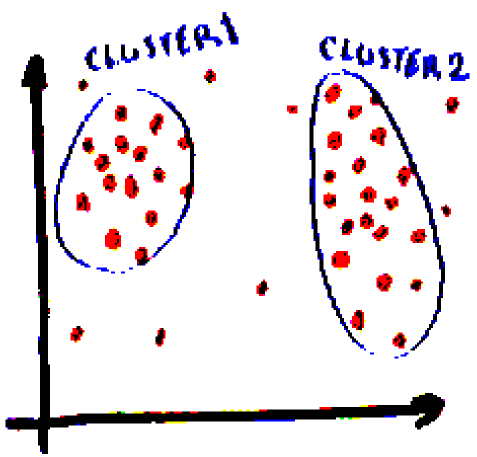
MACHINE LEARNING AND PATTERN RECOGNITION

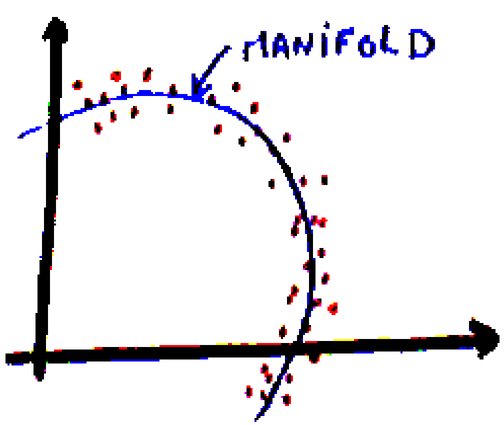
Spring 2005, Lecture 7a: Unsupervised Learning: Density Estimation

Yann LeCun
The Courant Institute,
New York University
http://yann.lecun.com

Unsupervised Learning

The basics idea of unsupervised learning: Learn an energy function $E(Y)$ such that $E(Y)$ is small if Y is "similar" to the training samples, and $E(Y)$ is large if Y is "different" from the training samples. What we mean by "similar" and "different" is somewhat arbitrary and must be defined for each problem.





- Probabilistic unsupervised learning: Density Estimation. Find a function f such $f(Y)$ approximates the empirical probability density of Y, $p(Y)$, as well as possible.
- Clustering: discover "clumps" of points
- Embedding: discover low-dimensional manifold or surface that is as close as possible to all the samples.
- Compression/Quantization: discover a function that for each input computes a compact "code" from which the input can be reconstructed.

Parametric Density Estimation

Use Maximum Likelihood: Given a model $P(Y|W)$, find the parameter W that best "explains" the training samples, i.e. the W that maximizes the likelihood of the training samples $Y^1, Y^2, ...Y^P$. Assuming that the total data likelihood factorizes into individual sample likelihoods:

$$P(Y^1, Y^2, ...Y^P|W) = \prod_i P(Y^i|W)$$

Equivalently, find the W that minimizes the negative log likelihood.

$$L(W) = -log \prod_i P(Y^i|W) = \sum_i -log P(Y^i|W)$$

This is called *parametric* estimation because we assume that the family of possible densities is parameterized by W.

Parametric Density Estimation

Assuming $P(Y|W)$ is the normalized exponential of an energy function:

$$P(Y|W) = \frac{\exp(-\beta E(Y,W))}{\int \exp(-\beta E(Y,W))dY}$$

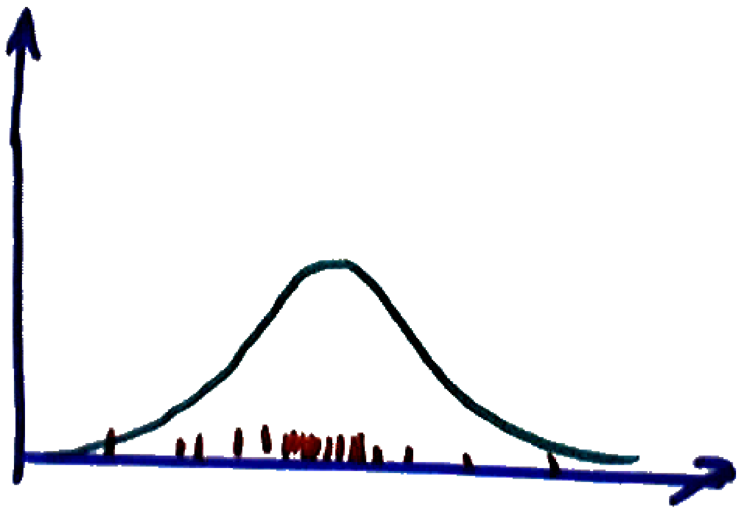
and after an irrelevant division by β, we get the loss function:

$$L(W) = \sum_i \left(E(Y^i, W) + \frac{1}{\beta} \log \int \exp(-\beta E(Y,W))dY \right)$$

The Maximum A Posteriori Estimate is similar but includes a penalty on W:

$$L(W) = \sum_i \left(E(Y^i, W) + \frac{1}{\beta} \log \int \exp(-\beta E(Y,W))dY \right) + H(W)$$

Example: Univariate Gaussian



- Maximum Likelihood: find the parameters of a Gaussian that best "explains" the training samples $y^1, y^2, ...y^P$.
- negative log-likelihood of the data (one dimension): $L(m, v) = -\sum_i \log \frac{1}{\sqrt{2\pi v}} \exp(-\frac{1}{2v}(y^i - m)^2)$

$$L(m, v) = \frac{1}{2} \sum_i \frac{1}{v}(y^i - m)^2 + \log 2\pi v$$

Minimize $L(m, v)$ with respect to m and v.

Example: Univariate Gaussian

- Minimize $L(m, v)$ with respect to m

$$\frac{\partial L(m, v)}{\partial m} = \frac{1}{2} \sum_i \frac{1}{v}(y^i - m) = 0$$

  Hence, $m = \frac{1}{P} \sum_i y^i$

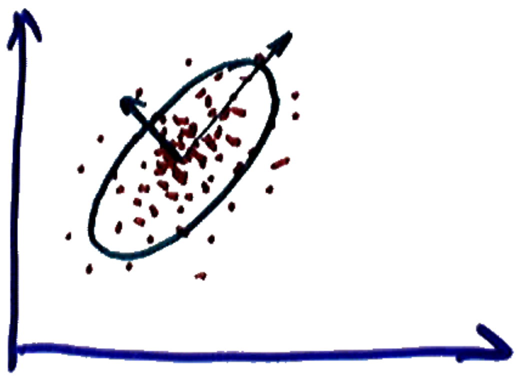
- Now minimize $L(m, v)$ with respect to v

$$\frac{\partial L(m, v)}{\partial v} = \frac{1}{2} \sum_i \left(-\frac{1}{v^2}(y^i - m)^2 + \frac{1}{v} \right) = 0$$

  Hence $v = \frac{1}{P} \sum_i (y^i - m)^2$

- surprise-surprise: The maximum likelihood estimates of the mean and variance of a Gaussian are the mean and variance of the samples.

Example: Multi-variate Gaussian



Maximum Likelihood: find the parameters of a Gaussian that best “explains” the training samples $Y^1, Y^2, ...Y^P$.
The negative log-likelihood of the data (M is a vector, V is a matrix):

$$L(M,V) = -\sum_i \log\left(|2\pi V|^{-1/2} \exp(-1/2(Y^i - M)'V^{-1}(Y^i - M))\right)$$

$$L(M,V) = \frac{1}{2}\sum_i (Y^i - M)'V^{-1}(Y^i - M) - \log|V^{-1}| + \log(2\pi)$$

Multi-variate Gaussian (continued)

$$L(M,V) = \frac{1}{2}\sum_i (Y^i - M)'V^{-1}(Y^i - M) - \log|V^{-1}| + \log(2\pi)$$

$$\frac{\partial L(M,V)}{\partial M} = \frac{1}{2}\sum_i V^{-1}(Y^i - M) = 0$$

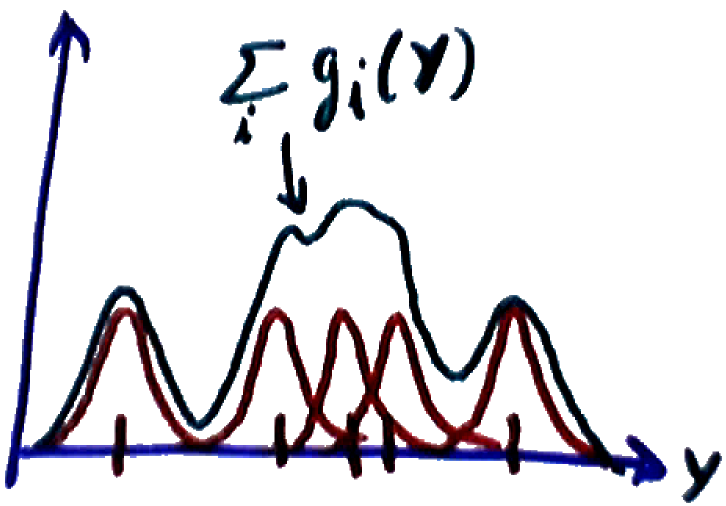
Hence, $M = \frac{1}{P}\sum_i Y^i$ Now minimize $L(M,V)$ with respect to V^{-1}

$$\frac{\partial L(M,V)}{\partial V^{-1}} = \frac{1}{2}\sum_i \left((Y^i - M)(Y^i - M)' - V\right)$$

(using the fact $\frac{\partial \log|V^{-1}|}{\partial V^{-1}} = V'$).

Hence $V = \frac{1}{P}\sum_i (Y^i - M)(Y^i - M)'$

Non-Parametric Methods: Parzen Windows



- The sample distribution can be seen as a bunch of delta functions. Idea: make it smooth.
- Place a “bump” around each training sample Y^i.
- example: Gaussian bump $g_i(Y) = \frac{1}{Z} \exp(-K||Y - Y^i||^2)$ where Z is the Gaussian normalization constant.
- The density is $P(Y) = \frac{1}{P} \sim_{i=1}^{P} g_i(Y)$
- It’s simple, but it’s expensive.