Rule-based Regression
Frederik Janssen
Outline

1. Machine Learning
   ▶ Classifiers
   ▶ Data Representation
   ▶ Concept Representation

2. Separate-and-conquer Rule Learning

3. Regression
   ▶ Regression by classification
   ▶ Regression measures

4. Current implementation

5. Some results

6. Discussion
Definition (Mitchell, 1997)

“A computer program is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with experience $E$.”

Given:
- a task $T$
- a performance measure $P$
- some experience $E$ with the task

Goal:
- generalize the experience in a way that allows to improve your performance on the task
Indroduction of Classifiers

**Inductive Machine Learning**

Inductive Machine Learning algorithms induce a classifier from *labeled* training examples. The classifier generalizes the training examples, i.e., it is able to assign labels to new cases.

An inductive learning algorithm searches in a given family of hypotheses (e.g., decision trees, neural networks) for a member that optimizes given quality criteria (e.g., estimated predictive accuracy or misclassification costs).
Introduction of Classifiers

The most “popular” learning problem:

- **Task:**
  - learn a model that predicts the outcome of a dependent variable for a given instance

- **Experience:**
  - experience is given in the form of a data base of examples
  - an example describes a single previous observation
    - *instance*: a set of measurements that characterize a situation
    - *label*: the outcome that was observed in this situation

- **Performance Measure:**
  - compare the predicted outcome to the observed outcome
  - estimate the probability of predicting the right outcome in a new situation
Data Representation

Attribute-Value Data

- Each example is described with values for a fixed number of attributes (also called features)
  - **Nominal Attributes:**
    - store an unordered list of symbols (e.g., *color*)
  - **Numeric Attributes:**
    - store a number (e.g., *income*)
A sample task

<table>
<thead>
<tr>
<th>Day</th>
<th>Temperature</th>
<th>Outlook</th>
<th>Humidity</th>
<th>Windy</th>
<th>Play Golf?</th>
</tr>
</thead>
<tbody>
<tr>
<td>07-05</td>
<td>26</td>
<td>sunny</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>07-06</td>
<td>28</td>
<td>sunny</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-07</td>
<td>29</td>
<td>overcast</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-09</td>
<td>23</td>
<td>rain</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-10</td>
<td>20</td>
<td>overcast</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-12</td>
<td>12</td>
<td>sunny</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>07-14</td>
<td>8</td>
<td>sunny</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-15</td>
<td>25</td>
<td>rain</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-20</td>
<td>18</td>
<td>sunny</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-21</td>
<td>18</td>
<td>overcast</td>
<td>high</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-22</td>
<td>20</td>
<td>overcast</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-23</td>
<td>19</td>
<td>rain</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-26</td>
<td>11</td>
<td>rain</td>
<td>normal</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-30</td>
<td>16</td>
<td>rain</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
</tbody>
</table>

today  9  sunny  normal  false  ?  
tomorrow  13  sunny  normal  false  ?
A sample task

<table>
<thead>
<tr>
<th>Day</th>
<th>Temperature</th>
<th>Outlook</th>
<th>Humidity</th>
<th>Windy</th>
<th>Play Golf?</th>
</tr>
</thead>
<tbody>
<tr>
<td>07-05</td>
<td>26</td>
<td>sunny</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>07-06</td>
<td>28</td>
<td>sunny</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-07</td>
<td>29</td>
<td>overcast</td>
<td>high</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-09</td>
<td>23</td>
<td>rain</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-10</td>
<td>20</td>
<td>overcast</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-12</td>
<td>12</td>
<td>sunny</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>07-14</td>
<td>8</td>
<td>sunny</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-15</td>
<td>25</td>
<td>rain</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-20</td>
<td>18</td>
<td>sunny</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-21</td>
<td>18</td>
<td>overcast</td>
<td>high</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-22</td>
<td>20</td>
<td>overcast</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-23</td>
<td>19</td>
<td>rain</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-26</td>
<td>11</td>
<td>rain</td>
<td>normal</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-30</td>
<td>16</td>
<td>rain</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
</tbody>
</table>

possible rules:

- \(\text{play}=\text{no} \leftarrow \text{temperature} \geq 25.5 \land \text{temperature} < 28.5\)
- \(\text{play}=\text{no} \leftarrow \text{temperature} < 14 \land \text{temperature} \geq 9.5\)
- \(\text{play}=\text{no} \leftarrow \text{outlook}=\text{rainy} \land \text{windy}=\text{true}\)

<table>
<thead>
<tr>
<th>Day</th>
<th>Temperature</th>
<th>Outlook</th>
<th>Humidity</th>
<th>Windy</th>
<th>Play Golf?</th>
</tr>
</thead>
<tbody>
<tr>
<td>today</td>
<td>9</td>
<td>sunny</td>
<td>normal</td>
<td>false</td>
<td>?</td>
</tr>
<tr>
<td>tomorrow</td>
<td>13</td>
<td>sunny</td>
<td>normal</td>
<td>false</td>
<td>?</td>
</tr>
</tbody>
</table>
A sample task

<table>
<thead>
<tr>
<th>Day</th>
<th>Temperature</th>
<th>Outlook</th>
<th>Humidity</th>
<th>Windy</th>
<th>Play Golf?</th>
</tr>
</thead>
<tbody>
<tr>
<td>07-05</td>
<td>26</td>
<td>sunny</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>07-06</td>
<td>28</td>
<td>sunny</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-07</td>
<td>29</td>
<td>overcast</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-09</td>
<td>23</td>
<td>rain</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-10</td>
<td>20</td>
<td>overcast</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-12</td>
<td>12</td>
<td>sunny</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>07-14</td>
<td>8</td>
<td>sunny</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-15</td>
<td>25</td>
<td>rain</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-20</td>
<td>18</td>
<td>sunny</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-21</td>
<td>18</td>
<td>overcast</td>
<td>high</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>07-22</td>
<td>20</td>
<td>overcast</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>07-23</td>
<td>19</td>
<td>rain</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-26</td>
<td>11</td>
<td>rain</td>
<td>normal</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>07-30</td>
<td>16</td>
<td>rain</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
</tbody>
</table>

today  9  sunny  normal  false  ?
tomorrow 13  sunny  normal  false  ?

possible rules:

- play=no ← temperature ≥ 25.5 ∧ temperature < 28.5
- play=no ← temperature < 14 ∧ temperature ≥ 9.5
- play=no ← outlook=rainy ∧ windy=true

but also (t=temperature):

- play=no ← t < 26.5 ∧ t ≥ 25.5 ∧ outlook=sunny ∧ humidity=high ∧ windy=false
- play=no ← t < 28.5 ∧ t ≥ 27.5 ∧ outlook=sunny ∧ humidity=high ∧ windy=true
  ...

July 2, 2009 | Oberseminar Stochastik 2009 | Janssen | 7
Separate-and-conquer Rule Learning

- Separate-and-conquer (or Covering) paradigm (originated from the AQ algorithm (Michalski, 1969))
- still used in most Rule Learning systems (e.g., Ripper (Cohen, 1995))

1. **Generalization**: extend the current theory by a “good” rule
2. **Separate**: remove all examples covered by this rule
3. **Conquer**: if examples left, goto 1.

- rules are combined in a decision list
  - sorted list of rules
  - the first rule that “covers” the example is used to classify the example
  - if no rule covers the example the last rule is used as a default rule (predicts the majority class)
Searching for a single rule

- generate the first rule that covers all examples
- generate all refinements of the current rule by creating all attribute-value pairs from the data
  - nominal attributes: use equality tests (i.e., =)
  - numerical attributes: use inequality tests (i.e., ≥ and <)
- add each refinement to the current rule and test which is the best for a given (heuristic) criterion
- if a new best is found store it
- if the error of the rule is 0 stop the process and return the best rule that was found during this process
Combining rules in a decision list

- if a rule is found add the rule to the sorted list of rules
- remove all the examples that are covered by the rule
- if all but the remaining $n$ examples are covered stop inducing rules (currently $n = 1$)
- else: search for the next rule on the remaining examples
- as last rule add a default rule that predicts the majority class
Rule Learning Heuristics

- Rule Learning Heuristics implement the criterion for evaluating rules
- many Rule Learning Heuristics for classification are known (based on positive and negative examples)
- Parametrized trade-off between
  - Consistency: \((1 - \text{error})\) of the rule and
  - Coverage: how many examples are covered by the rule
- Heuristics for Regression (positive and negative examples are not known here) rely on
  - the current error/loss (Consistency in classification) of the rule
  - the coverage of the rule
- Regression Heuristics may also feature a parameter that trades off between the error and the Coverage of the rule
From Classification to Regression

- instead of predicting a discrete outcome in Regression the outcome is continuous
- 2 ways to deal with this:
  1. discretize numeric outcome and use standard classification algorithms
     - problem: number of classes has to be known in advance
     - algorithm used to discretize: P-CLASS (Weiss and Indurkhya, 1995))
  2. adapt the algorithm to Regression tasks
     - example for an adaption in Rule Learning
       - either predict a certain value (Median or Mean) in the head of the rule directly (like we did)
       - or use a (linear) model in the head to predict the value (algorithm M5Rules (Holmes, Hall, and Frank, 1999), (Quinlan, 1992))
Regression measures

- Mean Absolute Error $MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \bar{y}_i|$
- Mean Squared Error $MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y}_i)^2$
- Deviation from Mean $\text{def} = \frac{1}{n} \sum_{i=1}^{n} (y_i - y')^2$
- Normalized Mean Squared Error $NMSE = \frac{MSE}{\text{def}}$
- Relative Coverage $RC = \frac{\text{Coverage}(r)}{n}$
- Relative Cost Measure $h_{rcm} = c \cdot (1 - NMSE) + (1 - c) \cdot RC$

where $n = \#$ of examples left, $y_i = \text{true value}$, $\bar{y}_i = \text{predicted value}$, $y' = \text{mean of all instances}$, $r = \text{the current rule}$
Current implementation

- numerical and nominal attributes, numerical target variable
- covering paradigm
- interchangeable heuristics and splitpoint computing methods
- parameters:
  - parameter of the heuristic
  - parameter for splitpoint computation
    - to reduce the number of splitpoints for a numerical attribute a clustering was used
    - the parameter determines how many clusters are computed
  - percentage of coverage of ruleset (for inducing the default rule)
    - currently all but the last remaining example has to be covered
Handling of numerical attributes

- if all possible splitpoints (those between 2 instances) for all numeric attributes are used the search space explodes
- remedy: do not create all splitpoints but cluster examples together that minimize some error criterion
- and use only the splitpoints between these clusters (currently about 5-10)
- Algorithm:
  - sort the examples of the attribute in ascending order
  - remove duplicates by setting the mean over all duplicates as target value
  - merge examples that minimize the mean absolute error
Handling of numerical attributes

- if all possible splitpoints (those between 2 instances) for all numeric attributes are used the search space explodes
- remedy: do not create all splitpoints but cluster examples together that minimize some error criterion
- and use only the splitpoints between these clusters (currently about 5-10)
- Algorithm:
  - sort the examples of the attribute in ascending order
  - remove duplicates by setting the mean over all duplicates as target value
  - merge examples that minimize the mean absolute error
Handling of numerical attributes

- if all possible splitpoints (those between 2 instances) for all numeric attributes are used the search space explodes
- remedy: do not create all splitpoints but cluster examples together that minimize some error criterion
- and use only the splitpoints between these clusters (currently about 5-10)
- Algorithm:
  - sort the examples of the attribute in ascending order
  - remove duplicates by setting the mean over all duplicates as target value
  - merge examples that minimize the mean absolute error
Handling of numerical attributes

- if all possible splitpoints (those between 2 instances) for all numeric attributes are used the search space explodes
- remedy: do not create all splitpoints but cluster examples together that minimize some error criterion
- and use only the splitpoints between these clusters (currently about 5-10)
- Algorithm:
  - sort the examples of the attribute in ascending order
  - remove duplicates by setting the mean over all duplicates as target value
  - merge examples that minimize the mean absolute error
Handling of numerical attributes

- if all possible splitpoints (those between 2 instances) for all numeric attributes are used the search space explodes
- remedy: do not create all splitpoints but cluster examples together that minimize some error criterion
- and use only the splitpoints between these clusters (currently about 5-10)
- Algorithm:
  - sort the examples of the attribute in ascending order
  - remove duplicates by setting the mean over all duplicates as target value
  - merge examples that minimize the mean absolute error
Handling of numerical attributes

- if all possible splitpoints (those between 2 instances) for all numeric attributes are used the search space explodes
- remedy: do not create all splitpoints but cluster examples together that minimize some error criterion
- and use only the splitpoints between these clusters (currently about 5-10)
- Algorithm:
  - sort the examples of the attribute in ascending order
  - remove duplicates by setting the mean over all duplicates as target value
  - merge examples that minimize the mean absolute error

<table>
<thead>
<tr>
<th>Attribute Value</th>
<th>Target Value</th>
<th>Attribute Value</th>
<th>Target Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>10</td>
<td>6</td>
</tr>
</tbody>
</table>
Handling of numerical attributes

- if all possible splitpoints (those between 2 instances) for all numeric attributes are used the search space explodes
- remedy: do not create all splitpoints but cluster examples together that minimize some error criterion
- and use only the splitpoints between these clusters (currently about 5-10)
- Algorithm:
  - sort the examples of the attribute in ascending order
  - remove duplicates by setting the mean over all duplicates as target value
  - merge examples that minimize the mean absolute error
Evaluation of the models

- for domain-dependent evaluation we used $MAE$ and $RMSE = \sqrt{MSE}$
- for domain-independent evaluation we used the correlation coefficient (between predicted and actual value)
- we also record model complexity by measuring the number of rules and conditions (for rule based models)
- 1x10 cross-validation with same folds for each model
- our approach was compared to M5Rules, Linear Regression, SVMReg (all implemented in weka (Witten and Frank, 2005))
Results

In terms of MAE

- preliminary results \((sp = 10, c = 0.45)\) for 13 datasets from the UCI-Repository (Asuncion and Newman, 2007)
- second number describes standard deviation among the 10 folds of the CV

<table>
<thead>
<tr>
<th>dataset</th>
<th>SeCo</th>
<th>M5Rules</th>
<th>Linear Regression</th>
<th>SVMReg</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto-horse</td>
<td>16.61 ± 6.35</td>
<td>15.85 ± 10.25</td>
<td>13.64 ± 3.24</td>
<td>13.48 ± 4.0</td>
</tr>
<tr>
<td>auto-mpg</td>
<td>4.44 ± 1.49</td>
<td>3.03 ± 0.81</td>
<td>2.87 ± 0.98</td>
<td>2.83 ± 0.98</td>
</tr>
<tr>
<td>auto-price</td>
<td>2526.6 ± 773.1</td>
<td>2157.8 ± 937.4</td>
<td>2450.5 ± 1084.0</td>
<td>2292.32 ± 1012.05</td>
</tr>
<tr>
<td>breast-tumor</td>
<td>8.02 ± 0.73</td>
<td>7.79 ± 0.74</td>
<td>7.9 ± 0.72</td>
<td>8.2 ± 0.76</td>
</tr>
<tr>
<td>cloud</td>
<td>0.45 ± 0.15</td>
<td>0.3 ± 0.12</td>
<td>0.27 ± 0.07</td>
<td>0.28 ± 0.09</td>
</tr>
<tr>
<td>cpu</td>
<td>36.80 ± 29.38</td>
<td>15.19 ± 9.17</td>
<td>47.7 ± 20.89</td>
<td>24.95 ± 23.52</td>
</tr>
<tr>
<td>echo-month</td>
<td>13.41 ± 2.62</td>
<td>8.68 ± 2.97</td>
<td>8.48 ± 3.15</td>
<td>9.08 ± 2.73</td>
</tr>
<tr>
<td>housing</td>
<td>5.43 ± 2.98</td>
<td>3.39 ± 1.44</td>
<td>3.99 ± 2.13</td>
<td>3.73 ± 2.05</td>
</tr>
<tr>
<td>meta</td>
<td>95.59 ± 170.29</td>
<td>232.52 ± 190.24</td>
<td>146.54 ± 148.08</td>
<td>96.91 ± 166.08</td>
</tr>
<tr>
<td>sensory</td>
<td>0.64 ± 0.13</td>
<td>0.73 ± 0.14</td>
<td>0.76 ± 0.18</td>
<td>0.77 ± 0.19</td>
</tr>
<tr>
<td>servo</td>
<td>0.54 ± 0.15</td>
<td>0.32 ± 0.11</td>
<td>0.62 ± 0.12</td>
<td>0.53 ± 0.17</td>
</tr>
<tr>
<td>strike</td>
<td>274.61 ± 116.46</td>
<td>287.0 ± 87.31</td>
<td>264.73 ± 84.0</td>
<td>228.49 ± 83.23</td>
</tr>
<tr>
<td>veteran</td>
<td>91.28 ± 59.05</td>
<td>92.99 ± 44.4</td>
<td>92.99 ± 44.4</td>
<td>82.58 ± 54.89</td>
</tr>
</tbody>
</table>

average rank  | 3.08 | 2.27 | 2.58 | 2.08 |
Results

In terms of different parametrizations

- the number of splitpoints are fixed to 10 but the parameter of the heuristic is varied
- lowest errors are marked blue

<table>
<thead>
<tr>
<th>dataset</th>
<th>$c = 0.45$</th>
<th></th>
<th>$c = 0.5$</th>
<th></th>
<th>$c = 0.6$</th>
<th></th>
<th>$c = 0.7$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td># rules</td>
<td>MAE</td>
<td># rules</td>
<td>MAE</td>
<td># rules</td>
<td>MAE</td>
<td># rules</td>
</tr>
<tr>
<td>auto-horse</td>
<td>16.6</td>
<td>2</td>
<td>15.2</td>
<td>16</td>
<td>21.4</td>
<td>35</td>
<td>16.5</td>
<td>57</td>
</tr>
<tr>
<td>auto-mpg</td>
<td>4.44</td>
<td>1</td>
<td>3.92</td>
<td>157</td>
<td>3.62</td>
<td>184</td>
<td>3.64</td>
<td>226</td>
</tr>
<tr>
<td>auto-price</td>
<td>2526</td>
<td>6</td>
<td>2922</td>
<td>7</td>
<td>3104</td>
<td>46</td>
<td>2836</td>
<td>48</td>
</tr>
<tr>
<td>breast-tumor</td>
<td>8.0</td>
<td>0</td>
<td>8.5</td>
<td>13</td>
<td>10.7</td>
<td>209</td>
<td>10.4</td>
<td>236</td>
</tr>
<tr>
<td>cloud</td>
<td>0.45</td>
<td>7</td>
<td>0.46</td>
<td>6</td>
<td>0.42</td>
<td>12</td>
<td>0.39</td>
<td>42</td>
</tr>
<tr>
<td>cpu</td>
<td>36.8</td>
<td>5</td>
<td>37.8</td>
<td>7</td>
<td>38.8</td>
<td>9</td>
<td>29.3</td>
<td>15</td>
</tr>
<tr>
<td>echo-month</td>
<td>13.4</td>
<td>0</td>
<td>14.2</td>
<td>79</td>
<td>14.2</td>
<td>92</td>
<td>13.2</td>
<td>87</td>
</tr>
<tr>
<td>housing</td>
<td>5.43</td>
<td>5</td>
<td>4.7</td>
<td>43</td>
<td>4.54</td>
<td>369</td>
<td>4.47</td>
<td>427</td>
</tr>
<tr>
<td>meta</td>
<td>95.6</td>
<td>3</td>
<td>95.2</td>
<td>30</td>
<td>147.8</td>
<td>69</td>
<td>147</td>
<td>124</td>
</tr>
<tr>
<td>sensory</td>
<td>0.63</td>
<td>0</td>
<td>0.82</td>
<td>430</td>
<td>0.86</td>
<td>404</td>
<td>0.9</td>
<td>428</td>
</tr>
<tr>
<td>servo</td>
<td>0.54</td>
<td>4</td>
<td>0.39</td>
<td>20</td>
<td>0.39</td>
<td>22</td>
<td>0.38</td>
<td>29</td>
</tr>
<tr>
<td>strike</td>
<td>274</td>
<td>0</td>
<td>362</td>
<td>234</td>
<td>361</td>
<td>300</td>
<td>368</td>
<td>359</td>
</tr>
<tr>
<td>veteran</td>
<td>91</td>
<td>0</td>
<td>116</td>
<td>70</td>
<td>119</td>
<td>82</td>
<td>123</td>
<td>91</td>
</tr>
</tbody>
</table>
Discussion

- our algorithm implements a Separate-and-conquer Regression Rule Learner
- trade-off between consistency and coverage is more complex than it is in classification
  - tuning of the parameters has to be analyzed better
- but the current implementation is competitive to other rule-based implementations (that do not predict models in the head)
- a new splitpoint computing method was introduced
  - only about 10 splitpoints are sufficient for most of the datasets
  - much more faster than computing all splitpoints
  - but optimal cluster number still has to be found
Future Work

- this is work-in-progress so there are many ways to improve the algorithm
  - by determine a suitable setting of the cluster parameter
  - by systematically tune the parameter of the heuristic
    - previously we tuned the parameters of 5 heuristics for classification
    - we also want to find the best parameter for regression
  - by avoiding overfitting by leaving more examples uncovered
- predict (linear) models in the head of the rule
- try to visualize the behaviour of the different heuristics in a space similar to 
  \textit{Coverage Spaces}
- include domain-independent comparison with $\text{RRMSE} = \sqrt{\frac{\text{MSE}}{\text{def}}}$
References


