Advanced Probabilistic Machine Learning SSY316

Unsupervised learning

Alexandre Graell i Amat
alexandre.graell@chalmers.se
https://sites.google.com/site/agraellamat

December 12 and 15, 2023

Unsupervised learning

A branch of machine learning that operates over unlabeled data sets $\mathcal{D} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$

Aim: Find structure in the data $\longrightarrow$ study of generative models.

- Typically characterized by the presence of hidden (latent) variables that help explain structure of the data

Goal: Given data set $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N} \sim_{i.i.d} p(\boldsymbol{x})$, learn some useful properties of $p(\boldsymbol{x})$.

Unsupervised learning

Key differences to supervised learning:

- Model $p(\boldsymbol{x})$ (not $p(t|\boldsymbol{x})$)
- Need to work with multivariate probability models ($\boldsymbol{x}$ a vector), whereas t often a scalar in supervised learning

Unsupervised learning

A few examples:

Discovering clusters: Clustering data into groups

- Assumes presence of an unobserved label r_n (hidden, latent variable) associated to each $\boldsymbol{x}_n$
- Goal: Recovering labels r_n for all $\boldsymbol{x}_n \in \mathcal{D}$

Dimensionality reduction: Reduce dimensionality of data by projecting it to lower dimensional subspace which captures its essence

- Motivation: Data may appear high dimensional, but there may be a small number of degrees of variability (latent factors)
- Possible to highlight independent explanatory factors; ease visualization and interpretation

Generation of new samples: Learn a machine that produces samples approximately distributed according to $p(\boldsymbol{x})$

- Examples: Computer graphics for gaming; software that can produce artificial scenes based on a given description

Clustering

Two approaches:

- K-means clustering
- Mixture of Gaussians model

K-means clustering

Goal: Given data set $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$, partition it into K clusters.

- Cluster indices encoded by categorical variables $\boldsymbol{r}_n$ via one-hot encoding
- r_{nk}: k-th component of $\boldsymbol{r}_n$
- $r_{nk} = 1$ if $\boldsymbol{x}_n$ assigned to cluster k, $r_{nk} = 0$ otherwise

How should we cluster?

Intuition: Clusters comprise points closer in distance $\longrightarrow$ cluster together points that are mutually close in Euclidean distance.

- K-means assigns all points in same cluster to a prototype representative $\boldsymbol{\mu}_k \longrightarrow \{\boldsymbol{\mu}_k\}$ represent centers of clusters

Goal (revisited): Find assignment of data points to clusters and $\{\boldsymbol{\mu}_k\}$ such that all points within a given cluster can be quantized to μ_k with minimal quadratic loss.

K-means clustering

Goal (revisited): Find assignment of data points to clusters and $\{\boldsymbol{\mu}_k\}$ such that all points within a given cluster can be quantized to μ_k with minimal quadratic loss.

$$\begin{aligned}\min_{\{\boldsymbol{r}_n\},\{\boldsymbol{\mu}_k\}} \; J &\triangleq \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk} d(\boldsymbol{x}_n, \boldsymbol{\mu}_k) \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2\end{aligned}$$

When $d(\boldsymbol{x}_n, \boldsymbol{\mu}_k)$ is a general distance metric: K-medoids.

K-means clustering

$$\min_{\{\boldsymbol{r}_n\},\{\boldsymbol{\mu}_k\}} J \triangleq \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2 \tag{1}$$

K-means: Solves minimization problem by alternately optimizing $\boldsymbol{r}_n$ and $\boldsymbol{\mu}_k$.

Algorithm:

1. Initialization: Initialize $\{\boldsymbol{\mu}_k^{(0)}\}$

For $\ell = 1, \dots$:

2. Expectation step (E step): For fixed $\{\boldsymbol{\mu}_k^{(\ell-1)}\}$, solve (1) over $\{\boldsymbol{r}_n\}$:

$$r_{nk}^{(\ell)} = \begin{cases} 1 & \text{for } k = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{\mu}_j^{(\ell-1)}\|^2 \\ 0 & \text{otherwise} \end{cases}$$

3. Maximization step (M step): For fixed $\{\boldsymbol{r}_n\}$, solve (1) over $\{\boldsymbol{\mu}_k\}$:

$$\boldsymbol{\mu}_k^{(\ell)} = \frac{\sum_{n=1}^{N} r_{nk}^{(\ell)} \boldsymbol{x}_n}{\sum_{n=1}^{N} r_{nk}^{(\ell)}}$$

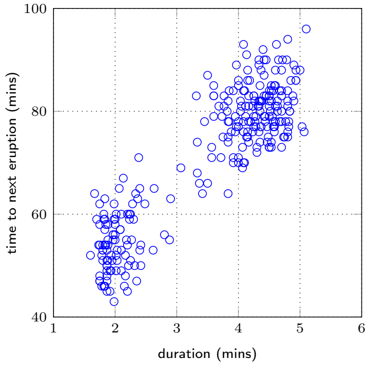
4. Repeat 2.–4. until convergence

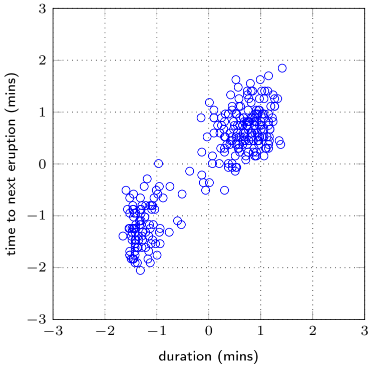
K-means algorithm applied to the Old Faithful dataset

Old Faithful dataset:

- 272 measurements of the eruption of Yellowstone's Old Faithful geyser
- Two variables: duration and time until next eruption (in mins)

The Old Faithful dataset

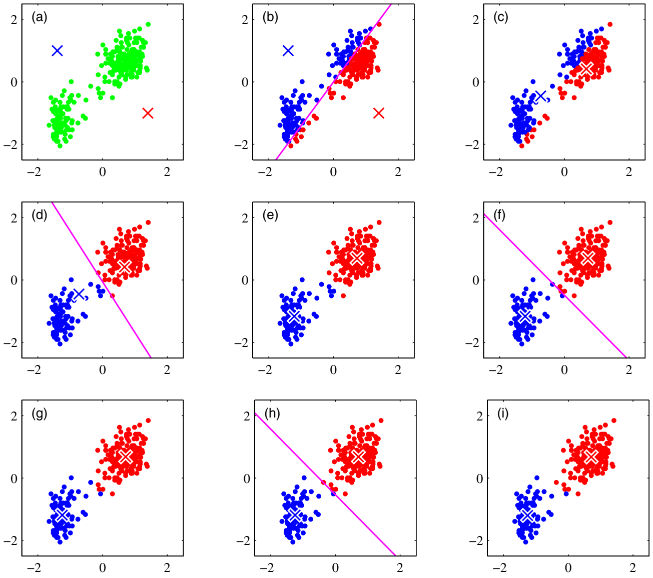




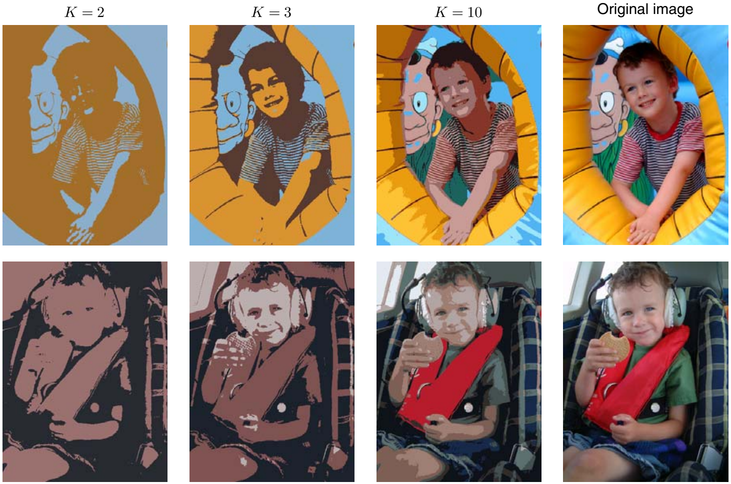
Old Faithful dataset:

- 272 measurements of the eruption of Yellowstone's Old Faithful geyser
- Two variables: duration and time until next eruption (in mins)

K-means algorithm applied to the Old Faithful dataset

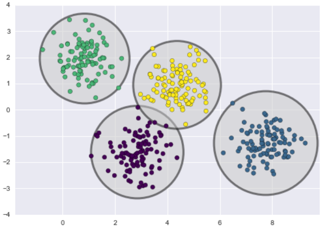


K-means clustering for image segmentation

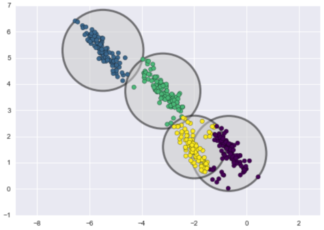


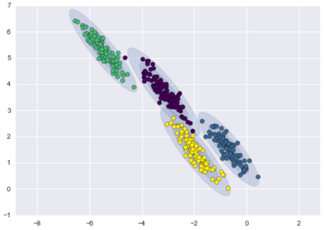
The Gaussian mixture model

K-means: Places a hyper-sphere at the center of each cluster



And if clusters are far from circular?





The Gaussian mixture model

Idea: Fit a mixture model, then compute posterior probability that point $\boldsymbol{x}_n$ belongs to cluster k, $p(\boldsymbol{z}_n = k|\boldsymbol{x}_n, \boldsymbol{\theta})$.

z: Random categorical variable, taking values on values $1, \dots, K$

- Can be represented by one-hot vector

Gaussian mixture model:

$$p(z_k = 1) = \pi_k$$
$$p(\boldsymbol{x}|\boldsymbol{z} = k) = p(\boldsymbol{x}|z_k = 1) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

We can write:

$$p(\boldsymbol{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$$
$$p(\boldsymbol{x}|\boldsymbol{z}) = \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k}$$

The Gaussian mixture model

We can write $p(\boldsymbol{x})$ as

$$p(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}) = \sum_{\boldsymbol{z}} p(\boldsymbol{z}) p(\boldsymbol{x}|\boldsymbol{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

and

$$\begin{aligned} p(z_k = 1|\boldsymbol{x}) &= \frac{p(z_k = 1) p(\boldsymbol{x}|z_k = 1)}{p(\boldsymbol{x})} \\ &= \frac{\pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \end{aligned}$$

$p(z_k = 1|\boldsymbol{x})$: Posterior probability that point $\boldsymbol{x}$ belongs to cluster k.

- K-means: Hard assignment of data points to clusters
- Mixture of Gaussians: Soft assignment based on posterior probabilities

The Gaussian mixture model

$\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$: Model parameters

Maximum likelihood learning:

$$\begin{aligned}
\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} \ \ln p(\mathcal{D}|\boldsymbol{\theta}) \\
&= \arg\max_{\boldsymbol{\theta}} \ \ln \left(\prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta}) \right) \\
&= \arg\max_{\boldsymbol{\theta}} \ \left(\sum_{n=1}^{N} \ln p(\boldsymbol{x}_n|\boldsymbol{\theta}) \right) \\
&= \arg\max_{\boldsymbol{\theta}} \ \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)
\end{aligned}$$

Analytical solution difficult! $\longrightarrow$ Expectation maximization.

The expectation maximization (EM) algorithm

- ML and MAP parameter estimate in general easy in machine learning with all relevant random variables observed (complete data)
- For missing data or latent variables (as in unsupervised learning) ML and MAP difficult

EM algorithm: An iterative algorithm to finding ML solutions for probabilistic models with latent (or missing) variables with (often) closed-form updates at each step.

- Alternates between inferring missing values given $\boldsymbol{\theta}$ (E step), then optimizing $\boldsymbol{\theta}$ given the “filled in” data (M step)

The expectation maximization (EM) algorithm

- $\mathcal{D} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$: Observed variables
- $\boldsymbol{Z} = (\boldsymbol{z}_1 \ \cdots \ \boldsymbol{z}_N)$: Latent variables

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} \ \ln p(\mathcal{D}|\boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \ \ln \left(\sum_{\boldsymbol{Z}} p(\mathcal{D}, \boldsymbol{Z}|\boldsymbol{\theta}) \right)\end{aligned}$$

Idea: Consider the maximization of the complete data log-likelihood

$$\ell_{\mathsf{c}} \triangleq \ln p(\mathcal{D}, \boldsymbol{Z}|\boldsymbol{\theta})$$

The expectation maximization (EM) algorithm

Complete data log-likelihood:

$$\ell_{\mathsf{c}}(\boldsymbol{\theta}) \triangleq \ln p(\mathcal{D}, \boldsymbol{Z}|\boldsymbol{\theta})$$

As we do not have values for $\boldsymbol{Z}$, we consider the expectation.

E step: Compute expected complete data log-likelihood

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(\ell-1)}) &= \mathbb{E}_{\mathbf{Z}\sim p(\boldsymbol{Z}|\mathcal{D},\boldsymbol{\theta}^{(\ell-1)})}\left[\ell_{\mathsf{c}}(\boldsymbol{\theta})\right] \\ &= \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\mathcal{D}, \boldsymbol{\theta}^{(\ell-1)})\ell_{\mathsf{c}}(\boldsymbol{\theta}) \\ &= \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\mathcal{D}, \boldsymbol{\theta}^{(\ell-1)}) \ln p(\mathcal{D}, \boldsymbol{Z}|\boldsymbol{\theta}) \end{aligned}$$

Q: auxiliary function

Expectation over posterior distribution of $\boldsymbol{Z}$ conditioned on old parameters $\boldsymbol{\theta}^{(\ell-1)}$ and observed data $\mathcal{D}$.

The expectation maximization (EM) algorithm

M step: Optimize auxiliary function Q

$$\boldsymbol{\theta}^{(\ell)} = \arg\max_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(\ell-1)})$$

Goal: Given $p(\mathcal{D}, \boldsymbol{Z}|\boldsymbol{\theta})$, maximize $p(\mathcal{D}|\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$.

Algorithm:

1. Initialization: Initialize $\boldsymbol{\theta}^{(0)}$

For $\ell = 1, \ldots$:

2. E step: Evaluate $p(\boldsymbol{Z}|\mathcal{D}, \boldsymbol{\theta}^{(\ell-1)})$
3. M step: Evaluate $\boldsymbol{\theta}^{(\ell)}$,

$$\boldsymbol{\theta}^{(\ell)} = \arg\max_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(\ell-1)})$$

4. If convergence criterion not satisfied, return to 2.

EM for Gaussian mixture models

Goal: Solve

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \ln p(\mathcal{D}|\boldsymbol{\theta})$$

We consider maximization of the complete data log-likelihood $\ln p(\mathcal{D}, \boldsymbol{Z}|\boldsymbol{\theta})$,

$$\begin{aligned}
\ln p(\mathcal{D}, \boldsymbol{Z}|\boldsymbol{\theta}) &= \ln \left(\prod_{n=1}^{N} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\boldsymbol{\theta}) \right) \\
&= \ln \left(\prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}} \right) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \ln \left(\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left(\ln \pi_k + \ln \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)
\end{aligned}$$

EM for Gaussian mixture models

As latent variables $\boldsymbol{Z}$ not available, we consider the expectation,

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(\ell-1)}) = \mathbb{E}_{\mathbf{Z}\sim p(\mathcal{Z}|\mathcal{D},\boldsymbol{\theta}^{(\ell-1)})}\left[\ln p(\mathcal{D}, \mathcal{Z}|\boldsymbol{\theta})\right]$$
$$= \sum_{n=1}^{N}\sum_{k=1}^{K}\mathbb{E}[z_{nk}]\left(\ln \pi_k + \ln \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right)$$

Expected value $\mathbb{E}[z_{nk}]$:

$$\mathbb{E}_{\boldsymbol{z}_n\sim p(\boldsymbol{z}_n|\boldsymbol{x}_n,\boldsymbol{\theta}^{(\ell-1)})}[z_{nk}] = p(z_{kn}=1|\boldsymbol{x}_n, \boldsymbol{\theta}^{(\ell-1)})$$

EM for Gaussian mixture models

Algorithm:

1. Initialization: Initialize $\boldsymbol{\theta}^{(0)} = (\boldsymbol{\pi}^{(0)}, \boldsymbol{\mu}^{(0)}, \boldsymbol{\Sigma}^{(0)})$

For $\ell = 1, \dots :$

2. E step: Evaluate $p(\boldsymbol{Z}|\mathcal{D}, \boldsymbol{\theta}^{(\ell-1)})$ via Bayes' theorem for each $\boldsymbol{x}_n$,

$$p(z_{kn} = 1|\boldsymbol{x}_n, \boldsymbol{\theta}^{(\ell-1)}) = \frac{\pi_k^{(\ell-1)} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k^{(\ell-1)}, \boldsymbol{\Sigma}_k^{(\ell-1)})}{\sum_{j=1}^{N} \pi_j^{(\ell-1)} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j^{(\ell-1)}, \boldsymbol{\Sigma}_j^{(\ell-1)})}$$

3. M step: Evaluate $\boldsymbol{\theta}^{(\ell)} = (\boldsymbol{\pi}^{(\ell)}, \boldsymbol{\mu}^{(\ell)}, \boldsymbol{\Sigma}^{(\ell)})$, obtaining

$$\pi_k^{(\ell)} = \frac{N_k}{N} \qquad \boldsymbol{\mu}_k^{(\ell)} = \frac{1}{N_k} \sum_{n=1}^{N} p(z_{kn} = 1|\boldsymbol{x}_n, \boldsymbol{\theta}^{(\ell-1)}) \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k^{(\ell)} = \frac{1}{N_k} \sum_{n=1}^{N} p(z_{kn} = 1|\boldsymbol{x}_n, \boldsymbol{\theta}^{(\ell-1)}) (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\mathsf{T}$$

with

$$N_k = \sum_{n=1}^{N} p(z_{kn} = 1|\boldsymbol{x}_n, \boldsymbol{\theta}^{(\ell-1)})$$

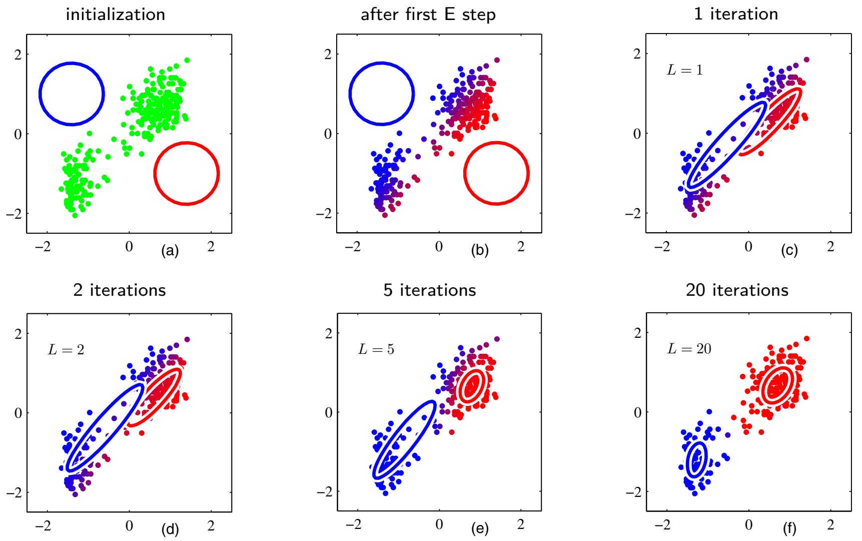
4. If convergence criterion not satisfied, return to 2.

EM algorithm applied to the Old Faithful dataset

Old Faithful dataset:

- 272 measurements of the eruption of Yellowstone's Old Faithful geyser
- Two variables: duration and time until next eruption (in mins)

EM algorithm applied to the Old Faithful dataset



- Initialization: $\boldsymbol{\mu}_1^{(0)}$, $\boldsymbol{\mu}_2^{(0)}$, $\boldsymbol{\Sigma}_1^{(0)}$, $\boldsymbol{\Sigma}_2^{(0)}$
- $\text{color}(n,\ell) = p(z_n = \text{blue}|\boldsymbol{x}_n, \boldsymbol{\theta}^{(\ell-1)})\text{blue} + p(z_n = \text{red}|\boldsymbol{x}_n, \boldsymbol{\theta}^{(\ell-1)})\text{red}$

The expectation maximization (EM) algorithm

EM algorithm can also be used to find MAP solution.

- Need to introduce a prior on $\boldsymbol{\theta}$, $p(\boldsymbol{\theta})$
- E step same as for ML
- M step modified to

$$\boldsymbol{\theta}^{(\ell)} = \arg\max_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(\ell-1)}) + \ln p(\boldsymbol{\theta})$$

Dimensionality reduction: Principal component analysis

Goal: Reduce the dimensionality of high-dimensional data $\mathcal{D} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_N)$, $\boldsymbol{x} \in \mathbb{R}^d$, by linearly projecting them onto a lower-dimensional subspace such that reconstruction error made by projection is minimum (retain as much information as possible).

- PCA accomplishes this by finding the directions (principal components) that capture the maximum variance in the data

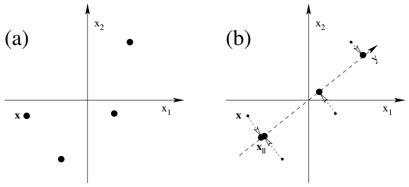
Why PCA?

Data may appear high dimensional, but maybe only a small number of degrees of variability (latent factors)

Face recognition: Only a few latent factors describe most of the variability (lighting, pose, identity).

Principal component analysis

How can we find linear subspace that minimizes reconstruction error?



data points in 2D projection onto 1D

- Have to choose 1-dimensional space$\longrightarrow$ can be specified by a unit vector $\boldsymbol{u}$
- Project 2-dimensional data points onto it: $\boldsymbol{u}^\mathsf{T}\boldsymbol{x}$ (a scalar)
- Actual coordinate in the 1-dimensional space: $\boldsymbol{u}(\boldsymbol{u}^\mathsf{T}\boldsymbol{x})$

Reconstruction error:

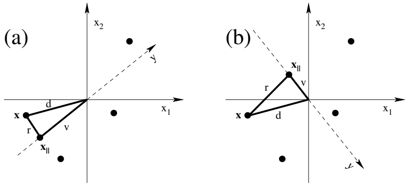
$$\mathrm{MSE}(\boldsymbol{u}) = \frac{1}{N}\sum_{i=1}^{N}||\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i||^2 \quad , \tilde{\boldsymbol{x}}_i = \boldsymbol{u}\boldsymbol{u}^\mathsf{T}\boldsymbol{x}_i$$

Reconstruction error and variance

Reconstruction error:

$$\mathsf{MSE}(\boldsymbol{u}) = \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i||^2 \quad , \tilde{\boldsymbol{x}}_i = \boldsymbol{u}\boldsymbol{u}^\mathsf{T}\boldsymbol{x}_i$$

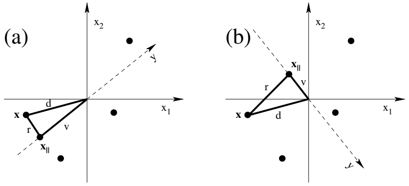
How can we find the linear subspace that minimizes reconstruction error?



- d: distance of $\boldsymbol{x}$ from origin
- r: distance of $\boldsymbol{x}$ from $\tilde{\boldsymbol{x}}$ in the subspace
- v: distance of $\tilde{\boldsymbol{x}}$ from origin the origin

r and v depend on direction of subspace while d does not.

Reconstruction error and variance



We have:

$$r^2 + v^2 = d^2$$

- r^2 contributes to reconstruction error
- v^2 contributes to variance of projected data within the subspace

Minimizing reconstruction error is equivalent to maximizing variance of projected data!

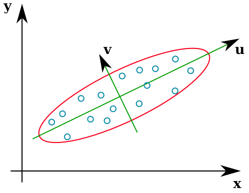
PCA can be “misled” by directions in which variance is high because of the measurement scale $\longrightarrow$ Standardize data first.

Principal component analysis

Principal components: Sequence of m (unit) vectors, with i-th vector the direction of a line that best fits data while being orthogonal to first $i-1$ vectors

Best-fitting line $\equiv$ direction along which data shows maximum variance

1. 1st PC: direction in space along which projections have largest variance
2. 2nd PC: direction maximizing variance among all directions orthogonal to 1st PC
3. k-th PC: variance-maximizing direction orthogonal to previous $k-1$ components



Reconstruction error and variance

- $\{\boldsymbol{x}_i \in \mathbb{R}^d\}$: high-dimensional observation
- $\boldsymbol{u}_j \in \mathbb{R}^d$: j-th principal direction
- $z_i = \boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i$: projection of $\boldsymbol{x}_i$ onto $\boldsymbol{u}_1$

Consider best 1-dimensional space, $\boldsymbol{u}_1$:

$$
\begin{aligned}
\mathsf{MSE}(\boldsymbol{u}_1) &= \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i||^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} ||\boldsymbol{x}_i - \boldsymbol{u}_1 z_i||^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - z_i\boldsymbol{u}_i)^\mathsf{T}(\boldsymbol{x}_i - z_i\boldsymbol{u}_1) \\
&= \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i^\mathsf{T}\boldsymbol{x}_i - 2z_i\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i + z_i^2\boldsymbol{u}_i^\mathsf{T}\boldsymbol{u}_i) \\
&= \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i^\mathsf{T}\boldsymbol{x}_i - 2z_i\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i + z_i^2) = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i^\mathsf{T}\boldsymbol{x}_i - (\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i)^2)
\end{aligned}
$$

Reconstruction error and variance

$$\mathsf{MSE}(\boldsymbol{u}_1) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i^\mathsf{T}\boldsymbol{x}_i - (\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i)^2)$$

Minimizing $\mathsf{MSE}(\boldsymbol{u}_1)$ is equivalent to maximizing

$$\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i)^2$$

(sample mean of $(\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i)^2$!)

Recall:

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

Hence:

$$\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i)^2 = \left(\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i\right)^2 + \mathsf{Var}[\tilde{\boldsymbol{z}}_1] = \mathsf{Var}[\tilde{\boldsymbol{z}}_1]$$

Reconstruction error and variance

1-st principal component:

$$\begin{aligned}\boldsymbol{u}_1^* &= \arg\min_{\boldsymbol{u}_1} \mathsf{MSE}(\boldsymbol{u}_1)\\ &= \arg\max_{\boldsymbol{u}_1} \mathsf{Var}[\tilde{\boldsymbol{z}}_1]\end{aligned}$$

$\tilde{\boldsymbol{z}}_1$: projection of all data vectors $\{\boldsymbol{x}_i\}$ onto 1-dimensional space defined by $\boldsymbol{u}_1$

Principal component analysis

d-dimensional data $\mathcal{D} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_N)$

Goal: Project data onto a space of dimension $m < d$ while maximizing variance of projected data.

Projection onto a one-dimensional space ($m = 1$):

- Can define direction of this space using a d-dimensional vector $\boldsymbol{u}_1$ ($\|\boldsymbol{u}\| = \boldsymbol{u}_1^\mathsf{T}\boldsymbol{u} = 1$)
- Project each $\boldsymbol{x}_i$ onto a scalar value as $\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i$
- Variance of projected data:

$$\begin{aligned}\mathrm{Var}[\tilde{z}_1] &= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i)^2 \\ &= \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{u}_1^\mathsf{T}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T}\boldsymbol{u}_1 = \boldsymbol{u}_1^\mathsf{T}\boldsymbol{S}\boldsymbol{u}_1\,,\end{aligned}$$

with $\boldsymbol{S} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i\boldsymbol{x}_i^\mathsf{T}$ (empirical covariance matrix)

Principal component analysis

Projection onto a one-dimensional space $(m = 1)$:

Want to choose direction that maximizes variance $\longrightarrow$ Need to maximize variance $\boldsymbol{u}_1^\mathsf{T} \boldsymbol{S} \boldsymbol{u}_1$ with respect to $\boldsymbol{u}_1$.

- Need to prevent $\|\boldsymbol{u}_1\| \to \infty$: Can set the constraint $\|\boldsymbol{u}\| = \boldsymbol{u}_1^\mathsf{T} \boldsymbol{u} = 1$,

$$\boldsymbol{u}_1^* = \arg\max_{\boldsymbol{u}_1} \ \boldsymbol{u}_1^\mathsf{T} \boldsymbol{S} \boldsymbol{u}_1 + \lambda_1 (1 - \boldsymbol{u}_1^\mathsf{T} \boldsymbol{u}_1)$$

  λ_1: Lagrange multiplier

Solution: Differentiate with respect to $\boldsymbol{u}_1$ and equate to zero:

$$\boldsymbol{S} \boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1$$

$\boldsymbol{S} \boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1$ corresponds to the eigenvector equation of matrix $\boldsymbol{S}$!

Eigenvalues and eigenvectors

Eigenvalue and eigenvector:

Let $\boldsymbol{A}$ be a (real-valued) $n \times n$ matrix. Then $\lambda \in \mathbb{R}$ and a nonzero vector $\boldsymbol{v}$ are said to be an eigenvalue and corresponding eigenvector of $\boldsymbol{A}$ if

$$\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{v}$$

Action of $\boldsymbol{A}$ on eigenvector $\boldsymbol{v}$ as if it were multiplied by a scalar $\longrightarrow$ Direction does not change, only its length is scaled.

Eigendecomposition: Most commonly used matrix decomposition.

An $n \times n$ matrix $\boldsymbol{A}$ with n linearly independent eigenvectors $\boldsymbol{v}_i$ can be factorized as

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\lambda}\boldsymbol{V}^{-1}$$

- $\boldsymbol{V}$: $n \times n$ matrix, $\boldsymbol{V} = (\boldsymbol{v}_1\, \boldsymbol{v}_2\, \ldots\, \boldsymbol{v}_n)$
- $\boldsymbol{\lambda} = \text{diag}(\boldsymbol{\lambda})$, $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n)$

Principal component analysis

Projection onto a one-dimensional space ($m = 1$):

- d-dimensional vector $\boldsymbol{u}_1$: define direction of space

$$\boldsymbol{S}\boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1$$

$\boldsymbol{u}_1$ must be an eigenvector of $\boldsymbol{S}$!

Left-multiplying $\boldsymbol{S}\boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1$ by $\boldsymbol{u}_1^\mathsf{T}$ and using $\boldsymbol{u}_1^\mathsf{T}\boldsymbol{u}_1 = 1$:

$$\begin{aligned}\boldsymbol{u}_1^\mathsf{T}\boldsymbol{S}\boldsymbol{u}_1 &= \lambda_1 \boldsymbol{u}_1^\mathsf{T}\boldsymbol{u}_1 \\ &= \lambda_1\end{aligned}$$

$\mathsf{Var}[\boldsymbol{u}_1^\mathsf{T}\mathbf{x}] \approx \boldsymbol{u}_1^\mathsf{T}\boldsymbol{S}\boldsymbol{u}_1$ is maximized when we set $\boldsymbol{u}_1$ equal to the eigenvector with largest eigenvalue (λ_1).

- Eigenvector $\boldsymbol{u}_1$: 1-st principal component

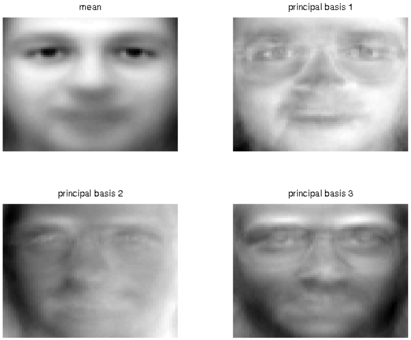
Principal component analysis

Can define additional principal components in an incremental fashion by choosing each new direction to be that which maximizes projected variance amongst all possible directions orthogonal to those already considered.

Optimal projection onto an m-dimensional space:

Defined by the m eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ of the data covariance matrix $\boldsymbol{S}$ corresponding to the m largest eigenvalues $\lambda_1 > \cdots > \lambda_m$

Principal component analysis for face recognition



Reading

“Pattern recognition and machine learning,”

Chapter 9 (Intro, 9.1, 9.2, 9.3 (until 9.3.2), 12.1 (until 12.1.3), 12.2 (until 12.2.2))