Preference Learning: A Tutorial Introduction

Johannes Fürnkranz
Knowledge Engineering
Dept. of Computer Science
Technical University Darmstadt, Germany

Eyke Hüllermeier
Knowledge Engineering & Bioinformatics Lab
Dept. of Mathematics and Computer Science
Marburg University, Germany

DS 2011, Espoo, Finland, Oct 2011
What is Preference Learning?

- Preference learning is an emerging *subfield of machine learning*
- Roughly speaking, it deals with the *learning of (predictive) preference models* from observed (or extracted) preference information
Workshops and Related Events

- NIPS–01: New Methods for Preference Elicitation
- NIPS–02: Beyond Classification and Regression: Learning Rankings, Preferences, Equality Predicates, and Other Structures
- KI–03: Preference Learning: Models, Methods, Applications
- NIPS–04: Learning With Structured Outputs
- NIPS–05: Workshop on Learning to Rank
- IJCAI–05: Advances in Preference Handling
- SIGIR 07–10: Workshop on Learning to Rank for Information Retrieval
- **ECML/PDKK 08–10: Workshop on Preference Learning**
- NIPS–09: Workshop on Advances in Ranking
- American Institute of Mathematics Workshop in Summer 2010: The Mathematics of Ranking
- NIPS-11: Workshop on Choice Models and Preference Learning
Preferences in Artificial Intelligence

More generally, „preferences“ is a key topic in current AI research.

User preferences play a key role in various fields of application:
- recommender systems,
- adaptive user interfaces,
- adaptive retrieval systems,
- autonomous agents (electronic commerce),
- games, ...

Preferences in AI research:
- preference representation (CP nets, GAU networks, logical representations, fuzzy constraints, ...)
- reasoning with preferences (decision theory, constraint satisfaction, non-monotonic reasoning, ...)
- preference acquisition (preference elicitation, preference learning, ...)
AGENDA

1. Preference Learning Tasks
2. Performance Assessment and Loss Functions
3. Preference Learning Techniques
4. Conclusions
Preference Learning

Preference learning problems can be distinguished along several **problem dimensions**, including

- **representation of preferences, type of preference model:**
  - utility function (ordinal, numeric),
  - preference relation (partial order, ranking, ...),
  - logical representation, ...

- **description of individuals/users and alternatives/items:**
  - identifier, feature vector, structured object, ...

- **type of training input:**
  - direct or indirect feedback,
  - complete or incomplete relations,
  - utilities, ...

- ...
Preferenc Learning

Preferences

- assessing
  - absolute
  - binary
    - A B C D
    - 1 1 0 0
  - gradual
    - numeric
      - A B C D
      - .9 .8 .1 .3
    - ordinal
      - A B C D
      - + + - 0
- comparing
  - relative
    - total order
      - A ≥ B ≥ C ≥ D
    - partial order
      - A ≻ B ≻ C ≻ D

→ (ordinal) regression
→ classification/ranking
Structure of this Overview

(1) Preference learning as an extension of conventional supervised learning: Learn a mapping

\[ \mathcal{X} \rightarrow \mathcal{Y} \]

that maps instances to preference models (structured/complex output prediction).

(2) Other settings (object ranking, instance ranking, CF, ...)


Structure of this Overview

(1) Preference learning as an extension of conventional supervised learning:
Learn a mapping
\[ \mathcal{X} \rightarrow \mathcal{P} \]

that maps instances to preference models (\( \rightarrow \) structured/complex output prediction).

Instances are typically (though not necessarily) characterized in terms of a feature vector.

The output space consists of preference models over a fixed set of alternatives (classes, labels, ...) represented in terms of an identifier
\( \rightarrow \) extensions of multi-class classification
Multilabel Classification [Tsoumakas & Katakis 2007]

Training

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1.04</td>
<td>0</td>
<td>33</td>
<td>158</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Binary preferences on a fixed set of items: liked or disliked

Prediction

| 0.92 | 1  | 81 | 382 | 0 | 1 | 0 | 1 |

Ground truth

| 0.92 | 1  | 81 | 382 | 1 | 1 | 0 | 1 |

LOSS
## Multilabel Ranking

### Training

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1.04</td>
<td>0</td>
<td>33</td>
<td>158</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Binary preferences on a fixed set of items: liked or disliked

### Prediction

<table>
<thead>
<tr>
<th>0.92</th>
<th>1</th>
<th>81</th>
<th>382</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>D</td>
<td>C</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

A ranking of all items

### Ground truth

<table>
<thead>
<tr>
<th>0.92</th>
<th>1</th>
<th>81</th>
<th>382</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

LOSS
## Graded Multilabel Classification [Cheng et al. 2010]

### Training

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>--</td>
<td>+</td>
<td>++</td>
<td>0</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>0</td>
<td>++</td>
<td>--</td>
<td>+</td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>--</td>
<td>--</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>+</td>
<td>0</td>
<td>++</td>
<td>--</td>
</tr>
<tr>
<td>1.04</td>
<td>0</td>
<td>33</td>
<td>158</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>--</td>
</tr>
</tbody>
</table>

### Prediction

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.92</td>
<td>1</td>
<td>81</td>
<td>382</td>
<td>--</td>
<td>+</td>
<td>0</td>
<td>++</td>
</tr>
</tbody>
</table>

### Ground truth

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.92</td>
<td>1</td>
<td>81</td>
<td>382</td>
<td>0</td>
<td>++</td>
<td>--</td>
<td>+</td>
</tr>
</tbody>
</table>

Ordinal preferences on a fixed set of items: liked, disliked, or something in-between
Graded Multilabel Ranking

### Training

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>--</td>
<td>+</td>
<td>++</td>
<td>0</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>0</td>
<td>++</td>
<td>--</td>
<td>+</td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>--</td>
<td>--</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>+</td>
<td>0</td>
<td>++</td>
<td>--</td>
</tr>
<tr>
<td>1.04</td>
<td>0</td>
<td>33</td>
<td>158</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>--</td>
</tr>
</tbody>
</table>

**Ordinal preferences on a fixed set of items: liked, disliked, or something in-between**

### Prediction

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

A ranking of all items

### Ground truth

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>++</td>
<td>--</td>
<td>+</td>
</tr>
</tbody>
</table>
# Label Ranking [Hüllermeier et al. 2008]

## Training

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>Preferences</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>A ⊃ B, B ⊃ C, C ⊃ D</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>B ⊃ C</td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>B ⊃ D, A ⊃ D, C ⊃ D, A ⊃ C</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>C ⊃ A, C ⊃ D, A ⊃ B</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>B ⊃ D, A ⊃ D</td>
</tr>
<tr>
<td>1.04</td>
<td>0</td>
<td>33</td>
<td>158</td>
<td>D ⊃ A, A ⊃ B, C ⊃ B, A ⊃ C</td>
</tr>
</tbody>
</table>

Instances are associated with pairwise preferences between labels.

## Prediction

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.92</td>
<td>1</td>
<td>81</td>
<td>382</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

A ranking of all labels

## Ground truth

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.92</td>
<td>1</td>
<td>81</td>
<td>382</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

LOSS
Calibrated Label Ranking [Fürnkranz et al. 2008]

Combining absolute and relative evaluation:

\[ a \succ b \succ c \quad d \succ e \succ f \succ g \]

- relevant positive liked
- irrelevant negative disliked
Structure of this Overview

(1) Preference Learning as an extension of conventional supervised learning:
   Learn a mapping
   \[ \mathcal{X} \rightarrow \mathcal{P} \]
   that maps instances to preference models (\(\rightarrow\) structured output prediction).

(2) Other settings (no clear distinction between input/output space)
   object ranking, instance ranking, collaborative filtering
Object Ranking [Cohen et al. 99]

Training

\((0.74, 1, 25, 165) \succ (0.45, 0, 35, 155)\)
\((0.47, 1, 46, 183) \succ (0.57, 1, 61, 177)\)
\((0.25, 0, 26, 199) \succ (0.73, 0, 46, 185)\)
\((0.95, 0, 73, 133) \succ (0.25, 1, 35, 153)\)
\((0.68, 1, 55, 147) \succ (0.67, 0, 63, 182)\)

Prediction (ranking a new set of objects)

\(Q = \{ \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9, \mathbf{x}_{10}, \mathbf{x}_{11}, \mathbf{x}_{12}, \mathbf{x}_{13} \} \)

\(\mathbf{x}_{10} \succ \mathbf{x}_4 \succ \mathbf{x}_7 \succ \mathbf{x}_1 \succ \mathbf{x}_{11} \succ \mathbf{x}_2 \succ \mathbf{x}_8 \succ \mathbf{x}_{13} \succ \mathbf{x}_9 \succ \mathbf{x}_3 \succ \mathbf{x}_{12} \succ \mathbf{x}_5 \succ \mathbf{x}_6 \)

Ground truth (ranking or top-ranking or subset of relevant objects)

\(\mathbf{x}_{11} \succ \mathbf{x}_7 \succ \mathbf{x}_4 \succ \mathbf{x}_2 \succ \mathbf{x}_{10} \succ \mathbf{x}_1 \succ \mathbf{x}_8 \succ \mathbf{x}_{13} \succ \mathbf{x}_9 \succ \mathbf{x}_{12} \succ \mathbf{x}_3 \succ \mathbf{x}_5 \ succ \mathbf{x}_6 \)
\(\mathbf{x}_{11} \succ \mathbf{x}_7 \succ \mathbf{x}_4 \succ \mathbf{x}_2 \succ \mathbf{x}_{10} \)
\(P = \{ \mathbf{x}_{11}, \mathbf{x}_7, \mathbf{x}_4, \mathbf{x}_2, \mathbf{x}_{10}, \mathbf{x}_1 \} \quad N = \{ \mathbf{x}_8, \mathbf{x}_{13}, \mathbf{x}_9, \mathbf{x}_{12}, \mathbf{x}_3, \mathbf{x}_5, \mathbf{x}_6 \} \)
Instance Ranking [Fürnkranz et al. 2009]

Training

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(x_3)</th>
<th>(x_4)</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>--</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>0</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>++</td>
</tr>
</tbody>
</table>

... ... ... ... ...

<table>
<thead>
<tr>
<th>(x_n)</th>
<th>(x_2)</th>
<th>(x_3)</th>
<th>(x_4)</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>+</td>
</tr>
</tbody>
</table>

Absolute preferences on an ordinal scale.

Prediction (ranking a new set of objects)

\[ Q = \{ x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13} \} \]

\[ x_{10} \succ x_4 \succ x_7 \succ x_1 \succ x_{11} \succ x_2 \succ x_8 \succ x_{13} \succ x_9 \succ x_3 \succ x_{12} \succ x_5 \succ x_6 \]

Ground truth (ordinal classes)

\[ \begin{align*}
    x_{10} & \succ 0 \succ ++ \succ ++ \succ -- \succ + \succ 0 \succ + \succ -- \succ 0 \succ 0 \succ -- \succ --
\end{align*} \]
Instance Ranking [Fürnkranz et al. 2009]

Extension of AUC maximization to the polytomous case, in which instances are rated on an ordinal scale such as \{bad, medium, good\}.

Query set of instances to be ranked (true labels are unknown).

predicted ranking, e.g., through sorting by estimated score.

most likely good

most likely bad

ranking error
Collaborative Filtering [Goldberg et al. 1992]

<table>
<thead>
<tr>
<th>USERS</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>…</th>
<th>P38</th>
<th>…</th>
<th>P88</th>
<th>P89</th>
<th>P90</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>1</td>
<td>4</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U2</td>
<td>2</td>
<td>2</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U98</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U99</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1: very bad,  2: bad,  3: fair,  4: good,  5: excellent

Inputs and outputs as identifiers, absolute preferences in terms of ordinal degrees.
Dyadic Prediction [Menon & Elkan 2010]

<table>
<thead>
<tr>
<th></th>
<th>U1</th>
<th>P1</th>
<th>U2</th>
<th>P2</th>
<th>P3</th>
<th>…</th>
<th>P38</th>
<th>…</th>
<th>P88</th>
<th>P89</th>
<th>P90</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>14</td>
<td>45</td>
<td>32</td>
<td>52</td>
<td>61</td>
<td>16</td>
<td>33</td>
<td>53</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>P2</td>
<td>P3</td>
<td>…</td>
<td>P38</td>
<td>…</td>
<td>P88</td>
<td>P89</td>
<td>P90</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U1</td>
<td>1</td>
<td>4</td>
<td>…</td>
<td>…</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U2</td>
<td>2</td>
<td>2</td>
<td>…</td>
<td>…</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U98</td>
<td>5</td>
<td></td>
<td>…</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U99</td>
<td>1</td>
<td></td>
<td>…</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Additional side-information: observed features + latent features of users and items
## Preference Learning Tasks

<table>
<thead>
<tr>
<th>task</th>
<th>input</th>
<th>output</th>
<th>training</th>
<th>prediction</th>
<th>ground truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>collaborative filtering</td>
<td>identifier</td>
<td>identifier</td>
<td>absolute</td>
<td>absolute</td>
<td>absolute</td>
</tr>
<tr>
<td>multilabel classification</td>
<td>feature</td>
<td>identifier</td>
<td>absolute</td>
<td>absolute</td>
<td>absolute</td>
</tr>
<tr>
<td>multilabel ranking</td>
<td>feature</td>
<td>identifier</td>
<td>absolute</td>
<td>ranking</td>
<td>absolute</td>
</tr>
<tr>
<td>graded multilabel classification</td>
<td>feature</td>
<td>identifier</td>
<td>absolute</td>
<td>absolute</td>
<td>absolute</td>
</tr>
<tr>
<td>label ranking</td>
<td>feature</td>
<td>identifier</td>
<td>relative</td>
<td>ranking</td>
<td>ranking</td>
</tr>
<tr>
<td>object ranking</td>
<td>feature</td>
<td>--</td>
<td>relative</td>
<td>ranking</td>
<td>ranking or subset</td>
</tr>
<tr>
<td>instance ranking</td>
<td>feature</td>
<td>identifier</td>
<td>absolute</td>
<td>ranking</td>
<td>absolute</td>
</tr>
</tbody>
</table>

Two main directions: (1) ranking and variants (2) generalizations of classification.
Loss Functions

Things to be compared:

- absolute utility degree
- subset of preferred items
- fuzzy subset of preferred items
- subset of preferred items
- ranking of items
- ranking of items

standard comparison of scalar predictions

- prediction
- ground truth

non-standard comparisons
References

AGENDA

1. Preference Learning Tasks
2. Loss Functions
   a. Evaluation of Rankings
   b. Weighted Measures
   c. Evaluation of Bipartite Rankings
   d. Evaluation of Partial Rankings
3. Preference Learning Techniques
4. Conclusions
Rank Evaluation Measures

- In the following, we do not discriminate between different ranking scenarios
  - we use the term *items* for both, objects and labels

- All measures are applicable to both scenarios
  - sometimes have different names according to context

- Label Ranking
  - measure is applied to the ranking of the labels of each example
  - averaged over all examples

- Object Ranking
  - measure is applied to the ranking of a set of objects
  - we may need to average over different sets of objects which have disjoint preference graphs
    - e.g. different sets of query / answer set pairs in information retrieval
Given:

- a set of items $X = \{x_1, \ldots, x_c\}$ to rank

Example:

$X = \{A, B, C, D, E\}$

Items can be objects or labels.
Ranking Errors

- **Given:**
  - a set of items $X = \{x_1, \ldots, x_c\}$ to rank
    - *Example:* $X = \{A, B, C, D, E\}$
  - a target ranking $r$
    - *Example:* $E > B > C > A > D$
Ranking Errors

- Given:
  - a set of items $X = \{x_1, \ldots, x_c\}$ to rank
    - Example: $X = \{A, B, C, D, E\}$
  - a target ranking $r$
    - Example: $E > B > C > A > D$
  - a predicted ranking $\hat{r}$
    - Example: $A > B > E > C > D$

- Compute:
  - a value $d(r, \hat{r})$ that measures the distance between the two rankings

\[ d(\hat{r}, r) \]
Notation

- $r$ and $\hat{r}$ are functions from $X \rightarrow \mathbb{N}$
  - returning the rank of an item $x$
    $$\hat{r}(A) = 1$$

- the inverse functions $r^{-1}: \mathbb{N} \rightarrow X$
  - return the item at a certain position
    $$\hat{r}^{-1}(1) = A \quad r^{-1}(4) = A$$

- as a short-hand for $r \circ \hat{r}^{-1}$, we also define function $R: \mathbb{N} \rightarrow \mathbb{N}$
  - $R(i)$ returns the true rank of the $i$-th item in the predicted ranking
    $$R(1) = r(\hat{r}^{-1}(1)) = 4$$
Spearman's Footrule

- Key idea:
  - Measure the sum of absolute differences between ranks

\[
D_{SF}(r, \hat{r}) = \sum_{i=1}^{c} \left| r(x_i) - \hat{r}(x_i) \right| = \sum_{i=1}^{c} \left| i - R(i) \right|
\]

\[
= \sum_{i=1}^{c} d_{x_i}(r, \hat{r})
\]

\[
\sum_{x_i} d_{x_i} = 3 + 0 + 1 + 0 + 2 = 6
\]
Spearman Distance

- **Key idea:**
  - Measure the sum of absolute differences between ranks
  
  \[ D_s(r, \hat{r}) = \sum_{i=1}^{c} (r(x_i) - \hat{r}(x_i))^2 = \sum_{i=1}^{c} (i - R(i))^2 \]
  
  \[ = \sum_{i=1}^{c} d_x(r, \hat{r})^2 \]

- **Value range:**
  
  \[ \min D_s(r, \hat{r}) = 0 \]
  
  \[ \max D_s(r, \hat{r}) = \sum_{i=1}^{c} ((c-i)-i)^2 = \frac{c \cdot (c^2-1)}{3} \]

  \[ \rightarrow \text{Spearman Rank Correlation Coefficient} \]

  \[ 1 - \frac{6 \cdot D_s(r, \hat{r})}{c \cdot (c^2-1)} \in [-1, 1] \]

\[ \sum_{x_i} d^2_{x_i} = 3^2 + 0^2 + 1^2 + 0 + 2^2 = 14 \]
Kendall's Distance

- **Key idea:**
  - number of item pairs that are inverted in the predicted ranking

\[ D_\tau(r, \hat{r}) = | \{(i, j) \mid r(x_i) < r(x_j) \land \hat{r}(x_i) > \hat{r}(x_j)\} | \]

- **Value range:**
  - \( \min D_\tau(r, \hat{r}) = 0 \)
  - \( \max D_\tau(r, \hat{r}) = \frac{c \cdot (c-1)}{2} \)

\[ 1 - \frac{4 \cdot D_\tau(r, \hat{r})}{c \cdot (c-1)} \in [-1, +1] \]

\[ D_\tau(r, \hat{r}) = 4 \]
AGENDA

1. Preference Learning Tasks
2. Loss Functions
   a. Evaluation of Rankings
   b. Weighted Measures
   c. Evaluation of Bipartite Rankings
   d. Evaluation of Partial Rankings
3. Preference Learning Techniques
4. Conclusions
Weighted Ranking Errors

- The previous ranking functions give equal weight to all ranking positions
  - i.e., differences in the first ranking positions have the same effect as differences in the last ranking positions

\[ D(C, B) = D(C, B) \]

- In many applications this is not desirable
  - ranking of search results
  - ranking of product recommendations
  - ranking of labels for classification
  - ...

⇒ Higher ranking positions should be given more weight
### Position Error

- **Key idea:**
  - In many applications, we are interested in providing a ranking where the target item appears as high as possible in the predicted ranking.
  - E.g., ranking a set of actions for the next step in a plan.
  - Error is the number of wrong items that are predicted before the target item.

\[
D_{PE}(r, \hat{r}) = \hat{r}(\arg\min_{x \in X} r(x)) - 1
\]

- **Note:**
  - Equivalent to Spearman's footrule with all non-target weights set to 0.

\[
D_{PE}(r, \hat{r}) = \sum_{i=1}^{c} w_i \cdot d_{x_i}(r, \hat{r})
\]

with \( w_i = [x_i = \arg\min_{x \in X} r(x)] \)

\[
D_{PE}(r, \hat{r}) = 2
\]
Discounted Error

- Higher ranks in the target position get a higher weight than lower ranks

\[ D_{DR}(r, \hat{r}) = \sum_{i=1}^{c} w_i \cdot d_{x_i}(r, \hat{r}) \]

with \( w_i = \frac{1}{\log(r(x_i)+1)} \)

\[ D_{DR}(r, \hat{r}) = \frac{3}{\log 2} + 0 + \frac{1}{\log 4} + 0 + \frac{2}{\log 6} \]
(Normalized) Discounted Cumulative Gain

- A “positive” version of discounted error:
  
  **Discounted Cumulative Gain (DCG)**
  \[
  DCG(r, \hat{r}) = \sum_{i=1}^{c} \frac{c - R(i)}{\log(i+1)}
  \]

- Maximum possible value:
  - The predicted ranking is correct, i.e. \( \forall i : i = R(i) \)
  - **Ideal Discounted Cumulative Gain (IDCG)**
    \[
    IDCG = \sum_{i=1}^{c} \frac{c - i}{\log(i+1)}
    \]

- **Normalized DCG (NDCG)**
  \[
  NDCG(r, \hat{r}) = \frac{DCG(r, \hat{r})}{IDCG}
  \]

\[
NDCG(r, \hat{r}) = \frac{1}{\log 2 + \frac{1}{\log 3} + \frac{2}{\log 4} + \frac{2}{\log 5} + \frac{0}{\log 6}}
\frac{3}{\log 2} + \frac{4}{\log 3} + \frac{2}{\log 4} + \frac{1}{\log 5} + \frac{0}{\log 6}
\]

Diagram:
- A
- B
- C
- D
- E
- \( \hat{r} \)
- \( r \)
AGENDA

1. Preference Learning Tasks
2. Loss Functions
   a. Evaluation of Rankings
   b. Weighted Measures
   c. Evaluation of Bipartite Rankings
   d. Evaluation of Partial Rankings
3. Preference Learning Techniques
4. Conclusions
Bipartite Rankings

- The target ranking is not totally ordered but a \textit{bipartite graph}.
- The two partitions may be viewed as preference levels $L = \{0, 1\}$
  - all $c_1$ items of level 1 are preferred over all $c_0$ items of level 0.

- We now have fewer preferences
  - for a total order: $\frac{c}{2} \cdot (c - 1)$
  - for a bipartite graph: $c_1 \cdot (c - c_1)$
Evaluating Partial Target Rankings

- Many Measures can be directly adapted from total target rankings to partial target rankings
- Recall: Kendall's distance
  - number of item pairs that are inverted in the target ranking
    \[ D_{\tau}(r, \hat{r}) = | \{(i, j) | r(x_i) < r(x_j) \land \hat{r}(x_i) > \hat{r}(x_j)\} | \]
  - can be directly used
  - in case of normalization, we have to consider that fewer items satisfy \( r(x_i) < r(x_j) \)
- Area under the ROC curve (AUC)
  - the AUC is the fraction of pairs of \((p,n)\) for which the predicted score \( s(p) > s(n) \)
    - Mann Whitney statistic is the absolute number
    - This is \(1 - \) normalized Kendall's distance for a bipartite preference graph with \( L = \{p,n\} \)

\[ D_{\tau}(r, \hat{r}) = 2 \]
\[ AUC(r, \hat{r}) = \frac{4}{6} \]
Evaluating Multipartite Rankings

- Multipartite rankings:
  - like Bipartite rankings
  - but the target ranking \( r \) consists of multiple relevance levels \( L = \{1 \ldots l\} \), where \( l < c \)
  - total ranking is a special case where each level has exactly one item

- \# of preferences \( \sum_{(i,j)} c_i \cdot c_j \leq \frac{c^2}{2} \cdot (1 - \frac{1}{l}) \)
  - \( c_i \) is the number of items in level \( I \)

- C-Index [Gnen & Heller, 2005]
  - straight-forward generalization of AUC
  - fraction of pairs \((x_i,x_j)\) for which
    \[ I(i) > I(j) \land \hat{r}(x_i) < \hat{r}(x_j) \]

\[ D_\tau(r, \hat{r}) = 3 \]
\[ C-\text{Index}(r, \hat{r}) = \frac{5}{8} \]
Evaluating Multipartite Rankings

C-Index

- the C-index can be rewritten as a weighted sum of pairwise AUCs:

$$\text{C-Index}(r, \hat{r}) = \frac{1}{\sum_{i,j>i} c_i \cdot c_j} \sum_{i,j<i} c_i \cdot c_j \cdot \text{AUC}(r_{i,j}, \hat{r}_{i,j})$$

where $r_{i,j}$ and $\hat{r}_{i,j}$ are the rankings $r$ and $\hat{r}$ restricted to levels $i$ and $j$.

Jonckheere-Terpstra statistic

- is an *unweighted* sum of pairwise AUCs:

$$\text{m-AUC} = \frac{2}{l \cdot (l-1)} \sum_{i,j>i} \text{AUC}(r_{i,j}, \hat{r}_{i,j})$$

- equivalent to well-known multi-class extension of AUC

[Hand & Till, MLJ 2001]
Normalized Discounted Cumulative Gain
[Jarvelin & Kekalainen, 2002]

- The original formulation of (normalized) discounted cumulative gain refers to this setting:
  \[
  DCG(r, \hat{r}) = \sum_{i=1}^{c} \frac{l(i)}{\log(i+1)}
  \]
  - the sum of the true (relevance) levels of the items
  - each item weighted by its rank in the predicted ranking

- Examples:
  - retrieval of relevant or irrelevant pages
    - 2 relevance levels
  - movie recommendation
    - 5 relevance levels
AGENDA

1. Preference Learning Tasks
2. Loss Functions
   a. Evaluation of Rankings
   b. Weighted Measures
   c. Evaluation of Bipartite Rankings
   d. Evaluation of Partial Rankings
3. Preference Learning Techniques
4. Conclusions
Evaluating Partial Structures in the Predicted Ranking

- For fixed types of partial structures, we have conventional measures
  - bipartite graphs → binary classification
    - accuracy, recall, precision, F1, etc.
    - can also be used when the items are labels!
      - e.g., accuracy on the set of labels for multilabel classification
  - multipartite graphs → ordinal classification
    - multiclass classification measures (accuracy, error, etc.)
    - regression measures (sum of squared errors, etc.)

- For general partial structures
  - some measures can be directly used on the reduced set of target preferences
    - Kendall's distance, Gamma coefficient
  - we can also use set measures on the set of binary preferences
    - both, the source and the target ranking consist of a set of binary preferences
    - e.g. Jaccard Coefficient
      - size of intersection over size of union of the binary preferences in both sets
Gamma Coefficient

- Key idea: normalized difference between
  - number of correctly ranked pairs
    (Kendall's distance)
    \[ d = D_\tau(r, \hat{r}) \]
  - number of incorrectly ranked pairs
    \[ \bar{d} = | \{(i, j) | r(x_i) < r(x_j) \land \hat{r}(x_i) < \hat{r}(x_j)\} | \]

- Gamma Coefficient
  [Goodman & Kruskal, 1979]
  \[ \gamma(r, \hat{r}) = \frac{d - \bar{d}}{d + \bar{d}} \in [-1, 1] \]
  - Identical to Kendall's tau if both rankings are total
    - i.e., if \( d + \bar{d} = \frac{c(c-1)}{2} \)
    \[ \gamma(r, \hat{r}) = \frac{2 - 1}{2 + 1} = \frac{1}{3} \]
References

- Kendall, M. *A New Measure of Rank Correlation*. Biometrika 30 (1-2): 81–89 (1938)
- Mann H. B., Whitney D. R. *On a test of whether one of two random variables is stochastically larger than the other*. Annals of Mathematical Statistics, 18:50–60 (1947)
AGENDA

1. Preference Learning Tasks
2. Performance Assessment and Loss Functions
3. Preference Learning Techniques
   a. Learning Utility Functions
   b. Learning Preference Relations
   c. Structured Output Prediction
   d. Model-Based Preference Learning
   e. Local Preference Aggregation
4. Conclusions
**Two Ways of Representing Preferences**

- **Utility-based approach**: Evaluating single alternatives
  
  \[ U : \mathcal{A} \rightarrow \mathbb{R} \]

- **Relational approach**: Comparing pairs of alternatives

  \[ a \succeq b \iff a \text{ is not worse than } b \]

  weak preference

  \[ a \succ b \iff (a \succeq b) \land (b \not\succ a) \]

  strict preference

  \[ a \sim b \iff (a \succeq b) \land (b \succeq a) \]

  indifference

  \[ a \perp b \iff (a \not\succeq b) \land (b \not\succeq a) \]

  incomparability
Utility Functions

- A utility function assigns a utility degree (typically a real number or an ordinal degree) to each alternative.
- Learning such a function essentially comes down to solving an (ordinal) regression problem.
- Often additional conditions, e.g., due to bounded utility ranges or monotonicity properties (→ learning monotone models)

- A utility function induces a ranking (total order), but not the other way around!
- But it can not represent more general relations, e.g., a partial order!
- The feedback can be direct (exemplary utility degrees given) or indirect (inequality induced by order relation):

\[
(x, u) \Rightarrow U(x) \approx u, \quad x > y \Leftrightarrow U(x) > U(y)
\]

absolute feedback \quad relative feedback
Predicting Utilities on Ordinal Scales

(Graded) multilabel classification

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>--</td>
<td>+</td>
<td>++</td>
<td>0</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>0</td>
<td>++</td>
<td>--</td>
<td>+</td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>--</td>
<td>--</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>+</td>
<td>0</td>
<td>++</td>
<td>--</td>
</tr>
<tr>
<td>1.04</td>
<td>0</td>
<td>33</td>
<td>158</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>--</td>
</tr>
</tbody>
</table>

Collaborative filtering

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>...</th>
<th>P38</th>
<th>...</th>
<th>P88</th>
<th>P89</th>
<th>P90</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>1</td>
<td>4</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U2</td>
<td>2</td>
<td>2</td>
<td>...</td>
<td>...</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>2</td>
<td>2</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>2</td>
<td>2</td>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U98</td>
<td>5</td>
<td>...</td>
<td>...</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U99</td>
<td>1</td>
<td>...</td>
<td>...</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Exploiting dependencies (correlations) between items (labels, products, ...)

→ see work in MLC and RecSys communities
A (latent) utility function can also be used to solve ranking problems, such as instance, object or label ranking
→ ranking by (estimated) utility degrees (scores)

Object ranking
(0.74, 1, 25, 165) ≥ (0.45, 0, 35, 155)
(0.47, 1, 46, 183) ≥ (0.57, 1, 61, 177)
(0.25, 0, 26, 199) ≥ (0.73, 0, 46, 185)
(0.95, 0, 73, 133) ≥ (0.25, 1, 35, 153)
(0.68, 1, 55, 147) ≥ (0.67, 0, 63, 182)

Find a utility function that agrees as much as possible with the preference information in the sense that, for most examples,

\[ x_i \succ y_i \iff U(x_i) > U(y_i) \]

Instance ranking

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>--</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td>277</td>
<td>0</td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>--</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>++</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>273</td>
<td>+</td>
</tr>
</tbody>
</table>

Absolute preferences given, so in principle an ordinal regression problem. However, the goal is to maximize ranking instead of classification performance.
Ranking versus Classification

A ranker can be turned into a classifier via thresholding:

$$f(x) > t$$  \hspace{2cm} $$f(x) < t$$

A good classifier is not necessarily a good ranker:

\[ \rightarrow \text{learning AUC-optimizing scoring classifiers!} \]
The idea is to minimize a convex upper bound on the empirical ranking error over a class of (kernelized) ranking functions:

\[ f^* \in \arg \min_{f \in \mathcal{F}} \left\{ \frac{1}{|P| \cdot |N|} \sum_{x \in P} \sum_{x' \in N} L(f, x, x') + \lambda \cdot R(f) \right\} \]

- convex upper bound on \( \mathbb{I} (f(x) < f(x')) \)
- check for all positive/negative pairs
- regularizer
The bipartite RankSVM algorithm [Herbrich et al. 2000, Joachimes 2002]:

\[
f^* \in \arg \min_{f \in \mathcal{F}_K} \left\{ \frac{1}{|P| \cdot |N|} \sum_{x \in P} \sum_{x' \in N} (1 - (f(x) - f(x'))) + \frac{\lambda}{2} \cdot \|f\|_K^2 \right\}
\]

→ learning comes down to solving a QP problem
RankSVM and Related Methods (Bipartite Case)

- The bipartite RankBoost algorithm [Freund et al. 2003]:

\[ f^* \in \arg \min_{f \in \mathcal{L}(\mathcal{F}_{base})} \left\{ \frac{1}{|P| \cdot |N|} \sum_{x \in P} \sum_{x' \in N} \exp\left(-\left(f(x) - f(x')\right)\right) \right\} \]

class of linear combinations of base functions

→ learning by means of boosting techniques
Learning Utility Functions for Label Ranking

Label ranking is the problem of learning a function $\mathcal{X} \rightarrow \Omega$, with $\Omega$ the set of rankings (permutations) of a label set $\mathcal{Y} = \{y_1, y_2, \ldots, y_k\}$, from exemplary pairwise preferences $y_i \succ x y_j$.

Can be tackled by learning utility functions $U_1(\cdot), \ldots, U_k(\cdot)$ that are in appropriate agreement with the preferences in the training data. Given a new query $x$, the labels are ranked according to utility degrees, i.e., a permutation $\pi$ is predicted such that

$$U_{\pi^{-1}(1)}(x) > U_{\pi^{-1}(2)}(x) > \ldots > U_{\pi^{-1}(k)}(x)$$
Label Ranking: Reduction to Binary Classification [Har-Peled et al. 2002]

Proceeding from linear utility functions

\[ U_i(x) = w_i \times x = (w_{i,1}, w_{i,2}, \ldots, w_{i,m})(x_1, x_2, \ldots, x_m)^\top, \]

a binary preference \( y_i \succ x_y j \) is equivalent to

\[ U_i(x) > U_j(x) \iff w_i \times x > w_j \times x \iff (w_i - w_j) \times x > 0 \]

and can be modeled as a linear constraint

\[ (w_1, w_2 \ldots w_k) \times (0 \ldots 0 x 0 \ldots 0 - x 0 \ldots 0)^\top > 0 \]

Each pairwise comparison is turned into a binary classification example in a high-dimensional space!
AGENDA

1. Preference Learning Tasks
2. Performance Assessment and Loss Functions
3. Preference Learning Techniques
   a. Learning Utility Functions
   b. **Learning Preference Relations**
   c. Structured Output Prediction
   d. Model-Based Preference Learning
   e. Local Preference Aggregation
4. Conclusions
Learning Binary Preference Relations

- Learning **binary preferences** (in the form of predicates $P(x,y)$) is often simpler, especially if the training information is given in this form, too.
- However, it implies an additional step, namely **extracting a ranking** from a (predicted) preference relation.
- This step is not always trivial, since a predicted preference relation may exhibit inconsistencies and may not suggest a unique ranking in an unequivocal way.

<table>
<thead>
<tr>
<th>$f_{i,j}$</th>
<th>$y_1$</th>
<th>$y_2$</th>
<th>$y_3$</th>
<th>$y_4$</th>
<th>$y_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$y_2$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$y_3$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$y_4$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$y_5$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**Inference:** $y_4 \succ y_5 \succ y_1 \succ y_3 \succ y_2$
Object Ranking: Learning to Order Things [Cohen et al. 99]

- In a first step, a **binary preference function** $\text{PREF}$ is constructed; $\text{PREF}(x,y) \in [0,1]$ is a measure of the certainty that $x$ should be ranked before $y$, and $\text{PREF}(x,y)=1- \text{PREF}(y,x)$.

- This function is expressed as a linear combination of base preference functions:

$$\text{PREF}(x, y) = \sum_{i=1}^{N} w_i \cdot R_i(x, y)$$

- The weights can be learned, e.g., by means of the weighted majority algorithm [Littlestone & Warmuth 94].

- In a second step, a total order is derived, which is as much as possible in agreement with the binary preference relation.
Object Ranking: Learning to Order Things [Cohen et al. 99]

- The weighted feedback arc set problem: Find a permutation \( \pi \) such that

\[
\sum_{(x, y) : \pi(x) > \pi(y)} \text{PREF}(x, y)
\]

becomes minimal.

\[
\text{cost} = 0.1 + 0.6 + 0.8 + 0.5 + 0.3 + 0.4 = 2.7
\]
Object Ranking: Learning to Order Things [Cohen et al. 99]

- Since this is an NP-hard problem, it is solved heuristically.

\[ \text{Input: an instance set } X; \text{ a preference function } \text{PREF} \]
\[ \text{Output: an approximately optimal ordering function } \hat{\rho} \]
\[ \text{let } V = X \]
\[ \text{for each } v \in V \text{ do} \]
\[ \text{while } V \text{ is non-empty do} \]
\[ \pi(v) = \sum_{u \in V} \text{PREF}(v, u) - \sum_{u \in V} \text{PREF}(u, v) \]
\[ \text{let } t = \arg \max_{u \in V} \pi(u) \]
\[ \hat{\rho}(t) = |V| \]
\[ V = V - \{t\} \]
\[ \text{for each } v \in V \text{ do} \]
\[ \pi(v) = \pi(v) + \text{PREF}(t, v) - \text{PREF}(v, t) \]
\[ \text{endwhile} \]

- The algorithm successively chooses nodes having maximal "net-flow" within the remaining subgraph.

- It can be shown to provide a 2-approximation to the optimal solution.
Label Ranking: Learning by Pairwise Comparison (LPC) [Hüllermeier et al. 2008]

Label ranking is the problem of learning a function $\mathcal{X} \rightarrow \Omega$, with $\Omega$ the set of rankings (permutations) of a label set $\mathcal{Y} = \{y_1, y_2, \ldots, y_k\}$, from exemplary pairwise preferences $y_i \succ x y_j$.

LPC trains a model

$$\mathcal{M}_{i,j} : \mathcal{X} \rightarrow [0, 1]$$

for all $i < j$. Given a query instance $x$, this model is supposed to predict whether $y_i \succ y_j$ ($\mathcal{M}_{i,j}(x) = 1$) or $y_j \succ y_i$ ($\mathcal{M}_{i,j}(x) = 0$).

More generally, $\mathcal{M}_{i,j}(x)$ is the estimated probability that $y_i \succ y_j$.

Decomposition into $k(k-1)/2$ binary classification problems.
## Label Ranking: Learning by Pairwise Comparison (LPC) [Hüllermeier et al. 2008]

Training data (for the label pair A and B):

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th><strong>X4</strong></th>
<th><strong>class</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>0</td>
<td>10</td>
<td>174</td>
<td>1</td>
</tr>
<tr>
<td>1.45</td>
<td>0</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.22</td>
<td>1</td>
<td>46</td>
<td>421</td>
<td>0</td>
</tr>
<tr>
<td>0.74</td>
<td>1</td>
<td>25</td>
<td>165</td>
<td>1</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>72</td>
<td>158</td>
<td>1</td>
</tr>
<tr>
<td>1.04</td>
<td>0</td>
<td>33</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Label Ranking: Learning by Pairwise Comparison (LPC) [Hüllermeier et al. 2008]
Label Ranking: Learning by Pairwise Comparison (LPC) [Hüllermeier et al. 2008]

At prediction time, a query instance is submitted to all models, and the predictions are combined into a binary preference relation:

\[ M_{i,j}(x) \]

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>0.3</td>
<td>0.8</td>
<td>0.4</td>
</tr>
<tr>
<td>B</td>
<td>0.7</td>
<td></td>
<td>0.7</td>
<td>0.9</td>
</tr>
<tr>
<td>C</td>
<td>0.2</td>
<td>0.3</td>
<td></td>
<td>0.3</td>
</tr>
<tr>
<td>D</td>
<td>0.6</td>
<td>0.1</td>
<td>0.7</td>
<td></td>
</tr>
</tbody>
</table>
Label Ranking: Learning by Pairwise Comparison (LPC) [Hüllermeier et al. 2008]

At prediction time, a query instance is submitted to all models, and the predictions are combined into a binary preference relation:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.3</td>
<td>0.8</td>
<td>0.4</td>
<td>1.5</td>
</tr>
<tr>
<td>B</td>
<td>0.7</td>
<td>0.7</td>
<td>0.9</td>
<td>2.3</td>
</tr>
<tr>
<td>C</td>
<td>0.2</td>
<td>0.3</td>
<td>0.3</td>
<td>0.8</td>
</tr>
<tr>
<td>D</td>
<td>0.6</td>
<td>0.1</td>
<td>0.7</td>
<td>1.4</td>
</tr>
</tbody>
</table>

From this relation, a ranking is derived by means of a ranking procedure. In the simplest case, this is done by sorting the labels according to their sum of weighted votes.
AGENDA

1. Preference Learning Tasks
2. Performance Assessment and Loss Functions
3. Preference Learning Techniques
   a. Learning Utility Functions
   b. Learning Preference Relations
   c. Structured Output Prediction
   d. Model-Based Preference Learning
   e. Local Preference Aggregation
4. Conclusions
Structured Output Prediction [Bakir et al. 2007]

- Rankings, multilabel classifications, etc. can be seen as specific types of **structured** (as opposed to scalar) outputs.
- Discriminative structured prediction algorithms infer a **joint scoring function** on input-output pairs and, for a given input, predict the output that maximises this scoring function.
- Joint feature map and scoring function

\[ \phi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^d, \quad f(x, y; w) = \langle w, \phi(x, y) \rangle \]

- The learning problem consists of estimating the weight vector, e.g., using structural risk minimization.
- Prediction requires solving a **decoding problem**:

\[ \hat{y} = \arg\max_{y \in \mathcal{Y}} f(x, y; w) = \arg\max_{y \in \mathcal{Y}} \langle w, \phi(x, y) \rangle \]
Structured Output Prediction [Bakir et al. 2007]

- **Preferences** are expressed through **inequalities** on inner products:

\[
\min_{w, \xi} \|w\|^2 + \nu \sum_{i=1}^{m} \xi_i \\
\text{s.t. } \langle w, \phi(x_i, y_i) \rangle - \langle w, \phi(x_i, y) \rangle \geq \Delta(y_i, y) - \xi_i \text{ for all } y \in \mathcal{Y} \\
\xi_i \geq 0 \quad (i = 1, \ldots, m)
\]

- The potentially huge **number of constraints** cannot be handled explicitly and calls for specific techniques (such as cutting plane optimization)
AGENDA

1. Preference Learning Tasks
2. Performance Assessment and Loss Functions
3. Preference Learning Techniques
   a. Learning Utility Functions
   b. Learning Preference Relations
   c. Structured Output Prediction
   d. Model-Based Preference Learning
   e. Local Preference Aggregation
4. Conclusions
Model-Based Methods for Ranking

- **Model-based approaches** to ranking proceed from specific assumptions about the possible rankings (*representation bias*) or make use of **probabilistic models** for rankings (parametrized probability distributions on the set of rankings).

- In the following, we shall see examples of both type:
  - Restriction to lexicographic preferences
  - Conditional preference networks (CP-nets)
  - Label ranking using the Plackett-Luce model  
    see our talk tomorrow
Learning Lexicographic Preference Models [Yaman et al. 2008]

- Suppose that objects are represented as feature vectors of length $m$, and that each attribute has $k$ values.
- For $n = k^m$ objects, there are $n!$ permutations (rankings).
- A lexicographic order is uniquely determined by
  - a total order of the attributes
  - a total order of each attribute domain
- **Example:** Four binary attributes ($m=4$, $k=2$)
  - there are $16! \approx 2 \cdot 10^{13}$ rankings
  - but only $(2^4) \cdot 4! = 384$ of them can be expressed in terms of a lexicographic order
- [Yaman et al. 2008] present a learning algorithm that explicitly maintains the version space, i.e., the attribute-orders compatible with all pairwise preferences seen so far (assuming binary attributes with 1 preferred to 0). Predictions are derived based on the „votes“ of the consistent models.
Learning Conditional Preference (CP) Networks [Chevaleyre et al. 2010]

Compact representation of a partial order relation, exploiting conditional independence of preferences on attribute values.

Induces partial order relation, e.g.,

(meat, red wine, Italian) > (meat, white wine, Chinese)
(fish, white wine, Chinese) > (fish, red wine, Chinese)
(meat, white wine, Italian) ? (meat, red wine, Chinese)
Learning Conditional Preference (CP) Networks [Chevaleyre et al. 2010]

Compact representation of a partial order relation, exploiting conditional independence of preferences on attribute values.

Training data (possibly noisy):

(meat, red wine, Italian) > (veggie, red wine, Italian)
(fish, white wine, Chinese) > (veggie, red wine, Chinese)
(veggie, white wine, Chinese) > (veggie, red wine, Italian)
...    ...    ...

Drinks:
- meat: red wine > white wine
- veggie: red wine > white wine
- fish: white wine > red wine

Restaurants:
- meat: Italian > Chinese
- veggie: Chinese > Italian
- fish: Chinese > Italian

...
AGENDA

1. Preference Learning Tasks
2. Performance Assessment and Loss Functions
3. Preference Learning Techniques
   a. Learning Utility Functions
   b. Learning Preference Relations
   c. Structured Output Prediction
   d. Model-Based Preference Learning
   e. Local Preference Aggregation see our talk tomorrow
4. Conclusions
Summary of Main Algorithmic Principles

- **Reduction** of ranking to (binary) classification (e.g., constraint classification, LPC)
- **Direct optimization** of (regularized) smooth approximation of ranking losses (RankSVM, RankBoost, ...)
- **Structured output prediction**, learning joint scoring („matching“) function
- Learning parametrized **probabilistic ranking models** (e.g., Plackett-Luce)
- **Restricted model classes**, fitting (parametrized) deterministic models (e.g., lexicographic orders)
- **Lazy learning**, local preference aggregation (lazy learning)
## References

AGENDA

1. Preference Learning Tasks
2. Performance Assessment and Loss Functions
3. Preference Learning Techniques
4. Conclusions
Conclusions

- Preference learning is an **emerging subfield** of machine learning, with many **applications** and **theoretical challenges**.

- Prediction of **preference models** instead of scalar outputs (like in classification and regression), hitherto with a focus on **rankings**.

- Many existing machine learning problems can be cast in the framework of preference learning (→ preference learning „in a broad sense“)

- „**Qualitative“ alternative** to conventional numerical approaches
  - pairwise comparison instead of numerical evaluation,
  - order relations instead of individual assessment.

- Still many **open problems** (unified framework, predictions more general than rankings, incorporating numerical information, etc.)

- **Interdisciplinary field**, connections to many other areas.
Connections to Other Fields

- Structured Output Prediction
- Learning Monotone Models
- Ordinal Classification
- Ranking in Information Retrieval
- Multilabel Classification
- Recommender Systems
- Economics & Decision Theory
- Operations Research
- Multiple Criteria Decision Making
- Social Choice

Preference Learning
Edited Book on Preference Learning

- Preference Learning: An Introduction
  A Preference Optimization based Unifying Framework for Supervised Learning Problems

**Part I – Label Ranking**
- Label Ranking Algorithms: A Survey
- Preference Learning and Ranking by Pairwise Comparison
- Decision Tree Modeling for Ranking Data
- Co-regularized Least-Squares for Label Ranking

**Part II – Instance Ranking**
- A Survey on ROC-Based Ordinal Regression
- Ranking Cases with Classification Rules

**Part III – Object Ranking**
- A Survey and Empirical Comparison of Object Ranking Methods
- Dimension Reduction for Object Ranking
- Learning of Rule Ensembles for Multiple Attribute Ranking Problems

**Part IV – Preferences in Multiattribute Domains**
- Learning Lexicographic Preference Models
- Learning Ordinal Preferences on Multiattribute Domains: the Case of CP-nets
- Choice-Based Conjoint Analysis: Classification vs. Discrete Choice Models
- Learning Aggregation Operators for Preference Modeling

**Part V – Preferences in Information Retrieval**
- Evaluating Search Engine Relevance with Click-Based Metrics
- Learning SVM Ranking Function from User Feedback Using Document Metadata and Active Learning in the Biomedical Domain

**Part VI – Preferences in Recommender Systems**
- Learning Preference Models in Recommender Systems
- Collaborative Preference Learning
- Discerning Relevant Model Features in a Content-Based Collaborative Recommender System
Edited Book on Preference Learning

Preference Learning: An Introduction
A Preference Optimization based Unifying Framework for Supervised Learning Problems

Part I – Label Ranking
Label Ranking Algorithms: A Survey
Preference Learning and Ranking by Pairwise Comparison
Decision Tree Modeling for Ranking Data
Co-regularized Least-Squares for Label Ranking

Part II – Instance Ranking
A Survey on ROC-Based Ordinal Regression
Ranking Cases with Classification Rules

Part III – Object Ranking
A Survey and Empirical Comparison of Object Ranking Methods
Dimension Reduction for Object Ranking
Learning of Rule Ensembles for Multiple Attribute Ranking Problems

Part IV – Preferences in Multiattribute Domains
Learning Lexicographic Preference Models
Learning Ordinal Preferences on Multiattribute Domains: the Case of CP-nets
Choice-Based Conjoint Analysis: Classification vs. Discrete Choice Models
Learning Aggregation Operators for Preference Modeling

Part V – Preferences in Information Retrieval
Evaluating Search Engine Relevance with Click-Based Metrics
Learning SVM Ranking Function from User Feedback Using Document Metadata and Active Learning in the Biomedical Domain

Part VI – Preferences in Recommender Systems
Learning Preference Models in Recommender Systems
Collaborative Preference Learning
Discerning Relevant Model Features in a Content-Based Collaborative Recommender System

J. Fürnkranz & E. Hüllermeier (eds.)
Preference Learning
Springer-Verlag 2010

includes several introductions and survey articles
Preference Learning Website

http://www.preference-learning.org/

- Working groups
- Software
- Data Sets
- Workshops
- Tutorials
- Books
- ...

http://www.preference-learning.org/