Meta-uczenie z analizą profili

Krzysztof Grąbczewski

Katedra Informatyki Stosowanej
Uniwersytet Mikołaja Kopernika
Toruń

Warszawa, 13 listopada 2013

Contents

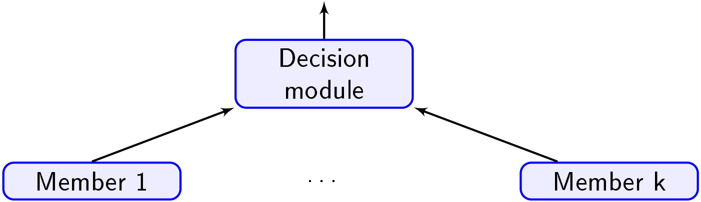
What is meta-learning?

- Generally, **meta-learning** encompasses all efforts to **learn how to learn** including gathering **meta-knowledge** and using meta-knowledge in further learning.
- **Meta-knowledge** is knowledge about learning processes, about influence of machine parameters on final results, etc.

During last two decades, the term **meta-learning** has been used in many different contexts:

- building **committees** of decision models,
- building **regression models** predicting machine accuracy,
- building **algorithms rankings** for given datasets,
- **searching through spaces** of learning machines parameters augmented by meta-knowledge and gathering new meta-knowledge.

Committees of decision models



- Simple committees **do not learn at meta-level**: e.g. simple majority voting.
- Some “intelligent” decision modules perform meta-analysis.
  - **Bagging, arcing, boosting** — perform some meta-analysis to build more stable decision makers (Dietterich 1997) and are very popular, but this is not exactly what we would name “meta-learning”.
  - **Stacking** — the decision module is a meta-level learner.
  - Many advanced, heterogeneous, **undemocratic** committees have been published.

Stacking

- **Learning machines are trained on results** of a group of models.
- Stolfo et al (1997), Prodromidis and Chan (2000) — **JAM** (Java Agents for Meta-learning) — a parallel, distributed system for scalable computing.
- Todorovski and Dzeroski (2003) — **Meta Decision Trees** — properly adapted C4.5 decision trees determine which model to use.
- NOEMON — Kalousis and Theoharis (1999), Kalousis and Hilario (2000) — also called stacking a meta-learning.

Undemocratic committees

Meta-analysis may lead to estimation of the **areas and degrees of competence** of each base learner to provide more reasonable decision of the decision module.

- Chan and Stolfo (1993, 1996):
  - Meta-learning by **arbitration** and **combining**.
  - **Arbiters**: binary tree of arbiters (members organized in pairs, arbiter for each pair, arbiters in pairs, and so on),
  - **Combiners**: a sort of stacking.
  - **Combiners compute** a prediction that may be entirely different from any proposed by base models, whereas **arbiters choose** one of the predictions of the base models.
- Duch and Itert (2003) define **incompetence functions** that describes member (in)competence in particular points of the data space.
- Jankowski and Grąbczewski (2005) reflect **global and local** competence in final ensemble decisions.

Meta-level regression

- Regression methods **predict accuracies** of different learning machines on the basis of dataset descriptions.
- Köpf et al (2000), Bensusan and Kalousis (2001):
  - Input: **dataset description** as a series of values derived from information theory and statistics.
  - Output: **accuracy** of the model (usually classifier).
- **Ranking** learning machines:
  - One regression model for each algorithm to rank.
  - Machines are **ranked** in the decreasing order of predicted accuracy.

Rankings of algorithms

- The most popular approach initiated by (probably largest so far) meta-learning project **MetaL** (1998-2002).
- Rankings learned from **simple descriptions** of data.

- **Meta-attributes** are basic data characteristics: number of instances, number of features, types of features (continuous or discrete, how many of which), data statistics etc.
- **Rankings** are generated by meta-learners:
  - **for each pair of algorithms** to be ranked, a classification algorithm is trained on two-class datasets describing wins and losses of the algorithms on some collection of datasets,
  - decisions of **meta-classifiers are combined** to build final ranking.

Algorithm selection problem

Algorithm selection problem (ASP)

- May be regarded as **equivalent** to building algorithm rankings.
- ASP was addressed already by Rice (1974, 1976):

$D \in \mathcal{D}$ problem space $\xrightarrow{S(D)}$ $A \in \mathcal{A}$ algorithm space $\xrightarrow{p(A,D)}$ $m \in \mathcal{R}^n$ performance measure space $\xrightarrow{||\cdot||}$ $||m|| \in \mathcal{R}$ performance norm

- Most often, it gets reduced to the problem of assigning optimal algorithm to a **vector of features** describing data, which is quite **restrictive**.

No Free Lunch theorems

No free lunch theorems, in this context, may be expressed as:

> *Each single learning algorithm tested* ***on all possible datasets*** *will be, on average,* ***as accurate as random choice***.

So does building learning machines make any sense?

- Yes, because **all possible datasets** is what makes NFL **provable but useless**!
- In the context of training and test, "all possible" means also those, where training and test come from completely different distributions, are **completely unrelated**.
- We expect training data **representative** for the population and NFL does not care about representativeness.
- **Inductive bias** of algorithms is not an explanation.

Conclusion: let's **not pay much attention** to NFL!

Landmarking

- Pfahringer et al (2000)– the idea of **landmarking**: using meta-features measuring the **performance of some simple and efficient** learning algorithms (**landmarkers**).
  - **linear discriminant** learner,
  - **naive bayes** learner,
  - **C5.0 tree** learner.

  Meta-learners used:
  - C5.0 trees and rules,
  - boosted C5.0,
  - RIPPER,
  - LTREE,
  - linear discriminant,
  - naive bayes,
  - nearest neighbor.

Landmarking continued

Fürnkranz and Petrak (2001):

- **Relative landmarking** – meta-attributes describe relations between results instead of accuracies:
  - **Ranks** of landmarkers,
  - **Order** of landmarkers (inverse of ranks),
  - **Pairwise** comparisons between accuracies of landmarkers (+1, -1, ?),
  - Pairwise accuracies **ratios** (continuous).
- **Subsampling** – original datasets reduced to facilitate landmarking by algorithms of larger computational complexity.

Landmarking continued

Soares et al (2001):

- Relative landmarking and subsampling combined.
- **Adjusted ratio of ratios** (ARR) index — a **combination of accuracy and time** to assess relative performance:

$$ARR_{i,j}^{d} = \frac{\frac{A_i^d}{A_j^d}}{1 + \log\left(\frac{T_i^d}{T_j^d}\right) * X}$$

  A_i^d and T_i^d are accuracy and time of i'th landmarker on data d, X is a parameter: „the amount of accuracy we are willing to trade for 10-times speed-up".
- When $n > 2$ algorithms are involved, they calculate **relative landmark**:

$$rl_i^d = \frac{\sum_{j \neq i} ARR_{i,j}^{d}}{n-1}.$$

Still landmarking. . .

Brazdil and Soares (2000), Brazdil et al (2003):

- more **advanced statistical measures** of datasets (including histogram analysis and information theory based indices) as meta-attributes,
- **k nearest neighbor** (kNN) algorithm choses similar datasets; ranking created from **results** obtained by ranked algorithms on the nearest neighbors,
- methods of **combining results** to create rankings:
  - **ARR** – adjusted ratio of ratios,
  - **counting statistically significant differences** in results: **average ranks** (AR) and **significant wins** (SW).
  - ranking methods estimated by **comparison to the ideal ranking**
    - Spearman's rank correlation coefficient,
    - Friedman's significance test,
    - Dunn's multiple comparison technique.

Other landmarking related approaches

- DecT by Peng et al (2002):
  - Data characteristics derived from the **structure of C5.0 decision trees** built on the data.
  - Like in other approaches:
    - kNN to select similar datasets,
    - rankings by ARR,
    - Spearman's correlation coefficient to estimate rankings.
- Bensusan et al (2000):
  - Landmarking and decision trees techniques combined.
  - **Typed higher-order inductive learning** directly from decision trees instead of trees characteristics.
- Todorovski et al (2002):
  - Meta-data obtained from statistics, information theory and landmarking.
  - **Predictive Clustering Trees** – multi-split decision trees
    - minimization of intra-clusters variance and maximization of inter-clusters variance, clusters contain data with similar relative performance of algorithms,
    - ranks instead of accuracies – ranking trees.

Rankings of algorithms – general remarks

- Very **naive approach**: simple data transformation may completely change the ranking; simple descriptions are not likely to contain information about successful methods.
- Resembles common approach to split data analysis process into **data preprocessing** stage and **final learning**.
- We are not interested in raw rankings, but in **complex machine combinations** that model the data as accurately as possible.
- Even very accurate rankings **do not give hints about data transformations** that could improve the results.
- No human expert would use such technique to select most promising learning algorithms – **validation required**.
- Landmarking goes in a right direction, but is **passive** (does not adapt on-line).
- Ranking quality measures — **the top is more important** than the bottom.

Meta-learning as an advanced search process

- **Fundamental aim** of meta-learning is to be more successful in object-level (base-level) learning.
- **What do human experts do** to obtain optimal model for given data?
  - **search** for solutions by **testing** subsequent candidates,
  - test candidates not at random but **after selection and in order** based on some meta-knowledge,
  - **gain new meta-knowledge** (general and specific to the task being solved) while learning.
- Grąbczewski and Jankowski (2007, 2011): **Automated meta-learning** should mimic behavior of human experts. Therefore, in our approach, we:
  - **generate candidate machine configurations** according to meta-knowledge (initially from human experts),
  - **order** candidates with special **complexity measure**,
  - **test candidates** to create a ranking,
  - **gather new** meta-knowledge and **refine** human experts' meta-knowledge to successfully drive the search process.

Ranking-based meta search algorithm

Input: A ranking $\mathcal{C}$ of machine configurations, a validation scenario (VS), a time deadline.
Output: Machine configuration ranking.
The algorithm:

1. $C_R \leftarrow \emptyset$
2. $step \leftarrow 0$
3. While the time deadline is not exceeded:
   a. $step \leftarrow step + 1$
   b. $c \leftarrow \mathcal{C}[step]$
   c. $r \leftarrow VS(c)$ (perform VS for c and get the result r)
   d. Add (c, r) to C_R
4. Return the configurations from C_R in the order of decreasing results

Rankings I

- Computationally complex ranking algorithms (ARR with RL, SRR or SW) not eligible for large numbers of machine configurations.
- **Average test accuracy**:

$$AA(c) = \frac{1}{n} \sum_{i=1}^{n} Acc_c(i). \tag{1}$$

- **Average difference** with the best method, in the units of standard deviations of the best method:

$$AD(c) = \frac{1}{n} \sum_{i=1}^{n} \frac{Acc_c(i) - Acc_{Best(i)}(i)}{\sigma_{Best(i)}(i)}. \tag{2}$$

- **Average ranks**:

$$AR(c) = \frac{1}{n} \sum_{i=1}^{n} Rank(c, i), \tag{3}$$

- **Average p-value**:

$$AP(c) = \frac{1}{n} \sum_{i=1}^{n} p(c, i). \tag{4}$$

- Weighting or nearest neighbors methods may be used to select "similar" datasets before averaging indices.

The idea of profiles

- The problems of handling **many machine configurations of similar complexity** — tiny or no difference in time and memory consumption, but significant difference in accuracy.
- **Complexity estimates** would be equally successful as random guess in this case.
- The idea: **relative differences** between the results obtained by different machines may point the directions.
- **Adaptive (or active) relative landmarking** — any machine can be a landmarker.
- **Profiles** — the results of arbitrary selection of learning machines
- Profiles **can change in time**, when the feedback shows that the current profile predictions are inaccurate.
- **Active search** process.

PBML algorithm I

- **Search** within a set of candidate machine configurations $\mathcal{C}$.
- Machine **validation procedure** returns some information about the quality of the machine as an element of a set $\mathcal{R}$ (with a defined order relation).
- The algorithm is based on **three collections**:
  - C_R – a collection of pairs $(c, r) \in \mathcal{C} \times \mathcal{R}$ of machine configurations c validated so far (in the search process) with the validation results r,
  - $C_P \subseteq C_R$ – a collection of specially selected results (the profile),
  - C_Q – a sequence of candidate configurations (the queue) ordered with respect to estimated qualities and step numbers at which they were added.

PBML algorithm II

Input: A set $\mathcal{C}$ of machine configurations,
a validation scenario (VS), a time deadline, a profile manager (PM).

1. $C_R \leftarrow \emptyset$, $C_Q \leftarrow \emptyset$, $step \leftarrow 0$
2. Initialize PM
3. While the time deadline is not exceeded:
   - If $step == 0$ or PM changed the profile since last time:
     1. $C_B \leftarrow$ new ranking for current profile
     2. For each $c \in C_B$, if c does not occur in C_R then add $\left(c, step + \frac{rank\ of\ c\ in\ C_B}{length(C_B)}\right)$ to C_Q
   - If C_Q is empty then break the loop
   - Pop an item c with maximum rank from C_Q
   - $r \leftarrow$ VS(c) (perform VS for c and get the result r)
   - Add (c, r) to C_R
   - Adjust the profile by PM with (c, r)
   - $step \leftarrow step + 1$
4. Return configurations from C_R (ordered)

PBML algorithm III

- **New ranking** is generated each time a **profile is changed**.
- To speed up, the queue is updated only if several configurations have been validated.
- **Queue order**—first the most recent ranking is considered.
- The algorithm is very general, **configurable**:
  - machine configuration space,
  - validation scenario (VS)—here CV-test and a query,
  - profile management (PM).

Profile management

- The functionality of the **profile manager**
  - decides about the shape of the profile, that is **when and how the profile is modified**,
  - calculates **profile similarities** for ranking generation,
  - manages the **knowledge base**.
- Example profile manager **parameters**:
  - the number of configuration–result pairs to keep in the profile,
  - the strategy of determining the configurations to remove from the profile, when new configuration is provided and the profile has already reached its destination size.
- Needed research:
  - What **size of the profile** is optimal?
  - Which **configurations** should be kept?
  - How to measure **profile similarity**?

Experiments — the knowledge base

- To test the PBML framework, a **nontrivial knowledge base** had to be available.
- Results from another research task on DT CV Committees (Grąbczewski, 2013).
- 21 UCI datasets.
- **13660** machine configurations:
  - 13560 different settings of cross-validation committees.
  - 100 parameter settings of single DT induction methods
- 10×10-fold CV results for all configurations.

Experiments datasets

Dataset	classes	instances	features	ordered f.
appendicitis	2	106	7	7
Australian credit	2	690	14	6
breast cancer (Wisconsin)	2	699	9	9
flag	8	194	28	10
glass	6	214	9	9
heart	2	303	13	13
image	7	2310	19	19
ionosphere (trn+tst)	2	351	34	34
iris	3	150	4	4
kr-vs-kp	2	3196	36	0
Ljubjlana breast cancer	2	286	9	1
letter recognition	26	20000	16	16
Pima indians diabetes	2	768	8	8
sonar	2	208	60	60
soybean large	19	307	35	0
splice	3	3190	60	0
thyroid (trn+tst)	3	7200	21	6
vote	2	435	16	0
vowel	6	871	3	3
waveform	3	5000	21	21
wine	3	178	13	13

Experiments — learning machines

13560 DT CV Committees configurations

- 4 **DT induction algorithms** (Gini index, information gain, QUEST, SSV),
- **committee size** in the range from 1 to 10 (10-fold CV-based validation),
- 6 DT **validation methods**: Reduced Error Pruning (REP), cost-complexity (CC), degree-based pruning, OPTimal pruning, Minimum Error Pruning 2 and Depth Impurity,
- respecting **standard error**: 0SE, .5SE, 1SE, and estimated from sample .5SE and 1SE,
- **training error factor**: 0, 0.5, 1,
- **common or separate** parameter optimization,
- **decision making** by: proportions, Laplace correction, m-estimates.

Experiments — learning machines

100 parameter settings of **single DT** induction methods

- 4 **DT induction algorithms** (Gini index, information gain, QUEST, SSV),
- 6 DT **validation** methods: Reduced Error Pruning (REP), cost-complexity (CC), degree-based pruning, OPTimal pruning, Minimum Error Pruning 2 and Depth Impurity,
- Respecting **standard error**: 0SE, .5SE, 1SE, and estimated from sample .5SE and 1SE,

Experiments — PBML configuration

- Profiles of **variable size** (full profiles, up to 100 configurations).
- Profiles **updated** after each 5 configurations.
- Profile **similarity** measure — Pearson linear correlation coefficient (truncated to 0 if negative)
- **First ranking** on the basis of average p-values (4).
- **Weighted p-values**, when profile with at least 2 results:

$$WPV(c) \leftarrow \sum_{D \in KB} Max(0, CC(P,D)) * PV(c,D), \quad (5)$$

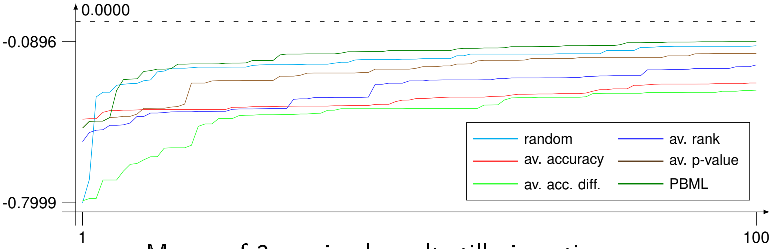
where
  - $D \in KB$ means "dataset D in the knowledge base",
  - $CC(P,D)$ is the Pearson linear correlation coefficient,
  - $PV(c,D)$ is the p-value obtained in paired t-test.

Experiment 1 — PBML vs passive rankings

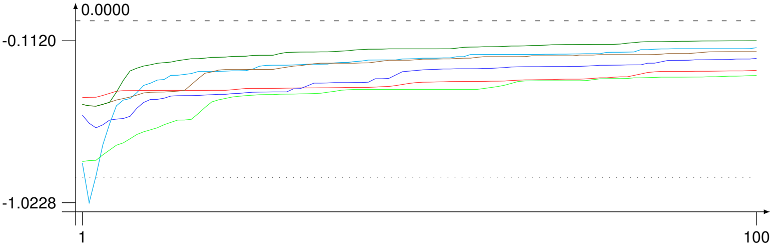
- PBML algorithm **compared** with 5 ranking methods:
  - completely random,
  - average accuracy (1),
  - average accuracy difference in st. deviations (2),
  - average ranks (3),
  - average p-values (4).
- **Leave-one-out** procedure for the 21 datasets.
- The most important aspect: what **maximum validation accuracy** can be obtained in given time.
- **Time unit** ≈ the number of configurations validated so far.
- Rankings of 100 configurations **visualized** as:
  - maximum accuracy till given time,
  - average of 3 maximum accuracies till given time,
  - average accuracy difference,
  - average mean accuracy till given time.

Experiment 1 results I

Maxima found till given time

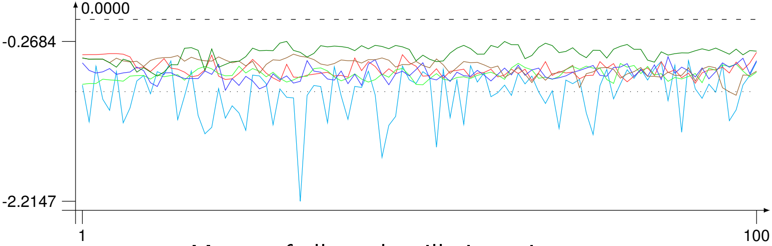


Means of 3 maximal results till given time

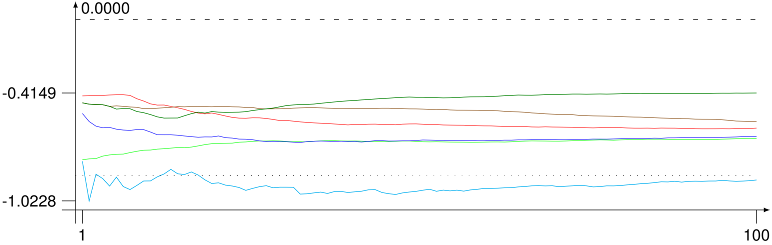


Experiment 1 results II

Accuracy difference



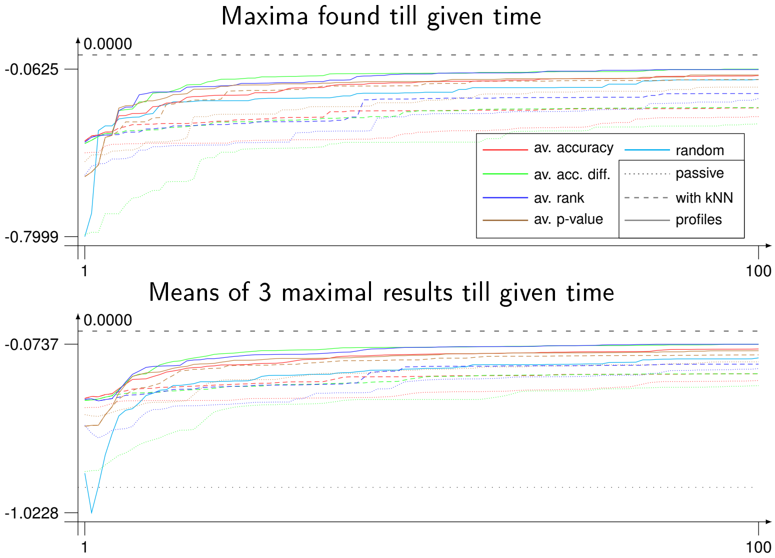
Means of all results till given time



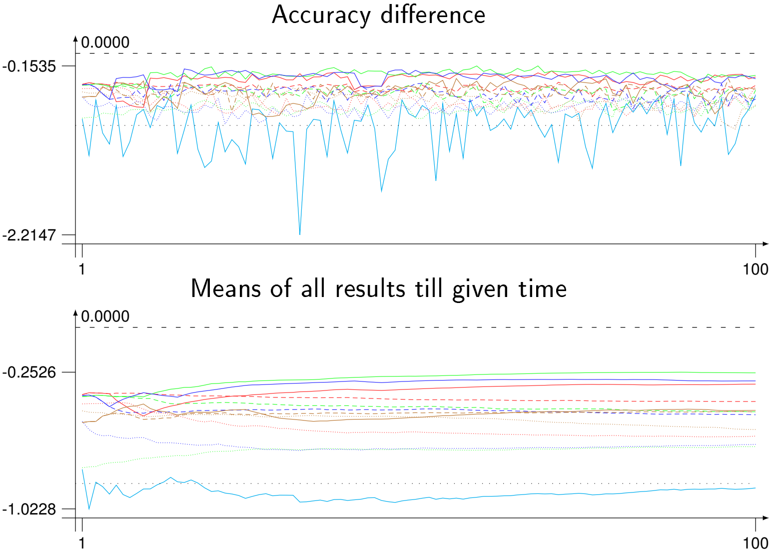
Experiment 2 — passive vs kNN vs active

- Passive rankings can be significantly improved by **averaging over NNs**.
  - kNN analysis by **landmarking** with selected machines of the examined population.
  - 5NN with Euclidean distance used here.
- As before, 4 ranking measures:
  - average accuracy (1),
  - average accuracy difference in st. deviations (2),
  - average ranks (3),
  - average p-values (4).
- PBML framework also suitable for passive methods with kNN.
- **Three versions** of each ranking method:
  - passive,
  - passive with kNN selection,
  - active.

Experiment 2 results I



Experiment 2 results II



Conclusions and further research

- PBML is an **open framework**, that facilitates easy implementation of many meta-learning algorithms.
- Different kinds of problems solved with appropriate **validation scenario** and **profile manager**.
- Active management of **learning results profiles** leads to more adequately adapted meta-learning algorithms.
- **Further reseach on PBML**:
  - intelligent methods for profile management,
  - knowledge base properties analysis for most eligible form of profiles for meta-learning.
- **Profile management** problems
  - profiles diversity,
  - continuous profile control,
  - adaptive methods of dataset similarity measurement,
  - most suitable ranking generation.
- Efficient specialized PBMLs as modules of more general meta-search processes (Jankowski and Grąbczewski, 2011).

Future of meta-learning

- **Transfer of knowledge** between learning processes
  - by means of **ontologies** (knowledge repositories),
  - **data-independent** representation,
- Knowledge verbalization
  - **rules** describing results relations,
  - extraction of informative **features**,
- Ontologies
  - more experiments **results**,
  - **representation** of knowledge,
  - **queries/interfaces** for knowledge extraction.

Dziękuję!

Serdecznie dziękuję za uwagę!

Bibliography I

Bensusan H, Kalousis A (2001) Estimating the predictive accuracy of a classifier. In: Lecture Notes in Computer Science, vol 2167, pp 25–31

Bensusan H, Giraud-Carrier C, Kennedy C (2000) A higher-order approach to meta-learning. Tech. rep., Bristol, UK, UK

Brazdil P, Soares C (2000) Ranking classification algorithms based on relevant performance information. In: The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases

Brazdil P, Soares C, Costa JPD (2003) Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. Machine Learning 50(3):251–277, URL http://dx.doi.org/10.1023/A:1021713901879

Chan P, Stolfo SJ (1993) Toward parallel and distributed learning by meta-learning. In: In AAAI Workshop in Knowledge Discovery in Databases, pp 227–240

Chan P, Stolfo SJ (1996) On the accuracy of meta-learning for scalable data mining. Journal of Intelligent Information Systems 8:5–28

Duch W, Itert L (2003) Committees of undemocratic competent models. In: Proceedings of the Joint Int. Conf. on Artificial Neu-ral Networks (ICANN) and Int. Conf. on Neural Information Processing (ICONIP), Istanbul, Turkey, pp 33–36

Fürnkranz J, Petrak J (2001) An evaluation of landmarking variants. In: Giraud-Carrier C, Lavra N, Moyle S, Kavsek B (eds) Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning

Grąbczewski K (2013) Decision tree cross-validation committees. Data Mining and Knowledge Discovery Submitted

Bibliography II

Grąbczewski K, Jankowski N (2007) Versatile and efficient meta-learning architecture: Knowledge representation and management in computational intelligence. In: IEEE Symposium Series on Computational Intelligence (SSCI 2007), IEEE, pp 51–58

Grąbczewski K, Jankowski N (2011) Efficient and friendly environment for computational intelligence. Knowledge-Based Systems p 41, (in print)

Jankowski N, Grąbczewski K (2005) Heterogenous committees with competence analysis. In: Nedjah N, Mourelle L, Vellasco M, Abraham A, Köppen M (eds) Fifth International conference on Hybrid Intelligent Systems, IEEE, Computer Society, Brasil, Rio de Janeiro, pp 417–422

Jankowski N, Grąbczewski K (2011) Universal meta-learning architecture and algorithms. In: Jankowski N, Duch W, Grąbczewski K (eds) Meta-Learning in Computational Intelligence, Studies in Computational Intelligence, vol 358, Springer Berlin / Heidelberg, pp 1–76

Kalousis A, Hilario M (2000) Model selection via meta-learning: a comparative study. In: Proceedings of the 12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'00), IEEE Computer Society, Vancouver, BC, Canada, pp 406–413

Kalousis A, Theoharis T (1999) NOEMON: An intelligent assistant for classifier selection

Köpf C, Taylor CC, Keller J (2000) Meta-analysis: From data characterisation for meta-learning to meta-regression. In: Proceedings of the PKDD-00 Workshop on Data Mining, Decision Support, Meta-Learning and ILP: Forum for Practical Problem Presentation and Prospective Solutions, Lyon, France

Peng Y, Flach PA, Soares C, Brazdil P (2002) Improved dataset characterisation for meta-learning. In: DS '02: Proceedings of the 5th International Conference on Discovery Science, Springer-Verlag, London, UK, pp 141–152

Pfahringer B, Bensusan H, Giraud-Carrier C (2000) Meta-learning by landmarking various learning algorithms. In: Proceedings of the Seventeenth International Conference on Machine Learning, Morgan Kaufmann, pp 743–750

Bibliography III

Prodromidis A, Chan P (2000) Meta-learning in distributed data mining systems: Issues and approaches. In: Kargupta H, Chan P (eds) Book on Advances of Distributed Data Mining, AAAI press

Rice JR (1974) The algorithm selection problem – abstract models. Tech. rep., Computer Science Department, Purdue University, West Lafayette, Indiana, cSD-TR 116

Rice JR (1976) The algorithm selection problem. Advances in Computers 15:65–118

Soares C, Petrak J, Brazdil P (2001) Sampling-based relative landmarks: Systematically test-driving algorithms before choosing. In: EPIA '01: Proceedings of the10th Portuguese Conference on Artificial Intelligence on Progress in Artificial Intelligence, Knowledge Extraction, Multi-agent Systems, Logic Programming and Constraint Solving, Springer-Verlag, London, UK, pp 88–95

Stolfo S, Prodromidis A, Tselepis S, Lee W, Fan D, Chan P (1997) JAM: Java agents for meta-learning over distributed databases. In: Third Intl. Conf. on Knowledge Discovery and Data Mining, pp 74–81

Todorovski L, Dzeroski S (2003) Combining classifiers with meta decision trees. Machine Learning Journal 50(3):223–249

Todorovski L, Blockeel H, Dzeroski S (2002) Ranking with predictive clustering trees. In: ECML '02: Proceedings of the 13th European Conference on Machine Learning, Springer-Verlag, London, UK, pp 444–455