Information-Theoretic Modeling

Lecture 10: MDL Principle — Part II

Jyrki Kivinen

Department of Computer Science, University of Helsinki

Autumn 2012
1 MDL for Gaussian Models
- Encoding Continuous Data
- Differential Entropy
- Linear Regression
- Subset Selection Problem
- Wavelet Denoising

Jyrki Kivinen
Information-Theoretic Modeling
1. MDL for Gaussian Models
   - Encoding Continuous Data
   - Differential Entropy
   - Linear Regression
   - Subset Selection Problem
   - Wavelet Denoising

2. MDL for Multinomial Models
   - Universal Codes
   - Fast NML Computation
   - Histogram Density Estimation
   - Clustering
Gaussian models

The graphs show several Gaussian distributions with different parameters. Here are the details:

- Blue line: \( \mu = 0, \sigma^2 = 0.2 \)
- Red line: \( \mu = 0, \sigma^2 = 1.0 \)
- Yellow line: \( \mu = 0, \sigma^2 = 5.0 \)
- Green line: \( \mu = -2, \sigma^2 = 0.5 \)

The x-axis represents the variable \( X \), and the y-axis represents the probability density \( \phi_{\mu,\sigma^2}(x) \).

Gaussian models

Density function:

\[
\phi_{\mu,\sigma^2}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x - \mu)^2}{2\sigma^2}}.
\]
Gaussian models

Density function:

\[
\phi_{\mu,\sigma^2}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x - \mu)^2}{2\sigma^2}}.
\]

Mean: \( \mu = E[X] \), variance \( \sigma^2 = E[(X - \mu)^2] \)
Gaussian models

Density function:

\[ \phi_{\mu,\sigma^2}(x_1, \ldots, x_n) \overset{(i.i.d.)}{=} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}}. \]

Mean: \( \mu = E[X] \), variance \( \sigma^2 = E[(X - \mu)^2] \)
Gaussian models

Density function:

\[
\phi_{\mu,\sigma^2}(x_1, \ldots, x_n) \overset{(i.i.d.)}{=} \left( \frac{1}{\sqrt{2\pi\sigma^2}} \right)^n \prod_{i=1}^{n} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}}.
\]

Mean: \( \mu = E[X] \), variance \( \sigma^2 = E[(X - \mu)^2] \)
Gaussian models

Density function:

\[ \phi_{\mu,\sigma^2}(x_1, \ldots, x_n) \overset{(i.i.d.)}{=} \left( \frac{1}{2\pi\sigma^2} \right)^{n/2} \prod_{i=1}^{n} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}}. \]

Mean: \( \mu = E[X] \), variance \( \sigma^2 = E[(X - \mu)^2] \)
Gaussian models

Density function:

\[ \phi_{\mu, \sigma^2}(x_1, \ldots, x_n)^{\text{(i.i.d.)}} = \left(2\pi\sigma^2\right)^{-n/2} \prod_{i=1}^{n} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}}. \]

Mean: \( \mu = E[X]\), variance \( \sigma^2 = E[(X - \mu)^2] \)
Gaussian models

Density function:

\[ \phi_{\mu,\sigma^2}(x_1, \ldots, x_n) \overset{(i.i.d.)}{=} \left(2\pi \sigma^2 \right)^{-n/2} e^{-\frac{\sum_{i=1}^{n}(x_i - \mu)^2}{2\sigma^2}}. \]

Mean: \( \mu = E[X] \), variance \( \sigma^2 = E[(X - \mu)^2] \).
Gaussian models

Density function:

\[ \phi_{\mu,\sigma^2}(x_1, \ldots, x_n)^{(i.i.d.)} = (2\pi\sigma^2)^{-n/2} e^{-\frac{\sum_{i=1}^{n}(x_i - \mu)^2}{2\sigma^2}}. \]

Mean: \( \mu = E[X], \) variance \( \sigma^2 = E[(X - \mu)^2] \)

Maximum likelihood: \( \hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i, \) \( \hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n}(x_i - \hat{\mu})^2. \)
How to Encode Continuous Data?

In order to encode data using, say, the Gaussian density we face the problem of **How to encode continuous data?**
How to Encode Continuous Data?

In order to encode data using, say, the Gaussian density we face the problem of **How to encode continuous data?**

We already know how to encode using models with continuous parameters:
How to Encode Continuous Data?

In order to encode data using, say, the Gaussian density we face the problem of **How to encode continuous data?**

We already know how to encode using models with continuous parameters:

- two-part with optimal quantization \( \approx \frac{k}{2} \log_2 n \),
How to Encode Continuous Data?

In order to encode data using, say, the Gaussian density we face the problem of **How to encode continuous data?**

We already know how to encode using models with continuous *parameters*:

- two-part with optimal quantization \( \approx \frac{k}{2} \log_2 n \),
- mixture code,
How to Encode Continuous Data?

In order to encode data using, say, the Gaussian density we face the problem of **How to encode continuous data?**

We already know how to encode using models with continuous parameters:

- two-part with optimal quantization \( \approx \frac{k}{2} \log_2 n \),
- mixture code,
- NML.
How to Encode Continuous Data?

In order to encode data using, say, the Gaussian density we face the problem of **How to encode continuous data?**

We already know how to encode using models with continuous parameters:

- two-part with optimal quantization \((\approx \frac{k}{2} \log_2 n)\),
- mixture code,
- NML.

Obviously not possible to encode data with infinite precision. Have to **discretize**: encode \(x\) only up to precision \(\delta\).
What is the optimal rate for encoding (compressing) continuous data (up to precision $\delta$)?
Differential Entropy

What is the optimal rate for encoding (compressing) continuous data (up to precision $\delta$)?

The answer involves again an entropy. However, not the familiar kind of entropy but instead...
Differential Entropy

What is the optimal rate for encoding (compressing) continuous data (up to precision $\delta$)?

The answer involves again an entropy. However, not the familiar kind of entropy but instead...

**Differential entropy**
Differential Entropy

What is the optimal rate for encoding (compressing) continuous data (up to precision $\delta$)?

The answer involves again an entropy. However, not the familiar kind of entropy but instead...

**Differential entropy**

Let $X \in \mathbb{R}$ be a continuous random variable with probability density $f : \mathbb{R} \to \mathbb{R}^+$. 
Differential Entropy

What is the optimal rate for encoding (compressing) continuous data (up to precision $\delta$)?

The answer involves again an entropy. However, not the familiar kind of entropy but instead...

**Differential entropy**

Let $X \in \mathbb{R}$ be a continuous random variable with probability density $f : \mathbb{R} \rightarrow \mathbb{R}^+$. The differential entropy of $X$ is defined as

$$h(X) = E_{X \sim f} \left[ \log_2 \frac{1}{f(X)} \right] = \int f(x) \log_2 \frac{1}{f(x)} \, dx.$$
If $\delta > 0$ is small, the probability that $X \in [(t - \frac{1}{2})\delta, (t + \frac{1}{2})\delta]$ is well approximated by $f(t\delta)\delta$. 
Differential Entropy

If $\delta > 0$ is small, the probability that $X \in [(t - \frac{1}{2})\delta, (t + \frac{1}{2})\delta]$ is well approximated by $f(t\delta)\delta$.

Hence, the minimum coding rate of the discretized random variable $X^\delta$ is given by

$$H(X^\delta) \approx \sum_{x=t\delta : t \in \mathbb{Z}} f(x)\delta \log_2 \frac{1}{f(x)\delta}$$
Differential Entropy

If $\delta > 0$ is small, the probability that $X \in [(t - \frac{1}{2})\delta, (t + \frac{1}{2})\delta]$ is well approximated by $f(t\delta)\delta$.

Hence, the minimum coding rate of the discretized random variable $X^\delta$ is given by

$$H(X^\delta) \approx \sum_{x = t\delta : t \in \mathbb{Z}} f(x)\delta \log_2 \frac{1}{f(x)\delta} \rightarrow 0 \int_{-\infty}^{+\infty} f(x) \log_2 \frac{1}{f(x)\delta} \, dx.$$
If $\delta > 0$ is small, the probability that $X \in [(t - \frac{1}{2})\delta, (t + \frac{1}{2})\delta]$ is well approximated by $f(t\delta)\delta$.

Hence, the minimum coding rate of the discretized random variable $X^\delta$ is given by

$$H(X^\delta) \approx \sum_{x = t\delta : t \in \mathbb{Z}} f(x)\delta \log_2 \frac{1}{f(x)\delta}$$

$$\overset{\delta \to 0}{\longrightarrow} \int_{-\infty}^{+\infty} f(x) \log_2 \frac{1}{f(x)} \, dx - \log_2 \delta.$$
If $\delta > 0$ is small, the probability that $X \in [(t - \frac{1}{2})\delta, (t + \frac{1}{2})\delta]$ is well approximated by $f(t\delta)\delta$.

Hence, the minimum coding rate of the discretized random variable $X^\delta$ is given by

$$H(X^\delta) \approx \sum_{x = t\delta : t \in \mathbb{Z}} f(x)\delta \log_2 \frac{1}{f(x)\delta}$$

$$\xrightarrow[\delta \to 0]{} \int_{-\infty}^{+\infty} f(x) \log_2 \frac{1}{f(x)} \, dx - \log_2 \delta.$$ 

Hence, the rate is approximately $H(X^\delta) \approx h(X) - \log_2 \delta$. 

\text{Information-Theoretic Modeling}
Differential Entropy

The minimum coding rate \( h(X) - \log_2 \delta \) is achieved if and only if the code-word lengths are chosen according to

\[
\ell(x) = \log_2 \frac{1}{f(x) \delta} = \log_2 \frac{1}{f(x)} + \log_2 \frac{1}{\delta}.
\]
The minimum coding rate \( h(X) - \log_2 \delta \) is achieved if and only if the code-word lengths are chosen according to

\[
\ell(x) = \log_2 \frac{1}{f(x)\delta} = \log_2 \frac{1}{f(x)} + \log_2 \frac{1}{\delta}.
\]

The term \( \log_2(1/\delta) \) depends only on the precision we chose and is same for all models. Therefore, we can ignore it for the purpose of comparing models.
The minimum coding rate $h(X)$ is achieved if and only if the code-word lengths are chosen according to

$$\ell(x) = \log_2 \frac{1}{f(x)} = \log_2 \frac{1}{f(x)}.$$ 

The term $\log_2(1/\delta)$ depends only on the precision we chose and is same for all models. Therefore, we can ignore it for the purpose of comparing models.
Recall the Gaussian density function:

\[
\phi_{\mu, \sigma^2}(x_1, \ldots, x_n) \overset{(i.i.d.)}{=} \left(2\pi\sigma^2\right)^{-n/2} e^{-\frac{\sum_{i=1}^{n}(x_i - \mu)^2}{2\sigma^2}}.
\]
Recall the Gaussian density function:

\[ \phi_{\mu,\sigma^2}(x_1, \ldots, x_n) \overset{(i.i.d.)}{=} (2\pi\sigma^2)^{-n/2} e^{-\frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)^2}. \]

The code-length is then

\[ \frac{n}{2} \log_2(2\pi\sigma^2) - \frac{1}{(2\ln 2)\sigma^2} \sum_{i=1}^{n}(x_i - \mu)^2. \]
Ok, we have our Gaussian code-length formula:

$$\frac{n}{2} \log_2(2\pi\sigma^2) - \frac{1}{(2\ln 2)\sigma^2} \sum_{i=1}^{n} (x_i - \mu)^2.$$
Back to Gaussians

Ok, we have our Gaussian code-length formula:

$$\frac{n}{2} \log_2(2\pi \sigma^2) - \frac{1}{(2 \ln 2)\sigma^2} \sum_{i=1}^{n} (x_i - \mu)^2.$$ 

Let’s use the two-part code and plug in the maximum likelihood parameters:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i, \quad \hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu})^2.$$
Ok, we have our Gaussian code-length formula:

\[
\frac{n}{2} \log_2(2\pi \hat{\sigma}^2) - \frac{1}{(2 \ln 2)\hat{\sigma}^2} \sum_{i=1}^{n} (x_i - \hat{\mu})^2.
\]

Let’s use the two-part code and plug in the maximum likelihood parameters:

\[
\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i, \quad \hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu})^2.
\]
Back to Gaussians

Ok, we have our Gaussian code-length formula:

\[
\frac{n}{2} \log_2(2\pi\hat{\sigma}^2) - \frac{1}{(2 \ln 2)\hat{\sigma}^2} \sum_{i=1}^{n} (x_i - \hat{\mu})^2.
\]

Let’s use the two-part code and plug in the maximum likelihood parameters:

\[
\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i, \quad \hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu})^2.
\]
Ok, we have our Gaussian code-length formula:

$$\frac{n}{2} \log_2(2\pi\hat{\sigma}^2) - \frac{n}{2 \ln 2}.$$ 

Let’s use the two-part code and plug in the maximum likelihood parameters:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i, \quad \hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu})^2.$$
Ok, we have our Gaussian code-length formula:

$$\frac{n}{2} \log_2 \hat{\sigma}^2 + \text{constant}.$$ 

Let’s use the two-part code and plug in the maximum likelihood parameters:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i, \quad \hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu})^2.$$
We get the total (two-part) code-length formula:

\[ \frac{n}{2} \log_2 \hat{\sigma}^2 + \frac{k}{2} \log_2 n + \text{constant}. \]
We get the total (two-part) code-length formula:

\[ \frac{n}{2} \log_2 \hat{\sigma}^2 + \frac{k}{2} \log_2 n + \text{constant}. \]

Since we have two parameters, \( \mu \) and \( \sigma^2 \), we let \( k = 2 \).
We get the total (two-part) code-length formula:

$$\frac{n}{2} \log_2 \hat{\sigma}^2 + \frac{2}{2} \log_2 n + constant.$$ 

Since we have two parameters, $\mu$ and $\sigma^2$, we let $k = 2$.

Notice that depending on what exactly you are doing, you may or may not care about the constant.
Linear Regression

A similar treatment can be given to linear regression models.
Linear Regression

A similar treatment can be given to *linear regression models*.

The model includes a set of *regressor variables* $x_1, \ldots, x_p \in \mathbb{R}$, and a set of *coefficients* $\beta_1, \ldots, \beta_p$. 
Linear Regression

A similar treatment can be given to linear regression models.

The model includes a set of regressor variables $x_1, \ldots, x_p \in \mathbb{R}$, and a set of coefficients $\beta_1, \ldots, \beta_p$.

The dependent variable, $Y$, is assumed to be Gaussian:

- the mean $\mu$ is given as a linear combination of the regressors:

$$\mu = \beta_1 x_1 + \cdots + \beta_p x_p = \beta^T x,$$

- variance is some parameter $\sigma^2$. 
Linear Regression

For a sample of size $n$, the matrix notation is convenient:

\[
Y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix} \quad X = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{pmatrix} \quad \beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} \quad \epsilon = \begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{pmatrix}
\]
Linear Regression

For a sample of size $n$, the matrix notation is convenient:

$$
Y = \begin{pmatrix}
Y_1 \\
\vdots \\
Y_n
\end{pmatrix} \quad X = \begin{pmatrix}
x_{11} & \cdots & x_{1p} \\
\vdots & \ddots & \vdots \\
x_{n1} & \cdots & x_{np}
\end{pmatrix} \quad \beta = \begin{pmatrix}
\beta_1 \\
\vdots \\
\beta_p
\end{pmatrix} \quad \epsilon = \begin{pmatrix}
\epsilon_1 \\
\vdots \\
\epsilon_n
\end{pmatrix}
$$

Then the model can be written as

$$
Y = X\beta + \epsilon,
$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. 
Linear Regression

The maximum likelihood estimators are now

\[
\hat{\beta} = (X^T X)^{-1} X^T Y, \quad \hat{\sigma}^2 = \frac{1}{n} \| Y - X\hat{\beta} \|^2 = \frac{\text{RSS}}{n},
\]

where \( \text{RSS} \) is the “residual sum of squares”.
Linear Regression

The maximum likelihood estimators are now

\[
\hat{\beta} = (X^T X)^{-1} X^T Y, \quad \hat{\sigma}^2 = \frac{1}{n} \| Y - X \hat{\beta} \|_2^2 = \frac{\text{RSS}}{n},
\]

where RSS is the “residual sum of squares”.

Since the errors are assumed Gaussian, our code-length formula applies:

\[
\frac{n}{2} \log_2 \hat{\sigma}^2 + \frac{k}{2} \log_2 n + \text{constant}.
\]
Linear Regression

The maximum likelihood estimators are now

$$\hat{\beta} = (X^T X)^{-1} X^T Y, \quad \hat{\sigma}^2 = \frac{1}{n} \| Y - X \hat{\beta} \|_2^2 = \frac{\text{RSS}}{n},$$

where RSS is the “residual sum of squares”.

Since the errors are assumed Gaussian, our code-length formula applies:

$$\frac{n}{2} \log_2 \text{RSS} + \frac{k}{2} \log_2 n + \text{constant}.$$
Linear Regression

The maximum likelihood estimators are now

$$
\hat{\beta} = \left( X^T X \right)^{-1} X^T Y, \quad \hat{\sigma}^2 = \frac{1}{n} \| Y - X \hat{\beta} \|^2_2 = \frac{\text{RSS}}{n},
$$

where RSS is the “residual sum of squares”.

Since the errors are assumed Gaussian, our code-length formula applies:

$$
\frac{n}{2} \log_2 \text{RSS} + \frac{k}{2} \log_2 n + \text{constant}.
$$

The number of parameters is now $p + 1$ ($p$ of the $\beta$s and $\sigma^2$), so we get...
Linear Regression

The maximum likelihood estimators are now

$$\hat{\beta} = (X^T X)^{-1} X^T Y, \quad \hat{\sigma}^2 = \frac{1}{n} \| Y - X \hat{\beta} \|^2_2 = \frac{RSS}{n},$$

where $RSS$ is the “residual sum of squares”.

Since the errors are assumed Gaussian, our code-length formula applies:

$$\frac{n}{2} \log_2 RSS + \frac{p + 1}{2} \log_2 n + \text{constant}.$$  

The number of parameters is now $p + 1$ ($p$ of the $\beta$s and $\sigma^2$), so we get...
Subset Selection Problem

Often we have a large set of potential regressors, some of which may be irrelevant.
Often we have a large set of potential regressors, some of which may be irrelevant.

The MDL principle can be used to select a subset of them by comparing the total code-lengths:

$$\min_S \left[ \frac{n}{2} \log_2 \text{RSS}_S + \frac{|S| + 1}{2} \log_2 n \right],$$

where $\text{RSS}_S$ is the RSS obtained by using subset $S$ of the regressors.
Subset Selection Problem

Often we have a large set of potential regressors, some of which may be irrelevant.

The MDL principle can be used to select a subset of them by comparing the total code-lengths:

$$\min_S \left[ \frac{n}{2} \log_2 \text{RSS}_S + \frac{|S| + 1}{2} \log_2 n \right],$$

where $\text{RSS}_S$ is the RSS obtained by using subset $S$ of the regressors.

⇒ Exercise
Wavelet Denoising

One particularly useful way to obtain the regressor (design) matrix is to use **wavelets**.
Wavelet Denoising

One particularly useful way to obtain the regressor (design) matrix is to use wavelets.

Image by Gabriel Peyré
MDL Denoising Revisited

Teemu Roos Member, Petri Myllymäki, and Jorma Rissanen Fellow

Abstract—We refine and extend an earlier minimum description length (MDL) denoising criterion for wavelet-based denoising. We start by showing that the denoising problem can be reformulated as a clustering problem, where the goal is to obtain separate clusters for informative and non-informative wavelet coefficients, respectively. This suggests two refinements, adding a code-length for the model index, and extending the model in order to account for subband-dependent coefficient distributions. A third refinement is the derivation of soft thresholding inspired by predictive universal coding with weighted mixtures. We propose a practical method incorporating all three refinements, which is shown to achieve good performance and robustness in denoising both artificial and natural signals.

Index Terms—Minimum description length (MDL) principle, wavelets, denoising.

(both of which include the Gaussian and densities as special cases).

A third approach to denoising is based description length (MDL) principle [16]–[20]ent MDL denoising methods have been suggested [21]–[25]. We focus on what we consider the MDL approach, namely that of Rissanen [24] is two-fold: First, as an immediate result extending the earlier MDL denoising meth new practical method with greatly improved and robustness. Secondly, the denoising pro to illustrate theoretical issues related to the involving the problem of unbounded para and the necessity of encoding the model cl
Wavelet Denoising

Main effort in constructing a universal code:
Wavelet Denoising

Main effort in constructing a universal code:

1. combines two-part, mixture, and NML universal codes,
Main effort in constructing a universal code:

1. combines two-part, mixture, and NML universal codes,
2. bounds on NML normalization region required,
Wavelet Denoising

Main effort in constructing a universal code:

1. combines two-part, mixture, and NML universal codes,
2. bounds on NML normalization region required,
3. important lesson: remember to encode model class.
Wavelet Denoising

- Original
- Noisy
- (Rissanen, 2000)
- MDL (A)
- MDL (A-B)
- MDL (A-B-C)
- VisuShrink
- SureShrink
- BayesShrink
Wavelet Denoising
1. MDL for Gaussian Models
   - Encoding Continuous Data
   - Differential Entropy
   - Linear Regression
   - Subset Selection Problem
   - Wavelet Denoising

2. MDL for Multinomial Models
   - Universal Codes
   - Fast NML Computation
   - Histogram Density Estimation
   - Clustering
The multinomial model — the generalization of Bernoulli — is very simple:

\[ p(x_j) = \theta_j, \quad \text{for } j \in \{1, \ldots, m\}. \]
The multinomial model — the generalization of Bernoulli — is very simple:

\[ p(x_j) = \theta_j, \quad \text{for } j \in \{1, \ldots, m\}. \]

Maximum likelihood:

\[ \hat{\theta}_j = \frac{\#\{x_i = j\}}{n}. \]
The multinomial model — the generalization of Bernoulli — is very simple:

\[ p(x_j) = \theta_j, \quad \text{for } j \in \{1, \ldots, m\}. \]

Maximum likelihood:

\[ \hat{\theta}_j = \frac{\#\{x_i = j\}}{n}. \]

Two-part, mixture, and NML models readily defined.
The multinomial model — the generalization of Bernoulli — is very simple:

\[ p(x_j) = \theta_j, \quad \text{for } j \in \{1, \ldots, m\}. \]

Maximum likelihood:

\[ \hat{\theta}_j = \frac{\#\{x_i = j\}}{n}. \]

Two-part, mixture, and NML models readily defined.

⇒ Exercises 5.1 & 5.2
Fast NML for Multinomials

The naïve way to compute the normalizing constant in the NML model

\[
p_{\hat{\theta}}(x^n) \div C^m_n, \quad C^m_n = \sum_{y^n \in X^n} p_{\hat{\theta}}(y^n),
\]

takes exponential time (\(\Omega(m^n)\)).
Fast NML for Multinomials

The naïve way to compute the normalizing constant in the NML model

\[
p_{\hat{\theta}}(x^n) / C_n^m, \quad C_n^m = \sum_{y^n \in \mathcal{X}^n} p_{\hat{\theta}}(y^n),
\]

takes exponential time (\(\Omega(m^n)\)).

The second most naïve way takes “only” polynomial time, \(O(n^{m-1})\), but is still intractable unless \(m \leq 3\) (or maybe \(m \leq 4\)).
Fast NML for Multinomials

There is a way — which is not naïve at all! — to do it in linear time, $O(n + m)$, using the following recursion:

$$C_n^m = C_n^{m-1} + \frac{n}{m-2} C_n^{m-2},$$

where $C_n^m$ is the normalizing constant for an $m$-ary multinomial and sample size $n$. 

Fast NML for Multinomials

There is a way — which is not naïve at all! — to do it in linear time, $O(n + m)$, using the following recursion:

$$C_n^m = C_n^{m-1} + \frac{n}{m - 2} C_n^{m-2},$$

where $C_n^m$ is the normalizing constant for an $m$-ary multinomial and sample size $n$.

The trick is to reduce the general case to $C_n^1 = 1$ and $C_n^2$, the latter of which can be computed in linear time (using the second most naïve approach).
Fast NML for Multinomials

There is a way — which is not naïve at all! — to do it in linear time, $O(n + m)$, using the following recursion:

$$C_n^m = C_n^{m-1} + \frac{n}{m-2} C_n^{m-2},$$

where $C_n^m$ is the normalizing constant for an $m$-ary multinomial and sample size $n$.

The trick is to reduce the general case to $C_1^1 = 1$ and $C_2^2$, the latter of which can be computed in linear time (using the second most naïve approach).

For a histogram density, we get again a code-length formula where \( \log_2 \frac{1}{f(x)} \) is the only essential term.
For a histogram density, we get again a code-length formula where 
\[ \log_2 \frac{1}{f(x)} \] is the only essential term.

Choosing the number \textit{and the positions} of break-points can be done by MDL.
For a histogram density, we get again a code-length formula where \( \log_2 \frac{1}{f(x)} \) is the only essential term.

Choosing the number *and the positions* of break-points can be done by MDL.

The code-length is equivalent (up to additive constants) to the code-length in a multinomial model.
For a histogram density, we get again a code-length formula where \( \log_2 \frac{1}{f(x)} \) is the only essential term.

Choosing the number and the positions of break-points can be done by MDL.

The code-length is equivalent (up to additive constants) to the code-length in a multinomial model.
\( \Rightarrow \) Linear time algorithm can be used.
Abstract

We regard histogram density estimation as a model selection problem. Our approach is based on the information-theoretic minimum description length (MDL) principle, which can be applied for tasks such as data clustering, density estimation, image denoising and model selection in general. MDL only on finding the optimal bin count. These regular histograms are, however, often problematic. It has been argued (Rissanen, Speed, & Yu, 1992) that regular histograms are only good for describing roughly uniform data. If the data distribution is strongly non-uniform, the bin count must necessarily be high if one wants to capture the details of the high density portion of the data. This in turn means that an unnecessary large amount of bins is wasted in the low density re-
Histogram Density Estimation
Consider the problem of clustering vectors of (independent) multinomial variables.
Consider the problem of clustering vectors of (independent) multinomial variables.

This can be seen as a way to encode (compress) the data:
Clustering

Consider the problem of clustering vectors of (independent) multinomial variables.

This can be seen as a way to encode (compress) the data:

1. first encode the cluster index of each observation vector,
Consider the problem of clustering vectors of (independent) multinomial variables.

This can be seen as a way to encode (compress) the data:

1. first encode the cluster index of each observation vector,
2. then encode the observations using separate (multinomial) models.
Clustering

Consider the problem of clustering vectors of (independent) multinomial variables.

This can be seen as a way to encode (compress) the data:
1. first encode the cluster index of each observation vector,
2. then encode the observations using separate (multinomial) models.

Again, the problem is reduced to the multinomial case, and the fast NML algorithm can be applied.
The clustering model can be interpreted as the naïve Bayes structure:

\[ \text{label} = \text{cluster index} \quad \text{and} \quad f_1, \ldots, f_n \text{ are features} \]
The clustering model can be interpreted as the **naïve Bayes** structure:

![Diagram](image)

- `label` = cluster index
- \( f_1, \ldots, f_n \) are **features**

The structure is very restrictive. Generalization achieved by **Bayesian networks**.
Clustering

The clustering model can be interpreted as the naïve Bayes structure:

\[ \text{label} = \text{cluster index} \quad f_1, \ldots, f_n \text{ are features} \]

The structure is very restrictive. Generalization achieved by Bayesian networks.

MDL criterion for learning Bayesian network structures again based on fast NML for multinomials.
The final week covers some additional topics:

- Kolmogorov complexity
- gambling