Machine Learning Department
School of Computer Science
Carnegie Mellon University

PCA
+
AdaBoost

Matt Gormley
Lecture 30
April 27, 2018

Reminders

- Homework 8: Reinforcement Learning
  - Out: Tue, Apr 17
  - Due: Fri, Apr 27 at 11:59pm
- Homework 9: Learning Paradigms
  - Out: Sat, Apr 28
  - Due: Fri, May 4 at 11:59pm

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)


The Tribune Prints the News

Hilo Tribune.

We Lead—Others Follow

KILAUEA CRATER CONTINUES IN ERUPTION

FOUR DAYS OF ACTIVITY

ANOTHER OUTBREAK OCCURS

HONOLULU GETS APPROPRIATIONS FOR HARBOR AND LIGHTHOUSE

Drs. Grace and Irwin

Chas. M. LeBlond

Ridgway & Ridgway

River and Harbor Bill Passes.

Postmaster Pratt Confirmed.

Latest Sugar Quotations.

Fire At New Orleans.


叻報

大清光緒十六年


Македония

PRAVA BRAĆA

Истински братя

DEKLARACIJA.

Декларация


குடி அரசு

High Dimension Data

Examples of high dimensional data:

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

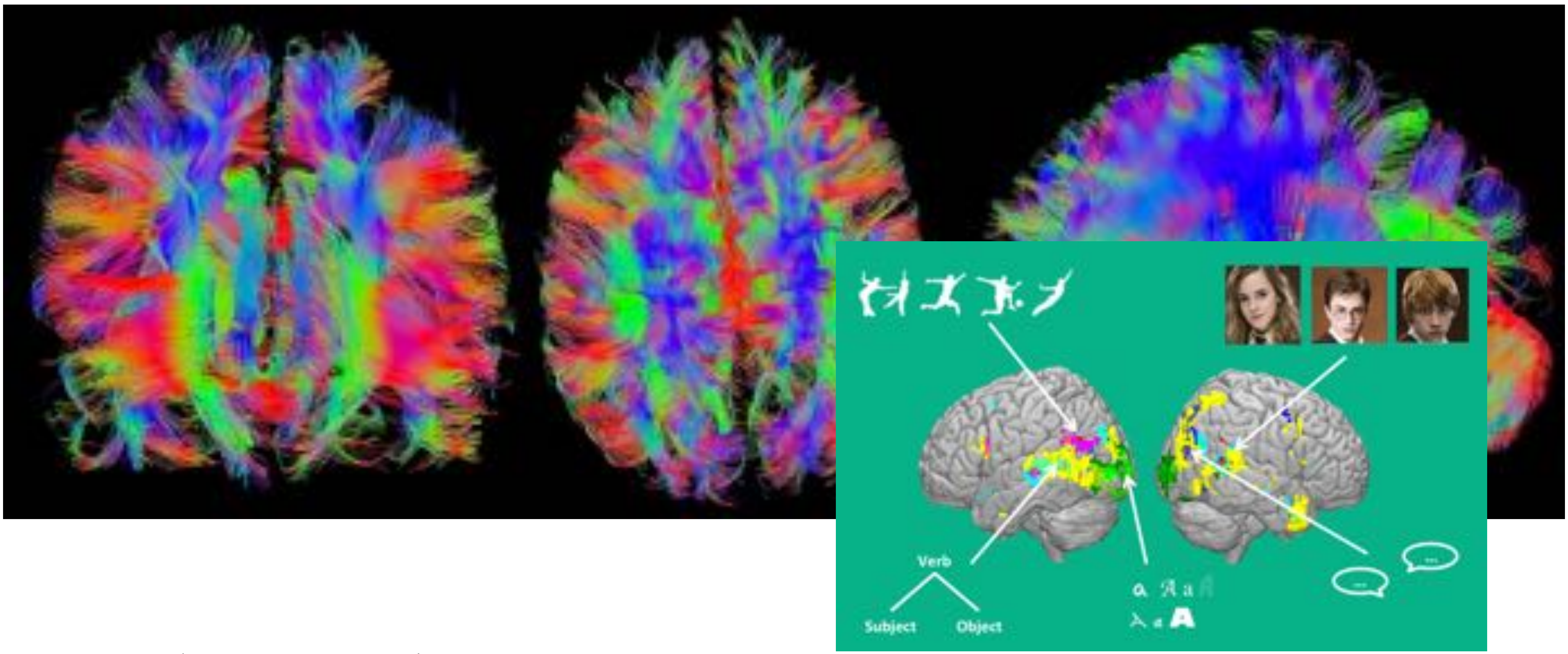


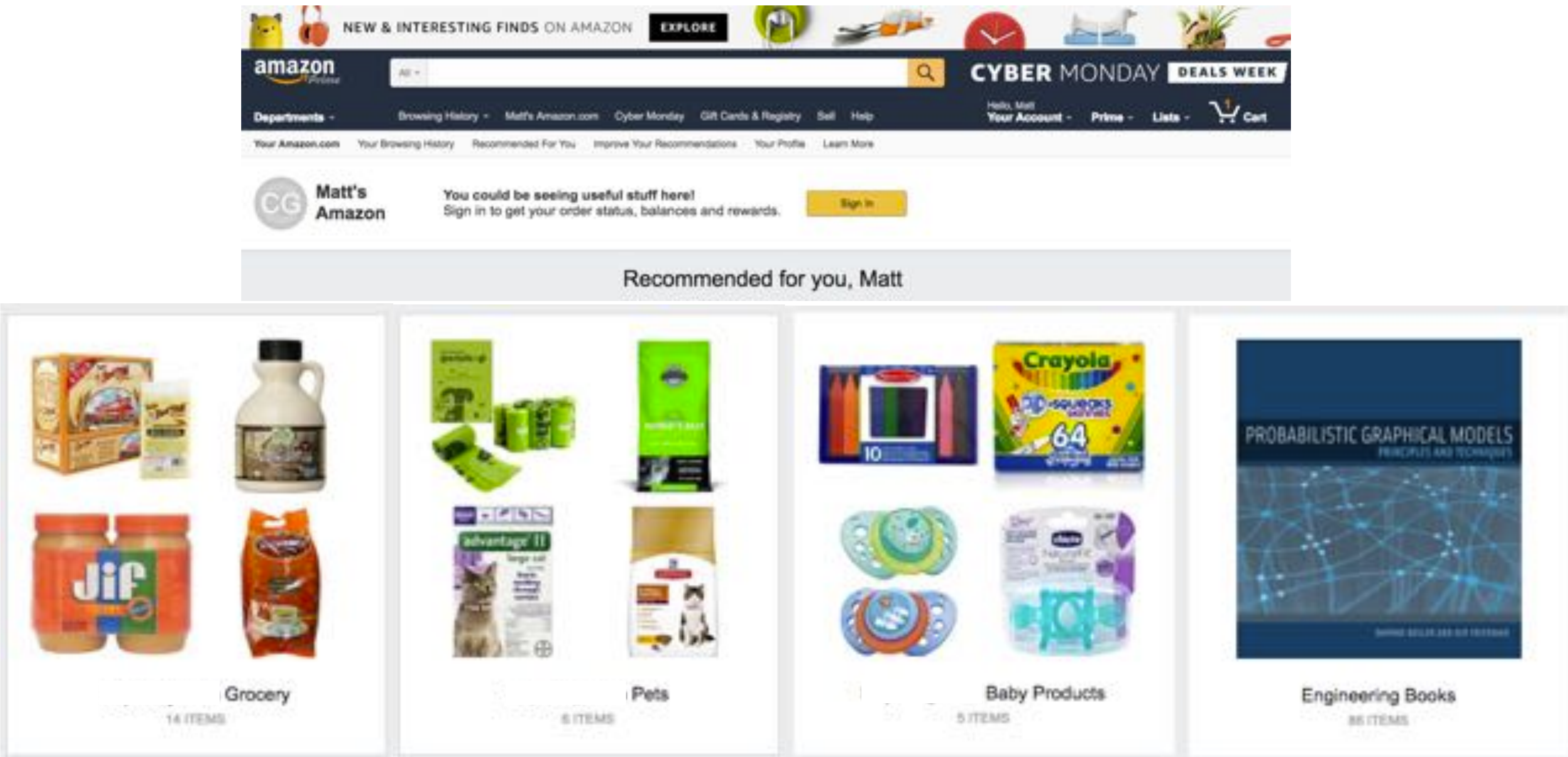
Image from (Wehbe et al., 2014)

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

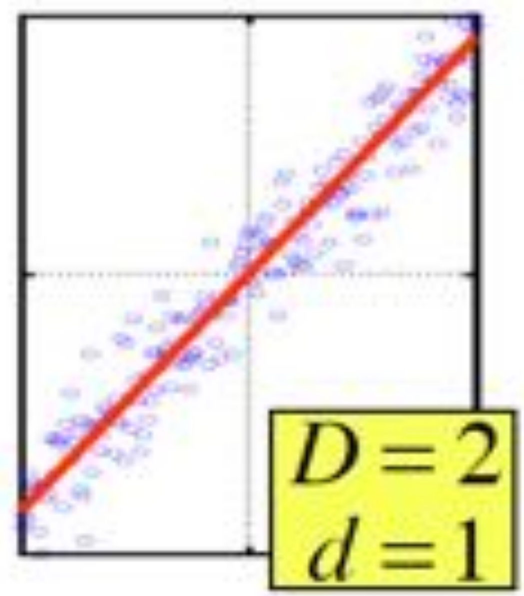
Photo from https://www.springcarnival.org/booth.shtml

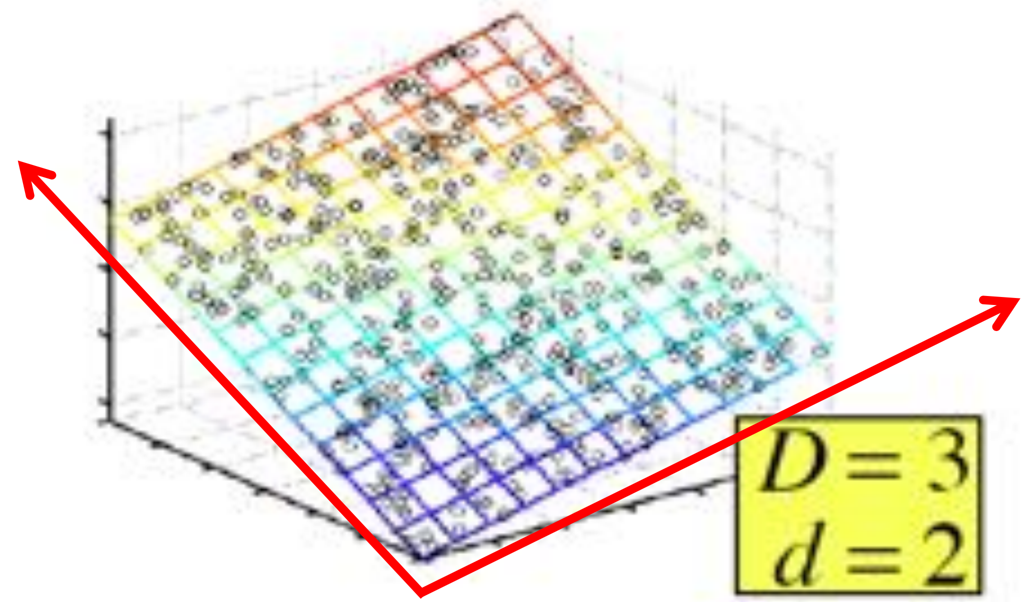
PRINCIPAL COMPONENT ANALYSIS (PCA)

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)



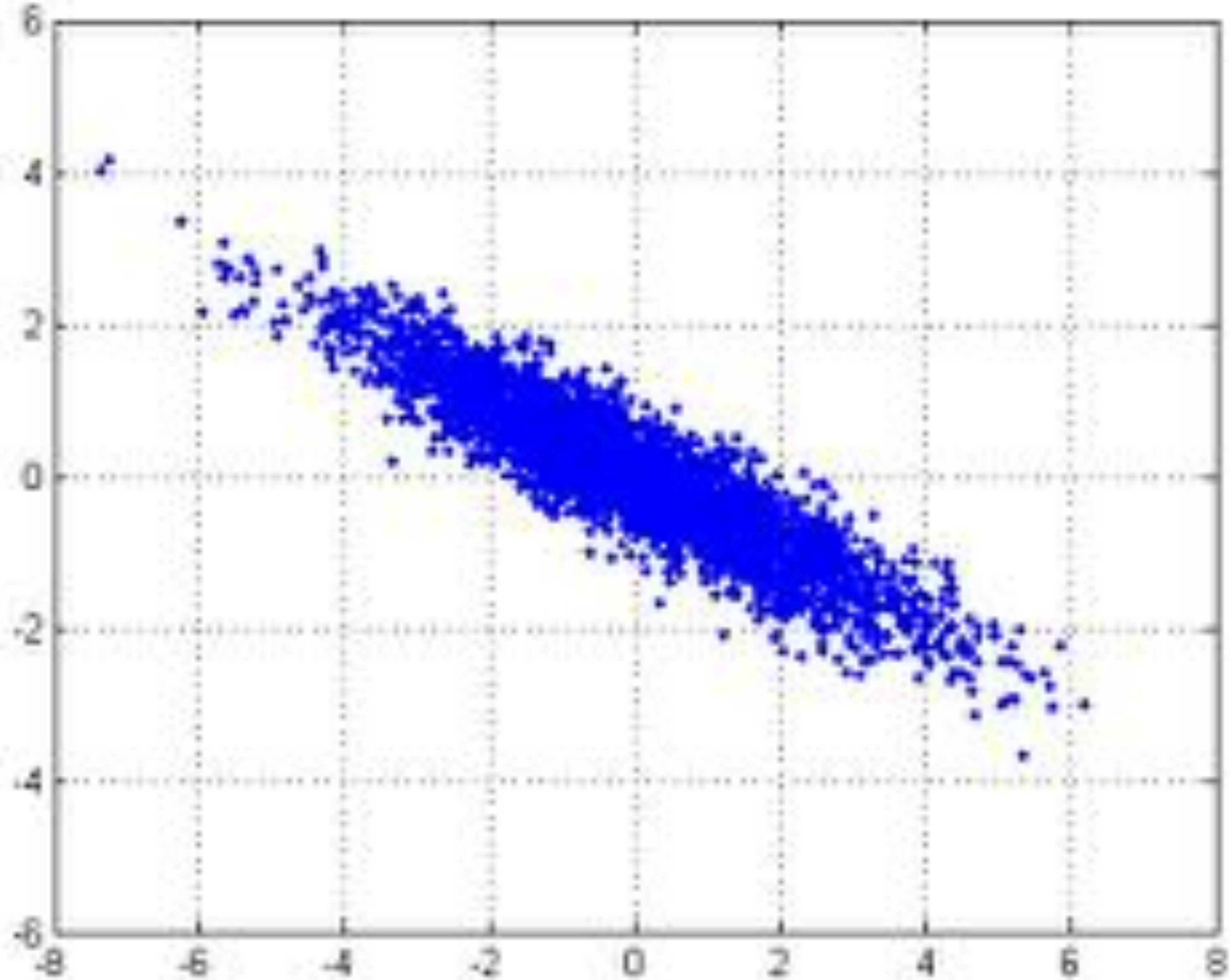


In case where data lies on or near a low d-dimensional linear subspace, axes of this subspace are an effective representation of the data.

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

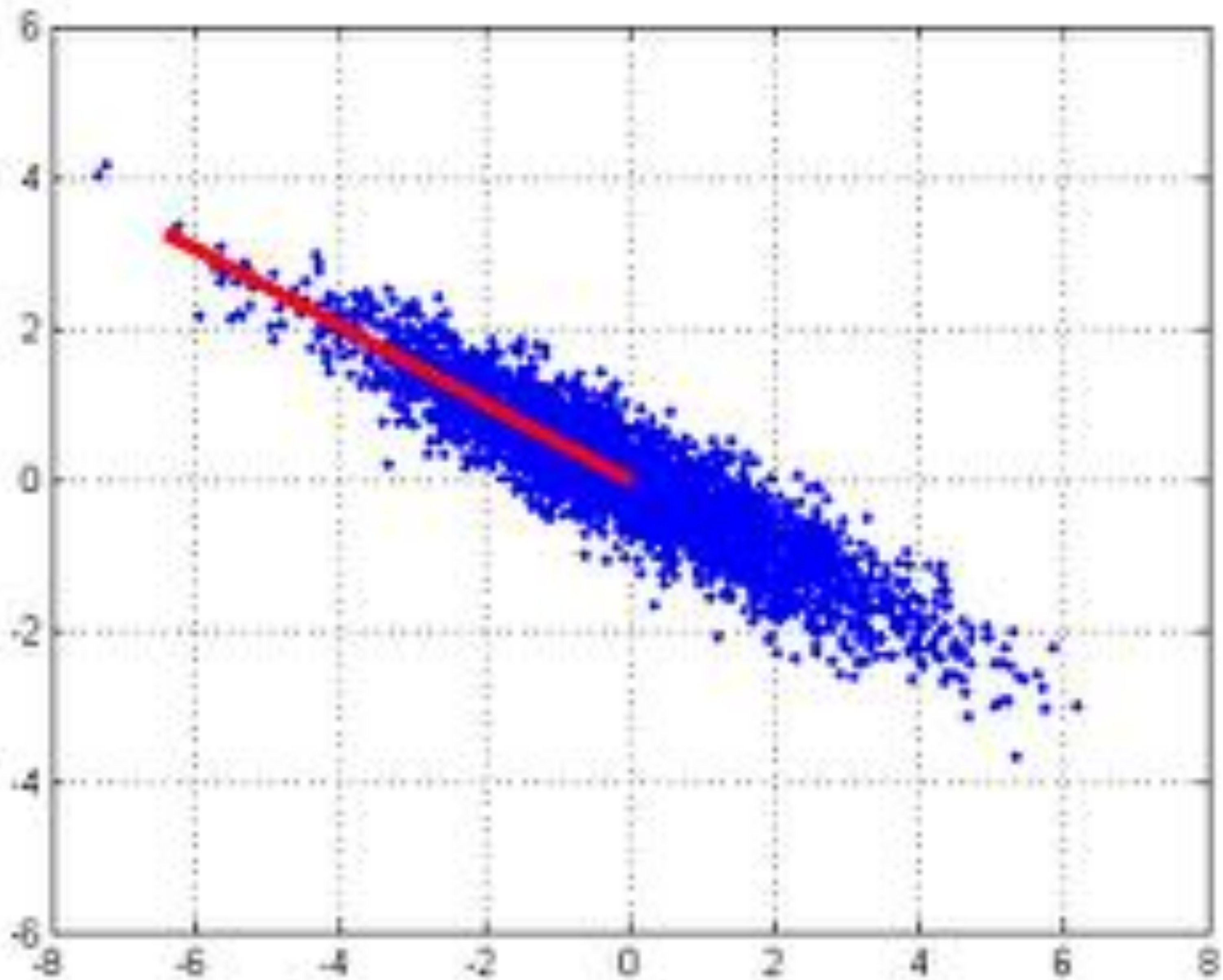
Slide from Nina Balcan

2D Gaussian dataset



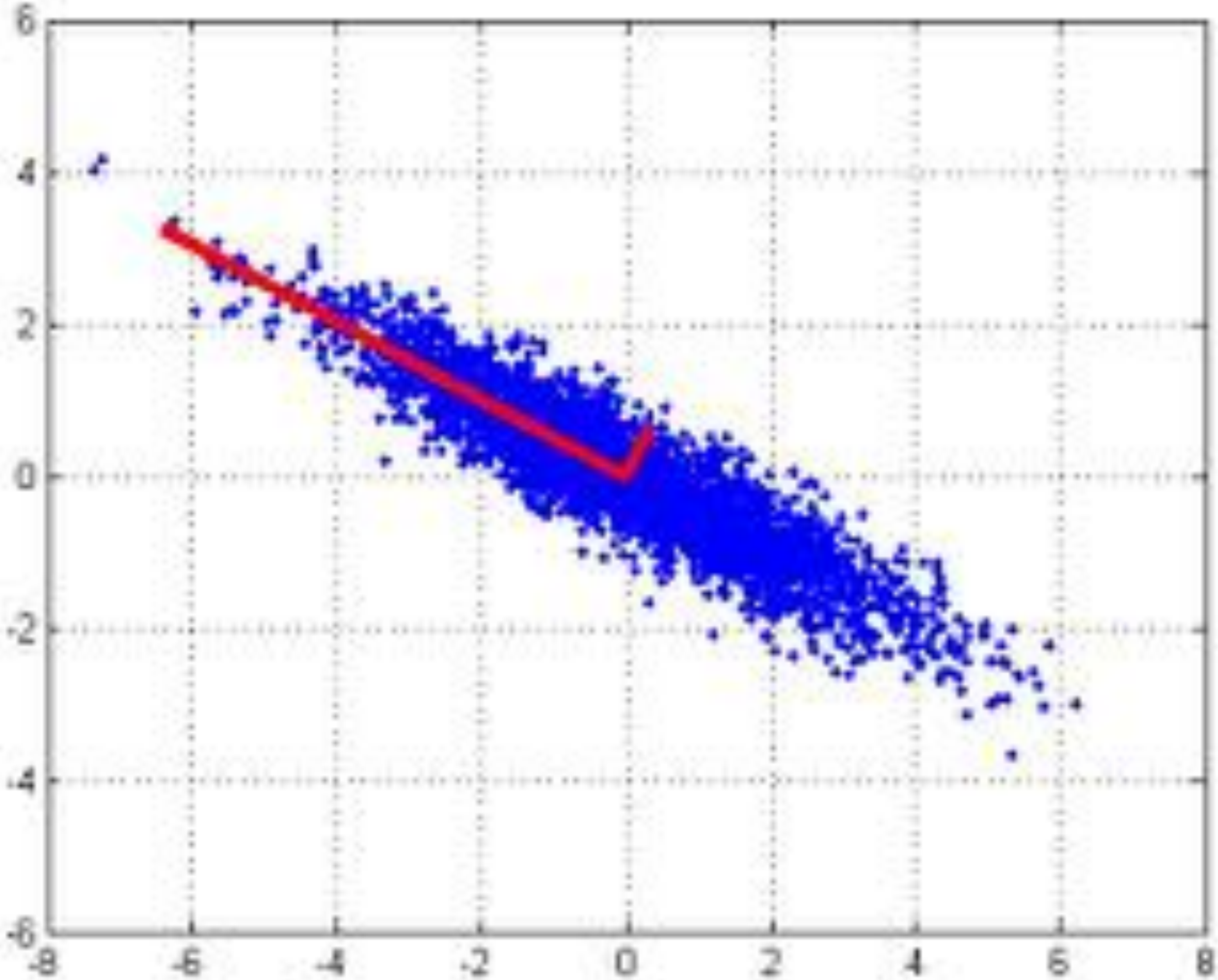
Slide from Barnabas Poczos

1st PCA axis



Slide from Barnabas Poczos

2^{nd} PCA axis



Slide from Barnabas Poczos

Principal Component Analysis (PCA)

Whiteboard

- Data for PCA
- PCA Definition
- Objective functions for PCA

Data for PCA

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

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

We assume the data is **centered**

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

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

A: Subtract off the sample mean

Sample Covariance Matrix

The sample covariance matrix is given by:

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

Since the data matrix is centered, we rewrite as:

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

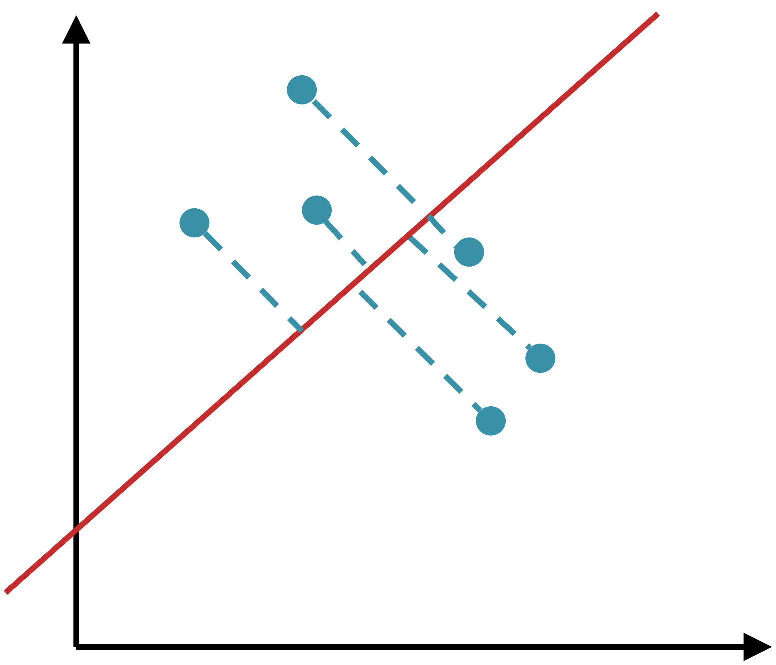
$$\mathbf{X} = \begin{bmatrix} \left(\mathbf{x}^{(1)}\right)^T \\ \left(\mathbf{x}^{(2)}\right)^T \\ \vdots \\ \left(\mathbf{x}^{(N)}\right)^T \end{bmatrix}$$

Maximizing the Variance

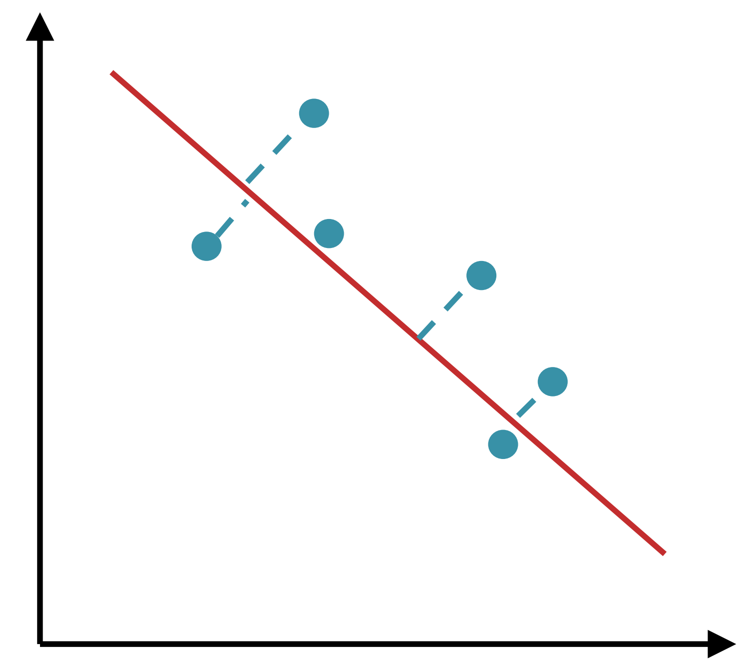
Quiz: Consider the two projections below

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

Option A



Option B



PCA

Equivalence of Maximizing Variance and Minimizing Reconstruction Error

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

Proof: First, note that:

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

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.

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

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

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

(5)

Principal Component Analysis (PCA)

Whiteboard

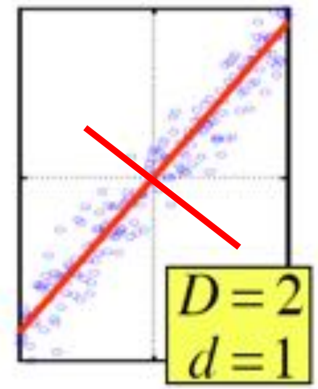
- PCA, Eigenvectors, and Eigenvalues
- Algorithms for finding Eigenvectors / Eigenvalues

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

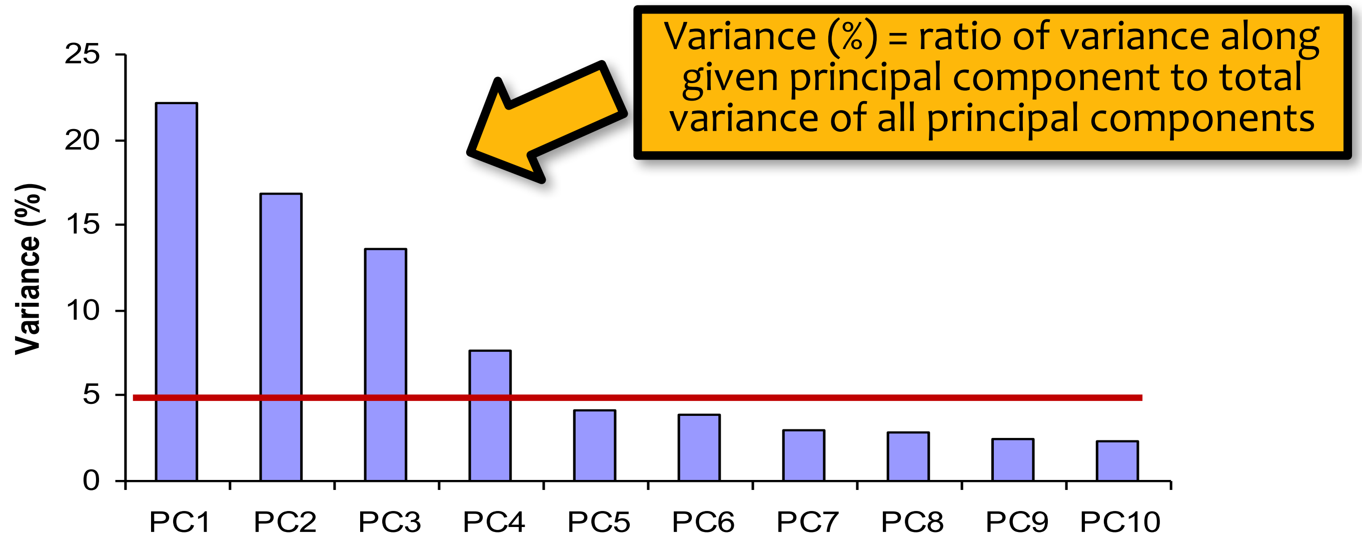
- The 1st PC v_1 is the the eigenvector of the sample covariance matrix $X\,X^T$ associated with the largest eigenvalue
- The 2nd PC v_2 is the the eigenvector of the sample covariance matrix $X\,X^T$ associated with the second largest eigenvalue
- And so on …

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.

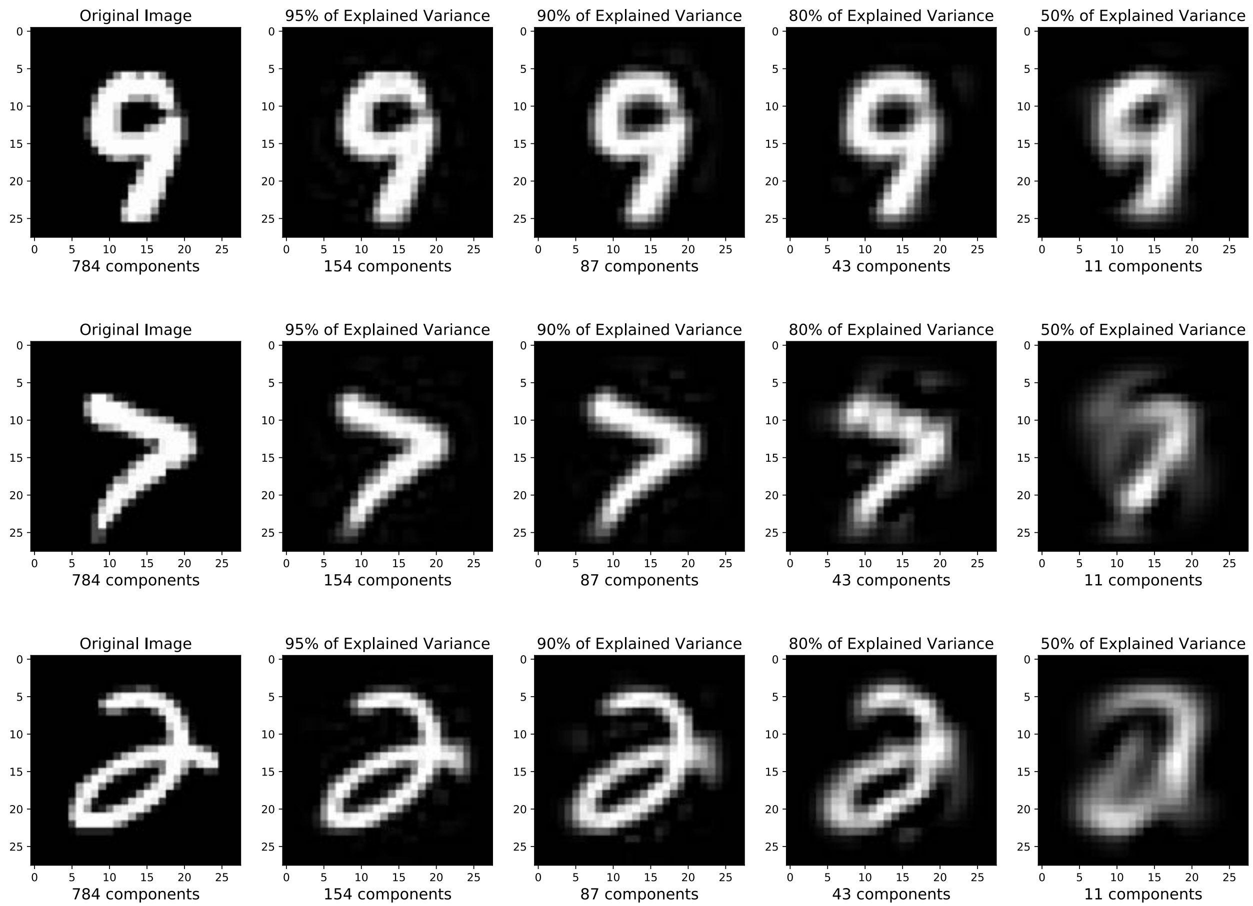


- You do lose some information, but if the eigenvalues are small, you don't lose much
  - M dimensions in original data
  - calculate M eigenvectors and eigenvalues
  - choose only the first D eigenvectors, based on their eigenvalues
  - final data set has only D dimensions

PCA EXAMPLES

Projecting MNIST digits

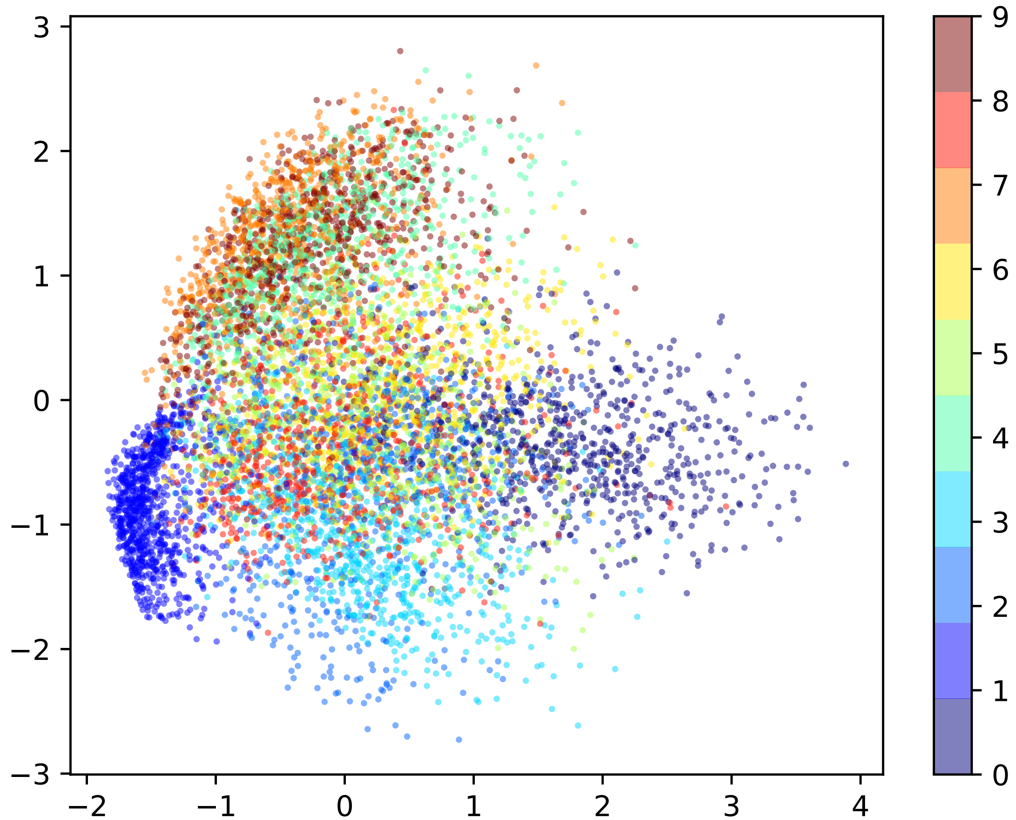
Task Setting:
1. Take 25x25 images of digits and project them down to K components
2. Report percent of variance explained for K components
3. Then project back up to 25x25 image to visualize how much information was preserved



Projecting MNIST digits

Task Setting:

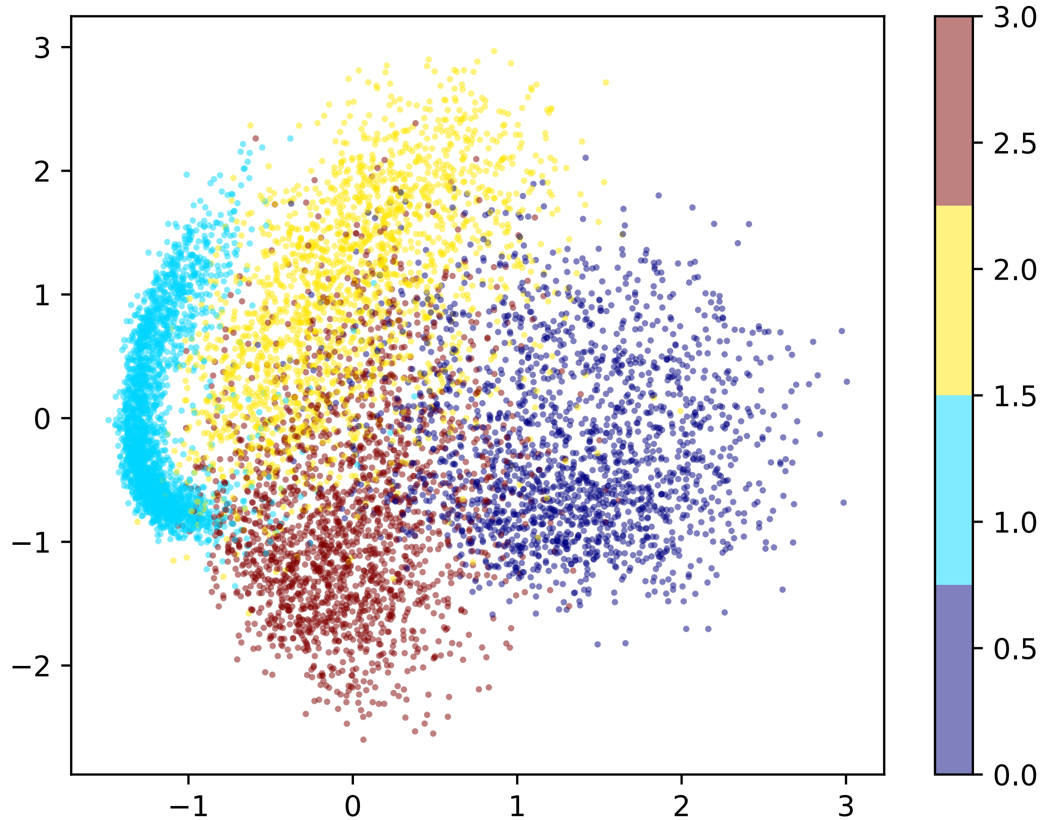
1. Take 25x25 images of digits and project them down to 2 components
2. Plot the 2 dimensional points



Projecting MNIST digits

Task Setting:

1. Take 25x25 images of digits and project them down to 2 components
2. Plot the 2 dimensional points



Slides from Barnabas Poczos

Original sources include:

- Karl Booksh Research group
- Tom Mitchell
- Ron Parr

PCA EXAMPLES

Face recognition

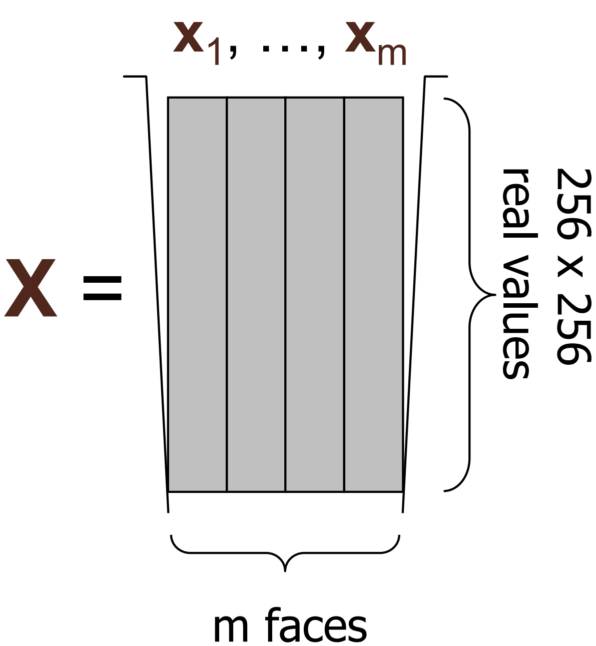
Challenge: Facial Recognition

- Want to identify specific person, based on facial image
- Robust to glasses, lighting,…

$\Rightarrow$ Can't just use the given 256 x 256 pixels

Applying PCA: Eigenfaces

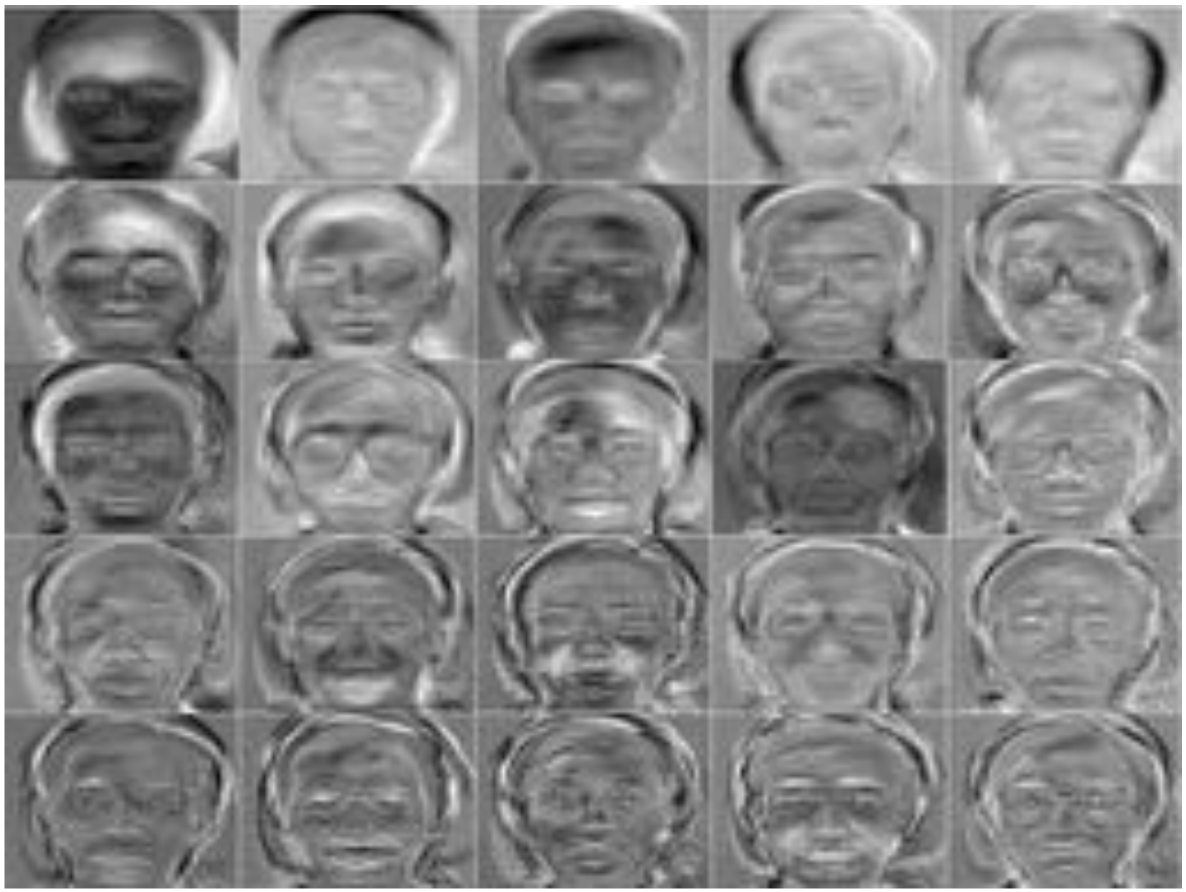
Method: Build one PCA database for the whole dataset and then classify based on the weights.



- Example data set: Images of faces
  - Famous Eigenface approach
    [Turk & Pentland], [Sirovich & Kirby]
- Each face **x** is …
  - 256×256 values (luminance at location)
  - **x** in $\Re^{256 \times 256}$ (view as 64K dim vector)

Slide from Barnabas Poczos

Principle Components



Slide from Barnabas Poczos

Reconstructing…

- … faster if train with…
  - only people w/out glasses
  - same lighting conditions

Slide from Barnabas Poczos

Shortcomings

- Requires carefully controlled data:
  - All faces centered in frame
  - Same size
  - Some sensitivity to angle
- Alternative:
  - "Learn" one set of PCA vectors for each angle
  - Use the one with lowest error

- Method is completely knowledge free
  - (sometimes this is good!)
  - Doesn't know that faces are wrapped around 3D objects (heads)
  - Makes no effort to preserve class distinctions

Slide from Barnabas Poczos

Image Compression

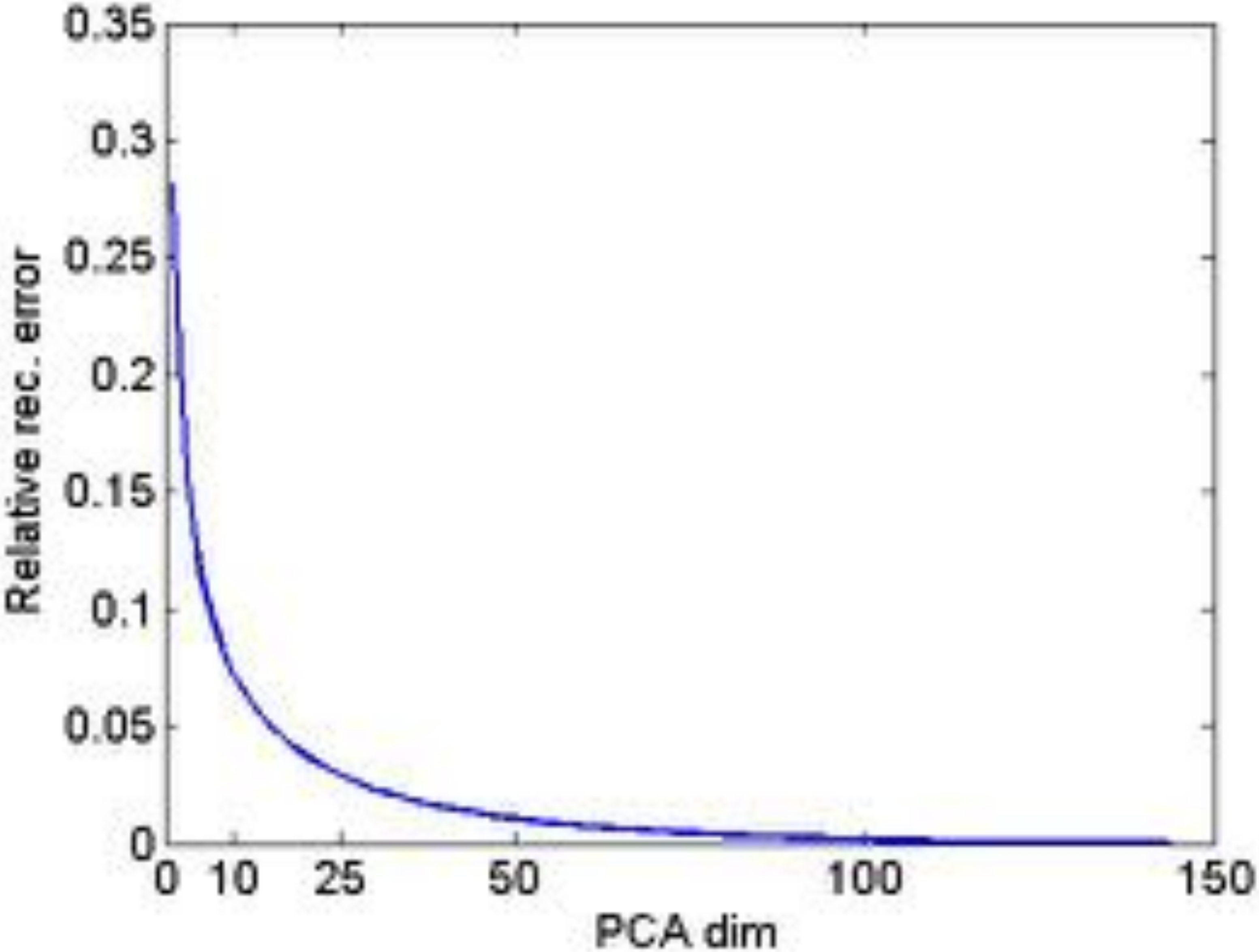
Slide from Barnabas Poczos

Original Image

- Divide the original 372x492 image into patches:
  - Each patch is an instance that contains 12x12 pixels on a grid
- View each as a 144-D vector

Slide from Barnabas Poczos

L_2 error and PCA dim



Slide from Barnabas Poczos

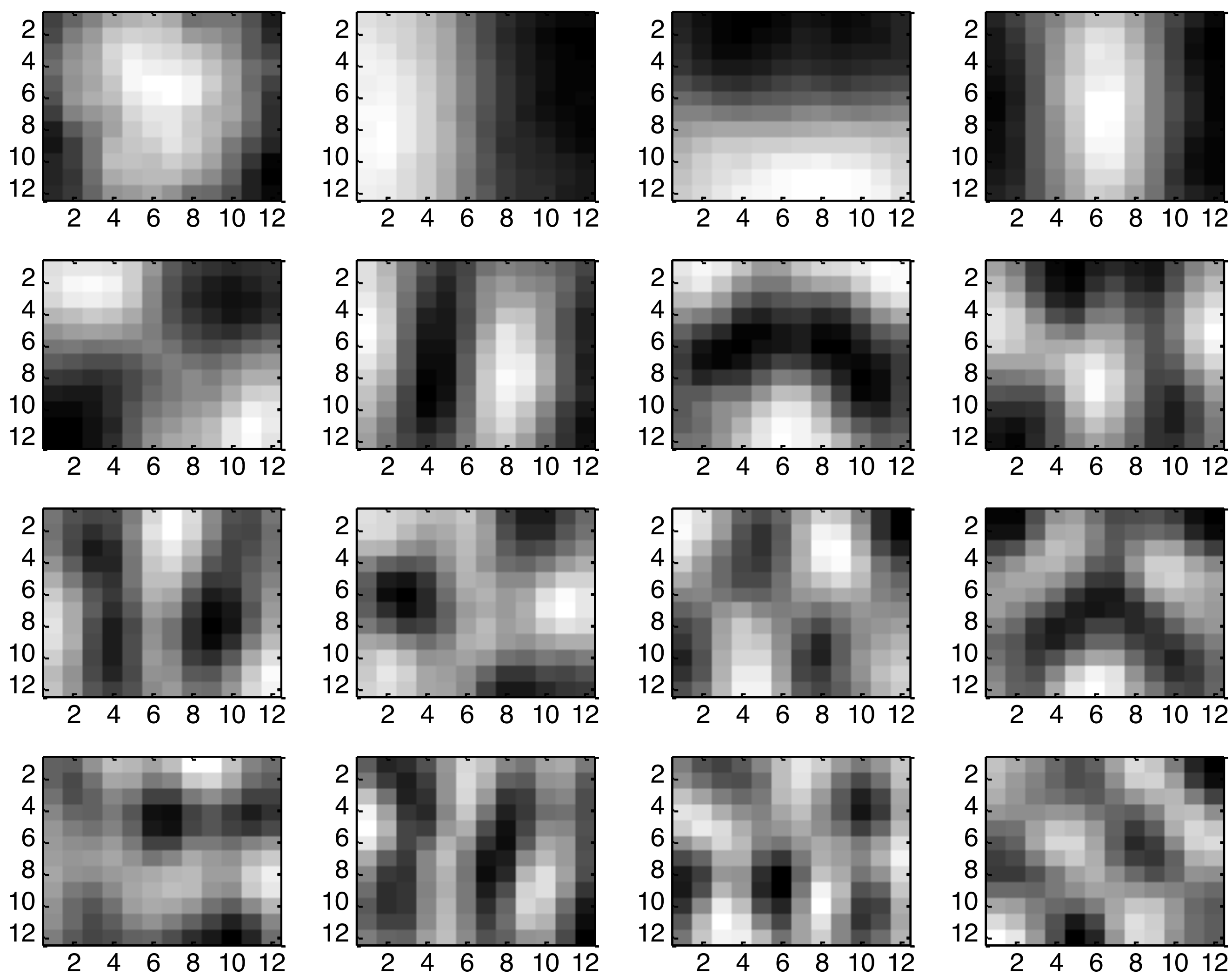
PCA compression: 144D → 60D

Slide from Barnabas Poczos

PCA compression: 144D → 16D

Slide from Barnabas Poczos

16 most important eigenvectors

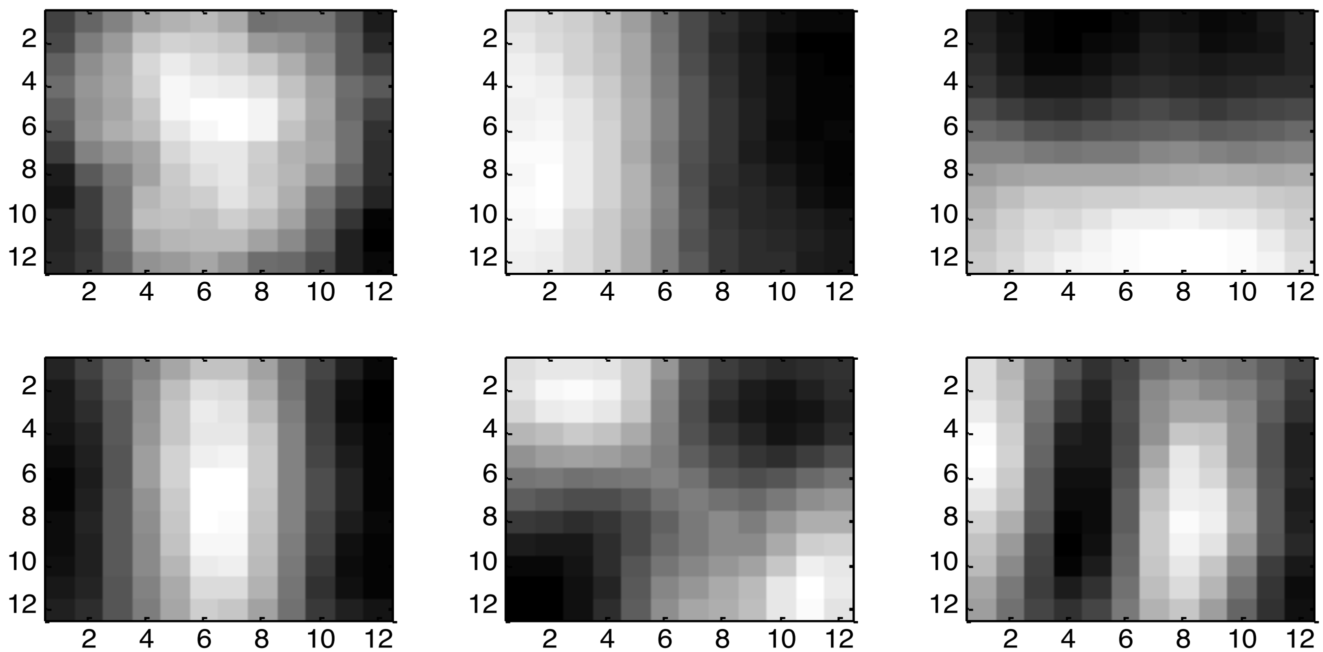


Slide from Barnabas Poczos

PCA compression: 144D → 6D

Slide from Barnabas Poczos

6 most important eigenvectors

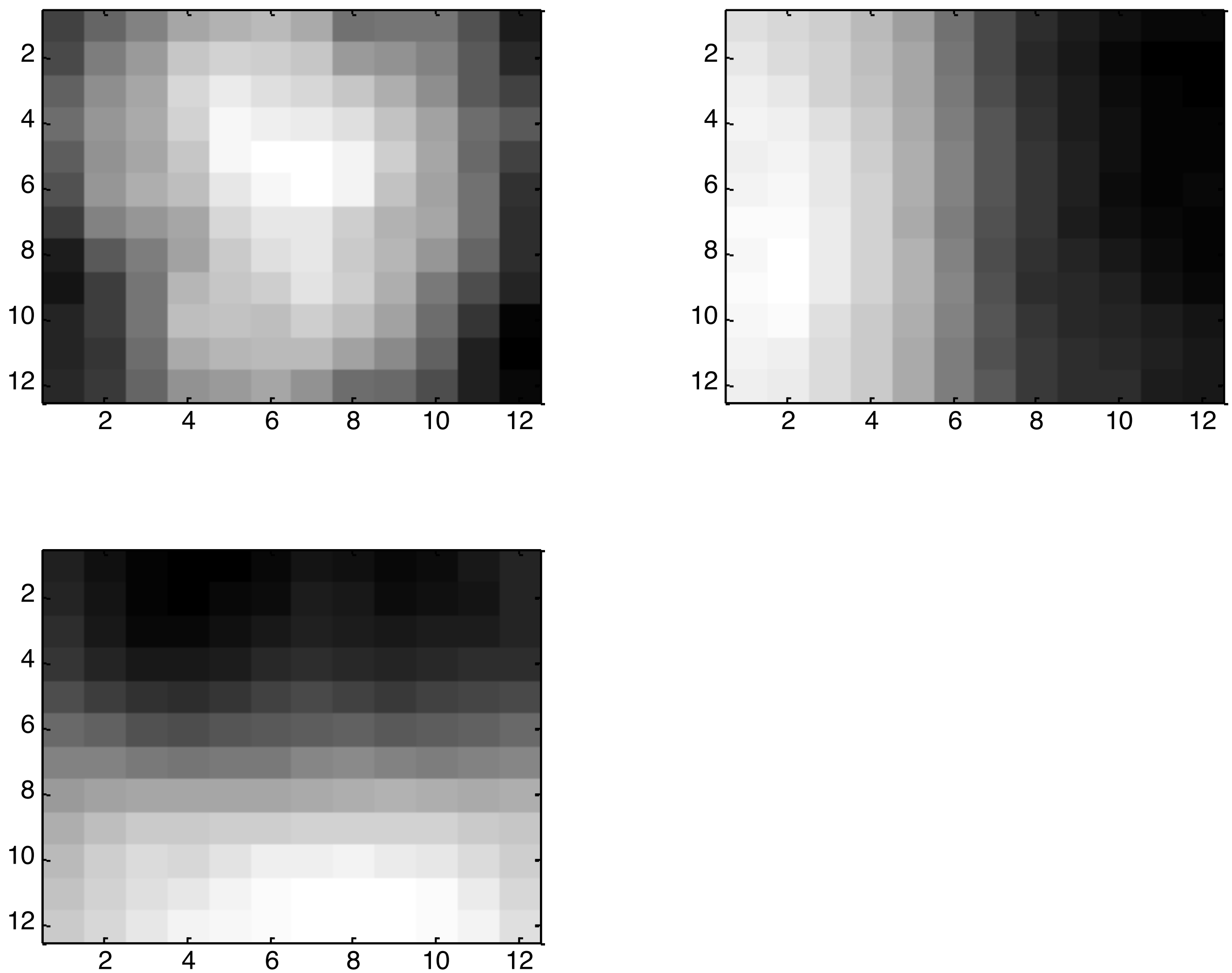


Slide from Barnabas Poczos

PCA compression: 144D → 3D

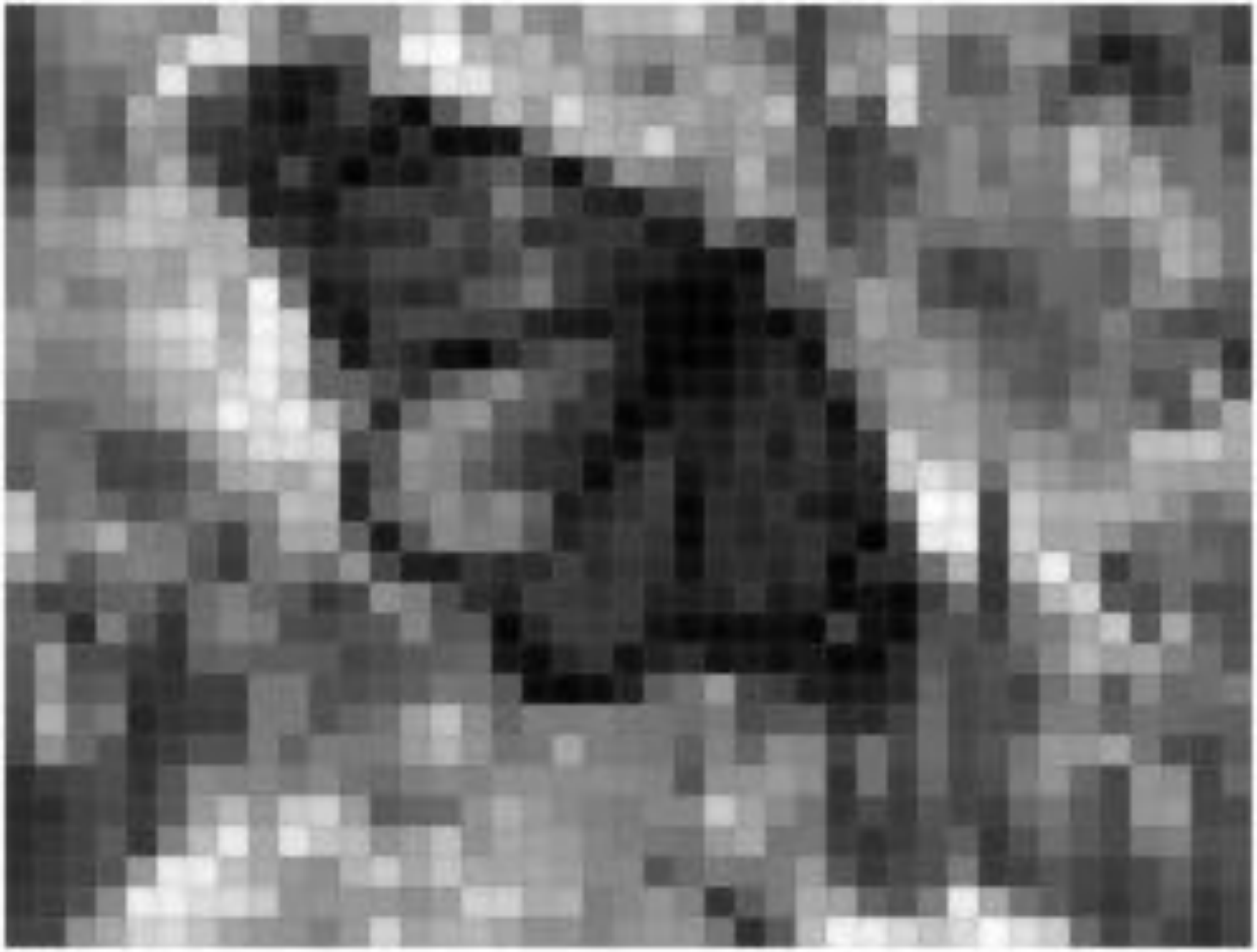
Slide from Barnabas Poczos

3 most important eigenvectors



Slide from Barnabas Poczos

PCA compression: 144D → 1D



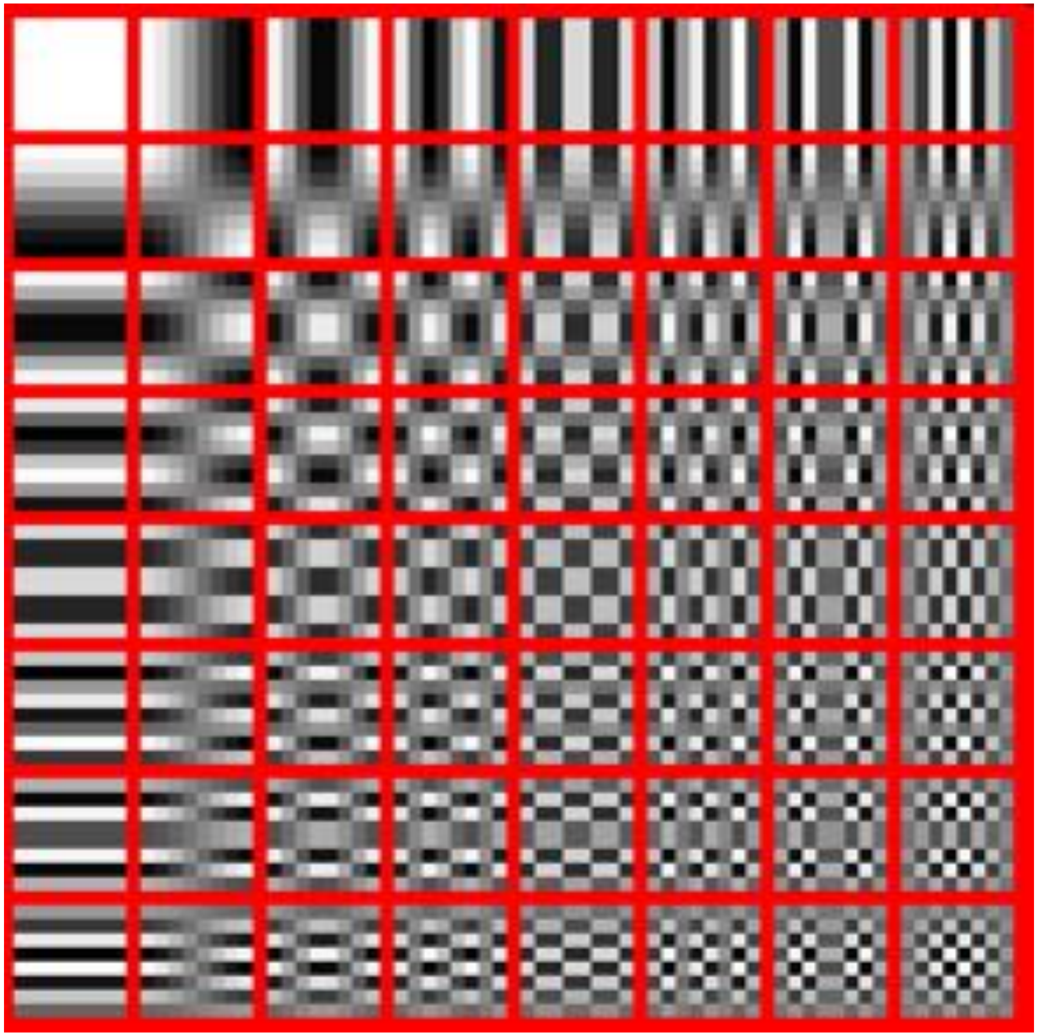
Slide from Barnabas Poczos

60 most important eigenvectors

Looks like the discrete cosine bases of JPG!...

Slide from Barnabas Poczos

2D Discrete Cosine Basis



http://en.wikipedia.org/wiki/Discrete_cosine_transform

Slide from Barnabas Poczos

Learning Objectives

Dimensionality Reduction / PCA

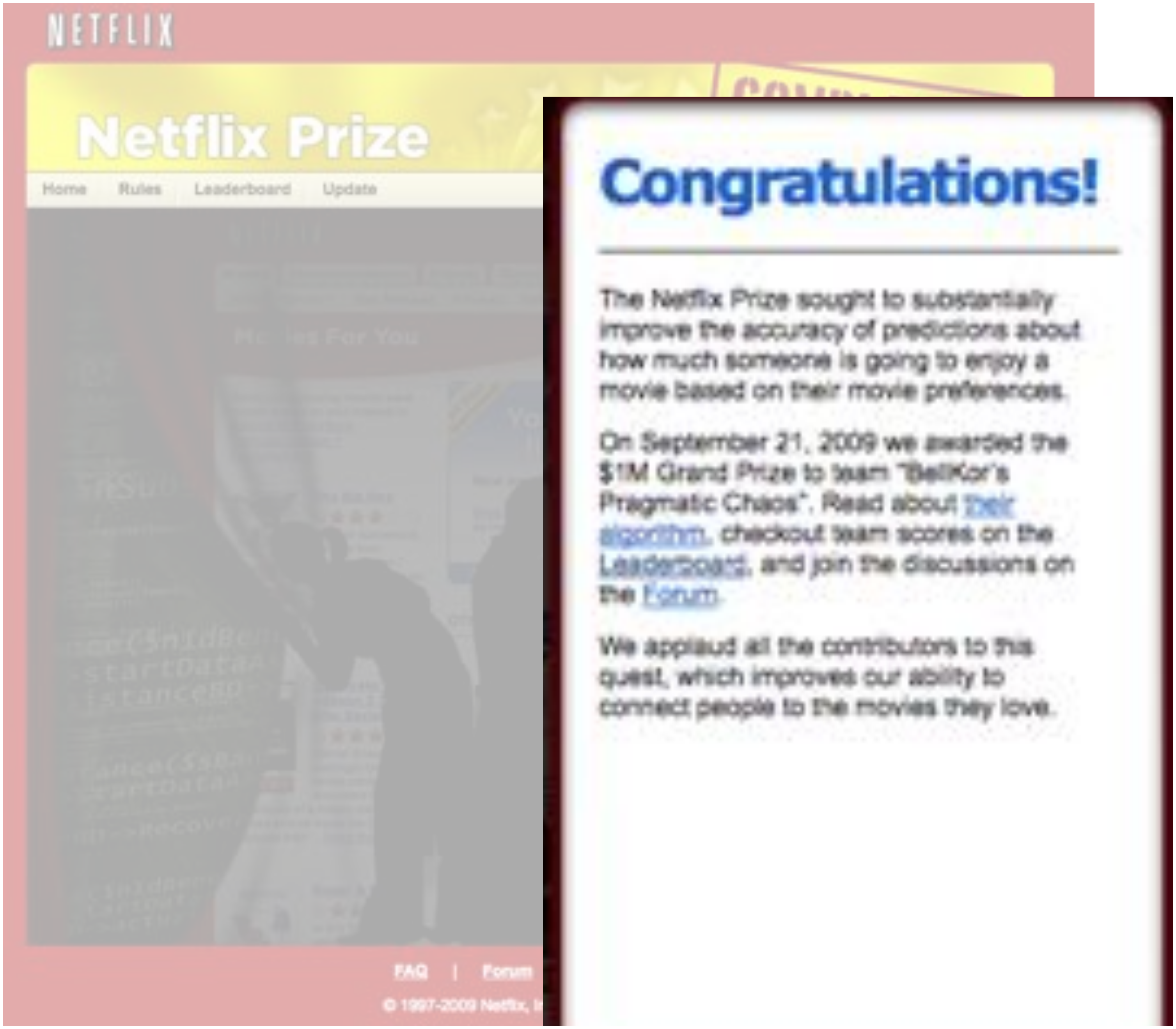
You should be able to…

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

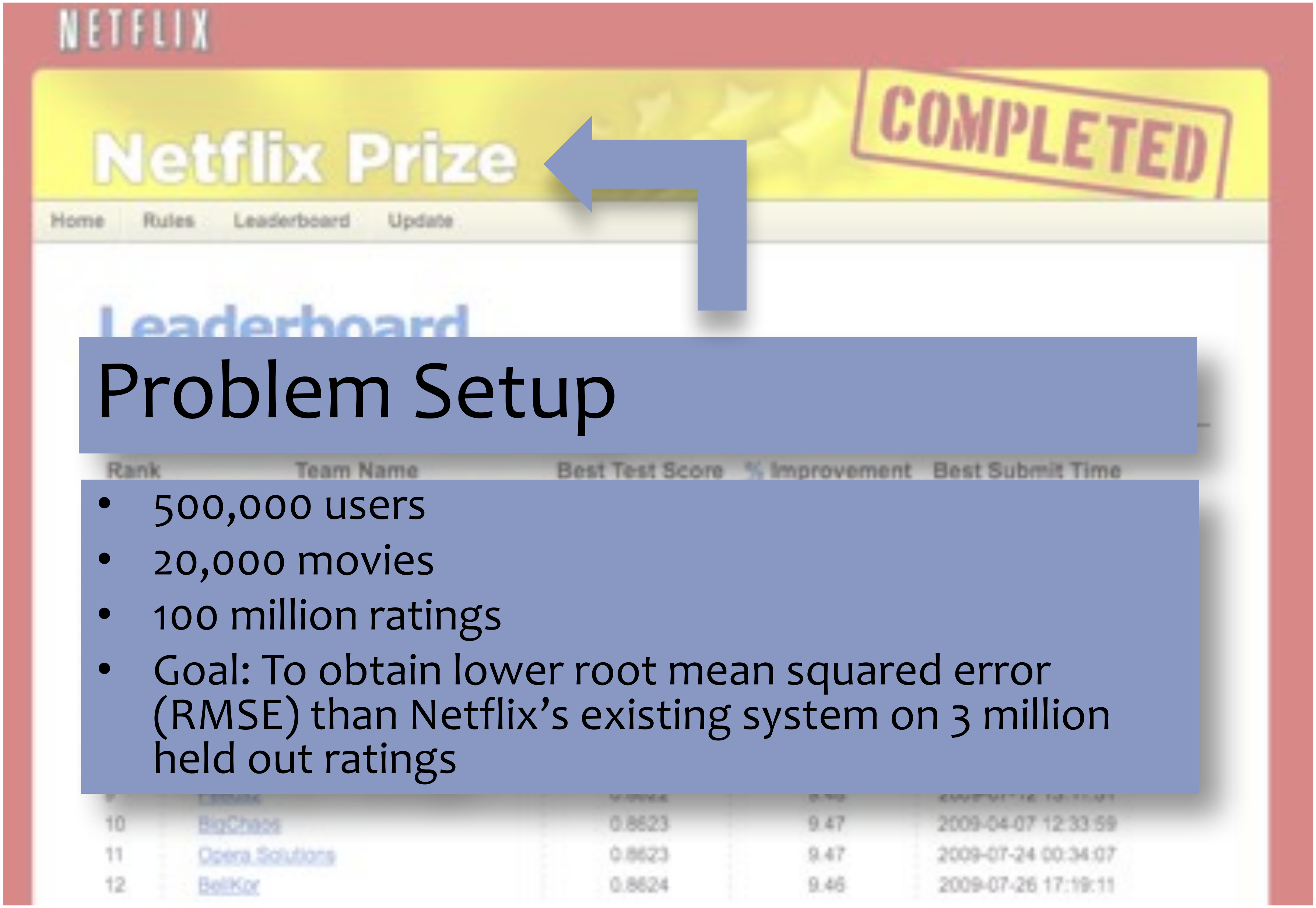
ENSEMBLE METHODS

Recommender Systems

Recommender Systems



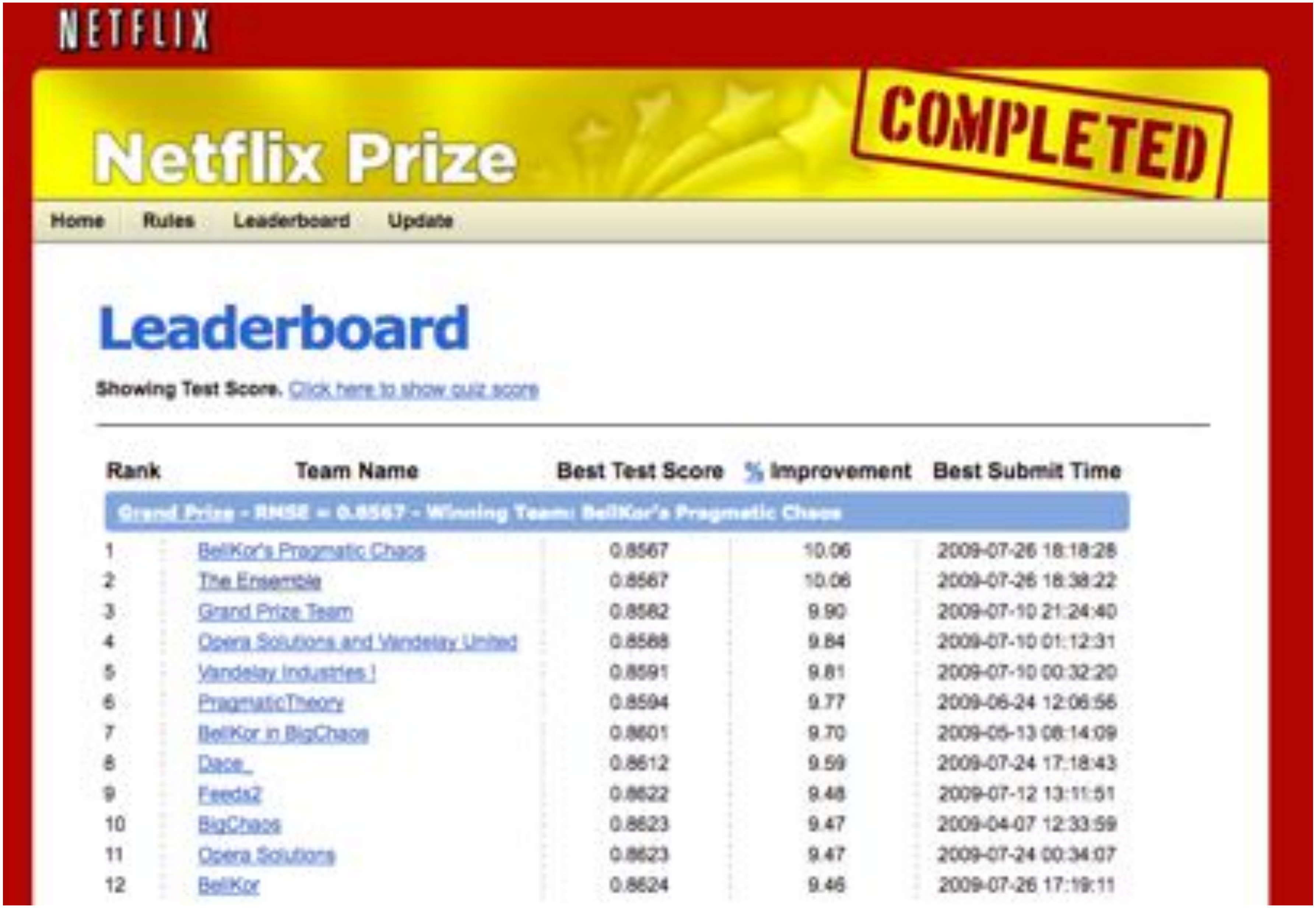
Recommender Systems



Problem Setup

- 500,000 users
- 20,000 movies
- 100 million ratings
- Goal: To obtain lower root mean squared error (RMSE) than Netflix's existing system on 3 million held out ratings

Recommender Systems



NETFLIX

Netflix Prize

COMPLETED

Home Rules Leaderboard Update

Leaderboard

Showing Test Score. Click here to show quiz score

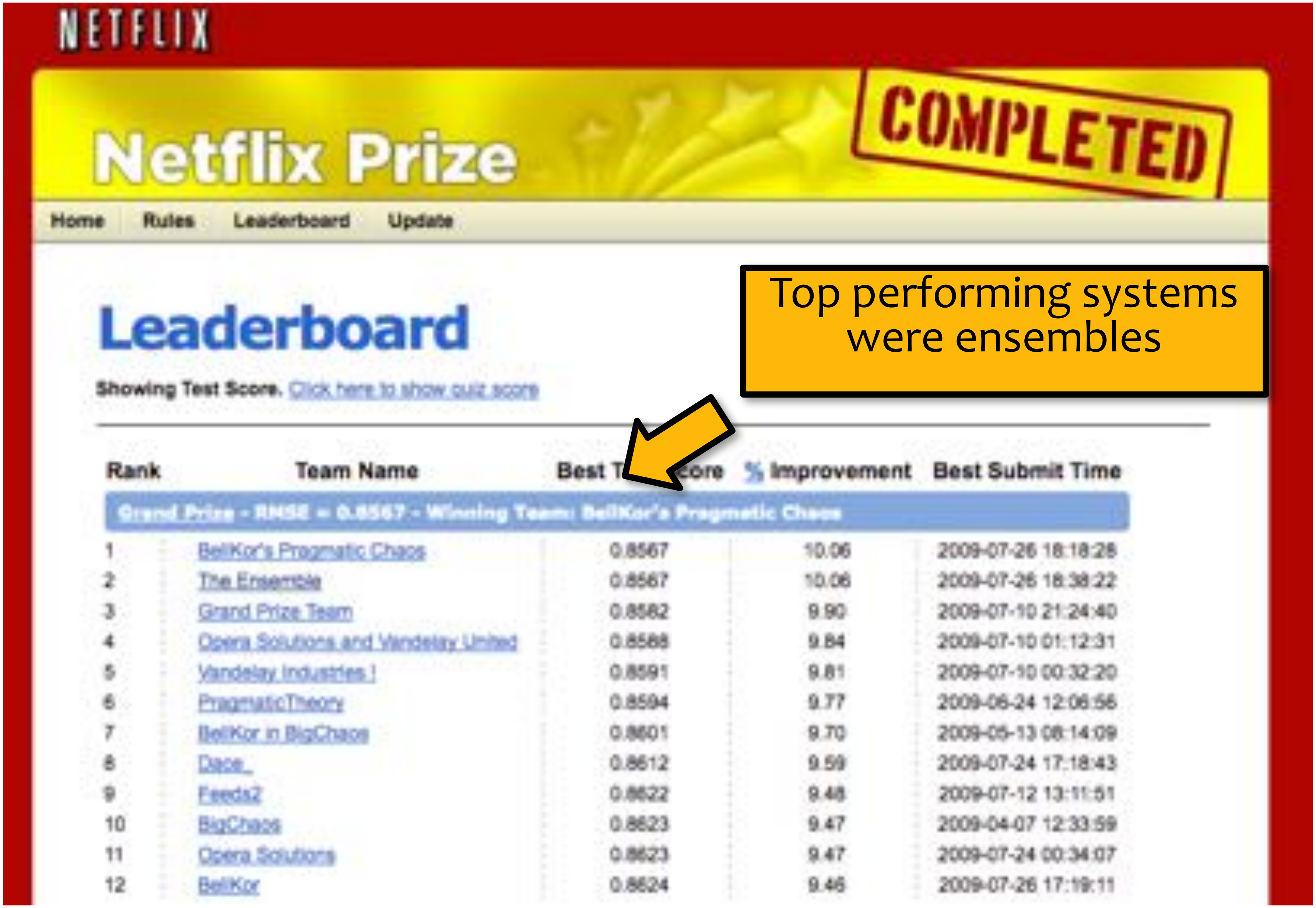
Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace_	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11

Recommender Systems

- **Setup:**
  - Items:
    movies, songs, products, etc.
    (often many thousands)
  - Users:
    watchers, listeners, purchasers, etc.
    (often many millions)
  - Feedback:
    5-star ratings, not-clicking ‘next’, purchases, etc.
- **Key Assumptions:**
  - Can represent ratings numerically as a user/item matrix
  - Users only rate a small number of items (the matrix is sparse)

	Doctor Strange	Star Trek: Beyond	Zootopia
Alice	1		5
Bob	3	4	
Charlie	3	5	2

Recommender Systems

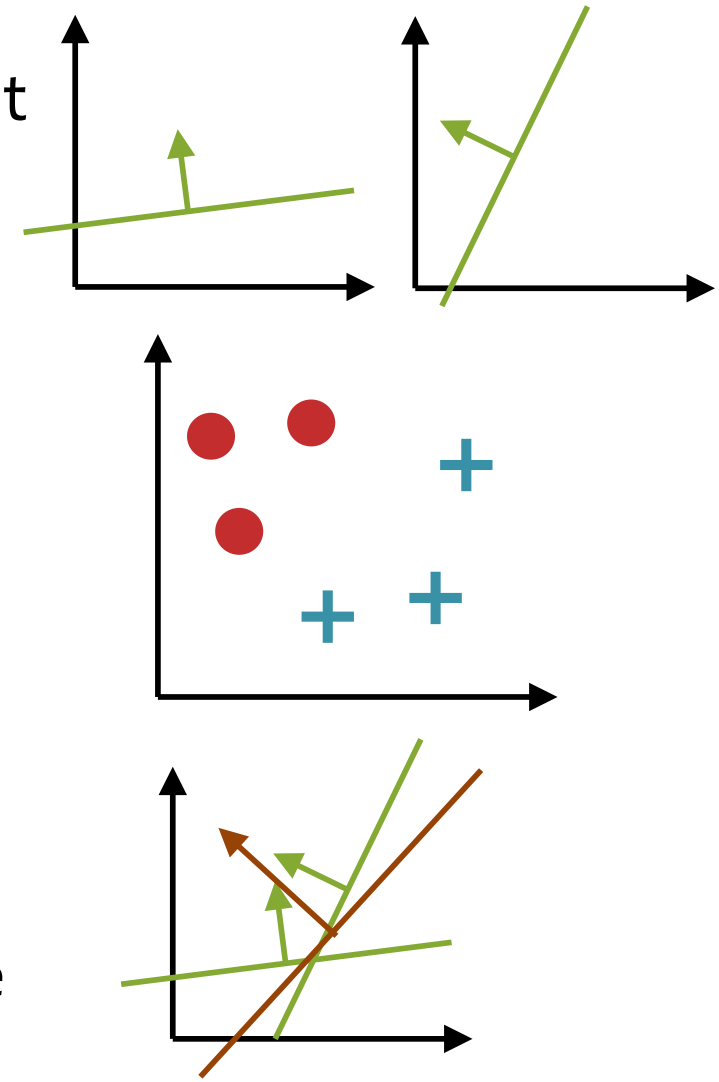


Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace_	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11

Weighted Majority Algorithm

(Littlestone & Warmuth, 1994)

- **Given:** pool *A* of binary classifiers (that you know nothing about)
- **Data:** stream of examples (i.e. online learning setting)
- **Goal:** design a new learner that uses the predictions of the pool to make new predictions
- **Algorithm:**
  - Initially weight all classifiers equally
  - Receive a training example and predict the (weighted) majority vote of the classifiers in the pool
  - Down-weight classifiers that contribute to a mistake by a factor of β



Weighted Majority Algorithm

(Littlestone & Warmuth, 1994)

Suppose we have a pool of T binary classifiers $\mathcal{A} = \{h_1, \ldots, h_T\}$ where $h_t : \mathbb{R}^M \rightarrow \{+1, -1\}$. Let α_t be the weight for classifier h_t.

Algorithm 1 Weighted Majority Algorithm

procedure WEIGHTEDMAJORITY($\mathcal{A}$, β)
    Initialize classifier weights $\alpha_t = 1, \ \forall t \in \{1, \ldots, T\}$
    for each training example $(\mathbf{x}, y)$ **do**
        Predict majority vote class (splitting ties randomly)

$$\hat{h}(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

        if a mistake is made $\hat{h}(x) \neq y$ **then**
            for each classifier $t \in \{1, \ldots, T\}$ **do**
                If $h_t(x) \neq y$, then $\alpha_t \leftarrow \beta\alpha_t$

Weighted Majority Algorithm

(Littlestone & Warmuth, 1994)

Theorem 0.1 (Littlestone & Warmuth, 1994). *If the Weighted Majority Algorithm is applied to a pool $\mathcal{A}$ of classifiers, and if each algorithm makes at most m mistakes on the sequence of examples, then the total number of mistakes is upper bounded by $2.4(\log|\mathcal{A}| + m)$.*

This is a "mistake bound" of the variety we saw for the Perceptron algorithm

ADABOOST

Comparison

Weighted Majority Algorithm

- an example of an ensemble method
- assumes the classifiers are learned ahead of time
- only learns (majority vote) weight for each classifiers

AdaBoost

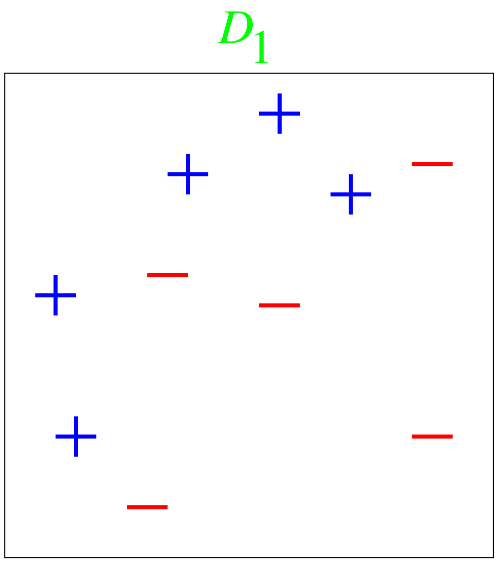
- an example of a boosting method
- simultaneously learns:
  - the classifiers themselves
  - (majority vote) weight for each classifiers

AdaBoost

Whiteboard:

– Weak Learners vs. Strong Learners
– Weak Learning = Strong Learning
– Key Idea behind AdaBoost
– AdaBoost algorithm
– Toy Example: Learning with Decision Stumps

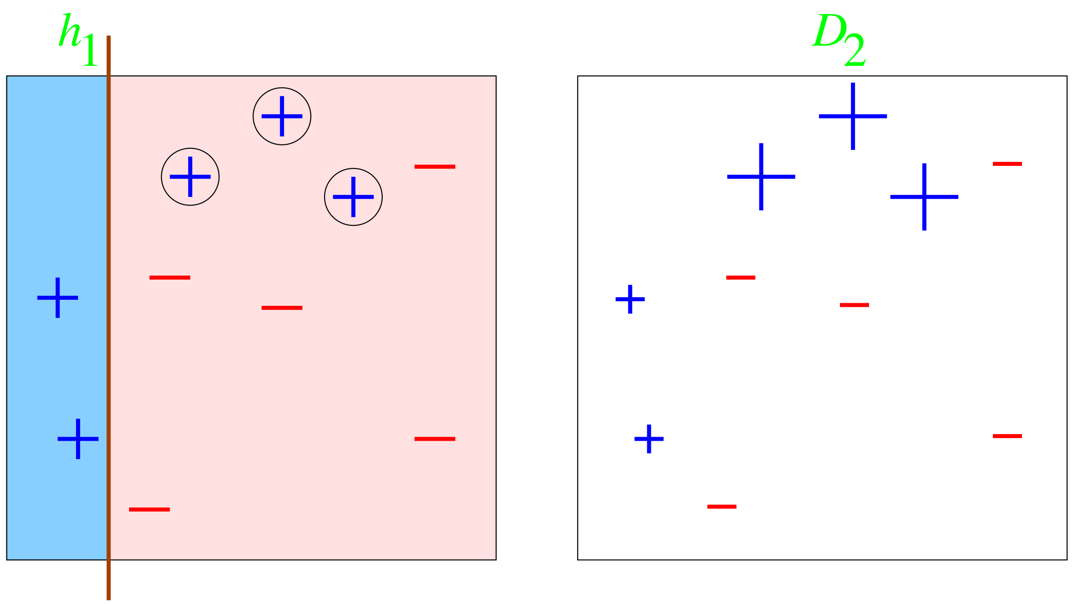
AdaBoost: Toy Example



weak classifiers = vertical or horizontal half-planes

Slide from Schapire NIPS Tutorial

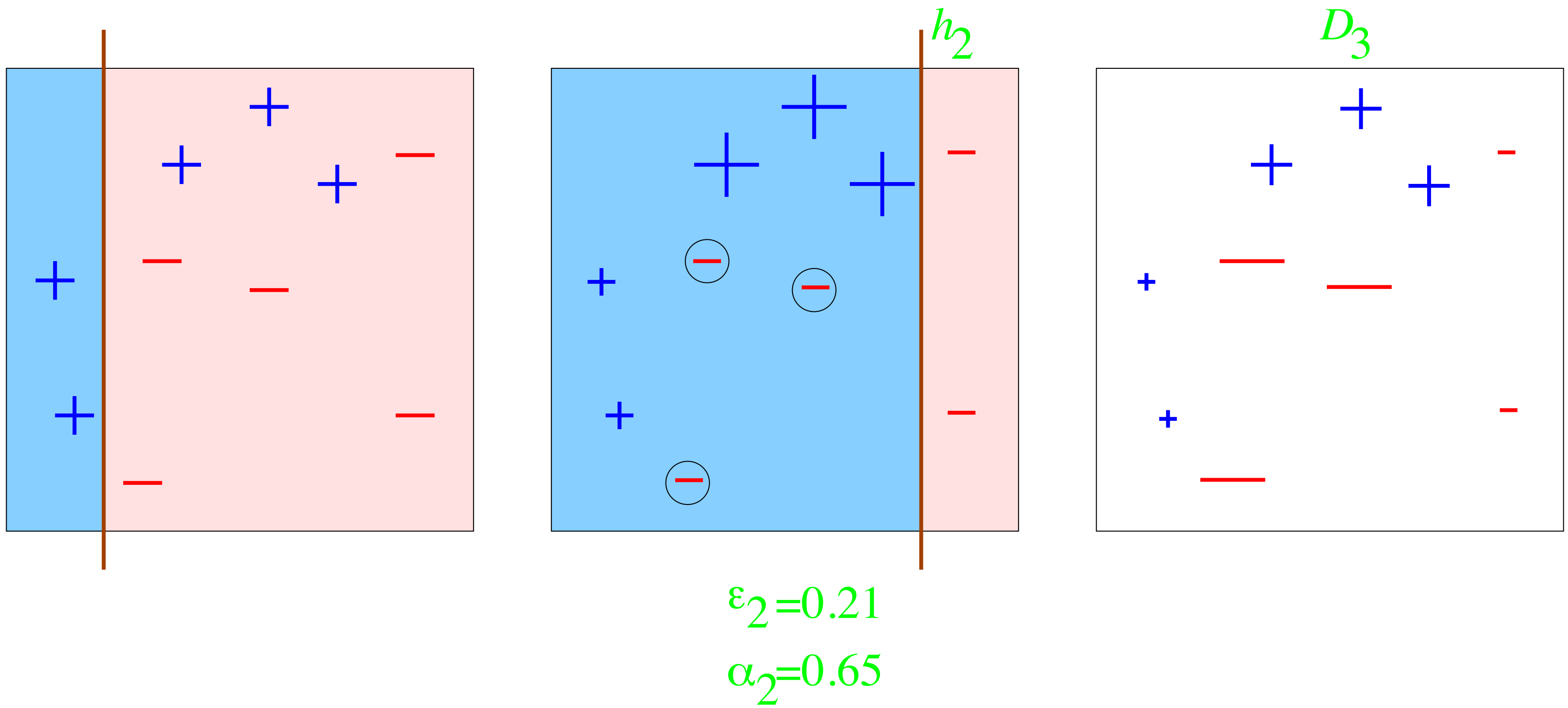
AdaBoost: Toy Example



$\varepsilon_1 = 0.30$

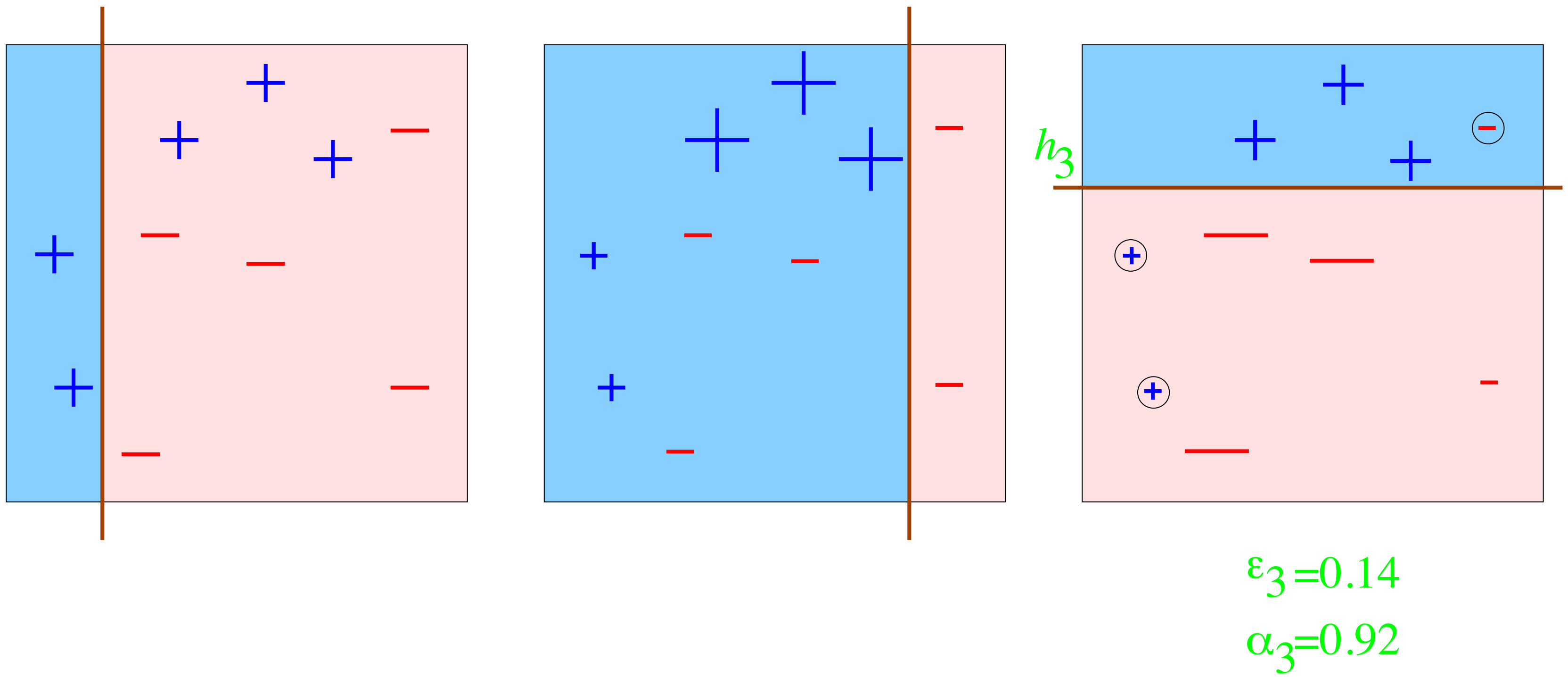
$\alpha_1 = 0.42$

AdaBoost: Toy Example

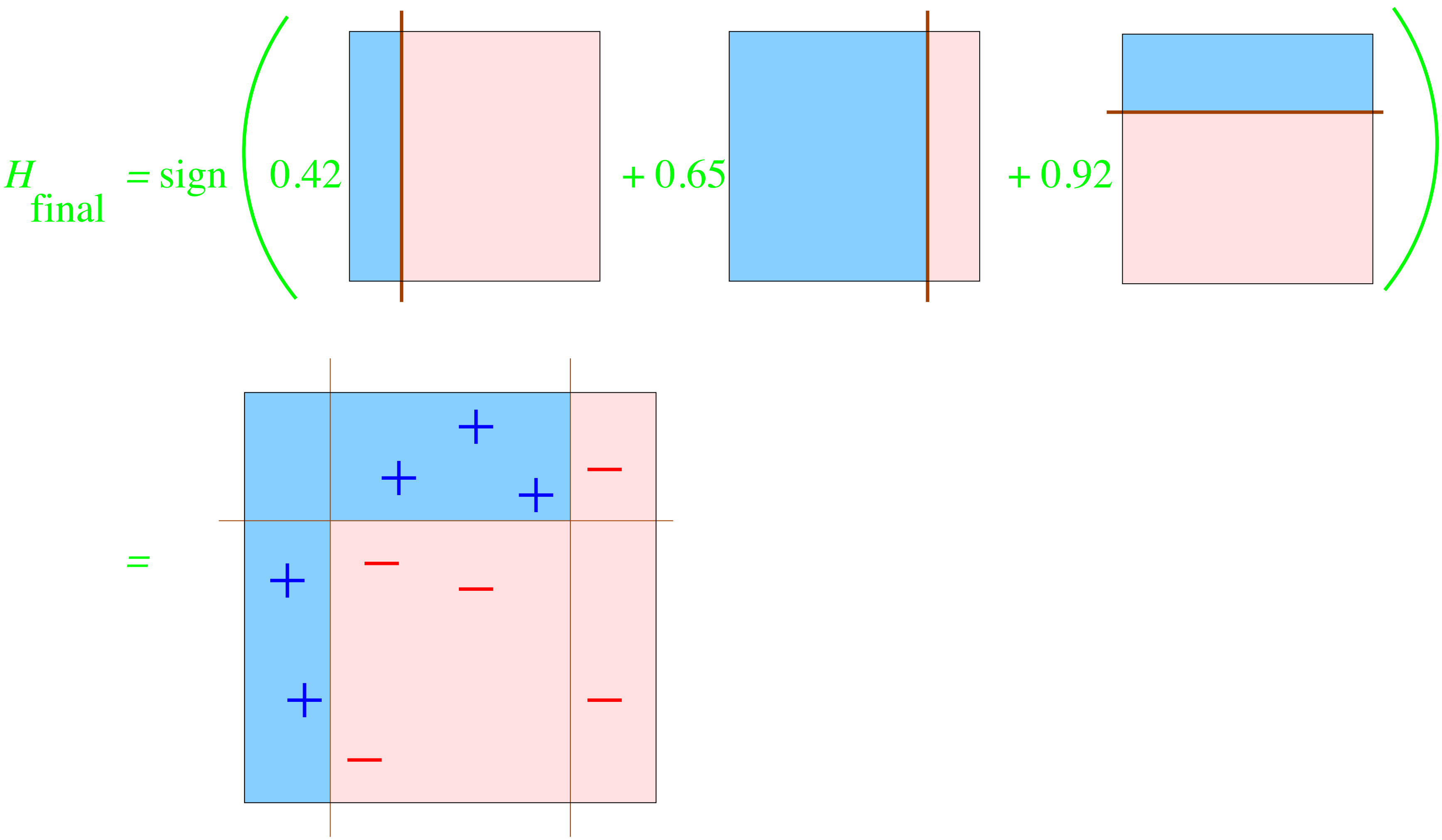


Slide from Schapire NIPS Tutorial

AdaBoost: Toy Example



AdaBoost: Toy Example



AdaBoost

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Algorithm from (Freund & Schapire, 1999)

AdaBoost

Whiteboard:

- Theoretical Results:
  - training error
  - generalization error

AdaBoost

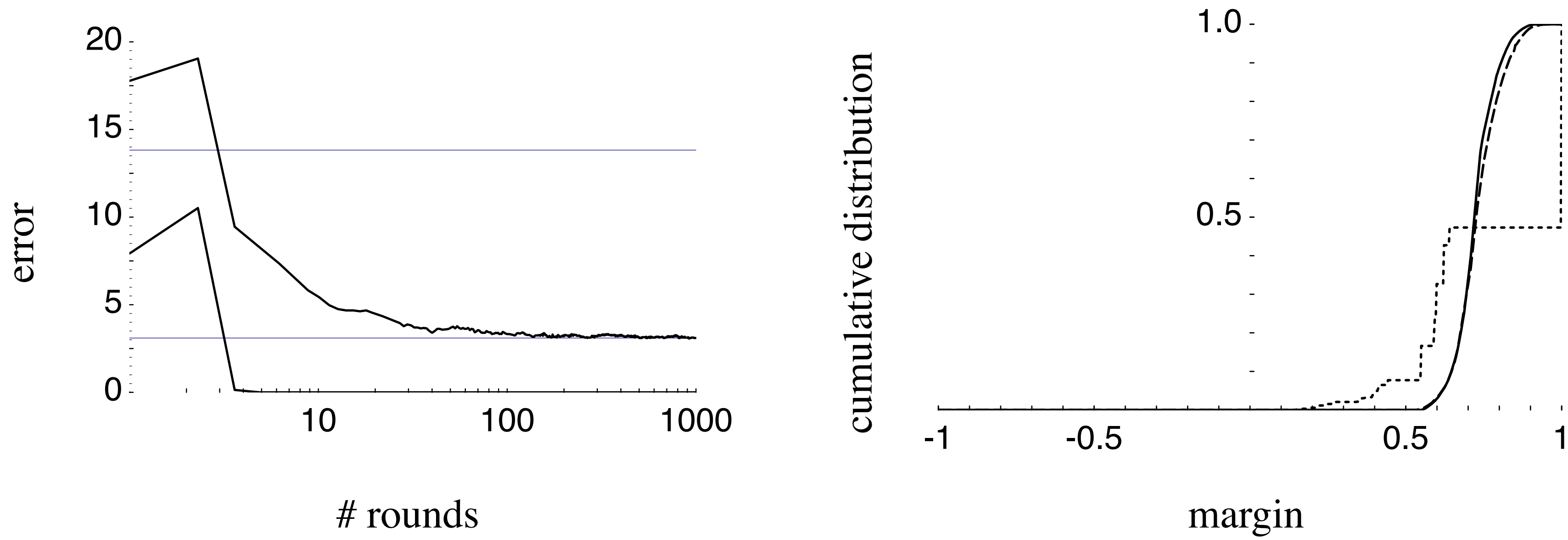


Figure 2: Error curves and the margin distribution graph for boosting C4.5 on the letter dataset as reported by Schapire et al. [41]. *Left*: the training and test error curves (lower and upper curves, respectively) of the combined classifier as a function of the number of rounds of boosting. The horizontal lines indicate the test error rate of the base classifier as well as the test error of the final combined classifier. *Right*: The cumulative distribution of margins of the training examples after 5, 100 and 1000 iterations, indicated by short-dashed, long-dashed (mostly hidden) and solid curves, respectively.

Figure from (Freund & Schapire, 1999)

Learning Objectives

Ensemble Methods / Boosting

You should be able to…

1. Implement the Weighted Majority Algorithm
2. Implement AdaBoost
3. Distinguish what is learned in the Weighted Majority Algorithm vs. Adaboost
4. Contrast the theoretical result for the Weighted Majority Algorithm to that of Perceptron
5. Explain a surprisingly common empirical result regarding Adaboost train/test curves