Extreme Data Mining: the Killer App for Metalearning?

Carlos Soares
csoares@fep.up.pt
MAP-I 2007
Trading in the Stock Exchange: The Machine Learning Way

- Goal: decide whether to buy or sell shares

\[ f : \text{FinancialVariablesAndOthers} \rightarrow \{1 = \text{buy}, -1 = \text{sell}\} \]

- Table of data

\[
\begin{array}{cccccc}
   i & x_{i,1} & x_{i,2} & x_{i,3} & x_{i,4} & \text{decision} \\
   1 & 0.7 & 327.2 & 0 & 5 & -1 \\
   2 & -0.6 & 1234.2 & 1 & 4 & 1 \\
   3 & \ldots & \ldots & \ldots & \ldots & \ldots \\
\end{array}
\]

\[
\text{decision} = 1.04 + 0.38 \times x_1 + \ldots
\]

\[
\begin{array}{cccccc}
   -0.8 & 37.2 & 1 & 15 & ? \\
   0.2 & 14.32 & 1 & 9 & ? \\
   \ldots & \ldots & \ldots & \ldots & \ldots \\
\end{array}
\]
Problem of Algorithm Selection

WHICH ONE TO USE?
and then some…
Plan: Part I

PART I

• Background: why is this a problem?
  – approximating functions with machine learning
  – algorithm selection and bias
  – a few solutions

• Meta-Learning: THE solution

PART II

• Meta-learning for Algorithm Recommendation
Problem (1/2)

- Quantifiable phenomena...

\[ y = 2^x \]

\[ \text{black: } (x > 0.5) \text{ XOR } (y > 0.5) \]
Problem (2/2)

- ... known only through samples
Applications of Machine Learning

- Based on data samples

- ... model
  - which function describes the relationship between x and y?
  - how to describe the area in black as a function of x and y?

- ... predict
  - which is the value of y given x=0.35?
  - which is the color of the point (x=0.3, y=0.7)?
Learning: Summary

- Find the function (model) that best fits the data sample

\[ y = ?(x) \]

\[ y = 2^x \]

\[ \text{color: } ?(x, y) \]

\[ \text{black: } (x > 0.5) \text{ XOR } (y > 0.5) \]
Issues (1/2)

- Representativeness of sample

![Decision Tree Diagram]

**decision tree**
black: $x < 0.57$ AND $y > 0.67$

![Black XOR Diagram]

black: $(x > 0.5) \text{ XOR } (y > 0.5)$
Issues (2/2)

- Noise

black: (x > 0.5) XOR (y > 0.5)
Machine Learning Algorithms: Examples

- Neural Networks
- Support Vector Machines
- Decision trees
- Rule induction
- Linear Discriminants
- Naive Bayes
- k-Nearest Neighbors

WHAT IS THE DIFFERENCE?
Bias

- Criteria to select one from a set of models
  - adapted from des Jardins e Gordon (95)
  - which is extended from Mitchell (80, 90), which did not include error

- Types of bias
  - representation: hypotheses space
  - procedural: search algorithm

- Corollary
  - given a data sample and a learning algorithm
  - ... not every model is possible
Example: Bias of ID3

- Top-Down Induction of Decision Trees
  - Quinlan (86)

- Hypotheses space: DNF expressions
  - disjunction of conjunctions
  - ... defining hyperplanes that are orthogonal to the axes

- Search algorithm: Top-Down Induction
  - start with simple models (large hyperplanes)
  - increase complexity gradually (smaller hyperplanes)
Example: ID3 is Suitable

sample: 100 examples
Example: ID3 is Not Suitable
Types of Hypotheses Spaces (According to Langley - 2000)

- Decision trees
- Logical rules
- Cases
- Neural networks
- Probabilistic descriptions
Choice of Algorithm: Summary

- Limits to the models that may be obtained from a data sample using any algorithm
  - can be successful
  - ... or not
Solution I: Bias-Free Algorithm

- Bias-free learning is futile (Mitchell 97, Ch. 2)
  - an algorithm that assumes nothing concerning the function it is trying to learn has no rational basis to classify unknown cases

- Going back to the definition of bias
  - criteria to prefer one model relative to another

- ... and the goal of learning
  - find function (model) that best fits a data sample

- ... how to select the best model if all models are considered equally suitable?
Solution II: The Hard Way

- Test all algorithms
  - pick the one with the best results

- Computationally impossible
  - many algorithms
  - ... most with several parameters
  - ... limited time
Solution III: The Chosen One

- Experimental study
  - a few algorithms (a few parameter settings)
  - ... a few problems
  - ... pick best
  - ... use always

- No-Free Lunch Theorem
  - Wolpert (96)
  - the mean error of all algorithms for all problems in the universe is the same (assuming no information about the problem is used)

- ... limited practical value

- ... but the corollary is verified in practice
  - empirical studies show that “the chosen one” does not exist
Solution IV: Meta-Learning

Learning by experience when different biases are suitable for particular problems

Rendell, Seshu e Tcheng (1987)

- Biases, biases... not really
  - ... implementations of biases
  - ... or algorithms

- There are other definitions
Solutions for Algorithm Selection: Summary

- Bias is necessary
- Trying all alternatives is not possible
- Choice is important
- Meta-learning: learning models to select which algorithm to use for which problems
  - and more...
Plan: Part I

PART I

• Background: why is this a problem?

• Meta-Learning: **THE solution**
  – difference between base-level and meta-level learning
  – different meta-learning approaches

PART II

• Meta-learning for Algorithm Recommendation
The Meta-Learning Picture
Said Differently, Meta-Learning...

- Applying learning methods to model the relationship between the characteristics of learning problems and the suitability of biases

- [Meta-]data
  - sample: learning problems
    - points
  - problem characteristics
    - $x$ and $y$
  - suitability of biases
    - black or white

![Graph showing suitability of bias: $\bar{p}(x, y)$]
Base vs. Meta: Examples

**Base**
- individuals of interest in the domain
  - e.g. patients; clients

**Meta**
- learning problems
  - e.g. medical diagnosis of disease $x$ in hospital $y$; direct marketing for company $z$

$\text{target} = 1.04 \times x_1 + 0.38 \times x_1 + \cdots$
# Base vs. Meta: Target Variable

<table>
<thead>
<tr>
<th>i</th>
<th>$x_{i,1}$</th>
<th>$x_{i,2}$</th>
<th>target</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

**Base**
- problem-specific decision
  - e.g. diagnosis; send catalog or not

**Meta**
- algorithm
  - e.g. decision trees, MLP

\[ \text{target} = 1.04 \times x_1 + 0.38 \times x_2 + \cdots \]
Base vs. Meta: Independent Variables

- Different characteristics of individuals
  - e.g. job, age and income of person; price and type of product

- Problem characteristics and suitability measures
  - e.g. number of variables, number of classes and classification error

\[
\text{target} = 1.04 \times x_1 + 0.38 \times x_1 + \cdots
\]
Base vs. Meta: Problem

<table>
<thead>
<tr>
<th>i</th>
<th>$x_{i,1}$</th>
<th>$x_{i,1}$</th>
<th>target</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

Base
- relationships between variables from a domain
  - e.g. individual profile and income; symptoms and diagnosis

Meta
- relationship between problem characteristics and suitability of biases
  - e.g. sample morphology and performance of algorithms

$target = 1.04 \times x_1 + 0.38 \times x_1 + \cdots$

Train

Apply
Meta-Learning Approaches

• Type of goal
  – continuous adaptation of models/algorithms
  – model construction
  – model combination
  – algorithm selection

• Other classifications are possible
Continuous Adaptation of Models/Algorithms

• Change bias while solving a problem and across different problems
  – also known as “knowledge transfer” or “learning to learn”

• Example: Self-Modifying Policies
  – Schmidhuber, Zhao and Schraudolph (97)
  – parts of the algorithm are able to change other parts of the same algorithm
  – ... even themselves
  – probability of application of a part depends on its past merit
  – illustrated with a reinforcement learning algorithm
  – ... assumes problems are similar
Model Construction

- Build a model made of parts with different biases

Example: Model Class Selection
  - Brodley (93)
    - recursive partitioning algorithm (decision trees-alike)
    - choice of better bias at each node
    - tested with 3 different biases
    - ... no real learning at the meta-level: choice is based on fixed rules
Model Combination

- Meta-model combining several different base models

- Examples: Stacked Generalization/Cascade Generalization
  - Wolpert (92)/Gama and Brazdil (00)
  - meta-dataset including the predictions made by the base models
  - meta-model obtained by learning with the meta-dataset
  - possibly recursive
Algorithm Selection

• Choosing one (or more) algorithms for a given problem

• Examples: algorithm recommendation
  – Rendell, Seshu and Tcheng (87)
    • Variable Bias Management System
  – Aha (92)
    • generalization from case studies
  – Brazdil, Gama and Henery (94)
    • systematic approach
    • as part of the StatLog Project
  – METAL project (02 – many publications by many authors)
    • follow-up to the work on the StatLog project
    • Data Mining Advisor website
      – www.metal-kdd.org
Plan: Part II

PART I

• Background: why is this a problem?
• Meta-Learning: THE solution

PART II

• Meta-learning for Algorithm Recommendation
  – context & goals
  – ranking methodology based on the k-NN
  – evaluation methodology
  – results on the problem of recommending classification algorithms
• Discussion
• Recommendation of Parameter Settings of SVM
Application Context: Data Mining

• Many alternative methods
  • common approach is experimentation

Goal of algorithm recommendation: save time/computational resources with minimal loss in the quality of results
**Research Context: Meta-Learning**

The goal of meta-learning is to accurately predict the relative performance of algorithms (i.e., ranking).

<table>
<thead>
<tr>
<th>i</th>
<th>$x_{i,1}$</th>
<th>$x_{i,1}$</th>
<th>target</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

Apply

Train

Goal of **meta-learning**: accurately predict the relative performance of algorithms (i.e., ranking)
Data Preparation

- Define alternatives
  - 10 classification algorithms

- Define examples
  - 57 problems mostly from the UCI repository

- Obtain performance meta-data
  - run all algorithms on all the data sets
  - computationally intensive...
  - but time to obtain results is not critical
  - measure classification accuracy

- Characterize data sets
  - meta-features: # examples, # continuous variables, etc.
**k-NN Ranking Method**

Given new data set
1. characterize it
   - e.g., # attributes, # examples
2. select $k$ nearest neighbors
3. retrieve performance information
   - e.g., accuracy
4. build **recommended ranking** by aggregating performance information
**k-NN Ranking Method:** Ranking Aggregation Method

- Solve conflicts between the $k$ selected rankings
- For instance, Average Ranks
  - for each selected data set $p$, calculate ranking of algorithms
  - average rank of each algorithm $i$ across all selected data sets

$$
\bar{R}_i = \frac{1}{k} \sum_{p=1}^{k} R_{p,i}
$$

- rank algorithms according to their average rank
## $k$-NN Ranking Method: Example

- **Recommendation for the letter data set using 3-NN**

### Data Sets

<table>
<thead>
<tr>
<th>data sets</th>
<th>bC5</th>
<th>C5r</th>
<th>C5t</th>
<th>MLP</th>
<th>RBFN</th>
<th>LD</th>
<th>Lt</th>
<th>IB1</th>
<th>NB</th>
<th>RIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>byzantine</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>isole</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>1</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>pendigits</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>10</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>9</td>
<td>5</td>
</tr>
</tbody>
</table>

### Algorithms

<table>
<thead>
<tr>
<th>ranks</th>
<th>bC5</th>
<th>C5r</th>
<th>C5t</th>
<th>MLP</th>
<th>RBFN</th>
<th>LD</th>
<th>Lt</th>
<th>IB1</th>
<th>NB</th>
<th>RIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicted</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>10</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>
Evaluation of Methods to Predict Rankings

Test data set

C5b → 86%
Lt → 91%
...

Training data sets

C5b → 82%
Lt → 73%
...

Target ranking

Lt
C5b
...

Score

C5b
Lt
...

Predicted ranking

Score
Measuring Ranking Accuracy

- Target ranking
  - representing performance of algorithms on the “new” data set
- Spearman’s Rank Correlation Coefficient
  - values range from $-1$ to $1$

\[
r_S = 1 - \frac{6 \sum_{i=1}^{n} (\hat{R}_i - R_i)^2}{n^3 - n}
\]

<table>
<thead>
<tr>
<th>ranks</th>
<th>bC5</th>
<th>C5r</th>
<th>C5t</th>
<th>MLP</th>
<th>RBFN</th>
<th>LD</th>
<th>Lt</th>
<th>IB1</th>
<th>NB</th>
<th>RIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicted</td>
<td>1.5</td>
<td>5.5</td>
<td>7</td>
<td>9</td>
<td>10</td>
<td>4</td>
<td>3</td>
<td>1.5</td>
<td>5.5</td>
<td>8</td>
</tr>
<tr>
<td>target</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>9</td>
<td>6</td>
</tr>
</tbody>
</table>

\[r_S = 0.709\]
Default Ranking

• Baseline
  • simple method
  • assess whether ranking method is finding useful patterns

• Default ranking
  • apply ranking aggregation method on all the rankings

<table>
<thead>
<tr>
<th>Ranks</th>
<th>bC5</th>
<th>C5r</th>
<th>C5t</th>
<th>MLP</th>
<th>RBFN</th>
<th>LD</th>
<th>Lt</th>
<th>IB1</th>
<th>NB</th>
<th>RIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>7</td>
<td>10</td>
<td>8</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>target</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>9</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ r_S = 0.879 > 0.709 \]
$k$-NN vs. Default Ranking: Ranking Accuracy Results

- Baseline default ranking
  - fixed prediction…
  - but quite accurate
- $k$-NN more accurate than DR
  - small $k$
- Significance of differences between methods
  - Friedman’s and Dunn’s tests

Possible to predict the relative performance of algorithms
Measuring Value of Recommended Rankings

- Accurate rankings are not necessarily useful
  - goal is to save time with minimal loss in (classification) accuracy

- Value depends on the use
  - order defined by ranking is followed…
  - but number of algorithms executed is not known beforehand

- Top-N evaluation
  - best accuracy of top-N algorithms in the ranking
  - total cost of executing them

<table>
<thead>
<tr>
<th>recommendation for letter</th>
<th>bC5</th>
<th>IB1</th>
<th>LT</th>
<th>LD</th>
<th>C5R</th>
<th>NB</th>
<th>...</th>
<th>RBFN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>81%</td>
<td>87%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>56s</td>
<td>81s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>88%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>478s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### k-NN vs. Default Ranking: Top-N Results

<table>
<thead>
<tr>
<th></th>
<th>Acc. (%)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>88.3</td>
<td>2.5 h</td>
</tr>
<tr>
<td>Top-1</td>
<td>86.8 (-1.5)</td>
<td>11 min</td>
</tr>
<tr>
<td>Top-2</td>
<td>87.5 (-0.8)</td>
<td>20 min</td>
</tr>
<tr>
<td>bC5</td>
<td>86.5 (-1.8)</td>
<td>8 min</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Mean accuracy</th>
<th>Mean execution time (log s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Classification accuracy × nb. of algorithms</td>
<td>Classification accuracy × execution time</td>
</tr>
<tr>
<td></td>
<td>[1-NN]</td>
<td>[-2-NN]</td>
</tr>
</tbody>
</table>

- **Difficult problem!**
- Possible to save significant amounts of time with small loss in accuracy
- Ranking is more suitable for algorithm recommendation
Ranking According to Accuracy and Time

- Incorporate knowledge about the goal into the recommendation method
  - save time with minimal loss in (classification) accuracy

- Adjusted Ratio of Ratios

\[
ARR_{i,j} = \frac{SR_i}{SR_j} \frac{1}{1 + AccD \times \log\left(\frac{T_i}{T_j}\right)}
\]

- Parameter defining relative importance of accuracy and time
  - \(AccD\) = the accuracy the user is willing to trade for a 10 times speedup
**k-NN with ARR: Top-N Results**

- **Better results sooner**
  - less accurate algorithms at the top…
  - but more algorithms are executed

*Problem-specific knowledge is more important than the choice of aggregation method*
Plan: Part II

PART I

• Background: why is this a problem?
• Meta-Learning: THE solution

PART II

• Meta-learning for Algorithm Recommendation
• Discussion
  – pre-selection of alternatives
  – obtaining problems
  – characterization of problems
  – meta-accuracy
  – other applications
• Recommendation of Parameter Settings of SVM
Pre-selection of Alternatives

• Which algorithms?
  – the ones in available tools
  – constraints on acceptable models
    • e.g. understandability
  – preferences of the data analyst
    • ... or ignorance concerning others

• Which parameters?
  – domains are frequently infinite
    • continuous parameters

• Is the selected set adequate?
Pre-selection of Alternatives: Overall Relevance

- For most data sets there should be an alternative that obtains an error which is lower than the error of a given baseline.
Given some pre-selected set of alternatives, the results cannot be further significantly improved by adding additional ones.
Pre-selection of Alternatives: Individual Competitiveness

For every alternative, we should be able to identify at least one data set for which it is the best one, from the pre-selected set.
Pre-selection of Alternatives: Individual Relevance

• For every alternative, there should not exist another one such that the performance of former is never significantly better than that of latter for all data sets considered
  – each setting is significantly better than each of the others on at least one data set
Quantity/Quality of Meta-data

- Meta-models are based on a few dozen problems
  - small sample
  - results are positive but are they stable?

- How to generate more meta-data?
  - random methods are not suitable

```
universe

real problems
```
Simulated Applications using Real Data

- Get real data
  - any source is fine
  - e.g. transactions, time between events

- Simulate applications
  - one problem for each variable
  - ... corresponding value may not make any sense

- Goal
Characterization of Datasets

- Meta-Dataset

\[
\begin{array}{ccc}
1 & x_{i,1} & x_{i,1} \\
2 & . & . \\
3 & . & . \\
\end{array}
\]

Train

\[
\begin{array}{ccc}
1 & x_{i,1} & x_{i,1} & \text{target} \\
2 & . & . & . \\
3 & . & . & . \\
\end{array}
\]

\[
\begin{align*}
target &= 1.04 \times x_1 + 0.38 \times x_1 + \ldots
\end{align*}
\]

Apply

\[
\begin{array}{ccc}
1 & x_{i,1} & x_{i,1} & \text{target} \\
2 & . & . & . \\
3 & . & . & . \\
\end{array}
\]

meta-model: Datasets \rightarrow Performance
Meta-Features

meta-model: \( c(\text{DataSets}) \rightarrow \text{Performance} \)

- \( c \) is a mapping between a matrix of values of variable size and type and a set of values of fixed size.

<table>
<thead>
<tr>
<th>( i )</th>
<th>( x_{i,1} )</th>
<th>( x_{i,2} )</th>
<th>( x_{i,3} )</th>
<th>( x_{i,4} )</th>
<th>decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.7</td>
<td>327.2</td>
<td>0</td>
<td>5</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>-0.6</td>
<td>1234.2</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\( c \rightarrow 2399.49, 1, 0.65, ... \)

max. correlation continuous-target

examples

continuous variables

symbolic variables
Good Meta-features

- Measures that potentially contain information about the relative performance of algorithms
- ... but are computationally cheaper than the algorithms
- probably the hardest problem!
Approaches to Characterize Problems

- General, Statistical and Information-theoretic meta-features
  - à la StatLog
  - ex. # attributes, proportion of numeric attributes with outliers, class entropy

- Landmarkers
  - Bensusan and Giraud-Carrier (2000)
  - results of simple algorithms used to predict the performance of more complex ones
  - subsampling landmarkers: results of complex algorithms on subsamples of the data
    - Furnkranz and Petrak (01), Soares, Petrak and Brazdil (01)

- Model-based
  - properties of an induced model
Meta-Feature Selection

• Adequate choice of meta-features is essential
  • $k$-NN algorithm assigns the same weight to all variables
  • measures that potentially contain information about the relative performance of algorithms

• Knowledge-based approach
  • identify properties of the data
  • select/design meta-features representing those properties
Meta-Feature Selection: Classification

- **Set used so far**

<table>
<thead>
<tr>
<th>Property</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability</td>
<td># examples</td>
</tr>
<tr>
<td>Nominal vs. numeric attributes</td>
<td>proportion of symbolic attributes</td>
</tr>
<tr>
<td>Robustness to missing values</td>
<td>proportion of missing values</td>
</tr>
<tr>
<td>Robustness to outliers</td>
<td>proportion of numeric attributes with outliers</td>
</tr>
<tr>
<td>Number of classes</td>
<td>class entropy</td>
</tr>
<tr>
<td>Frequency of classes</td>
<td>mean mutual information of class and attributes</td>
</tr>
<tr>
<td>Information in nominal attributes</td>
<td>canonical correlation of the most discriminating</td>
</tr>
<tr>
<td></td>
<td>single linear combination of numeric attributes</td>
</tr>
<tr>
<td></td>
<td>and the class distribution</td>
</tr>
</tbody>
</table>

- **Other properties and measures could be used**
Results with Selected Classification Meta-Features

- Significantly better than previous sets of meta-features
- general, statistical and information-theoretic measures

Informal method for meta-feature selection/design
Cost of Data Characterization

- Gains in execution time achieved by executing less alternatives compensate for data characterization time
  - approximately mean execution time of single algorithm
Weighted Ranking Accuracy

- The highest the rank, the more harmful the error
  - top-ranked algorithms are selected more often
- Weighted Rank Correlation Coefficient

\[
r_W(X, Y) = 1 - \frac{6 \sum_{i=1}^{n} (R(X_i) - R(Y_i))^2 (2n + 1 - R(X_i) - R(Y_i))}{n^4 + n^3 - n^2 - n}
\]

- \(r_W\) yields values quite different from \(r_S\) in some cases…
  - up to 0.1 just by swapping a pair of ranks
  - but similar results in the algorithm recommendation problem
Log Ranking Accuracy

- The highest the rank, the more harmful the error
  - top-ranked algorithms are selected more often

- Log Ranking Accuracy Measure

$$r_{log}(X, Y) = 1 - \frac{2 \sum_{i=1}^{n} \log_{1+R}(X_i) \left(1 + (R(X_i) - R(Y_i))\right)^2}{\sum_{i=1}^{n} \log_{1+i} \left(1 + (i - (n - 1 + 1))\right)^2}$$

- Complementary information

  - \(k\)-NN makes fewer errors at top ranks than the default ranking

Measure of ranking accuracy that assigns more importance to algorithms that are most likely to be selected
Other Applications

• Tested
  – regression
  – recommendation of parameters for pre-processing methods

• Current work
  – time series
  – outlier detection
  – optimization

• Future work
  – pre-processing methods + algorithm + parameters
Plan: Part II

PART I

• Background: why is this a problem?
• Meta-Learning: THE solution

PART II

• Meta-learning for Algorithm Recommendation
• Discussion
• Recommendation of Parameter Settings of SVM
• Support Vector Machines
  – linear learning machines that maximize the margin
  – duality
  – kernel trick

• According to Bennet & Campbell, “Support Vector Machines: Hype of Hallelujah?”, SIGKDD Explorations, 2000
  – geometrical intuition
  – elegant math
  – theoretical guarantees
  – practical (and successful) algorithms

• Successful but...
  – heavy tuning usually required
\(k\)-NN Ranking for Parameter Setting

- **Goal:** test methodology on different problem
- **Application:** Support Vector Machines for regression
  - width of the Gaussian kernel, \(\sigma\)
  \[
  K(x_i, x_j) = e^{-\frac{|x_i - x_j|^2}{2\sigma^2}}
  \]
- **Pre-selection of a set values required**
  - continuous parameter
  - set of 11 \(\sigma\) values
- **Pre-selected set valid?**
  - (explained earlier)
$k$-NN Ranking vs. Default Ranking: Parameter Setting Results

- More accurate rankings than DR
- Significantly more accurate algorithms at the top ranks
**k-NN Ranking vs. Jaakkola’s Heuristic: Results**

- Heuristic commonly used in SVMs
  - for all examples \( x \)
    \[
    d_x = \min_y \left( \sqrt{\sum_i (x_i - y_i)^2} \right)
    \]
  - set width to
    \[
    \sigma = \overline{d_x}
    \]

- **k-NN achieves better results**
  - also more robust
  - top-5 results confirms advantage of ranking
  - heuristic obtains surprisingly good results
The Kernel Trick

- Kernels project data into a (potentially) higher-dimensionality space
  - potentially infinite
  - a complex pattern-classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in low-dimensional space (Cover 95)
  - “separable” -> “fit”

\[ \Phi : \circ \rightarrow \triangle \]

- ... implicitly
  - algorithms use a matrix representing the distances between the examples in projected space
Meta-Features Based on the Kernel Matrix

- Kernel matrix represents distance of examples in projected space
- Design meta-features based on the values of the kernel matrix
  - off-diagonal values are close to 0 when examples are “isolated”
    - MF₁: mean of off-diagonal values
  - off-diagonal values should vary when there is structure
    - MF₂: variance of off-diagonal values
  - measures “correlation” between a kernel function and ideal kernel
    - MF₃: kernel-target alignment
- Calculate for all parameter settings
Results with Selected Kernel Meta-Features

- Significantly improved results with the set of Kernel meta-features
  - unexpected results with the combination of the two sets

- Successful adaptation of the methodology to a different problem
- Successful design of problem-specific meta-features
Conclusions

- Relating characteristics of problems to (relative) performance of learning algorithms is possible

- Recommendation is not the only goal
  - understanding behavior of algorithms
  - insights leading to improvements/new algorithms

- Research on meta-learning is at an early stage
  - data characterization
  - insufficient examples (i.e. data sets)

- Learning to predict rankings is interesting too!
(Very) Short Bibliography
somewhat biased too...

• L. Rendell, R. Seshu and D. Tcheng (87), More Robust Concept Learning Using Dynamically-Variable Bias, in Proc. of the Fourth Int. Workshop on Machine Learning, pp 66-78


• C. Soares, P. Brazdil and P. Kuba (04), A Meta-Learning Method to Select the Kernel Width in Support Vector Regression, Machine Learning, 54, pp 195-20

• C. Soares (03), Is the UCI Repository Useful for Data Mining?, Proceedings of the 11th Portuguese Conference on Artificial Intelligence (EPIA2003), LNAI 2902, pp 209-223
Acknowledgements

- Pavel Brazdil and Joaquim Costa (supervisors)
- Petr Kuba
- Rui Pereira and Adam Woznica
  - www.metal-kdd.org
- METAL project