Maximum likelihood estimation
(of the parameters of class conditional probability)

Problem formulation

- To design an optimal Bayesian classifier we need priors $P(\omega_j)$ and class-conditional probabilities $p(\mathrm{x}|\omega_j)$
- In practice, they are usually not available
- Available is some (hopefully representative) data
- Problem: how to design a classifier using this training data?
- Priors are easier to estimate $P(\omega_j)$
- Specific problem: estimation of class-conditional probabilities $p(\mathrm{x}|\omega$
- Simplification: estimation of the parameters of a function of known type, e.g. μ_i and Σ_i of normal density

Purpose of MLE

Assume a **data set** $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$ **of *n* feature vectors from class ω**

Assume that $p(x|\omega)$ **has a known parametric form, such as** $p(x \mid \omega) \sim N(\mu, \Sigma)$

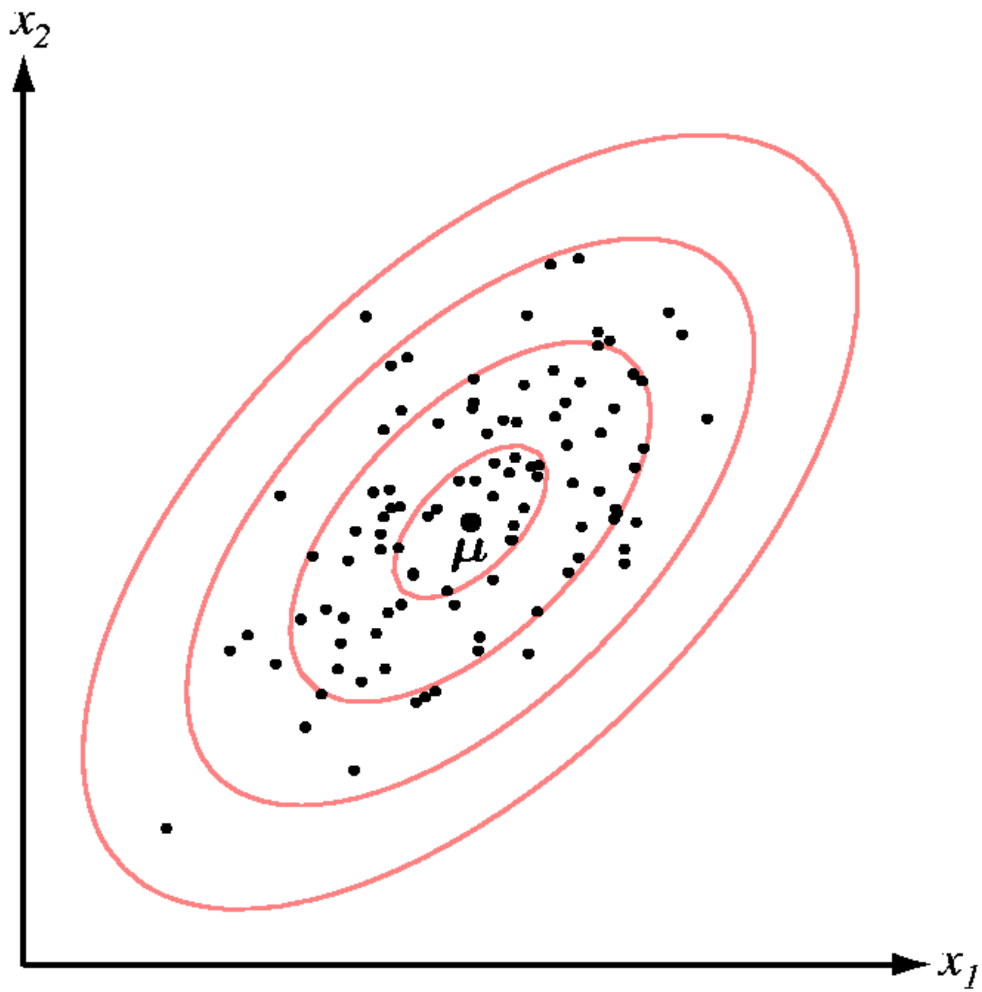
Denote by θ the parameters of the distribution, e.g. $\theta = [\mu, \Sigma]$

Hence, we know the form of $p(x \mid \theta)$

but we do not know the values of the parameters θ

The purpose of MLE is to estimate the values of the parameters θ

using the observed data $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$



A hyper-ellipsoidal cluster formed by points drawn from a population which has normal density. What are the parameters of this normal density?

from Duda, Hart, Stork (2001) Pattern classification

The maximum likelihood estimate $\hat{\theta}$ **is the value of** θ **that maximizes** $p(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{n} \mid \theta)$

For analytical purposes, we use the logarithm of the likelihood, called log-likelihood

$$l(\theta) \equiv \ln p(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{n} \mid \theta)$$

The solution is the value $\hat{\theta}$ of the argument that maximizes the log-likelihood:

$$\hat{\theta} = \arg\max_{\theta} l(\theta)$$

What to do?

The events x_i, i = 1 … n, are statistically independent, i.e.

$$p(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{n} \mid \theta) = \prod_{k=1}^{n} p(\mathrm{x}_\mathrm{k} \mid \theta)$$

or

$$l(\theta) = \sum_{k=1}^{n} \ln p(\mathrm{x}_k \mid \theta)$$

Denote by $\nabla_\theta = \begin{bmatrix} \dfrac{\partial}{\partial \theta_1} \\ \ldots \\ \dfrac{\partial}{\partial \theta_p} \end{bmatrix}$ the gradient operator.

What to do?

The gradient of the log-likelihood is:

$$\nabla_\theta l = \sum_{k=1}^{n} \nabla_\theta \ln p(\mathrm{x}_k \mid \theta)$$

A set of necessary conditions for $\hat{\theta}$ can be formulated as:

$$\nabla_\theta l = 0$$

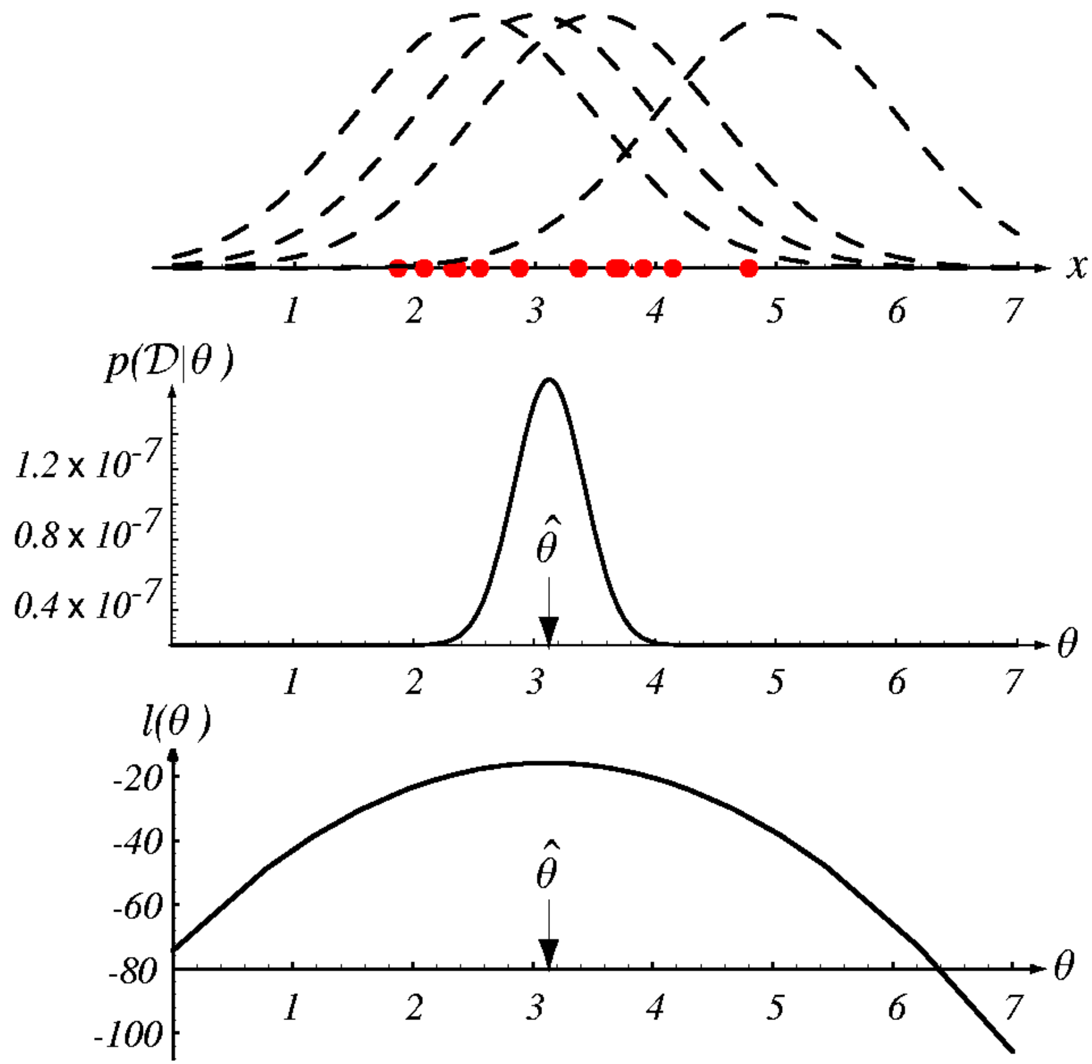
Solve this (system of) equation(s)!

(The solutions of this equation can be *global, local* maxima or minima, or saddle points. Don't forget to check that it is a maximum!)

Example

Model: Gaussian with fixed variance

Estimated parameter: mean



from Duda, Hart, Stork (2001) Pattern classification

ML Estimation for normal distribution – unknown mean μ

Recall that: $\ln p(\mathrm{x}_k \mid \mu) = -\frac{1}{2}\ln[(2\pi)^d \mid \Sigma \mid] - \frac{1}{2}(\mathrm{x}_k - \mu)^t \Sigma^{-1}(\mathrm{x}_k - \mu)$

It follows: $\nabla_\mu \ln p(\mathrm{x}_k \mid \mu) = \Sigma^{-1}(\mathrm{x}_k - \mu)$

Solve:

$$\sum_{k=1}^{n} \Sigma^{-1}(\mathrm{x}_k - \mu) = 0$$

$$\Rightarrow \quad \hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} \mathrm{x}_k$$

In plain text: the best estimate of the mean of a distribution is the mean of the sample!

This kind of pedantic, algebra-filled and ultimately unsurprising fact is exactly the reason people throw down their "Statistics" book and pick up their "Agent Based Evolutionary Data Mining Using The Neuro-Fuzzy Transform" book. (from slides on MLE by Andrew W. Moore)

ML Estimation for normal distribution – unknown μ and Σ

In the univariate case, $\theta_1 = \mu; \theta_2 = \sigma^2$

$$\ln p(\mathrm{x}_k \mid \theta) = -\frac{1}{2}\ln 2\pi\theta_2 - \frac{1}{2\theta_2}(\mathrm{x}_k - \theta_1)^2$$

The gradient is:

$$\nabla_\theta l = \nabla_\theta \ln p(\mathrm{x}_k \mid \theta) = \begin{bmatrix} \dfrac{1}{\theta_2}(x_k - \theta_1) \\ -\dfrac{1}{2\theta_2} + \dfrac{(x_k - \theta_1)^2}{2\theta_2{}^2} \end{bmatrix}$$

Imposing $\nabla_\theta l = 0$:

$$\sum_{k=1}^{n} \frac{1}{\hat{\theta}_2}(x_k - \hat{\theta}_1) = 0$$

$$-\sum_{k=1}^{n} \frac{1}{\hat{\theta}_2} + \sum_{i=1}^{n} \frac{(x_k - \hat{\theta}_1)^2}{\hat{\theta}_2^{\;2}} = 0$$

Substituting $\hat{\mu} = \hat{\theta}_1, \hat{\sigma}^2 = \hat{\theta}_2$ and rearranging:

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k$$

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k - \hat{\mu})^2$$

MLE for normal distribution - multidimensional case

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} \mathrm{x}^{(\mathrm{k})}$$

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n} (\mathrm{x}^{(\mathrm{k})} - \hat{\mu})(\mathrm{x}^{(\mathrm{k})} - \hat{\mu})^{t}$$

$$\hat{\Sigma}_{ij} = \frac{1}{n}\sum_{k=1}^{n} (x_i^{(k)} - \hat{\mu}_i)(x_j^{(k)} - \hat{\mu}_j)$$

where $x_i^{(k)}$ is the *i*-th feature of the k-th feature vector $\mathrm{x}^{(\mathrm{k})}$
and $\hat{\mu}_i$ is the *i*-th feature of the mean $\hat{\mu}$ of all *n* feature vectors

ML Estimation

The ML estimate for the variance σ^2 is *biased*, i.e. the expected value over all possible (random!) data sets of size n is not equal to the true variance:

$$\varepsilon\left[\frac{1}{n}\sum_{k=1}^{n}(\mathrm{x}_k - \bar{x})^2\right] = \frac{n-1}{n}\sigma^2 \neq \sigma^2$$

… but it is *asymptotically unbiased* (for large *n*)

ML Estimation

An unbiased estimator for Σ is the *sample covariance matrix*:

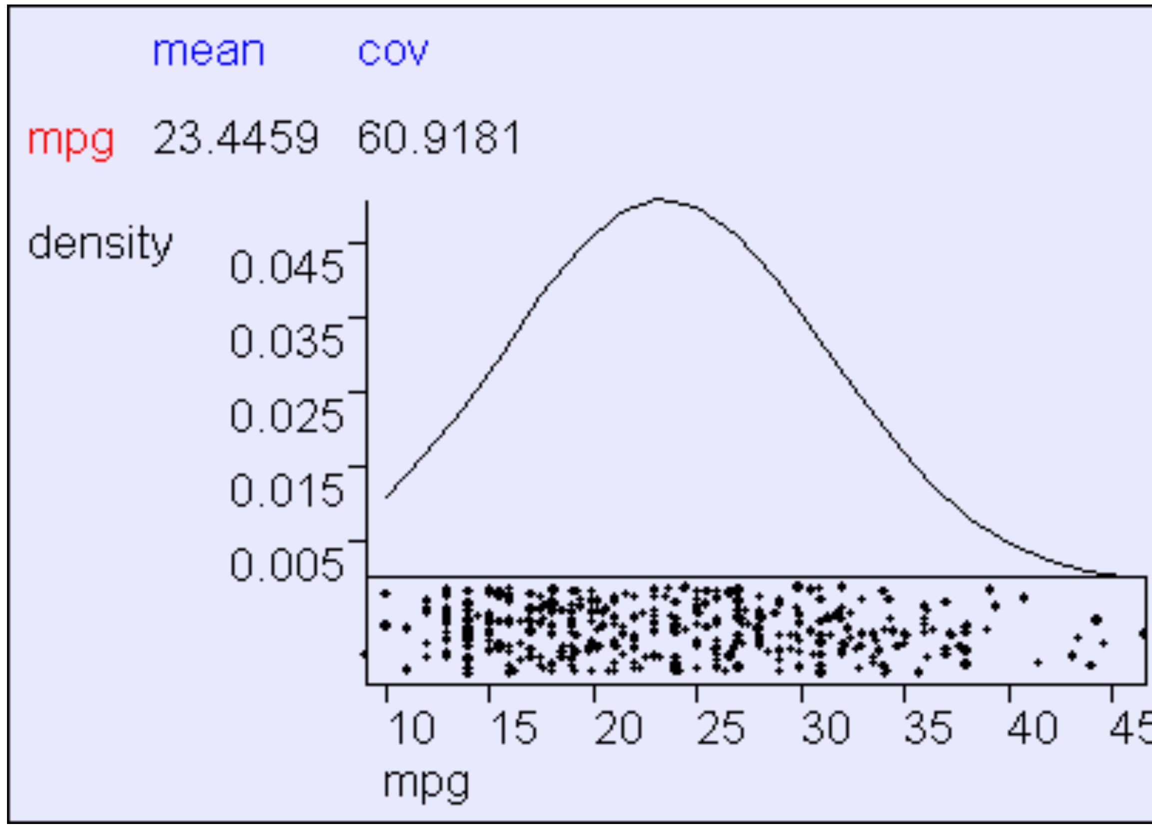
$$C = \frac{1}{n-1} \sum_{k=1}^{n} (\mathrm{x}_k - \hat{\mu})(\mathrm{x}_k - \hat{\mu})^t$$

An *asymptotically (i.e. large n) unbiased estimator* for Σ is:

$$C = \frac{1}{n} \sum_{k=1}^{n} (\mathrm{x}_k - \hat{\mu})(\mathrm{x}_k - \hat{\mu})^t$$

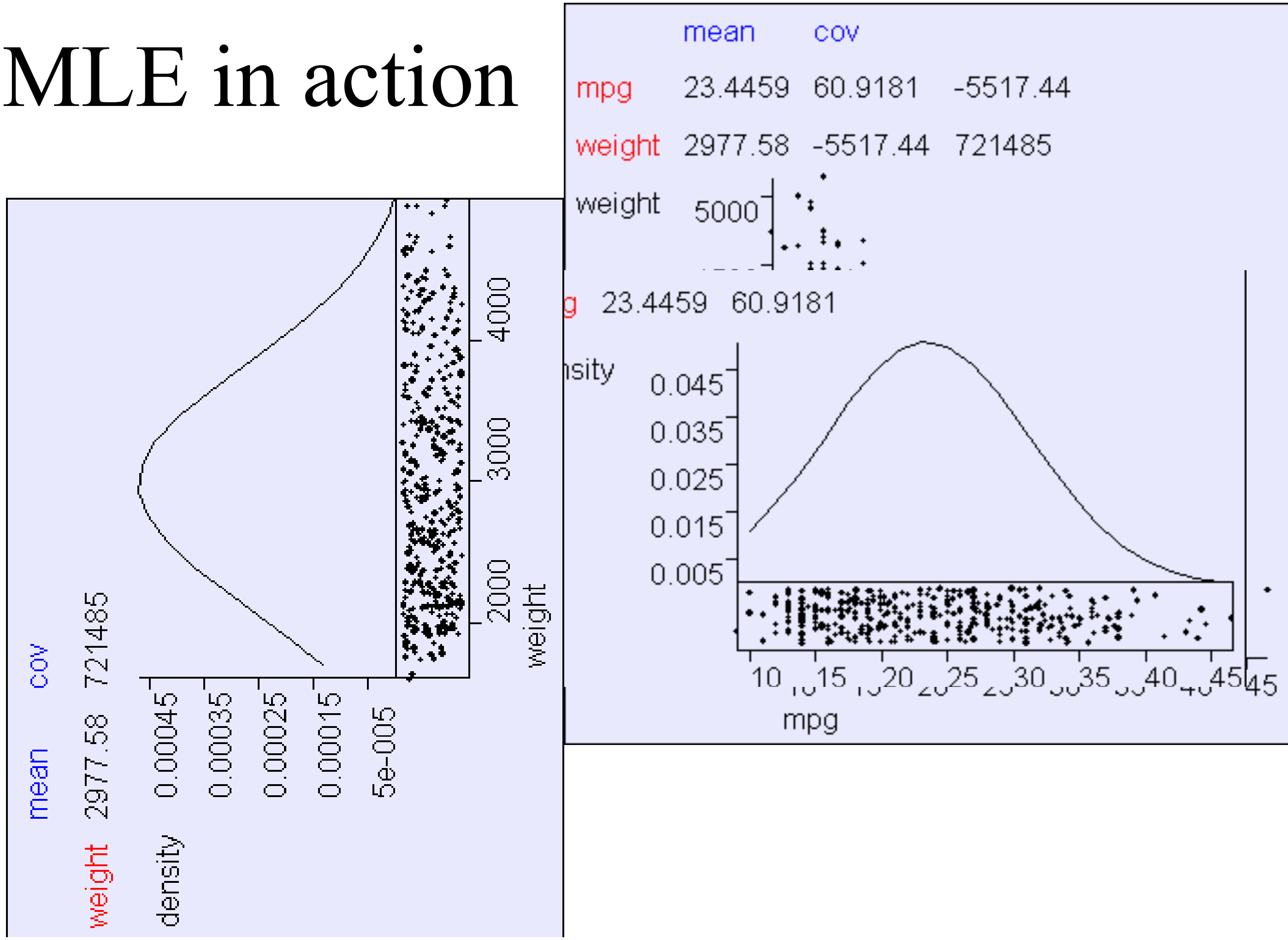
Gaussian MLE in action

Using n=392 cars from the "MPG" UCI dataset supplied by Ross Quinlan



Copyright © 2001, 2004, Andrew W. Moore

Bivariate MLE in action



Copyright © 2001, 2004, Andrew W. Moore

Multivariate MLE

	mean	cov						
mpg	23.4459	60.9181	-10.3529	-657.585	-233.858	-5517.44	9.11551	16.6915
cylinders	5.47194	-10.3529	2.9097	169.722	55.3482	1300.42	-2.37505	-2.17193
displacement	194.412	-657.585	169.722	10950.4	3614.03	82929.1	-156.994	-142.572
horsepower	104.469	-233.858	55.3482	3614.03	1481.57	28265.6	-73.187	-59.0364
weight	2977.58	-5517.44	1300.42	82929.1	28265.6	721485	-976.815	-967.228
acceleration	15.5413	9.11551	-2.37505	-156.994	-73.187	-976.815	7.61133	2.95046
modelyear	75.9796	16.6915	-2.17193	-142.572	-59.0364	-967.228	2.95046	13.5699

Covariance matrices are not exciting to look at

Copyright © 2001, 2004, Andrew W. Moore