The Coding Divergence for Measuring the Complexity of Separating Two Sets

Mahito SUGIYAMA\textsuperscript{†,‡}, Akihiro YAMAMOTO\textsuperscript{†}
\textsuperscript{†}Kyoto University
\textsuperscript{‡}JSPS Research Fellow
Outline

• Main results:
  1. We propose the coding divergence, a novel measure of the similarity between two sets of continuous data
     – Measure the complexity of separating the two sets
  2. We constructed the lazy learner, and showed the competitive performance in classification by experiments
Outline

• **Main results:**
  1. We propose the **coding divergence**, a novel measure of the similarity between two sets of continuous data
     - Measure the **complexity of separating** the two sets
  2. We constructed the lazy learner, and showed the competitive performance in classification by experiments

• **Key processes:**
  1. Embed continuous data in the **Euclidean space** $\mathbb{R}^d$ into the **Cantor space** $\Sigma^\omega$ topologically (discretization)
  2. **Learn** the simplest model (open set) in $\Sigma^\omega$
  3. Count the **length of the code** encoding the model
Examples of the Coding Divergence
Examples of the Coding Divergence

Binary-coding of real numbers in $[0, 1]$
Examples of the Coding Divergence

Binary-coding of real numbers in [0, 1]

Position
Examples of the Coding Divergence
Examples of the Coding Divergence
Examples of the Coding Divergence

The coding divergence:
\[ \frac{2}{10} + \frac{2}{10} = 0.4 \]
Examples of the Coding Divergence

The coding divergence:
\[ \frac{2}{10} + \frac{2}{10} = 0.4 \]

\[(0, 0), (1, 1)\]
Examples of the Coding Divergence

The coding divergence:

\[
\frac{2}{10} + \frac{2}{10} = 0.4
\]

\((0, 0)\)

\((1, 1)\)
Examples of the Coding Divergence

The coding divergence:
\[ \frac{2}{10} + \frac{2}{10} = 0.4 \]
Examples of the Coding Divergence

The coding divergence:
\[ \frac{2}{10} + \frac{2}{10} = 0.4 \]
Examples of the Coding Divergence

The coding divergence:
\[
\frac{26}{10} + \frac{26}{10} = \frac{52}{10} + \frac{2}{10} = 0.4
\]

The coding divergence:
\[
(11, 11), (10, 10), (111, 101), (010, 010), (011, 011), (110, 110), (0, 0), (1, 1), (010, 011), (011, 010), (00, 01), (00, 00)
\]

The coding divergence:
\[
\frac{26}{10} + \frac{26}{10} = 5.2
\]
In experimental science, **controlled experiments** are the standard method to test hypotheses

- Example: Compare two groups, one of which receives a placebo (control) and the other receives a new drug (treatment), to test the effect of the drug

**Statistical hypothesis testing** (*e.g.*, $t$-test) is a typical method, but has many problems [Johnson, 99]

- Non-verifiable assumptions and arbitrary $p$ values

We can treat in the Machine Learning context, since all we have to do is comparing two classes

The coding divergence can achieve this task
Motivation

Continuous data (reals)

<table>
<thead>
<tr>
<th></th>
<th>att. A</th>
<th>att. B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><img src="blue" alt="Bar 1" /></td>
<td><img src="blue" alt="Bar 1" /></td>
</tr>
<tr>
<td>2</td>
<td><img src="blue" alt="Bar 2" /></td>
<td><img src="blue" alt="Bar 2" /></td>
</tr>
<tr>
<td>3</td>
<td><img src="red" alt="Bar 3" /></td>
<td><img src="red" alt="Bar 3" /></td>
</tr>
<tr>
<td>4</td>
<td><img src="red" alt="Bar 4" /></td>
<td><img src="red" alt="Bar 4" /></td>
</tr>
</tbody>
</table>

Encoded by infinite sequences
## Motivation

### Continuous data (reals)

<table>
<thead>
<tr>
<th></th>
<th>att. A</th>
<th>att. B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.239582</td>
<td>0.6469</td>
</tr>
<tr>
<td>2</td>
<td>0.426711</td>
<td>0.2655</td>
</tr>
<tr>
<td>3</td>
<td>1.111577</td>
<td>0.4998</td>
</tr>
<tr>
<td>4</td>
<td>1.801501</td>
<td>0.7569</td>
</tr>
</tbody>
</table>

Encoded by infinite sequences
Motivation

<table>
<thead>
<tr>
<th>Continuous data (reals)</th>
<th>Discretization</th>
<th>Discrete data (rationals)</th>
<th>Stored in databases</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>att. A</td>
<td>att. B</td>
<td>att. A</td>
</tr>
<tr>
<td></td>
<td>1.239582...</td>
<td>0.6469...</td>
<td>1.2</td>
</tr>
<tr>
<td></td>
<td>0.426711...</td>
<td>0.2655...</td>
<td>0.4</td>
</tr>
<tr>
<td></td>
<td>1.111577...</td>
<td>0.4998...</td>
<td>1.1</td>
</tr>
<tr>
<td></td>
<td>1.801501...</td>
<td>0.7569...</td>
<td>1.8</td>
</tr>
</tbody>
</table>

Encoded by infinite sequences

Keep only finite prefixes
## Motivation

### Continuous data (reals)

<table>
<thead>
<tr>
<th></th>
<th>att. A</th>
<th>att. B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.239582...</td>
<td>0.6469...</td>
</tr>
<tr>
<td>2</td>
<td>0.426711...</td>
<td>0.2655...</td>
</tr>
<tr>
<td>3</td>
<td>1.111377...</td>
<td>0.4998...</td>
</tr>
<tr>
<td>4</td>
<td>1.801501...</td>
<td>0.7569...</td>
</tr>
</tbody>
</table>

Encoded by infinite sequences

### Discrete data (rationals)

<table>
<thead>
<tr>
<th></th>
<th>att. A</th>
<th>att. B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.2</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>1.8</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Data assumed theoretically

Data used for learning

Keep only finite prefixes

Discretization

Stored in databases
Discretization Using the Cantor Space

- The Cantor topology $\tau_{\Sigma^\omega} := \{ W\Sigma^\omega \mid W \subseteq \Sigma^* \}$, and the topological space $(\Sigma^\omega, \tau_{\Sigma^\omega})$ is called the Cantor space
  - The Cantor space is the standard topological space induced on the set of infinite sequences $\Sigma^\omega$
    - $w\Sigma^\omega = \{ p \in \Sigma^\omega \mid w \subseteq p \}$
    - $W\Sigma^\omega = \{ p \in \Sigma^\omega \mid \exists w \in W (w \subseteq p) \}$
  - The set $\{ w\Sigma^\omega \mid w \in \Sigma^* \}$ becomes a base of the space
  - If $P \subseteq \Sigma^\omega$ is open, then $P$ is finitely observable
    - A discretized datum is a base of an open set

- An embedding $\gamma : \subseteq \mathbb{R}^d \rightarrow \Sigma^\omega$ from the $d$-dimensional Euclidean space $\mathbb{R}^d$ into the Cantor space corresponds to a discretization process of continuous (real-valued) data
Example: The Binary Embedding $\gamma_2$

\[ \gamma_2(0.3) = 01001... \]

\[ \gamma_2((0.625, 0.75]) = 101 \sum_\omega \]

Position

0 1 2 3 4

$\gamma_2(0.3) = 01001...$

$\gamma_2((0.625, 0.75]) = 101 \sum_\omega$
Tree representation of the Binary Embedding $\gamma_2$
Tree representation of the Binary Embedding $\gamma_2$

$\gamma_2(0.3) = 01001...$

$\gamma_2((0.625, 0.75]) = 101\Sigma^\omega$

Open set

8/16
The Coding Divergence

For non-empty finite sets $X, Y \subset \mathcal{I}$ ($\mathcal{I}$ is the unit interval), define the coding divergence w.r.t. $\gamma$ by

$$C_\gamma(X, Y) := \left\{ \begin{array}{ll}
\infty & \text{if } X \cap Y \neq \emptyset, \\
D_\gamma(X; Y) + D_\gamma(Y; X) & \text{otherwise},
\end{array} \right.$$

- $D_\gamma$ is the directed coding divergence:

$$D_\gamma(X; Y) := \|X\|^{-1} \min\{|O| \mid O \text{ is open, and consistent with } (\gamma(X), \gamma(Y)) \}$$

- $\|X\|$ is the number of elements in $X$
- $|O| := \sum_{w \in W} |w|$, where $O = W \Sigma^\omega$
- $O$ is consistent $\iff O \supseteq \gamma(X)$ and $O \cap \gamma(Y) = \emptyset$
The Coding Divergence (cont.)

- For non-empty finite sets $X, Y \subset \mathcal{I}$ ($\mathcal{I}$ is the unit interval), define the coding divergence w.r.t. $\gamma$ by

$$C_\gamma(X, Y) := \begin{cases} \infty & \text{if } X \cap Y \neq \emptyset, \\ D_\gamma(X; Y) + D_\gamma(Y; X) & \text{otherwise,} \end{cases}$$

- The coding divergence depends on only the topological structure of the Cantor space
  - Machine Learning and Data Mining without probabilistic distributions can be realized
    - Different from statistical approaches
The Learning Algorithm M

function $\text{MAIN}(X, Y, k_{max})$

$(H_1, H_2) \leftarrow \text{LEARNING}(X, Y, \emptyset, \emptyset, 0, k_{max})$

return $\frac{1}{\|X\|} \sum_{v \in H_1} |v| + \frac{1}{\|Y\|} \sum_{w \in H_2} |w|$

function $\text{LEARNING}(X, Y, H_1, H_2, k, k_{max})$

$V \leftarrow \text{OBSERVE}(X, k), \quad W \leftarrow \text{OBSERVE}(Y, k)$

$H_1 \leftarrow H_1 \cup \{ v \in V \mid v \notin W \}, \quad H_2 \leftarrow H_2 \cup \{ w \in W \mid w \notin V \}$

$X \leftarrow \{ x \in X \mid x \notin \rho(H_1 \Sigma^W) \}, \quad Y \leftarrow \{ y \in Y \mid y \notin \rho(H_2 \Sigma^W) \}$

if $X = \emptyset$ and $Y = \emptyset$ then return $(H_1, H_2)$
else if $k = k_{max}$ then return $(H_1 \cup V, H_2 \cup W)$
else return $\text{LEARNING}(X, Y, H_1, H_2, k + 1, k_{max})$

function $\text{OBSERVE}(X, k)$

return $\{ y(x)[n] \mid x \in X \}$ (n = (k + 1)d − 1)
Learning of the Coding Divergence

\[ D_2(X; Y) = \{ 11, 011, 100 \} \]
\[ D_2(Y; X) = \{ 00, 010, 101 \} \]
\[ C_2(X, Y) = \{ \} \]
Learning of the Coding Divergence

\[ D_2(X; Y) \rightarrow \{11, 011, 100\} \]

\[ D_2(Y; X) \rightarrow \{00, 010, 101\} \]

\[ C_2(X, Y) = Y : X \]
Learning of the Coding Divergence

\[ D_2(X; Y) \rightarrow \{11, 011, 100\} \]
\[ D_2(Y; X) \rightarrow \{00, 010, 101\} \]

\[ C_2(X, Y) = : Y : X \]
Learning of the Coding Divergence

\[ D_2(X; Y) \to \{ 11, 011, 100 \} \]
\[ D_2(Y; X) \to \{ 00, 010, 101 \} \]
\[ C_2(X, Y) = : Y : X \]
Learning of the Coding Divergence

\[
D_2(X; Y) \rightarrow \{ 11, 011, 100 \}
\]

\[
D_2(Y; X) \rightarrow \{ 00, 010, 101 \}
\]

\[
C_2(X, Y) =
\]

\[
: Y \\
: X
\]
Learning of the Coding Divergence

\[ D_2(X; Y) = \{ 11, 011, 100 \} \]

\[ D_2(Y; X) = \{ 00, 010, 101 \} \]

\[ C_2(X, Y) = \]
Learning of the Coding Divergence

\[ D_2(X; Y) \rightarrow \{11, 011, 100\} \]
\[ D_2(Y; X) \rightarrow \{00, 010, 101\} \]

\[ C_2(X, Y) = \]

\[ D_2(X; Y) \rightarrow \{11\} \]
\[ D_2(Y; X) \rightarrow \{\} \]
Learning of the Coding Divergence

\[ D_2(X; Y) \rightarrow \{11\} \quad \text{and} \quad D_2(Y; X) \rightarrow \{\} \]

\[ \text{H}_2(X, Y) = \]
Learning of the Coding Divergence

$D_2(X; Y) \rightarrow \{11, 011, 100\}$  \quad $C_2(X, Y) =$

$D_2(Y; X) \rightarrow \{\}$
Learning of the Coding Divergence

\[ D_2(X; Y) \rightarrow \{11, 011, 100\} \]
\[ C_2(X, Y) = \frac{8}{5} + \frac{8}{5} = 3.2 \]
\[ D_2(Y; X) \rightarrow \{00, 010, 101\} \]
Learning of the Coding Divergence

\[ D_2(X; Y) \rightarrow \{11, 011, 100\} \]
\[ D_2(Y; X) \rightarrow \{00, 010, 101\} \]

\[ C_2(X, Y) = \frac{8}{5} + \frac{8}{5} = 3.2 \]

The computational complexity: \( O(mn) \) (\( m = \|X\|, \ n = \|Y\| \))
Classification with the Coding Div.

- Build a lazy learner using the coding divergence
- It receives training data $X$ in class A and $Y$ in class B, and classifies test data $Z$ to A or B
  - Assumption: All labels in $Z$ are same
- Use the learning algorithm $M$

$$Z \text{ is in } \begin{cases} A & \text{if } M(X, Z, k_{\text{max}}) > M(Y, Z, k_{\text{max}}), \\ B & \text{otherwise.} \end{cases}$$
Classification with the Coding Div.

- Build a lazy learner using the coding divergence
- It receives training data $X$ in class A and $Y$ in class B, and classifies test data $Z$ to A or B
  - Assumption: All labels in $Z$ are same
- Use the learning algorithm $M$
  
  $Z$ is in \[ \begin{cases} 
  A & \text{if } M(X, Z, k_{\text{max}}) > M(Y, Z, k_{\text{max}}), \\
  B & \text{otherwise.} 
  \end{cases} \]
Classification with the Coding Div.

• Build a lazy learner using the coding divergence
• It receives training data $X$ in class A and $Y$ in class B, and classifies test data $Z$ to A or B
  – Assumption: All labels in $Z$ are same
• Use the learning algorithm $M$

\[
Z \text{ is in } \begin{cases} 
  A & \text{if } M(X, Z, k_{\text{max}}) > M(Y, Z, k_{\text{max}}), \\
  B & \text{otherwise.}
\end{cases}
\]
Experimental Methods

- Implemented in **R language 2.10.1**
- Used **UCI data sets** (abalon, sonar, ...)
- Repeated the following procedure 10,000 times, and obtain accuracy from sensitivity and specificity
  - Choose attributes randomly
  - Sample $n$ data twice from each class ($X$, $T_+$ and $Y$, $T_-$)
    - $X$ and $Y$ are training data, $T_+$ and $T_-$ are test data
  - Normalize data (min-max normalization)
  - Classify $T_+$ and $T_-$ by our lazy learner and other methods
- Obtained accuracy by $(t_{pos} + t_{neg})/20000$, where $t_{pos}$ and $t_{neg}$ are the number of true positive and true negative, resp.
Experimental Results

- **Accuracy**
  - abalon
  - sonar

- **Data Sets**
  - (1, 10) (2, 10) (3, 10) (1, 30) (2, 30) (3, 30)
  - (Number of attributes, Size of each data set)

- **Methods**
  - CD
  - SVM (RBF)
  - SVM (pory)
  - 1NN
  - 5NN

- **Graphs**
  - abalon
  - sonar
  - Accuracy vs. (Number of attributes, Size of each data set)
Experimental Results

<table>
<thead>
<tr>
<th>Number of attributes, Size of each data set</th>
<th>CD SVM (RBF)</th>
<th>SVM (pory)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 10)</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>(2, 10)</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>(3, 10)</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>(1, 30)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2, 30)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3, 30)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Accuracy

- segmentation
- madelon

(Number of attributes, Size of each data set)
Experimental Results

<table>
<thead>
<tr>
<th>Number of attributes</th>
<th>Size of each data set</th>
<th>CD</th>
<th>SVM (RBF)</th>
<th>SVM (pory)</th>
<th>1NN</th>
<th>5NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 10)</td>
<td></td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
<td>0.5</td>
</tr>
<tr>
<td>(2, 10)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3, 10)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1, 30)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2, 30)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3, 30)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Accuracy

- waveform
- transfusion

(Number of attributes, Size of each data set)
Experimental Results

<table>
<thead>
<tr>
<th>(Number of attributes, Size of each data set)</th>
<th>CD SVM (RBF)</th>
<th>SVM (pory)</th>
</tr>
</thead>
<tbody>
<tr>
<td>glass</td>
<td>1.0</td>
<td>0.6</td>
</tr>
<tr>
<td>ionosphere</td>
<td>1.0</td>
<td>0.8</td>
</tr>
</tbody>
</table>
## Experimental Results

<table>
<thead>
<tr>
<th>(Number of attributes, Size of each data set)</th>
<th>CD</th>
<th>SVM (RBF)</th>
<th>SVM (pory)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 10)</td>
<td>0.5</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>(2, 10)</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3, 10)</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1, 30)</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2, 30)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3, 30)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Visual Representation

#### magic
- **CD**: Red bars
- **SVM (RBF)**: Green bars
- **SVM (pory)**: Blue bars

#### yeast
- **CD**: Red bars
- **SVM (RBF)**: Green bars
- **SVM (pory)**: Blue bars

The accuracy values are shown for different combinations of attributes and data set sizes.
Conclusion

- We proposed the **coding divergence** to measure the similarity between sets of continuous data
  - Embed continuous data in $\mathbb{R}^d$ into the **Cantor space** $\Sigma^\omega$ (discretization process)
  - Learn the simplest, consistent model (an open set in $\Sigma^\omega$)
  - Measure the similarity by the **length of the code** encoding the model

- We constructed a lazy learner for classification
  - This showed competitive performance compared to SVM and the $k$-nearest neighbor method
Appendix
Related Works

- Liu et al. constructed decision trees by partitioning intervals, and detected anomalies by measuring the height of the trees [Liu et al. 08]
  - Our works formulated this “partition” mathematically as embedding into the Cantor space
- Kernel methods (e.g., SVM) measure similarity of graphs and strings by mapping them to $\mathbb{R}^d$ or $\mathbb{R}^\infty$
  - Our strategy is inverted
    - Map $\mathbb{R}^d$ to the space of sequences $\Sigma^\omega$
  - Natural to treat feature space in a discrete manner
An Fatal Error Caused by Discretization

- Solve the system of linear equations [Schroder, 03]
  \[ 40157959.0 \, x + 67108865.0 \, y = 1 \]
  \[ 67108864.5 \, x + 112147127.0 \, y = 0 \]
  - Obtained by the well-known formula
    \[
    x = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11} a_{22} - a_{21} a_{12}}, \quad y = \frac{b_2 a_{11} - b_1 a_{21}}{a_{11} a_{22} - a_{21} a_{12}}
    \]
- By floating point arithmetic with double precision variables (IEEE 754):
  \[ x = 112147127, \quad y = -67108864.5 \]
- The correct solution:
  \[ x = 224294254, \quad y = -134217729 \]