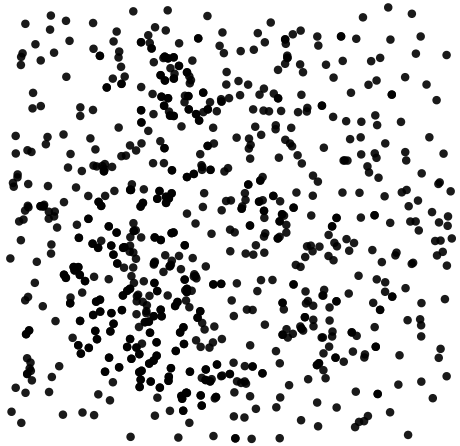
The universal information metric on finite binary strings

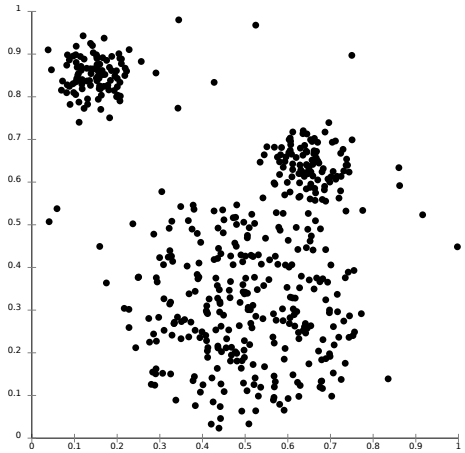
Jason Linehan [1]

[1] B.Sc. Candidate
Department of Mathematics
University of Florida

The clustering problem

Question: How to take raw, unstructured data, and introduce enough structure to allow analysis and classification?





Answer: Define a **distance function** to measure the pairwise distance between points in the sample space.

Data points are binary strings

Definition

A **binary string** x is a finite sequence of digits drawn from a two-element set, typically $\{0, 1\}$.

The space of all finite binary strings is notated $2^{<\omega}$.

Clearly, we can encode many different kinds of information as binary strings.

- DNA sequenced into ATGC, stands for an organism
- An array of RGB values could stand for handwriting
- EEG signal, sampled and digitized, stands for brain activity

When we cluster, we assume that the relationships we observe between binary strings also hold for the objects those strings encode.

Distances between data points

Definition (Distance function)

A **distance function** D on a set S has the form

$$D : S \times S \to \mathbb{R}^+.$$

Definition (Normalized distance function)

A **normalized distance function** $\hat{D}$ on a set S has the form

$$\hat{D} : S \times S \to [0, 1].$$

Similarity is normalized distance

Definition (Similarity)

The **similarity** between two objects is defined as the normalized distance between them.

Similarity is inherently relative.

- Two newspapers differing in only five words may have an absolute distance of 5, and be highly similar.
- Two headlines differing by five words may also have an absolute distance of 5, and be highly dissimilar.

To normalize is to scale by a notion of "size": to account for the number of differences available when quantifying the number of distances observed.

Defining a distance function

"How is a raven like a writing desk?"

Every property of two objects can be the basis for a distance: how similar the two objects are **in that aspect**.

How can we choose features that provide meaningful measures of distance, without contextual knowledge of the data?

Question: Is there a universal feature which can always be used to measure the similarity between two objects?

Information theory

Turing machines

Definition (Turing machine)

A **Turing machine** is the abstraction of an algorithm or computer program. For any computable function f, there is a Turing machine T_f such that $f(x) = T_f(x)$ for all $x \in \mathbb{N}$.

Definition (Universal Turing machine)

There exists a **universal Turing machine** U, such that for any Turing machine T and $x \in \mathbb{N}$, there is some $p \in \mathbb{N}$ so that $U(p) = T(x)$.

Think: U is the interpreter for a programming language. Its input p is a program written in that language. U compiles and then executes p. The result of the execution of p is the value $U(p)$.

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x)\downarrow\}$ forms a **prefix-free code**.

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x) \downarrow\}$ forms a **prefix-free code**.

Theorem (Kraft's inequality)

If a set $S \subseteq 2^{<\omega}$ is a prefix-free code, then $\sum_{x \in S} 2^{-|x|} \leq 1$.

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x) \downarrow\}$ forms a **prefix-free code**.

Theorem (Kraft's inequality)

If a set $S \subseteq 2^{<\omega}$ is a prefix-free code, then $\sum_{x \in S} 2^{-|x|} \leq 1$.

Why care about prefix-free machines?

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x)\downarrow\}$ forms a **prefix-free code**.

Theorem (Kraft's inequality)

If a set $S \subseteq 2^{<\omega}$ is a prefix-free code, then $\sum_{x \in S} 2^{-|x|} \leq 1$.

Why care about prefix-free machines?

- Consider a function $P(x) = \sum_{p:U(p)\downarrow=x} 2^{-|p|}$

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x)\downarrow\}$ forms a **prefix-free code**.

Theorem (Kraft's inequality)

If a set $S \subseteq 2^{<\omega}$ is a prefix-free code, then $\sum_{x \in S} 2^{-|x|} \leq 1$.

Why care about prefix-free machines?

- Consider a function $P(x) = \sum_{p:U(p)\downarrow=x} 2^{-|p|}$
- For every $x \in 2^{<\omega}$, the program id_x, computes x.

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x)\downarrow\}$ forms a **prefix-free code**.

Theorem (Kraft's inequality)

If a set $S \subseteq 2^{<\omega}$ is a prefix-free code, then $\sum_{x \in S} 2^{-|x|} \leq 1$.

Why care about prefix-free machines?

- Consider a function $P(x) = \sum_{p:U(p)\downarrow=x} 2^{-|p|}$
- For every $x \in 2^{<\omega}$, the program id_x, computes x.
- Then $P(x) \geq 2^{-|id_x|} = 2^{-|x|} = 2^{-log_2 x} = 1/x$

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x)\downarrow\}$ forms a **prefix-free code**.

Theorem (Kraft's inequality)

If a set $S \subseteq 2^{<\omega}$ is a prefix-free code, then $\sum_{x \in S} 2^{-|x|} \leq 1$.

Why care about prefix-free machines?

- Consider a function $P(x) = \sum_{p:U(p)\downarrow=x} 2^{-|p|}$
- For every $x \in 2^{<\omega}$, the program id_x, computes x.
- Then $P(x) \geq 2^{-|id_x|} = 2^{-|x|} = 2^{-log_2 x} = 1/x$
- Then $\sum_{x \in 2^{<\omega}} P(x) \to \infty$.

Prefix-free Universal Turing machines

Definition

A Turing machine T is **prefix-free** if for any $x, y \in Dom(T)$, $x \not\prec y$, and $y \not\prec x$. The set $Dom(T) = \{x \in 2^{<\omega} : T(x)\downarrow\}$ forms a **prefix-free code**.

Theorem (Kraft's inequality)

If a set $S \subseteq 2^{<\omega}$ is a prefix-free code, then $\sum_{x \in S} 2^{-|x|} \leq 1$.

Why care about prefix-free machines?

- Consider a function $P(x) = \sum_{p:U(p)\downarrow=x} 2^{-|p|}$
- For every $x \in 2^{<\omega}$, the program id_x, computes x.
- Then $P(x) \geq 2^{-|id_x|} = 2^{-|x|} = 2^{-log_2 x} = 1/x$
- Then $\sum_{x \in 2^{<\omega}} P(x) \to \infty$.
- What if U was a prefix-free machine?

Universal probability

Definition (Solomonoff)

Fix a universal prefix-free Turing machine U. Let the **universal discrete probability** be given by

$$\mathbf{P}(x) = \sum_{U(p)\downarrow = x} 2^{-|p|}.$$

The probability of computing x by flipping a coin to generate a program, then running that program on U.

Universal probability

Definition (Solomonoff)

Fix a universal prefix-free Turing machine U. Let the **universal discrete probability** be given by

$$\mathbf{P}(x) = \sum_{U(p)\downarrow = x} 2^{-|p|}.$$

The probability of computing x by flipping a coin to generate a program, then running that program on U.

Theorem (Levin)

If P is any lower semi-computable semi-measure, then there is a constant c such that $c \cdot \mathbf{P}(x) \geq P(x)$, for all $x \in 2^{<\omega}$.

Kolmogorov complexity

Definition (Kolmogorov complexity)

Given fixed prefix-free universal Turing machine U, define the **prefix-free Kolmogorov complexity** K of x to be the length of the smallest U-program which computes x.

- $K(x) = \min\{|p| : U(p) \downarrow = x\}$.
- $K(x) \leq_+ |id_x| = |x|$

Kolmogorov complexity

Definition (Kolmogorov complexity)

Given fixed prefix-free universal Turing machine U, define the **prefix-free Kolmogorov complexity** K of x to be the length of the smallest U-program which computes x.

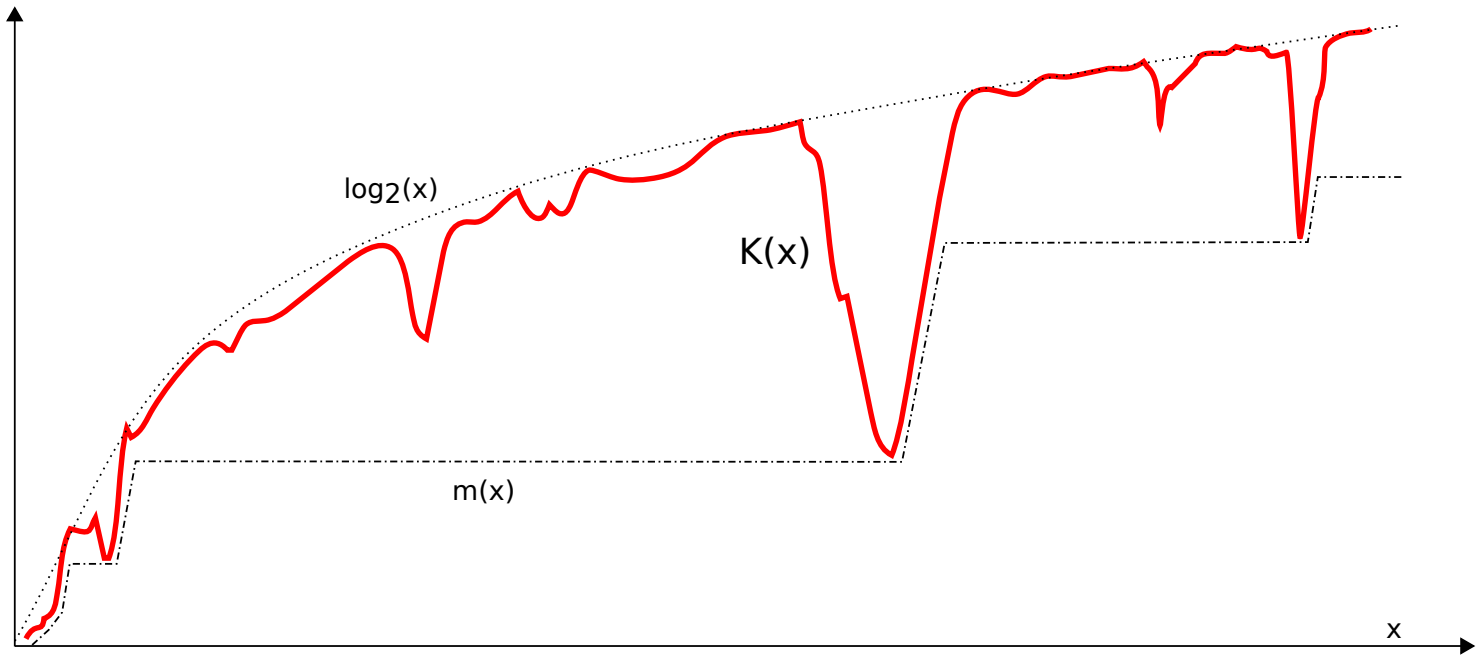
- $K(x) = \min\{|p| : U(p) \downarrow = x\}$.
- $K(x) \leq_+ |id_x| = |x|$

Definition (Conditional Kolmogorov complexity)

Given fixed prefix-free universal Turing machine U, define the **conditional prefix-free Kolmogorov complexity** of x relative to y as the length of the smallest U-program which computes x, given y as input.

- $K(x|y) = \min\{|p| : U(p, y) \downarrow = x\}$
- $K(x|\epsilon) = K(x)$
- $K(x|x) =_+ 0$

Schematic representation of K



$K(x)$, for $x \in \mathbb{N}$.

- $K(x)$ has a computable upper bound
- $m(x)$ diverges more slowly than any unbounded partial computable function.

Relationship between $\mathbf{P}(x|y)$ and $K(x|y)$

Theorem (Coding theorem)

For all $x \in 2^{<\omega}$,

$$\mathbf{P}(x) = 2^{-K(x)} + O(1)$$

Relationship between $\mathbf{P}(x|y)$ and $K(x|y)$

Theorem (Coding theorem)

For all $x \in 2^{<\omega}$,

$$\mathbf{P}(x) = 2^{-K(x)} + O(1)$$

Theorem (Conditional coding theorem)

For all $x, y \in 2^{<\omega}$,

$$\mathbf{P}(x|y) = 2^{-K(x|y)} + O(1)$$

Information distance

Recall the earlier question

Recall the earlier question

- **Question**: Is there a universal feature which can always be used to measure the similarity between two objects?

Recall the earlier question

- **Question**: Is there a universal feature which can always be used to measure the similarity between two objects?
- **Answer**: The Kolmogorov complexity of a program to translate from one object to another.

Information Distance

Bennett, Gacs, Vitanyi, et al. (1998)

Definition

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **information distance**

- $ID(x, y) = \min\{|p| : U(p, x) \downarrow = y \wedge U(p, y) \downarrow = x\}$

Information Distance

Bennett, Gacs, Vitanyi, et al. (1998)

Definition

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **information distance**

- $ID(x, y) = \min\{|p| : U(p, x) \downarrow = y \wedge U(p, y) \downarrow = x\}$

Remark

Suppose $K(x|y) = 10$, and $K(y|x) = 5$.

Information Distance

Bennett, Gacs, Vitanyi, et al. (1998)

Definition

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **information distance**

- $ID(x, y) = \min\{|p| : U(p, x) \downarrow = y \wedge U(p, y) \downarrow = x\}$

Remark

Suppose $K(x|y) = 10$, and $K(y|x) = 5$.

- We could build a conversion program using 15 bits, by concatenating the shortest programs.

Information Distance

Bennett, Gacs, Vitanyi, et al. (1998)

Definition

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **information distance**

- $ID(x, y) = \min\{|p| : U(p, x) \downarrow = y \wedge U(p, y) \downarrow = x\}$

Remark

Suppose $K(x|y) = 10$, and $K(y|x) = 5$.

- We could build a conversion program using 15 bits, by concatenating the shortest programs.
- Is this the shortest conversion program?

Information Distance

Bennett, Gacs, Vitanyi, et al. (1998)

Definition

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **information distance**

- $ID(x, y) = \min\{|p| : U(p, x) \downarrow = y \wedge U(p, y) \downarrow = x\}$

Remark

Suppose $K(x|y) = 10$, and $K(y|x) = 5$.

- We could build a conversion program using 15 bits, by concatenating the shortest programs.
- Is this the shortest conversion program?
- No! There is always a program having length roughly $\max\{K(x|y), K(y|x)\}$.

Information Distance

Bennett, Gacs, Vitanyi, et al. (1998)

Definition

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **information distance**

- $ID(x, y) = \min\{|p| : U(p, x) \downarrow = y \wedge U(p, y) \downarrow = x\}$

Remark

Suppose $K(x|y) = 10$, and $K(y|x) = 5$.

- We could build a conversion program using 15 bits, by concatenating the shortest programs.
- Is this the shortest conversion program?
- No! There is always a program having length roughly $\max\{K(x|y), K(y|x)\}$.

Theorem

- $ID(x, y) \stackrel{+}{=} \max\{K(x|y), K(y|x)\}$.

Properties of *ID*

ID satisfies three nice properties:

1. *ID* is a metric on $2^{<\omega}$
2. *ID* is an "admissible" distance
3. *ID* minorizes every other "admissible" distance

ID is a metric

Theorem (Bennett et al.)

$ID(x, y) = \max\{K(x|y), K(y|x)\}$ *is a metric on* $2^{<\omega}$.

Proof.

ID is a metric

Theorem (Bennett et al.)

$ID(x, y) = \max\{K(x|y), K(y|x)\}$ *is a metric on* $2^{<\omega}$.

Proof.

- $ID(x, x) = \max\{K(x|x), K(x|x)\} = K(x|x) = O(1)$

ID is a metric

Theorem (Bennett et al.)

$ID(x, y) = \max\{K(x|y), K(y|x)\}$ *is a metric on* $2^{<\omega}$.

Proof.

- $ID(x, x) = \max\{K(x|x), K(x|x)\} = K(x|x) = O(1)$
- $ID(x, y) = \max\{K(x|y), K(y|x)\} = ID(y, x)$

ID is a metric

Theorem (Bennett et al.)

$ID(x, y) = \max\{K(x|y), K(y|x)\}$ *is a metric on* $2^{<\omega}$.

Proof.

- $ID(x, x) = \max\{K(x|x), K(x|x)\} = K(x|x) = O(1)$
- $ID(x, y) = \max\{K(x|y), K(y|x)\} = ID(y, x)$
- $ID(x, z) \leq ID(x, y) + ID(y, z)$

ID is a metric

Theorem (Bennett et al.)

$ID(x, y) = \max\{K(x|y), K(y|x)\}$ *is a metric on* $2^{<\omega}$.

Proof.

- $ID(x, x) = \max\{K(x|x), K(x|x)\} = K(x|x) = O(1)$
- $ID(x, y) = \max\{K(x|y), K(y|x)\} = ID(y, x)$
- $ID(x, z) \leq ID(x, y) + ID(y, z)$
  - *Assume* $ID(x, z) = K(z|x)$.

ID is a metric

Theorem (Bennett et al.)

$ID(x, y) = \max\{K(x|y), K(y|x)\}$ *is a metric on* $2^{<\omega}$.

Proof.

- $ID(x, x) = \max\{K(x|x), K(x|x)\} = K(x|x) = O(1)$
- $ID(x, y) = \max\{K(x|y), K(y|x)\} = ID(y, x)$
- $ID(x, z) \leq ID(x, y) + ID(y, z)$
  - *Assume* $ID(x, z) = K(z|x)$.
  - $K(z|x) < K(y, z|x)$
  - $< K(y|x) + K(z|x, y)$
  - $< K(y|x) + K(z|y)$
  - $\leq ID(x, y) + ID(y, z)$.

□

ID is admissible

Definition

A distance D is admissible if it is total, non-negative, upper semi-computable, and satisfies the density condition.

ID is admissible

Definition

A distance D is admissible if it is total, non-negative, upper semi-computable, and satisfies the density condition.

Definition (Density condition)

We want to exclude certain distances like the discrete metric.

- Only finitely many objects y should be distance d from x

ID is admissible

Definition

A distance D is admissible if it is total, non-negative, upper semi-computable, and satisfies the density condition.

Definition (Density condition)

We want to exclude certain distances like the discrete metric.

- Only finitely many objects y should be distance d from x
- How fast the distances go to ∞ is a matter of scaling

ID is admissible

Definition

A distance D is admissible if it is total, non-negative, upper semi-computable, and satisfies the density condition.

Definition (Density condition)

We want to exclude certain distances like the discrete metric.

- Only finitely many objects y should be distance d from x
- How fast the distances go to ∞ is a matter of scaling
- Define the **density condition** as $\sum_{y:y\neq x} 2^{-D(x,y)} \leq 1$.

ID is admissible

Definition

A distance D is admissible if it is total, non-negative, upper semi-computable, and satisfies the density condition.

Definition (Density condition)

We want to exclude certain distances like the discrete metric.

- Only finitely many objects y should be distance d from x
- How fast the distances go to ∞ is a matter of scaling
- Define the **density condition** as $\sum_{y:y\neq x} 2^{-D(x,y)} \leq 1$.

Theorem (Bennett et al.)

ID is an admissible distance.

ID is admissible

Definition

A distance D is admissible if it is total, non-negative, upper semi-computable, and satisfies the density condition.

Definition (Density condition)

We want to exclude certain distances like the discrete metric.

- Only finitely many objects y should be distance d from x
- How fast the distances go to ∞ is a matter of scaling
- Define the **density condition** as $\sum_{y:y\neq x} 2^{-D(x,y)} \leq 1$.

Theorem (Bennett et al.)

ID is an admissible distance.

- *Why is* $\sum_{y:y\neq x} 2^{-ID(x,y)} \leq 1$*?*

ID is universal

Theorem (Bennett et al.)

$ID(x, y) \leq_+ D(x, y)$ *for every admissible metric distance* D

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy* $\sum_{y:y\neq x} 2^{-D(x,y)} \leq 1$.

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy* $\sum_{y:y\neq x} 2^{-D(x,y)} \leq 1$.
- *So* $2^{-D(x,y)}$ *is a l.s.c. semi-measure.*

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy* $\sum 2^{-D(x,y)}_{y:y\neq x} \leq 1$.
- *So* $2^{-D(x,y)}$ *is a l.s.c. semi-measure.*
- *Then* $\mathbf{P}(x|y) >_\times 2^{-D(x,y)}$

□

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy* $\sum 2^{-D(x,y)}_{y:y\neq x} \leq 1$.
- *So* $2^{-D(x,y)}$ *is a l.s.c. semi-measure.*
- *Then* $\mathbf{P}(x|y) >_\times 2^{-D(x,y)}$
- *So* $2^{-K(x|y)} >_\times 2^{-D(x,y)}$

□

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy* $\sum 2^{-D(x,y)}_{y:y\neq x} \leq 1$.
- *So* $2^{-D(x,y)}$ *is a l.s.c. semi-measure.*
- *Then* $\mathbf{P}(x|y) >_\times 2^{-D(x,y)}$
- *So* $2^{-K(x|y)} >_\times 2^{-D(x,y)}$
- *So* $K(x|y) <_+ D(x,y)$

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy $\sum 2^{-D(x,y)}_{y:y\neq x} \leq 1$.*
- *So $2^{-D(x,y)}$ is a l.s.c. semi-measure.*
- *Then $\mathbf{P}(x|y) >_\times 2^{-D(x,y)}$*
- *So $2^{-K(x|y)} >_\times 2^{-D(x,y)}$*
- *So $K(x|y) <_+ D(x,y)$*
- *Repeat for $D(y,x)$*

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy* $\sum 2^{-D(x,y)}_{y:y\neq x} \leq 1$.
- *So* $2^{-D(x,y)}$ *is a l.s.c. semi-measure.*
- *Then* $\mathbf{P}(x|y) >_\times 2^{-D(x,y)}$
- *So* $2^{-K(x|y)} >_\times 2^{-D(x,y)}$
- *So* $K(x|y) <_+ D(x,y)$
- *Repeat for* $D(y,x)$
- *Then* $\max\{K(x|y), K(y|x)\} <_+ D(x,y)$

ID is universal

Theorem (Bennett et al.)

$ID(x,y) \leq_+ D(x,y)$ for every admissible metric distance D

Proof.

- *Admissible D must satisfy* $\sum 2^{-D(x,y)}_{y:y\neq x} \leq 1$.
- *So* $2^{-D(x,y)}$ *is a l.s.c. semi-measure.*
- *Then* $\mathbf{P}(x|y) >_\times 2^{-D(x,y)}$
- *So* $2^{-K(x|y)} >_\times 2^{-D(x,y)}$
- *So* $K(x|y) <_+ D(x,y)$
- *Repeat for* $D(y,x)$
- *Then* $\max\{K(x|y), K(y|x)\} <_+ D(x,y)$
- *So* $ID(x,y) <_+ D(x,y)$, *for all admissible* D.

□

Summary

- $ID(x, y) =_+ \max\{K(x|y), K(y|x)\}$
- ID is a metric on $2^{<\omega}$
- ID is admissible
- ID minorizes all other admissible distances

Summary

- $ID(x, y) =_+ \max\{K(x|y), K(y|x)\}$
- ID is a metric on $2^{<\omega}$
- ID is admissible
- ID minorizes all other admissible distances

Definition (Recall)

The **similarity** between two objects is defined as the normalized distance between them.

Summary

- $ID(x,y) =_+ \max\{K(x|y), K(y|x)\}$
- ID is a metric on $2^{<\omega}$
- ID is admissible
- ID minorizes all other admissible distances

Definition (Recall)

The **similarity** between two objects is defined as the normalized distance between them.

Definition

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **normalized information distance**

$$NID(x,y) =_+ \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}$$

Compression distance

Compression functions

Definition

A **compressor** $f : 2^{<\omega} \to 2^{<\omega}$ is a partial computable injection such that $Dom(f)$ and $Range(f)$ are computable.

Compression functions

Definition

A **compressor** $f : 2^{<\omega} \to 2^{<\omega}$ is a partial computable injection such that $Dom(f)$ and $Range(f)$ are computable.

Definition

For a compressor f, $Z_f : 2^{<\omega} \to \mathbb{N}$ is defined by $Z_f(x) = |f(x)|$.

Compression functions

Definition

A **compressor** $f : 2^{<\omega} \to 2^{<\omega}$ is a partial computable injection such that $Dom(f)$ and $Range(f)$ are computable.

Definition

For a compressor f, $Z_f : 2^{<\omega} \to \mathbb{N}$ is defined by $Z_f(x) = |f(x)|$.

Example

Most data compression programs or binary encoders, e.g. `gzip`, `bzip2`, `mp3` encoders, fall into this category.

Effectivizing the *NID*
Li, Vitanyi, Cilibrasi (2003)

Definition (Recall)

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **normalized information distance**

$$NID(x,y) =_+ \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}$$

Definition

Fixing a compressor f, for $x, y \in 2^{<\omega}$, define the **normalized compression distance**

$$NCD_{Z_f}(x,y) = \frac{Z_f(xy) - \min\{Z_f(x), Z_f(y)\}}{\max\{Z_f(x), Z_f(y)\}}$$

Effectivizing the *NID*
Li, Vitanyi, Cilibrasi (2003)

Definition (Recall)

Fixing a universal machine U, for $x, y \in 2^{<\omega}$, define the **normalized information distance**

$$NID(x,y) =_+ \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}$$

Definition

Fixing a compressor f, for $x, y \in 2^{<\omega}$, define the **normalized compression distance**

$$NCD_{Z_f}(x,y) = \frac{Z_f(xy) - \min\{Z_f(x), Z_f(y)\}}{\max\{Z_f(x), Z_f(y)\}}$$

Does NCD_{Z_f} have the same properties as NID?

Definition

A compressor f is **normal** if Z_f satisfies the following conditions up to an additive $O(\log n)$ term.

NC1	$Z_f(xx) = Z_f(x)$	*(idempotence)*
NC2	$Z_f(xy) \geq Z_f(x)$	*(monotonicity)*
NC3	$Z_f(xy) \geq Z_f(yx)$	*(symmetry)*
NC4	$Z_f(xy) + Z_f(z) \leq Z_f(xz) + Z_f(yz)$	*(distributivity)*

Proposition

If f is a normal compressor, then NCD_{Z_f} is a normalized admissible distance satisfying the metric inequalities up to an additive logarithmic constant.

Remark

Universality is not so clear...

Summary

- There is a universal pairwise similarity metric on $2^{<\omega}$.
- Its value is not computable.
- It is possible to approximate the value using a special kind of data compressor.
- Even in approximation, such a similarity metric would have significant utility in machine learning and data analysis

Next time

- Do truly normal compressors exist?
- How should we understand the normality conditions?
- How can we promote or injure the normality?
- Are certain kinds of data more/less sensitive to this approach?
- Clustering and classification using actual data