10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

PCA

Matt Gormley
Lecture 26
April 20, 2020

Reminders

- **Homework 8: Reinforcement Learning**
  - Out: Fri, Apr 10
  - Due: Wed, Apr 22 at 11:59pm
- **Homework 9: Learning Paradigms**
  - Out: Wed, Apr. 22
  - Due: Wed, Apr. 29 at 11:59pm
  - Can only be submitted up to 3 days late, so we can return grades before final exam

- **Today's In-Class Poll**
  - http://poll.mlcourse.org

ML Big Picture

Learning Paradigms:

What data is available and when? What form of prediction?

- supervised learning
- unsupervised learning
- semi-supervised learning
- reinforcement learning
- active learning
- imitation learning
- domain adaptation
- online learning
- density estimation
- recommender systems
- feature learning
- manifold learning
- dimensionality reduction
- ensemble learning
- distant supervision
- hyperparameter optimization

Problem Formulation:

What is the structure of our output prediction?

boolean	Binary Classification
categorical	Multiclass Classification
ordinal	Ordinal Classification
real	Regression
ordering	Ranking
multiple discrete	Structured Prediction
multiple continuous	(e.g. dynamical systems)
both discrete & cont.	(e.g. mixed graphical models)

Application Areas

Key challenges?

NLP, Speech, Computer Vision, Robotics, Medicine, Search

Theoretical Foundations:

What principles guide learning?

- ❑ probabilistic
- ❑ information theoretic
- ❑ evolutionary search
- ❑ ML as optimization

Facets of Building ML Systems:

How to build systems that are robust, efficient, adaptive, effective?

1. *Data prep*
2. *Model selection*
3. *Training (optimization / search)*
4. *Hyperparameter tuning on validation data*
5. *(Blind) Assessment on test data*

Big Ideas in ML:

Which are the ideas driving development of the field?

- *inductive bias*
- *generalization / overfitting*
- *bias-variance decomposition*
- *generative vs. discriminative*
- *deep nets, graphical models*
- *PAC learning*
- *distant rewards*

Learning Paradigms

Paradigm	**Data**
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$
$\hookrightarrow$ Clustering	predict $\{z^{(i)}\}_{i=1}^N$ where $z^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Dimensionality Reduction	convert each $\mathbf{x}^{(i)} \in \mathbb{R}^M$ to $\mathbf{u}^{(i)} \in \mathbb{R}^K$ with $K << M$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \ldots\}$
Active Learning	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ and can query $y^{(i)} = c^*(\cdot)$ at a cost
Imitation Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \ldots\}$
Reinforcement Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}, r^{(1)}), (s^{(2)}, a^{(2)}, r^{(2)}), \ldots\}$

DIMENSIONALITY REDUCTION

PCA Outline

- **Dimensionality Reduction**
  - High-dimensional data
  - Learning (low dimensional) representations
- **Principal Component Analysis (PCA)**
  - Examples: 2D and 3D
  - Data for PCA
  - PCA Definition
  - Objective functions for PCA
  - PCA, Eigenvectors, and Eigenvalues
  - Algorithms for finding Eigenvectors / Eigenvalues
- **PCA Examples**
  - Face Recognition
  - Image Compression

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


叻報


Македония

PRAVA BRAĆA

Истински братя

Декларация


குடி அரசு

High Dimension Data

Examples of high dimensional data:

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

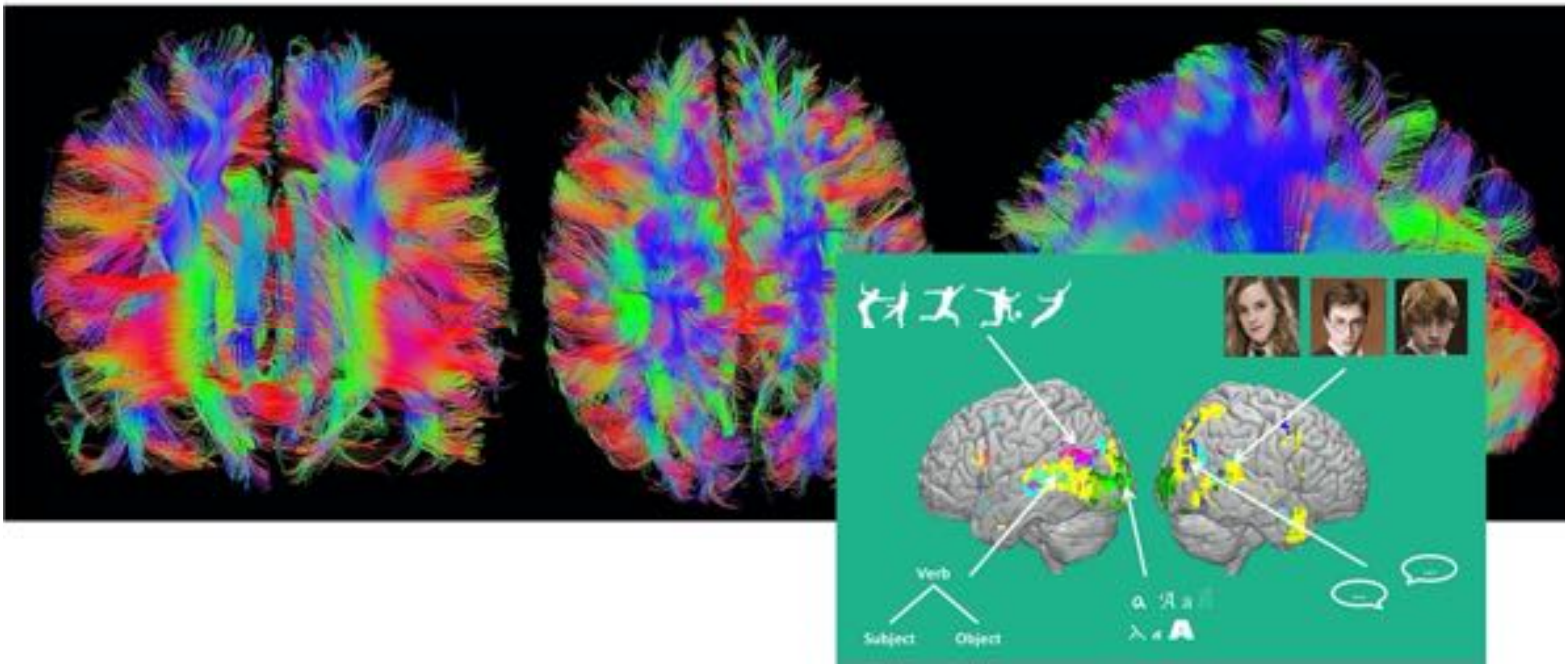


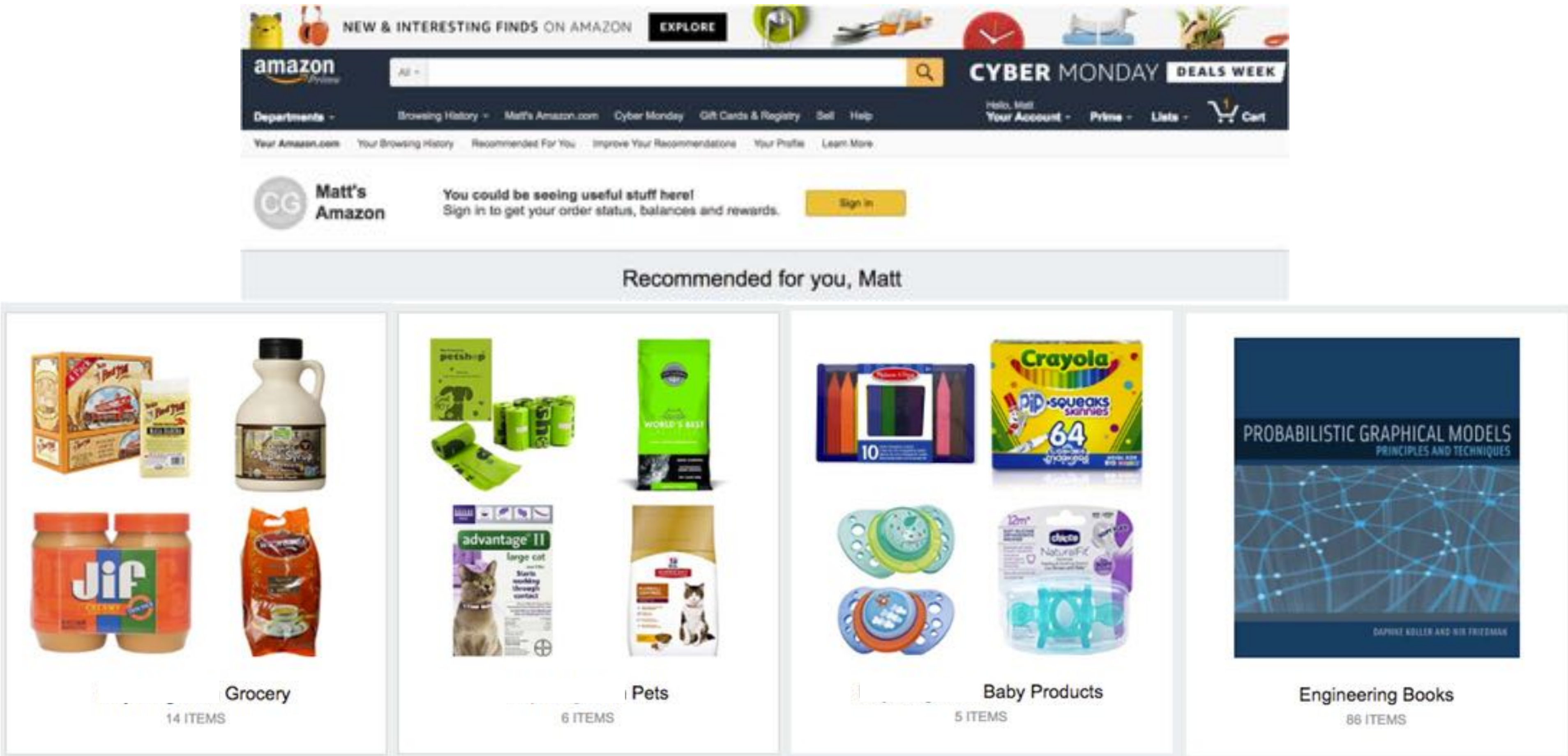
Image from (Wehbe et al., 2014)

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

High Dimension Data

Examples of high dimensional data:

– Customer Purchase Data

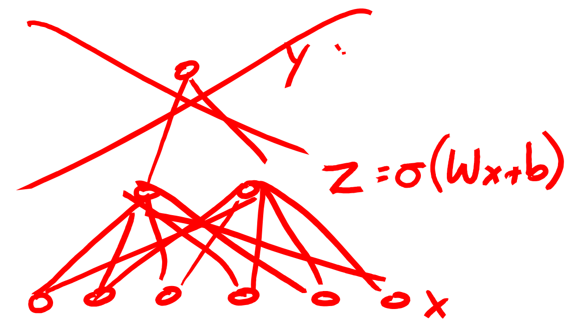


Learning Representations

PCA, Kernel PCA, ICA: Powerful unsupervised learning techniques for extracting hidden (potentially lower dimensional) structure from high dimensional datasets.

Useful for:

- Visualization
- More efficient use of resources (e.g., time, memory, communication)



- Statistical: fewer dimensions → better generalization
- Noise removal (improving data quality)
- Further processing by machine learning algorithms

Slide from Nina Balcan

Shortcut Example

https://www.youtube.com/watch?v=MlJN9pEfPfE

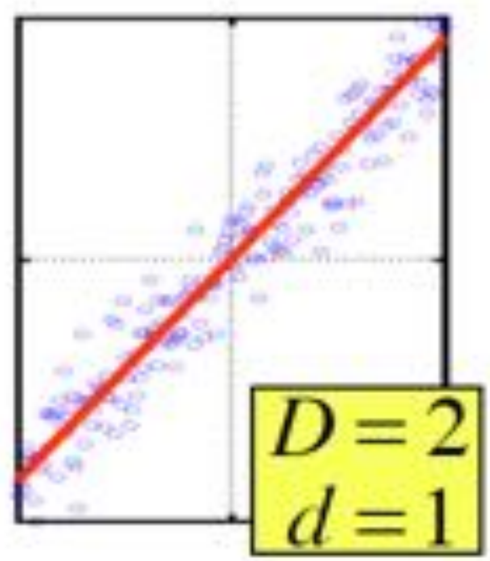
Photo from https://www.springcarnival.org/booth.shtml

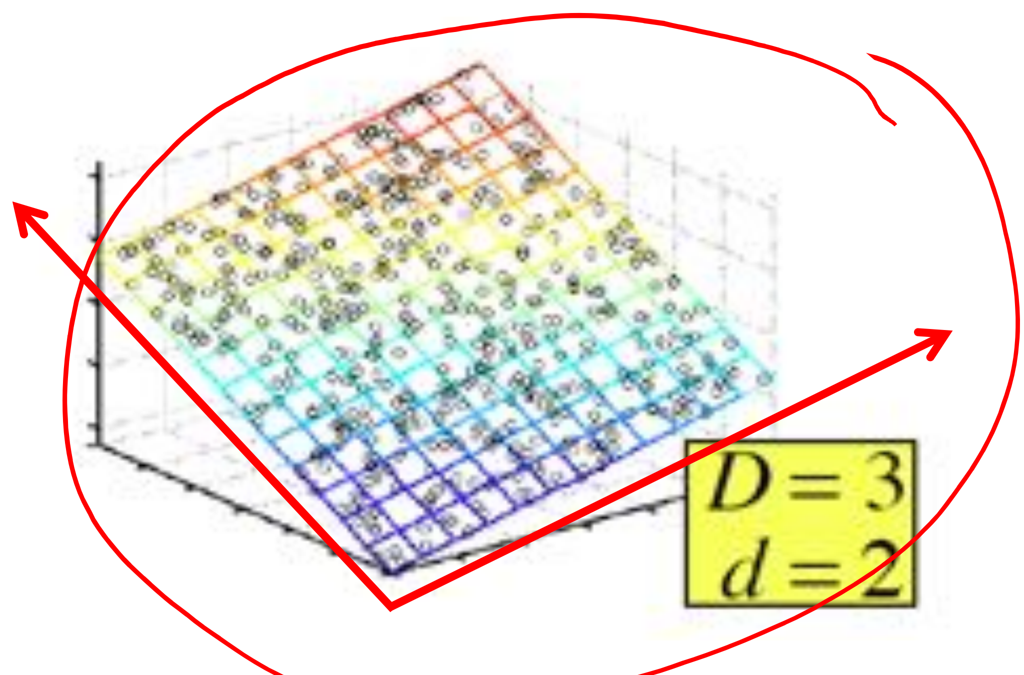
PRINCIPAL COMPONENT ANALYSIS (PCA)

PCA Outline

- **Dimensionality Reduction**
  - High-dimensional data
  - Learning (low dimensional) representations
- **Principal Component Analysis (PCA)**
  - Examples: 2D and 3D
  - Data for PCA
  - PCA Definition
  - Objective functions for PCA
  - PCA, Eigenvectors, and Eigenvalues
  - Algorithms for finding Eigenvectors / Eigenvalues
- **PCA Examples**
  - Face Recognition
  - Image Compression

Principal Component Analysis (PCA)



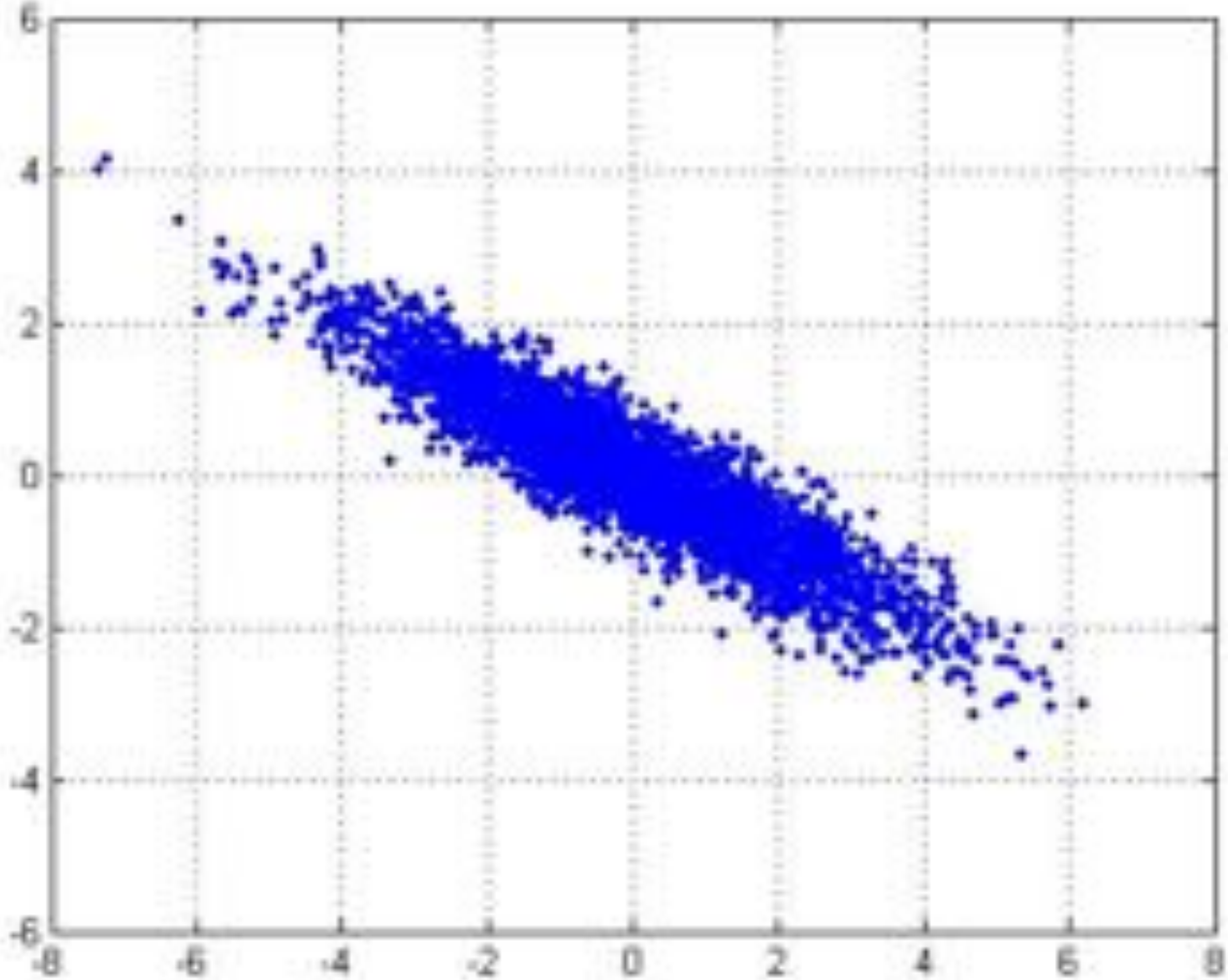


In case where data lies on or near a low d-dimensional linear subspace, axes of this subspace are an effective representation of the data.

Identifying the axes is known as Principal Components Analysis, and can be obtained by using classic matrix computation tools (Eigen or Singular Value Decomposition).

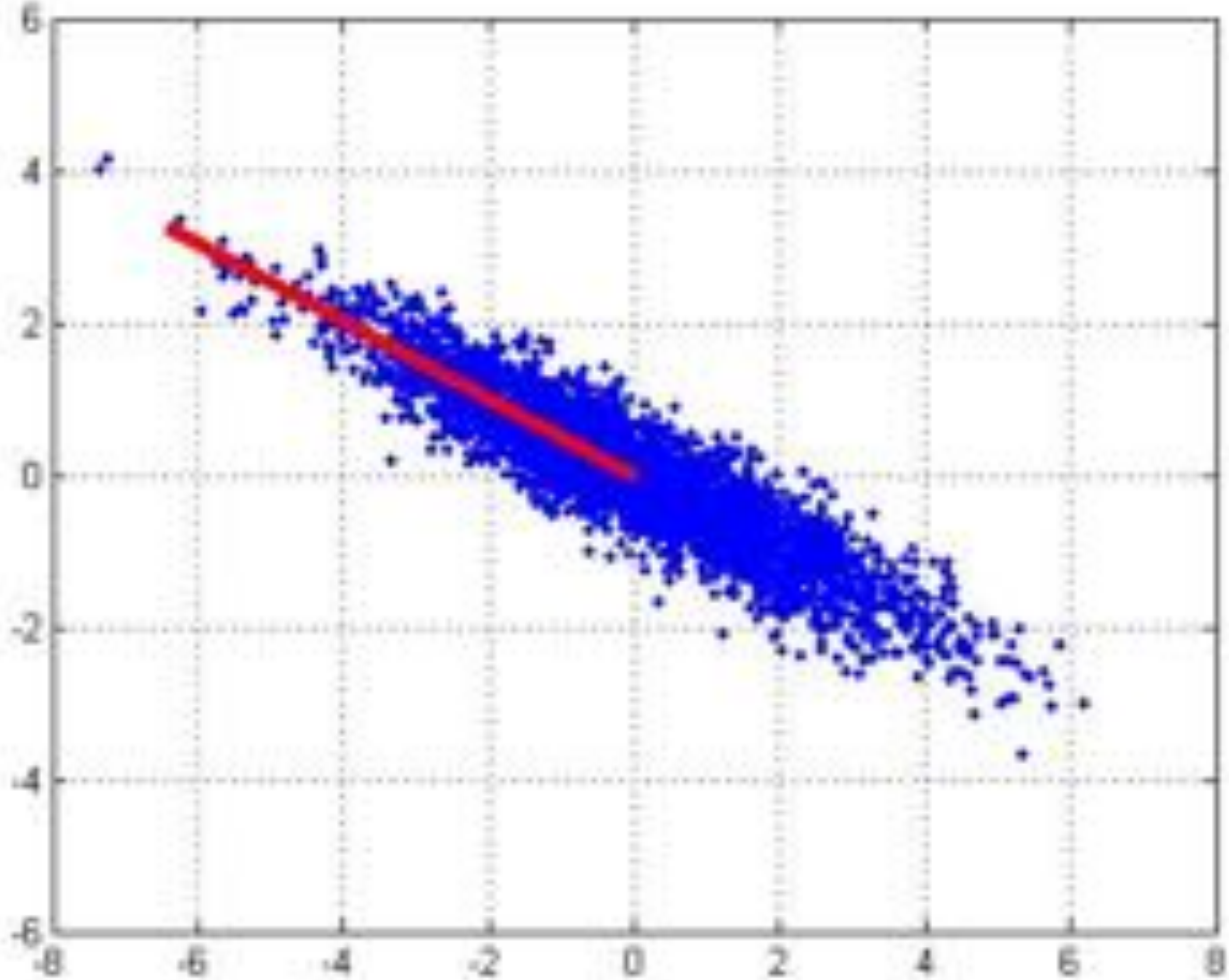
Slide from Nina Balcan

2D Gaussian dataset



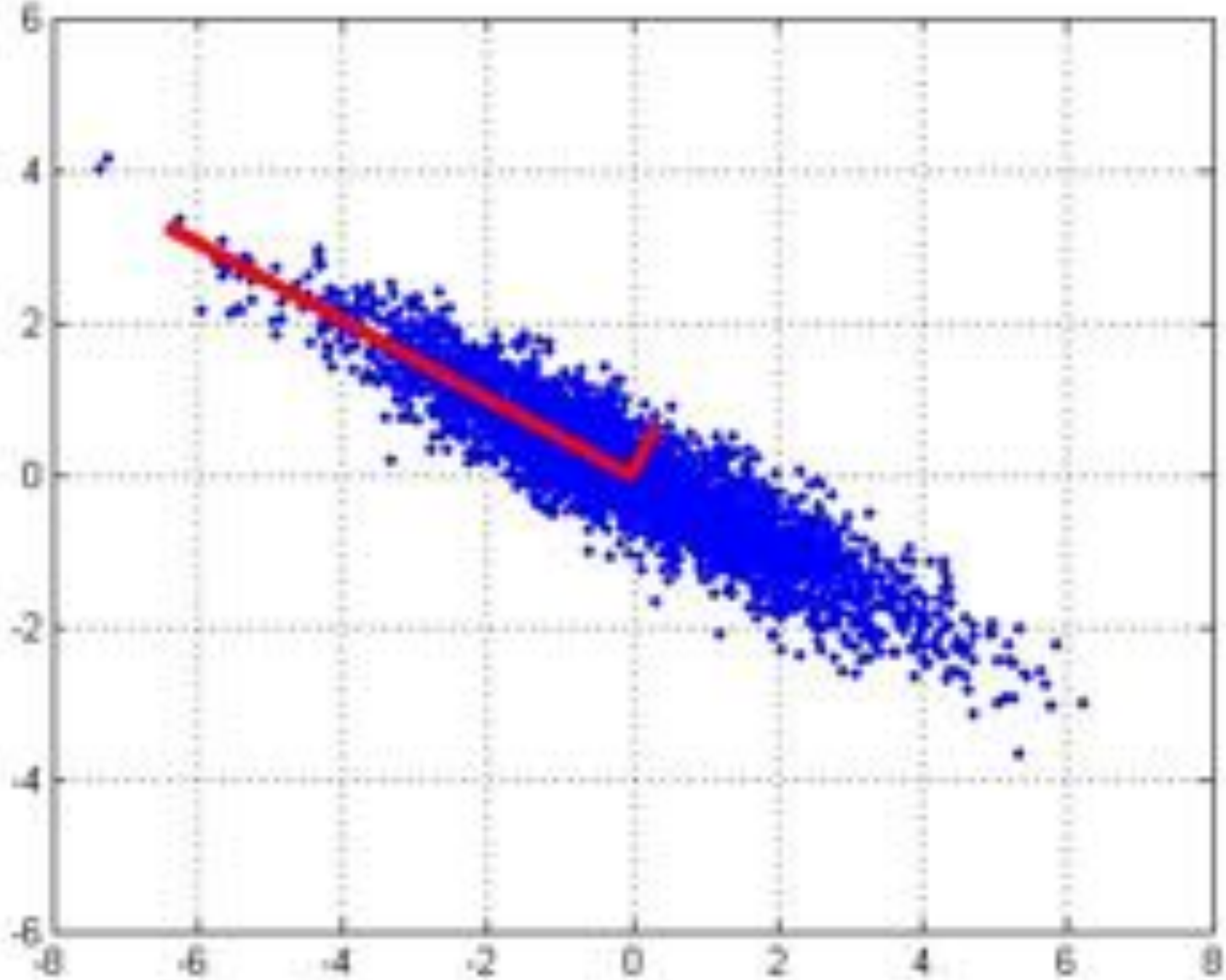
Slide from Barnabas Poczos

1st PCA axis



Slide from Barnabas Poczos

2^{nd} PCA axis



Slide from Barnabas Poczos

Data for PCA

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

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

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

We assume the data is **centered**

$(x^{(i)'} - \mu')$

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

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

A: Subtract off the sample mean

Sample Covariance Matrix

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

The sample covariance matrix is given by:

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

$\Sigma \in \mathbb{R}^{M \times M}$

Since the data matrix is centered, we rewrite as:

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

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

Principal Component Analysis (PCA)

Whiteboard

- Strawman: random linear projection
- PCA Definition
- Objective functions for PCA

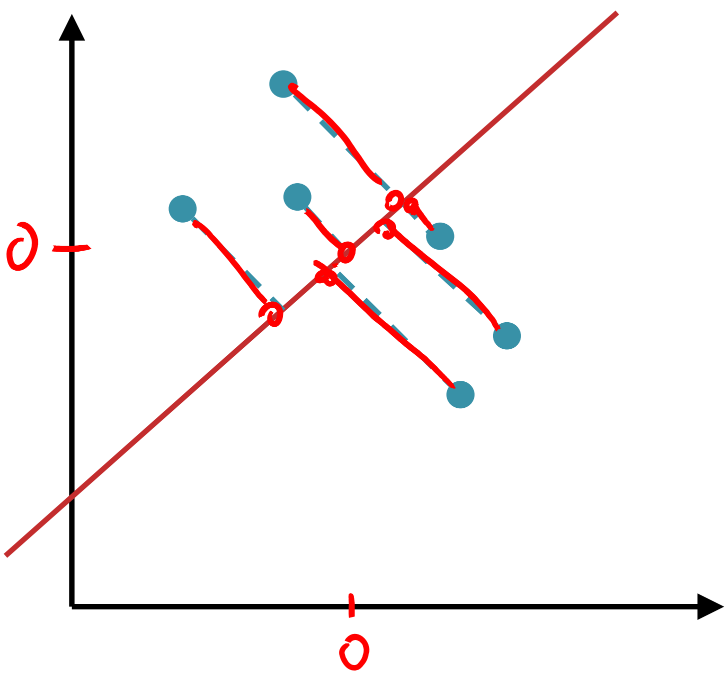
Maximizing the Variance

calamity = c

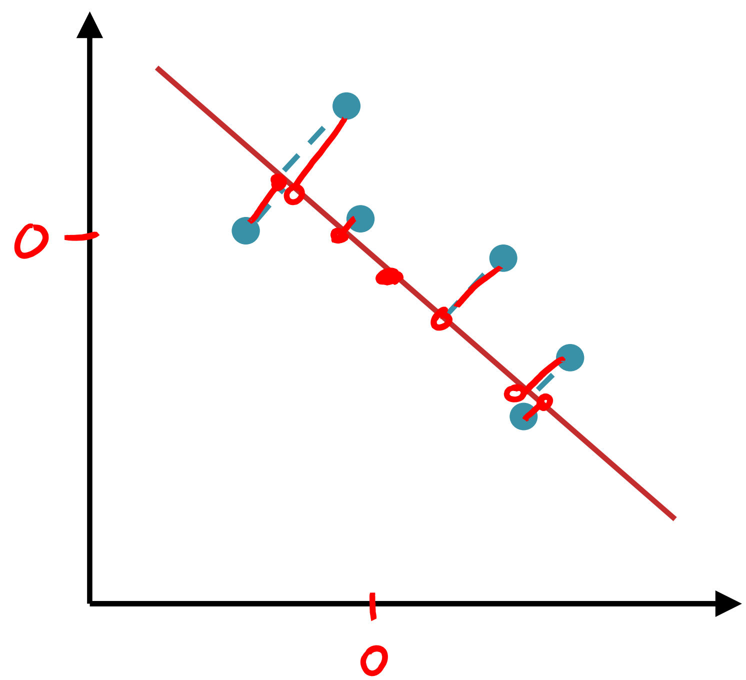
Quiz: Consider the two projections below

1. Which maximizes the variance? 45% A 55% B
2. Which minimizes the reconstruction error? 13% A 87% B

Option A



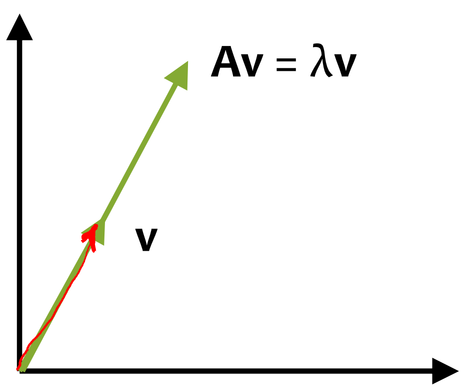
Option B



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:

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



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

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

Principal Component Analysis (PCA)

Whiteboard

– PCA, Eigenvectors, and Eigenvalues

PCA

Equivalence of Maximizing Variance and Minimizing Reconstruction Error

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.

$$\begin{aligned}
\mathbf{v}^* &= \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\text{argmin}} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)} - (\mathbf{v}^T\mathbf{x}^{(i)})\mathbf{v}||^2 && (2) \\
&= \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\text{argmin}} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)}||^2 - (\mathbf{v}^T\mathbf{x}^{(i)})^2 && (3) \\
&= \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\text{argmax}} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{v}^T\mathbf{x}^{(i)})^2 && (4)
\end{aligned}$$

PCA: the First Principal Component

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

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

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

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

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

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

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

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

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

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

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 $X^T X$ $= \Sigma$
    - 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

Background: SVD

Singular Value Decomposition (SVD)

For any arbitrary matrix $\mathbf{A}$, SVD gives a decomposition:

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{1}$$

where $\mathbf{\Lambda}$ is a diagonal matrix, and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices.

SVD for PCA

For any arbitrary matrix $\mathbf{A}$, SVD gives a decomposition:

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T \tag{1}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix, and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices.

Suppose we obtain an SVD of our data matrix $\mathbf{X}$, so that: T

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T \tag{1}$$

Now consider what happens when we rewrite $\boldsymbol{\Sigma} = \frac{1}{N}\mathbf{X}^T\mathbf{X}$ terms of this SVD.

$$\begin{aligned}
\boldsymbol{\Sigma} &= \frac{1}{N}\mathbf{X}^T\mathbf{X} && (2)\\
&= \frac{1}{N}(\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T)^T(\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T) && (3)\\
&= \frac{1}{N}(\mathbf{V}\boldsymbol{\Lambda}^T\cancel{\mathbf{U}^T})(\cancel{\mathbf{U}}\boldsymbol{\Lambda}\mathbf{V}^T) && (4)\\
&= \frac{1}{N}\mathbf{V}\boldsymbol{\Lambda}^T\boldsymbol{\Lambda}\mathbf{V}^T && (5)\\
&= \frac{1}{N}\mathbf{V}(\boldsymbol{\Lambda})^2\mathbf{V}^T && (6)
\end{aligned}$$

Above we used the fact that $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ since $\mathbf{U}$ is orthogonal by definition.

We find that $(\boldsymbol{\Lambda})^2$ is a diagonal matrix whose entries are $\Lambda_{ii} = \lambda_i^2$ the squares of the eigenvalues of ~~the SVD of $\mathbf{X}$~~. X^TX Further, both $\mathbf{X}$ and $\mathbf{X}^T\mathbf{X}$ share the same eigenvectors in their SVD.

Thus, we can run SVD on $\mathbf{X}$ without ever instantiating the large $\mathbf{X}^T\mathbf{X}$ to obtain the necessary principal components more efficiently.

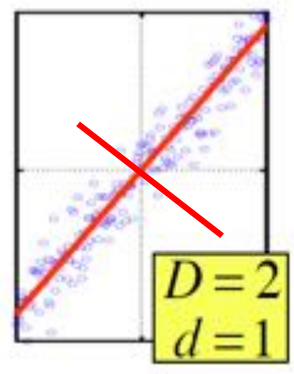
$$\underbrace{\underbrace{X^T}_{M\times N}\underbrace{X}_{N\times M}}_{M\times M}$$

eigenvectors $\left\{\begin{array}{l}\text{columns of } V\\ \text{rows of } U\end{array}\right.$

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).

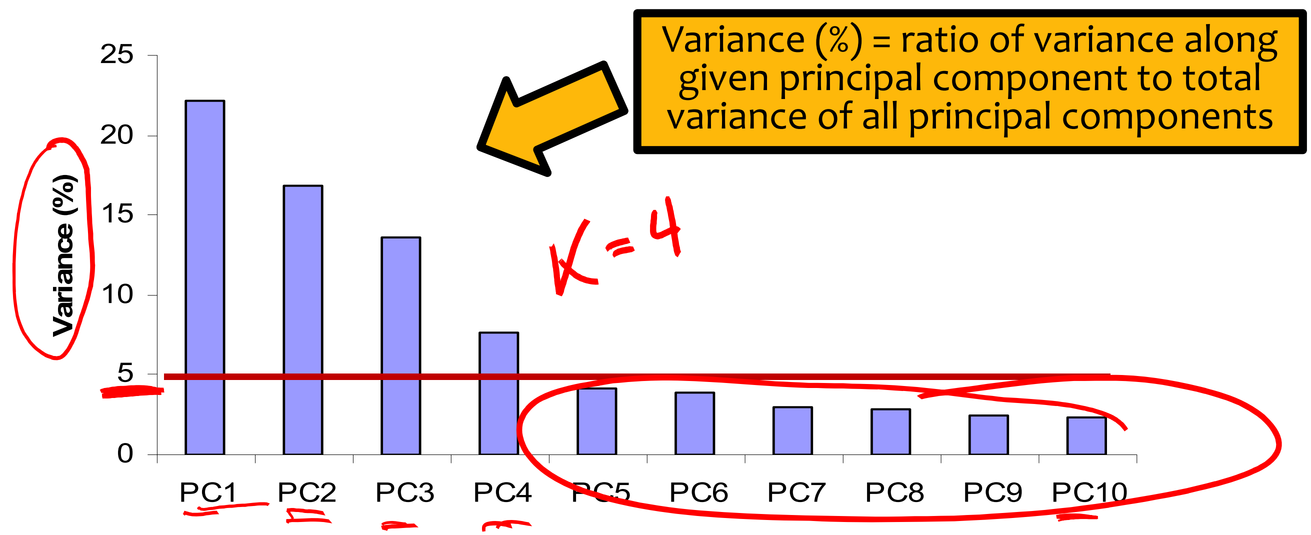
Eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots$

- 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

How Many PCs?

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



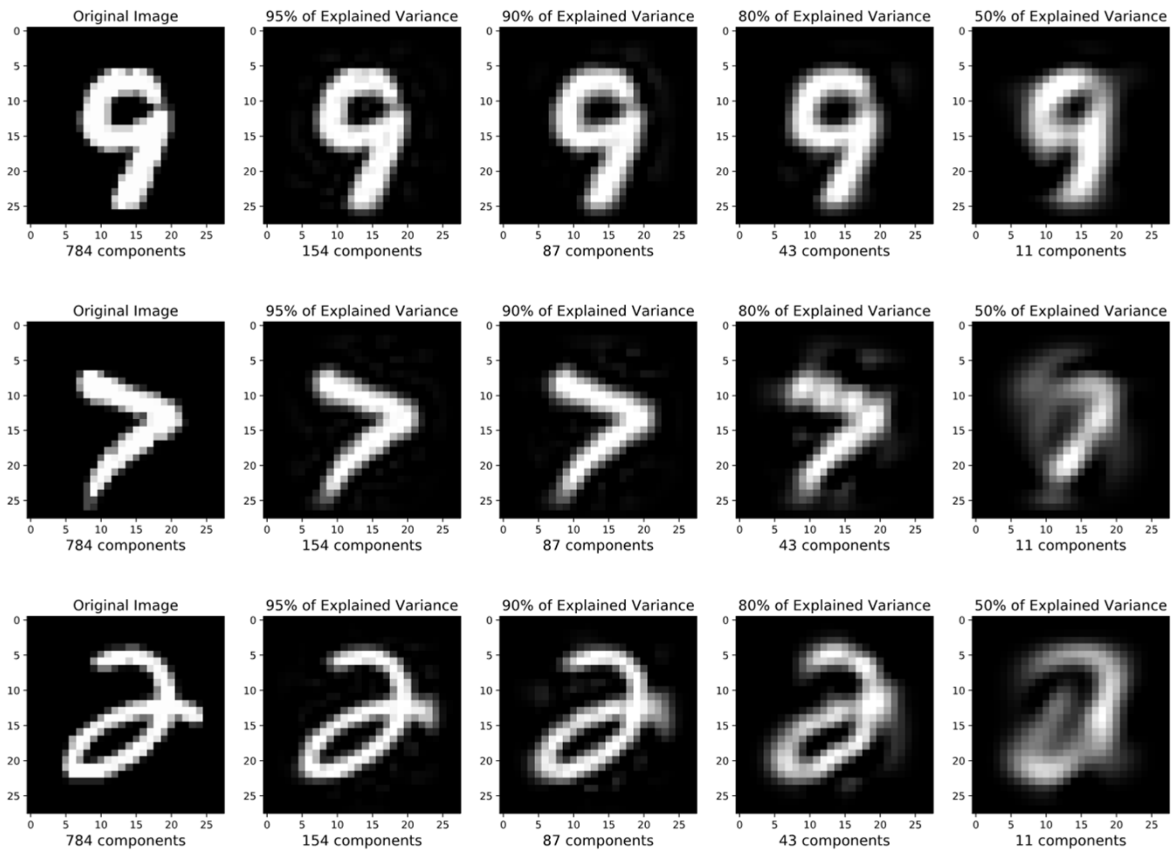
- You do lose some information, but if the eigenvalues are small, you don't lose much
  - M dimensions in original data
  - calculate M eigenvectors and eigenvalues
  - choose only the first D eigenvectors, based on their eigenvalues
  - final data set has only D dimensions

PCA EXAMPLES

Projecting MNIST digits

Task Setting: $28 \times 28 = 784$

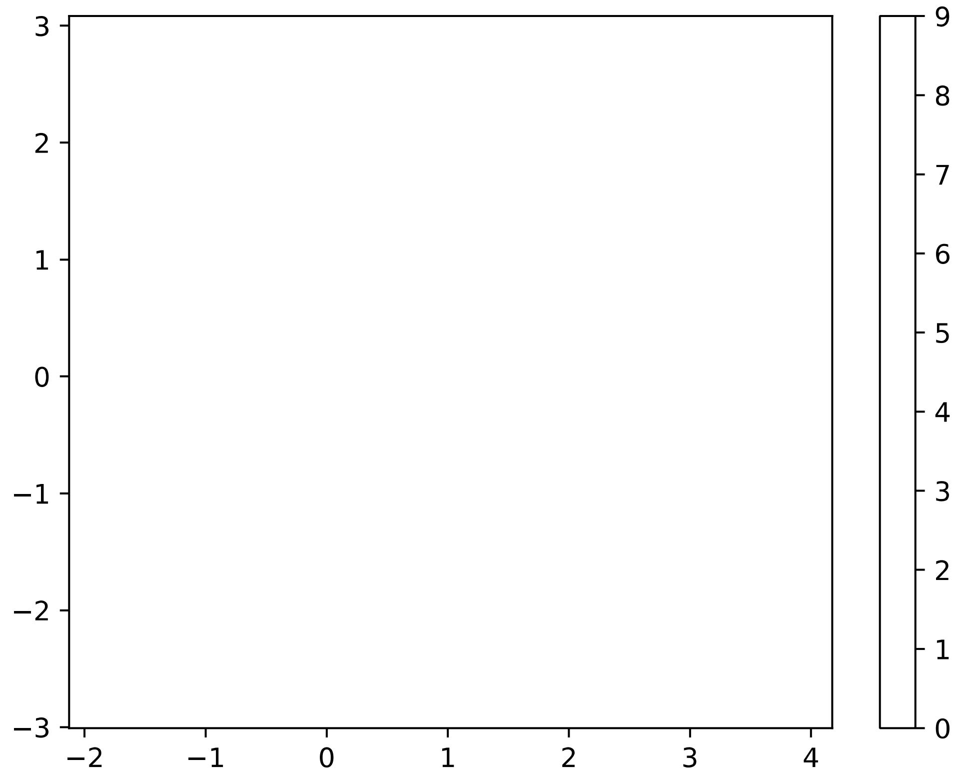
1. Take ~~25x25~~ images of digits and project them down to K components
2. Report percent of variance explained for K components
3. Then project back up to 25x25 image to visualize how much information was preserved



Projecting MNIST digits

Task Setting:

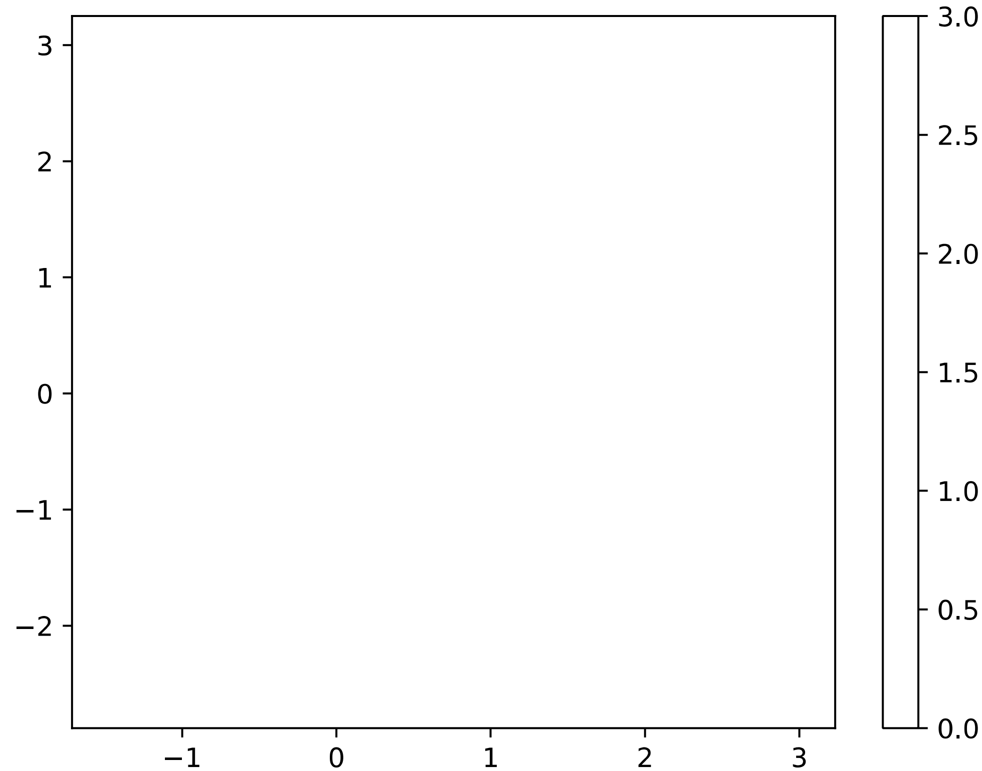
1. Take 25x25 images of digits and project them down to 2 components
2. Plot the 2 dimensional points
3. Here we look at all ten digits 0 - 9



Projecting MNIST digits

Task Setting:

1. Take 25x25 images of digits and project them down to 2 components
2. Plot the 2 dimensional points
3. Here we look at just four digits 0, 1, 2, 3



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