4F13: Machine Learning

Lecture 3-4: Unsupervised Learning

Zoubin Ghahramani
zoubin@eng.cam.ac.uk

Department of Engineering
University of Cambridge

Michaelmas, 2006

http://learning.eng.cam.ac.uk/zoubin/ml06/
Key Ingredients of Machine Learning

Data

Let \( y = (y_1, y_2, \ldots, y_D) \) denote a data point, and \( D = \{y_1, y_2, \ldots, y_N\} \), a data set.

Predictions

We are generally interested in predicting something based on the observed data set.

Given \( D \) what can we say about \( y_{N+1} \)?

Given \( D \) and \( y_{N+1,1}, y_{N+1,2}, \ldots, y_{N+1,D-1} \), what can we say about \( y_{N+1,D} \)?

Model

To make predictions, we need to make some assumptions. We can often express these assumptions in the form of a model, with some parameters, \( \theta \).

Given data \( D \), we learn the model parameters \( \theta \), from which we can predict new data points.

The model can often be expressed as a probability distribution over data points.
A few simple data sets

A more interesting data set:
Here $D = 2$, $y \in \mathbb{R}^2$. 
A very simple model

Univariate Gaussian density \((y \in \mathbb{R})\):

\[
p(y|\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{(y - \mu)^2}{2\sigma^2} \right\}
\]

This model has parameters \(\theta = \{\mu, \sigma\}\) which model the mean and standard deviation of the data, respectively.
A slightly more complicated model

Multivariate Gaussian density \((y \in \mathbb{R}^D)\):

\[
p(y|\mu, \Sigma) = |2\pi\Sigma|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (y - \mu)^\top \Sigma^{-1} (y - \mu) \right\}
\]

\[
\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]

This model has parameters \(\theta = \{\mu, \Sigma\}\) which model the mean and covariance matrix of the data.
The multivariate Gaussian density

\[ \mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix} \]

\[ \mu = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix} \]
Assume the data were generated independently from the model. We can measure the likelihood of the model:

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{N} p(y_n|\theta)$$

Clearly, the third model is a better fit to the data than the others:

$$\log p(\mathcal{D}|\theta_1) = -55.38$$
$$\log p(\mathcal{D}|\theta_2) = -238.29$$
$$\log p(\mathcal{D}|\theta_2) = -22.14$$
The likelihood function

Data set $\mathcal{D} = \{y_1, \ldots, y_N\}$, the likelihood: $p(\mathcal{D}|\mu, \Sigma) = \prod_{n=1}^{N} p(y_n|\mu, \Sigma)$ is a function of the model parameters

The maximum likelihood (ML) procedure finds parameters $\theta = \{\mu, \Sigma\}$ such that:

$$\theta_{ML} = \text{argmax}_\theta p(\mathcal{D}|\theta)$$
Finding Maximum Likelihood Estimate for a Gaussian

Data set \( \mathcal{D} = \{ \mathbf{y}_1, \ldots, \mathbf{y}_N \} \), likelihood: 
\[
p(\mathcal{D}|\mu, \Sigma) = \prod_{n=1}^{N} p(\mathbf{y}_n|\mu, \Sigma)
\]

Maximise likelihood \( \Leftrightarrow \) maximise log likelihood

**Goal:** find \( \mu \) and \( \Sigma \) that maximise log likelihood:

\[
\mathcal{L} = \log \prod_{n=1}^{N} p(\mathbf{y}_n|\mu, \Sigma) = \sum_{n} \log p(\mathbf{y}_n|\mu, \Sigma)
\]

\[
= -\frac{N}{2} \log |2\pi \Sigma| - \frac{1}{2} \sum_{n} (\mathbf{y}_n - \mathbf{\mu})^\top \Sigma^{-1} (\mathbf{y}_n - \mathbf{\mu})
\]

**Note:** equivalently, minimise \(-\mathcal{L}\), which is *quadratic* in \( \mu \)

**Procedure:** take derivatives and set to zero:

\[
\frac{\partial \mathcal{L}}{\partial \mu} = 0 \quad \Rightarrow \quad \hat{\mu} = \frac{1}{N} \sum_{n} \mathbf{y}_n \quad \text{(sample mean)}
\]

\[
\frac{\partial \mathcal{L}}{\partial \Sigma} = 0 \quad \Rightarrow \quad \hat{\Sigma} = \frac{1}{N} \sum_{n} (\mathbf{y}_n - \hat{\mu})(\mathbf{y}_n - \hat{\mu})^\top \quad \text{(sample covariance)}
\]
Two very simple data sets

What are the maximum likelihood estimates of $\theta$ for these data sets?

Does this make sense?
Bayesian Learning

Apply the basic rules of probability to learning from data. Use probability distributions to represent uncertainty.

Data set: \( \mathcal{D} = \{y_1, \ldots, y_N\} \)
Model parameters: \( \theta \)

Prior probabilities of model parameters: \( P(\theta) \)
Model of data given parameters (likelihood model): \( P(y|\theta) \)

If the data are independently and identically distributed then:

\[
P(\mathcal{D}|\theta) = \prod_{n=1}^{N} P(y_n|\theta)
\]

Posterior probability of model parameters:

\[
P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})}
\]
Limitations of the Multivariate Gaussian

Gaussians are fundamental and widespread, but not every distribution of interest is Gaussian.

- Some processes produce outliers.
- Some data has higher-order or non-linear structure.
- Not all random processes fit the central limit theorem.
- Even if data are Gaussian, if $D$ is large the full multivariate Gaussian model may be difficult to handle. There are $D(D + 1)/2$ parameters in the covariance matrix.
Factor Analysis

Factor analysis models high dimensional data \( y \) in terms of a linear transformation of some smaller number of latent factors, \( x \).

Linear generative model: 
\[
y_d = \sum_{k=1}^{K} \Lambda_{dk} x_k + \epsilon_d
\]

- \( x_k \) are independent \( \mathcal{N}(0, 1) \) Gaussian factors
- \( \epsilon_d \) are independent \( \mathcal{N}(0, \Psi_{dd}) \) Gaussian noise
- \( K < D \)

Properties:

- \( p(x) = \mathcal{N}(0, I) \) and \( y = \Lambda x + \epsilon \)
- Since \( p(\epsilon) = \mathcal{N}(0, \Psi) \), we get that \( p(y|x) = \mathcal{N}(\Lambda x, \Psi) \)
- \( p(y) = \int p(x)p(y|x)dx = \mathcal{N}(0, \Lambda \Lambda^\top + \Psi) \) where \( \Lambda \) is a \( D \times K \) matrix, and \( \Psi \) is diagonal.

latent = hidden = unobserved = missing
Ways of thinking about Factor Analysis (FA)

- FA models high dimensional data in terms of a linear transformation of some smaller number of latent factors.

- FA is a method for parameterizing a $D \times D$ covariance matrix $\Sigma$ in terms of $D \times K + D$ parameters, $\Lambda \Lambda^\top + \Psi$. Since $K$ can be chosen by the user, this means that factor analysis can be applied to very high dimensional datasets.

- FA is a method for modelling correlations among the observed variables.

- FA is a linear regression model, where the inputs are assumed to be hidden.

- FA is a method for doing dimensionality reduction. Given $y$ we can represent it by the mean of $x$. FA finds a low-dimensional projection of high dimensional data that captures the correlation structure of the data.

\[
p(x|y) = \frac{p(x)p(y|x)}{p(y)} = \mathcal{N}(\beta y, I - \beta \Lambda) \quad \text{where} \quad \beta = \Lambda^\top (\Lambda \Lambda^\top + \Psi)^{-1}
\]
Factor Analysis

- ML learning for FA aims to fit $\Lambda$ and $\Psi$ given data. There is no closed form solution for ML parameters.

- Number of free parameters (corrected for symmetries):

  $$DK + D - \frac{K(K-1)}{2} < \frac{D(D+1)}{2}$$

- A Bayesian treatment would start with priors over $\Lambda$ and $\Psi$ and infer their posterior given the data.

  $$p(\Lambda, \Psi | D) = \frac{p(D | \Lambda, \Psi)p(\Lambda, \Psi)}{p(D)}$$

\[ \mu = 0 \]
\[ \Sigma \approx \Lambda \Lambda^\top + \Psi \]
Latent Variable Models

Explain correlations in $y$ by assuming some latent variables $x$

$$x \sim p(x|\theta_x)$$

$$y|x \sim p(y|x, \theta_y)$$

$$p(x, y|\theta_x, \theta_y) = p(y|x, \theta_y)p(x|\theta_x)$$

$$p(y|\theta_x, \theta_y) = \int dx \ p(y|x, \theta_y)p(x|\theta_x)$$
Probabilistic Principal Components Analysis (pPCA)

Linear generative model: \( y_d = \sum_{k=1}^{K} \Lambda_{dk} x_k + \epsilon_d \)

- \( x_k \) are independent \( \mathcal{N}(0, 1) \) Gaussian factors
- \( \epsilon_d \) are independent \( \mathcal{N}(0, \sigma^2) \) Gaussian noise
- \( K < D \)
- pPCA is factor analysis with isotropic noise: \( \Psi = \sigma^2 I \)
- pPCA finds same principal subspace as PCA, but is a well-defined probabilistic model.
Principal Components Analysis (PCA)

Noise variable becomes infinitesimal compared to the scale of the data: \( \Psi = \lim_{\sigma^2 \to 0} \sigma^2 I \)

\[
p(x|y) = \mathcal{N}(\beta y, I - \beta \Lambda)
\]

\[
\beta = \lim_{\sigma^2 \to 0} \Lambda^\top (\Lambda \Lambda^\top + \sigma^2 I)^{-1} = (\Lambda^\top \Lambda)^{-1} \Lambda^\top
\]

Usually in PCA we choose columns of \( \Lambda \) to be orthonormal, i.e. \( \Lambda^\top \Lambda = I \), therefore:

\[
\beta = \Lambda^\top
\]
Eigenvalues and Eigenvectors

\( \lambda \) is an eigenvalue and \( x \) is an eigenvector of \( A \) if:

\[ Ax = \lambda x \]

and \( x \) is a unit vector (\( x^\top x = 1 \)).

**Interpretation:** the operation of \( A \) in direction \( x \) is a scaling by \( \lambda \).

The \( K \) Principal Components are the \( K \) eigenvectors with the largest eigenvalues of the data covariance matrix (i.e. \( K \) directions with the largest variance).

Note: \( \Sigma \) can be decomposed:

\[ \Sigma = USU^\top \]

where \( S \) is \( \text{diag}(\sigma_1^2, \ldots, \sigma_D^2) \) and \( U \) is an orthonormal matrix.
Example of PCA: Eigenfaces

from www-white.media.mit.edu/vismod/demos/facerec/basic.html
Mutual Information and PCA

**Problem:** Given $y$, find $x = Ay$ with columns of $A$ unit vectors, s.t. $I(x; y)$ is maximised (assuming that $P(y)$ is Gaussian).

$$I(x; y) = H(x) + H(y) - H(x, y) = H(x)$$

So we want to maximise the entropy of $x$. What is the entropy of a Gaussian?

$$H(z) = - \int dz \, p(z) \ln p(z) = \frac{1}{2} \ln |\Sigma| + \frac{D}{2}(1 + \ln 2\pi)$$

Therefore we want the distribution of $x$ to have largest volume (i.e. det of covariance matrix).

$$\Sigma_x = A\Sigma_y A^\top = AU S_y U^\top A^\top$$

So, $A$ should be aligned with the columns of $U$ which are associated with the largest eigenvalues (variances).
Gradient Methods for Learning FA

Write down negative log likelihood:

\[ \frac{1}{2} \log |2\pi(\Lambda\Lambda^\top + \Psi)| + \frac{1}{2} y^\top (\Lambda\Lambda^\top + \Psi)^{-1} y \]

Optimise w.r.t. \( \Lambda \) and \( \Psi \) (need matrix calculus) subject to constraints

We will soon see an easier way to learn latent variable models...
Appendix: Source Coding Under a Gaussian Model

Consider coding real valued numbers $x$ under a Gaussian model of the data.

- How many bits should we use for each $x$?

- Clearly we need to limit the precision of our code, otherwise we will need infinitely many bits. Let’s use precision $\Delta$.

- Remember, from Shannon’s source coding theorem.

$$l(x) = -\log P(x) \approx -\log[p(x)\Delta] = -\log p(x) - \log \Delta$$

$$= \frac{(x - \mu)^2}{2\sigma^2} + \frac{1}{2} \log 2\pi + \log \sigma - \log \Delta$$

- Note as $\Delta \Rightarrow 0$ then $l(x) \Rightarrow \infty$.

So we need $l(x)$ bits to code $x$, which grows quadratically with distance from $x$ to $\mu$. 
Appendix: FA vs PCA

- PCA is rotationally invariant; FA is not
- FA is measurement scale invariant; PCA is not
- FA and pPCA define valid probabilistic models; PCA does not