Machine Learning (CE 40717)

Fall 2024

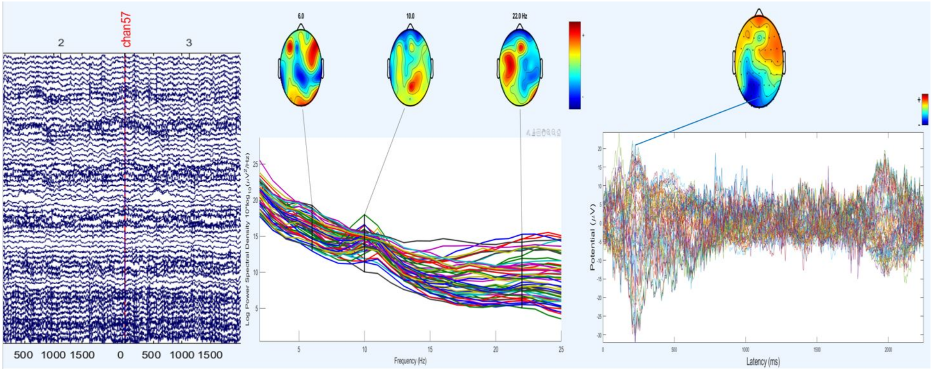
Ali Sharifi-Zarchi

CE Department
Sharif University of Technology

October 11, 2024

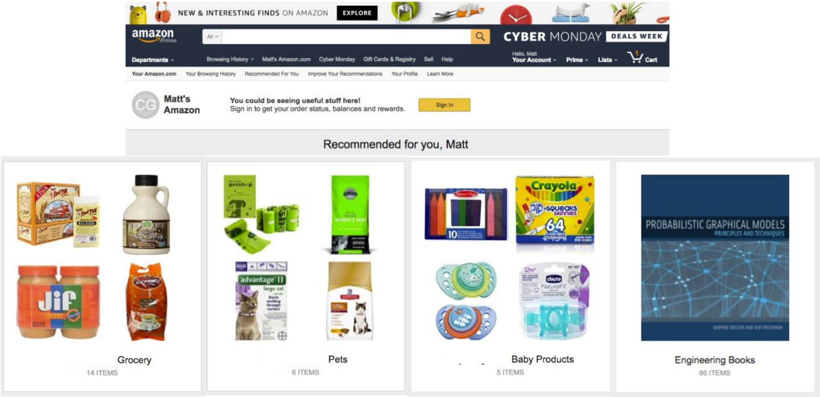
High Dimensional Data

- High-Dimensions = Lots of Features
- EEG Signals of Brain 56 Channels $*$ 3000 Time Points For Each Trial



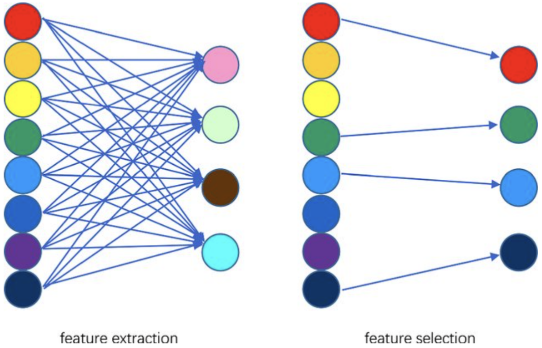
High Dimensional Data

- High-Dimensions = Lots of Features
- Customer Purchase Data



Dimensionality Reduction

- **Feature Selection**
  - Select a subset of a given feature set
- **Feature Extraction**
  - A linear or non-linear transform on the original feature space



Dimensionality Reduction

- Maximize the retention of **important information** while reducing the dimensionality
- What is information?

Dimensionality Reduction

- Maximize the retention of **important information** while reducing the dimensionality
- **Information:** Variance of whole data

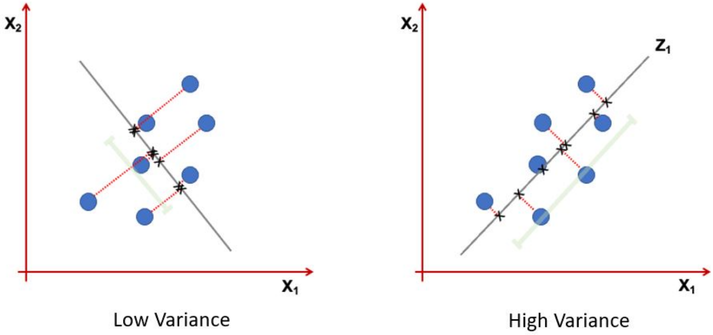


Figure 1: Figure reference

Dimensionality Reduction

- Maximize the retention of **important information** while reducing the dimensionality
- **Information:** Local relationships

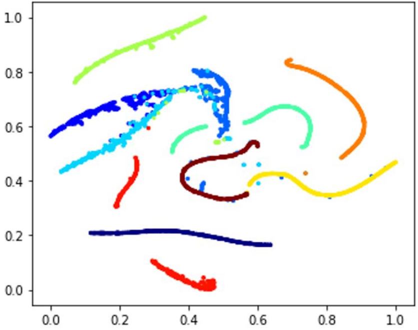
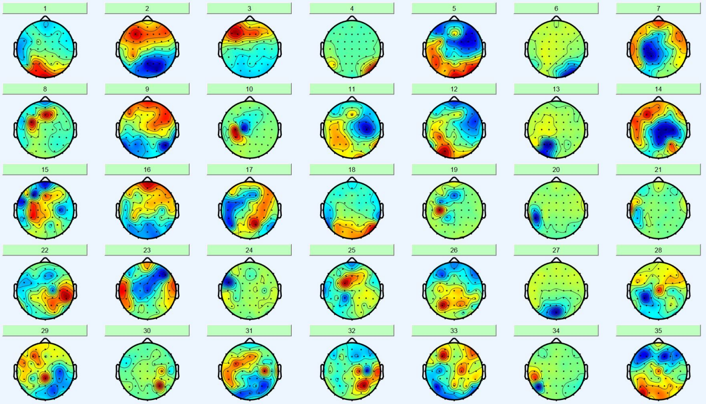


Figure 2: Figure reference

Dimensionality Reduction

- Maximize the retention of **important information** while reducing the dimensionality
- **Information:** Statistical independence



Dimensionality Reduction Benefits

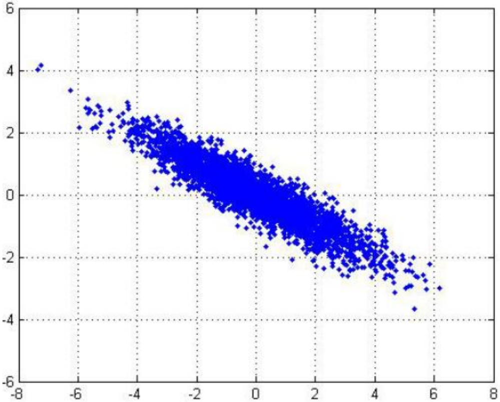
- **Visualization**
  - Project high dimensional data into 2D or 3D
- **More efficient use of resources**
  - Time, Memory, CPU
- **Pre-process**
  - Improve accuracy by reducing features
  - As a Preprocessing step to reduce dimensions for supervised learning tasks
  - Helps avoiding overfitting
- **Removing Noise**

Idea

- Given data points in a d-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 104-D data
- In particular, choose projection that minimizes squared error in reconstructing the original data

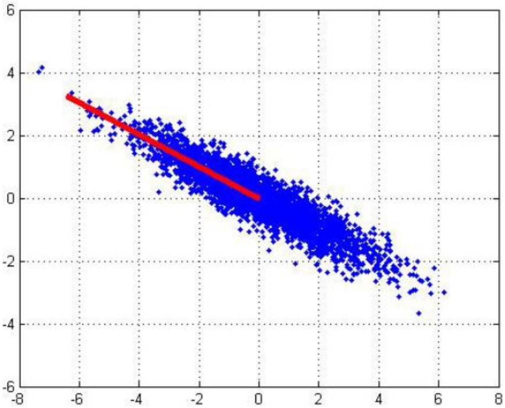
Idea

- 2D Gaussian dataset



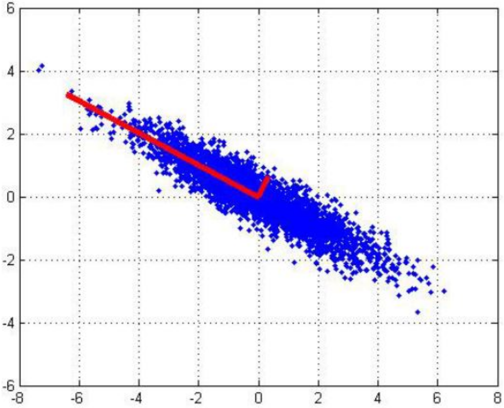
Idea

- 2D Gaussian dataset
- First PCA axis



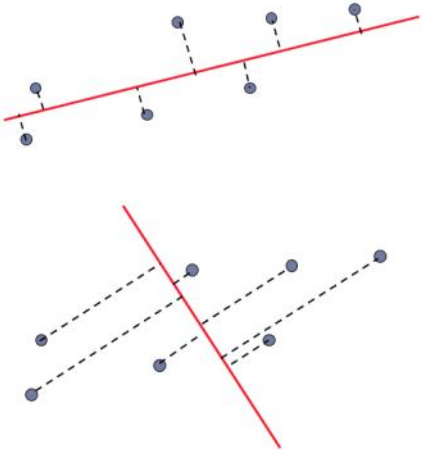
Idea

- 2D Gaussian dataset
- First and second PCA axes



Random vs Principal Projection

- Random direction vs. principal component



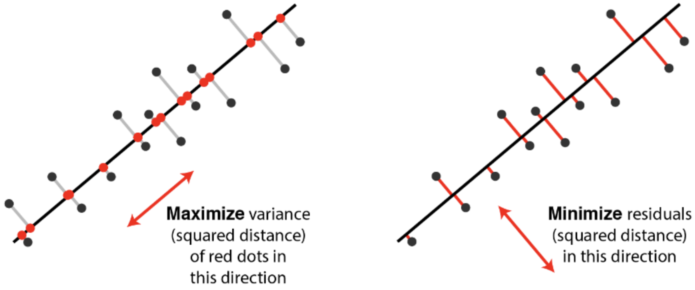
Definition

- **Goal**: reducing the dimensionality of the data while preserving important aspects of the data
- Suppose $\mathbf{X} = \begin{pmatrix} \mathbf{X}_1^T \\ \vdots \\ \mathbf{X}_n^T \end{pmatrix}_{n\times d}$
- $\mathbf{X}_{n\times d} \xrightarrow{\text{PCA}} \tilde{\mathbf{X}}_{n\times k}$ with $k \leq d$
- **assumption**: $\mu_x = \frac{1}{M}\sum_{i=1}^{M} X_i = 0_{d\times 1}$

Interpretations

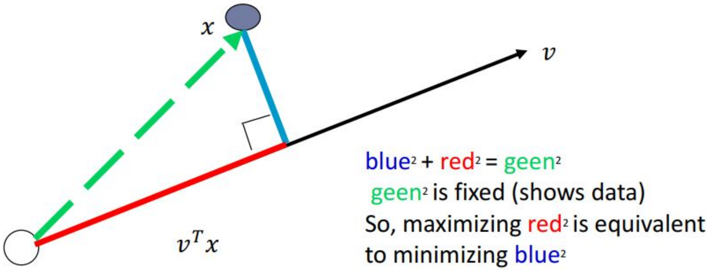
Orthogonal projection of the data onto a **lower-dimensional** linear space that:

- Interpretation 1. Maximizes variance of projected data
- Interpretation 2. Minimizes the sum of squared distances to the line



Equivalence of the interpretations

- Minimizing the sum of square distances to the line is **equivalent** to maximizing the sum of squares of the projections on that line.



Equivalence of the interpretations

Principal Components (PCs): A set of **orthonormal** vectors ($v = [v_1, v_2, ..., v_k]$) generated by PCA, which fulfill both of the interpretations.

Interpretation 1. Maximizes variance of projected data

- Projection of data points on v_1

$$\Pi = \Pi_{v_1}\{X_1, \ldots, X_n\} = \{v_1^T X_1, \ldots, v_1^T X_n\}$$

- Note that $\mathrm{Var}(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2$

$$\mathrm{Var}(\Pi) = \frac{1}{N}\left(\sum_{i=1}^{N}(v_1^T X_i)^2\right) - \left(\frac{1}{N}\sum_{i=1}^{N} v_1^T X_i\right)^2$$

Equivalence of the interpretations

Interpretation 1. Maximizes variance of projected data

- Based on the assumption, $\frac{1}{N}\sum_{i=1}^{N} X_i = 0$

$$v^T\left(\frac{1}{N}\sum_{i=1}^{N} X_i\right) = 0 \quad \longrightarrow \quad \frac{1}{N}\sum_{i=1}^{N} v_1^T X_i = 0$$

- So,

$$\mathrm{Var}(\Pi) = \frac{1}{N}\sum_{i=1}^{N} (v_1^T X_i)^2$$

- To find v_1 that maximizes the variance

$$\begin{aligned} _{v_1} \quad & \frac{1}{N}\sum_{i=1}^{N} (v_1^T X_i)^2 \\ \text{s.t.} \quad & v_1^T v_1 = 1 \end{aligned}$$

Equivalence of the interpretations

Interpretation 2. Minimizes the sum of squared distances to the line

- Squared distance of one point to the line

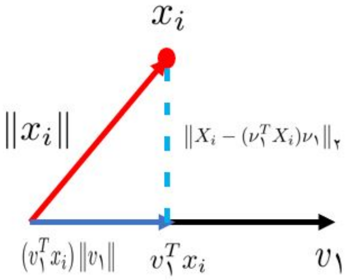
$$\left\| X_i - (v_1^T X_i) v_1 \right\|_2^2 \quad \text{which } v_1^T X_i \text{ is scalar.}$$

- Sum of squared distances

$$L_1 = \frac{1}{N} \sum_{i=1}^{N} \left\| X_i - (v_1^T X_i) v_1 \right\|_2^2$$

- By Pythagorean theorem

$$L_1 = \frac{1}{N} \sum_{i=1}^{N} (\|X_i\|^2 - (v_1^T X_i)^2 \|v_1\|_2^2) = \underbrace{\frac{1}{N} \sum_{i=1}^{N} (\|X_i\|^2)}_{\text{constant}} - \frac{1}{N} \sum_{i=1}^{N} ((v_1^T X_i)^2 \|v_1\|_2^2)$$



Equivalence of the interpretations

Interpretation 2. Minimizes the sum of squared distances to the line

- Removing constant to minimize
- Based on orthonormality, $||V_1||_2^2 = 1$
- To find v_1 that minimizes the sum of squared distances

$$\begin{aligned} & {}_{v_1} \quad -\frac{1}{N}\sum_{i=1}^{N}(v_1^T X_i)^2 \\ \text{s.t.} \quad & v_1^T v_1 = 1 \end{aligned} \quad \equiv \quad \begin{aligned} & {}_{v_1} \quad \frac{1}{N}\sum_{i=1}^{N}(v_1^T X_i)^2 \\ \text{s.t.} \quad & v_1^T v_1 = 1 \end{aligned}$$

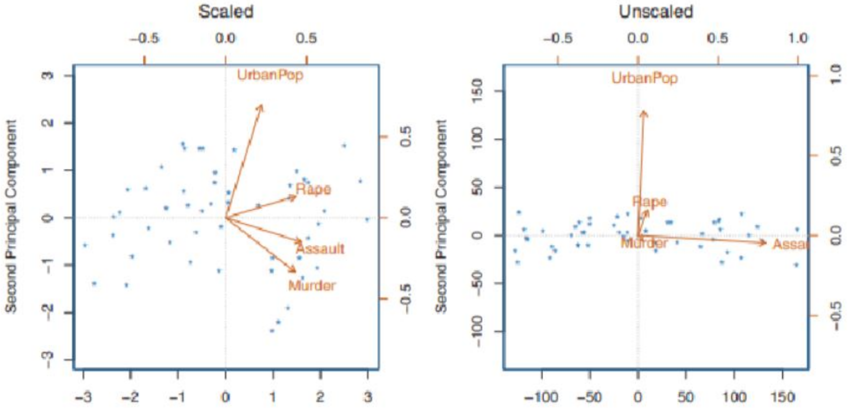
- So, the two interpretations are equivalent.

Pre-processing

- **Center the data**
  - **Zero**ing out the **mean** of each feature
- **Scaling to normalize each feature to have variance 1 (An arbitrary step)**
  - The final result may be wrong!
  - It helps when unit of measurements of features are different and some features may be ignored without normalization

Pre-processing

- Scaling to normalize each feature may affect the final result!!



Algorithms

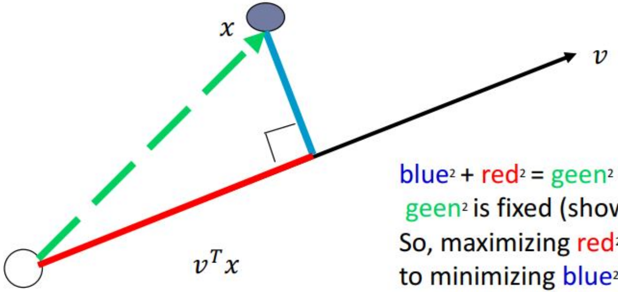
- Algorithm 1: sequential
- Algorithm 2: sample covariance matrix

Sequential Algorithm

- First view
  - Find directions with the maximum variations

$$\max_{v_1} \frac{1}{N}\sum_{n=1}^{N}(v_1^T x_n)^2 = \frac{1}{N}\sum_{n=1}^{N} v_1^T (x_n x_n^T) v_1 = v_1^T \left(\frac{1}{N}\sum_{n=1}^{N}(x_n x_n^T)\right) v_1 = v_1^T S v_1$$

$$\text{s.t. } v_1^T v_1 = 1$$



blue² + red² = geen²
geen² is fixed (shows data)
So, maximizing red² is equivalent to minimizing blue²

Sequential Algorithm

- To find v_2, we maximize the variance of the projection in the residual subspace

$$v_2 =_{v_2} \left(\frac{1}{N} \sum_{i=1}^{N} \left(x_i - v_1^T x_i \right)^2 \right)$$

$$\text{s.t. } v_2^T v_2 = 1$$

- To find v_k, we maximize the variance of the projection in the residual subspace

$$v_k =_{v_k} \left(\frac{1}{N} \sum_{i=1}^{N} \left(x_i - \sum_{j=1}^{k-1} W_j^T x_i \right)^2 \right)$$

$$\text{s.t. } v_k^T v_k = 1$$

Sequential Algorithm

- As we have $Sv_j = \lambda_j v_j$,

$$\Rightarrow \text{var}(v_j^T x) = v_j^T x x^T v_j = v_j^T S v_j = \lambda_j v_j^T v_j = \lambda_j$$

.

- The variance along an eigenvector v_j equals the eigenvalue λ_j.

Sequential Algorithm

- Eigenvalues: $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \ldots$
  - The first PC v_1 is the the eigenvector of the sample covariance matrix S associated with the largest eigenvalue
  - The 2nd PC v_2 is the the eigenvector of the sample covariance matrix S associated with the second largest eigenvalue
  - And so on ...
- To reduce the dimension of the data to k, we select eigenvectors with the top k eigenvalues

Sample Covariance Matrix

- Given data $x_1, \ldots, x_n$, compute covariance matrix Σ

$$\Sigma = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T \text{ where } \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

- PCA basis vectors = the eigenvectors of Σ
- Larger eigenvalue $\rightarrow$ more important eigenvectors

Sample Covariance Matrix

Sample covariance matrix

- It is symmetric $\Rightarrow$ Eigen-vectors are **orthogonal**
- It is symmetric $\Rightarrow$ Eigen-values are **real**
- It is positive semidefinite $\Rightarrow$ Eigen-values are **non-negative**

Sample Covariance Matrix

Principal component analysis

- Principal components are **orthonormal**
- Variances along each principal component are **real**
- Variances along each principal component are **non-negative**

Sample Covariance Matrix

Principal component analysis and sample covariance matrix

- Principal components are **eigen-vectors**
- Variance of each principal component is the **eigen-value** of the corresponding eigen-vector

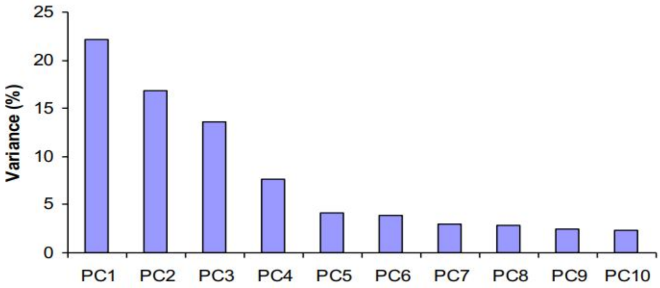
Sample Covariance Matrix

Algorithm 1 Sample Covariance Matrix

1: **Input:** $X \in \mathbb{R}^{N \times d}$ (data matrix with N data points and d dimensions)
2: Compute the mean of each feature: $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$
3: Subtract the mean from each data point (center the data): $\tilde{X} \leftarrow X - 1_N \bar{x}^T$
4: Compute the covariance matrix: $S = \frac{1}{N} \tilde{X}^T \tilde{X}$
5: Compute the eigenvalues and eigenvectors of S: $[\lambda_1, \lambda_2, \ldots, \lambda_d], [v_1, v_2, \ldots, v_d] = \text{eig}(S)$
6: Select the top K eigenvectors corresponding to the largest eigenvalues: $A \leftarrow [v_1, v_2, \ldots, v_K]$
7: Transform the data into the new subspace: $X' \leftarrow X \cdot A$
8: **Output:** $X' \in \mathbb{R}^{N \times K}$ (transformed data with reduced dimensions)

How many PCs?

- For n original dimensions, sample covariance matrix is $n * n$, and has up to n eigenvectors. So n PCs
- Can ignore the components of lesser significance



- You do lose some information, but if the eigenvalues are small, you don't lose much

How many PCs?

- Select the desired variance ratio and select the PCs

$$\min_k \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{d} \lambda_i} >= 0.9$$

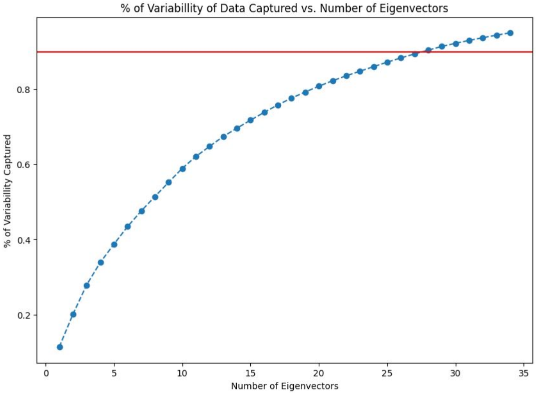


Image Compression

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

Image Compression

- 144D $\Rightarrow$ 60D

Image Compression

- 144D ⇒ 16D

Image Compression

- 16 most important eigenvectors

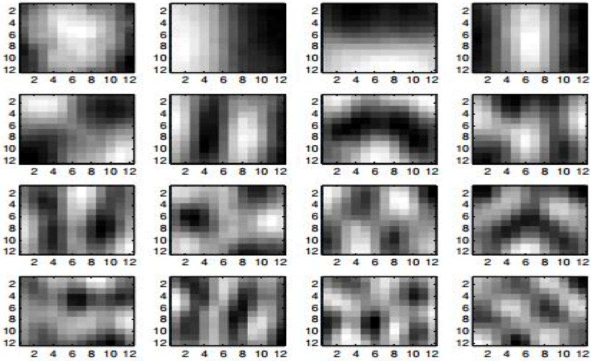


Image Compression

- $144D \Rightarrow 3D$

Image Compression

- 3 most important eigenvectors

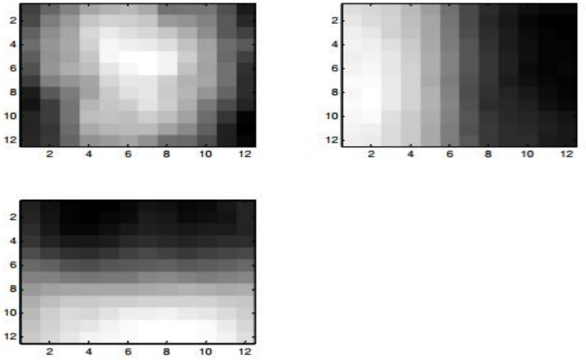
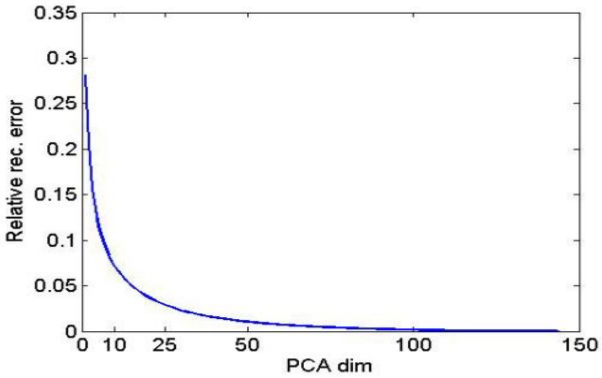


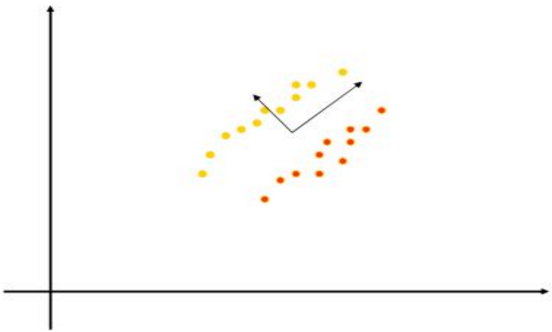
Image Compression

- L2 error and PCA dim



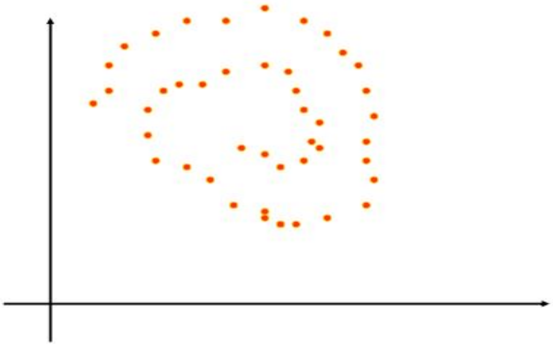
Class Labels

- PCA doesn't know about class labels!



Non-Linear

- PCA cannot capture Non-Linear structure!



Conclusion

- PCA
  - finds orthonormal basis for data
  - Sorts dimensions in order of “importance”
  - Discard low significance dimensions
- Applications
  - Get compact description
  - Remove noise
  - Improve classification (hopefully)
  - More efficient use of resources
  - Statistical
- Not magic
  - Doesn’t know class labels
  - Can only capture linear variations
- One of many tricks to reduce dimensionality!

References

- Advanced Introduction to Machine Learning, CMU-10715, Barnabás Póczos
- 10-701 Introduction to Machine Learning, CMU, Matt Gormley
- 10-301/10-601 Introduction to Machine Learning, CMU, Matt Gormley
- CE-477: Machine Learning - CS-828: Theory of Machine Learning, Sharif University of Technology, Fatemeh Seyyedsalehi