CS150: Database & Datamining
Lecture 25: Analytics & Machine Learning VI

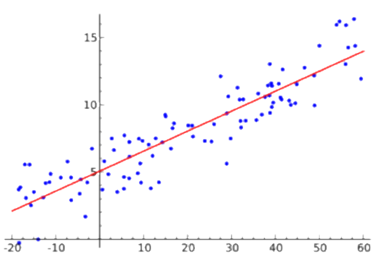
Xuming He
Spring 2019

Acknowledgement: Slides are adopted from the Berkeley course CS186 by Joey Gonzalez and Joe Hellerstein, Toronto CSC411 by Rich Zemel, Caltech CS155 by Yisong Yue.

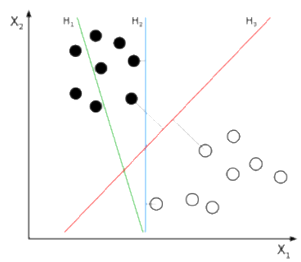
Taxonomy
of Machine Learning

- Labeled Data → **Supervised Learning**
  - Regression
  - Classification
- Indirect (reward) → **Reinforcement & Bandit Learning**
- Unlabeled Data → **Unsupervised Learning**
  - Dimensionality Reduction
  - Clustering

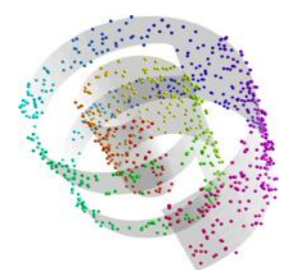
Regression



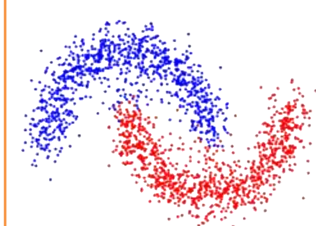
Classification



Dimensionality Reduction



Clustering



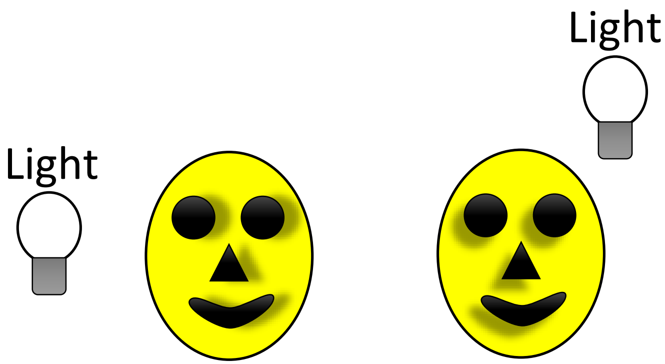
Dimensionality Reduction

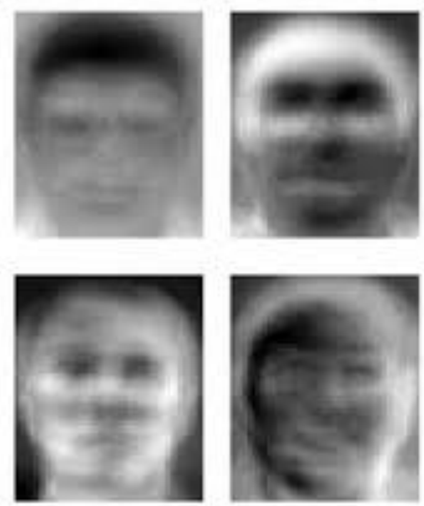
- Given images under different viewing conditions
  - What are the intrinsic latent dimensions in these two datasets?

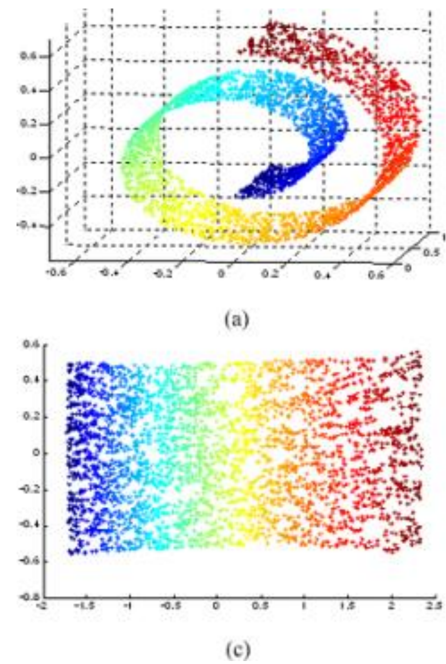
  - How can we find these dimensions from the data?

Dimensionality Reduction:

➢ Given images under different viewing conditions







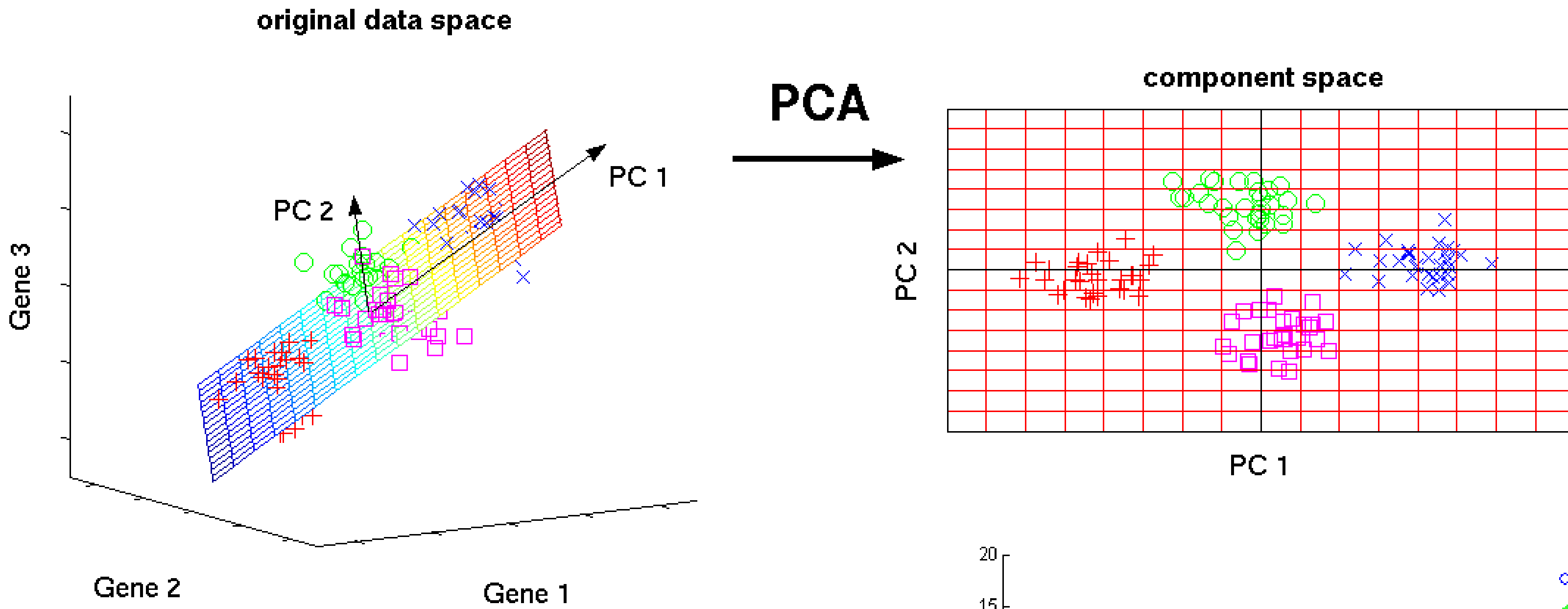
➢ **Machine Learning Approach:**

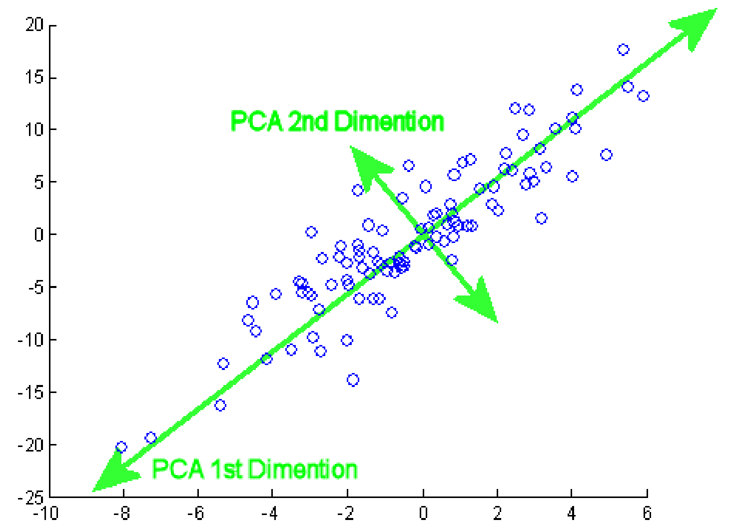
Embedding(*Image*; θ) → {x_1, x_2, x_3, x_4}

Recovery({x_1, x_2, x_3, x_4}; θ) → *Reconstructed Image*

➢ Use common structure in data to identify embedding

Principal Component Analysis





Big Ideas

- Identify dimensions of maximum variance
- Project data onto those dimensions

Principal Component Analysis

- Handle high-dimensional data
  - Avoid overfitting
- Data live in much lower dimensional space
- Useful for
  - Visualization
  - Preprocessing
  - Modeling – prior for new data
  - Compression

Finding Principal Components

- To find the principal component directions, we center the data (subtract the sample mean from each variable)
- Calculate the empirical covariance matrix:

$$C = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}^{(n)} - \bar{\mathbf{x}})(\mathbf{x}^{(n)} - \bar{\mathbf{x}})^T$$

  with $\bar{\mathbf{x}}$ the mean
- What's the dimensionality of C?
- Find the M eigenvectors with largest eigenvalues of C: these are the principal components
- Assemble these eigenvectors into a $D \times M$ matrix U
- We can now express D-dimensional vectors $\mathbf{x}$ by projecting them to M-dimensional $\mathbf{z}$

$$\mathbf{z} = U^T\mathbf{x}$$

Scaling Principal Component Analysis

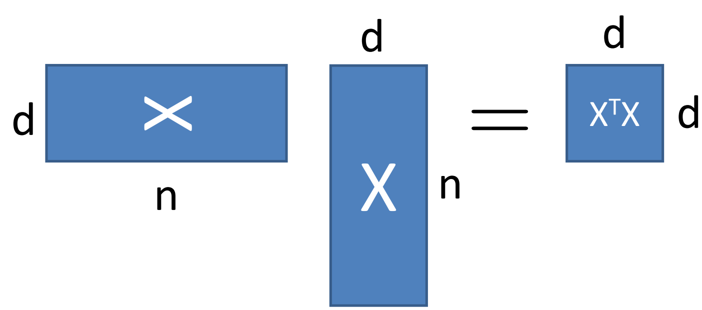
- **PCA Algorithm**
  - Computes eigenvectors of of covariance matrix

$$\mathbf{Cov}(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T = \frac{1}{n}X^T X - \bar{x}\bar{x}^T$$

  - The covariance matrix *d×d* is generally smaller than X (*n×d)*

- We therefore only need to compute:

$$X^T X =$$



$$= \sum_{i=1}^{n} x_i x_i^T$$

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i =$$

- In summation form
- Only one pass required!

PCA to faces

- Run PCA on 2429 19x19 grayscale images (CBCL data)
- Compresses the data: can get good reconstructions with only 3 components

- PCA for pre-processing: can apply classifier to latent representation
  - PCA with 3 components obtains 79% accuracy on face/non-face discrimination on test data vs. 76.8% for GMM with 84 states
- Can also be good for visualization

PCA for Anomaly Detection

- Run PCA and get top k eigenvectors: $V_{(k)}$

$$\mathbf{Proj}(x) = V_{(k)}^T (x - \bar{x})$$

$$\mathbf{Recv}(q) = V_{(k)} q + \bar{x}$$

- Compute the error in approximate recovery:

$$\mathbf{Error}(x) = \| x - \mathbf{Recv}\left(\mathbf{Proj}\left(x\right)\right) \|_2^2$$

- Outliers are those points far from their embedding