Dimensionality Reduction

CS 534: Machine Learning

Slides adapted from David Sontag, Fei Sha, Yan Liu, Trevor Hastie, and Rob Tibshirani

Unsupervised Learning: Motivation

- What if we don’t have a response variable?
  - Cases where it is easier to obtain unlabeled data than labeled data
- What if we have high-dimensional data?
- Is there an informative way to visualize this data?
- Can we discover subgroups amongst these variables?

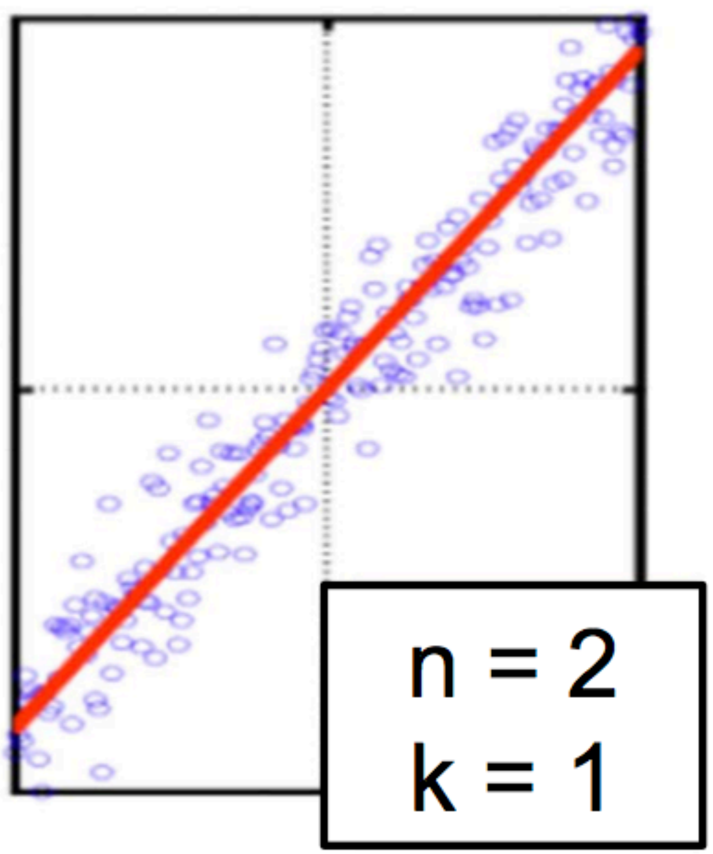
Unsupervised Learning: Challenge

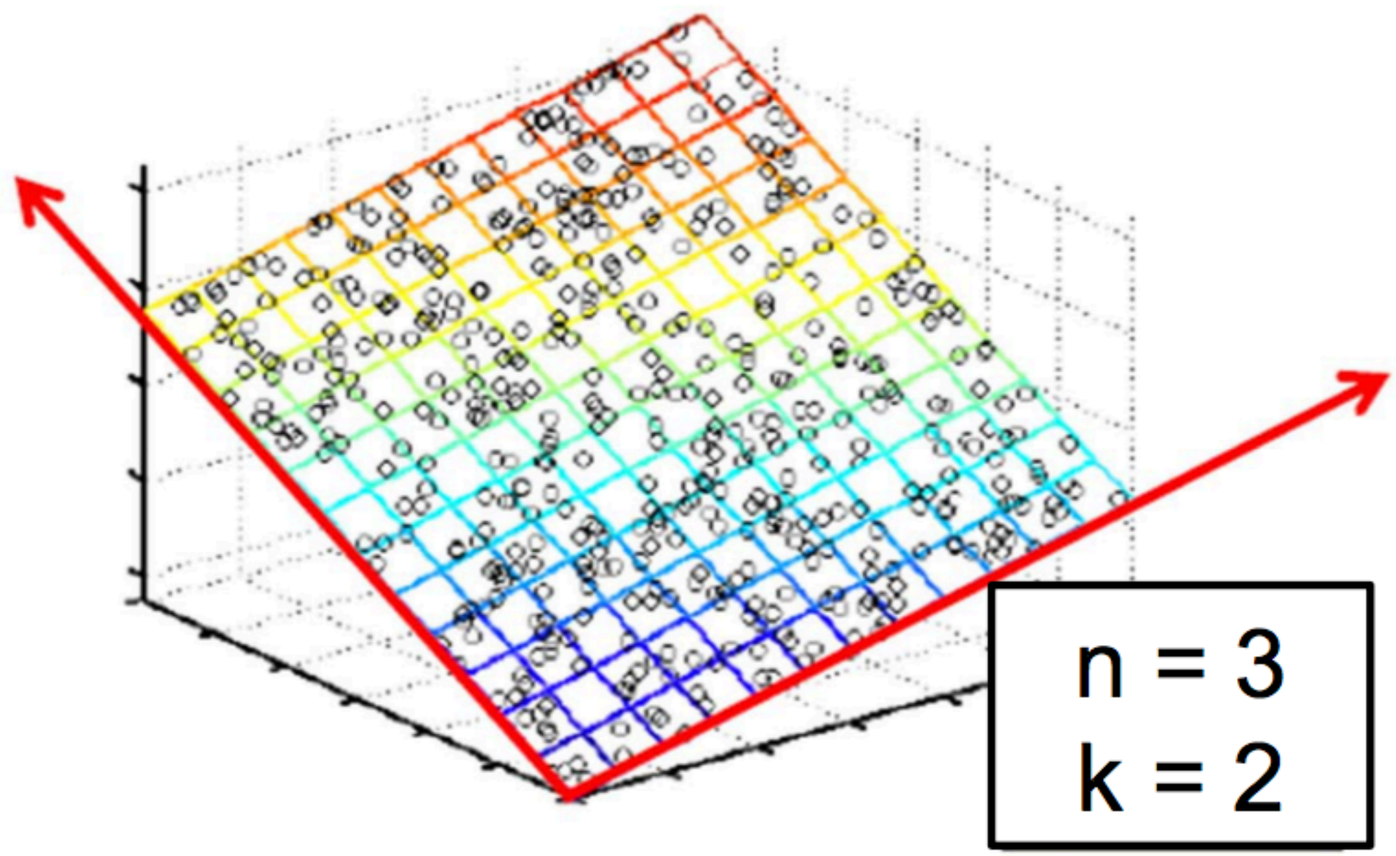
- How to evaluate the model?
  - No simple goal for analysis (e.g., prediction of response)
  - Even if there was something you want to assess, may not be easy to quantify (e.g., overall sentiment of a movie review)
  - What metric should be used?

Dimensionality Reduction

- Represent data with fewer dimensions
- Discover “intrinsic dimensionality” of data
- Why?
  - Noise reduction
  - Easier learning — less parameters
  - Easier visualization — show high dimensional data in 2D or 3D

Example: Dimensionality Reduction





Slide by Yi Zhang

Lower Dimensional Projections

- Transform dataset to have less features
- New feature space
  - Existing feature
  - Linear / non-linear combination of original features

$$z_k = \beta_0^{(k)} + \sum_i \beta_i^{(k)} \Phi(x_i)$$

- Typically done in an unsupervised setting

Review: Projection onto Unit Vectors

- Definition of dot product:

$$\mathbf{A} \cdot \mathbf{B} = ||\mathbf{A}||_2 ||\mathbf{B}||_2 \cos\theta$$

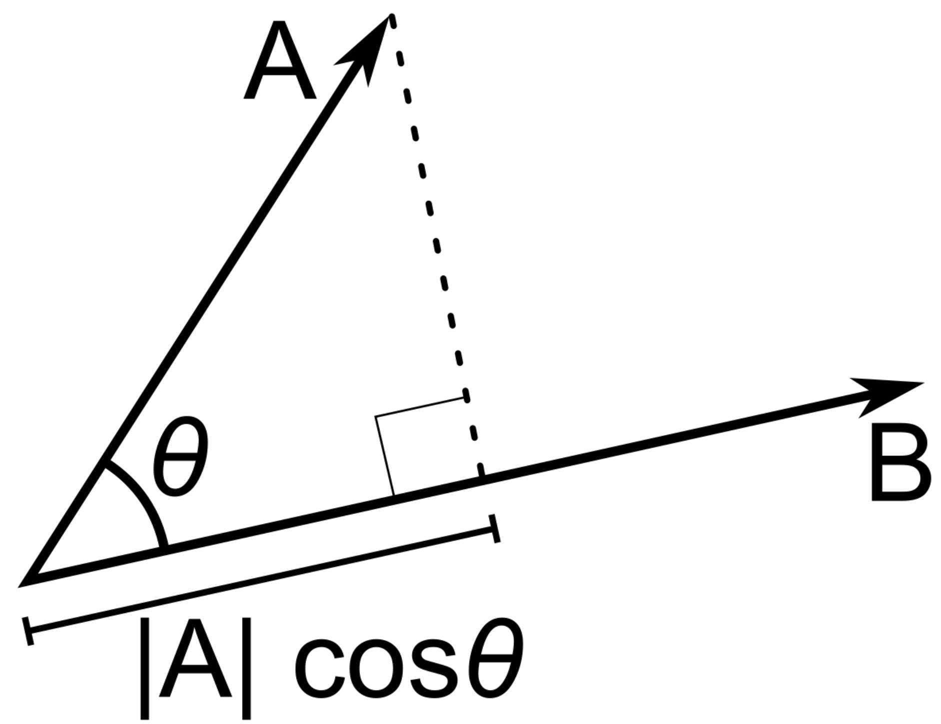
- If B is a unit vector, dot product is length of the projection

$$\mathbf{A} \cdot \mathbf{B} = ||\mathbf{A}||_2 \cos\theta$$

- Projection of A onto B:

$$(\mathbf{A} \cdot \mathbf{B})\mathbf{B}$$

Coefficient / score



https://en.wikipedia.org/wiki/Dot_product

Review: Projection onto Unit Vectors

- Consider a matrix, X, where we want to project each row onto vector v with unit norm

$$\mathbf{X} \in \mathbb{R}^{n \times p} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}$$

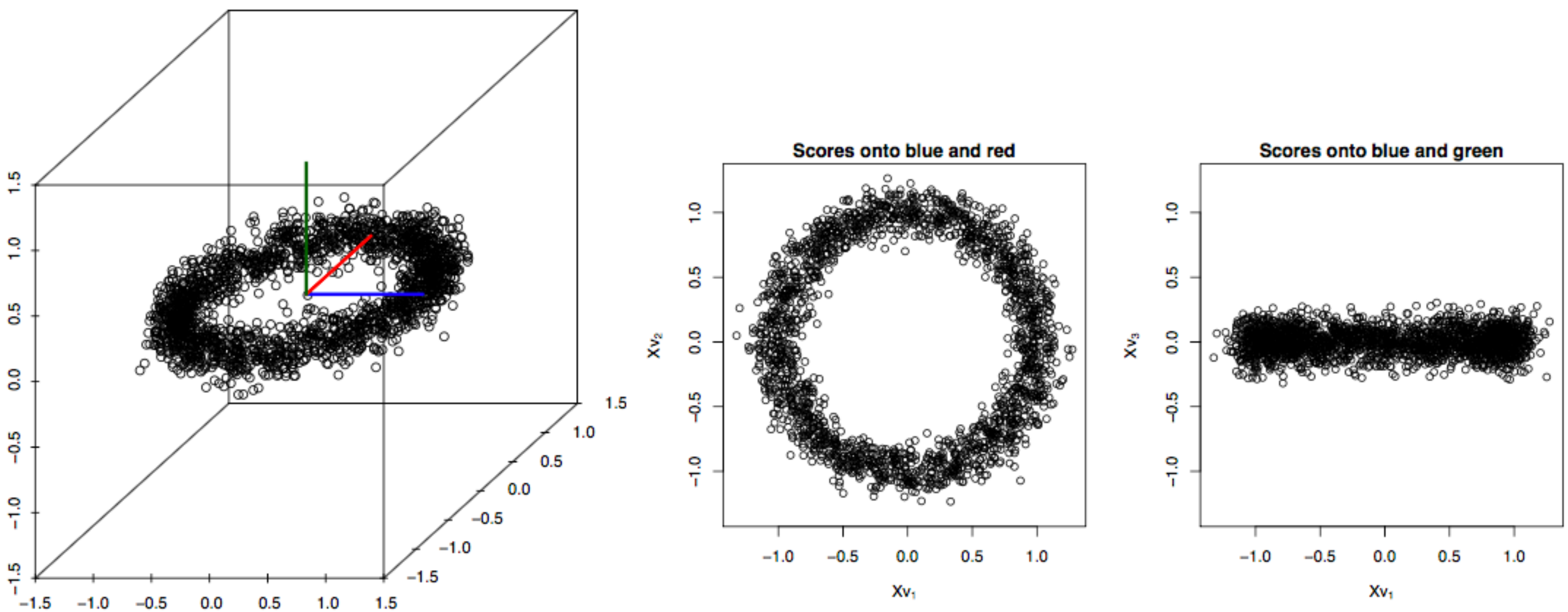
- Projection:

$$\mathbf{X}\mathbf{v} = \begin{bmatrix} \mathbf{x}_1^\top \mathbf{v}\mathbf{v}^\top \\ \vdots \\ \mathbf{x}_n^\top \mathbf{v}\mathbf{v}^\top \end{bmatrix}$$

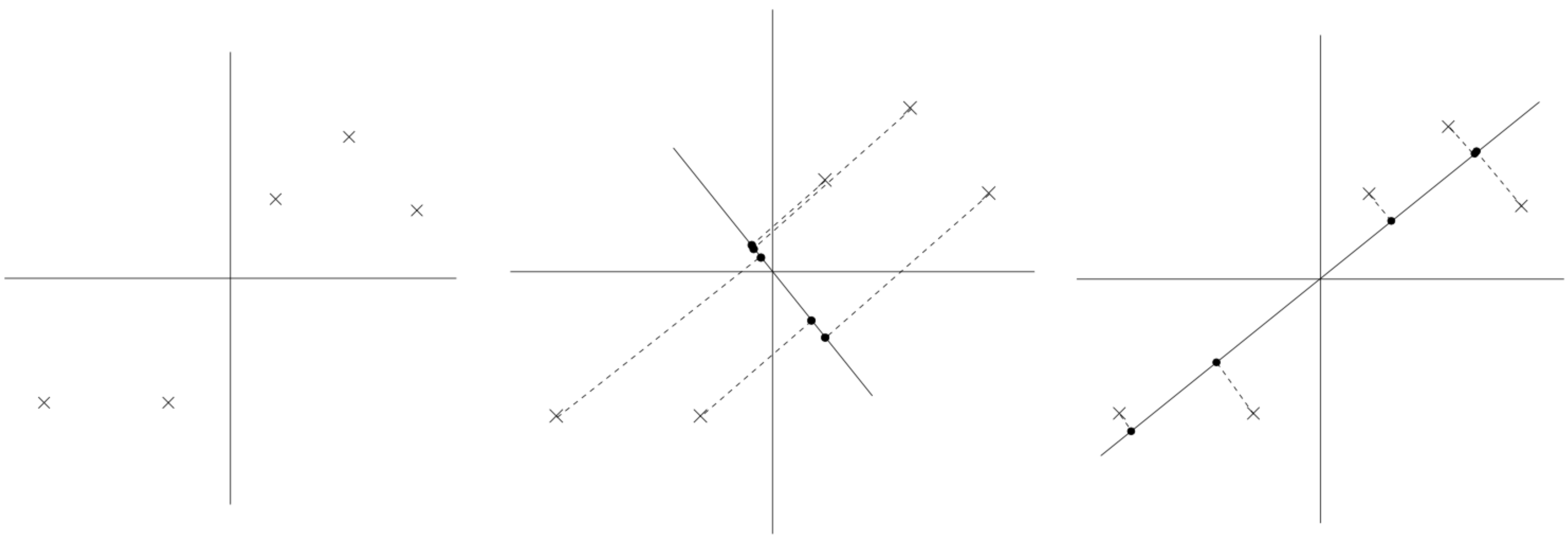
scores of projection

Review: Projection onto Orthonormal Vectors

Example: $X \in \mathbb{R}^{2000\times 3}$, and $v_1, v_2, v_3 \in \mathbb{R}^3$ are the unit vectors parallel to the coordinate axes



Which is Best Projection?



Notes from Andrew Ng

Principal Component Analysis

Principal Component Analysis (PCA)

- Developed by Pearson in 1901
- Popular and widely studied
- Finds sequence of linear combinations of the features (also known as principal components) that have maximal variance and are uncorrelated

PCA: 1st PC

- 1st PC of X is unit vector that maximizes the sample variance compared to all other unit vectors

$$\mathbf{v}_1 = \text{argmax}_{||\mathbf{v}||_2=1} (\mathbf{X}\mathbf{v})^\top (\mathbf{X}\mathbf{v})$$

- 1st PC score: $\mathbf{X}\mathbf{v}_1$

- Variance explained by first PC: $(\mathbf{X}\mathbf{v}_1)^\top (\mathbf{X}\mathbf{v}_1)/n$

PCA: Next PC

- Idea: Successively find orthogonal directions of highest variance
- Why orthogonal?
  - Want to minimize redundancy
  - Want to look at variance in different direction
  - Computation is easier

PCA: 2nd PC

- 2nd PC of X is unit vector that is orthogonal to the 1st PC such that it maximizes the sample variance compared to all other unit vectors that are orthogonal to the 1st PC

$$\mathbf{v}_2 = \text{argmax}_{||\mathbf{v}||_2=1, \mathbf{v}^\top \mathbf{v}_1 = 0} (\mathbf{X}\mathbf{v})^\top (\mathbf{X}\mathbf{v})$$

- 2nd PC score: $\mathbf{X}\mathbf{v}_2$

- Variance explained by 2nd PC: $(\mathbf{X}\mathbf{v}_2)^\top (\mathbf{X}\mathbf{v}_2)/n$

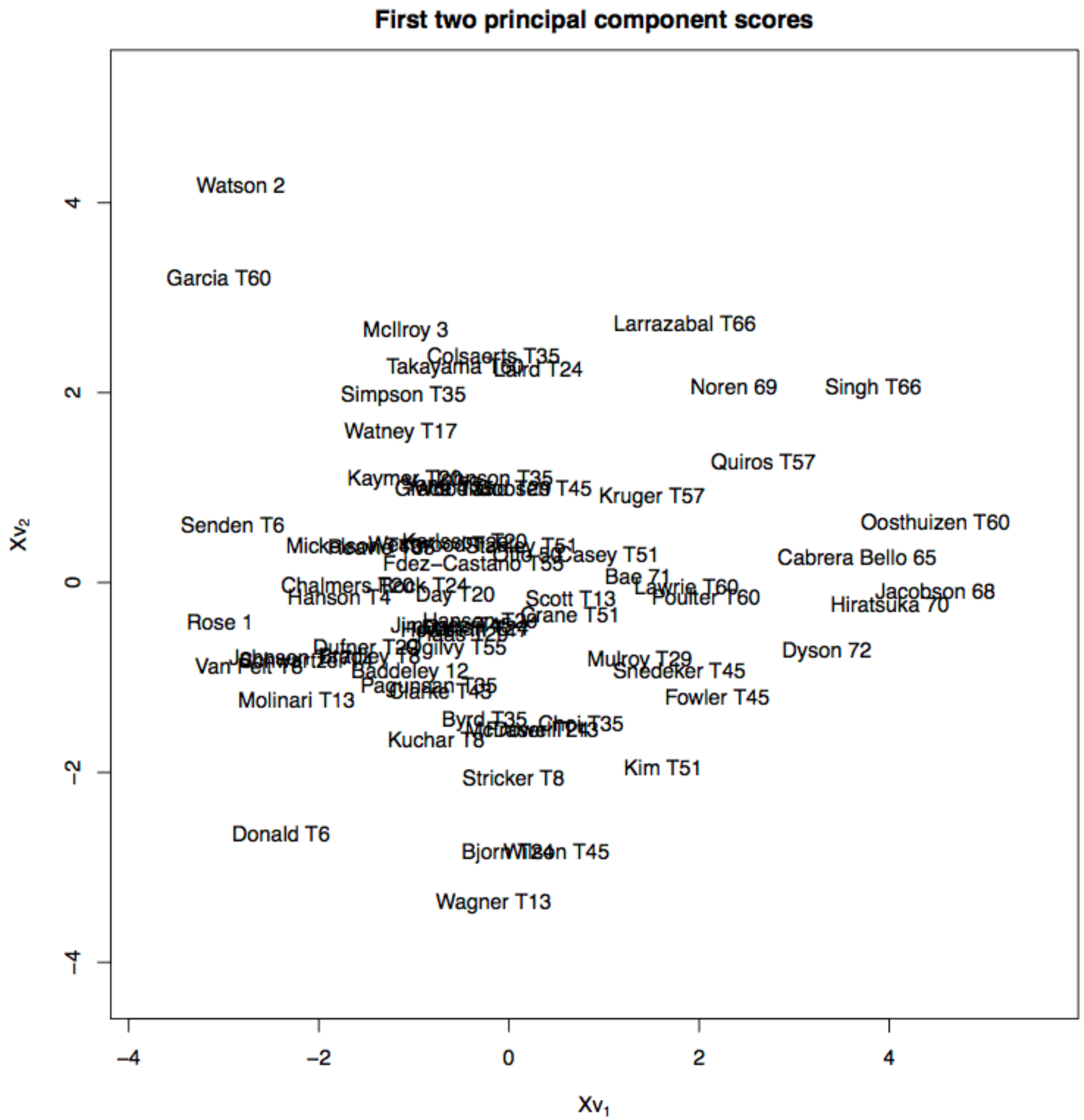
Example: 2012 Cadillac Championship

- 72 golfers with 12 features taken as average measurements from 4-day golf tournament
  - Eagles, birdies, pars, bogeys
  - Driving accuracy, driving distance
  - Strokes gained from putting, putts per round
  - …

Example: 2012 Cadillac Championship

	PC1	PC2
eagles	-0.139	0.208
birdies	-0.463	0.185
pars	0.168	-0.582
bogeys	0.303	0.420
double.bogeys	0.062	0.181
driving.accuracy	-0.128	-0.241
driving.distance	-0.036	0.430
strokes.gained.putting	-0.438	-0.091
putts.per.round	0.325	0.026
putts.per.gir	0.491	-0.158
greens.in.reg	-0.171	-0.099
sand.saves	-0.238	-0.296

Example: 2012 Cadillac Championship



PCA: Problem Formulation

- Given a feature matrix **X** with n data points, find **W** such that $||\mathbf{W}||_2 = 1$ and the Var(**XW**) is maximized and **W** consists of orthonormal vectors

$$\mathrm{Var}(\mathbf{X}\mathbf{W}) = \frac{1}{N}(\mathbf{W}^\top(\mathbf{X} - \mu_\mathbf{X})^\top(\mathbf{X} - \mu_\mathbf{X})\mathbf{W})$$

$$= \mathbf{W}^\top \boxed{\Sigma_\mathbf{X}} \mathbf{W}$$

Sample covariance matrix

- What does this look like?

Review: Symmetric Matrices

- Two remarkable properties
  - Eigenvalues of the matrix are real
  - Eigenvectors of the matrix are orthonormal

$$\mathbf{A} = \mathbf{U}\Lambda\mathbf{U}^\top$$

Review: Symmetric Matrices

- Two remarkable properties
  - Eigenvalues of the matrix are real
  - Eigenvectors of the matrix are orthonormal

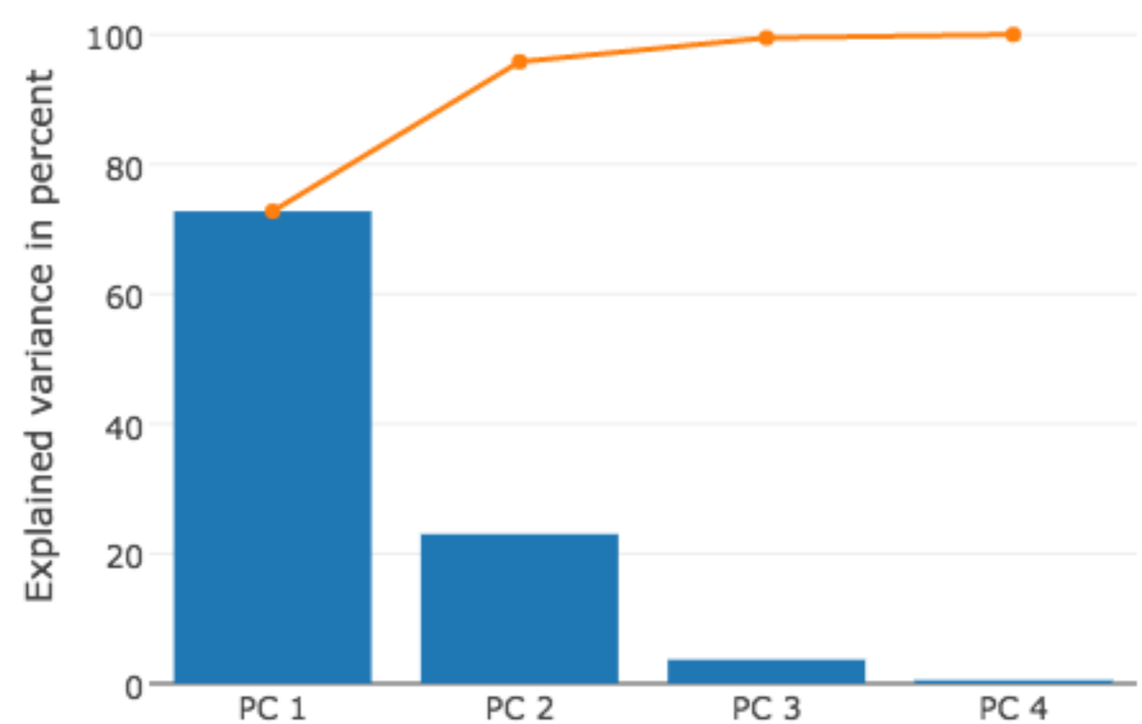
$$\mathbf{A} = \mathbf{U}\Lambda\mathbf{U}^\top$$

Basic PCA Algorithm

- Start with a zero-centered m x n data matrix **X**
- Compute covariance matrix
- Find eigenvectors of covariance matrix
- PCs: k eigenvectors with highest eigenvalues

PCA: Interpretation

- If variances of PCs drop off quickly, then X is highly collinear
- Reduce dimensionality of data by keeping only the PCs with highest variance
- Scree plot shows variance with the kth PC



https://plot.ly/ipython-notebooks/principal-component-analysis/

PCA: Minimum Projection Cost

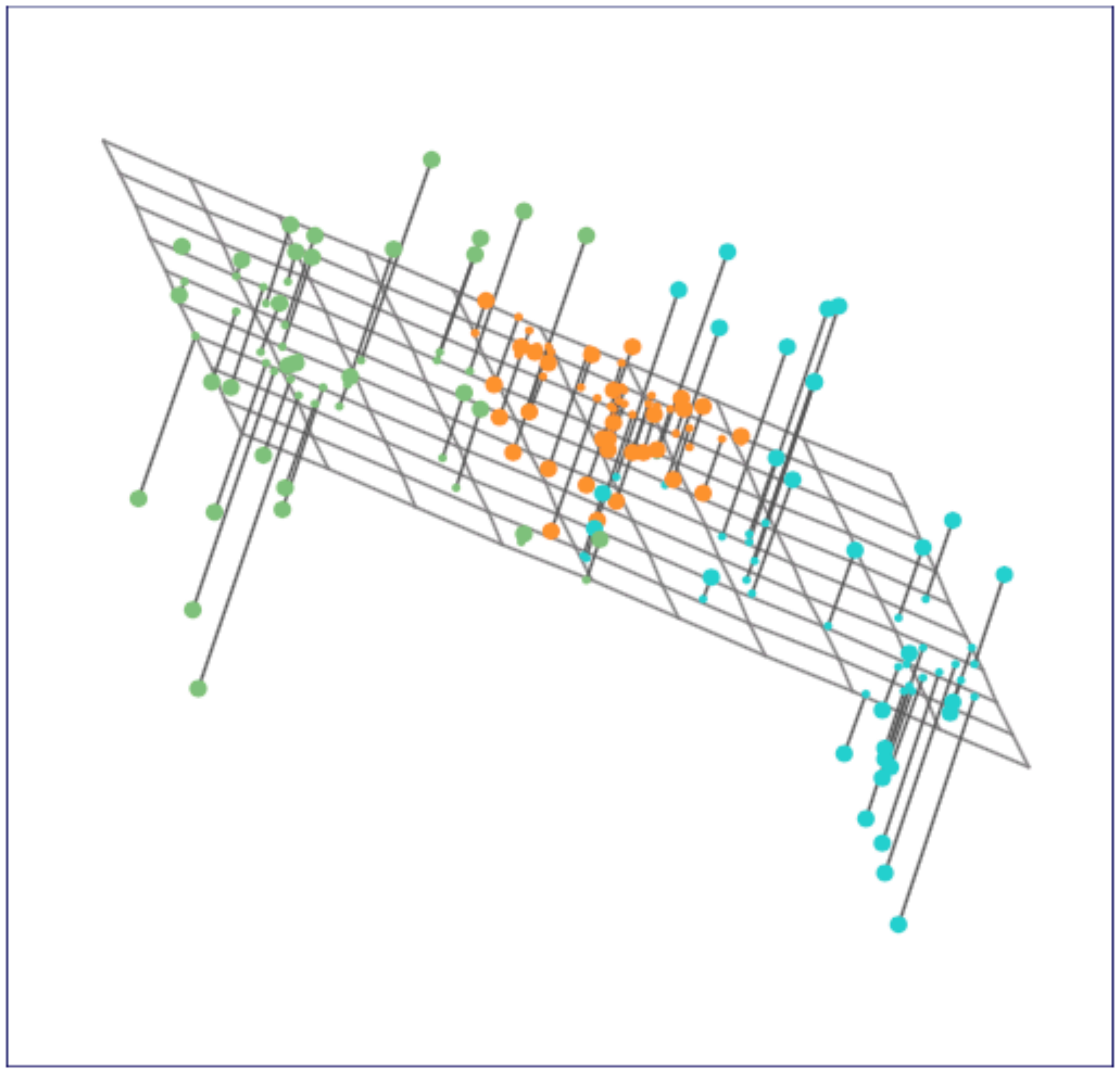
- Find projection onto principal subspace that minimizes the squared reconstruction error
  - Projection onto subspace

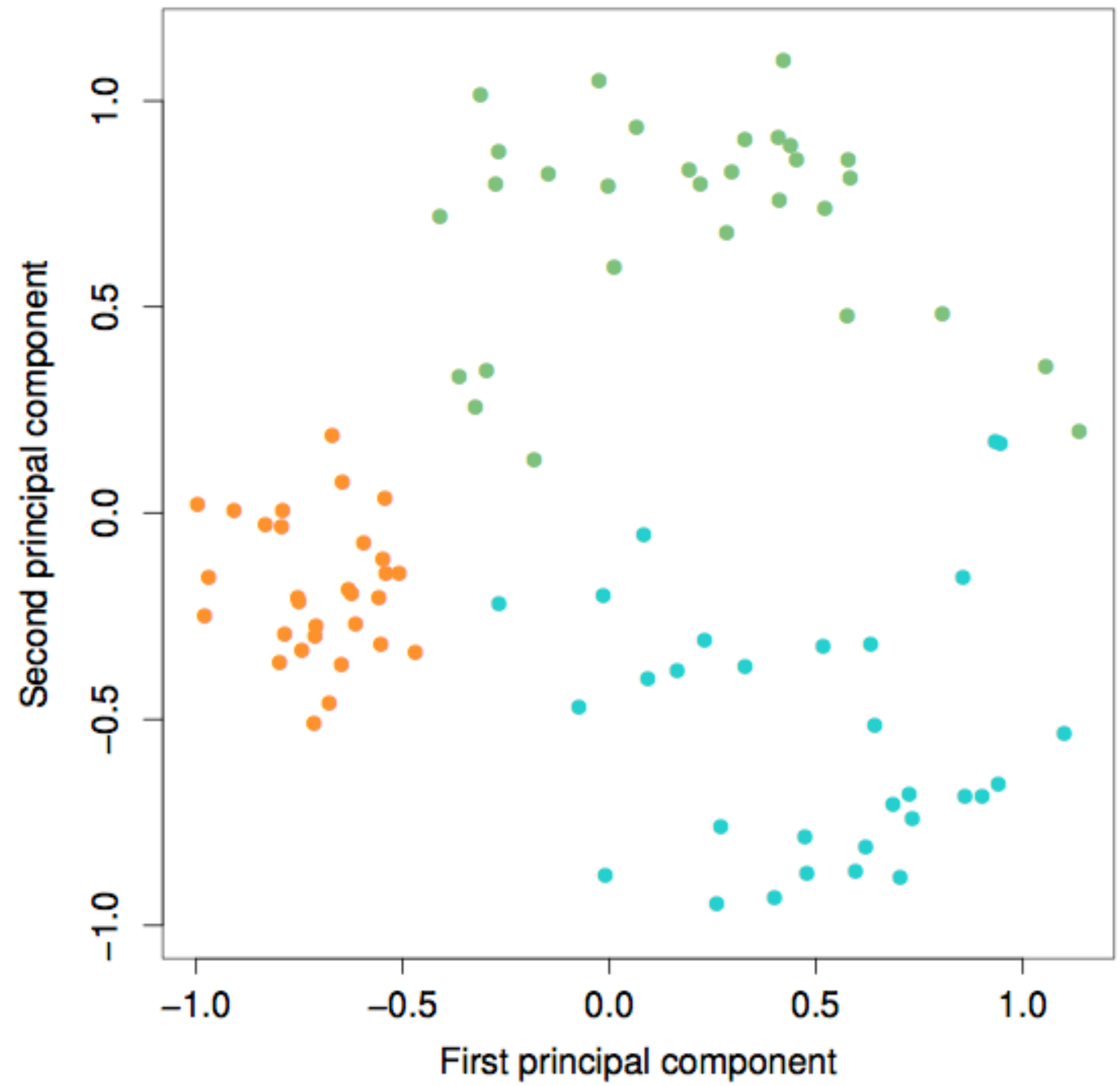
$$f(z) = \mu + \mathbf{w}_r \mathbf{z}$$

  - "Best fitting hyperplane":

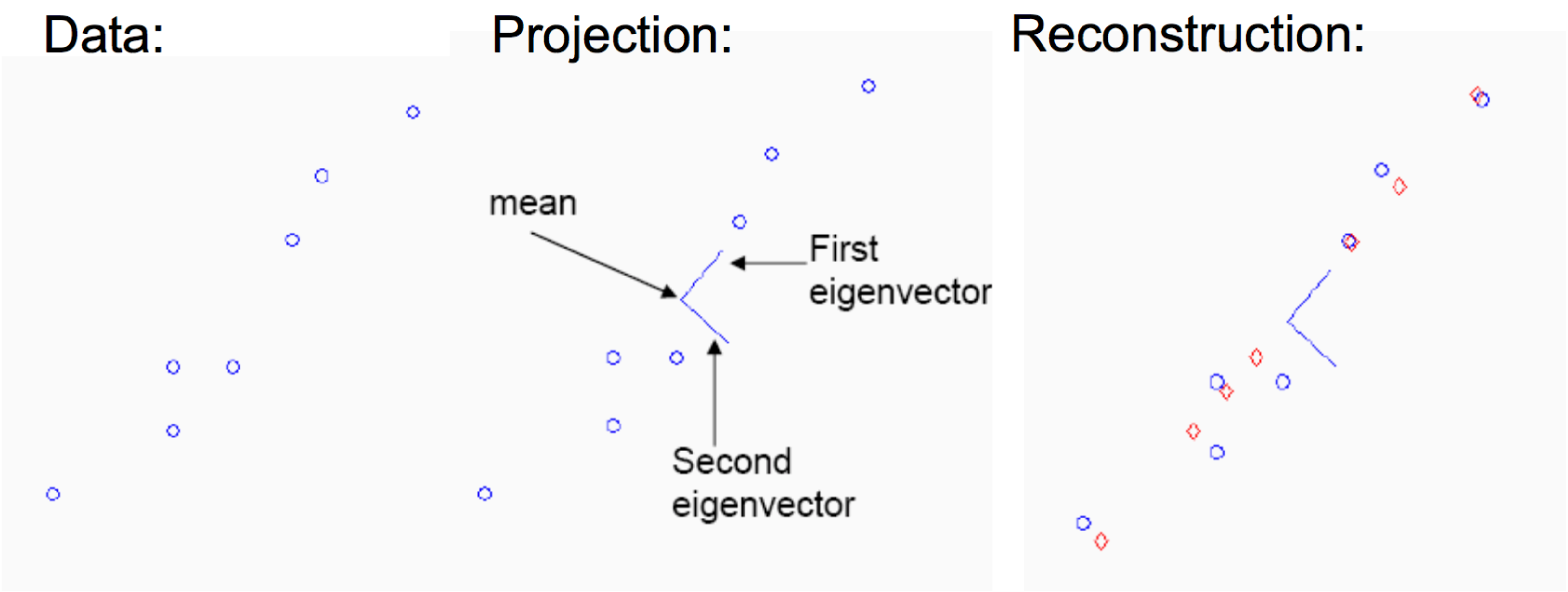
$$\min_{\mathbf{w}_r, \mathbf{z}_i} \sum_{i=1}^{n} ||\mathbf{x}_i - \mu - \mathbf{w}_r \mathbf{z}_i||_2^2$$

PCA: Minimum Projection Cost





PCA: Pictorially



Basic PCA Algorithm Revisited

- Start with a zero-centered m x n data matrix **X**
- Compute covariance matrix   what happens if n >> p?
- Find eigenvectors of covariance matrix
- PCs: k eigenvectors with highest eigenvalues

PCA: In Practice

- Forming the covariance matrix can require a lot of memory (number of samples >> number of features)
  - Need a faster way to compute this without forming the matrix explicitly
  - Typical approach: use singular value decomposition (SVD)

Singular Value Decomposition

- Each matrix can be decomposed using singular value decomposition (SVD):

$$\underbrace{\mathbf{X}}_{n\times p} = \underbrace{\mathbf{U}}_{n\times p}\underbrace{\mathbf{D}}_{p\times p}\underbrace{\mathbf{V}}_{p\times p}^{\top}$$

orthonormal columns which are principal components

orthonormal columns which are normalized PC scores

diagonal matrix which if each diagonal element is squared and divided by n gives variance explained

SVD & PCA

- Why does it work?

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$$
$$\Downarrow$$
$$\begin{aligned}\mathbf{X}^\top\mathbf{X} &= \mathbf{V}\mathbf{D}^\top\mathbf{U}^\top\mathbf{U}\mathbf{D}\mathbf{V}^\top \\ &= \mathbf{V}\mathbf{D}\mathbf{D}^\top\mathbf{V}^\top\end{aligned}$$

- Computing SVD of **X** gives us eigenvectors of covariance matrix and the eigenvalues!

SVD: A "Master" Algorithm

- Solve a linear system or any least squares problem
- Compute other factorizations: LU, QR, eigenvectors, etc.
- Standard algorithms are very stable, have only $O(n^3)$ asymptotic complexity and provide double precision accuracy

Review: Fisher's Linear Discriminant

- Find projection that maximizes ratio of between class variance to within class variance

$$\frac{\sigma^2_{\text{between}}}{\sigma^2_{\text{within}}} = \frac{(\mathbf{a}^\top(\mu_1 - \mu_2))^2}{\mathbf{a}^\top(\Sigma_1 + \Sigma_2)\mathbf{a}}$$

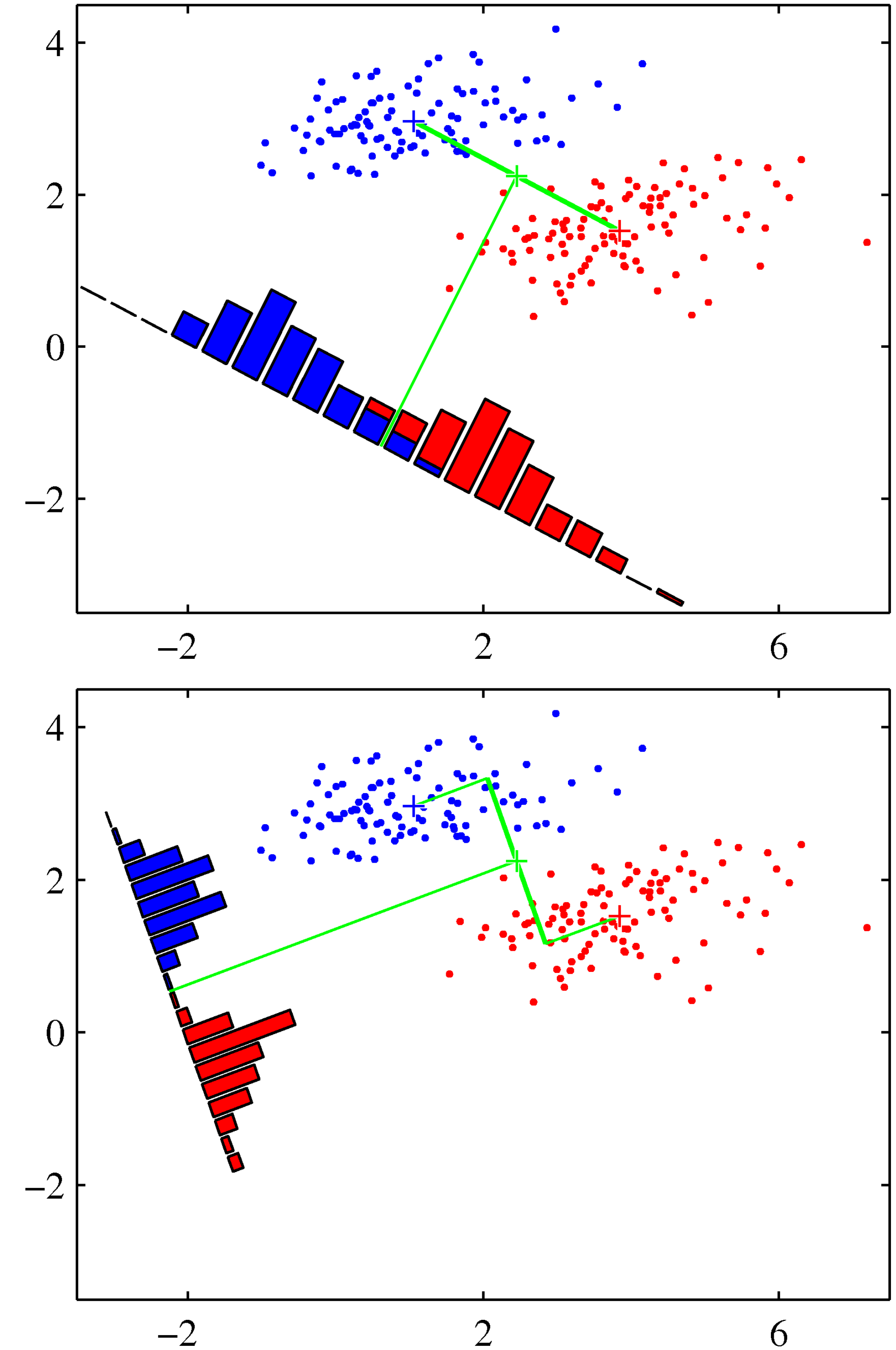
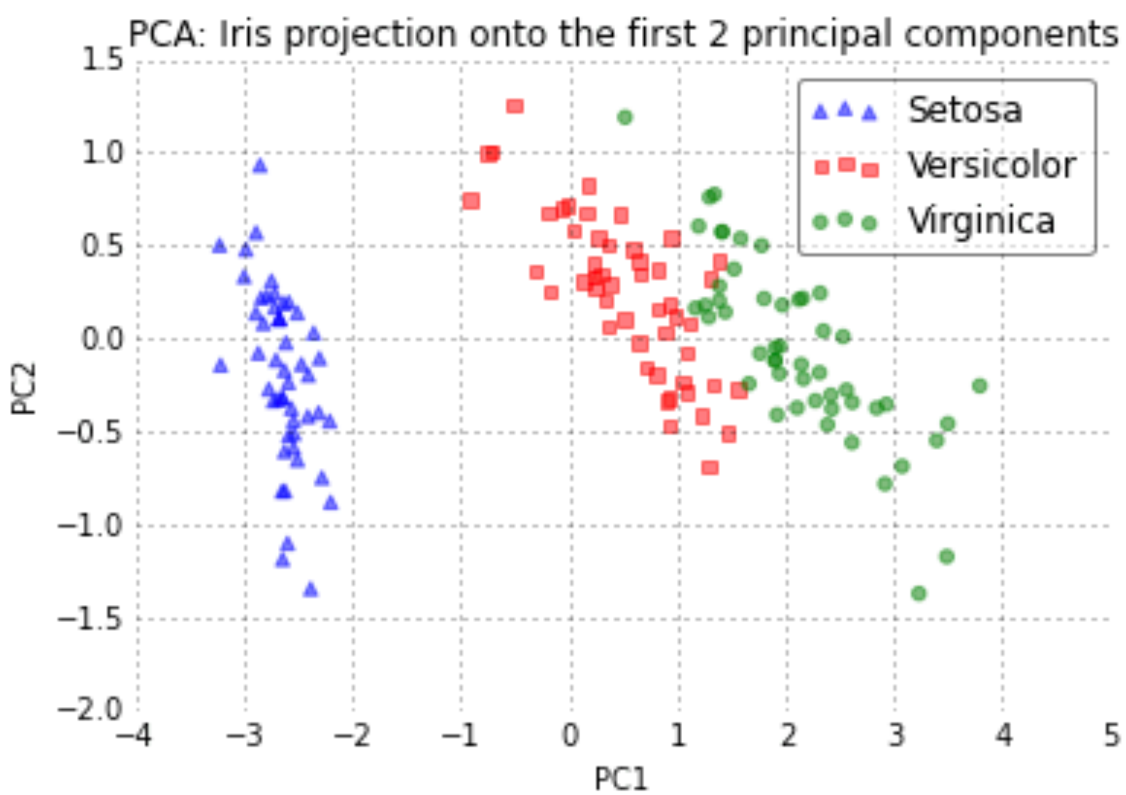


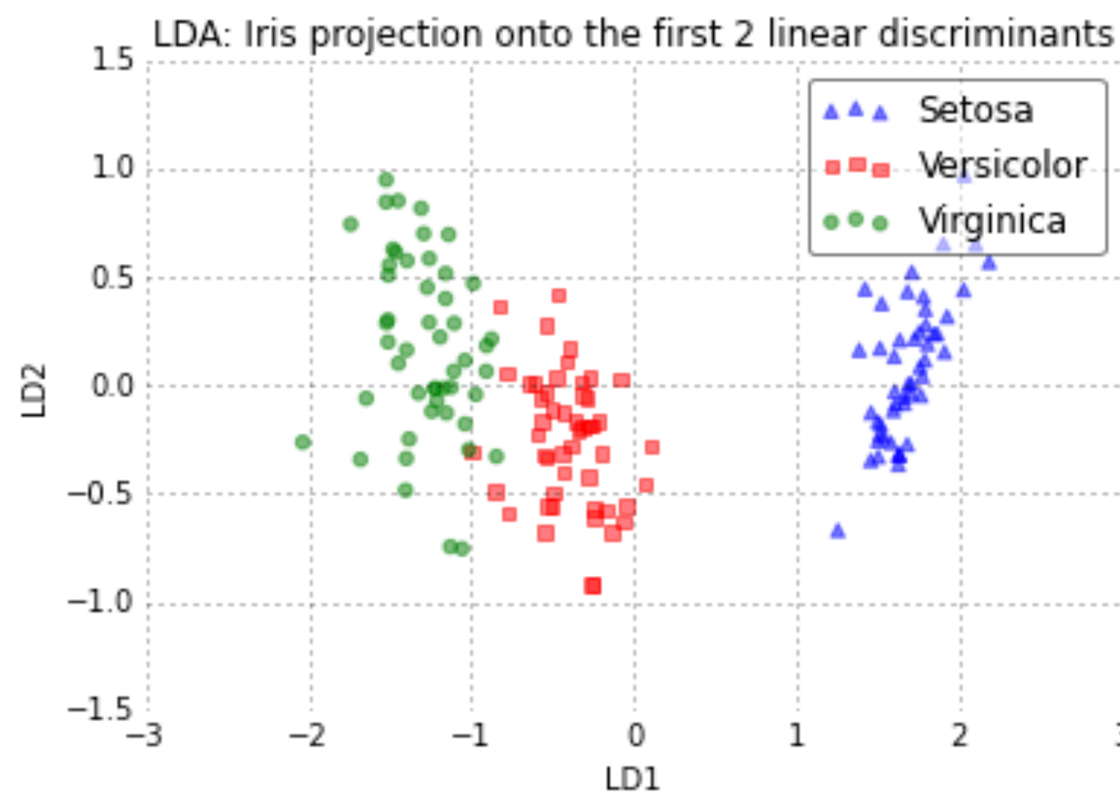
Figure 4.6 (Bishop)

PCA vs LDA

PCA



LDA



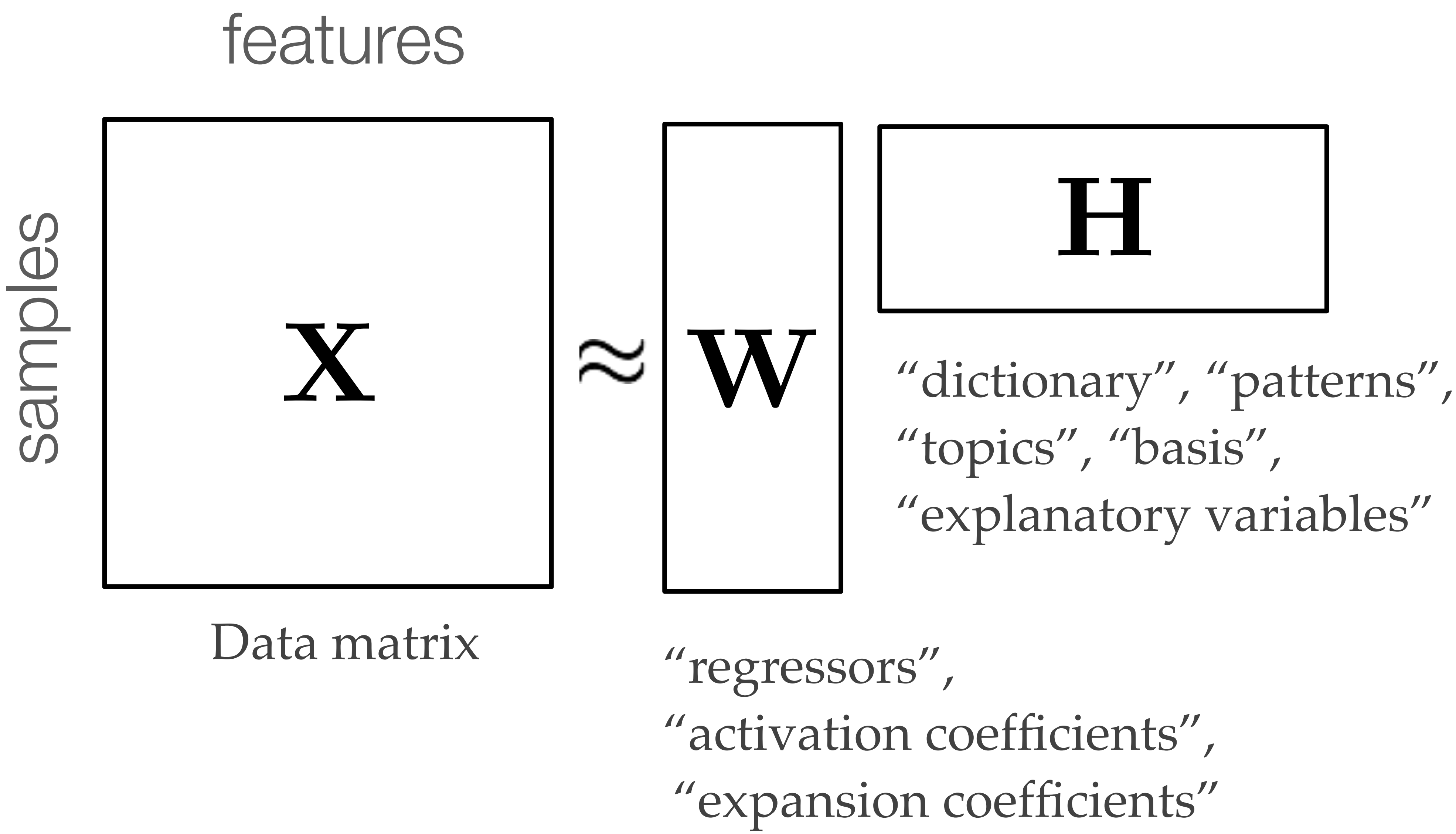
http://sebastianraschka.com/Articles/2014_intro_supervised_learning.html

Matrix Factorization

- Low rank approximation to original matrix
- Generalization of many methods (e.g., SVD, QR, CUR, Truncated SVD, etc.)
- Basic Idea: Find two (or more) matrices whose product best approximate the original matrix

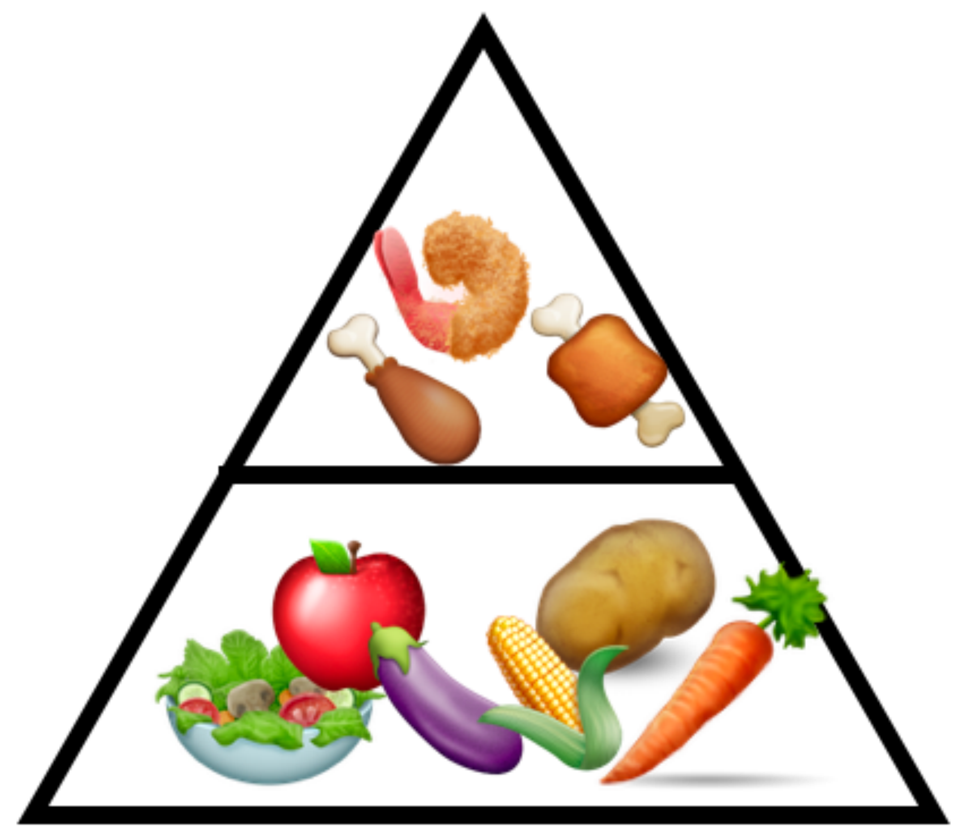
$$X \approx \underbrace{W}_{M \times R} \underbrace{H^\top}_{N \times R}, \; R << N$$

Matrix Factorization (Pictorially)



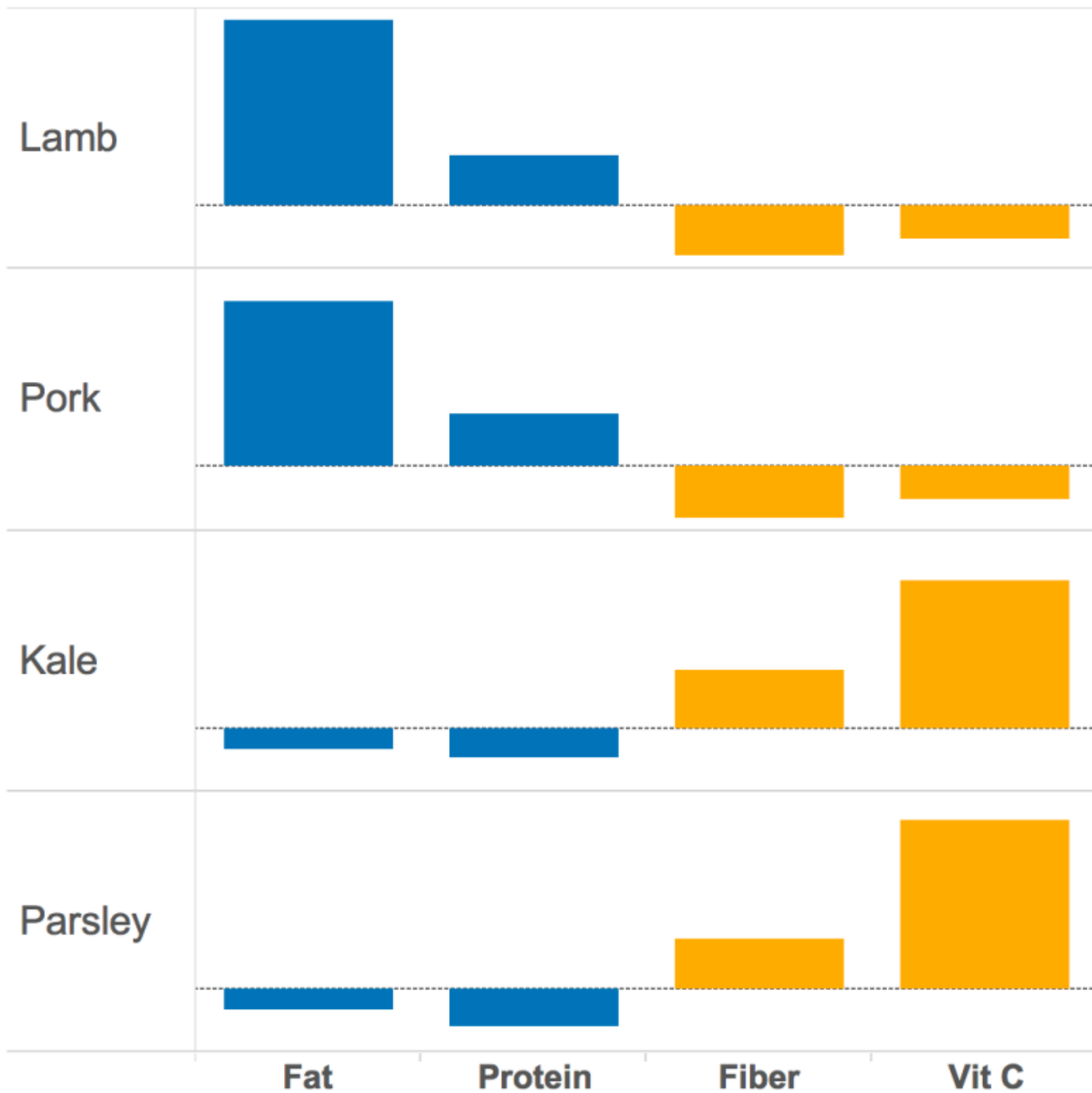
Example: Food Nutrition

- What is the best way to differentiate food items?
  - Vitamin content
  - Protein levels
  - Fat
  - Fiber



https://algobeans.com/2016/06/15/principal-component-analysis-tutorial/

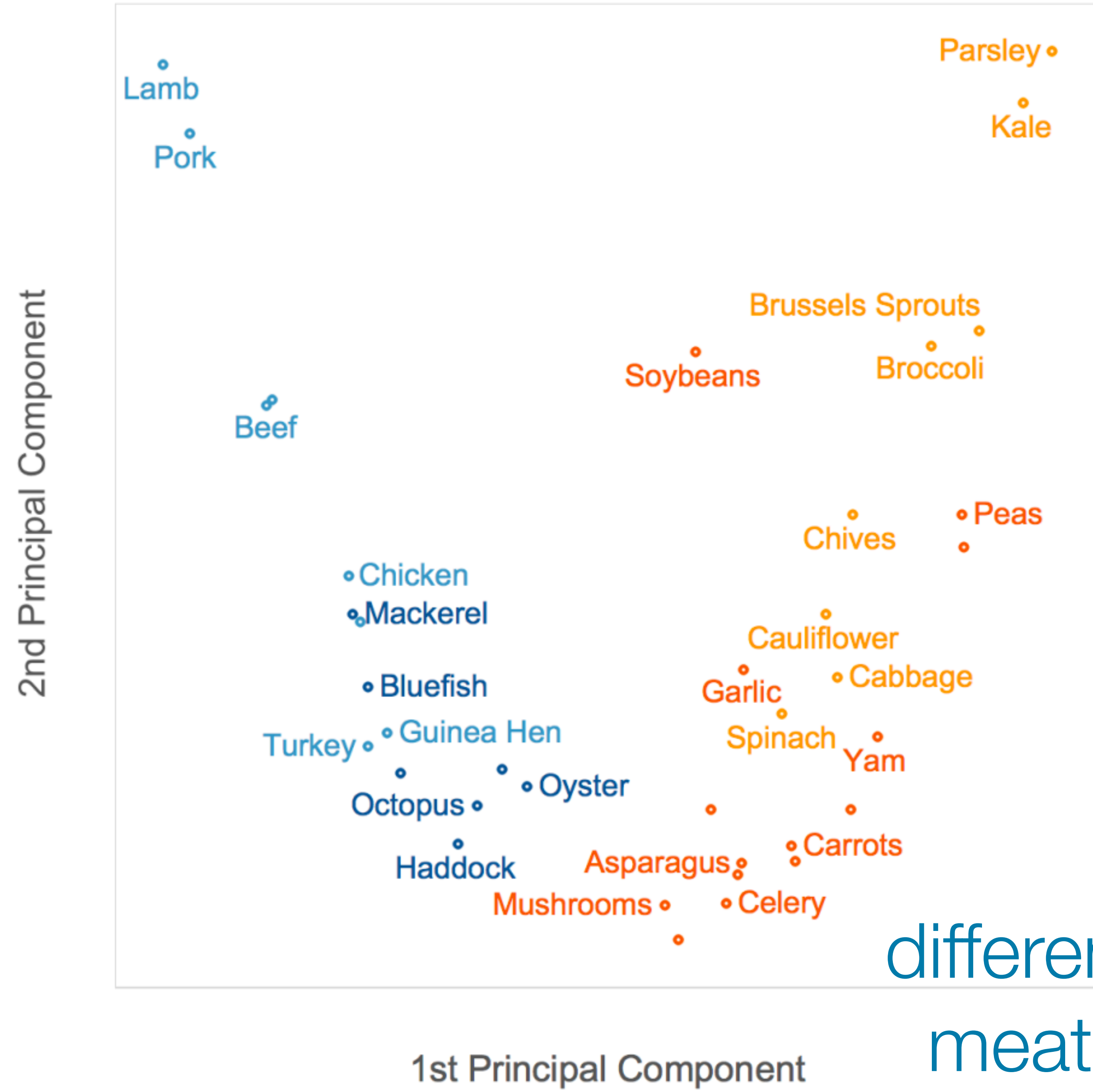
Example: Food Nutrition Data



https://algobeans.com/2016/06/15/principal-component-analysis-tutorial/

Example: PCA

differentiates between fat (meat) and vitamin c (vegetables)



differentiates between meat vs vegetables

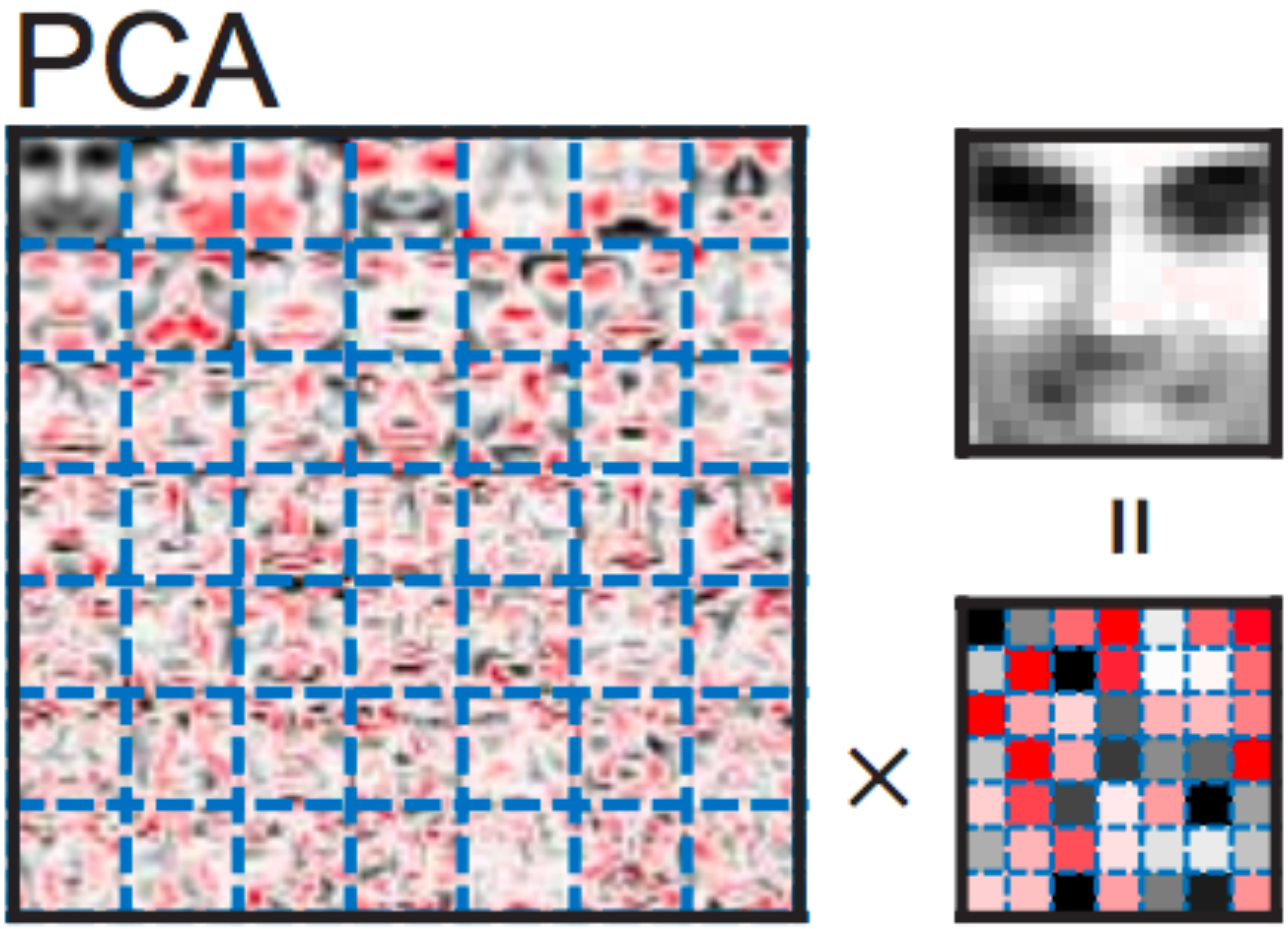
https://algobeans.com/2016/06/15/principal-component-analysis-tutorial/

Example: PCA Loadings

	PC1	PC2	PC3	PC4
Fat	-0.45	0.66	0.58	0.18
Protein	-0.55	0.21	-0.46	-0.67
Fiber	0.55	0.19	0.43	-0.69
Vitamin C	0.44	0.70	-0.52	0.22

What happens if negative combinations doesn't make sense?

Example: Face Representation



What does a negative pixel mean?

http://lsa.colorado.edu/LexicalSemantics/seung-nonneg-matrix.pdf

Non-negative Matrix Factorization

Nonnegative Matrix Factorization (NMF)

- Popularized by Lee and Seung (1999) for "learning the parts of objects"
- Both **W** and **H** are nonnegative
- Empirically induces sparsity
- Improved interpretability (sum of parts representation)

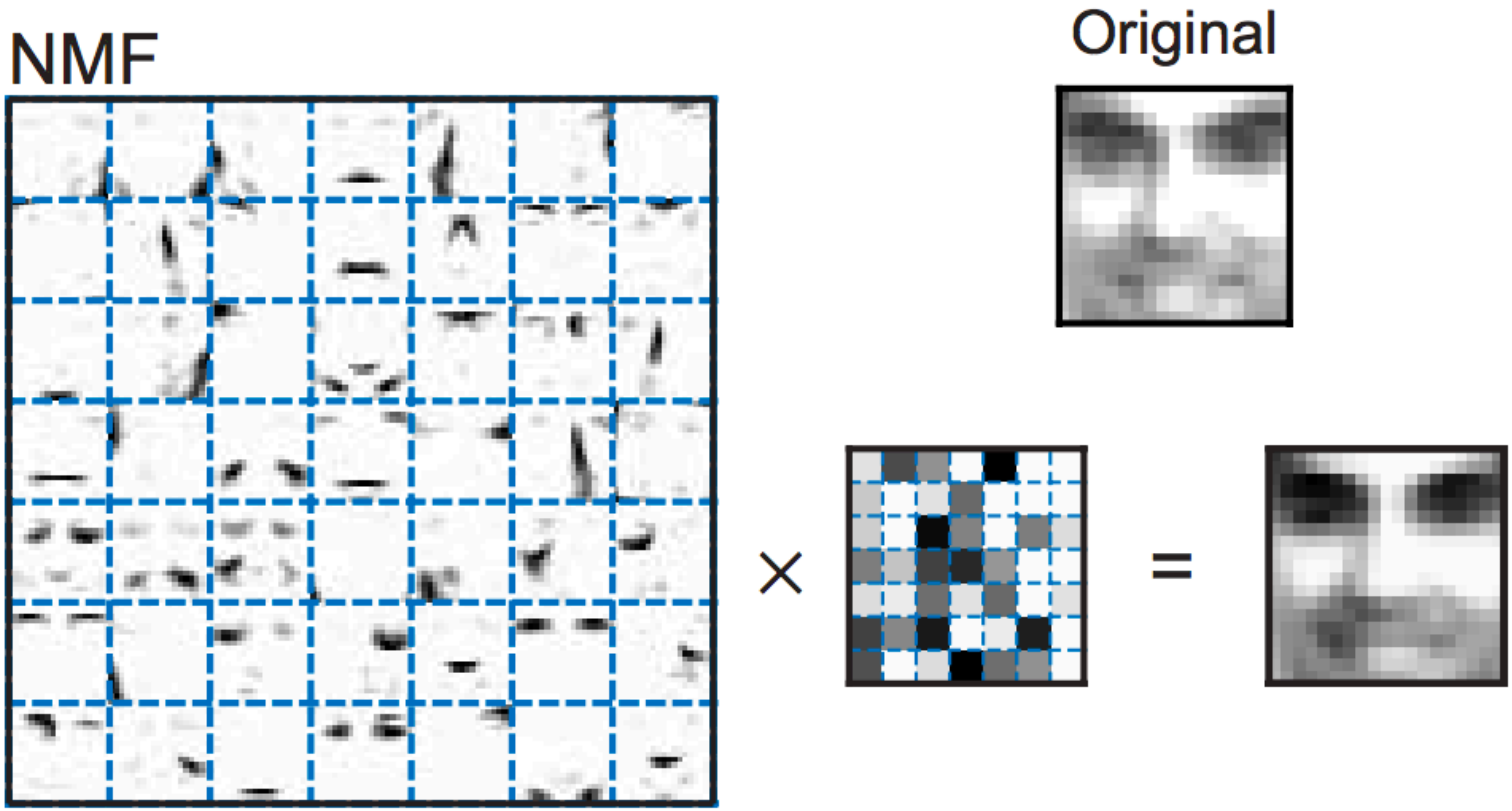
NMF: Algorithm

- Optimization problem

$$\min ||\mathbf{X} - \mathbf{WH}||_F$$
$$\text{s.t.} \mathbf{W} \geq 0, \mathbf{H} \geq 0$$

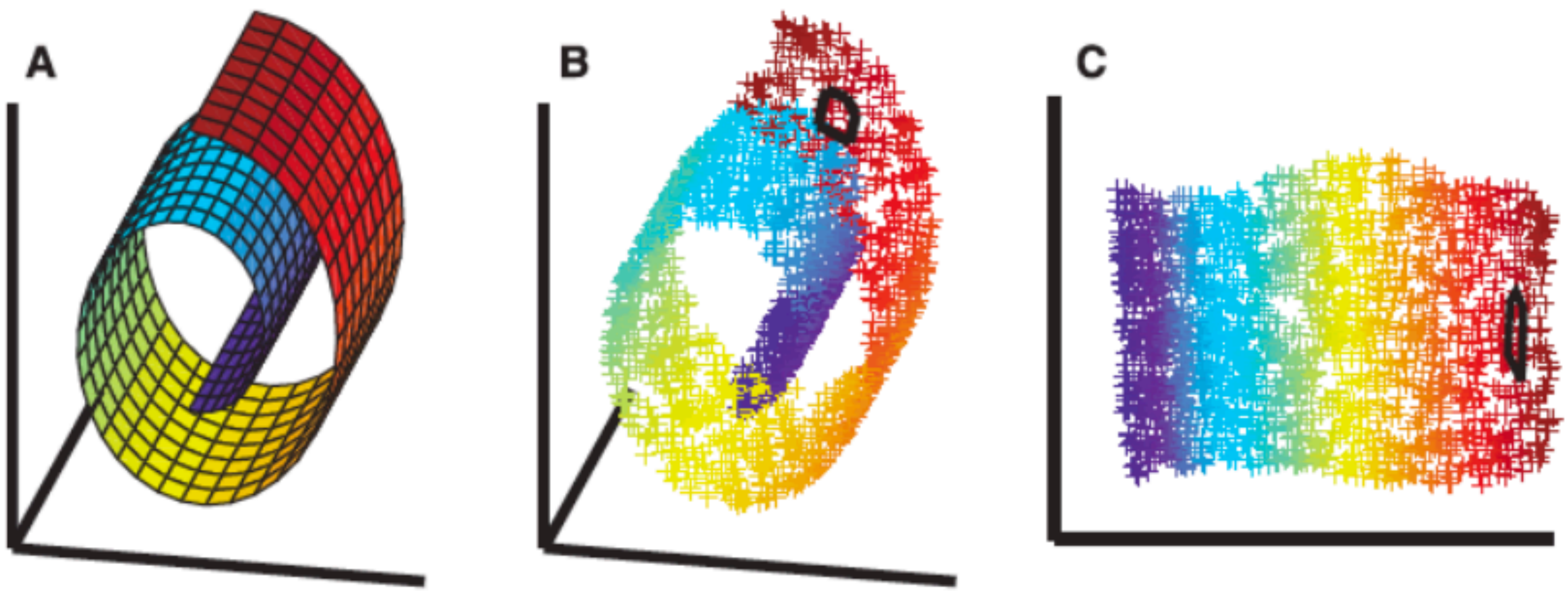
- Algorithm: Alternating minimization - given **W** find best **H**, given **H** find best **W**
  - Does not guarantee convergence to global optimum

Example: Face Representation



http://lsa.colorado.edu/LexicalSemantics/seung-nonneg-matrix.pdf

What About Non-Linear Data?



Roweis et al. (2000), "Nonlinear dimensionality reduction by locally linear embedding"

What if we only have distances between pairs of training points? Can we still learn low-dimensional representations?

Multidimensional Scaling (MDS)

- Given distance matrix $\boldsymbol{\Delta}$:
  - Recover the inner-product matrix $\mathbf{B} = \mathbf{X}\mathbf{X}^{\mathsf{T}}$

$$\mathbf{A}_{ij} = -\frac{1}{2}\boldsymbol{\Delta}_{ij}^2$$

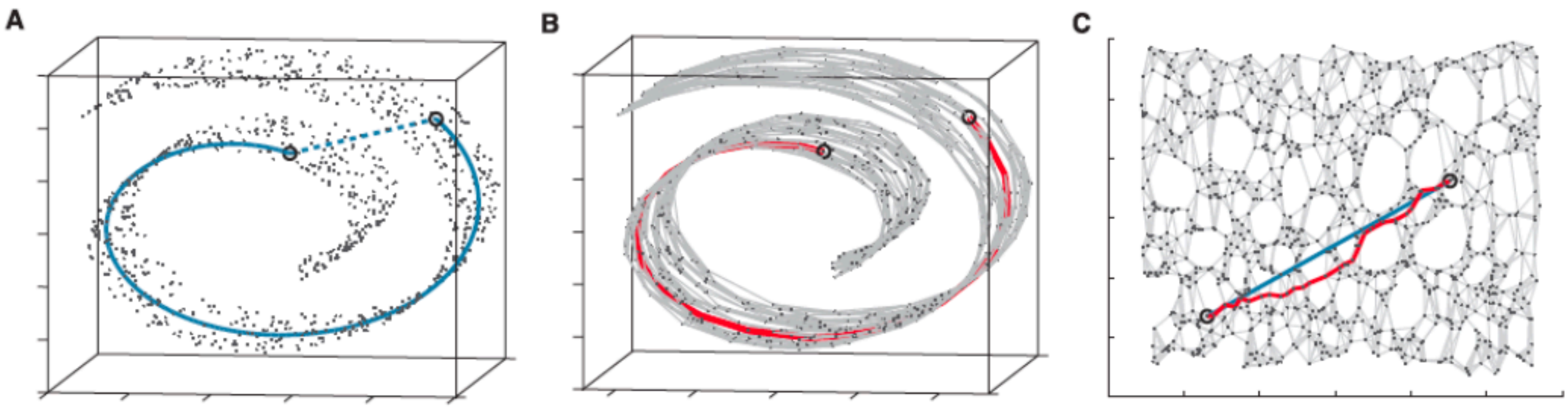
$$\mathbf{B} = (\mathbf{I} - \mathbf{M})\mathbf{A}(\mathbf{I} - \mathbf{M}), \ \mathbf{M} = \frac{1}{n}\mathbb{1}\mathbb{1}^\top$$

  - Factorize B to get the first k principal components

Isometric Feature Mapping (Isomap)

- Construct a graph based on the structure between points
  - Connect pair i, j with an edge if either i is one of j's m-nearest neighbors or j is one of i's m-nearest neighbors
  - Weight of edge is proportional to the distance between i and j
  - Define graph distance matrix based on shortest path between i and j
- Use MDS for low-dimensional representation

Example: Isomap

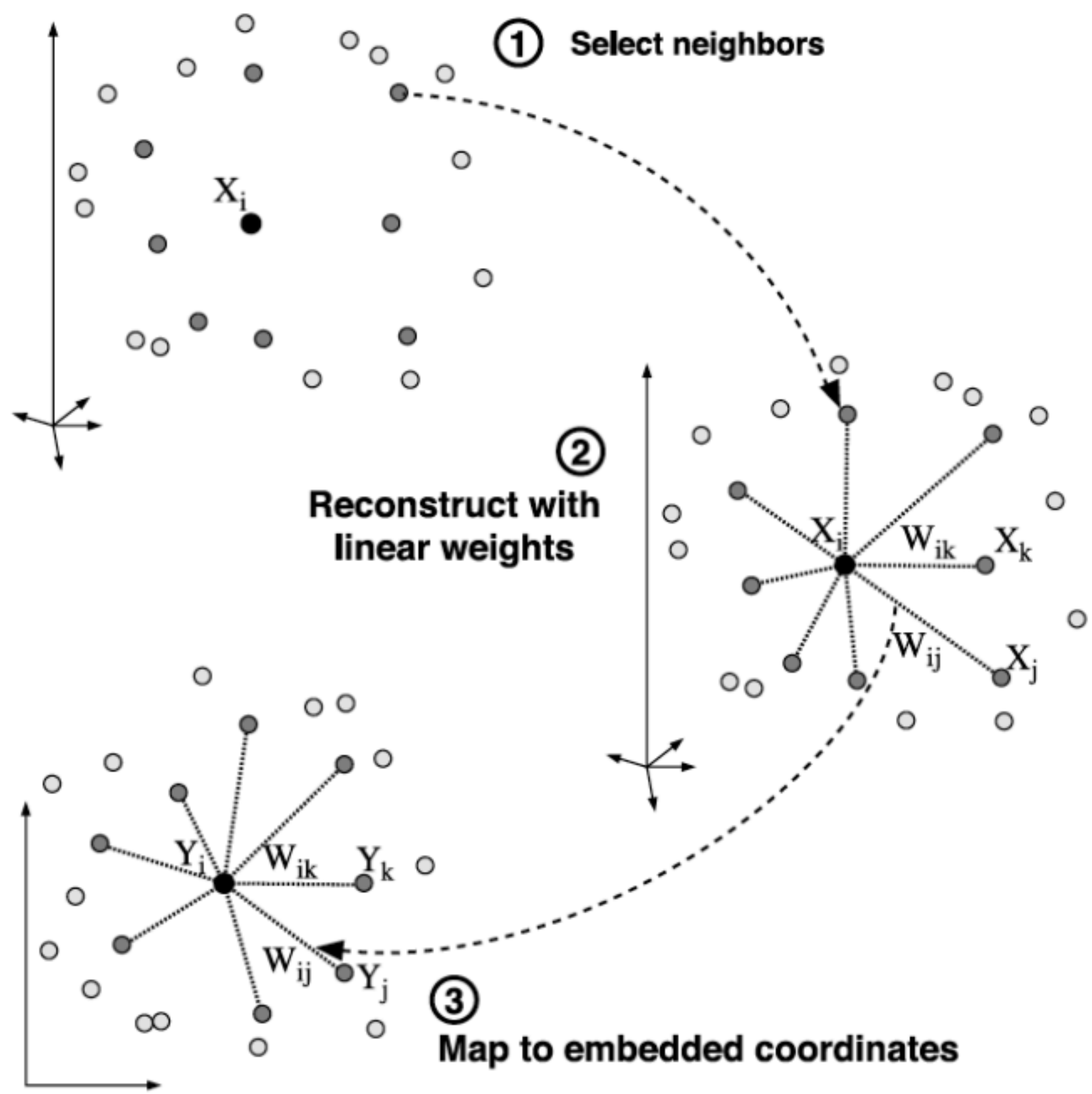


Tenenbaum et al. (2000), “A global geometric framework for nonlinear dimensionality reduction”

Local Linear Embedding (LLE)

- Idea:
  - Learn a bunch of local approximations (i.e., linear function to nearby points) to structure between the points
  - Learn a low-dimensional representation that best matches these local approximations

LLE: Illustration



Roweis et al. (2000), "Nonlinear dimensionality reduction by locally linear embedding"

Example: LLE

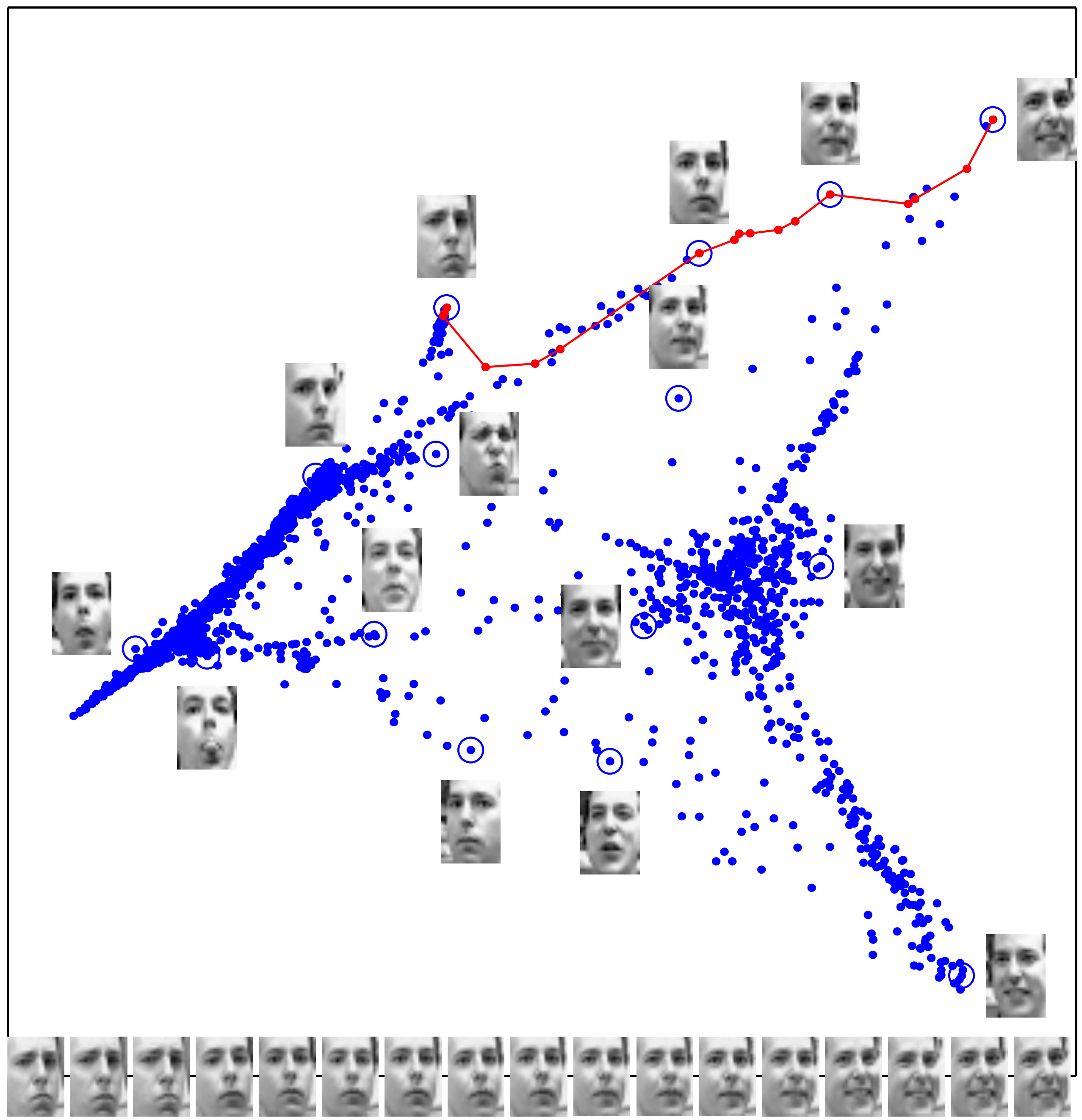


Figure 14.45 (Hastie et al.)