2019 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining

Modern MDL meets Data Mining
Insight, Theory, and Practice

Jilles Vreeken
CISPA Helmholtz Center for Information Security

Kenji Yamanishi
The University of Tokyo

About the presenters

Jilles Vreeken

CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

Kenji Yamanishi

東京大学
THE UNIVERSITY OF TOKYO

About this tutorial

Approximately 3.5 hours long

Extensive, but **incomplete** introduction to

- MDL theory
- MDL practice **in data mining**
- naturally a bit biased

Schedule

8:00am	Opening
8:10am	Introduction to MDL
8:50am	MDL in Action
9:30am	*break*
10:00am	Stochastic Complexity
11:00am	MDL in Dynamic Settings

Schedule

8:00am	Opening
8:10am	Introduction to MDL
8:50am	MDL in Action
9:30am	——— *break* ———
10:00am	Stochastic Complexity
11:00am	MDL in Dynamic Settings

Schedule

8:00am	Opening
8:10am	Introduction to MDL
8:50am	MDL in Action
9:30am	———*break*———
10:00am	Stochastic Complexity
11:00am	MDL in Dynamic Settings

Part 1

Introduction to MDL

Jilles Vreeken

Induction by Simplicity

"The simplest description of an object is the best"

Kolmogorov Complexity

$$K_U(x) = \min_y \{\, l(y) \mid U(y) \text{ halts and } U(y) = x \}$$

The Kolmogorov complexity of a binary string x
is the length of the shortest program y^*
for a universal Turing Machine U
that generates s and halts.

(Solomonoff 1960, Kolmogorov 1965, Chaitin 1969)

Kolmogorov Complexity

$$K_U(x) = \min_y \{\, l(y) \mid U(y) \text{ halts and } U(y) = x \}$$

The Kolmogorov complexity of a binary string x
is the length of the shortest program y^*
for a universal Turing Machine U
that generates s and **halts**.

Ultimately Impractical

Kolmogorov complexity $K(x)$, or rather,
the Kolmogorov optimal program x^* is not computable.

We can approximate it from above,
but, this is not very practical.
(simply not enough students to enumerate all Turing machines)

We can approximate it through
off-the-shelf compressors,
yet, this has serious drawbacks.
(big-O, what structure does a compressor reward, etc)

A practical variant

A more viable alternative is the
Minimum Description Length principle

*“the best model is the model
that gives the best lossless compression”*

There are two ways to motivate MDL

- we’ll discuss both at a high level
- then go into more details on what MDL is and can do

Two-Part MDL

The Minimum Description Length (MDL) principle

given a set of **hypotheses** $\mathcal{H}$, the best **hypothesis** $H \in \mathcal{H}$ for given data D is that H that minimises

$$L(H) + L(D \mid H)$$

in which

$L(H)$ is the length, in bits, of the description of H

$L(D \mid H)$ is the length, in bits, of the description of the data when encoded using H

(see, e.g., Rissanen 1978, 1983, Grünwald, 2007)

Bayesian Learning

Bayes tells us that

$$\Pr(H \mid D) = \frac{\Pr(D \mid H) \times \Pr(H)}{\Pr(D)}$$

This means we want the H that maximises $\Pr(H \mid D)$. Since $\Pr(D)$ is the same for all models, we have to **maximise** $\Pr(D \mid H) \times \Pr(H)$

Or, equivalently, **minimise** $-\log(\Pr(H)) - \log(\Pr(D \mid H))$

From Bayes to MDL

So, Bayesian Learning means minimising

$$-\log(\Pr(H)) - \log(\Pr(D \mid H))$$

Shannon tells us that the $-\log$ transform takes us from probabilities to optimal prefix-code lengths

This means we are actually minimizing

$$L(H) + L(D \mid H)$$

for some encoding L for H resp. $D \mid H$ corresponding to distribution $\Pr$

Bayesian MDL

If we want to do MDL this way
– i.e., being a Bayesian –
we need to specify

- a prior probability $\Pr(M)$ on the models, and
- a conditional probability $\Pr(D|M)$ on data given a model

What are reasonable choices?

What Distribution to Use?

For the data, this is 'easy': a maximum likelihood model

- a maximum entropy model for $\Pr(D \mid M)$ makes most sense

For the models, this is 'harder', we could, e.g., use

- 'whatever the expert says is a good distribution', or
- an uninformative prior on M, or
- (a derivative of) the universal prior from algorithmic statistics

These are not easy to compute, query, and ad hoc.

In MDL we say, if we are going to be ad hoc,
let us do so openly and use explicit universal encodings

Information Criteria

MDL might make you think of either

Akaike's Information Criterion (AIC)

$$k \ - \ln(\Pr(D|H))$$

or the Bayesian Information Criterion (BIC)

$$\frac{\mathrm{k}}{2}\ln(n) - \ln(\Pr(D|H))$$

Information Criteria

MDL might make you think of either

Akaike's Information Criterion (AIC)

$$k \ - L(D|H)$$

or the Bayesian Information Criterion (BIC)

$$\frac{\mathrm{k}}{2}\ln(n) - L(D|H)$$

Information Criteria

MDL might make you think of either

Akaike's Information Criterion (AIC)

$$L_{AIC}(H) = k$$

or the Bayesian Information Criterion (BIC)

$$L_{BIC}(H) = \frac{\mathrm{k}}{2} \ln(n)$$

We, however, do not assume that all parameters are created equal, we take their complexity into account

From Kolmogorov to MDL

Both Kolmogorov complexity and
MDL are based on compression.

Is there a relationship between the two?

Yes.

We can derive two-part MDL
from Kolmogorov complexity.
We'll sketch here how.

(see, e.g., Li & Vitanyi 1996, Vereshchagin & Vitanyi 2004 for details)

Objects and Sets

Recall that in Algorithmic Information Theory we are looking for (optimal) descriptions of **objects**.

One way to describe an object is

- describe **a set** of which it is a member
- **point out which** of these members **it is**.

In fact, we do this all the time

- the beach (i.e., the set of all beaches)
- over there (pointing out a specific one)

Algorithmic Statistics

We have, a set S

- which we call a model
- which has complexity $K(S)$

and an object $x \in S$

- S is a model of x
- the complexity of pointing out x in S is the complexity of x given S, i.e. $K(x \mid S)$

Obviously,

$$K(x) \leq K(S) + K(x \mid S)$$

So?

Algorithmic Information Theory states that

- every program that outputs x and halts encodes the information in x
- the smallest such program encodes only the information in x

If x is a data set, i.e. a random sample, we expect it has

- epistemic structure, “true” structure; captured by S
- aleatoric structure, “accidental” structure; captured by $x \mid S$

We are hence interested in that model S that minimizes

$$K(S) + K(x \mid S)$$

which is surprisingly akin to two-part MDL

More detail

For $K(S)$

- this is simply the length of the shortest program that outputs S and halts; i.e., a generative model of x

For $K(x \mid S)$

- if x is a typical element of S
  there is no more efficient way to find x in S than by an index, i.e.,
  $$K(x \mid S) \approx \log(|S|)$$

Kolmogorov's Structure Function

This suggests a way to discover the best model.

Kolmogorov's structure function is defined as

$$h_x(i) = \min_S \{\log(|S|) \mid x \in S, K(S) \leq i\}$$

That is, we start with very simple – in terms of complexity – models and gradually work our way up

(see, e.g., Li & Vitanyi 1996, Vereshchagin & Vitanyi 2004)

The MDL function

This suggests a way to discover the best model.

Kolmogorov's structure function is defined as

$$h_x(i) = \min_S \{\log(|S|) \mid x \in S, K(S) \leq i\}$$

which defines the MDL function as

$$\lambda_x(i) = \min_S \{K(S) + \log(|S|) \mid x \in S, K(S) \leq i\}$$

We try to find the minimum by considering increasingly complex models.

(see Vereshchagin & Vitanyi 2004)

The MDL function

This suggests a way to discover the best model.

Kolmogorov's structure function is defined as

$$h_x(i) = \min_S \{\log(|S|) \mid x \in S, K(S) \leq i\}$$

which defines the MDL function as

$$\lambda_x(i) = \min_S \{K(S) + \log(|S|) \mid x \in S, K(S) \leq i\}$$

We try to find the minimum by considering increasingly complex models.

(see Vereshchagin & Vitanyi 2004)

Two-Part MDL

The Minimum Description Length (MDL) principle

given a set of **hypotheses** $\mathcal{H}$, the best **hypothesis** $H \in \mathcal{H}$ for given data D is that H that minimises

$$L(H) + L(D \mid H)$$

in which

$L(H)$ is the length, in bits, of the description of H

$L(D \mid H)$ is the length, in bits, of the description of the data when encoded using H

(see, e.g., Rissanen 1978, 1983, Grünwald, 2007)

Example Binomial

Say we have a string

$x = 01011100001101010011$

of 10 zeroes and 10 ones

Suppose $\mathcal{H}$ consists of these binomials, e.g.

$p_1 = 0.1, p_2 = 0.2, p_3 = 0.5$

$$L(x \mid p_1) = -10 \log p_1 - 10 \log(1 - p_1) \ = 34.7 \text{ bits}$$
$$L(x \mid p_2) = -10 \log p_2 - 10 \log(1 - p_2) \ = 26.4 \text{ bits}$$
$$L(x \mid p_3) = -10 \log p_3 - 10 \log(1 - p_3) \ = 20.0 \text{ bits}$$

Example Binomial

Suppose $x = 01011100001101010011$,
and $\mathcal{H} = \{p_1 = 0.1, p_2 = 0.2, p_3 = 0.5\}$

Without prior preference over $H \in \mathcal{H}$
$L(H) = \log |\mathcal{H}|$

$L(p_1) + L(x \mid p_1) = 36.3$ bits
$L(p_2) + L(x \mid p_2) = 28.0$ bits
$L(p_3) + L(x \mid p_3) = 21.6$ bits

Example Binomial

Suppose $x = 010111000011010100 11$,
and $\mathcal{H} = \{p_1 = 0.1, p_2 = 0.2, p_3 = 0.5\}$

$$L(p_1) + L(x \mid p_1) = 36.3 \text{ bits}$$
$$L(p_2) + L(x \mid p_2) = 28.0 \text{ bits}$$
$$L(p_3) + L(x \mid p_3) = 21.6 \text{ bits}$$

However, when you receive $L(p_1)$ you know that p_2 and p_3 were disregarded by the sender as these did not lead to a minimal description.

Example Binomial

Suppose $x = 010111000011010100 11$,
and $\mathcal{H} = \{p_1 = 0.1, p_2 = 0.2, p_3 = 0.5\}$

$$L(p_1) + L(x \mid p_1) = 36.3 \text{ bits}$$
$$L(p_2) + L(x \mid p_2) = 28.0 \text{ bits}$$
$$L(p_3) + L(x \mid p_3) = 21.6 \text{ bits}$$

Models $H \in \mathcal{H}$ will only be used for data where they are optimal within the model class! Two-part MDL ignores this, it wastes bits!

Crude MDL

The Minimum Description Length (MDL) principle

given a set of **hypotheses** $\mathcal{H}$, the best **hypothesis** $H \in \mathcal{H}$ for given data D is that H that minimises

$$L(H) + L(D \mid H)$$

in which

$L(H)$ is the length, in bits, of the description of H

$L(D \mid H)$ is the length, in bits, of the description of the data when encoded using H

(see, e.g., Rissanen 1978, 1983, Grünwald, 2007)

Refined MDL

The main intuition, coming from crude MDL:

$L(H)$ is ad hoc, so we want to get rid of it, but keeping only $L(D \mid H)$ is going to give us a bad time, as maximising likelihood leads to overfitting.

$$L(D \mid \mathcal{H}) = \underbrace{L(D \mid H^*)}_{\text{Easy!}} + \underbrace{\mathrm{COMP}(\mathcal{H})}_{\text{Ehm...}}$$

aka the **stochastic complexity** of D given $\mathcal{H}$

Universal Codes

What Universal codes do we know?

- the two-part code (iff minimax guarantees, or large sample)
- prequential plug-in codes
- Bayesian mixtures codes (Jeffrey's prior)
- Normalised Maximum Likelihood (NML)

Each of these have quite a different nature, hence
different coding schemes, but all lead to very similar $L(D \mid \mathcal{H})$.

NML

Normalized Maximum Likelihood (Shtarkov, 1987)

$$L(D \mid \mathcal{H}) = -\log \frac{P(D \mid H^* \in \mathcal{H})}{\sum_{D' \in \mathcal{D}} P(D' \mid H' \in \mathcal{H})}$$

Interpretation:
The more special D is with respect to $\mathcal{H}$, the shorter its code.

One nasty detail, the normalization:
Enumerating every possible D’ requires many PhD students,
calculating the maximum likelihood H’ for every D’, even more so.

Crude in Practice

Refined MDL is **only** defined for a small set of cases.
Computing stochastic complexity is possible for **even fewer**.

Hence, in practice, as much as we may dislike it in theory,
we often have to resort to crude MDL.

However, as long as we're **aware of the biases** of the encoding,
that's **not a bad thing**.

In fact, as in two-part MDL we can steer our encoding towards models we (intuitively) like better, and hence for data mining purposes two-part MDL is a very often a good friend indeed.

MDL is a principle

MDL is **not** a single method

- it's a **general principle** for doing inductive inference

The main adage: **fewer bits is better**

- encode the data **universally**
  that is, without external input, only consider the data at hand
- ideally, uphold minimax optimality properties, try to make sure your encoding is never much worse than the best

Try to avoid, as much as possible, ad hoc biases

- **be explicit** about those that exist