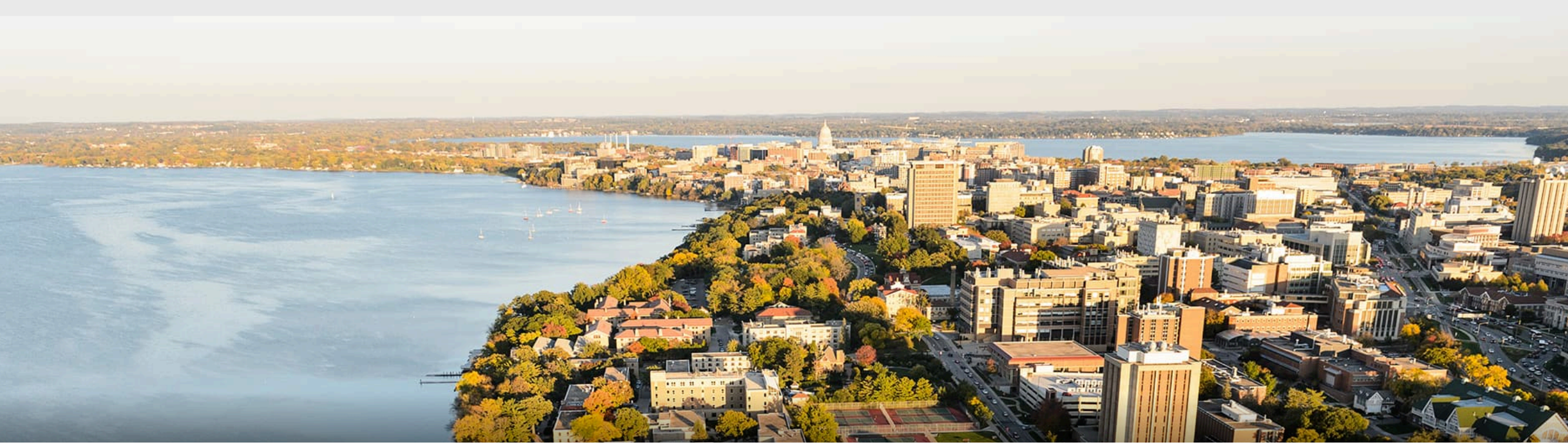


CS 760: Machine Learning
Unsupervised Learning II

Josiah Hanna

University of Wisconsin-Madison

October 31, 2023

Announcements

- Homework 4 due today; homework 5 released today.

- 👻 Happy Halloween! 🎃

Learning Outcomes

At the end of today's lecture, you will be able to:

1. Explain the four main types of clustering we consider in this course.
2. Explain the use of principle component analysis (PCA) and how it finds a lower dimensional representation of data.

Outline

- **Clustering Review**
  - k-means, hierarchical
- **Spectral clustering**
  - Graph Laplacian, algorithm, comparison to k-means
- **Principal Components Analysis**
  - Definition, Algorithm, Interpretations, Analysis

Outline

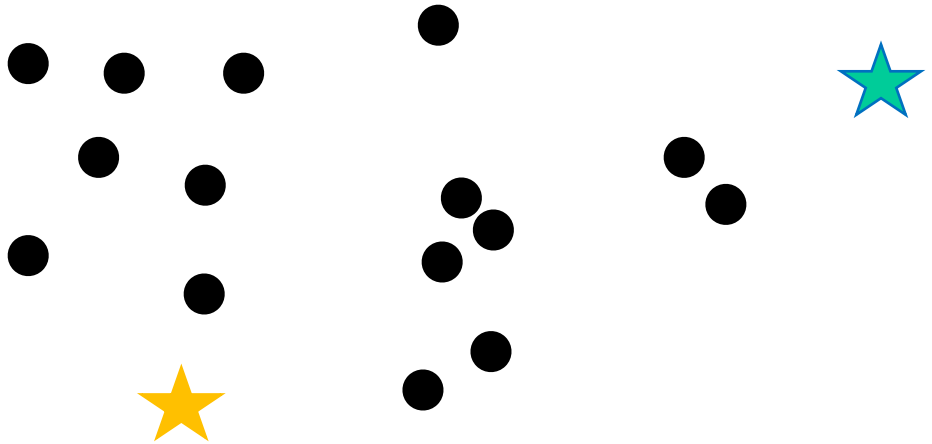
- **Clustering Review**
  - k-means, hierarchical
- **Spectral clustering**
  - Graph Laplacian, algorithm, comparison to k-means
- **Principal Components Analysis**
  - Definition, Algorithm, Interpretations, Analysis

K-Means (Lloyd's) Clustering

k-means is a type of partitional **centroid-based clustering**

Algorithm:

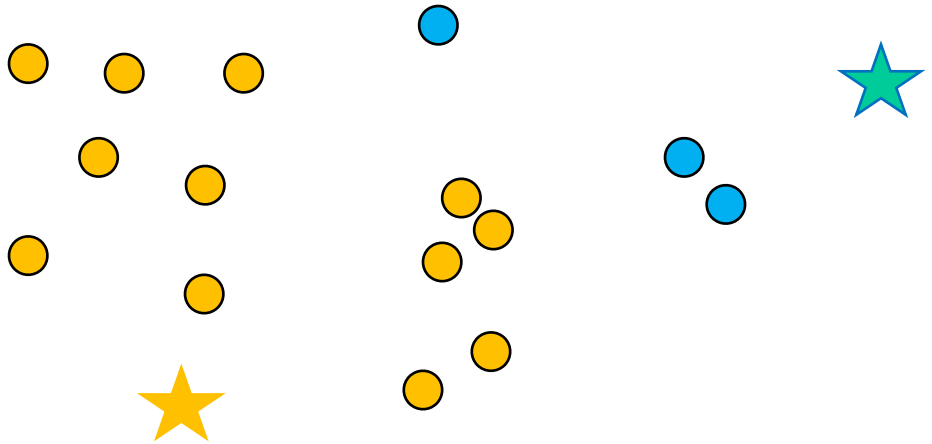
1. Randomly pick k cluster centers



K-Means Clustering: Algorithm

K-Means clustering

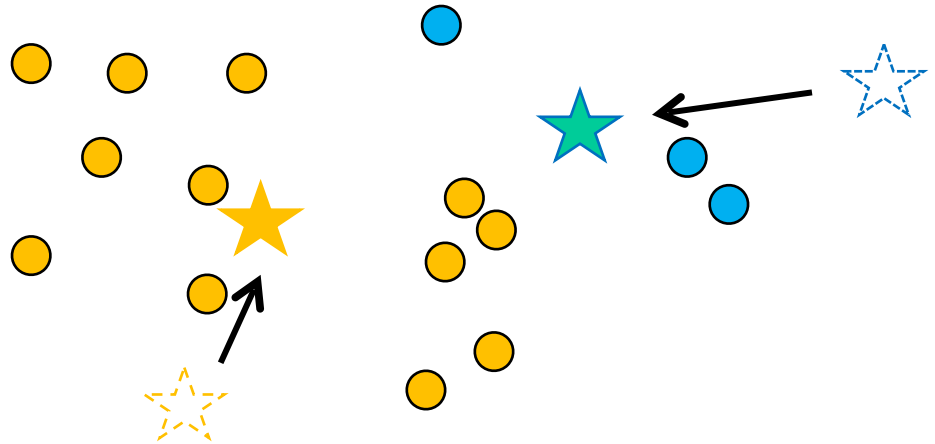
2. Find closest center for each point



K-Means Clustering: Algorithm

K-Means clustering

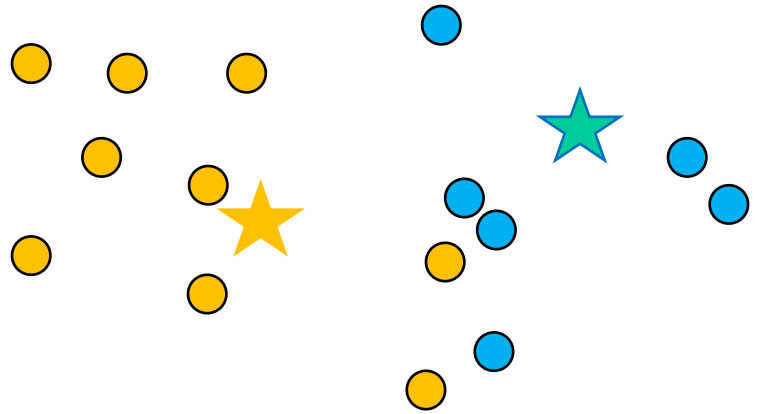
3. Update cluster centers by computing centroids



K-Means Clustering: Algorithm

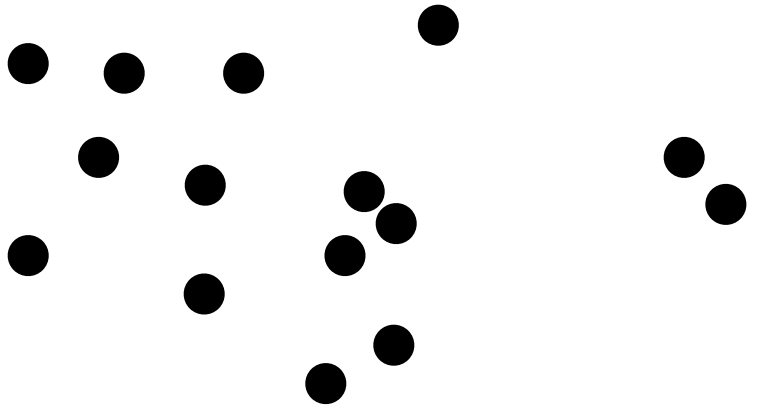
K-Means clustering

Repeat Steps 2 & 3 until convergence



HC: Agglomerative Clustering Example

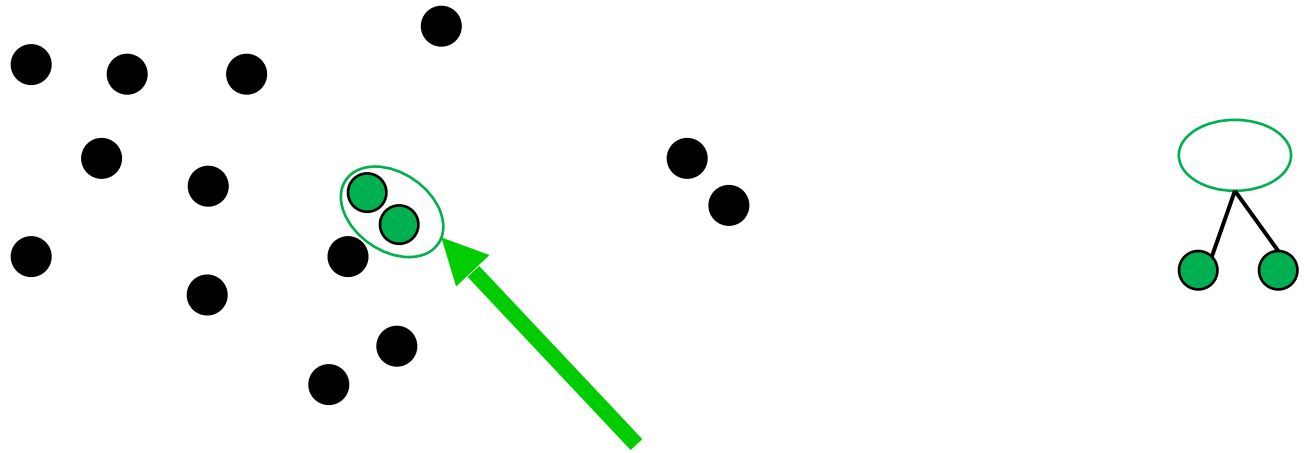
Agglomerative: Start: every point is its own cluster



HC: Agglomerative Clustering Example

Basic idea: build a “hierarchy”

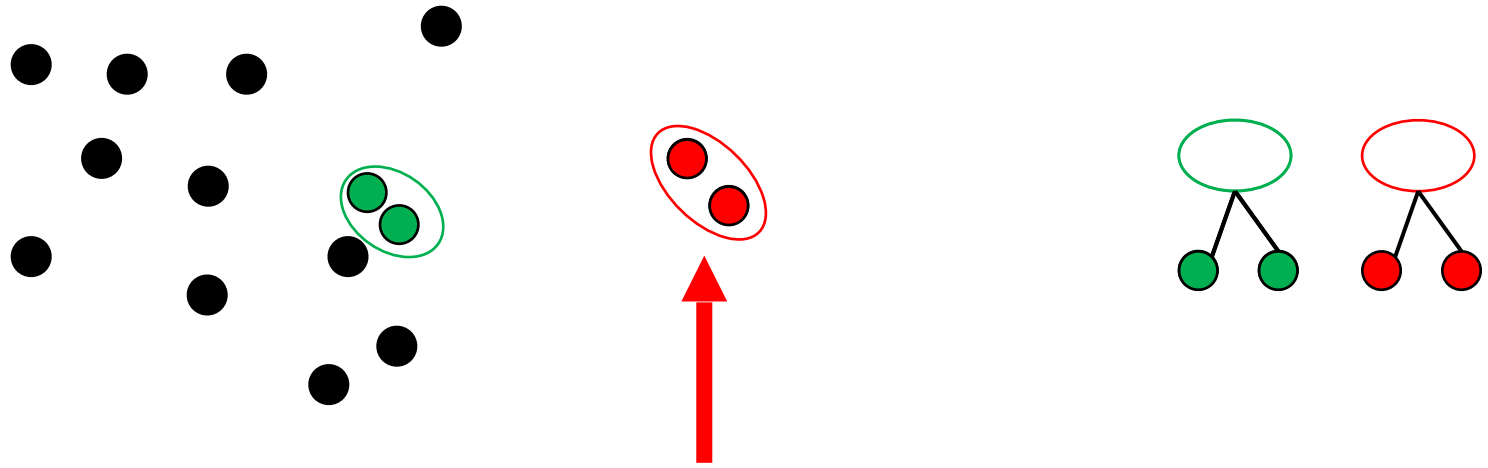
- Get pair of clusters that are closest and merge



HC: Agglomerative Clustering Example

Basic idea: build a “hierarchy”

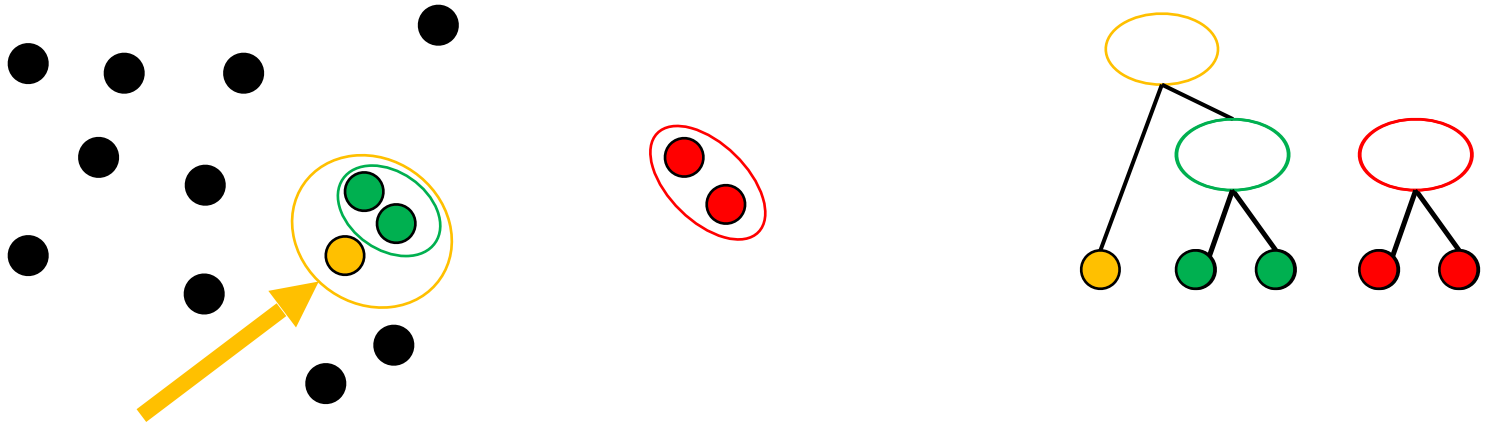
- **Repeat:** Get pair of clusters that are closest and merge



HC: Agglomerative Clustering Example

Basic idea: build a “hierarchy”

- **Repeat:** Get pair of clusters that are closest and merge



HC: Merging Criteria

Merge: use closest clusters. Define closest?

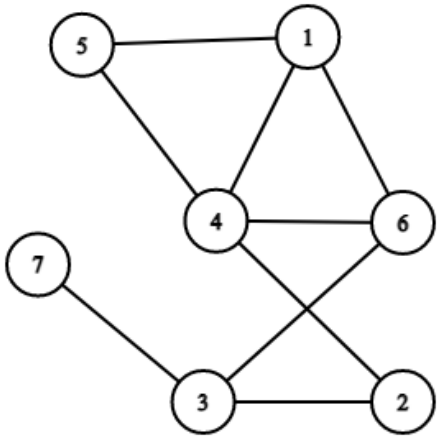
- Single-linkage $$d(A,B) = \min_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$
- Complete-linkage $$d(A,B) = \max_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$
- Average-linkage $$d(A,B) = \frac{1}{|A||B|} \sum_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$

Outline

- **Clustering Review**
  - k-means, hierarchical
- **Spectral clustering**
  - Graph Laplacian, algorithm, comparison to k-means
- **Principal Components Analysis**
  - Definition, Algorithm, Interpretations, Analysis

Graph/proximity based clustering

- Recall: Graph G = (V,E) has vertex set V, edge set E.
  - Edges can be weighted or unweighted
  - Encode **similarity**



- Treat each data point as a node in a graph.
- Edges based on similarity of data points
- E.g. for Euclidean vectors:

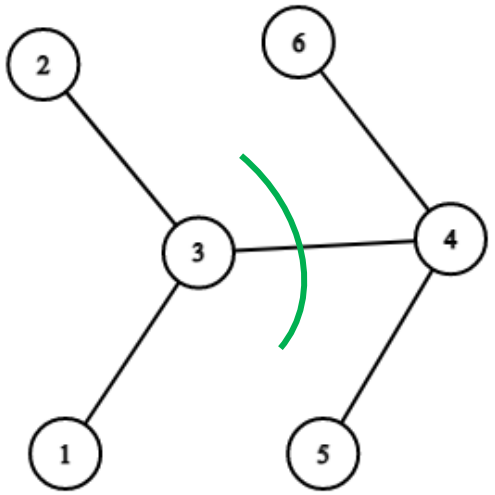
$$w_{ij} = e^{-\alpha \|x_i - x_j\|^2}$$

- But they don't need to be in Euclidean space!

Graph-Based Clustering

Want: partition V into k groups

- Implies a graph "cut"
- One idea: minimize the **weight** of the cut



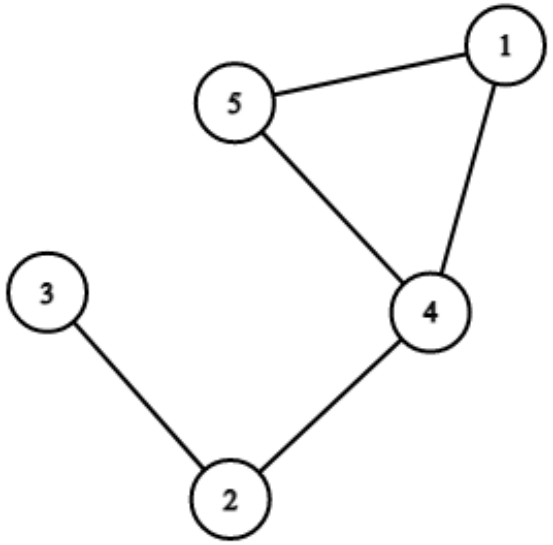
$$W(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

$$\text{cut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} W(A_i, \overline{A}_i).$$

Partition-Based Clustering

How do we compute these?

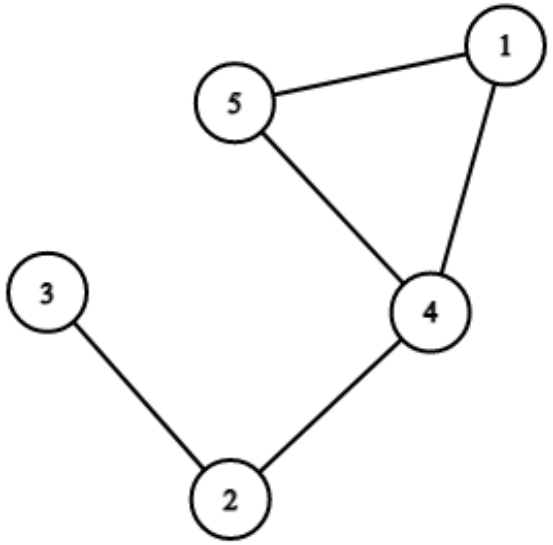
- Hard problem → heuristics
  - Greedy algorithm
  - “Spectral” approaches
- Spectral clustering approach:
  - **Adjacency** matrix



$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Partition-Based Clustering

- Spectral clustering approach:
  - **Adjacency** matrix
  - **Degree** matrix

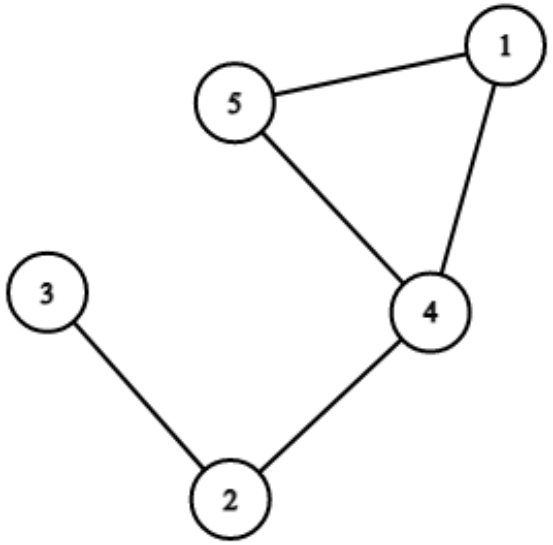


$$D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix} \quad A = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Spectral Clustering

- Spectral clustering approach:
  - 1. Compute **Laplacian L = D – A**

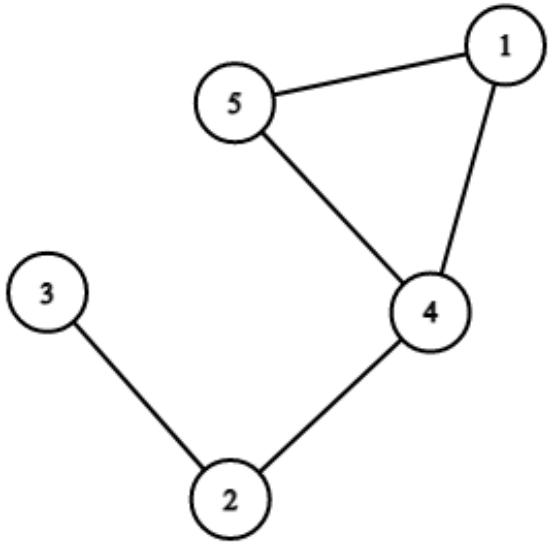
  (Important tool in graph theory)



$$L = \underbrace{\begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}}_{\text{Degree Matrix}} - \underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Adjacency Matrix}} = \underbrace{\begin{bmatrix} 2 & 0 & 0 & -1 & -1 \\ 0 & 2 & -1 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ -1 & -1 & 0 & 3 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{bmatrix}}_{\text{Laplacian}}$$

Spectral Clustering

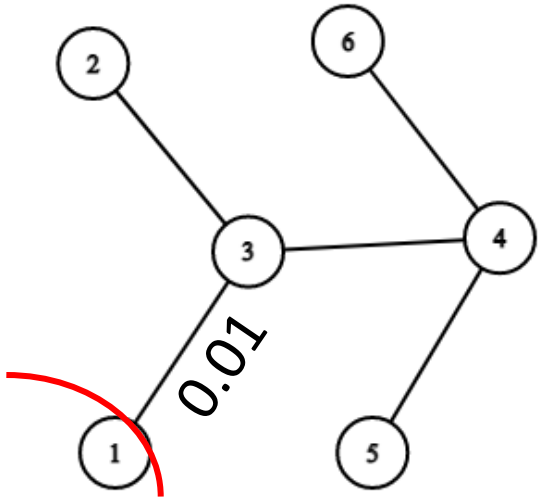
- Spectral clustering approach:
  - 1. Compute **Laplacian L = D – A**
  - 1a (optional): compute normalized Laplacian:
    $L = I - D^{-1/2}AD^{-1/2}$, or $L = I - D^{-1}A$
  - 2. Compute *k* **smallest** eigenvectors of **L**
  - 3. Set *U* to be the *n* x *k* matrix with eigenvectors $u_1, \ldots, u_k$ as columns. Take the *n* rows formed as points
  - 4. Run k-means on the representations

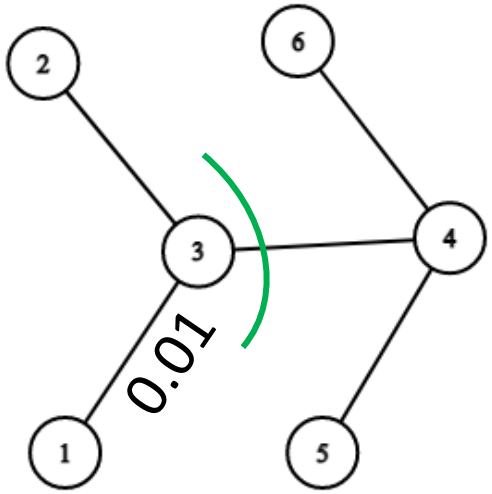


Why normalized Laplacian?

Want: partition V into V_1 and V_2

- Implies a graph "cut"
- One idea: minimize the **weight** of the cut
  - Downside: might just cut off one node
  - Need: "**balanced**" cut





Why Normalized Laplacian?

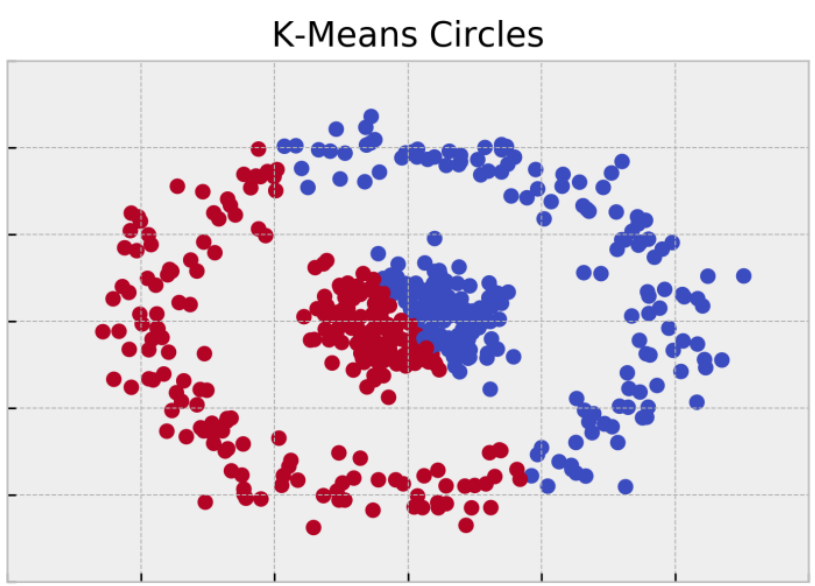
Want: partition V into V_1 and V_2

- Just minimizing weight is not always a good idea.
- We want **balance!**

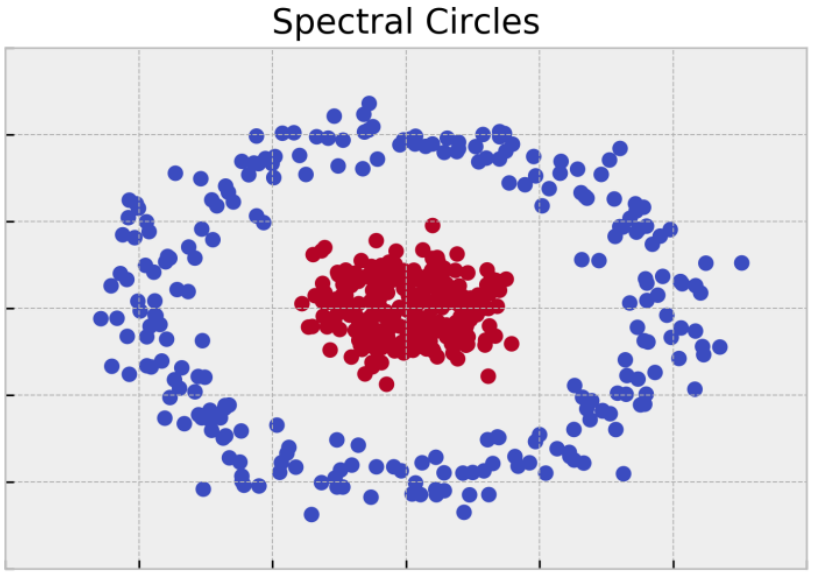
$$\mathrm{Ncut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A}_i)}{\mathrm{vol}(A_i)}$$

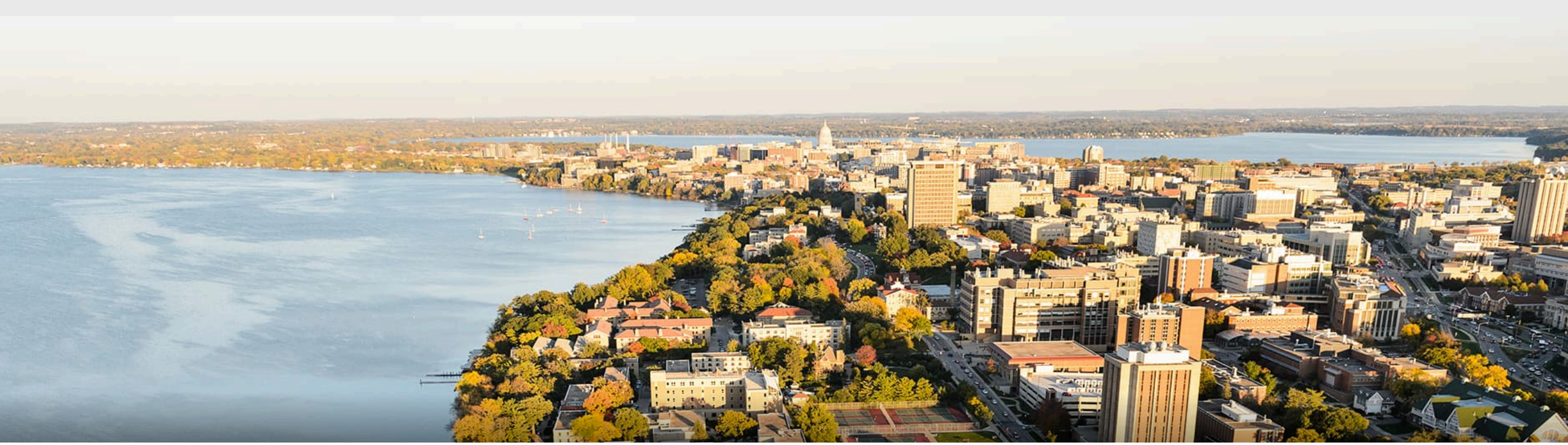
$$\mathrm{vol(A)} = \sum_{i \in A} \mathrm{degree(i)} = \sum_{i \in A} \sum_{j \in \mathrm{nbd}(i)} w_{ij}$$

Spectral Clustering



Credit: William Fleshman





Break & Quiz

Break & Quiz

Q 2.1: You have seven 2-dimensional points. You run 3-means on it, with initial clusters

$$C_1 = \{(2,2),(4,4),(6,6)\}, C_2 = \{(0,4),(4,0)\}, C_3 = \{(5,5),(9,9)\}$$

Cluster centroids at the next iteration are?

- A. C_1: (4,4), C_2: (2,2), C_3: (7,7)
- B. C_1: (6,6), C_2: (4,4), C_3: (9,9)
- C. C_1: (2,2), C_2: (0,0), C_3: (5,5)
- D. C_1: (2,6), C_2: (0,4), C_3: (5,9)

Break & Quiz

Q 2.1: You have seven 2-dimensional points. You run 3-means on it, with initial clusters

$$C_1 = \{(2,2),(4,4),(6,6)\}, C_2 = \{(0,4),(4,0)\}, C_3 = \{(5,5),(9,9)\}$$

Cluster centroids at the next iteration are?

- **A. C_1: (4,4), C_2: (2,2), C_3: (7,7)**
- B. C_1: (6,6), C_2: (4,4), C_3: (9,9)
- C. C_1: (2,2), C_2: (0,0), C_3: (5,5)
- D. C_1: (2,6), C_2: (0,4), C_3: (5,9)

Break & Quiz

Q 2.2: If we do hierarchical clustering on n points, the maximum depth of the resulting tree is

- A. 2
- B. log *n*
- C. *n*/2
- D. *n*-1

Break & Quiz

Q 2.2: If we do hierarchical clustering on n points, the maximum depth of the resulting tree is

- A. 2
- B. log *n*
- C. *n*/2
- **D. *n*-1**

Outline

- **Clustering Review**
  - k-means, hierarchical
- **Spectral clustering**
  - Graph Laplacian, algorithm, comparison to k-means
- **Principal Components Analysis**
  - Definition, Algorithm, Interpretations, Analysis

High-Dimensional Data

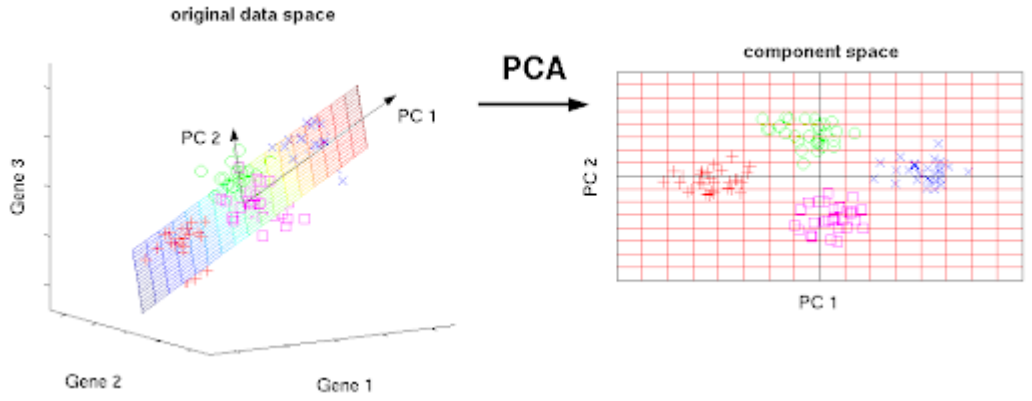
- High-dimensions = lots of features
- We've seen this repeatedly, but some examples:
- Example: Document classification
  - Features per document = thousands of words/unigrams millions of bigrams, contextual information
- **Example**: Surveys - Netflix

  480189 users x 17770 movies

	movie 1	movie 2	movie 3	movie 4	movie 5	movie 6
Tom	5	?	?	1	3	?
George	?	?	3	1	2	5
Susan	4	3	1	?	5	1
Beth	4	3	?	2	4	2

Dealing with Dimensionality

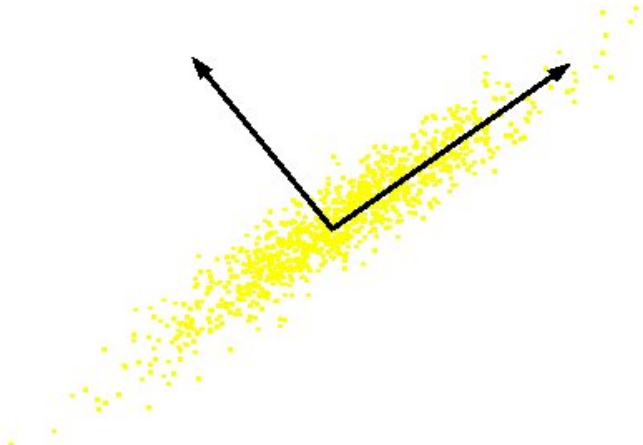
- **Goal:** Discover hidden (potentially lower dimensional) structure in high dimensional datasets.
- **Example algorithms:** PCA, Kernel PCA, ICA
- Some uses:
  - Visualization
  - More efficient use of resources (e.g., time, memory, communication)
  - Noise removal (improving data quality)
  - Further processing by machine learning algorithms



Principal Components Analysis

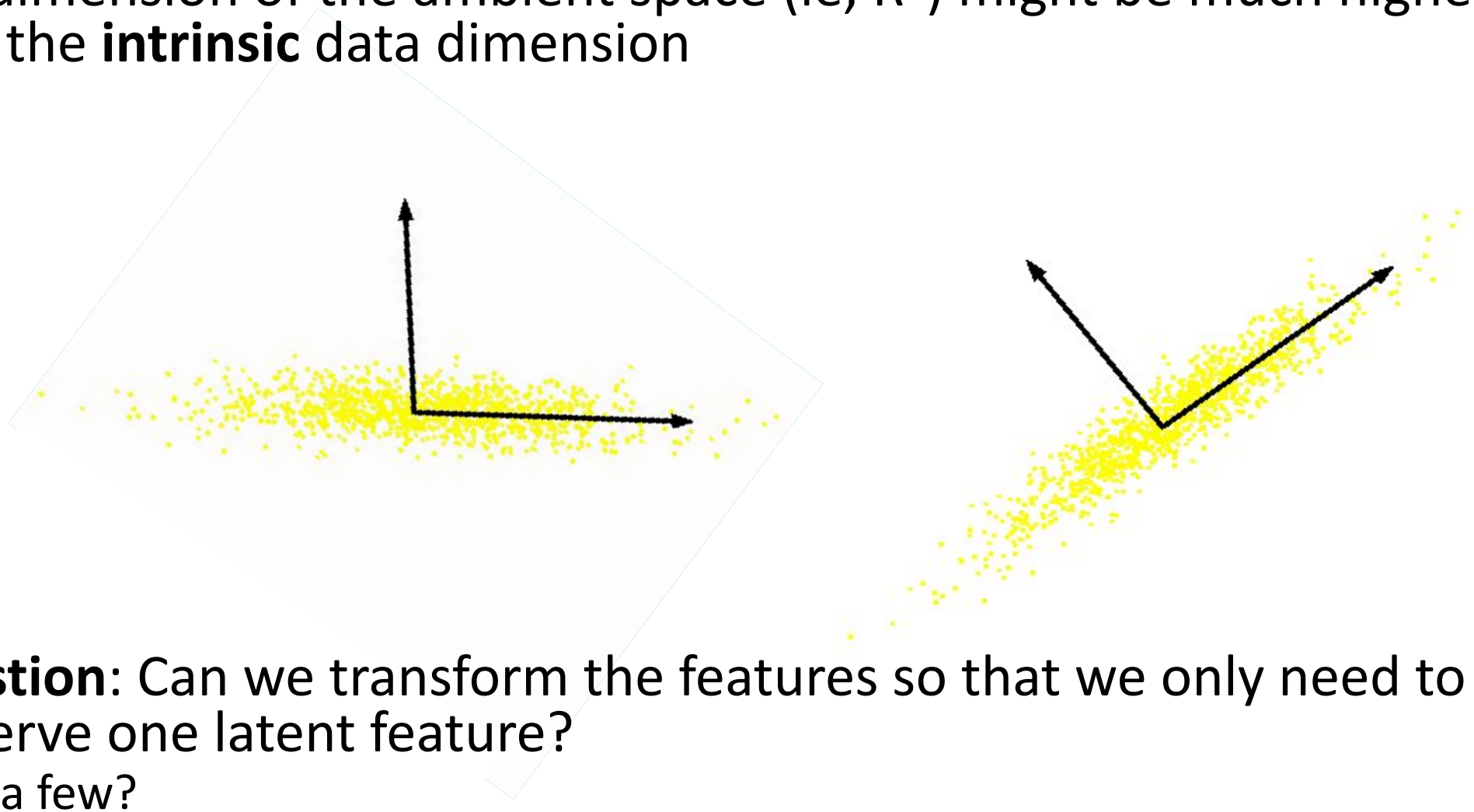
- Unsupervised technique for extracting variance structure from high dimensional datasets
  - And also reduces dimensionality

- PCA: orthogonal projection / transformation of the data
  - Into a (possibly lower dimensional) subspace
  - So that the variance of the projected data is maximized.



PCA Intuition

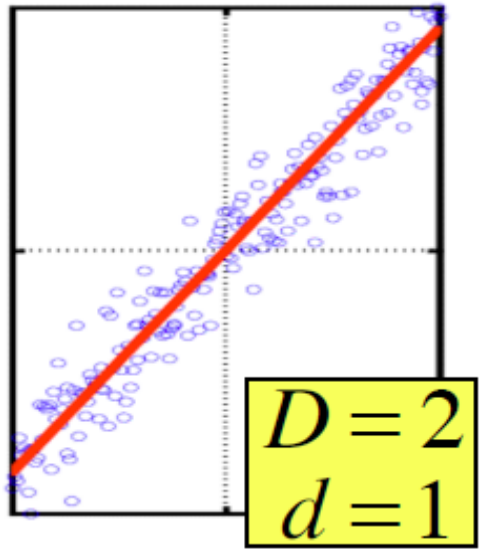
- The dimension of the ambient space (ie, R^d) might be much higher than the **intrinsic** data dimension

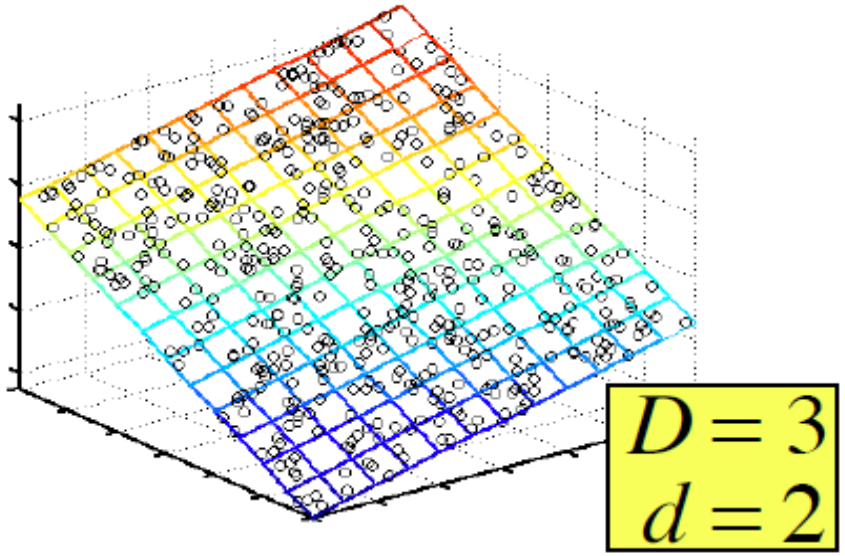


- **Question**: Can we transform the features so that we only need to preserve one latent feature?
  - Or a few?

PCA Intuition

- Some more visualizations

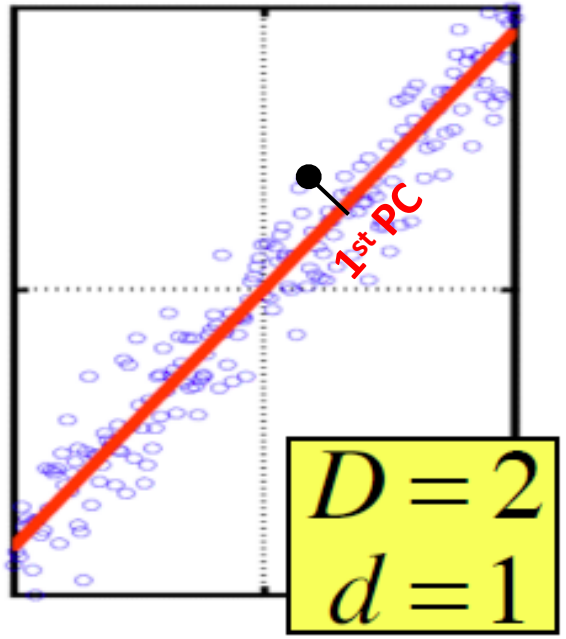




- In case where data lies on or near a low d-dimensional linear subspace, axes of this subspace are an effective representation of the data.

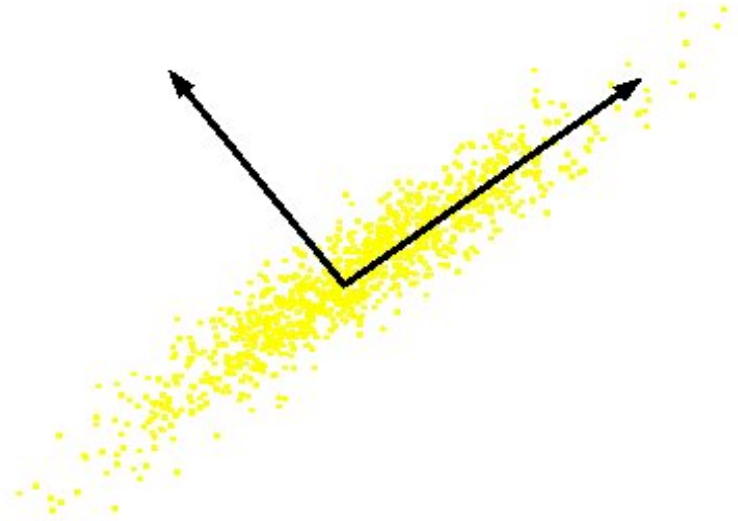
PCA: Principal Components

- **Principal Components (PCs)** are orthogonal directions that capture most of the variance in the data.
  - First PC – direction of greatest variability in data.
  - Projection of data points along first PC discriminates data most along any one direction



PCA: Principal Components and Projection

- How does dimensionality reduction work? From *d* dimensions to *r* dimensions:
  - Get $v_1, v_2, \ldots, v_r \in \mathbb{R}^d$
  - Orthogonal!



- Want to represent each $x \approx \alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_r v_r$.
  - New representation of x is $[\alpha_1, \alpha_2, \ldots, \alpha_r]$.
- Maximizing variability is equivalent to **minimizing reconstruction error**

- Obtain representation by projecting each x onto principle components.

PCA Approach Overview

- Want unit vector directions (i.e., components) so that:
  - Projecting data maximizes variance
  - Specifically, for centered (i.e., mean zero) data:

$$\max_{v} \sum_{i=1}^{n} \langle x_i, v \rangle^2 = ||Xv||^2$$

- Then transform X (how in 2 slides) and do this **recursively**
  - Get orthogonal directions

$$v_1, v_2, \ldots, v_r \in \mathbb{R}^d$$

PCA First Step

- First component,

$$v_1 = \arg\max_{\|v\|=1} \sum_{i=1}^{n} \langle v, x_i \rangle^2$$

- Same as getting

$$v_1 = \arg\max_{\|v\|=1} \|Xv\|^2$$

PCA Recursion

- Once we have *k-1* components, how do we get the next?

$$\hat{X}_k = X - \sum_{i=1}^{k-1} X v_i v_i^T$$

Deflation

- Then do the same thing

$$v_k = \arg \max_{\|v\|=1} \|\hat{X}_k w\|^2$$

PCA Interpretations

- The v's are eigenvectors of XX^T **(Gram matrix)**
  - We'll see why in a second

- XX^T (proportional to) sample covariance matrix
  - When data is 0 mean!
  - I.e., PCA is eigendecomposition of sample covariance

- Nested subspaces *span(v1), span(v1,v2),…,*

PCA Interpretations: First Component

- Two specific ways to think about the first component
- **Maximum variance direction**
  - What we saw so far

$$\sum_{i=1}^{n} (\mathbf{v}^T \mathbf{x}_i)^2 = \mathbf{v}^T \mathbf{X} \mathbf{X}^T \mathbf{v}$$

- **Minimum reconstruction error**
  - A direction so that projection yields minimum MSE in reconstruction

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

PCA Interpretations: Equivalence

- Interpretation 1.

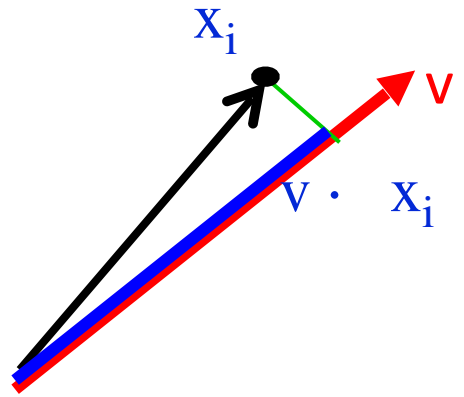
Maximum variance direction

$$\sum_{i=1}^{n} (\mathbf{v}^T \mathbf{x}_i)^2 = \mathbf{v}^T \mathbf{X}\mathbf{X}^T \mathbf{v}$$

- Interpretation 2.

Minimum reconstruction error

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



- Why are these equivalent?
  - Use Pythagorean theorem.
  - Maximizing **blue** segment is the same as minimizing the **green**

PCA Gram Matrix Interpretation

- Recall our first PC, maximized variance:

$$\max_{\mathbf{v}} \ \mathbf{v}^T\mathbf{X}\mathbf{X}^T\mathbf{v} \quad \text{s.t.} \quad \mathbf{v}^T\mathbf{v} = 1$$

- Constrained optimization
  - The usual approach: Lagrangian + KKT conditions

$$\text{Lagrangian: } \max_{\mathbf{v}} \mathbf{v}^T\mathbf{X}\mathbf{X}^T\mathbf{v} - \lambda\mathbf{v}^T\mathbf{v}$$

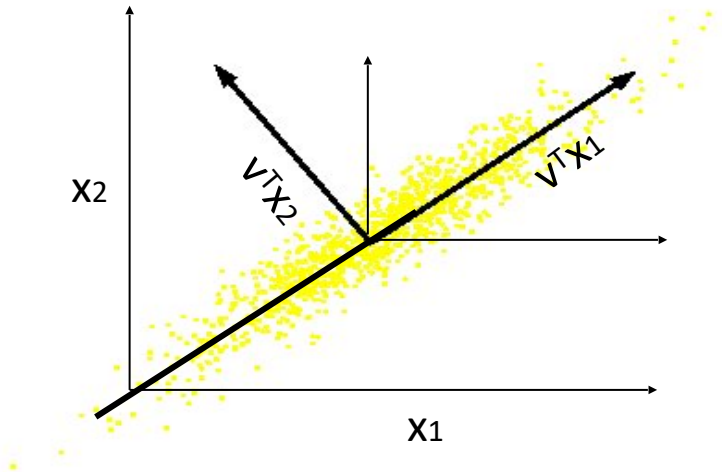
$$\partial/\partial\mathbf{v} = 0 \ \ (\mathbf{X}\mathbf{X}^T - \lambda\mathbf{I})\mathbf{v} = 0 \ \ \Rightarrow \boxed{(\mathbf{X}\mathbf{X}^T)\mathbf{v} = \lambda\mathbf{v}}$$

See appendix in “Pattern Recognition and Machine Learning” for intro to Lagrange multipliers.

PCA Covariance Matrix Interpretation

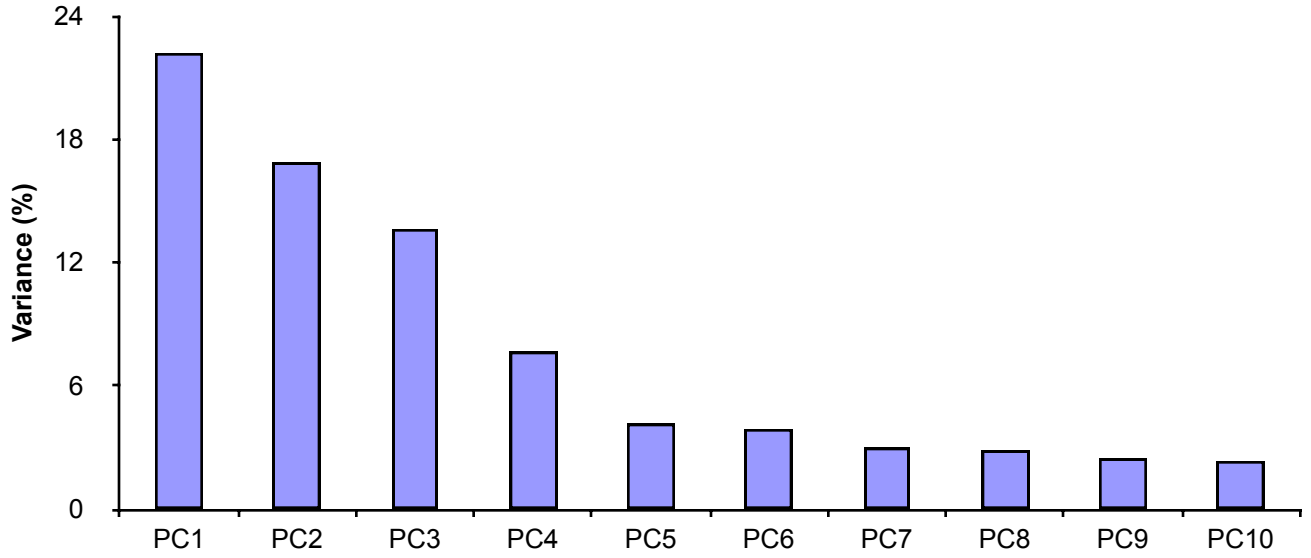
- So… $\Rightarrow (\mathbf{X}\mathbf{X}^T)\mathbf{v} = \lambda\mathbf{v}$
- Means that v (the first PC) is an eigenvector of XX^T
- Its eigenvalue λ denotes the amount of variability captured along that dimension

- PCs are just the eigenvectors…
  - How to find them? Eigendecomposition
- Don't need to keep all eigenvectors
  - Just the ones for largest eigenvalues



PCA Dimensionality Reduction

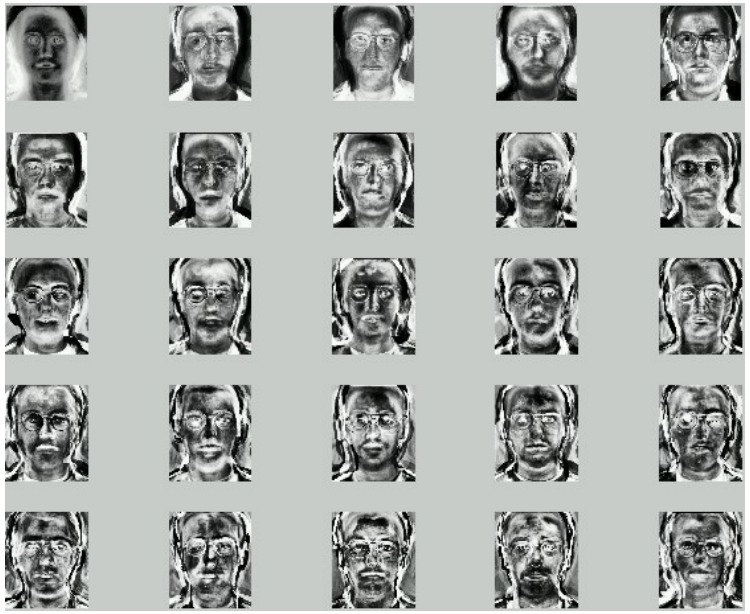
- In high-dimensional problems, data sometimes lies near a linear subspace, as noise introduces small variability
- Only keep data projections onto principal components with **large** eigenvalues
- Can *ignore* the components of smaller significance.



Example of PCA on Face Images

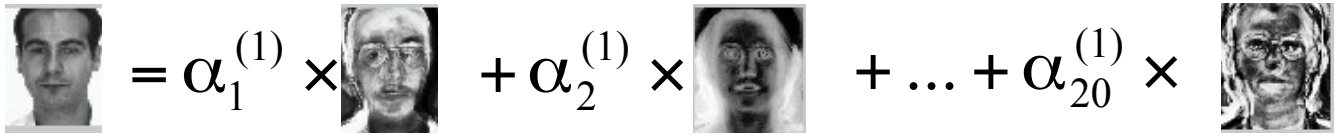
- Given instances $\{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$

- **Goal**: model *h* that represents *x* with
  - lower-dim. feature vectors
  - preserving information
- Example: Eigenfaces

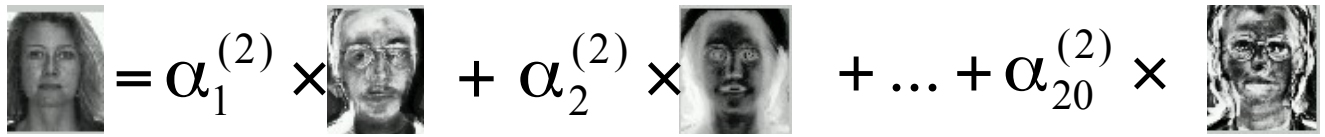


Dimensionality Reduction: Setup

Example: Eigenfaces



$$x^{(1)} = \langle \alpha_1^{(1)}, \alpha_2^{(1)}, \ldots, \alpha_{20}^{(1)} \rangle$$



$$x^{(1)} = \langle \alpha_1^{(2)}, \alpha_2^{(2)}, \ldots, \alpha_{20}^{(2)} \rangle$$

Application: Image Compression

- Start with image; divide into 12x12 patches
  - I.E., 144-D vector
  - **Original image:**

Application: Image Compression

- Project to 6D,

Compressed

Original

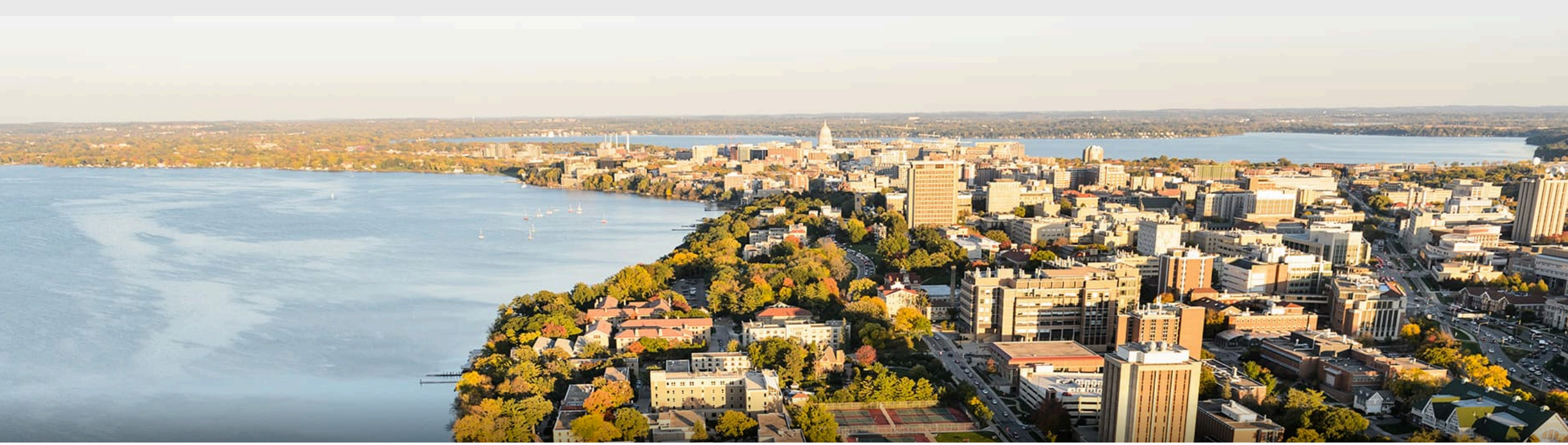
Q2-2: Are these statements true or false?
(A) The principal component with the largest eigenvalue maximizes the reconstruction error.
(B) The dimension of original data representation is always higher than the dimension of transformed representation of PCA.

1. True, True
2. True, False
3. False, True
4. False, False

Q2-2: Are these statements true or false?
(A) The principal component with the largest eigenvalue maximizes the reconstruction error.
(B) The dimension of the original data representation is always higher than the dimension of transformed representation of PCA.

1. True, True
2. True, False
3. False, True
4. False, False ⟵

(A) The principal component with the largest eigenvalue captures the maximum amount of variability which is equivalent to minimum reconstruction error.
(B) If the matrix XX^T is full-rank, they can be of the same dimension.



Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov