CSE 158 – Lecture 5

Web Mining and Recommender Systems

Dimensionality Reduction

This week

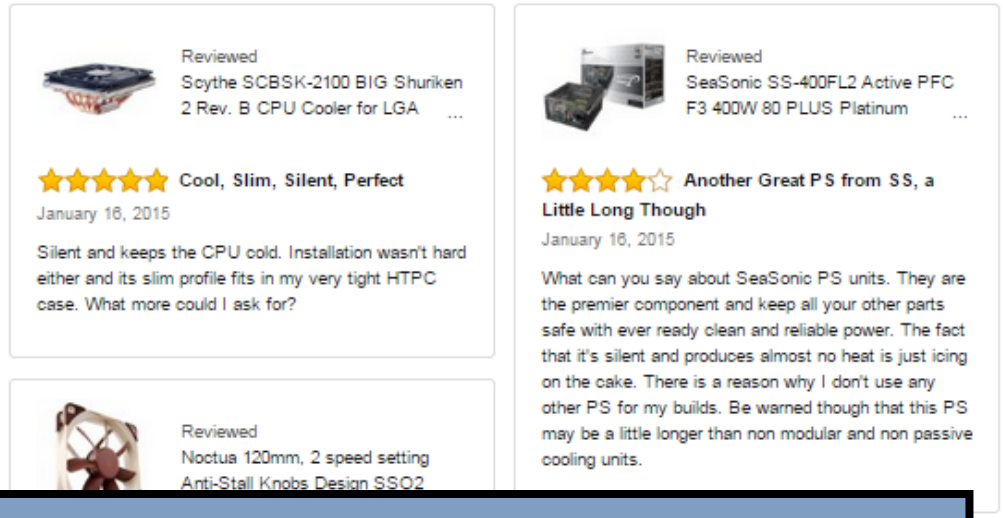
How can we build **low dimensional** representations of **high dimensional** data?

e.g. how might we (compactly!) represent

1. The ratings I gave to every movie I've watched?
2. The complete text of a document?
3. The set of my connections in a social network?

Dimensionality reduction

Q1: The ratings I gave to every movie I've watched
(or product I've purchased)



A1: A (sparse) vector including all movies

F_julian = [0.5, ?, 1.5, 2.5, ?, ?, … , 5.0]

A-team ABBA, the movie Zoolander

Dimensionality reduction

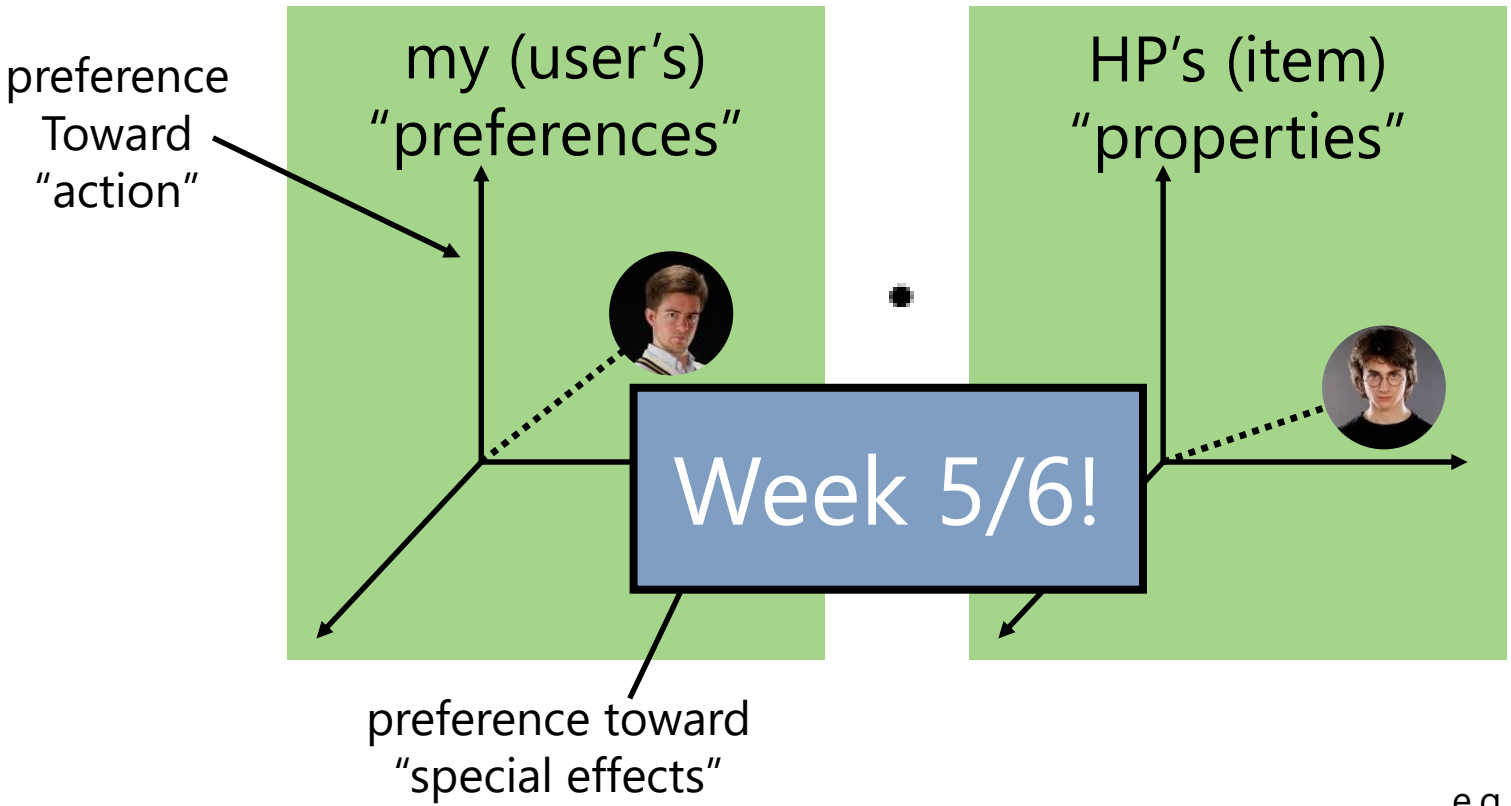
A1: A (sparse) vector including all movies

F_julian = [0.5, ?, 1.5, 2.5, ?, ?, … , 5.0]

— Too big
— Missing data
— What to do if new movies come along

Dimensionality reduction

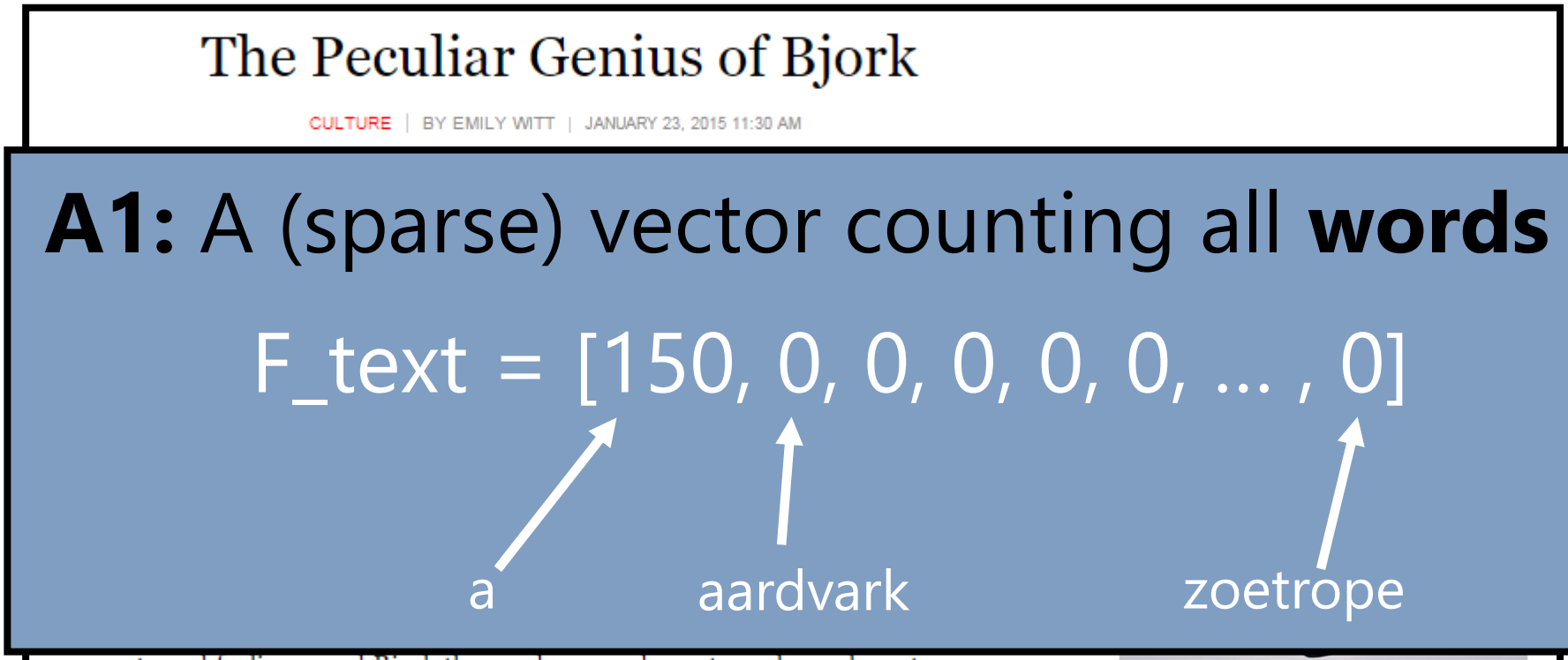
A2: Describe my preferences using a **low-dimensional** vector



e.g. Koren & Bell (2011)

Dimensionality reduction

Q2: How to represent the complete text of a document?



Dimensionality reduction

A1: A (sparse) vector counting all **words**

F_text = [150, 0, 0, 0, 0, 0, … , 0]

Incredibly high-dimensional…

- Costly to store and manipulate
- Many dimensions encode essentially the same thing
- Many dimensions devoted to the “long tail” of obscure words (technical terminology, proper nouns etc.)

Dimensionality reduction

A2: A low-dimensional vector describing the **topics in the document**

87 of 102 people found the following review helpful

★★★★★ **You keep what you kill**, December 27, 2004

By Schtinky "Schtinky" (Washington State) - See all my reviews

VINE™ VOICE

This review is from: The Chronicles of Riddick (Widescreen Unrated Director's Cut) (DVD)

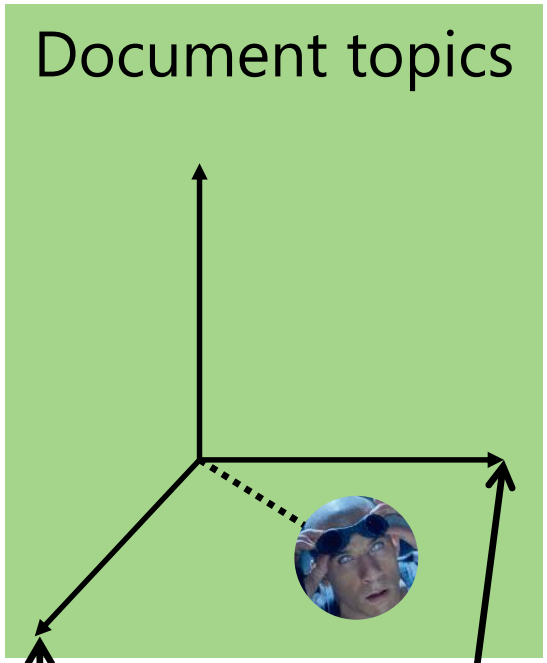
Even if I have to apologize to my Friends and Favorites, and my family, I have to admit that I really liked this movie. It's a Sci-Fi movie with a "Mad Maxx" appeal that, while changing many things, left Riddick from `Pitch Black' to be just Riddick. They did not change his attitude or soften him up or bring him out of his original character, which was very pleasing to `Pitch Black' fans like myself.

First off, let me say that when playing the DVD, the fi
Convert or Fight, and no explanation of the choices. T
I will mention off the bat that they are simply differen
menu has the very same options, simply different bac
either one and continue with the movie.

(review of "The Chronicles of Riddick")

topic model

Week 7!



Dimensionality reduction

Q3: How to represent connections in a social network?

A1: An adjacency matrix!

$$A = \begin{pmatrix} 1 & 0 & \cdots & 1 \\ 0 & 0 & & 1 \\ \vdots & & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

Dimensionality reduction

A1: An adjacency matrix

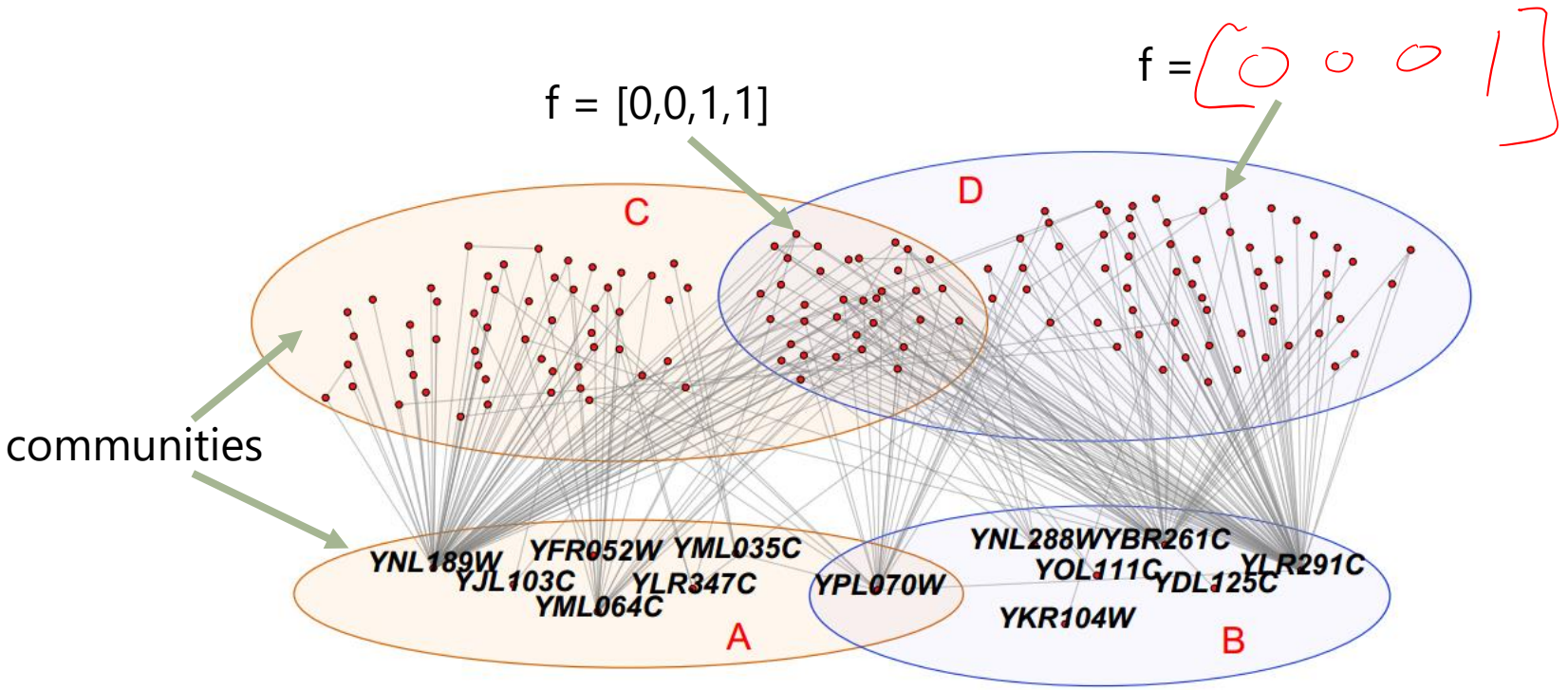
$$A = \begin{pmatrix} 1 & 0 & \cdots & 1 \\ 0 & 0 & & 1 \\ \vdots & & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

Seems almost reasonable, but…

- Becomes very large for real-world networks
- Very fine-grained – doesn't straightforwardly encode which nodes are similar to each other

Dimensionality reduction

A2: Represent each node/user in terms of the **communities** they belong to



e.g. from a PPI network; Yang, McAuley, & Leskovec (2014)

Why dimensionality reduction?

Goal: take **high-dimensional** data, and describe it compactly using a small number of dimensions

Assumption: Data lies (approximately) on some l**ow-dimensional** manifold space

(a few dimensions of opinions, a small number of topics, or a small number of communities)

Why dimensionality reduction?

Unsupervised learning

- Today our goal is not to solve some specific predictive task, but rather to **understand** the important features of a dataset
- We are not trying to understand the process which generated labels from the data, but rather the process which generated the data itself

Why dimensionality reduction?

Unsupervised learning

- **But!** The models we learn will prove useful when it comes to solving predictive tasks later on, e.g.

- **Q1:** If we want to predict which users like which movies, we need to understand the important dimensions of opinions
- **Q2:** To estimate the category of a news article (sports, politics, etc.), we need to understand topics it discusses
- **Q3:** To predict who will be friends (or enemies), we need to understand the communities that people belong to

Today…

Dimensionality reduction, clustering, and community detection

- Principal Component Analysis
- K-means clustering
- Hierarchical clustering
- Next lecture: Community detection
  - Graph cuts
  - Clique percolation
  - Network modularity

Principal Component Analysis

Principal Component Analysis (PCA) is one of the oldest (1901!) techniques to understand which dimensions of a high-dimensional dataset are "important"

Why?

- To **select** a few important features
- To **compress** the data by ignoring components which aren't meaningful

Principal Component Analysis

Motivating example:
Suppose we rate restaurants in terms of:
[value, service, quality, ambience, overall]

- Which dimensions are highly correlated (and how)?
- Which dimensions could we "throw away" without losing much information?
- How can we find which dimensions can be thrown away automatically?
- In other words, how could we come up with a "compressed representation" of a person's 5-d opinion into (say) 2-d?

Principal Component Analysis

Suppose our data/signal is an MxN matrix

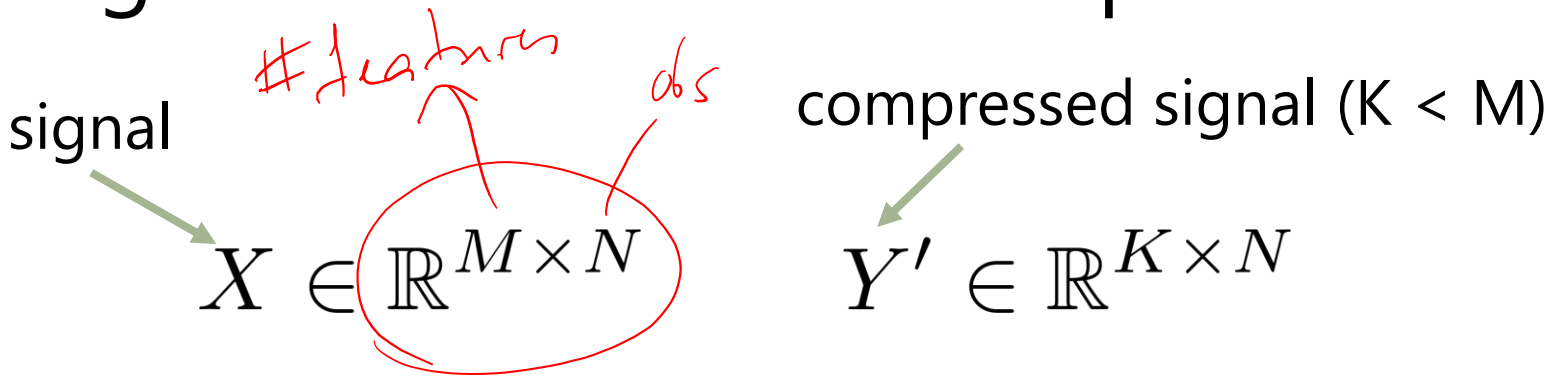
N = number of observations

$$X = \begin{pmatrix} 5 & 3 & \cdots & 1 \\ 4 & 2 & & 1 \\ 3 & 1 & & 3 \\ 2 & 2 & & 4 \\ 1 & 5 & & 2 \\ \vdots & & \ddots & \vdots \\ 1 & 2 & \cdots & 1 \end{pmatrix}$$

M = number of features
(each **column** is a data point)

Principal Component Analysis

We'd like (somehow) to recover this signal using as few dimensions as possible

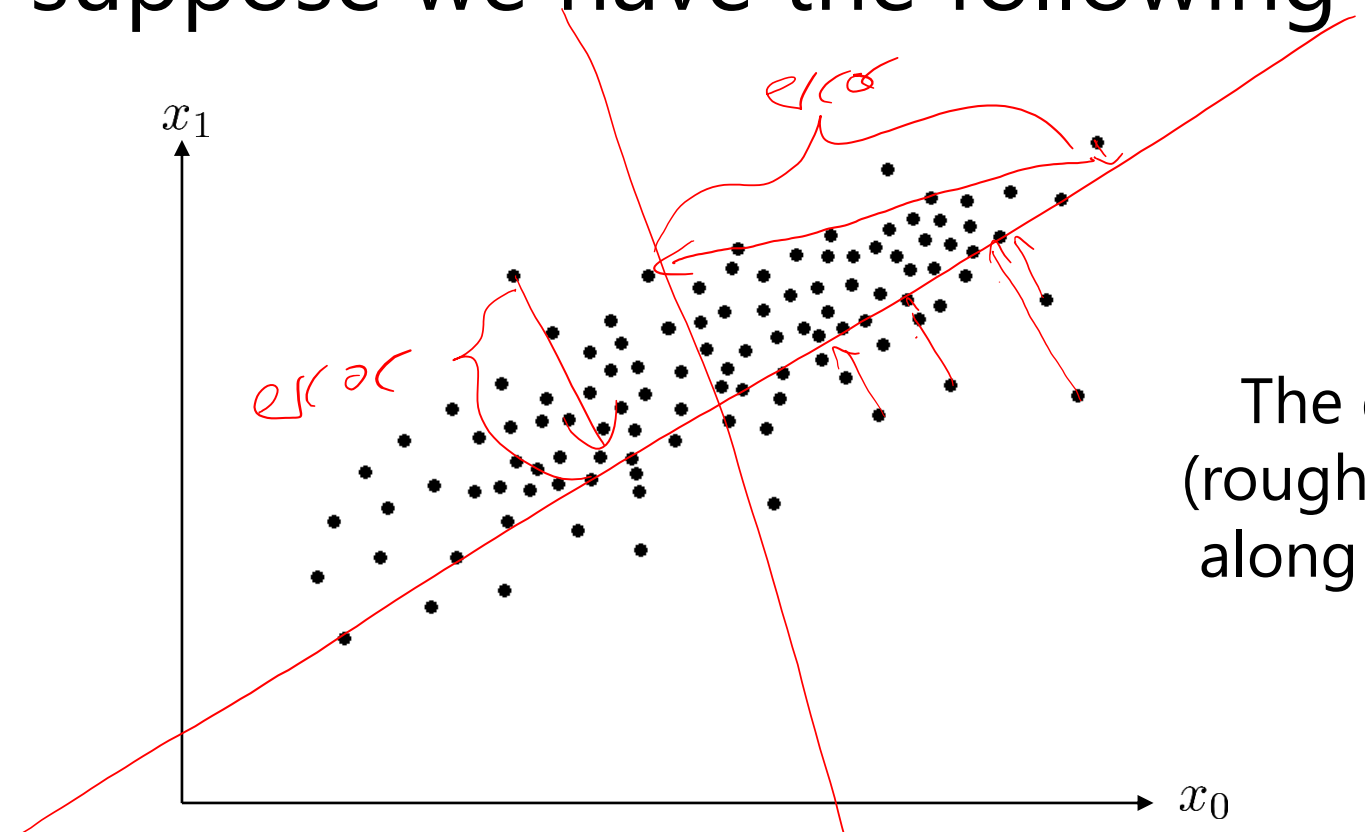


$$f(Y') \simeq X$$

(approximate) process to recover signal from its compressed version

Principal Component Analysis

E.g. suppose we have the following data:

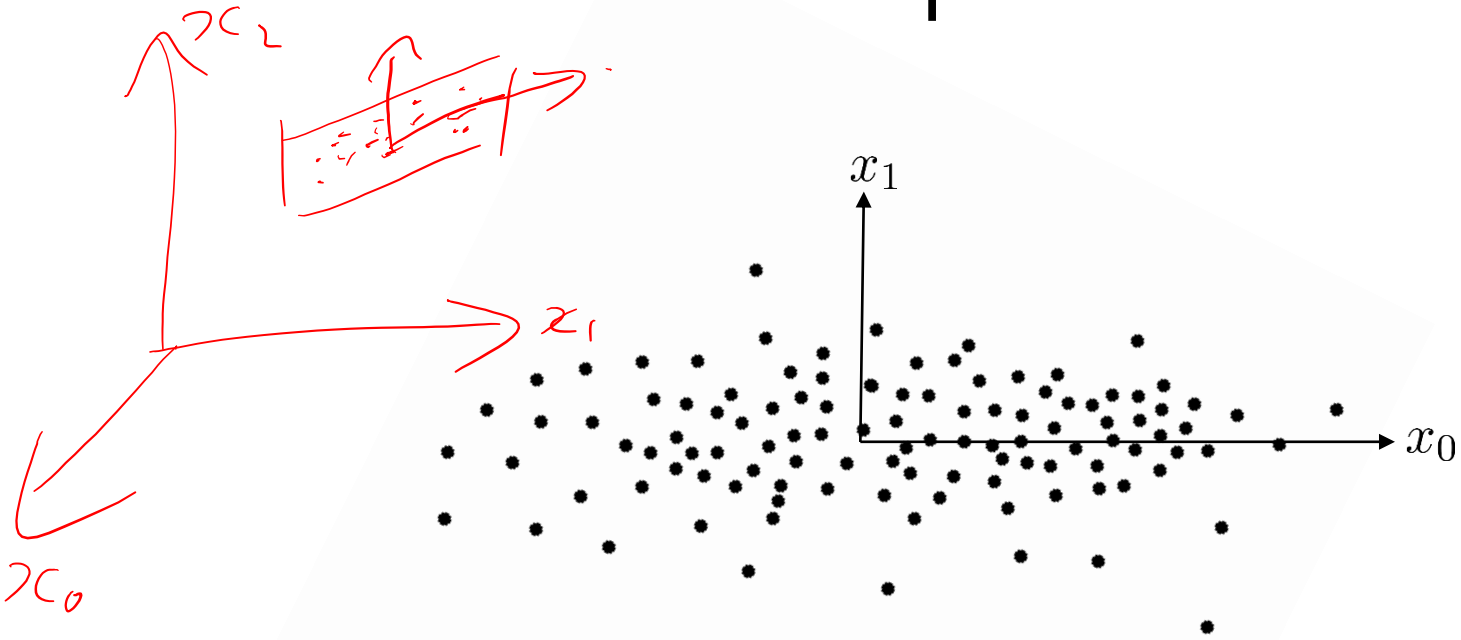


The data (roughly) lies along a line

Idea: if we know the position of the point on the line (1D), we can approximately recover the original (2D) signal

Principal Component Analysis

But how to find the important dimensions?



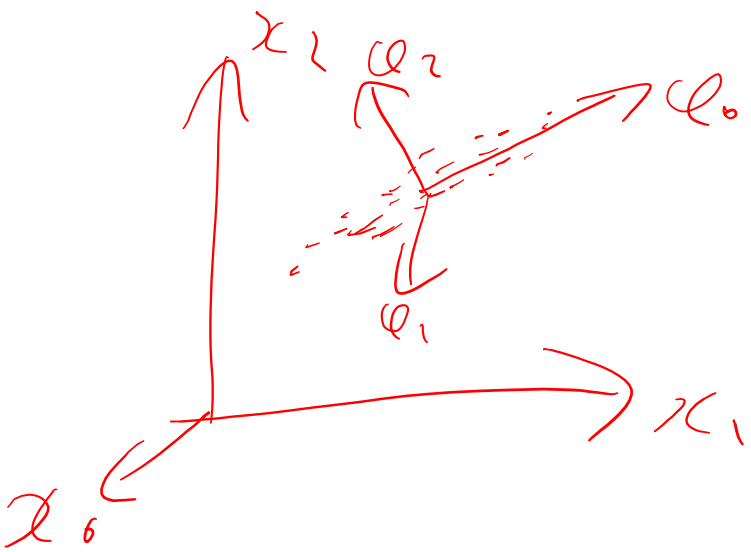
Find a new basis for the data (i.e., rotate it) such that

- **most** of the variance is along x0,
- most of the "leftover" variance (not explained by x0) is along x1,
- most of the leftover variance (not explained by x0,x1) is along x2,
- etc.

Principal Component Analysis

But how to find the important dimensions?

- Given an input $X \in \mathbb{R}^{M \times N}$
- Find a basis $\varphi \in \mathbb{R}^{M \times M}$



$\varphi_i \cdot \varphi_i = 0$

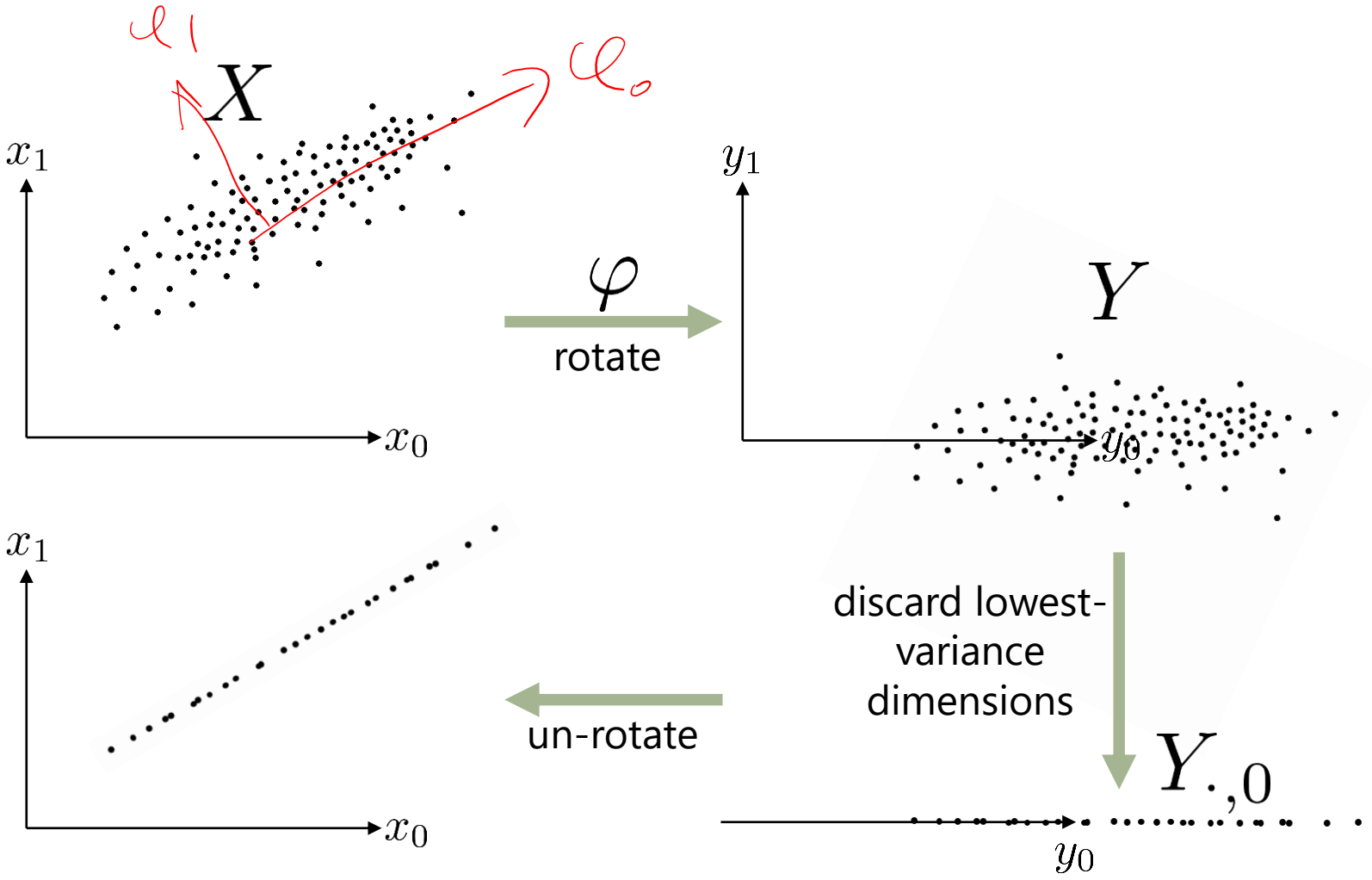
$\varphi_i \cdot \varphi_j = 1$

Principal Component Analysis

But how to find the important dimensions?

- Given an input $X \in \mathbb{R}^{M \times N}$
- Find a basis $\varphi \in \mathbb{R}^{M \times M}$
- Such that when X is rotated $(Y = \varphi X)$
  - Dimension with highest variance is y_0
  - Dimension with 2nd highest variance is y_1
  - Dimension with 3rd highest variance is y_2
  - Etc.

Principal Component Analysis



Principal Component Analysis

For a single data point: $y = \varphi x$ $x = \varphi^{-1} y = \varphi^T y$

"compress" "decompress"

$x = \varphi_1 y_1 + \varphi_2 y_2 \cdots \varphi_m y_m$

$x = \sum_j \varphi_j y_j$

Principal Component Analysis

$$x = \varphi_1 y_1 + \varphi_2 y_2 + \ldots + \varphi_M y_M = \sum_{j=1}^{M} \varphi_j y_j$$

$$x \simeq \sum_{j=1}^{K} \varphi_j y_j + \sum_{j=K+1}^{M} \varphi_j b_j$$

Principal Component Analysis

We want to fit the "best" reconstruction:

$$x = \underbrace{\varphi^T y}_{\text{"complete" reconstruction}} \qquad x_i \simeq \underbrace{\sum_{j=1}^{K} \varphi_j y_j + \sum_{j=K+1}^{M} \varphi_j b_j}_{\text{approximate reconstruction}}$$

i.e., it should minimize the **MSE**:

$$\min_{\varphi, b} \frac{1}{N} \sum_{y} \left\| \sum_{j=1}^{K} \varphi_j y_j + \sum_{j=K+1}^{M} \varphi_j b_j - x_i \right\|_2^2$$

$$\sum_{j=1}^{M} \varphi_j y_j$$

Principal Component Analysis

$$\min_{\varphi,b} \frac{1}{N} \sum_{y} \left\| \sum_{j=1}^{K} \varphi_j y_j + \sum_{j=K+1}^{M} \varphi_j b_j - \varphi^T y \right\|_2^2$$

Simplify…

$$\frac{1}{N} \sum_{y} \left\| \sum_{j=K+1}^{M} \varphi_j (b_j - y_j) \right\|_2^2$$

Principal Component Analysis

$$\min_{\varphi,b} \frac{1}{N} \sum_{y} \left\| \sum_{j=K+1}^{M} \varphi_j (y_j - b_j) \right\|_2^2$$

$\|x\|_2^2 = x^T x = \sum_i x_i^2$

Expand…

$$\frac{1}{N} \sum_{y} \sum_{i=K+1}^{M} \sum_{j=K+1}^{M} (y_i - b_i) \varphi_i^T \varphi_j (y_j - b_j)$$

$i \neq j = 0$

$i = j = 1$

$$\frac{1}{N} \sum_{y} \sum_{j=K+1}^{M} (y_j - b_j)^2$$

Principal Component Analysis

$$\min_{\varphi,b} \frac{1}{N} \sum_{y} \sum_{j=K+1}^{M} (y_j - b_j)^2$$

$$b_j = \overline{y_j}$$

Principal Component Analysis

$$\min_{\varphi} \frac{1}{N} \sum_{y} \sum_{j=K+1}^{M} (y_j - \bar{y}_j)^2$$

Equal to the **variance** in the discarded dimensions

Principal Component Analysis

PCA: We want to keep the dimensions with the highest variance, and discard the dimensions with the lowest variance, in some sense to maximize the amount of "randomness" that gets preserved when we compress the data

Principal Component Analysis

$$\min_{\varphi} \frac{1}{N} \sum_{y} \sum_{j=K+1}^{M} (y_j - \bar{y}_j)^2$$

(subject to φ orthonormal)

$y = \varphi x$

Expand in terms of X

$$\min_{\varphi} \frac{1}{N} \sum_{j=K+1}^{M} \varphi_j (X - \bar{X})(X - \bar{X})^T \varphi_j^T$$

(subject to φ orthonormal)

Principal Component Analysis

$$\min_{\varphi} \frac{1}{N} \sum_{j=K+1}^{M} \varphi_j \overbrace{(X-\bar{X})(X-\bar{X})^T}^{\mathrm{Cov}(X)} \varphi_j^T$$

(subject to φ orthonormal)

$\varphi_j \cdot \varphi_j = 1$

Lagrange multiplier

$$\min_{\varphi} \frac{1}{N} \sum_{j=K+1}^{M} \varphi_j \mathrm{Cov}(X) \varphi_j^T - \lambda_j(\varphi_j \varphi_j^T - 1)$$

Lagrange multipliers: Bishop appendix E

Principal Component Analysis

Solve:

$$\frac{\partial}{\partial \varphi_j} \sum_{j=K+1}^{M} \varphi_j \mathrm{Cov}(X) \varphi_j^T - \lambda_j (\varphi_j \varphi_j^T - 1) = 0$$

(Cov(X) is symmetric)

$$2(\mathrm{Cov}(X) \varphi_j^T - \lambda_j \varphi_j^T) = 0$$

- This expression can only be satisfied if phi_j and lambda_j are an **eigenvectors/eigenvalues** of the covariance matrix
- So to minimize the original expression we'd discard phi_j's corresponding to the **smallest** eigenvalues

Principal Component Analysis

Moral of the story: if we want to optimally (in terms of the MSE) project some data into a low dimensional space, we should choose the projection by taking the **eigenvectors** corresponding to the **largest eigenvalues of the covariance matrix**

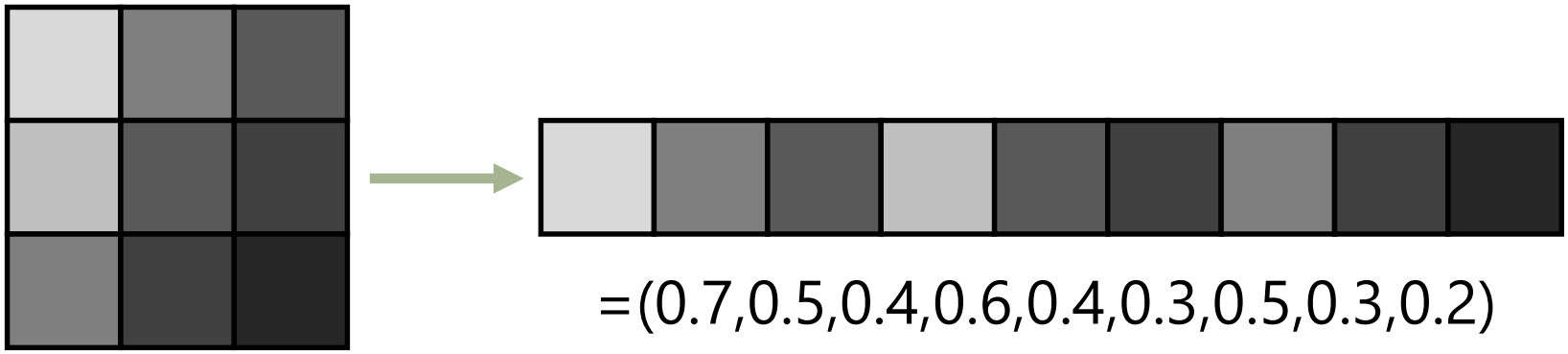
Principal Component Analysis

Example 1:
What are the principal components of people's opinions on beer?

(code available on)
http://jmcauley.ucsd.edu/cse158/code/week3.py

Principal Component Analysis

Example 2:
What are the principal dimensions of image patches?



=(0.7,0.5,0.4,0.6,0.4,0.3,0.5,0.3,0.2)

Principal Component Analysis

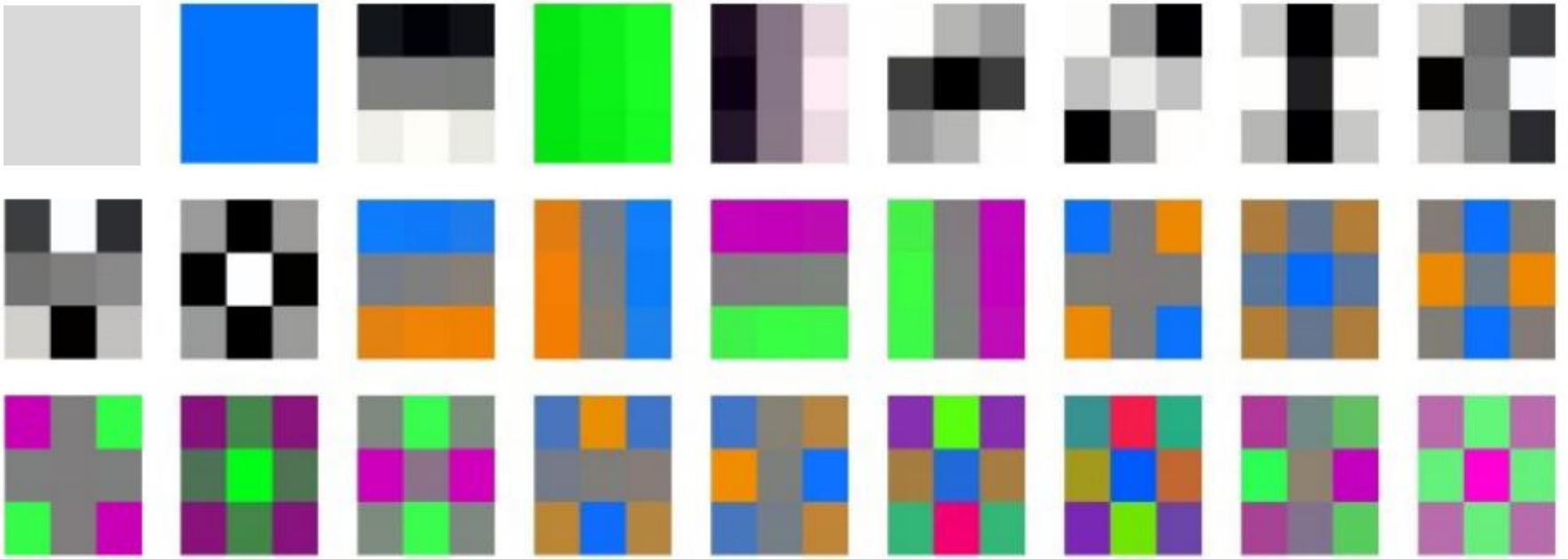
Construct such vectors from 100,000 patches from real images and run PCA:

Black and white:

Principal Component Analysis

Construct such vectors from 100,000 patches from real images and run PCA:

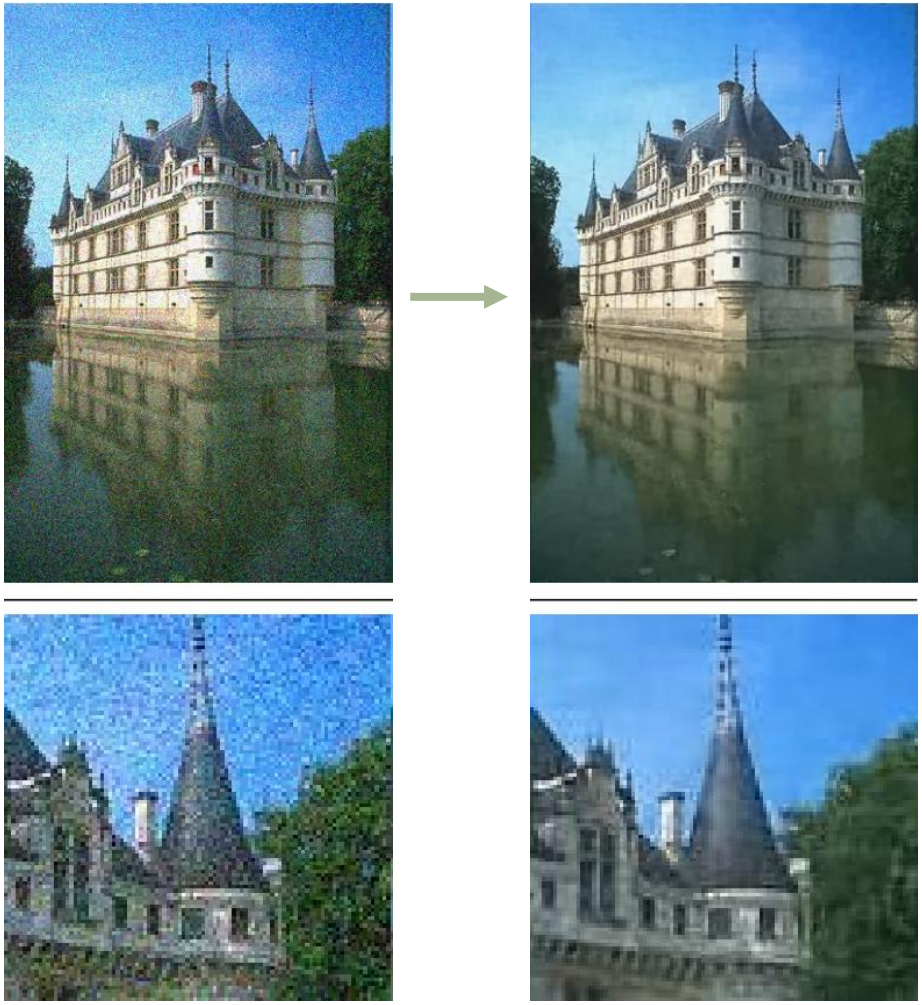
Color:



Principal Component Analysis

From this we can build an algorithm to "denoise" images

Idea: image patches should be **more like** the high-eigenvalue components and **less like** the low-eigenvalue components



input output

McAuley et. al (2006)

Principal Component Analysis

- We want to find a low-dimensional representation that best compresses or "summarizes" our data
- To do this we'd like to keep the dimensions with the highest variance (we proved this), and discard dimensions with lower variance. Essentially we'd like to capture the aspects of the data that are "hardest" to predict, while discard the parts that are "easy" to predict
- This can be done by taking the eigenvectors of the covariance matrix (we didn't prove this, but it's right there in the slides)

CSE 158 – Lecture 5

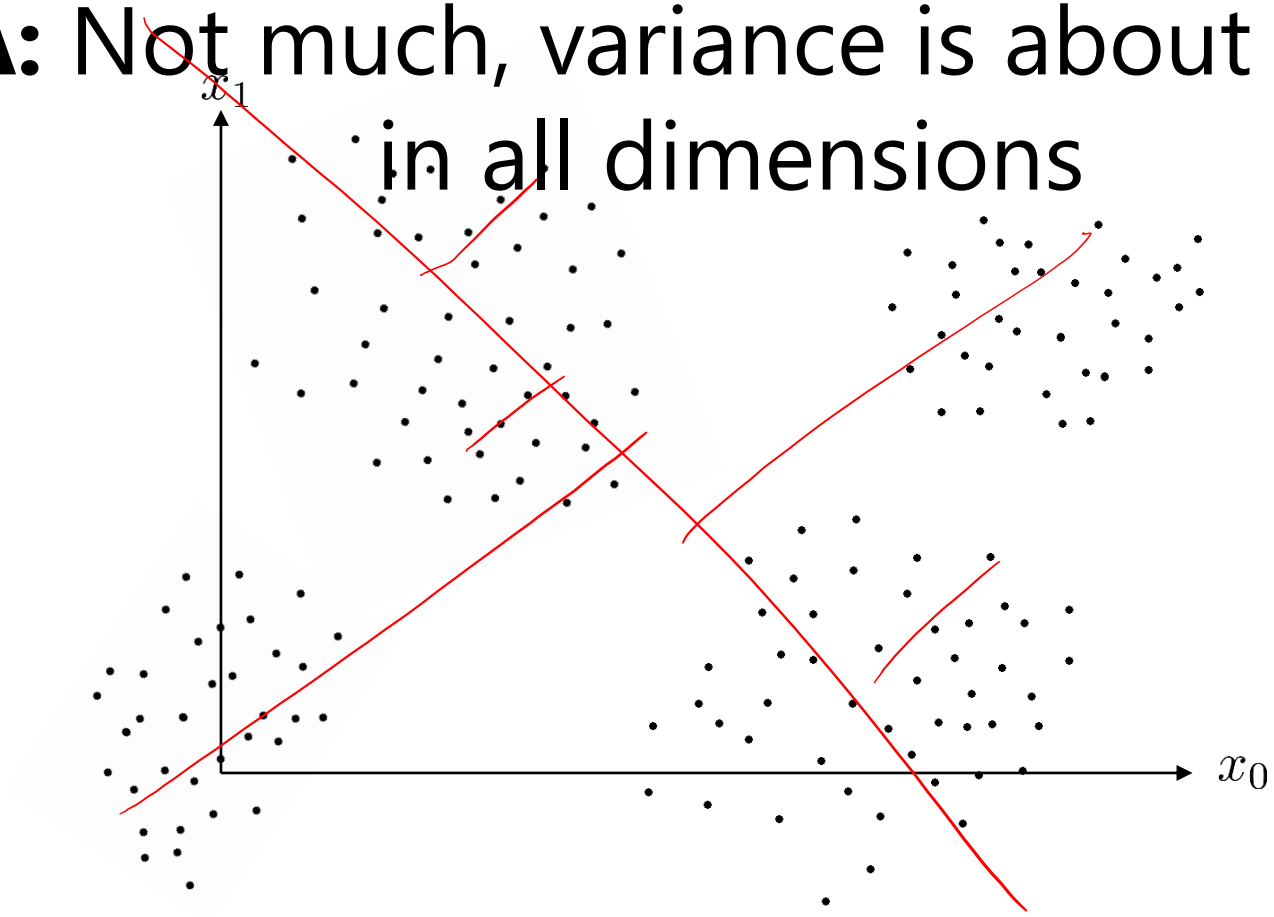
Web Mining and Recommender Systems

Clustering – K-means

Clustering

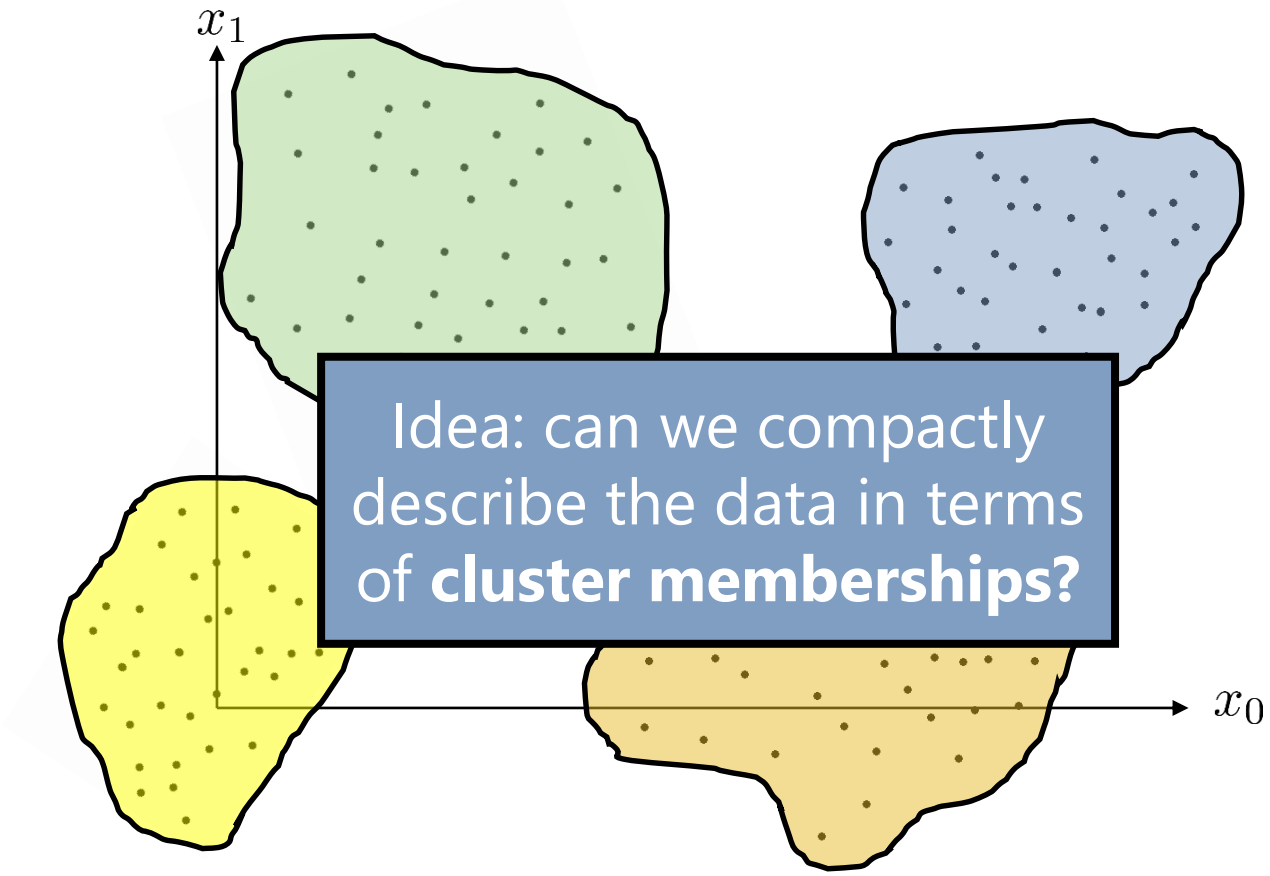
Q: What would PCA do with this data?

A: Not much, variance is about equal in all dimensions



Clustering

But: The data are highly **clustered**

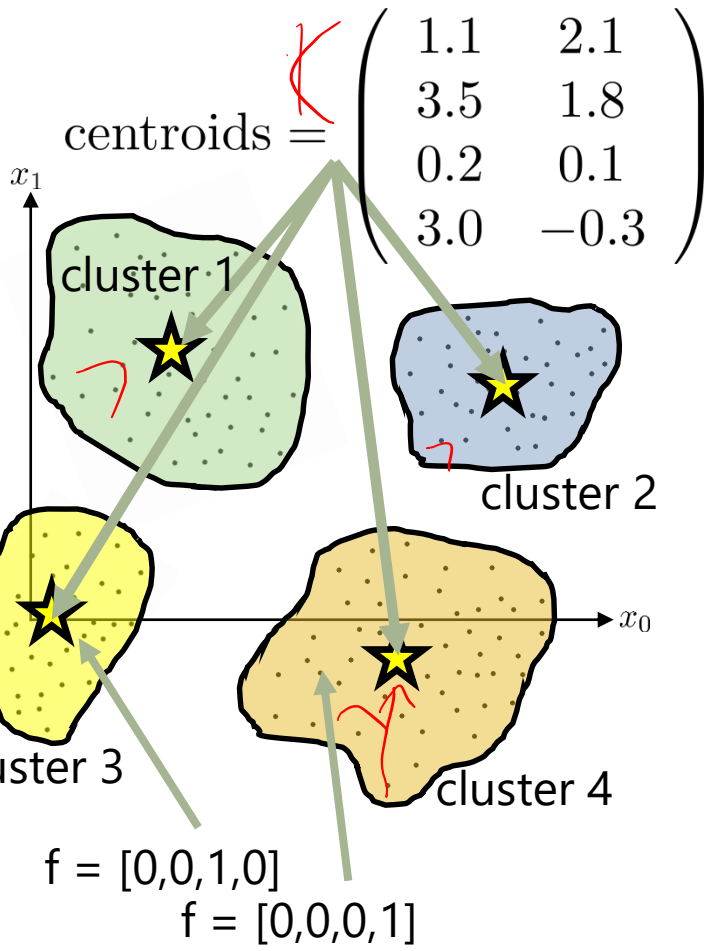


K-means Clustering

1. Input is still a matrix of features:

$$X = \begin{pmatrix} 5 & 3 & \cdots & 1 \\ 4 & 2 & & 1 \\ 3 & 1 & & 3 \\ 2 & 2 & & 4 \\ 1 & 5 & & 2 \\ \vdots & & \ddots & \vdots \\ 1 & 2 & \cdots & 1 \end{pmatrix}$$

2. Output is a list of cluster "centroids":



3. From this we can describe each point in X by its cluster membership:

$$Y = (1, 2, 4, 3, 4, 2, 4, 2, 2, 3, 3, 2, 1, 1, 3, \ldots, 2)$$

K-means Clustering

Given **features** (X) our goal is to choose K **centroids** (C) and **cluster assignments** (Y) so that the reconstruction error is minimized

Number of data points

$$X \in \mathbb{R}^{N \times M}$$

Feature dimensionality

Number of clusters

$$C \in \mathbb{R}^{K \times M}$$

$$Y \in \{1 \ldots K\}^{N}$$

$$\text{reconstruction error} = \sum_i \|X_i - C_{y_i}\|_2^2$$

(= sum of squared distances from assigned centroids)

K-means Clustering

Q: Can we solve this optimally?

$$\min_{C,y} \sum_i \|X_i - C_{y_i}\|_2^2$$

A: No. This is (in general) an **NP-Hard** optimization problem

See *"NP-hardness of Euclidean sum-of-squares clustering"*, Aloise et. Al (2009)

K-means Clustering

Greedy algorithm:

```
1. Initialize C (e.g. at random)
2. Do
3.       Assign each X_i to its nearest centroid
4.       Update each centroid to be the mean
         of points assigned to it
5. While (assignments change between iterations)
```

$$y_i = \arg\min_k \|X_i - C_k\|_2^2$$

$$C_k = \frac{\sum_i \delta(y_i=k) X_i}{\sum_i \delta(y_i=k)}$$

(also: reinitialize clusters at random should they become empty)

K-means Clustering

Further reading:

- K-**medians**: Replaces the mean with the meadian. Has the effect of minimizing the 1-norm (rather than the 2-norm) distance
- **Soft** K-means: Replaces "hard" memberships to each cluster by a proportional membership to each cluster

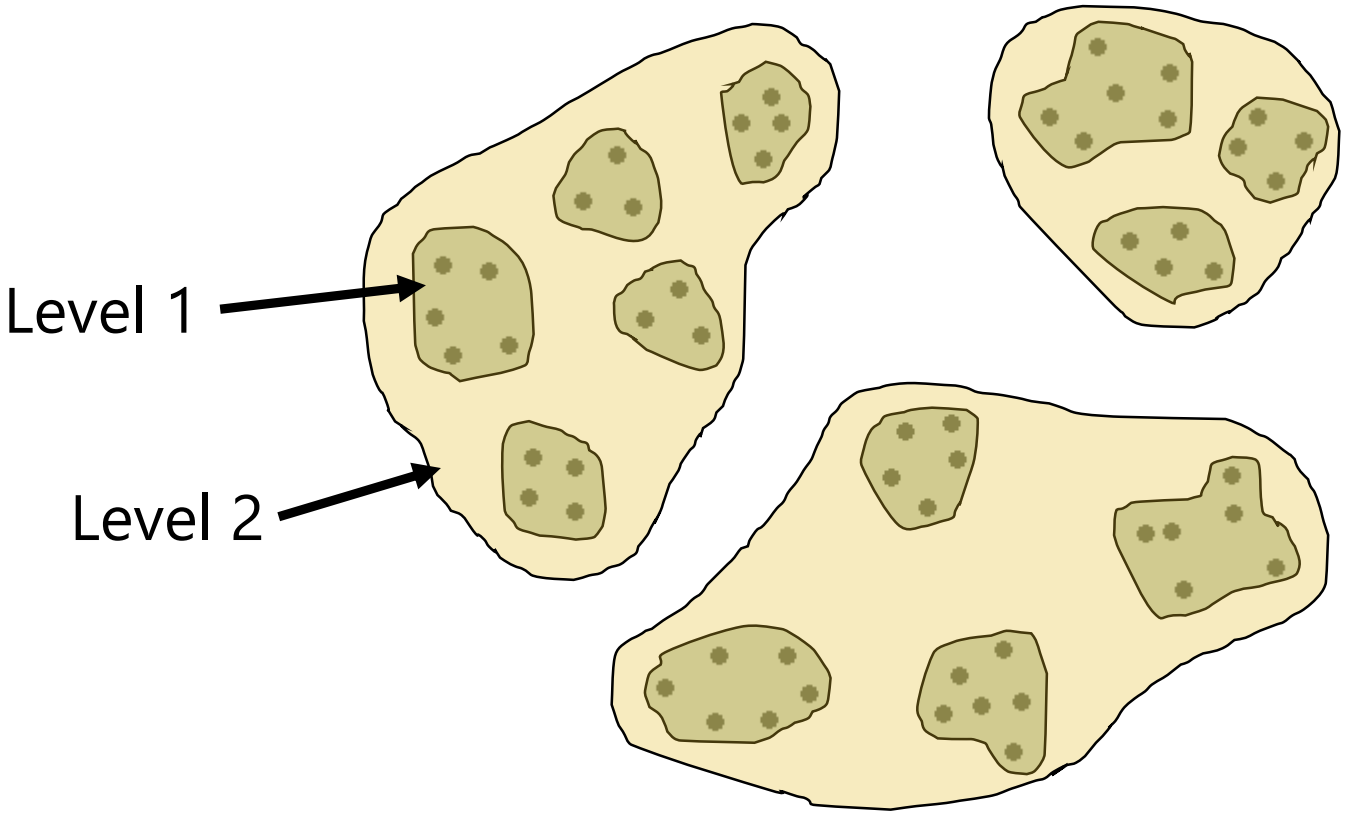
CSE 158 – Lecture 5

Web Mining and Recommender Systems

Clustering – hierarchical clustering

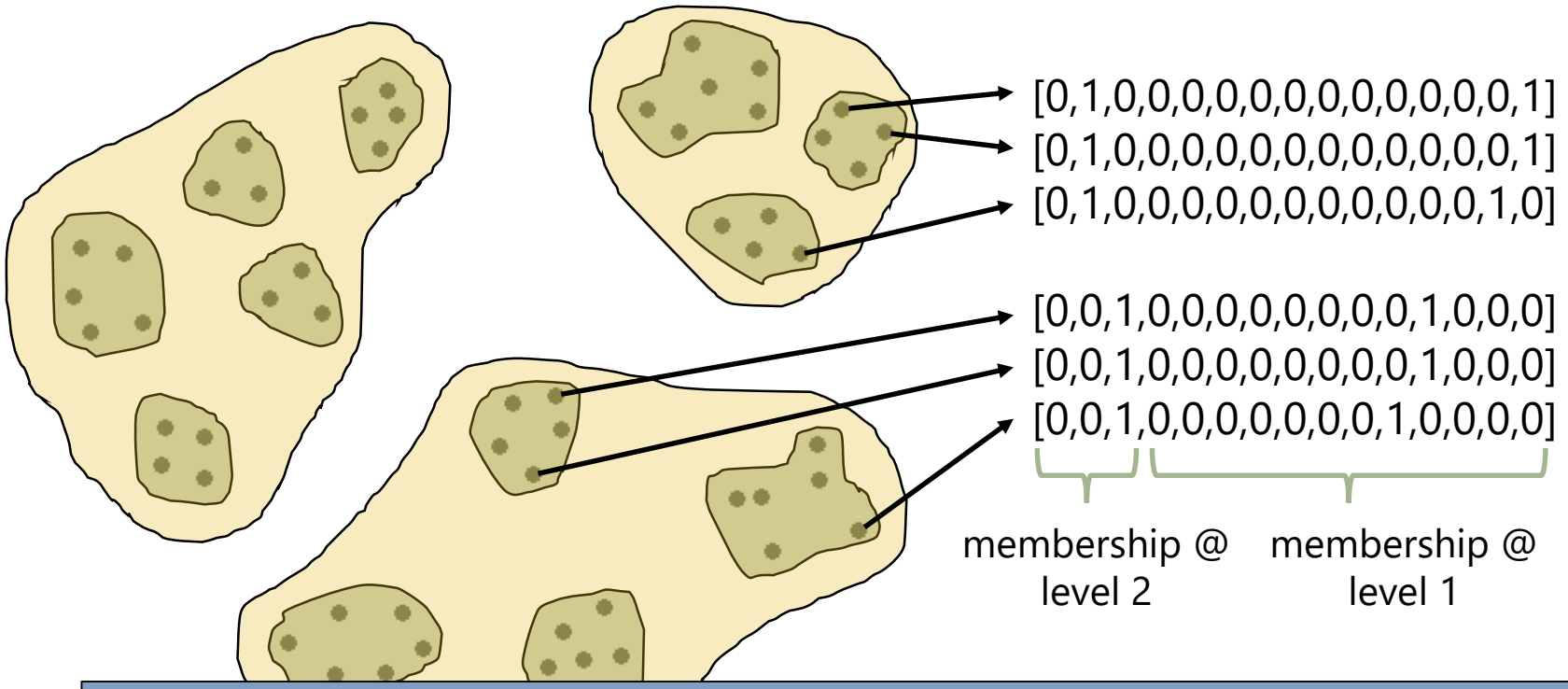
Hierarchical clustering

Q: What if our clusters are **hierarchical?**



Hierarchical clustering

Q: What if our clusters are **hierarchical?**



A: We'd like a representation that encodes that points have **some features** in common but not others

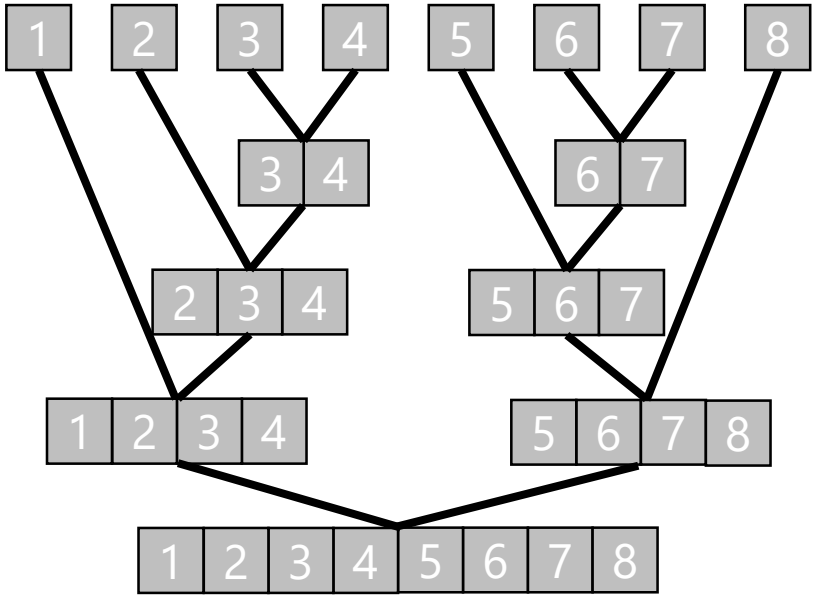
Hierarchical clustering

Hierarchical (agglomerative) clustering works by gradually fusing clusters whose points are closest together

```
Assign every point to its own cluster:
Clusters = [[1],[2],[3],[4],[5],[6],…,[N]]
While len(Clusters) > 1:
        Compute the center of each cluster
        Combine the two clusters with the nearest centers
```

Hierarchical clustering

If we keep track of the order in which clusters were merged, we can build a "hierarchy" of clusters



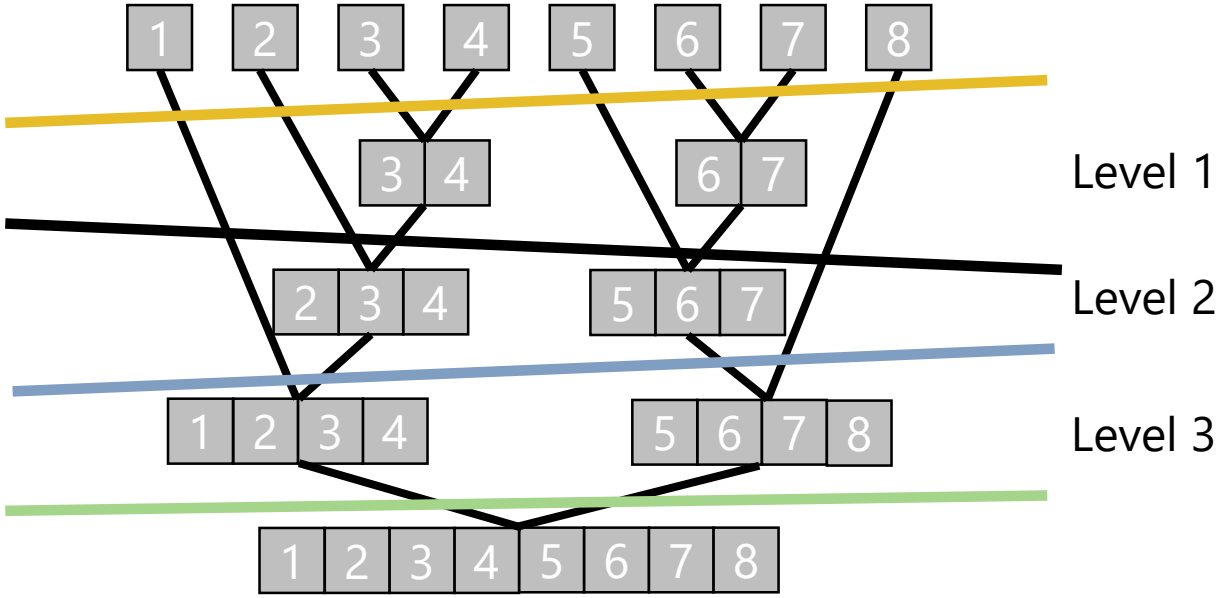
("dendrogram")

Hierarchical clustering

Splitting the dendrogram at different points defines cluster “levels” from which we can build our feature representation

L1, L2, L3

1: [0,0,0,0,1,0]
2: [0,0,1,0,1,0]
3: [1,0,1,0,1,0]
4: [1,0,1,0,1,0]
5: [0,0,0,1,0,1]
6: [0,1,0,1,0,1]
7: [0,1,0,1,0,1]
8: [0,0,0,0,0,1]



Model selection

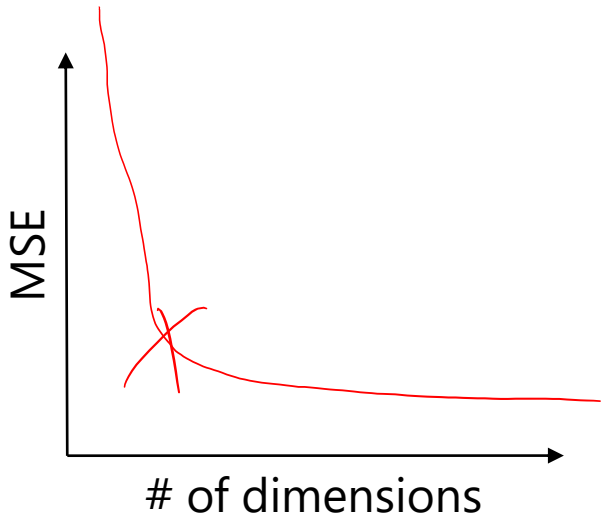
- **Q:** How to choose K in K-means?

(or:

- How to choose how many PCA dimensions to keep?
- How to choose at what position to “cut” our hierarchical clusters?
- (next week) how to choose how many communities to look for in a network)

Model selection

1) As a means of "compressing" our data
   - Choose however many dimensions we can afford to obtain a given file size/compression ratio
   - Keep adding dimensions until adding more no longer decreases the reconstruction error significantly



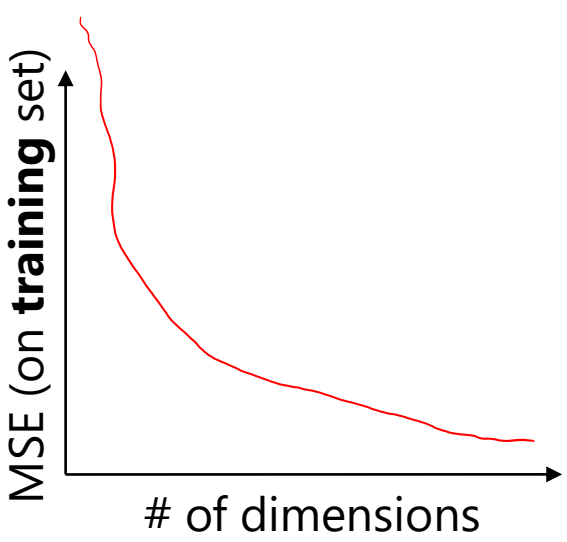
Model selection

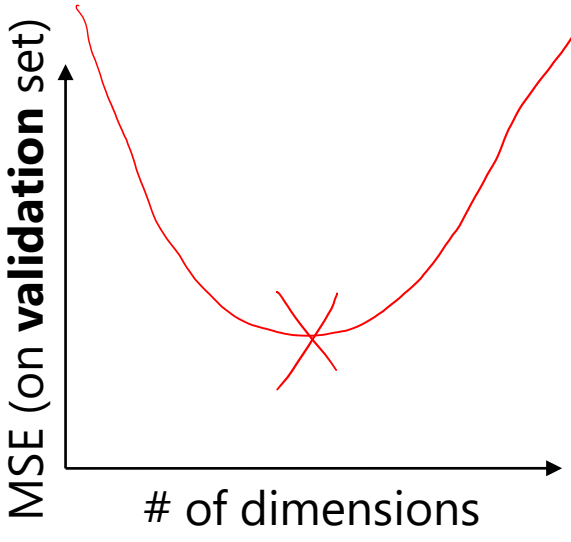
2) As a means of generating potentially useful features for some other predictive task (which is what we're more interested in in a predictive analytics course!)

- Increasing the number of dimensions/number of clusters gives us additional features to work with, i.e., a longer feature vector
- In some settings, we may be running an algorithm whose complexity (either time or memory) scales with the feature dimensionality (such as we saw last week!); in this case we would just take however many dimensions we can afford

Model selection

- Otherwise, we should choose however many dimensions results in the best prediction performance **on held out data**





- **Q:** Why does this happen? i.e., why doesn't the validation performance continue to improve with more dimensions

Questions?

Further reading:

- Ricardo Gutierrez-Osuna's PCA slides (slightly more mathsy than mine):
  http://research.cs.tamu.edu/prism/lectures/pr/pr_l9.pdf
- Relationship between PCA and K-means:
  http://ranger.uta.edu/~chqding/papers/KmeansPCA1.pdf
  http://ranger.uta.edu/~chqding/papers/Zha-Kmeans.pdf