to $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{j-1}$.

Vector $\mathbf{v}_j$ is called the **jth principal component.**

Notice: Two vectors **a** and **b** are **orthogonal** if $\mathbf{a}^T\mathbf{b} = 0$.
➔the K-dimensions in PCA are uncorrelated

Vector Projection

Recall: Projection



length of projection of $\vec{x}$ onto $\vec{v}$

$a = \vec{v}^T\vec{x}$ if $\|v\|_2 = 1$

$a = \dfrac{\vec{v}^T\vec{x}}{\|v\|_2}$ otherwise

projection of $\vec{x}$ onto $\vec{v}$

$\vec{u} = a\vec{v} = (\vec{v}^T\vec{x})\vec{v}$ if $\|v\|_2 = 1$

$\vec{u} = a\vec{v} = \dfrac{(\vec{v}^T\vec{x})}{\|\vec{v}\|_2^2}\vec{v}$ otherwise

Objectives for PCA

Minimize the Reconstruction Error

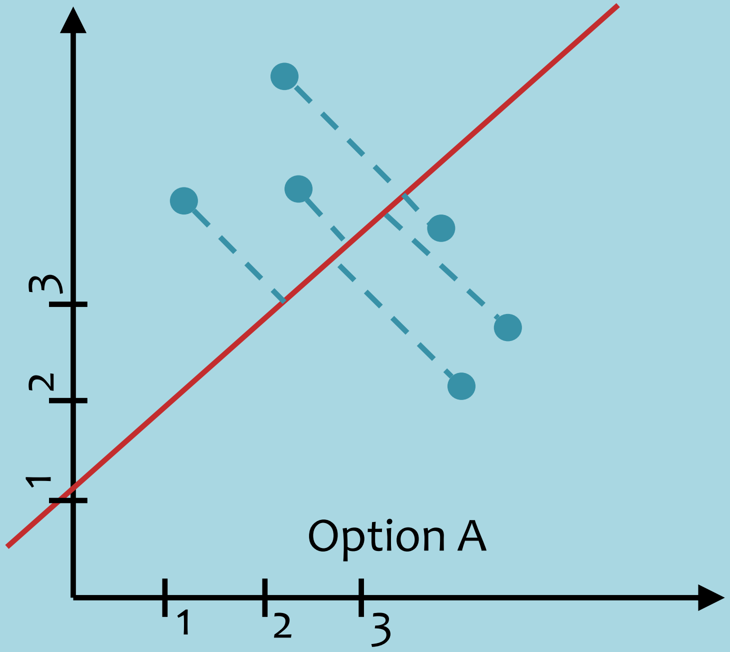
Maximize the Variance

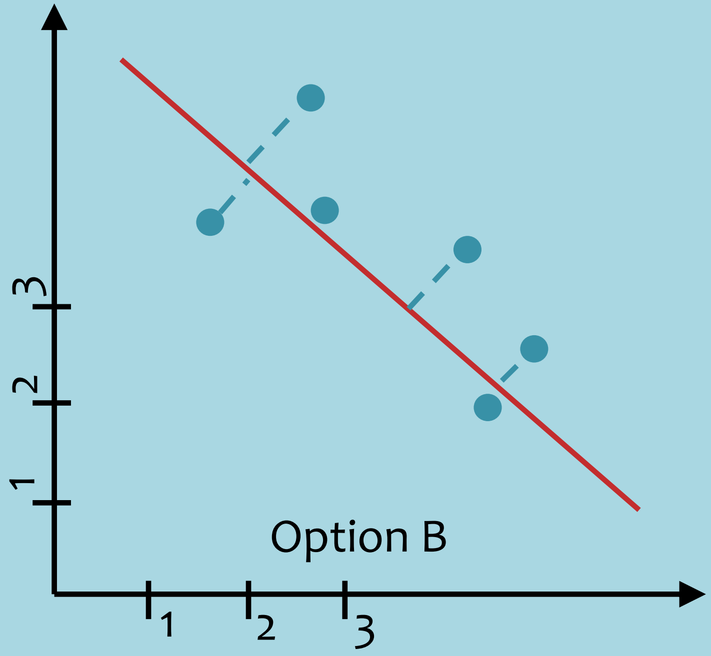
Projection Example

Question:
Below are two plots of the same dataset D. Consider the two projections shown.

1. Which maximizes the variance?
2. Which minimizes the reconstruction error?

Answer:





PCA Objective Functions

What is the first principal component $\mathbf{v}_1$ chosen by PCA?

Option 1: The vector that *minimizes* the **reconstruction error**

$$\mathbf{v}_1 = \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} ||\mathbf{x}^{(i)} - (\mathbf{v}^T \mathbf{x}^{(i)})\mathbf{v}||^2$$

Option 2: The vector that *maximizes* the **variance**

$$\mathbf{v}_1 = \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^{N} (\mathbf{v}^T \mathbf{x}^{(i)})^2$$

Equivalence of Maximizing Variance and Minimizing Reconstruction Error

PCA

Claim: Minimizing the reconstruction error is equivalent to maximizing the variance.

Proof: First, note that:

$$||\mathbf{x}^{(i)} - (\mathbf{v}^T\mathbf{x}^{(i)})\mathbf{v}||^2 = ||\mathbf{x}^{(i)}||^2 - (\mathbf{v}^T\mathbf{x}^{(i)})^2 \tag{1}$$

since $\mathbf{v}^T\mathbf{v} = ||\mathbf{v}||^2 = 1$.

Substituting into the minimization problem, and removing the extraneous terms, we obtain the maximization problem.

$$\mathbf{v}^* = \operatorname*{argmin}_{\mathbf{v}:||\mathbf{v}||^2=1} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)} - (\mathbf{v}^T\mathbf{x}^{(i)})\mathbf{v}||^2 \tag{2}$$

$$= \operatorname*{argmin}_{\mathbf{v}:||\mathbf{v}||^2=1} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)}||^2 - (\mathbf{v}^T\mathbf{x}^{(i)})^2 \tag{3}$$

$$= \operatorname*{argmax}_{\mathbf{v}:||\mathbf{v}||^2=1} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{v}^T\mathbf{x}^{(i)})^2 \tag{4}$$

PCA Objective Functions

What is the first principal component $\mathbf{v}_1$ chosen by PCA?

Option 1: The vector that *minimizes* the **reconstruction error**

$$\mathbf{v}_1 = \operatorname*{argmin}_{\mathbf{v}:||\mathbf{v}||^2=1} \frac{1}{N} \sum_{i=1}^{N} ||\mathbf{x}^{(i)} - (\mathbf{v}^T \mathbf{x}^{(i)})\mathbf{v}||^2$$

Option 2: The vector that *maximizes* the **variance**

$$\mathbf{v}_1 = \operatorname*{argmax}_{\mathbf{v}:||\mathbf{v}||^2=1} \frac{1}{N} \sum_{i=1}^{N} (\mathbf{v}^T \mathbf{x}^{(i)})^2$$

Question:
Why can't we just use gradient descent to find the minimum of the PCA optimization problem?

Answer:

Principal Component Analysis (PCA)

Linear Projection:
Given KxM matrix **V**, and Mx1 vector $\mathbf{x}^{(i)}$ we obtain the Kx1 projection $\mathbf{u}^{(i)}$ by:

$$\mathbf{u}^{(i)} = \mathbf{V}\,\mathbf{x}^{(i)}$$

$$\mathbf{V} = \begin{bmatrix} - & \mathbf{v}_1^T & - \\ - & \mathbf{v}_2^T & - \\ & \vdots & \\ - & \mathbf{v}_K^T & - \end{bmatrix}$$

Question:
Why can't we just use gradient descent to find the minimum of the PCA optimization problem?

Definition of PCA:
PCA repeatedly chooses a next vector $\mathbf{v}_j$ that minimizes the reconstruction error s.t. $\mathbf{v}_j$ is orthogonal to $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{j-1}$.

Vector $\mathbf{v}_j$ is called the **jth principal component.**

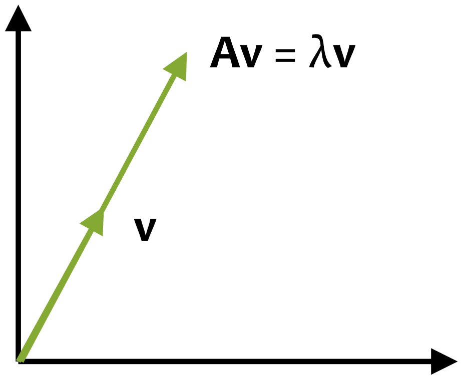
Notice: Two vectors **a** and **b** are **orthogonal** if $\mathbf{a}^T\mathbf{b} = 0$.
➔ the K-dimensions in PCA are uncorrelated

Answer:

Background: Eigenvectors & Eigenvalues

For a square matrix **A** (n x n matrix), the vector **v** (n x 1 matrix) is an **eigenvector** iff there exists **eigenvalue** λ (scalar) such that:

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



The linear transformation **A** is only stretching vector **v.**

That is, λ**v** is a *scalar multiple* of **v.**

Background: Eigenvectors & Eigenvalues

Fact #1: The eigenvectors of a **symmetric matrix** are **orthogonal** to each other.

Fact #2: The **covariance matrix $\boldsymbol{\Sigma}$** is **symmetric**.

The First Principal Component

PCA

Claim: The vector that maximizes the variances is the eigenvector of $\mathbf{\Sigma}$ with largest eigenvalue.

Proof Sketch: To find the first principal component, we wish to solve the following constrained optimization problem (variance minimization).

$$\mathbf{v}_1 = \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\operatorname{argmax}} \ \mathbf{v}^T \mathbf{\Sigma} \mathbf{v} \tag{1}$$

So we turn to the method of Lagrange multipliers. The Lagrangian is:

$$\mathcal{L}(\mathbf{v}, \lambda) = \mathbf{v}^T \mathbf{\Sigma} \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - 1) \tag{2}$$

Taking the derivative of the Lagrangian and setting to zero gives:

$$\frac{d}{d\mathbf{v}} \left(\mathbf{v}^T \mathbf{\Sigma} \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - 1) \right) = 0 \tag{3}$$

$$\mathbf{\Sigma} \mathbf{v} - \lambda \mathbf{v} = 0 \tag{4}$$

$$\mathbf{\Sigma} \mathbf{v} = \lambda \mathbf{v} \tag{5}$$

Recall: For a square matrix $\mathbf{A}$, the vector $\mathbf{v}$ is an **eigenvector** iff there exists **eigenvalue** λ such that:

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{v} \tag{6}$$

Rewriting the objective of the maximization shows that not only will the optimal vector $\mathbf{v}_1$ be an eigenvector, it will be one with maximal eigenvalue.

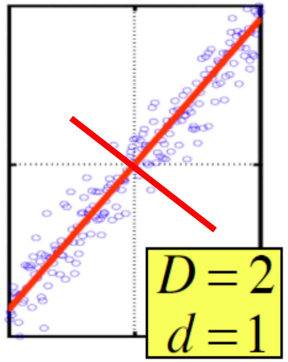
$$\begin{aligned} \mathbf{v}^T \mathbf{\Sigma} \mathbf{v} &= \mathbf{v}^T \lambda \mathbf{v} && (7) \\ &= \lambda \mathbf{v}^T \mathbf{v} && (8) \\ &= \lambda ||\mathbf{v}||^2 && (9) \\ &= \lambda && (10) \end{aligned}$$

Principal Component Analysis (PCA)

$(\mathrm{X}\,\mathrm{X}^{\mathrm{T}})\mathrm{v} = \lambda\mathrm{v}$, so v (the first PC) is the eigenvector of sample correlation/covariance matrix $X\,X^T$

Sample variance of projection $\mathrm{v}^T X\,X^T \mathrm{v} = \lambda \mathrm{v}^T \mathrm{v} = \lambda$

Thus, the eigenvalue λ denotes the amount of variability captured along that dimension (aka amount of energy along that dimension).



Eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots$

- The 1st PC v_1 is the the eigenvector of the sample covariance matrix $X\,X^T$ associated with the largest eigenvalue
- The 2nd PC v_2 is the the eigenvector of the sample covariance matrix $X\,X^T$ associated with the second largest eigenvalue
- And so on …

Slide from Nina Balcan

ALGORITHMS FOR PCA

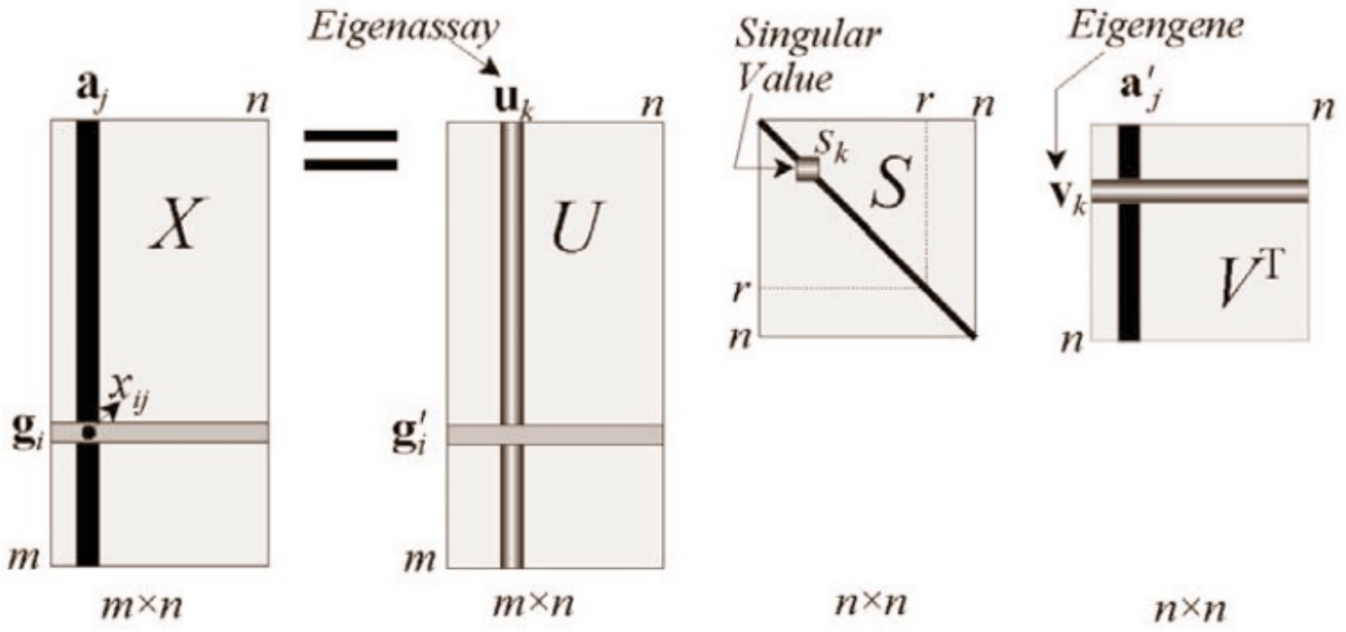
Algorithms for PCA

How do we find principal components (i.e. eigenvectors)?

- Power iteration (aka. Von Mises iteration)
  - finds **each** principal component **one at a time** in order
- Singular Value Decomposition (SVD)
  - finds **all** the principal components **at once**
  - two options:
    - Option A: run SVD on $X^T X$
    - Option B: run SVD on X
      (not obvious why Option B should work…)
- Stochastic Methods (approximate)
  - **very efficient** for high dimensional datasets with lots of points

SVD

$$X = USV^{T}$$



Data X, one row per data point

US gives coordinates of rows of X in the space of principle components

S is diagonal, $S_k > S_{k+1}$, S_k^2 is kth largest eigenvalue

Rows of V^T are unit length eigenvectors of X^TX

If cols of X have zero mean, then $X^TX = c\ \Sigma$ and eigenvects are the Principle Components

[from Wall et al., 2003]

Slide from Tom Mitchell

Singular Value Decomposition

To generate principle components:

- Subtract mean $\bar{\mathrm{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathrm{x}^n$ from each data point, to create zero-centered data
- Create matrix X with one row vector per (zero centered) data point
- Solve SVD: $X = USV^T$
- Output Principle components: columns of V (= rows of V^T)
  - Eigenvectors in V are sorted from largest to smallest eigenvalues
  - S is diagonal, with s_k^2 giving eigenvalue for kth eigenvector

Slide from Tom Mitchell

Singular Value Decomposition

To project a point (column vector x) into PC coordinates:

$V^T x$

If x_i is ith row of data matrix X, then

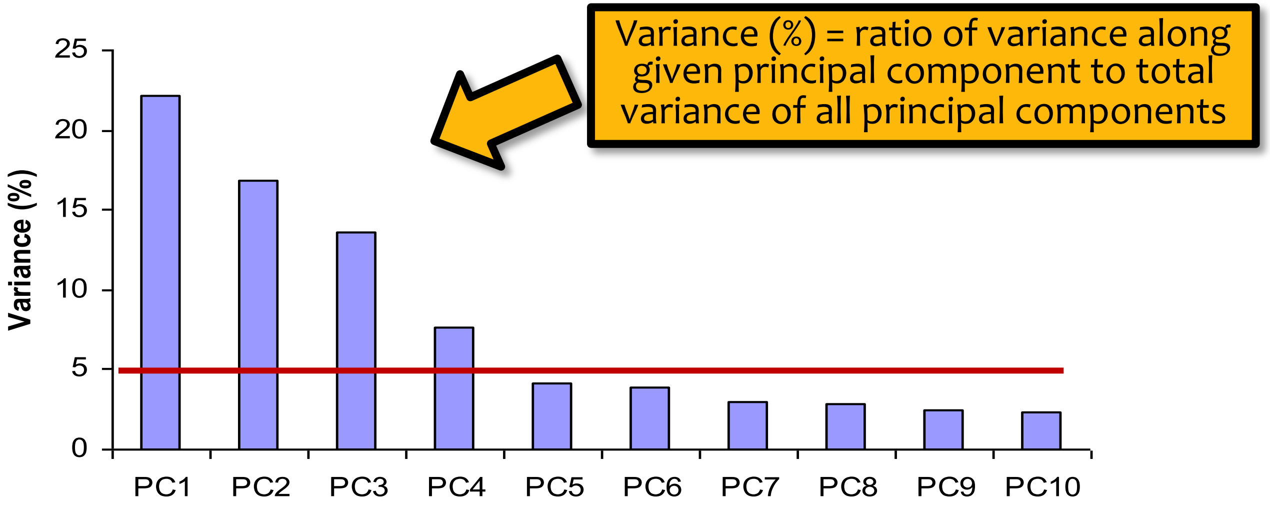
- (ith row of US) $= V^T x_i^T$
- $(US)^T = V^T X^T$

To project a column vector x to M dim Principle Components subspace, take just the first M coordinates of $V^T x$

Slide from Tom Mitchell

How Many PCs?

- For M original dimensions, sample covariance matrix is MxM, and has up to M eigenvectors. So M principal components (PCs).
- Where does dimensionality reduction come from?
  Can *ignore* the components of lesser significance.



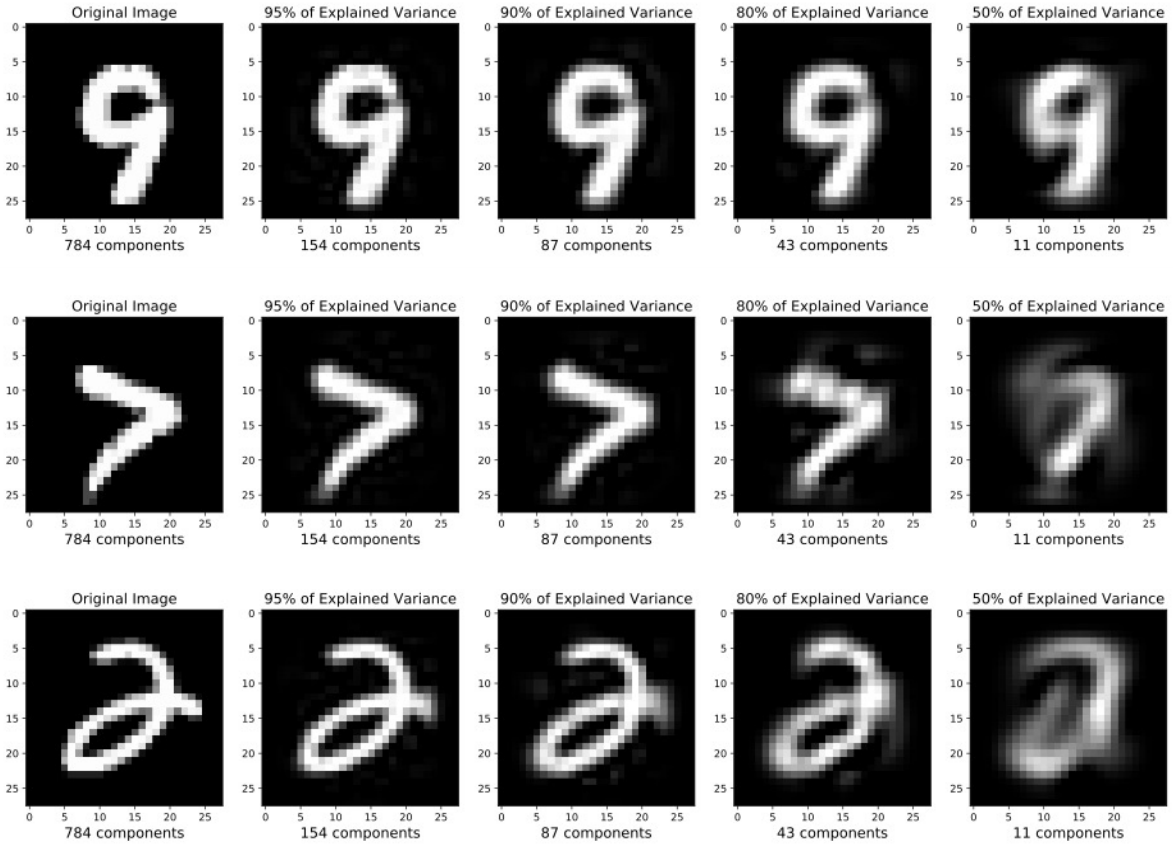
- You do lose some information, but if the eigenvalues are small, you don't lose much
  - M dimensions in original data
  - calculate M eigenvectors and eigenvalues
  - choose only the first D eigenvectors, based on their eigenvalues
  - final data set has only D dimensions

PCA EXAMPLES

Projecting MNIST digits

Task Setting:

1. Take each 28x28 image of a digit (i.e. a vector $\mathbf{x}^{(i)}$ of length 784) and project it down to K components (i.e. a vector $\mathbf{u}^{(i)}$)
2. Report percent of variance explained for K components
3. Then project back up to 28x28 image (i.e. a vector $\tilde{\mathbf{x}}^{(i)}$ of length 784) to visualize how much information was preserved



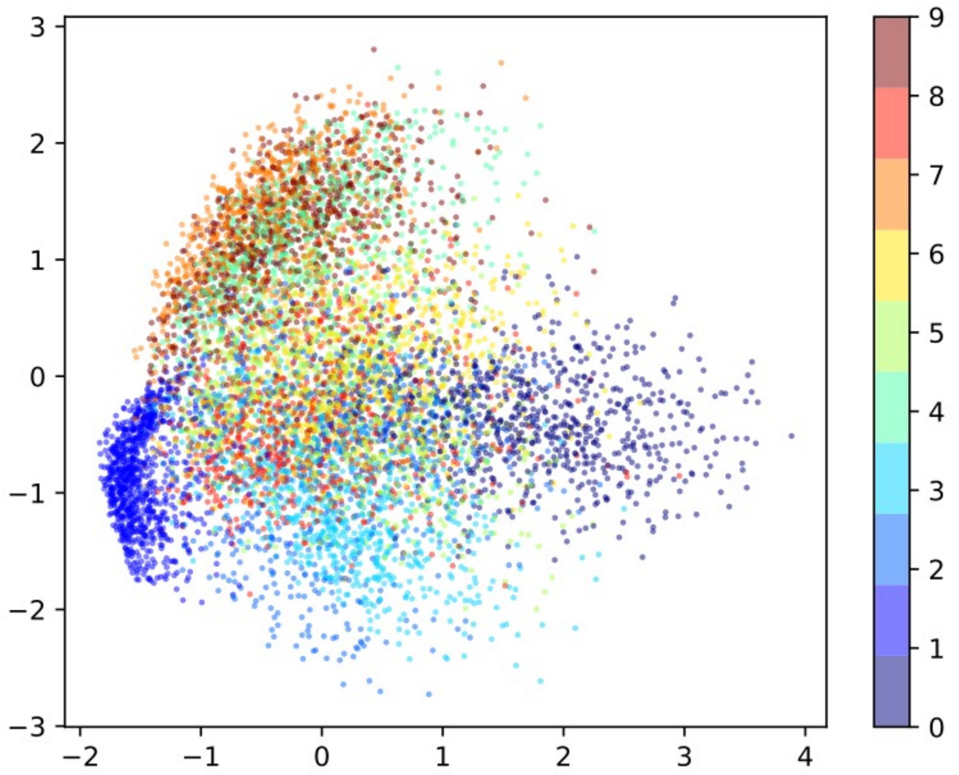
Takeaway: Using fewer principal components K leads to higher reconstruction error.

But even a small number (say 43) still preserves a lot of information about the original image.

Projecting MNIST digits

Task Setting:

1. Take each 28x28 image of a digit (i.e. a vector $\mathbf{x}^{(i)}$ of length 784) and project it down to K=2 components (i.e. a vector $\mathbf{u}^{(i)}$)
2. Plot the 2 dimensional points $\mathbf{u}^{(i)}$ and label with the (unknown to PCA) label $y^{(i)}$ as the color
3. Here we look at all ten digits 0 - 9

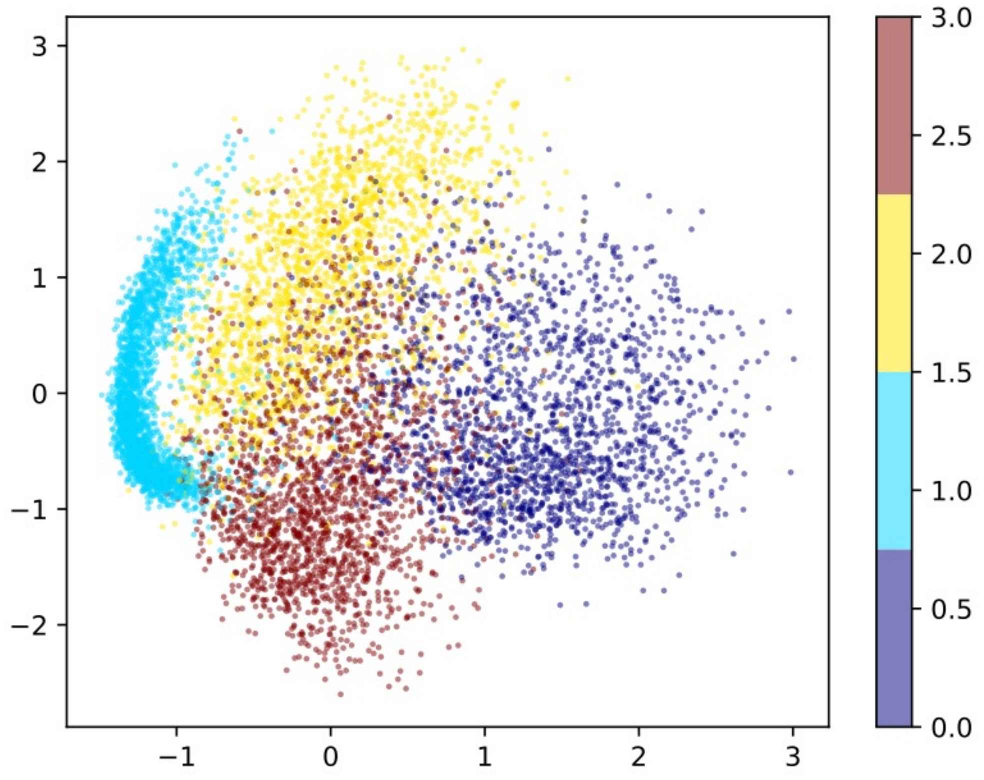


Takeaway: Even with a tiny number of principal components K=2, PCA learns a representation that captures the *latent* information about the type of digit

Projecting MNIST digits

Task Setting:

1. Take each 28x28 image of a digit (i.e. a vector $\mathbf{x}^{(i)}$ of length 784) and project it down to K=2 components (i.e. a vector $\mathbf{u}^{(i)}$)
2. Plot the 2 dimensional points $\mathbf{u}^{(i)}$ and label with the (unknown to PCA) label $y^{(i)}$ as the color
3. Here we look at just four digits 0, 1, 2, 3



Takeaway: Even with a tiny number of principal components K=2, PCA learns a representation that captures the *latent* information about the type of digit

Learning Objectives

Dimensionality Reduction / PCA

You should be able to…

1. Define the sample mean, sample variance, and sample covariance of a vector-valued dataset
2. Identify examples of high dimensional data and common use cases for dimensionality reduction
3. Draw the principal components of a given toy dataset
4. Establish the equivalence of minimization of reconstruction error with maximization of variance
5. Given a set of principal components, project from high to low dimensional space and do the reverse to produce a reconstruction
6. Explain the connection between PCA, eigenvectors, eigenvalues, and covariance matrix
7. Use common methods in linear algebra to obtain the principal components