Principal Component Analysis

STAT5241 Section 2

Statistical Machine Learning

Xiaofei Shi

Data Visualization

- How can we visualize data?
- Example: 53 features for 65 people?

- Matrix format (65x53)

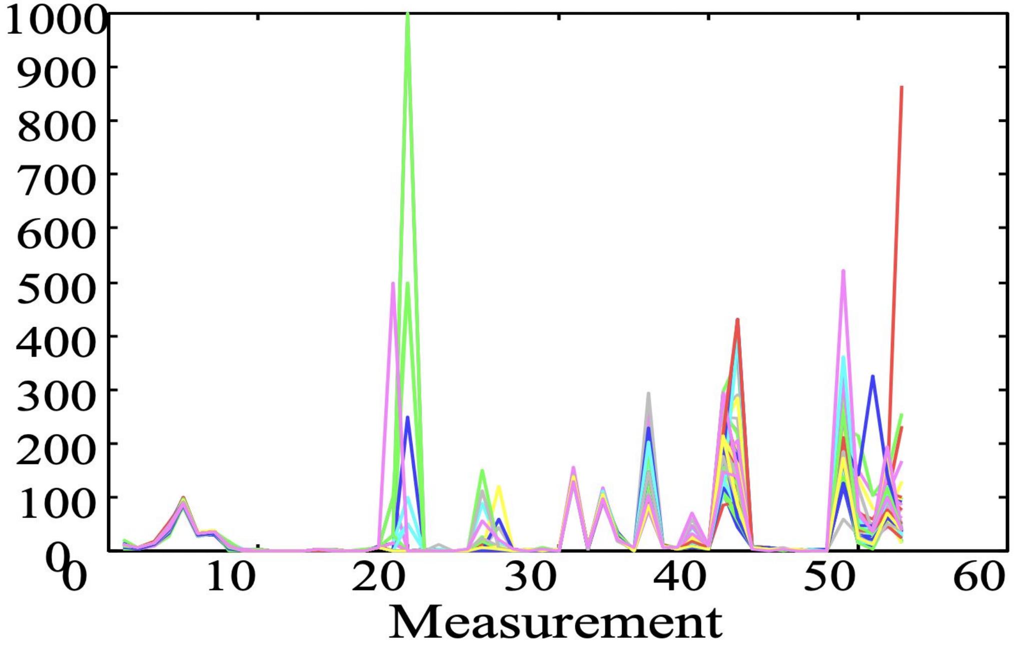
Instances

	H-WBC	H-RBC	H-Hgb	H-Hct	H-MCV	H-MCH	H-MCHC
A1	8.0000	4.8200	14.1000	41.0000	85.0000	29.0000	34.0000
A2	7.3000	5.0200	14.7000	43.0000	86.0000	29.0000	34.0000
A3	4.3000	4.4800	14.1000	41.0000	91.0000	32.0000	35.0000
A4	7.5000	4.4700	14.9000	45.0000	101.0000	33.0000	33.0000
A5	7.3000	5.5200	15.4000	46.0000	84.0000	28.0000	33.0000
A6	6.9000	4.8600	16.0000	47.0000	97.0000	33.0000	34.0000
A7	7.8000	4.6800	14.7000	43.0000	92.0000	31.0000	34.0000
A8	8.6000	4.8200	15.8000	42.0000	88.0000	33.0000	37.0000
A9	5.1000	4.7100	14.0000	43.0000	92.0000	30.0000	32.0000

Features

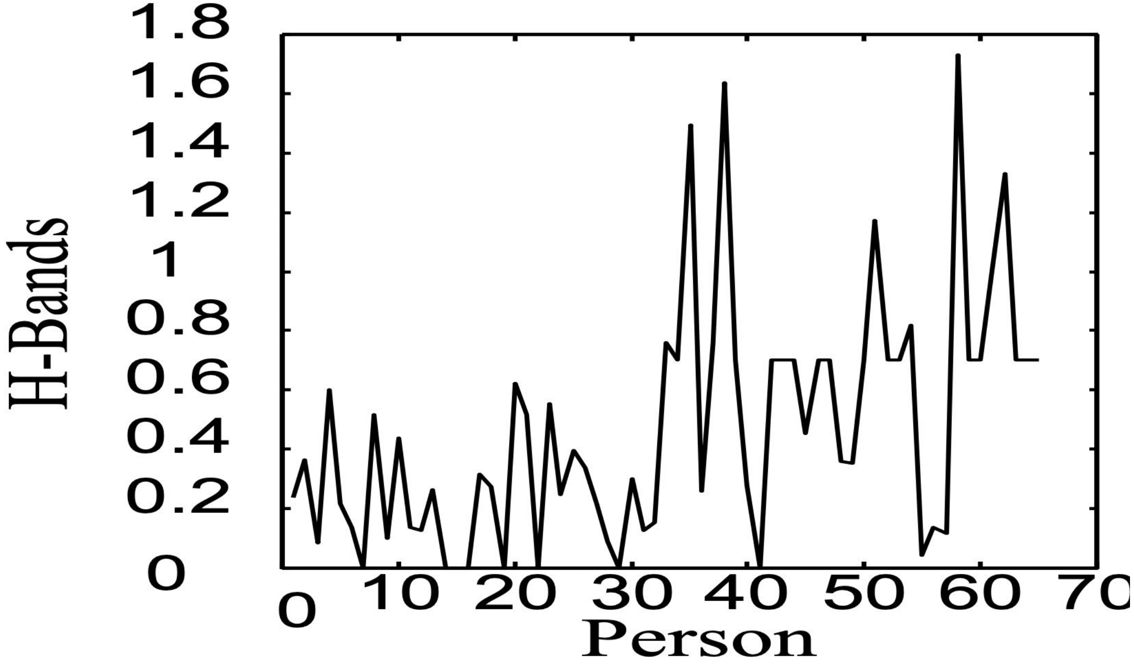
Difficult to see the correlations between the features...

- Curves (65 curves, one for each person)

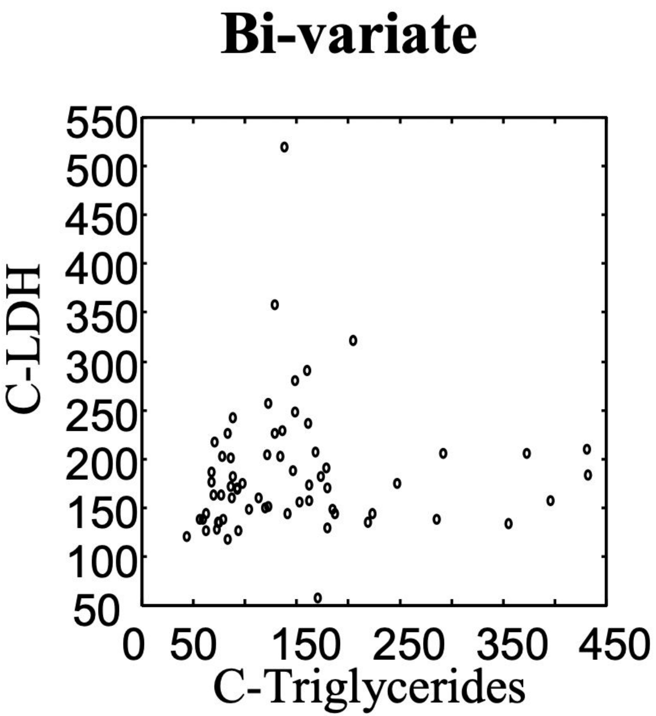


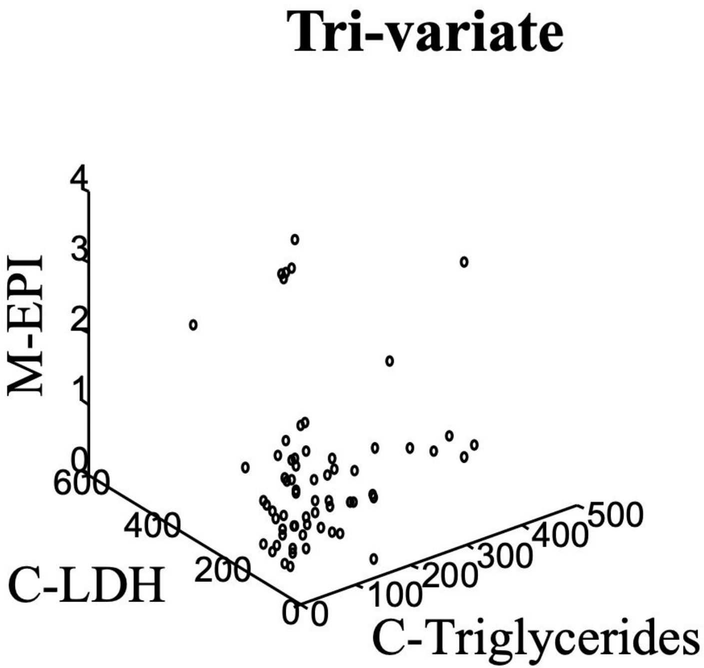
Difficult to compare the different patients...

- Curves (53 pictures, one for each feature)



Difficult to see the correlations between the features...





Columbia University

Moreover…

- Is there a representation better than the coordinate axes?

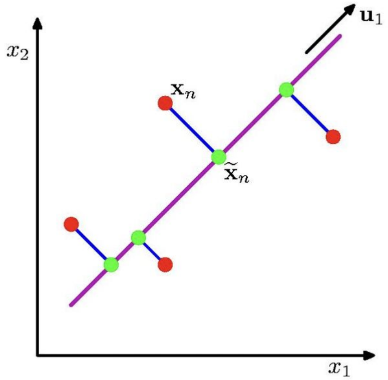
- Is it really necessary to show all the 53 dimensions?
  - … what if there are strong correlations between the features?

- How could we find
  the *smallest* subspace of the 53-D space that
  keeps the *most information* about the original data?

- A solution: **Principal Component Analysis**

COLUMBIA UNIVERSITY

PCA



PCA:

- Orthogonal projection of the data onto a lower-dimension linear space that <u>equivalently</u>...
  1. *minimizes* the mean squared distance between
     - data points (red points) and projections (green points)
     - i.e. sum of squares of blue line lengths
  2. *maximizes* variance of projected data (green points)

Columbia University

PCA

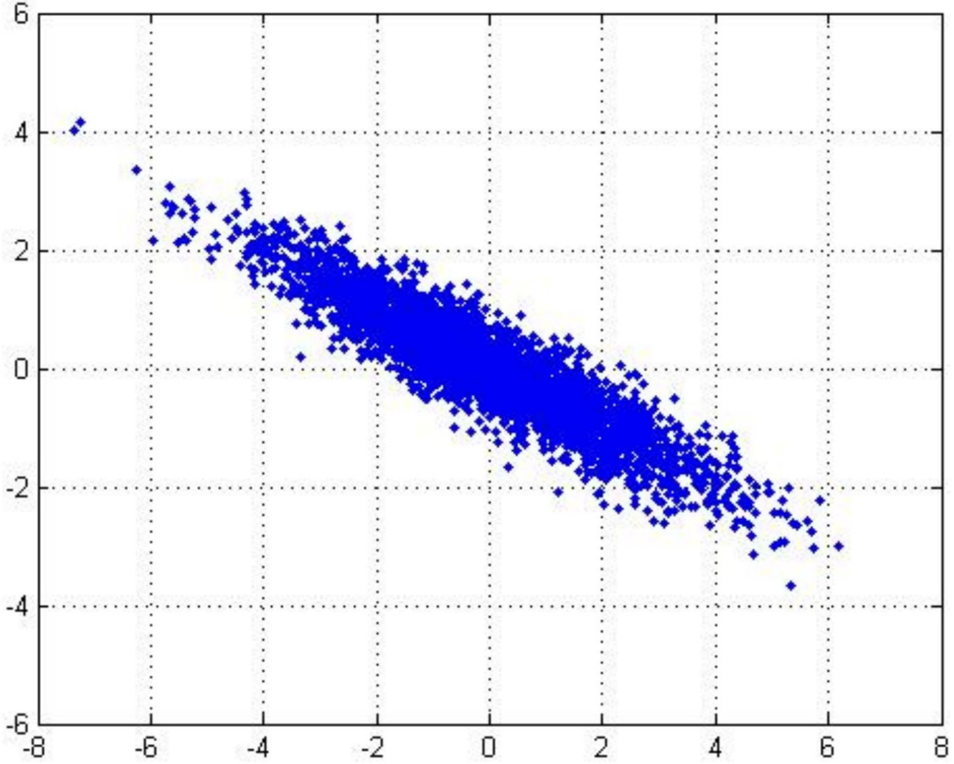
Idea:

- Given data points in a N-dimensional space, project them into a lower dimensional space while preserving as much information as possible.
  - Find best planar approximation of 3D data
  - Find best 12-D approximation of 10,000-D data

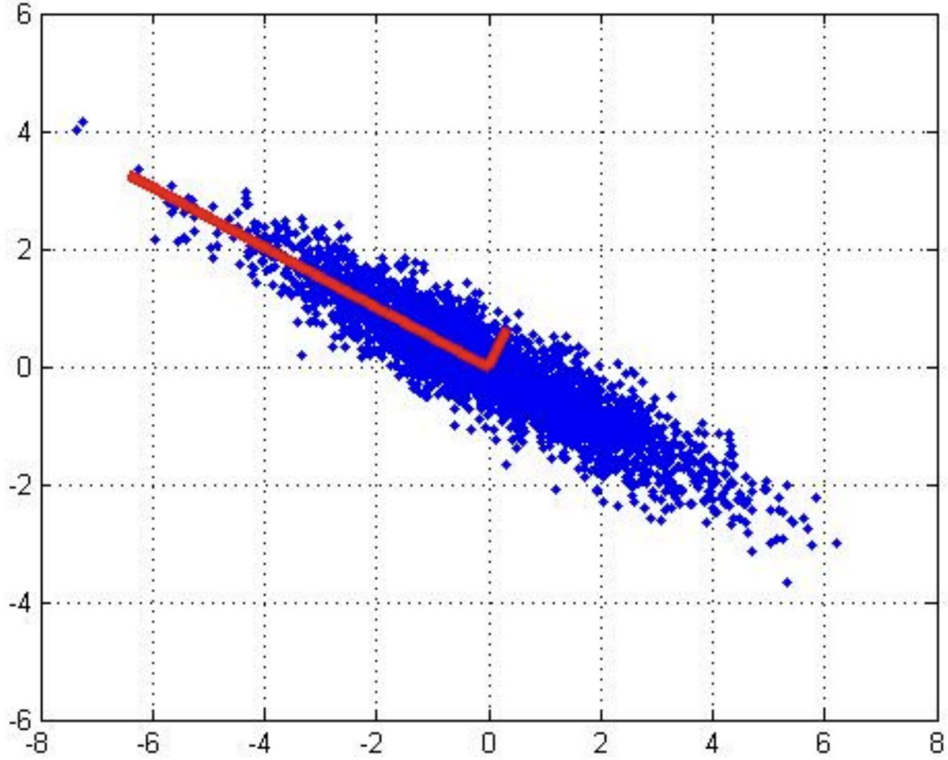
- In particular, choose **linear** projection that
  minimizes *squared error*
  in reconstructing the original data.

Columbia University

Example:



Example:



Algorithm: sequential

Given the **centered** data $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, compute the principal vectors:

$$\mathbf{w}_1 = \arg\max_{\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} \{(\mathbf{w}^T \mathbf{x}_i)^2\}$$ **1st PCA vector**

To find $\mathbf{w}_1$, maximize the variance of projection of $\mathbf{x}$

COLUMBIA UNIVERSITY

Algorithm: sequential

Given the **centered** data $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, compute the principal vectors:

$$\mathbf{w}_1 = \arg\max_{\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} \{(\mathbf{w}^T \mathbf{x}_i)^2\}$$

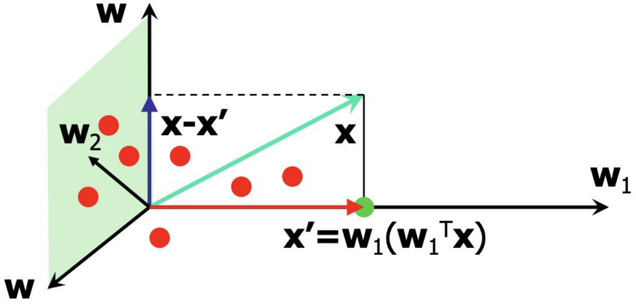
1st PCA vector

To find $\mathbf{w_1}$, maximize the variance of projection of $\mathbf{x}$

$$\mathbf{w}_2 = \arg\max_{\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} \{[\mathbf{w}^T (\mathbf{x}_i - \underbrace{\mathbf{w}_1 \mathbf{w}_1^T \mathbf{x}_i}_{\mathbf{x}' \text{ projection onto w\_1}})]^2\}$$

2nd PCA vector

To find $\mathbf{w_2}$, we maximize the **variance** of the projection in the **residual** subspace



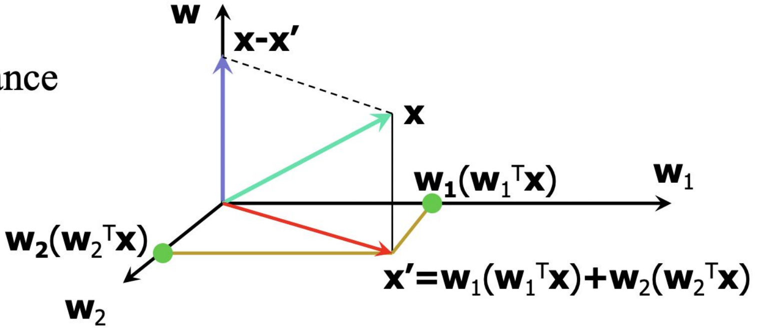
Algorithm: sequential

Given $\mathbf{w_1},\ldots,\mathbf{w_{k-1}}$, we calculate $\mathbf{w_k}$ principal vector as before:

Maximize the variance of projection of $\mathbf{x}$

$$\underbrace{\mathbf{w}_k}_{k^{th}\text{ PCA vector}} = \arg\max_{\|\mathbf{w}\|=1} \frac{1}{m}\sum_{i=1}^{m}\{[\mathbf{w}^T(\mathbf{x}_i - \underbrace{\sum_{j=1}^{k-1}\mathbf{w}_j\mathbf{w}_j^T\mathbf{x}_i}_{\mathbf{x'}\text{ projection onto previous directions}})]^2\}$$

We maximize the variance of the projection in the residual subspace



COLUMBIA UNIVERSITY

Algorithm: sample covariance matrix

- Given data $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, compute covariance matrix Σ

$$\boxed{\Sigma = \frac{1}{m}\sum_{i=1}^{m}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})^T}$$

where

$$\boxed{\bar{\mathbf{x}} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i}$$

- **PCA** basis vectors = the eigenvectors of Σ

- Larger eigenvalue $\Rightarrow$ more important eigenvectors

Columbia University

Algorithm: sample covariance matrix

PCA algorithm($\mathbf{X}$, k): top k eigenvalues/eigenvectors

% $\underline{\mathbf{X}}$ = N × m data matrix, N is number of features
% … each data point $\mathbf{x}_i$ = column vector, i=1..m

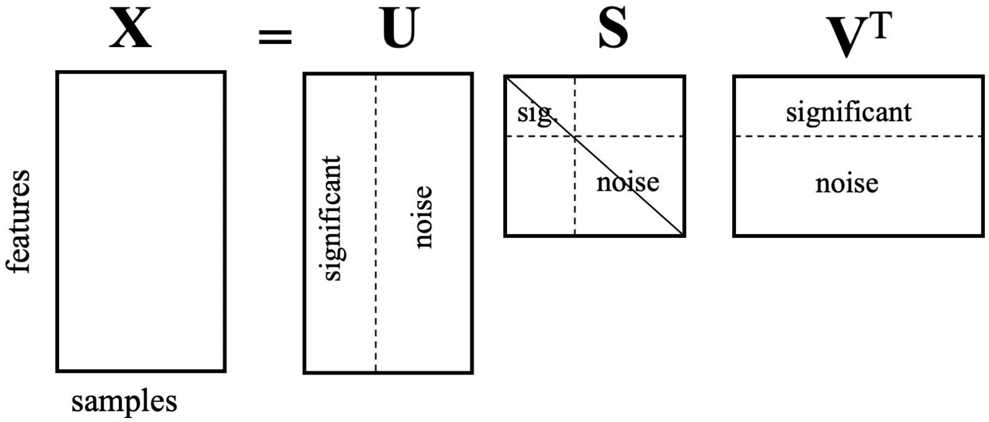
- $\underline{\mathbf{x}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}_i$
- $\mathbf{X} \leftarrow$ subtract mean $\underline{\mathbf{x}}$ from each column vector $\mathbf{x}_i$ in $\underline{\mathbf{X}}$
- $\Sigma \leftarrow \mathbf{X}\mathbf{X}^T$ … covariance matrix of $\mathbf{X}$
- $\{ \lambda_i, \mathbf{u}_i \}_{i=1..N}$ = eigenvectors/eigenvalues of Σ
  where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N$
- Return $\{ \lambda_i, \mathbf{u}_i \}_{i=1..k}$
  % top k PCA components

COLUMBIA UNIVERSITY

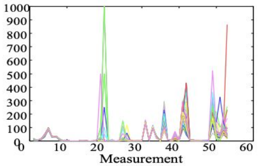
Singular value decomposition (SVD)

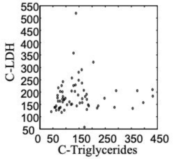
Singular Value Decomposition of the **centered** data matrix **X**.

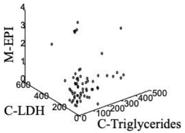
$$\mathbf{X}_{\text{features} \times \text{samples}} = \mathbf{USV}^T$$



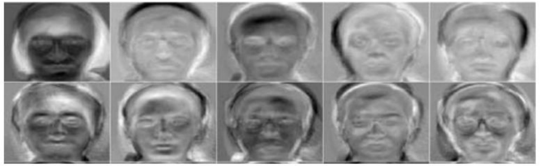
1. Data visualization (blood example)







2. Noise reduction (eigenfaces)



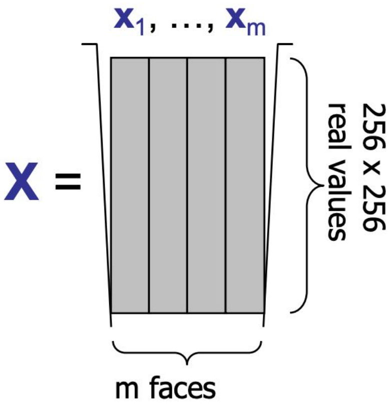
3. Data compression (image example)

Columbia University

Application: eigenface

- Example data set: Images of faces
  - Eigenface approach
    [Turk & Pentland], [Sirovich & Kirby]
- Each face $\mathbf{x}$ is …
  - 256 × 256 values (luminance at location)
  - $\mathbf{x}$ in $\Re^{256\times256}$ (view as 64K dim vector)
- Form $\mathbf{X} = [\, \mathbf{x}_1, \ldots, \mathbf{x}_m \,]$ **centered** data matrix
- Compute $\Sigma = \mathbf{XX}^T$
- Problem: Σ is 64K × 64K … HUGE!!!

  (34 GB in memory)



Columbia University

Computational complexity

- Suppose m instances, each of size N
  - Eigenfaces: $m=500$ faces, each of size $N=64K$
- Given $N \times N$ covariance matrix Σ, can compute
  - all N eigenvectors/eigenvalues in $O(N^3)$
  - first k eigenvectors/eigenvalues in $O(k\ N^2)$

- But if $N=64K$, EXPENSIVE!

Computational complexity: how about...

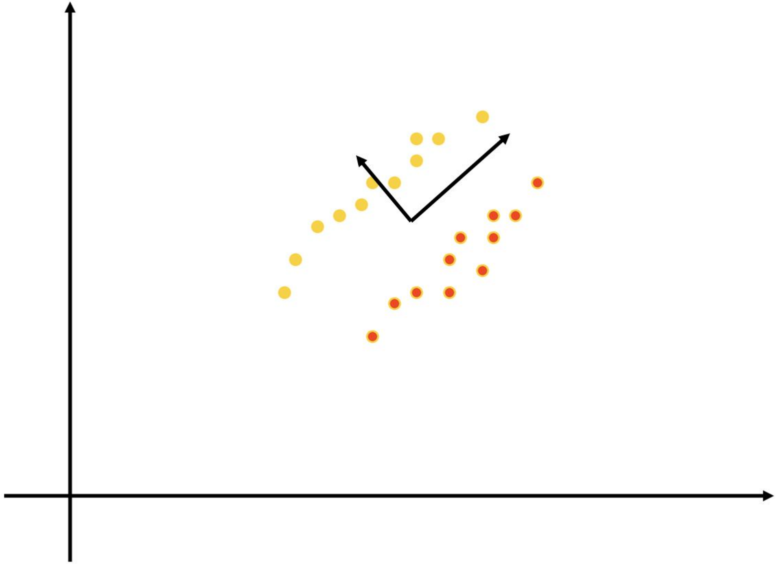
- Note that $m << 64K$
- Use $\mathbf{L}=\mathbf{X}^T\mathbf{X}$ instead of $\Sigma=\mathbf{X}\mathbf{X}^T$
- If $\mathbf{v}$ is eigenvector of $\mathbf{L}$
  then $\mathbf{Xv}$ is eigenvector of Σ
- $O(Nm^2) + O(km^2)$
- 64M vs 42,000M operations

(Dis)advantages

- Required carefully handled data. Sensitive to data preprocessing quality.
- Completely knowledge-free!

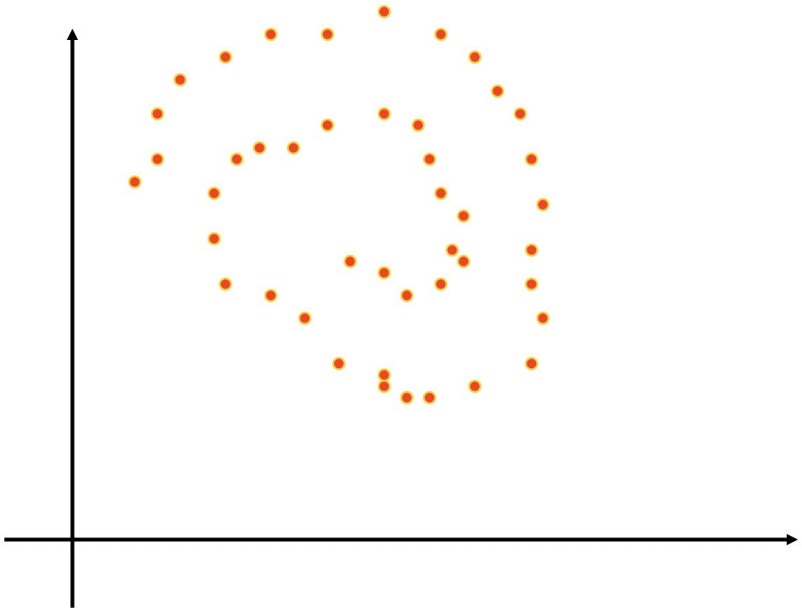
COLUMBIA UNIVERSITY

(Dis)advantages



PCA doesn't know labels!

(Dis)advantages



PCA cannot capture NON-LINEAR structure!

COLUMBIA UNIVERSITY

Takeaways

- PCA:
  - Finds orthonormal basis for data
  - Sorts dimensions in order of "importance"
  - Usually discard unimportant dimensions
- Applications:
  - Visualization
  - Data compression / compact representation
  - Remove noise to improve classification (hopefully)
- Disadvantages:
  - Doesn't know class labels
  - Can only capture linear variations
- One of many ways to reduce dimensionality!

Columbia University

References

- Trevor Hastie, Robert Tibshirani, Jerome Friedman: The Elements of Statistical Learning: Data Mining, Inference and Prediction, Chapter 14.5
- Ziv Bar-Joseph, Tom Mitchell, Pradeep Ravikumar and Aarti Singh: CMU 10-701