Lecture 10 Unsupervised Learning

Unsupervised Learning

1. Introduction
2. Cluster Analysis
4. Self-Organizing Maps
5. Principal Components, SVD
6. Non-negative Matrix Factorization
7. Independent Component Analysis
8. Multidimensional Scaling
9. Nonlinear Dimension Reduction
10. The Google PageRank Algorithm

Motivation

- Supervised Learning: Large scale of labeled data
- Unsupervised Learning:
  - Latent structures in data: linear or nonlinear
  - Dimension reduction
  - Relations in data
- Semi-supervised learning
- Reinforcement learning

1. Introduction

- *Unsupervised learning* or "learning without a teacher."
- A set of N observations

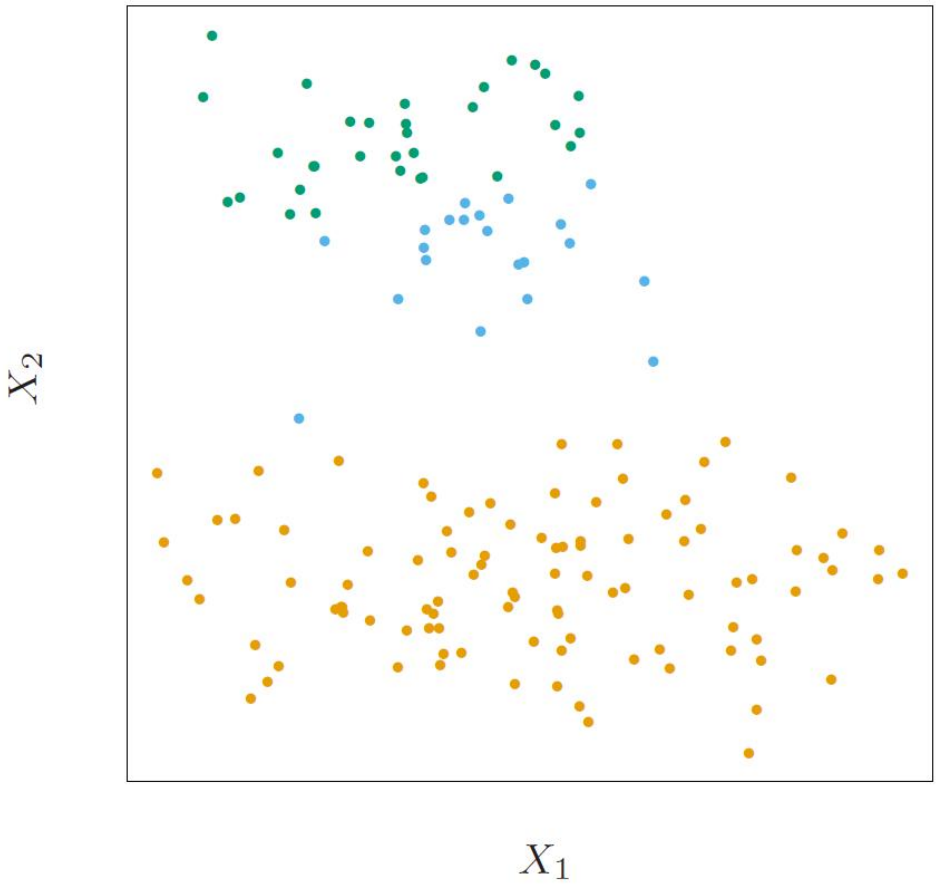
$$\{x_1, x_2, \cdots, x_N\}, X \in R^p \sim \mathrm{P}(X)$$

- In low-dimensional problems (say p ≤ 3), there are a variety of effective nonparametric methods for directly estimating the density $P(X)$ itself at all X-values
- In high-dimensional problems, one must settle for estimating rather crude global models, such as Gaussian mixtures or various simple descriptive statistics that characterize $P(X)$.

2. Cluster Analysis

- Cluster analysis, also called data segmentation, has a variety of goals.

All relate to grouping or segmenting a collection of objects into subsets or “clusters,” such that those within each cluster are more closely related to one another than objects assigned to different clusters.



2. Cluster Analysis

- Central to all of the goals of cluster analysis is the notion of the degree of similarity(相似值) (or dissimilarity（相异值）) between the individual objects being clustered.
- K-means clustering starts with guesses for three cluster centers.
  - for each data point, the closest cluster center (in Euclidean distance) is identified;
  - each cluster center is replaced by the coordinate-wise average of all data points that are closest to it.

Object Dissimilarity

- Dissimilarity $D(x_i, x_{i'})$ between two objects or observations

$$D(x_i, x_{i'}) = \sum_{j=1}^{p} w_j d(x_{ij}, x_{i'j}); \quad \sum_{j=1}^{p} w_j = 1$$

- The average object dissimilarity measure over all pairs of observations in the data set

$$\bar{D} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} D(x_i, x_{i'}) = \sum_{j=1}^{p} w_j \bar{d}_j,$$

$$\bar{d}_j = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_j(x_{ij}, x_{i'j})$$

Combinatorial Algorithms

- Assign each observation to a group or cluster without regard to a probability model describing the data.

$$\{x_i\}_{i=1}^{N} \qquad x_i \Rightarrow C(i) = k, k = 1, \ldots, K$$

- The Loss function

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')=k} d(x_i, x_{i'})$$

- assigned to the same cluster tend to be close to one another.

Combinatorial Algorithms

- The *total* point scatter

$$\mathrm{T} = \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N} d(x_i, x_{i'}) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\left(\sum_{C(i')=k} d(x_i, x_{i'}) + \sum_{C(i')\neq k} d(x_i, x_{i'})\right)$$

$$= W(C) + B(C)$$

- Between-Cluster Scatter:

$$\mathrm{B}(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')\neq k} d(x_i, x_{i'})$$

- Minimizing W(C) is equivalent to *maximizing* B(C).

K-means

- The within-point scatter can be written as

$$W(C)=\frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')=k}d(x_i,x_{i'})$$

$$=\frac{1}{2}\sum_{k=1}^{K}\sum_{C(i)=k}\sum_{C(i')=k}\left\|x_i-x_{i'}\right\|^2$$

$$=\sum_{k=1}^{K}N_k\sum_{C(i)=k}\left\|x_i-\bar{x}_k\right\|^2$$

where N_k is the sample number of k-th class

Algorithm 14.1 *K-means Clustering.*

1. For a given cluster assignment C, the total cluster variance (14.33) is minimized with respect to $\{m_1, \ldots, m_K\}$ yielding the means of the currently assigned clusters (14.32).

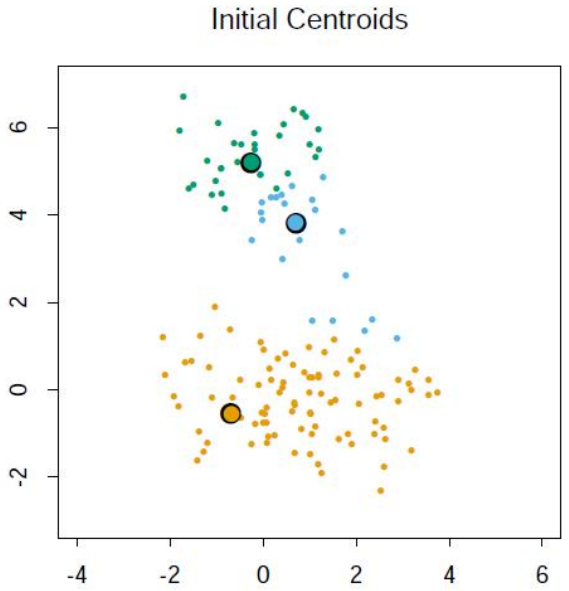
2. Given a current set of means $\{m_1, \ldots, m_K\}$, (14.33) is minimized by assigning each observation to the closest (current) cluster mean. That is,

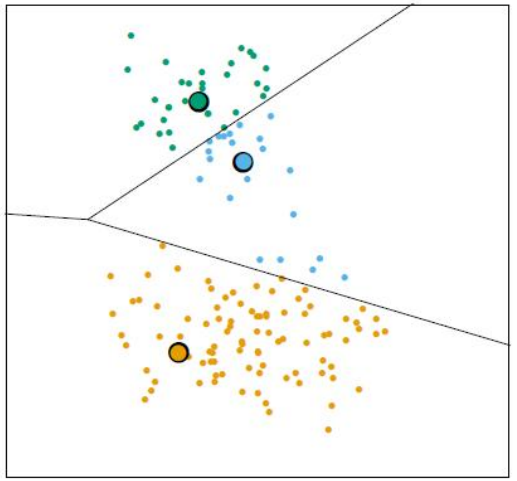
$$C(i) = \underset{1 \le k \le K}{\operatorname{argmin}} ||x_i - m_k||^2. \tag{14.34}$$

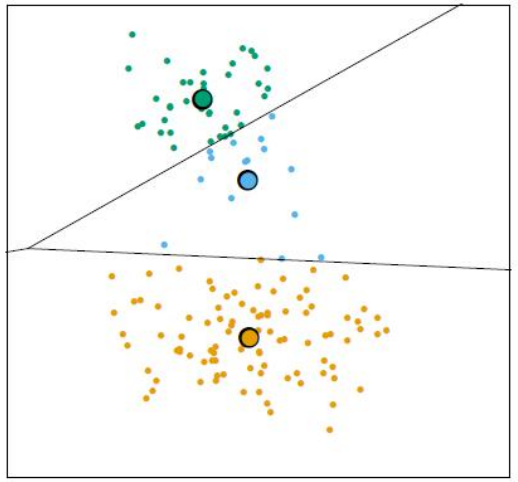
3. Steps 1 and 2 are iterated until the assignments do not change.

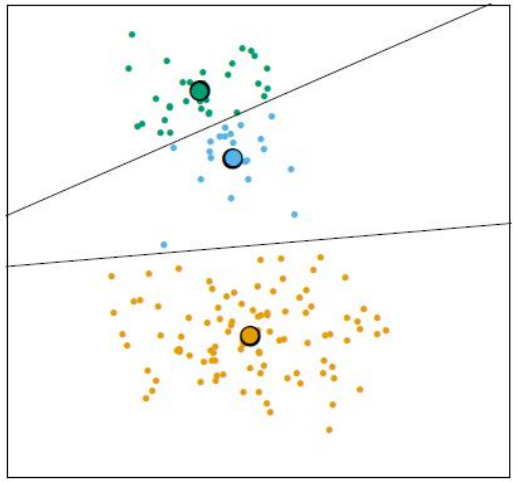
$$\bar{x}_S = \underset{m}{\operatorname{argmin}} \sum_{i \in S} ||x_i - m||^2. \tag{14.32}$$

$$\min_{C, \{m_k\}_1^K} \sum_{k=1} N_k \sum_{C(i)=k} ||x_i - m_k||^2. \tag{14.33}$$


Initial Centroids
6
4
2
0
-2
-4
-2
0
2
4
6


Initial Partition


Iteration Number 2


Iteration Number 20

Unsupervised Learning

- Introduction
- Association Rules & Cluster Analysis
- Self-Organizing Maps
- Principal Components, SVD
- Non-negative Matrix Factorization
- Independent Component Analysis
- Multidimensional Scaling
- Nonlinear Dimension Reduction
- The Google PageRank Algorithm

14.4 Self-Organizing Maps (SOMs)

Constrained version of K-means (VQ) with prototypes on a **topological map (grid)**

For K prototypes m_j in $\mathbb{R}^p$ placed on a given map

- Find closest m_j of considered x_i
- Move m_j and its neighbors m_k towards x_i

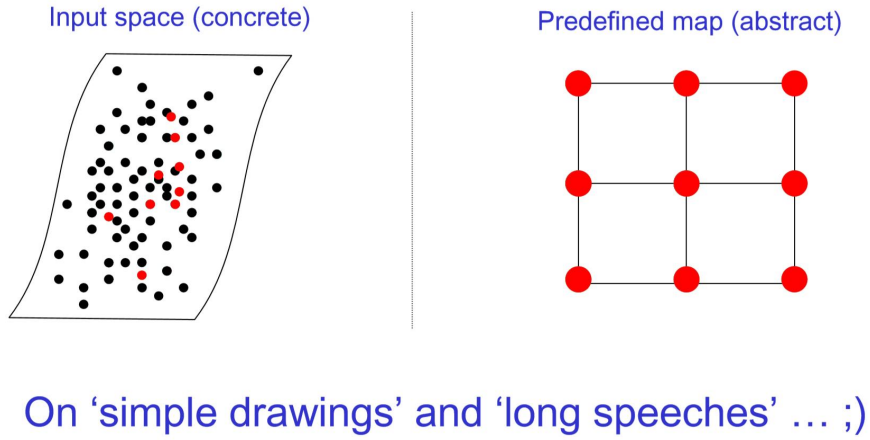
$$m_k = m_k + \alpha(x_i - m_k)$$

Running the algorithm moves the prototypes

inside the input distribution w.r.t. map constrains

l_j is integer pair of prototype m_j;

m_k *is* a set of neighbors of m_j



On ‘simple drawings’ and ‘long speeches’ … ;)

Extended SOMs

l_j is integer pair of prototype m_j; m_k *is* a set of neighbors of m_j

$h(\|l_i - l_k\|)$ is a kernel function

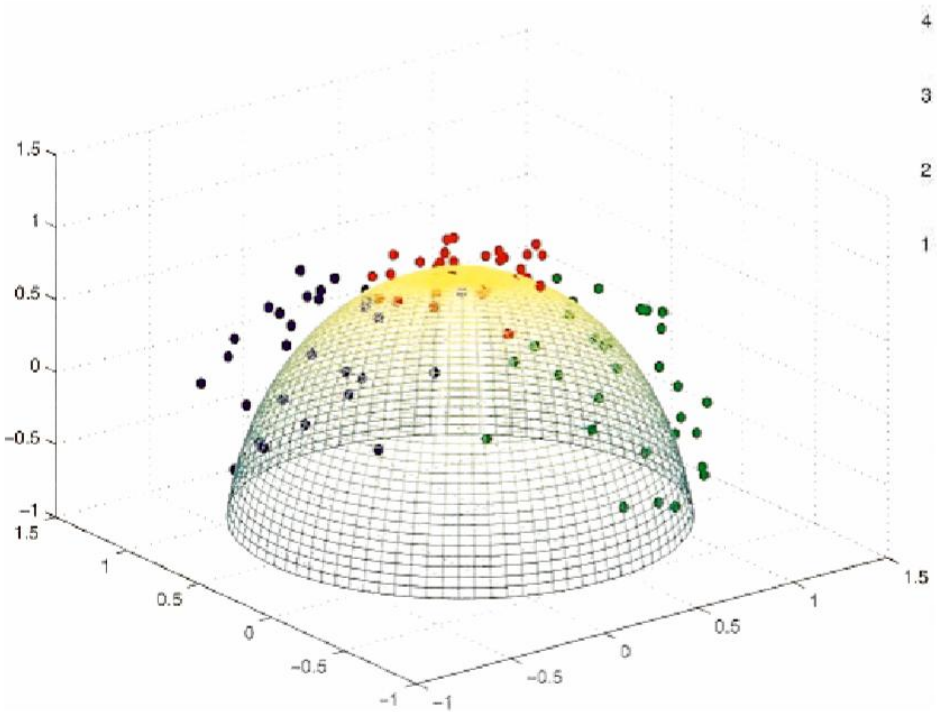
Enhanced version $m_k = m_k + \alpha h(\|l_i - l_k\|)(x_i - m_k)$

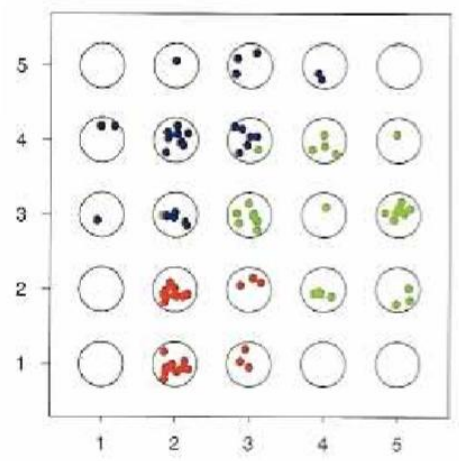
Batch one : $$m_j = \frac{\sum_k w_k x_k}{\sum_k w_k}$$

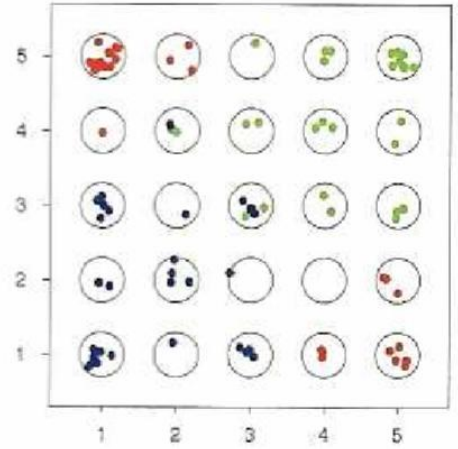
The sum is over points x_k that mapped (i.e., were closest to) neighbors m_k of prototype of m_j

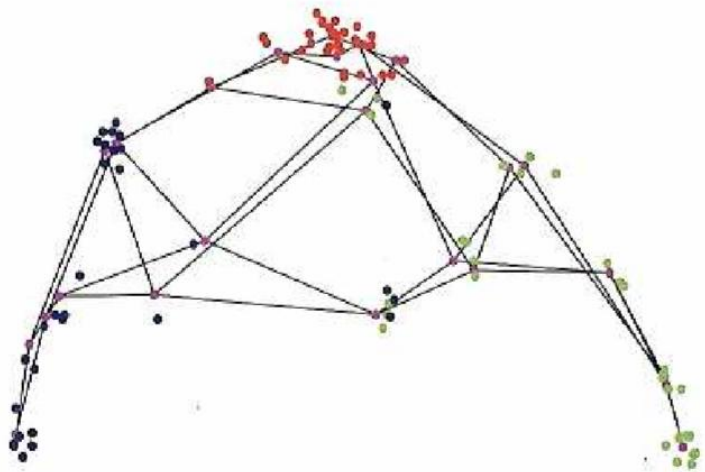
4. Self-Organizing Maps (SOMs)

- Can SOMs separate sphere data ?





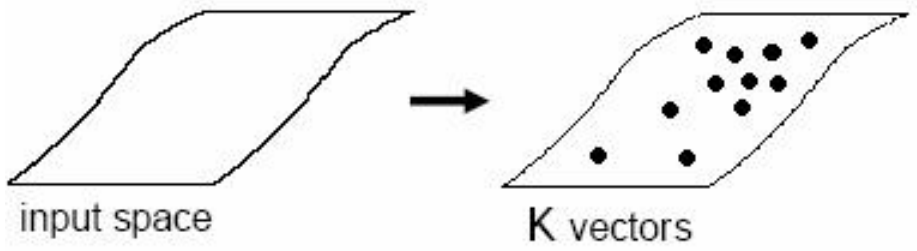




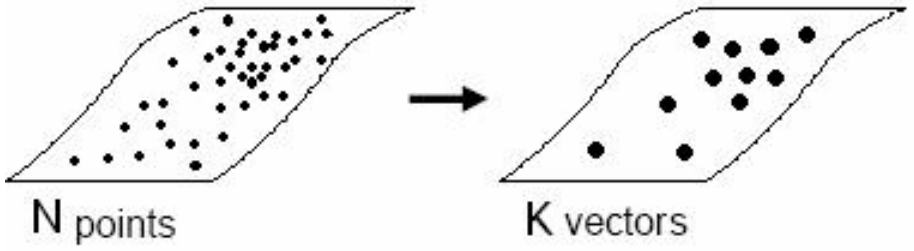
VQ aims to reduce the inputs (K≤N)

Vector Quantization (VQ) from 14.3.9 :

We aim to :

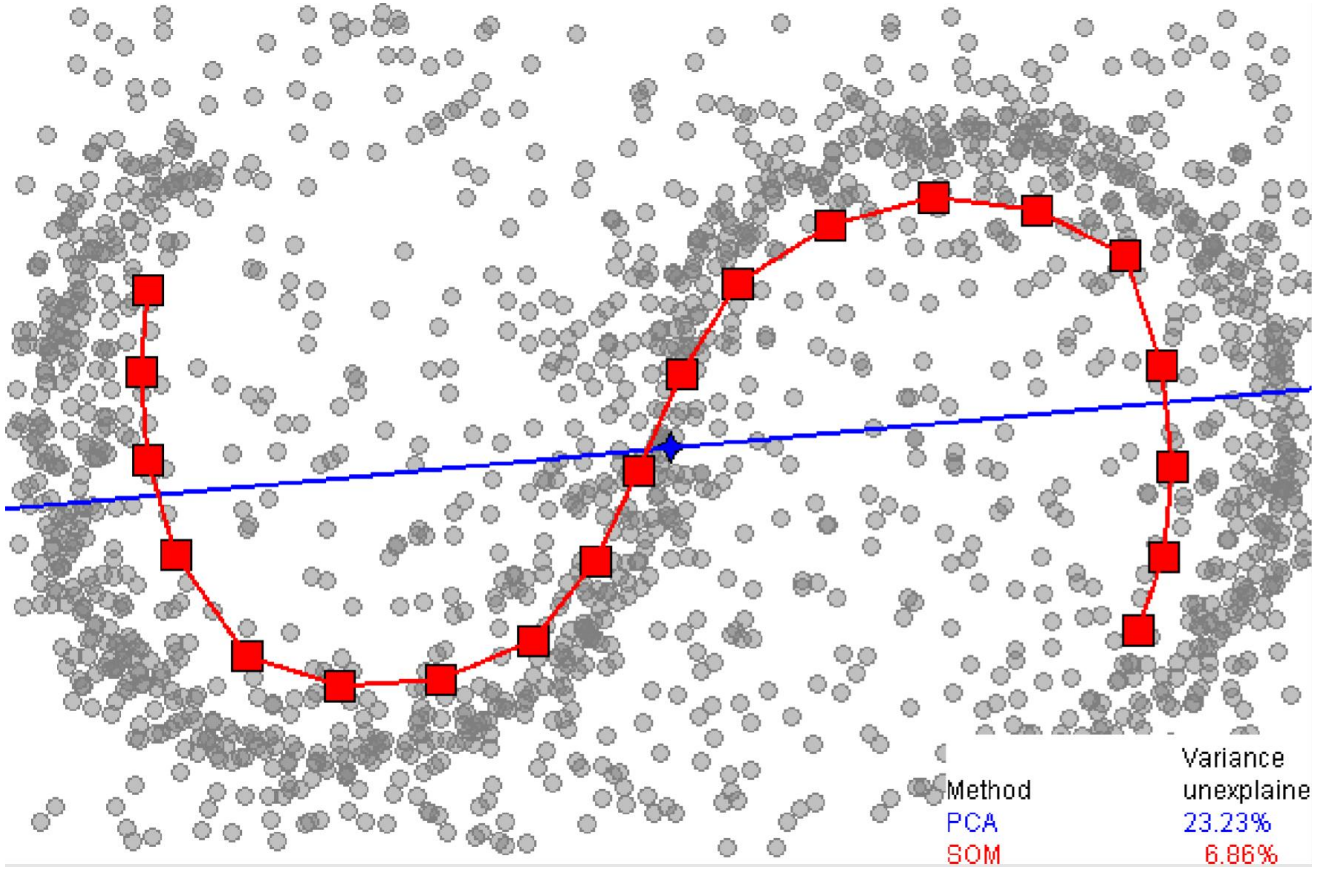


We use :



Neural VQ algorithms are an adaptative (iterated) version of K-means

4. Self-Organizing Maps (SOMs)



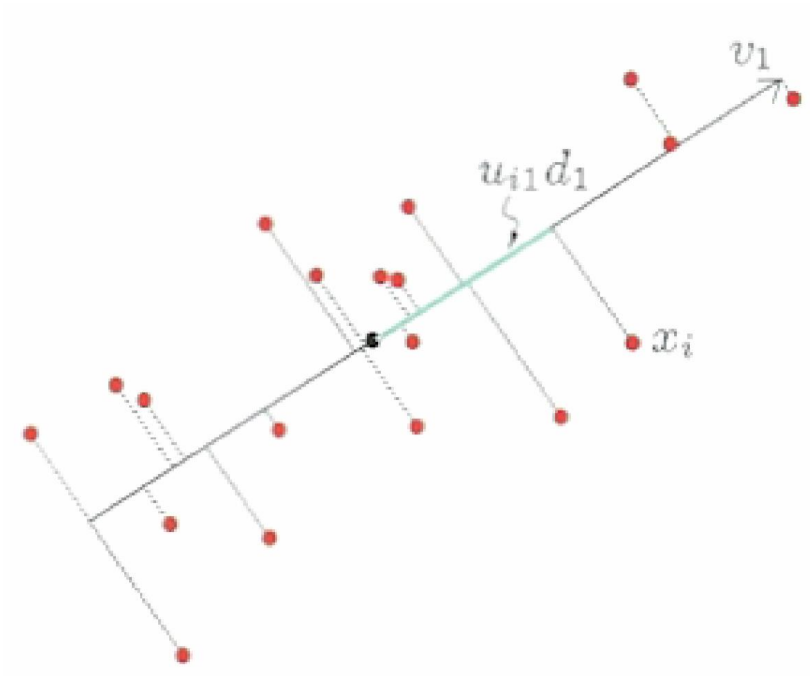
Principal Components (PCA)

Set of q directions towards which p-dimensional data are linearly projected

-> dimension reduction since $q \leq p$

Data : $x_1, x_2, \cdots, x_n$

Model : $f(\lambda) = (\mu + \lambda_i V_q)$



Model fitted by least squares

w.r.t. reconstruction error

$$\{\mu, \lambda_i, V_q\} = \text{var min}_{\mu,\lambda,V_q} \sum_{i=1}^{n} \left\| x_i - (\mu + \lambda_i V_q) \right\|^2$$

Principal Components (PCA)

Principal Components are computed from matrix operations

1. Given V_q, solve the optimal problem

$$\hat{\mu} = \bar{x}, \quad \hat{\lambda}_i = V_q^T (x_i - \bar{x}).$$

2. Substitute them into original cost function

$$V_q = \text{var min}_{V_q} \sum_{i=1}^{n} \left\| (x_i - \bar{x}) - V_q V_q^T (x_i - \bar{x}) \right\|^2$$

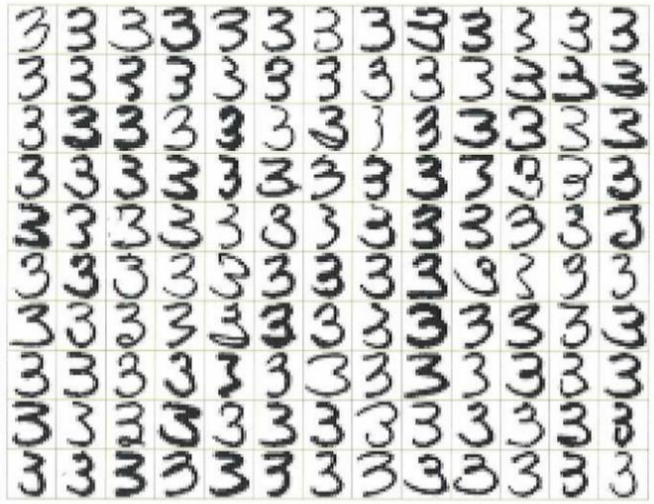
Data are usually centered, and $V_q V_q^T = H_q$ is the projection matrix. Solution V_q are the q first columns of V obtained from

$$\mathbf{X} = \mathbf{UDV}^T = \mathbf{SA}^T; \ \mathbf{U} \ is \ \text{N} \times \text{p} \ \text{orthognal matrix} \ U^T U = I$$

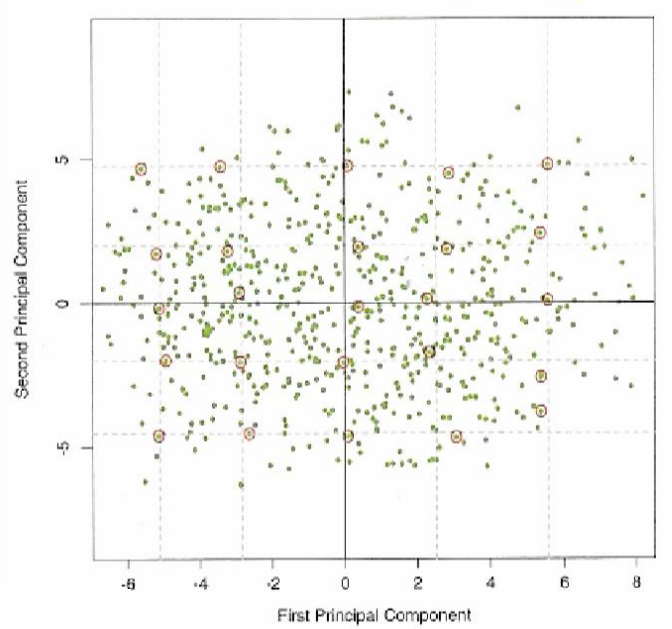
14.5. Principal Components (PCA)

- PCA can be used for handwritten digits

Data in $\mathbb{R}^{256}$



2-D projections



$$\hat{f}(\lambda) = \bar{x} + \lambda_1 v_1 + \lambda_2 v_2$$

$$= \bar{x} + \lambda_1 \cdot + \lambda_2 \cdot .$$

$$\hat{f}(\lambda) = \bar{x} + \lambda_1 v_1 + \lambda_2 v_2$$

SVD (Singular Value Decomposition)

- Definition:
  - **Singular value decomposition** is a method of decomposing a matrix into the product of three matrices. For any $m \times n$ matrix A, there exist an $m \times m$ orthogonal matrix U, an $n \times n$ orthogonal matrix V, and an $m \times n$ diagonal matrix D such that

$$\mathbf{A} = \mathbf{UDV}^T$$

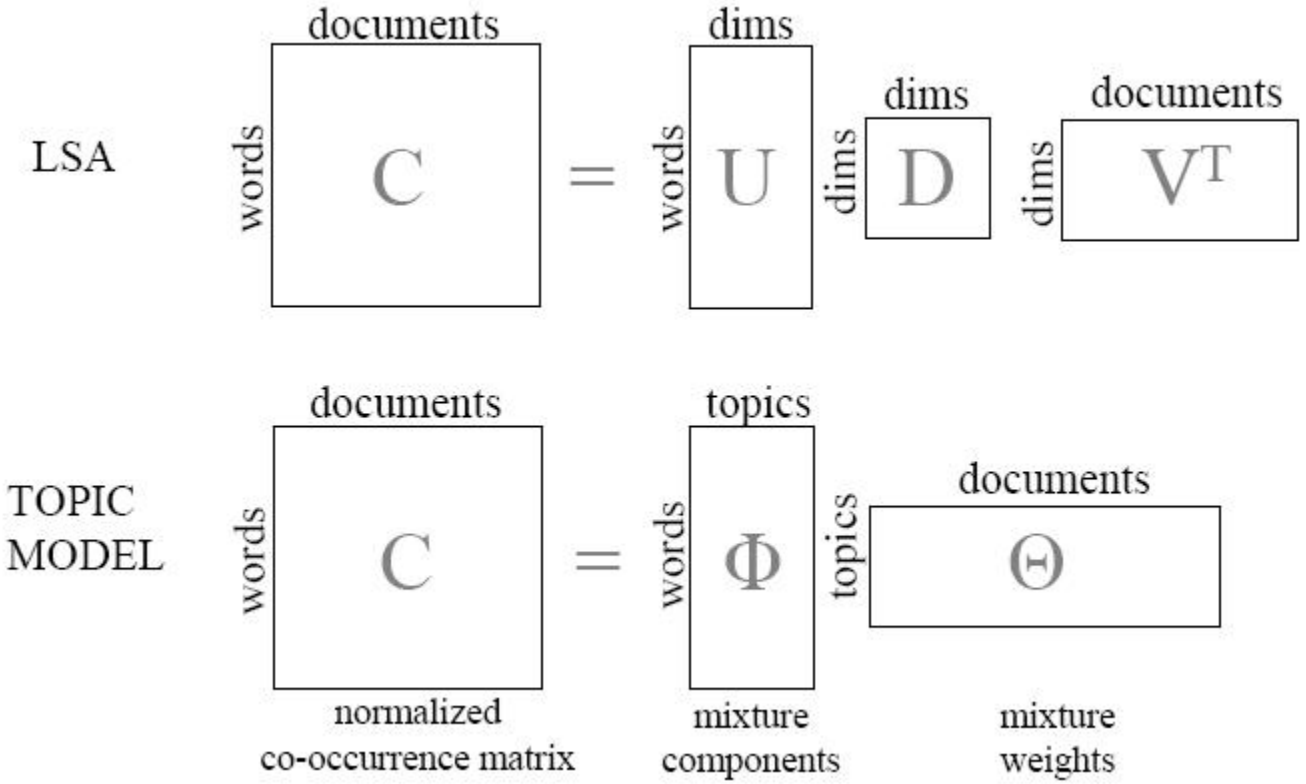
- Importance: It has wide applications in many fields such as data compression, image recognition, recommendation systems, and principal component analysis. It is an effective means of handling large-scale data and complex matrix operations.

SVD (Singular Value Decomposition)

- Interpretation:

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

- Data representation



SVD (Singular Value Decomposition)

- Proof outline:

$$\mathbf{A}^T\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^T, \qquad (\mathbf{AV})^T\mathbf{AV} = \mathbf{D} = \left[diag(\lambda_1^2, \lambda_2^2, \cdots, \lambda_n^2)\right],$$

$$\mathbf{P} = \mathbf{AV}; \qquad \mathbf{P}^T\mathbf{P} = \mathbf{D}; \qquad \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r; \ \lambda_k = 0, \ k > r$$

$$\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n], \qquad \mathbf{u}_j = \mathbf{p}_j / \lambda_j, \ j = 1,2,\cdots,r$$

$$\mathbf{u}_i^T\mathbf{u}_j = \delta_{ij}, \ i, j = 1,2,\cdots,r$$

- Construct the rest vectors $\mathbf{u}_j$, $j > r$ from the null space of $\mathbf{A}^T$ such that $\{\mathbf{u}_j\}_{j=1}^m$, are orthogonal.

Factor Analysis (FA)

Goal : find the latent variables within the observed ones

Latent representation of data (using SVD)

$$\mathbf{X} = \mathbf{UDV}^T = \mathbf{SA}^T;\ \mathbf{U}\ is\ \mathrm{N}\times\mathrm{p}\ \text{orthognal matrix}\ \mathrm{U}^T\mathrm{U}=\mathrm{I}$$

where $\mathbf{S} = \sqrt{N}\mathbf{U}$ and $\mathbf{A}^T = \dfrac{\mathbf{DV}^T}{\sqrt{N}}$

We have the following relations

$$\begin{cases} X_1 = a_{11}S_1 + a_{12}S_2 + \cdots + a_{1p}S_p \\ X_2 = a_{21}S_1 + a_{22}S_2 + \cdots + a_{2p}S_p \\ \vdots \\ X_p = a_{p1}S_1 + a_{p2}S_2 + \cdots + a_{pp}S_p \end{cases}$$

PCA estimates a latent variable model

$$\begin{cases} X_1 = a_{11}S_1 + a_{12}S_2 + \cdots + a_{1p}S_p \\ X_2 = a_{21}S_1 + a_{22}S_2 + \cdots + a_{2p}S_p \\ \vdots \\ X_p = a_{p1}S_1 + a_{p2}S_2 + \cdots + a_{pp}S_p \end{cases}$$

Due to hypothesis on X there are many solutions

$$\mathbf{X} = \mathbf{AS} = \mathbf{AR}^T\mathbf{RS} = \mathbf{A}^*\mathbf{S}^*$$

as

$$Cov(\mathbf{S}^*) = \mathbf{R}Cov(\mathbf{S})\mathbf{R}^T = \mathbf{I}$$

$\mathbf{S}^*$ is also the factor basis

Factor Analysis

Idea : use $q < p$

factor *loadings*

$$\begin{cases} X_1 = a_{11}S_1 + a_{12}S_2 + \cdots + a_{1q}S_q + \varepsilon_1 \\ X_2 = a_{21}S_1 + a_{22}S_2 + \cdots + a_{2q}S_q + \varepsilon_2 \\ \vdots \\ X_p = a_{p1}S_1 + a_{p2}S_2 + \cdots + a_{pq}S_q + \varepsilon_p \end{cases}$$

or $\mathbf{X} = \mathbf{AS} + \boldsymbol{\varepsilon}$, with $_j$ is mutually independent

Parameters are given by the covariance matrix

$$\boldsymbol{\Sigma} = \mathrm{AA}^T + \mathrm{D}_\varepsilon$$

with

$$\mathrm{D}_\varepsilon = \mathrm{diag}[\mathrm{var}(\varepsilon_1), \cdots, \mathrm{var}(\varepsilon_p)]$$

PCA and FA are related

FA can be further generalized

$$\begin{cases} X_1 = a_{11}S_1 + a_{12}S_2 + \cdots + a_{1q}S_q + \varepsilon_1 \\ X_2 = a_{21}S_1 + a_{22}S_2 + \cdots + a_{2q}S_q + \varepsilon_2 \\ \vdots \\ X_p = a_{p1}S_1 + a_{p2}S_2 + \cdots + a_{pq}S_q + \varepsilon_p \end{cases}$$

PCA : *all* variability in an item should be used
FA : only the variability in common is used

In most cases, the methods yield similar results PCA is preferred for data

reduction
FA is preferred for structure detection

Unsupervised Learning

1. Introduction
2. Association Rules & Cluster Analysis
4. Self-Organizing Maps
5. Principal Components, SVD
6. **Non-negative Matrix Factorization**
7. Independent Component Analysis
8. Multidimensional Scaling
9. Nonlinear Dimension Reduction
10. The Google PageRank Algorithm

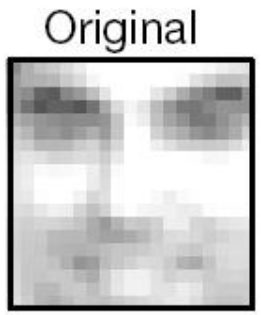
Problem

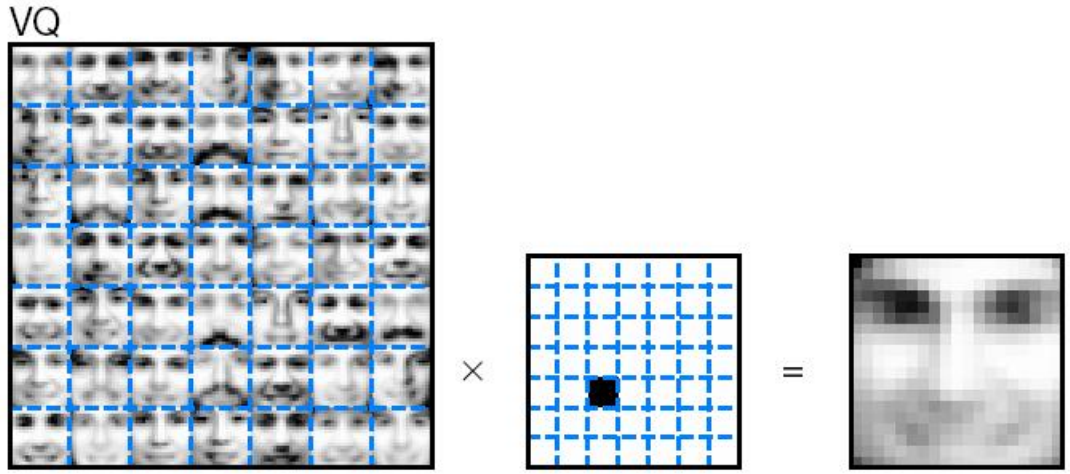
Given a set of images:

1. Create a set of basis images that can be linearly combined to create new images
2. Find the set of weights to reproduce every input image from the basis images
   - One set of weights for each input image

The reconstructed image is the basis image that is closest to the input image.

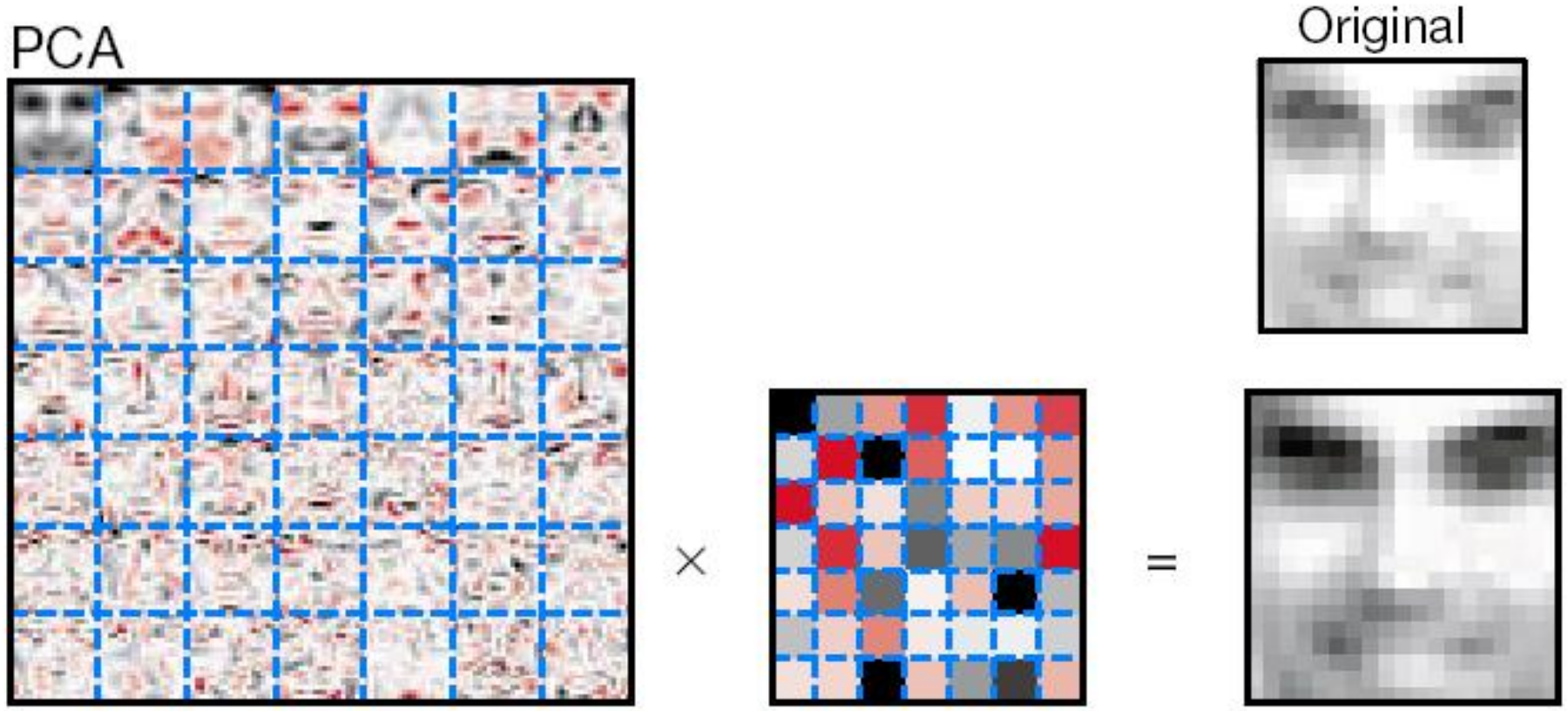
- Vector Quantization





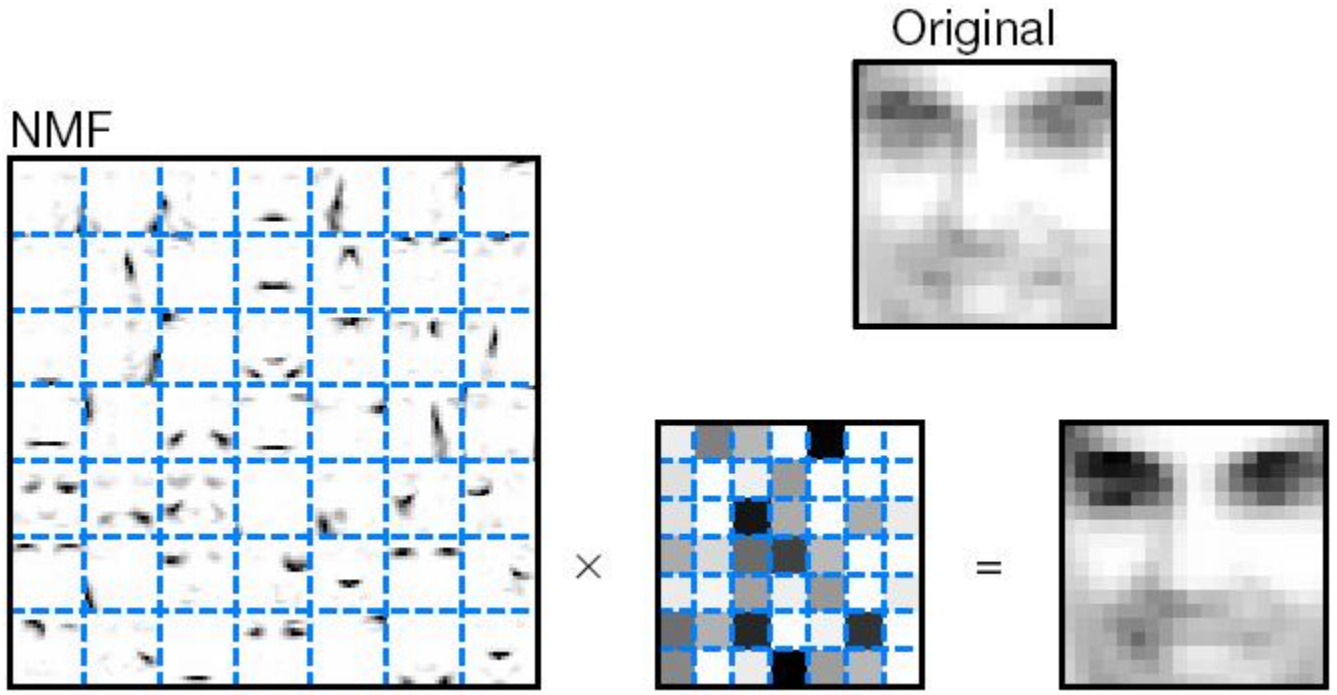
PCA

- Find a set of orthogonal basis images
- The reconstructed image is a linear combination of the basis images



Non-negative Matrix Factorization

- Like PCA, except the coefficients in the linear combination cannot be negative



- By constraining the weights, we can control how images are represented into the weighted sum of basis functions

Objective function

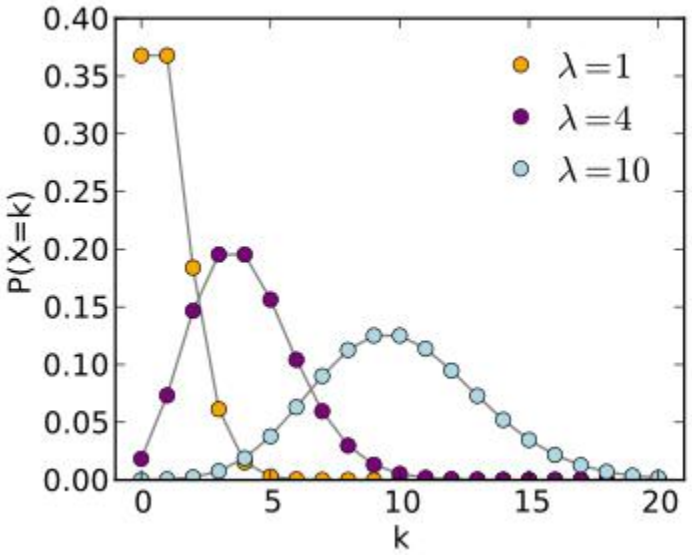
- Let the value of a pixel in an original input image be X.

$$X = WH + \varepsilon, \qquad W \in R^{N \times r}, H \in R^{r \times p}$$

- Let $(WH)_{ij}$ be the reconstructed pixel.
- If we consider X to be a noisy version of $(WH)_{ij}$, then the Poisson PDF of V is

$$P(x \mid (WH)_{ij}) = \frac{(WH)_{ij}^{x} e^{-(WH)_{ij}}}{x!}$$

$$P(x \mid \lambda) = \frac{\lambda^{x} e^{-\lambda}}{x!}$$



Objective function

$$P(x \mid (WH)_{ij}) = \frac{(WH)_{ij}^{x} e^{-(WH)_{ij}}}{x!}$$

- Now we will maximize the log probability of this PDF over W and H, leaving the relevant objective function to be:

$$L(W,H) = \sum_{i=1}^{N}\sum_{j}^{p}\left[x_{ij} \log(WH)_{ij} - (WH)_{ij} \right]$$

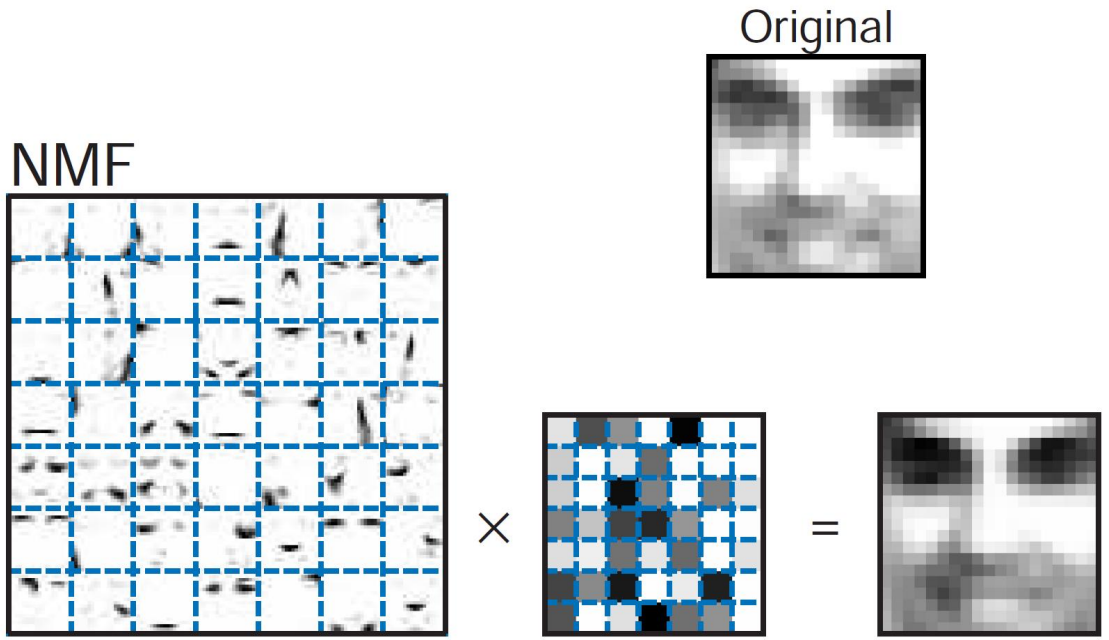
$$W = W + \eta \frac{\partial L(W,H)}{\partial W} = \left(w_{ik} + \eta \sum_{j=1}^{p} x_{ij} h_{kj} \,/\, (WH)_{ij} - \sum_{j=1}^{p} h_{kj} \right)$$

$$H = H + \eta \frac{\partial L(W,H)}{\partial H} = \left(h_{kj} + \eta \sum_{i=1}^{N} x_{ij} w_{ik} \,/\, (WH)_{ij} - \sum_{i=1}^{N} w_{ik} \right)$$

Deriving Update Rules

- Gradient Descent Rule： (Alternating algorithm)

$$w_{ik} \leftarrow w_{ik} \frac{\sum_{j=1}^{p} h_{kj}\, x_{ij} / (WH)_{ij}}{\sum_{j=1}^{p} h_{kj}}; \qquad h_{kj} \leftarrow h_{kj} \frac{\sum_{i=1}^{N} w_{ik}\, x_{ij} / (WH)_{ij}}{\sum_{i=1}^{N} w_{ik}}$$



Unsupervised Learning

1. Introduction
2. Association Rules & Cluster Analysis
4. Self-Organizing Maps
5. Principal Components, SVD
6. Non-negative Matrix Factorization
7. Independent Component Analysis
8. Multidimensional Scaling
9. Nonlinear Dimension Reduction
10. The Google PageRank Algorithm