European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD 2012)

The Silver Lining

International Workshop on Learning from Unexpected Results

Silver 2012

September 24, 2012

Bristol, UK

Edited by

Joaquin Vanschoren and Wouter Duivesteijn

Preface

This workshop is dedicated to the proposition that insight often begins with unexpected results. Successful methods do not simply fall from the sky: they are discovered based on clues gathered by trying several ideas, learning from surprising results, and building an understanding of what works, what does not, and why.

Unexpected results chart the boundaries of our knowledge: they identify errors, reveal false assumptions, and force us to dig deeper. When a system works we focus on its input/output behavior, but only when a problem occurs, we examine the underlying mechanisms to understand what went wrong.

Unfortunately, this process is rarely mentioned in the machine learning and data mining discourse, meaning that this insight is essentially lost. Ironically, while we have long understood that learning from only positive results is substantially harder than learning from both positives and negatives, there exists a publication bias that favours (incremental) successes over novel discoveries of why some ideas, while intuitive and plausible, do not work.

Good science consists of carefully designed experiments, systematic procedures, and honest evaluations. It is not mandatory for the results to be positive, only that they provide a deeper understanding of the field. As a scientific area where empirical methods dominate, it is a given that people try many ideas and obtain surprising results in the experimental stage. This may be due to lack of rigor, but often there are deeper, unexpected, and intriguing reasons. We can learn a lot if we analyze scientifically why an intuitive and plausible experiment did not work as expected.

Just as every cloud has a silver lining, these unexpected results define the actual boundaries of our field: they highlight what we do not yet understand, and often point to interesting and open problems that ought to be explored.

With this workshop, we want to give a voice to those unexpected results that deserve wider dissemination. These proceedings consist of 4 interesting contributions with shine a light on the value of unexpected results. These include stories that highlight the road to success: how a successful method was discovered after one or several failed attempts, as well as papers that question the way we currently perform and evaluate research and propose ways to improve on it.

We thank everybody for their sincere interest and their contributions, and especially thank our invited speakers:

Olivier Teytaud (Université Paris-Sud): *Unexpected Results in Monte-Carlo Tree Search.*
Albrecht Zimmermann (University of Leuven): *Science - we might be doing it wrong.*

We hope you will find it an interesting and inspiring workshop, leading to great new collaborations.

Leiden, September 2012

Joaquin Vanschoren
Wouter Duivesteijn

Workshop Organization

Workshop Chairs

Joaquin Vanschoren (LIACS, Leiden University)
Wouter Duivesteijn (LIACS, Leiden University)

ECML-PKDD Workshop Chairs

Arno Knobbe and Carlos Soares

Workshop Program Committee

Johannes Fürnkranz, Technische Universität Darmstadt, Germany
Jean-Gabriel Ganascia, Université Pierre et Marie Curie, LIP6, France
Christophe Giraud-Carrier, BYU, Utah, USA
Mehmet H. Göker, Salesforce.com
Geoffrey Holmes, University of Waikato, New Zealand
Melanie Hilario, University of Geneva, Switzerland
Ramon Lopez de Mantaras, CSIC Artificial Intelligence Research Institute, Canada
Guy Lapalme, Université de MontrŽal, Canada
Vivi Nastase, HITS, Germany
Johann Petrak, ÖFAI, Austria
Lutz Prechelt, Freie Universität Berlin, Germany
Ted Senator, SAIC, USA
Maarten van Someren, Universiteit van Amsterdam, The Netherlands
Luis Torgo, Universidade do Porto, Portugal
Martin Znidarsic, Jožef Stefan Institute, Slovenia

Table of Contents

Adventures in Feature Selection on an Industrial Dataset ...and Ensuing General Discoveries (Extended Abstract)

George Forman
HP Labs, Palo Alto, CA, USA

Abstract. We relate the story of an interesting failure of text feature selection methods on an industrial dataset of technical documents. Our detailed dissection and ultimate understanding of the failure led to the creation of general solutions that not only solved the robustness problem we faced, but were also able to improve classification accuracy for simpler, public datasets, which was crucial to enable the works' publishability.[1]

1 The Story

We were developing some simple text classification software for a Hewlett-Packard business division that wanted to sort a large collection of internal tech-support documents into various topic categories. Their previous *rule-based* system had, over the years, grown hard to maintain and was perceived as having poor accuracy. The rules consisted of over 8000 lines for English documents alone, containing several types of pattern matching, many variants of product names including third-party software, and prioritized, hierarchical Boolean logic for categorization into 100+ topics. This was the old state-of-the-art. Given an emailed report of some misclassified documents, it was quite difficult for the rule maintainers (different domain experts for different product lines) to know what to change about the many rules. Plus, any changes might lead to new misclassifications, damaging overall accuracy. It was hard to know, and at the time there were no ground-truth labels recorded from which we could measure its accuracy...nor try out a machine learning solution.

We pushed for machine learning, with hopes for better accuracy and a *much easier* way to improve the system whenever misclassified documents are found—just add them to the training set and retrain. Encouraged, the division worked with their domain experts to provide us a sample dataset of 10 classes, each with about 100 documents, on which we could develop our software and see how its accuracy compared with the existing solution.

This was, we thought, a simple multi-class (single-label) problem, to which we were applying straightforward techniques: traditional Naive Bayes with a bag-of-words vector representation of the text. Though feature extraction is conceptually simple, we had to make some hard decisions about whether '-' and '/' were punctuation or word characters: they were used in many technical names, such as 'HP-UX', 'MPE/XL', and 'FDDI/9000', and in many part numbers, such as 'C3166-69017'—but of course, there were many situations where they were better thought of as punctuation. We found that feature selection improved classification accuracy substantially, and we believed it would eventually be important for scalability on the full-size problem. We experimented with highly scalable feature filtering methods using Mutual Information, Chi-Squared, or Information Gain, promoted by publications such as Yang and Pedersen [2]. After cross-validation tests, we determined to ship the software using Information Gain to the internal division. Everything had gone smoothly...so far.

1.1 The Persistent Failure

After we delivered the software to the division, they slowly built up their labeled training set for over 230 technical topics. At first the classifications looked rather poor, but we had expected this initially and knew that acquiring additional training labels was necessary. Through discussions, we found some classes were trained inadequately, e.g. collecting its training documents by searching for only a single keyword or two, providing little diversity to learn from. At some point, their persistent perception of its poor results moved us to look for a problem. We first re-tested their use of our software on our sample data to verify its correct operation. After some delay, we were able to obtain a copy of their training data in order to reproduce the problem locally and perform cross-validation. Looking into the features selected, we saw oceans of unfamiliar technical words (e.g. s700_800, PHCO_3238, MIRRORDISK/UX, PHKL_1921), which seemed reasonable enough with our lack of knowledge about the large-scale domain problem. It took awhile for us to notice that words we might reasonably expect to find were missing, e.g. JETDIRECT or JETADMIN. We verified that such features did occur in the labeled training documents, and that the feature extraction routines did offer them. We found that the feature selection algorithm consistently threw them out, even if we wildly increased the number of selected features. We tested other feature selection functions and tried adjusting various parameters, such as the rare word cutoff and an assortment of choices for Laplace correction. We tried programming an assortment of additional methods for feature selection from the literature: different functions as well as different ways of aggregating them over classes. In measuring the Information Gain of a feature, one can alternatively measure it separately for each class vs. all others, and then aggregate the measurements for each of the 230+ classes in various ways: average, maximum, etc. Nothing worked to bring in the expected features. Perhaps they weren't such good features as we had thought, since every method refused to include them.

1.2 The Root Cause

It is rather difficult to scrutinize a confusion matrix from cross-validation with over 230 classes, but with some effort, we drew our attention to a class that was nicely accurate. Perhaps we might find a clue if we found out what was different about this class? It contained reports of standard HP/UX patch bundles. These documents looked somewhat unusual to our inexperienced eye, and they contained listings of many inscrutable patch names that were included or superseded by the current focus patch bundle. As a result, many early patch names occurred in many of the documents for this class. These made for excellently predictive features that rarely occurred in the other classes. *Aha!* Because there were incredibly many of these strong features, their excellent Information Gain scores—by whatever method we used—consistently crowded out most of the highly predictive features needed to distinguish other classes. Who would have thought that an abundance of good features available should be a problem?

1.3 The Solution

Clearly, the Information Gain scores of those patch features were excellent, but there was another selection criterion we needed to add: some notion of *fairness*. A difficult class with only weakly predictive features available would still need to get some of its best features selected, even if they had Information Gain scores that were completely uncompetitive with the scores of other features for other classes. This led to the development of the *SpreadFx* algorithm family [1], which is highly scalable and includes the idea of round-robin feature selection, giving each class (proportionate to its need or importance) equal chances to nominate features to be included. This gave good performance and proved much more robust for selecting features on such odd datasets. But there's more to the story.

1.4 The Research Path to its Eventual Publication

Though the solution was quite useful to us for this and other asymmetric classification tasks—and perhaps useful to others facing such anomalous datasets—would it be publishable? We suspected and confirmed that reviewers would balk at publishing our 'heuristic' and 'ad hoc' solution to such an 'atypical' problem, which 'should have treated the odd class separately to begin with.' Of course, this is perhaps good advice if you can afford to tailor the solution for each dataset. We regretted to be informed that our submission, though interesting, was not expected to be 'workable in other datasets.' Naturally, benchmark classification sets from UCI and elsewhere didn't exhibit such weird asymmetries, so there appeared to be little call for a method to treat it. Or was there?

To investigate its value more broadly, we had to go deeper: We suspected that even with typical, symmetric datasets, there would always be some classes that are easier than others, and that the feature selection phase could help the classifier by making sure to collect plenty of features for the difficult classes. We constructed an artificially balanced, 36-class dataset of computer science topics such that it had exactly the same number of documents in each class. Even on such an abnormally symmetric dataset, the difficulty of the individual classes varied, some having much better features available than others, as shown in the Figure 1 (reproduced from [1]). Using this dataset, we were able to show that

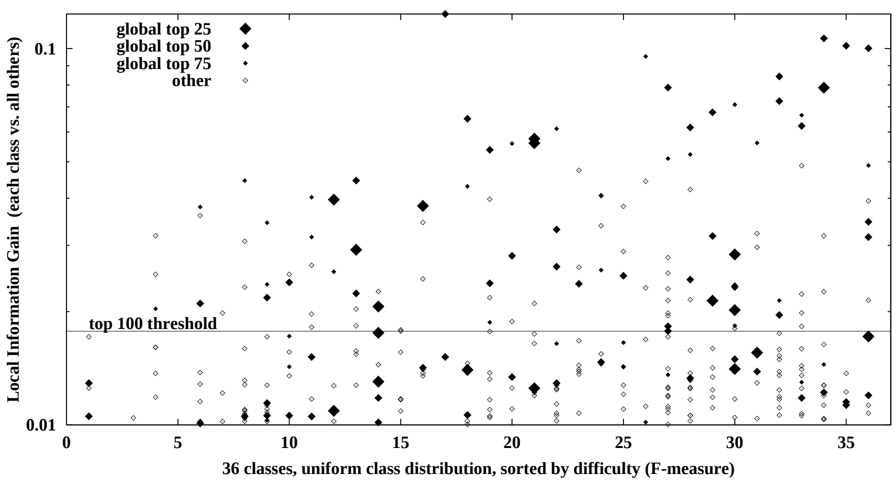

Figure 1. Information Gain scores of top features. Each column corresponds to one class (sorted by F-measure). We plot the local Information Gain of each feature in distinguishing that class from all other classes. Additionally, we indicate the top *global* Information Gain scores via point shapes, e.g. large diamonds mark features included in the global top 25 features.

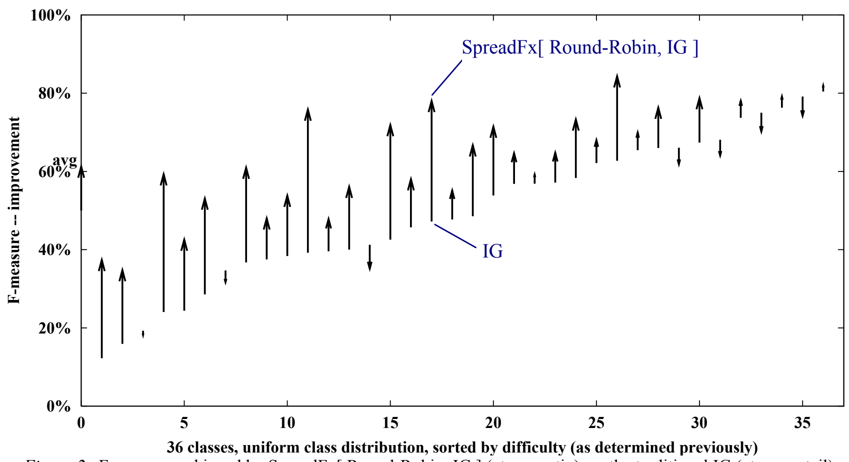

Figure 2. F-measure achieved by SpreadFx[Round-Robin, IG] (at arrow tip) vs. the traditional IG (at arrow tail).

SpreadFx improved the classification accuracy for most of the classes (Figure 2), resulting in a not insubstantial gain overall. The performance advantage for this unnaturally balanced dataset and 18 more normal datasets finally persuaded reviewers that the method was worth publishing.

2 Conclusions

There are three main take-away messages, each for a different audience. For *data mining practitioners*, there is the useful result of the SpreadFx algorithm [1] for robust, performant, and scalable multi-class feature selection. Additionally, the anecdote reminds us that we need to test our systems in settings and on datasets that are as realistic as possible; simplified versions may not reveal serious, lurking problems. For *individual researchers*, we need to get our hands on real-world, unsimplified datasets and tasks in order to discover those problems and to drill down into them to gain useful insights for innovation. For the *broader research community*, we face a persistent issue that it is difficult to publish failures or to publish results on proprietary datasets. Reviewers far prefer elegant work on a conceptually simple problem with improvements demonstrated on public benchmarks for reproducible science. But these simpler problems and public datasets are limited in scope and usually exclude the messy, real-world structure that comes with many interesting and worthwhile problems. Consider the SpreadFx work: what reviewer would take seriously a test problem concocted from public datasets with 100 topic classes plus one extremely different class, say, German documents? The work was only made publishable because it could *also* demonstrate gains for normal, public datasets as well. Yet it has a robustness benefit that is needed for at least some real-world business problems. For the ongoing progress of our science, we increasingly need to face real-world, sometimes messy datasets and, where important new problems are exposed, to accept publication on datasets that will not be publicly available.

References

1. G. Forman. A pitfall and solution in multi-class feature selection for text classification. In *Proceedings of the 21st International Conference on Machine Learning*, ICML '04, pages 38–45, Banff, Canada, 2004. See also HP Labs Technical Report 2004-86, www.hpl.hp.com/techreports/2004/HPL-2004-86.html.
2. Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, ICML '97, pages 412–420, Nashville, USA, 1997.

Generation of an Empirical Theory for Positive-Versus-Negative Ensemble Classification

Patricia E.N. Lutu
Patricia.Lutu@up.ac.za
Department of Computer Science, University of Pretoria
South Africa

Abstract: Scientific theories may be defined as ‘deep, principled explanations of phenomena’. For Computer Science, there are two types of possible theories: mathematical theories and empirical theories. Mathematical theories are founded in mathematical formalism while empirical theories are generated through hypothesis generation, experiment design, collection of experimental results and analysis of these results. It has been observed in the machine learning literature that many problems in machine learning will only be solved through empirical studies, and not through mathematical formulations. The generation of empirical theories is therefore an important activity for machine learning and data mining research. This paper provides an account of how a method for the design of ensemble base models was invented. The method, called positive-versus-negative (pVn) classification, provides ensemble classifiers with a high level of predictive performance. The paper discusses how experimentation with one-versus-all (OVA) classification and the unexpected negative results for OVA ensemble classification eventually led to the invention of pVn classification. This was realised through the systematic application of the steps of the scientific method and the design-science research paradigm. The paper also discusses how some unexpected negative results on pVn classification made it possible to formulate empirical theories on pVn classification.

Keywords: positive-versus-negative classification, one-versus-all classification, ensemble classification, scientific method, design-science research, empirical theories.

1. Introduction

It is generally accepted that scientific theories form the foundations of scientific knowledge. March and Smith [1] have defined theories as ‘deep, principled explanations of phenomena’. Cohen [2] has argued that theories may also be ‘propositions from which we can derive testable hypotheses’. Simon [3] has observed that, in general, there are two types of theories for Computer Science: mathematical theories and empirical theories. Mathematical theories are founded in mathematical formalism while empirical theories are generated through hypothesis generation, experiment design, collection of experimental results and analysis of these results [2]. Simon [3] has argued that there are many aspects of computer systems that are so complex that there are no feasible mathematical theories that can be developed to describe their design and behaviour. The formulation of empirical theories is therefore an important aspect of Computer Science research.

Specific to machine learning and data mining, the mathematical theories we employ largely originate from Computer Science and Statistics [4], and Operations Research [5]. Dietterich [6] has observed that many problems in machine learning will only be solved through empirical studies, and not through mathematical formulations. Cohen [2] has provided comprehensive guidelines on how to conduct empirical research in artificial intelligence and how to generate empirical theories from the empirical studies. According to Cohen [2] progress in science is gradually achieved by moving from descriptions of specific systems to predictive theories and finally to explanatory theories that provide causal explanations for systems in general.

The purpose of this paper is to provide an account of how I discovered an effective ensemble classification modeling method for multiclass prediction tasks, through experimentation and observation. I named this method positive-versus-negative (pVn) classification [7], [8]. pVn classification is a modification of one-versus-all (OVA) classification so that each base model specializes in the prediction of a subset of classes, instead of just one class, as is the case for OVA classification. Empirical enquiry is not an exact science. The researcher hypothesises that given a design and implementation of an artifact based on a set of theories, then a certain outcome / behaviour of the artifact should be realised. However, this is not always the case. The point I make in this paper is as follows: By following the steps of the design-science research paradigm [1] for empirical research, I was able to convert negative experimental results into positive outcomes. I demonstrate this by giving an account of how negative results for studies on OVA classification [7], [9], [10] were converted into positives that led me to the invention of pVn classification. I also demonstrate how further negative experimental results on pVn classification led to the formulation and refinement of a predictive theory for the effective application of pVn classification.

The rest of this paper is organised as follows: Section 2 provides a discussion of theory building and design-science research. Section 3 provides a summary of the theories that I used as a basis for the research activities. Sections 4 and 5 respectively give a brief discussion of the experimental studies on OVA and pVn ensemble classification. The theory that I have formulated on pVn classification is presented in Section 6. Section 7 concludes the paper.

2. Theory building and design-science research

As stated above, empirical theories are generated through hypothesis generation, experiment design, collection of experimental results and analysis of these results. The research activities reported in this paper were conducted through the design-science research paradigm [1] and the scientific method [11], [12]. The scientific method of Peirce and Popper [11], [12] was followed for purposes of building theories based on empirical studies. This method involves observation, hypothesis generation, experiment design, and testing the validity of the hypotheses. Design-science research is a research paradigm that is commonly used in Information Systems research [1], [13] and Operations Research research activities [14]. Design-science [3] and design-science research [1] are concerned with the design and study of artifacts. Hevner et al. [13] have provided the following definition for Information

Systems artifacts: '.. innovations that define ideas, practices, technical capabilities, and products, through which the analysis, design, implementation, and use of information systems can be effectively and efficiently accomplished.'

Design-science research involves two distinct steps. In the first step, an artifact is created. In the second step, an analysis of the usage and performance of the artifact is conducted. The purpose of the analysis is to understand, explain, and possibly improve on one or more aspects of the artifact [15]. According to Hevner et al [13], in the context of information systems, artifacts may be models (abstractions and representations), methods (algorithms and practices) and instantiations (implemented and prototype systems). The outputs of design-science research are constructs, models, methods, instantiations of methods, and better theories. Scientific research is about generating knowledge. For design-science research new knowledge is generated in terms of the new constructs, new models, new methods (the how-to knowledge), and the *better theories* that arise out of the design and evaluation activities. Constructs are the core vocabulary that is used to express the concepts of a field. Knowledge is created when statements or propositions are made to express the relationships between various constructs of the field. *Better theories*, in terms of the models, will result if the models are rigorously tested in order to establish the existence of the relationships.

The discussions provided in this paper are primarily concerned with three of the outputs of design-science research namely, *methods*, *instantiations* and *theories*. The methods presented are concerned with the design of pVn classification models. Instantiations are concerned with the construction and testing of the pVn models to demonstrate their effectiveness. The theory that is presented is a predictive theory on how to effectively apply pVn classification.

3. Existing theories and methods that were used for the research

The design activity for design-science research starts with the consultation of the knowledge base of the field in order to obtain those models, methods and theories that can guide the design of the proposed artifact. This section provides a brief discussion of the machine learning and statistical pattern recognition theories that were used as a basis for the empirical studies discussed in this paper. Section 3.1 discusses discriminative classification. The probably approximately correct (PAC) theory is summarised in section 3.2. Section 3.3 discusses the sources of classification error. Ensemble classification is summarised in Section 3.4. The no free lunch (NFL) theorems for machine learning are discussed in Section 3.5.

3.1 Discriminative classification

There are two well accepted conceptual views of classification, namely: *discriminative classification* [16] and *generative classification*. For discriminative classification [16], a classification model provides a mapping f, from an instance $\boldsymbol{x} = (x_1 ..., x_d)$ in the d-dimensional instance space to a set of classes $\{c_1, ..., c_k\}$. The d-dimensional instance space is viewed as consisting of regions with labels for each of the k classes. The mapping, f, defines the various regions of the instance space. For each class c_i, the union of all the regions with that class label is called the *decision* *region* for the class. The mapping may also be interpreted as a definition of the *decision boundaries* between the *decision regions.* Hand et al [16] have defined a decision boundary between two classes c_i and c_j as a 'contour' or 'surface' in the instance space which has $P_r(c_i,\boldsymbol{x}) = P_r(c_j,\boldsymbol{x}) = 0.5$ where $P_r(c_i,\boldsymbol{x})$ is the probability that instance $\boldsymbol{x}$ has the class label c_i and $P_r(c_j,\boldsymbol{x})$ is the probability that instance $\boldsymbol{x}$ has the class label c_j. The notion of decision boundaries for discrete classification was used as a guide for the design of OVA and pVn base models discussed in this paper. The overriding design objective was to use a large number of training instances that reside in the decision boundary regions of the instance space for the prediction task [7], [8], [9].

3.2 Probably approximately correct theory

The probably approximately correct (PAC) theoretical model of learning proposed by Valiant [17] and discussed by Mitchell [18] has been designed for purposes of characterising algorithms that learn target concepts by generating a hypothesis *h* from a set *H* of all possible hypotheses that belong to some concept class. The learning algorithms use training instances drawn at random according to some unknown, but fixed, probability distribution. PAC is concerned with the identification of classes of hypotheses that can and cannot be learned from a polynomial number of instances. Within the PAC theory various measures of hypothesis space complexity have been proposed for purposes of establishing bounds (sample complexity) for the number of training instances required for achieving a given level of accuracy for inductive learning algorithms. For the studies reported in this paper, the PAC sample complexity theories provided the important insight that, given a prediction task and a training sample size, a classification model which is designed to predict a small number of classes should achieve a higher level of accuracy compared to a model which is designed to predict a large number of classes.

3.3 Sources of error in predictive classification

The prediction error of a classification model has been decomposed into three components namely *bias, variance* and *intrinsic error* [19], [20], 21], [22]. The *bia*s of a predictive model reflects how closely, on average, the (estimated) predictive model is able to approximate the target function. *Bias* reflects the error in the estimation process for the model and is due to the algorithm or inference method as well as the domain for the modeling task [16], [19], [20], 21], [22], [23]. The *variance* reflects the sensitivity of the (estimated) predictive model to the training sample that is used to create the model. Low variance means that the (estimated) model is more stable to the variations introduced by sampling to obtain the training data [16], [20], [23]. The third component of the prediction error is called *intrinsic error* [19], [20], 21]. This is the irreducible component of the prediction error. Various researchers have observed that bias errors can be reduced through the use of simple models and through boosting. Boosting [23] is the practice of directing the greatest effort towards the most difficult aspects of the modeling problem. It has also been observed in the literature that variance error may be reduced through the reduction of model complexity through the use of simple models which are combined through aggregation [7], [8], [25].

3.4 Ensemble classification

Ensemble classification [24] also known as model aggregation [25] is a modeling method that provides classification models with a high level of predictive performance. Several researchers have argued that syntactic diversity of base models leads to a higher level of predictive accuracy for the aggregate model [24], [26], [27], [28]. Several researchers have also argued that a higher level of predictive performance may be achieved by making each member of the aggregate model as competent as possible [27], [28]. Furthermore, Freund and Schapire [29] have demonstrated that boosting results in ensembles with high predictive performance. The theories on bias and variance errors, base model syntactic diversity, competence and boosting were used for the research reported in this paper to guide the design of the base models for OVA and pVn ensemble classifiers. Each base model in an OVA ensemble was created from a different training set. Each base model in a pVn ensemble was also created from a different training set.

3.5 No free lunch theorems for machine learning

In general, no single method can be claimed to be suitable for all datasets and for all algorithms [3], [31]. Schaffer [30] has argued that no single strategy for machine learning is better at generalisation (prediction) than all other strategies for all problem domains. Wolpert [31] has conducted studies on noise-free datasets, and has demonstrated through the *no free lunch theorems for machine learning* that all algorithms are equivalent on average, in terms of predictive performance. The important lesson we learn from these theorems is that it is useful to define (whenever possible) the class of problems for which a proposed machine learning algorithm or method is expected to be effective. The predictive theory reported in this paper were generated for this purpose.

4. Studies of OVA classification

As stated in Section 1, the idea of pVn classification came to me while I was conducting experimental studies on OVA classification. In this section, I briefly discuss the OVA experimental activities in chronological order. Section 4.1 and 4.2 respectively provide a discussion of OVA and boosted OVA experimental results.

4.1 OVA classification

One-Versus-All (OVA) classification is a method where a *k*-class prediction task is decomposed into *k* 2-class sub-problems. One OVA base model is constructed for each sub-problem and the OVA base models are then combined into one ensemble model [10]. Typically, the same training dataset is used for all the base models. The only difference between the base model training datasets is that in each training set the instances for class c_i have class label c_i for base model OVA_i while the instances for all the other classes have class label '*other*'.

Based on the PAC theory, the theory on bias-variance decomposition, syntactic diversity theory, and boosting, I hypothesised that when large quantities of data are available for training, OVA classification can be used to improve predictive performance. I used two large datasets: forest cover type and KDD Cup 1999 [32] and two classification algorithms: 5NN [33] and See5 [34] to create OVA classification ensembles. Each OVA base model was created from a different training set. The details of the experiments that were conducted and the analysis of the results have been reported in [7] and [9]. A summary of the results is given in Table 1. Columns 3 to 6 provide the conclusions for the experiments based on discrete classification and probabilistic classification [35]. The results indicate that the method I proposed for the design of OVA ensemble base models from large datasets worked well for the 5NN algorithm, but did not provide any improvements for the See5 algorithm.

Table 1: Summary of the conclusions from the OVA modeling experiments

		Discrete classification		**Probabilistic classification**	
Dataset	Algorithm	Does mean accuracy increase?	#classes with increased TPRATE	Does mean AUC increase?	#classes with increased AUC
Forest cover type	5NN	yes	5	yes	5
	See5	**no**	1	**no**	3
KDD Cup 1999	5NN	yes	2	yes	2
	See5	**no**	2	**no**	2

4.2 Boosted OVA classification

Since OVA classification worked well only for the 5NN algorithm, I concluded that this must be a no-free-lunch phenomenon, that is, the approach I had used for OVA classification was not suitable for all algorithms. I further hypothesised that if the theory of boosting was applied, this could lead to improved performance for the See5 OVA ensembles. Boosting was conducted as follows: For each OVA base model, the training dataset was composed of instances for the positive class and the negative instances were for those classes that a single model commonly confuses with the positive class. The negative classes were identified by studying the confusion matrices for the single model. This decision was made based on the theory for discriminative classification. As an example, the (combined) confusion matrix (for five test sets) for the forest cover type dataset is given in Table 2. To avoid overfitting, I generated five confusion matrices for a (dataset, algorithm) combination, and then created a combined matrix.

As an example of base model boosting, the training dataset for the boosted OVA1 base model for the forest cover type dataset was composed of instances for class 1 (positive instances) and negative instance were for the classes 2 and 7. The reason for this was that since class 1 gets frequently confused with classes 2 and 7, a model with a large number of training instances from only classes 1, 2, and 7 should provide a classification algorithm with a large amount of relevant data to be able to distinguish between class 1 and the negative classes 2 and 7. The details of the boosted OVA experiments have also been reported in [7] and [9]. A summary of the results is given in Table 3. One can see that while boosting worked for the forest cover type models for both the 5NN and See5 algorithm, it did not work well for the KDD Cup 1999 dataset. Also, the performance improvements for the forest cover dataset were very small. Again, I concluded that the no-free-lunch phenomenon was the reason for the lack of performance improvement. In other words, boosting works well only for some (domain, algorithm) combinations.

Table 2: Confusion matrix for See5 classification tree single 7-class model for forest cover type (training set size = 12000, test set = 250 per class)

Actual class	Predicted class						
	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
Class 1		60			3	2	38
Class 2	43		5		32	8	8
Class 3				26	11	50	
Class 4			6				
Class 5		17	6			6	
Class 6		4	30	23	3		
Class 7	16						

Table 3: Summary of the conclusions from the boosted OVA modeling experiments

Dataset	Algorithm	**Discrete classification**		**Probabilistic classification**	
		Does mean accuracy increase?	#classes with increased TPRATE	Does mean AUC increase?	#classes with increased AUC
Forest cover type	5NN	yes	6	yes	6
	See5	yes	3	yes	5
KDD Cup 1999	5NN	**no**	0	yes	2
	See5	**no**	2	**no**	2

5. Studies of pVn classification

In this section I provide an account of how I moved from boosted OVA classification to pVn classification. pVn classification [7], [8] involves the decomposition of a k-class prediction task into j $(j < k)$ sub-problems. One base model is constructed for each sub-problem to predict a subset of the k classes. The base models are then combined into one ensemble model for prediction. The classes that the model can predict are called the positive classes or p-classes. The other classes that are included in the training data for the model are called negative classes or n-classes. The reasons for the derivation of confusion graphs from confusion matrices are discussed in Section 5.1. Section 5.2 provides a brief discussion of the preliminary theories that I formulated for pVn classification. Section 5.3 provides a discussion of the activities I am currently involved in. These activities are concerned with studies on whether pVn classification is effective for datasets with large numbers of classes.

5.1 From confusion matrices to confusion graphs

I further hypothesised that using more than one positive class in a model could lead to performance improvements compared to the use of just one positive class. In other words, a model should concurrently learn the decision boundaries of all the classes that were used in the boosted OVA model. For example, a model with positive classes 1, 2 and 7 could provide better performance compared to the boosted OVA counterpart. Here, I was still using the PAC theory (simple models) and boosting theory (concentrate most effort on the difficult aspects for classification). The first task was to identify groups of positive classes for each base model. After staring at confusion matrices for a while, I became confused! I decided to use a more graphical representation of a matrix. A graph was an obvious choice since I was looking for adjacency of class decision regions in the instance space. In computation we commonly use an adjacency matrix as a data structure for representing a graph. So, I invented the confusion graph [7], [8]. An example of a confusion graph is given in Fig. 1. Each class is represented by one node in the graph. An arc between two nodes indicates that there is confusion between the classes shown in the nodes. The arc label shows the confusion count given in the matrix. The value in brackets in each node gives the level of connectivity for the node.

	Predicted class				
		C1	C2	C3	C4
Actual class	C1	▨	15	25	
	C2	20	▨		20
	C3		10	▨	
	C4				▨

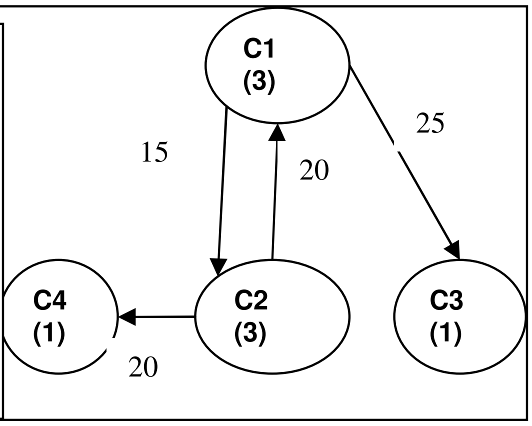

Fig. 1: Confusion matrix and confusion graph for a hypothetical classifier (adopted from [36])

I designed a manual (pen-and-pencil) algorithm for analysing a confusion graph to identify the base models and base model classes. This algorithm is described in detail in [7] and [8]. Manual application of the algorithm was made possible due to the fact that the datasets I studied initially had a moderate number of classes: five classes for the KDD Cup 1999 dataset and seven classes for the forest cover type dataset. I used the same datasets as for the Section 4 experiments to test the merits of pVn classification. I found that pVn classification provided performance improvements for both the forest cover type dataset and the KDD Cup 1999 dataset for the 5NN and See5 algorithms [7], [8]. The classification performance results are summarised in Table 4.

Table 4: Summary of the conclusions from the pVn modeling experiments

Dataset	Algorithm	Discrete classification		Probabilistic classification	
		Does mean accuracy increase?	#classes with increased TPRATE	Does mean AUC increase?	#classes with increased AUC
Forest cover type	5NN	yes	4	yes	7
	See5	yes	3	yes	6
KDD Cup 1999	5NN	yes	4	yes	4
	See5	yes	3	yes	4

5.2 Preliminary theory for pVn classification

In the process of experimentation I realised that pVn modeling is made possible when the confusion matrix of the single *k*-class model has a high level of sparsity, that is, when there is a large number of off-diagonal cells with zero counts. Based on the empirical studies that I conducted, I concluded that pVn classification is effective when the sparsity level is between 40% and 60%. Table 5 provides a summary of the sparsity level of the confusion matrices that were used as a basis for pVn base model design.

Table 5: Summary of confusion matrix properties (adopted from [8])

Dataset (number of classes)	Number of off-diagonal cells in confusion matrix	Algorithm	Sparsity count (cells with count < 5)	Percentage level of sparsity
Forest cover type (7 classes)	42	5NN	23	57%
		See5	25	60%
KDD Cup 1999 (5 classes)	20	5NN	10	50%
		See5	8	40%

During experimentation I came to realise that even when the confusion matrix has a high level of sparsity, it may be the case that each node in the corresponding confusion graph has a (in-going or out-going) arc to every other node in the graph. This was the case for example, for the KDD Cup 1999 dataset and the See5 algorithm. It then became necessary to further simplify the confusion graph as discussed in [7] and [8].

5.3 Studies on a dataset with a large number of classes

Given the foregoing observations, I hypothesised that pVn classification should especially be useful for datasets with a large number of classes (e.g. > 20 classes). I used the letter recognition dataset from the UCI machine learning repository [32] to conduct experiments to investigate if there was merit in this hypothesis. When a dataset has a large number of classes, it is infeasible to manually apply the pVn base model design algorithm as I had applied it previously. I therefore created a new version of the algorithm and implemented it in Borland C++ builder. The algorithm directly processes the confusion matrices. I created a training dataset of 10,400 training instances and 1,820 validation instances, and five test (holdout) sets each with 1,820 instances using stratification and simple random sampling to achieve a balanced class distribution. I created an ANN MLP [37] classification model using the

IBM SPSS version 20 and generated five confusion matrices using the five test sets. The five confusion matrices were used as input to the pVn base model design algorithm which outputs the base model specifications. The combined confusion matrix that was generated by the algorithm and used for the base model design specification is given in the appendix. Since I used five confusion matrices, I again initially used a cut-off value of 5, so that the algorithm interpreted any matrix cell with a count of less than five as a blank cell. Row 2 of Table 5 gives a summary of the pVn model specifications that were produced by the algorithm for a cut-off value of 5. One can see that, on average, each model should be trained on data for 23 out of 26 classes. This was an unexpected outcome. Two questions then arose: (1) Is it necessary to elaborately select the negative classes, or can the model simply use all the classes that are not p-classes as the n-classes? (2) Would a higher value for the cut-off point result in a reduced number of identified n-classes? The use of higher values for the cut-off point resulted in a reduction of the number of n-classes, but then exposed another potential problem. Rows 3 and 4 of Table 6 provide a summary of the pVn model specifications that were produced by the algorithm for cut-off values of 10 and 20. The main problem is that there are thirteen classes that are not included in any base model specification when a cut-off value of 20 is used. This is 50% of the letter recognition classes.

Table 6: Effect of cut-off value on base model specifications

Cut-off value	Number of models	Average number of p-classes and n-classes per model	Comments
5	22	p-classes: 6, n-classes: 17	large number of n-classes
10	22	p-classes: 4, n-classes: 6	moderate number of n-classes
20	10	p-classes: 2, n-classes: 2	(1)small number of n-classes (2) 13 classes do not appear as a p-class in any model

To answer the first question posed above, I created one of the pVn base models whose specification was produced using a cut-off value of 5. This base model has 5 positive classes (p-classes): {A, D, J, K, Q, S} and 17 negative classes (n-classes): {B,H,I,N,O,R,X,Z,C,G,F,L,T,E,Y,U,P}. The first implementation of this model (version 1) used the specified n-classes as the negative classes while the second implementation (version 2) used all the classes that are not p-classes as the negative classes. The classification results for the single *k*-class model and the two versions of the pVn base model are given in Table 7. One can see that both versions of the pVn base model do not provide significant predictive performance improvements compared to the single k-class model. When the pVn base models cannot provide performance improvements on the individual classes compared to the single mode, it is not possible for the pVn ensemble to provide improved performance [7].

A comparison of the letter recognition confusion matrix (given in the appendix) revealed that there is a big difference in the distribution of the confusion counts across the matrix, compared to the confusion matrices for the forest cover type and KDD Cup 1999 datasets. As an example, the confusion matrix given in Table 2, for forest cover type and the See5 algorithm, shows that there are many cells with counts that are much larger than the counts which appear in the other cells. In contrast, the letter recognition confusion matrix contains many small counts that appear almost everywhere in the matrix. I measured the sparsity level of the letter recognition confusion matrix (using a cut-off value of 5) and found it to be: 77%. This result by itself could not provide a clear indication as to why the pVn base model design did not work well for the letter recognition dataset. I conducted a statistical analysis of the confusion matrices in order to establish the nature of the distribution of the counts in the matrix cells. I used the Gini index of heterogeneity [23] for this purpose. Given a qualitative variable *X* with *L* levels and *n* observations for the variable, a statistical measure of heterogeneity summarises the level of distribution of the *n* observations among the *L* levels. Null heterogeneity means that all the *n* observations have the same level for *X*. Maximum heterogeneity means that the observations are uniformly (equally) distributed among the *L* levels. The Gini index of heterogeneity is defined as [23]

$$Gini = 1 - \sum_{i=1}^{L} p_i^2 \tag{1}$$

where $p_i = count(i)/n$ is the proportion of observations that have level *i*. This index takes on values in the interval [0,1]. For a confusion matrix, I treated each row of the matrix as a qualitative variable where each cell in the row corresponds to one level of the variable. I computed the Gini index of heterogeneity for each matrix row and summarised the results as shown in Table 8. The results of Table 8 indicate that the level of heterogeneity in the confusion matrices for forest cover type and KDD Cup 1999 (for which pVn classification worked well) is much lower than the heterogeneity for the letter recognition dataset. The conclusion that I made from this observation was that the level of heterogeneity in the confusion matrix counts determines whether pVn classification can be used successfully for a given dataset and algorithm.

Table 7: Classification results for one letter recognition pVn base model

Class	Class true positive rate TPRATE% and 95% CI of mean for:		
	Single model	pVn base model version 1	pVn base model version 2
A	88.6 ± 1.8	82.8 ± 2.3	88.3 ± 2.7
D	87.7 ± 1.4	86.3 ± 4.4	87.7 ± 1.4
J	77.9 ± 1.9	74.3 ± 5.1	82.5 ± 2.2
K	72.9 ± 3.7	45.1 ± 9.7	69.7 ± 3.0
Q	75.2 ± 4.8	78.8 ± 4.9	78.3 ± 2.8
S	64.6 ± 2.1	12.3 ± 19.2	66.0 ± 3.2

Table 8: Levels of heterogeneity for confusion matrices

Dataset (classes)	Algorithm	Gini index of heterogeneity			
		Minimum	Maximum	Mean	Heterogeneity is:
Forest cover type (7 classes)	5NN	0.46	0.71	0.58	medium
	See5	0.00	0.67	0.42	low
KDD Cup 1999 (5 classes)	5NN	0.12	0.55	0.38	low
	See5	0.18	0.59	0.33	low
Letter recognition (26 classes)	ANN MLP	0.67	0.91	0.85	high

6. Summary of the current predictive theory for pVn classification

To date, I have conducted experiments for pVn classification on two large datasets (forest cover type and KDD Cup 1999), one medium sized dataset (letter recognition) and one small dataset (wine quality). I have used four algorithms namely: 5NN, See5 CART and ANN MLP for the experiments. I have found that pVn classification provides performance improvements in all cases except for the letter recognition dataset. The *no free lunch theorems* for machine learning tell us that no single method can be claimed to be suitable for all datasets and for all algorithms [31]. At this point in my research activities on pVn classification I have formulated a number of empirical predictive theories specifying the characteristics of problems for which pVn classification should provide performance improvements. I have identified these characteristics based on the positive results and negative results that I obtained through experimentation. The predictive theories are as follows:

Condition 1: If a multiclass prediction task has more than four classes and less than ten classes, then pVn classification may provide performance improvements.
Condition 2: If condition 1 holds and the confusion matrix for the single *k*-class model has a sparsity level between 40% and 60% then pVn classification may provide performance improvements.
Condition 3: If condition 2 holds, but the level of heterogeneity in the rows of the confusion matrix is low then pVn classification will provide performance improvements, otherwise it will not.
Condition 4: If a multiclass prediction task has a large number of classes (e.g. more than 20) and the level of heterogeneity in the rows of the confusion matrix is high then pVn classification will not provide performance improvements (still a weak predictive theory)

I call condition 4 a weak predictive theory since I have based my conclusions on a single dataset and one algorithm. I will need to conduct studies on more datasets with large numbers of classes and more classification algorithms in order to obtain more evidence to support the claim for this condition.

7. Conclusions

The purpose of writing this paper was to provide an account of how I invented pVn classification, which I have found to be an effective ensemble classification modeling method. I achieved this through experimentation and observation based on the design-science research paradigm and the scientific method of conducting research. It is important to note that many of the negative results (dark clouds) that I encountered provided useful insights (silver linings) that guided me towards hypothesis formulation for pVn classification, and formulation of predictive theories on the conditions when pVn classification should be applicable for multiclass prediction tasks.

Appendix

This appendix provides tables with detailed results of experiment outcomes. The confusion matrix for the letter recognition dataset consists of 26 x 26 = 676 cells. Only the top half of the matrix is given in Tables A-1a and A1-b. Table A-2 provides the values for the Gini index of heterogeneity for letter recognition confusion matrices for the ANN MLP classification models. Table A-3 provides the values for the Gini index of heterogeneity for forest cover type (FCT) and KDD Cup 1999 confusion matrices for the 5NN and See5 classification algorithms.

Table A-1a: Combined confusion matrix for letter recognition: rows A to M , columns A to M

Actual class	Predicted class												
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
A	▨	1		6				1		5	7		
B		▨		10		5	1	3	2				
C			▨		16		25				14		4
D	2	18		▨		1	2	2					2
E		12	6		▨		24					5	
F		7	12		3	▨	1	3	1	2	2		
G		4	6	2	3	2	▨	4			9	11	6
H	2		1	17			7	▨		2	10		10
I	1	2	6	9	1	7			▨	11		10	
J	10			6	1	3	1	4	18	▨			
K	1	5	3	5	3		3	10		1	▨	1	4
L			6	2	6		4	2	2		1	▨	2
M	3	4						2			1		▨

Table A-1b: Combined confusion matrix for letter recognition: rows A to M ,columns N to Z

Actual class	Predicted class												
	N	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
A	1	1		2	1	8	3	1				3	
B		2		5	20	12					5		1
C	3	2					4	12					
D	3	3		2	3		4						1
E					9	5	10				5		19
F	5		10			5	7		3	6	2	12	
G		5	4	35	9	8			1	2	6		1
H	12	26	1	2	27	1	6	10	13		2	2	
I			2		1	14							3
J		2	2	1	2	15	2				3		7
K				6	29	3	4	2			15		
L	1			8	5	1		1			16	2	
M	14		1		11			3					

Table A-2: Gini index of heterogeneity for Letter recognition confusion matrices

Gini index of heterogeneity			
class	**Gini**	**class**	**Gini**
A	0.874	**N**	0.889
B	0.831	**O**	0.901
C	0.802	**P**	0.790
D	0.790	**Q**	0.889
E	0.848	**R**	0.867
F	0.907	**S**	0.884
G	0.872	**T**	0.874
H	0.894	**U**	0.748
I	0.866	**V**	0.803
J	0.867	**W**	0.668
K	0.853	**X**	0.894
L	0.869	**Y**	0.866
M	0.765	**Z**	0.867
Summary statistics for the Gini coefficient values for the 26 classes		**Mean**	**0.845**
		Minimum	**0.668**
		Maximum	**0.907**

Table A-3: Gini index of heterogeneity for the FCT and KDD Cup 1999 confusion matrices

Dataset	**Class**	**Gini index of heterogeneity**	
		5NN algorithm	**See5 algorithm**
Forest Cover Type	C1	0.605	0.523
	C2	0.708	0.672
	C3	0.550	0.564
	C4	0.463	0.000
	C5	0.594	0.571
	C6	0.635	0.596
	C7	0.490	0.000
Summary statistics for the Gini coefficient values for the 7 FCT classes	**Mean**	**0.578**	**0.418**
	Minimum	0.463	0.000
	Maximum	0.708	0.672
KDD Cup 1999	NORMAL	0.553	0.447
	DOS	0.409	0.585
	PROBE	0.391	0.176
	R2L	0.116	0.235
	U2R	0.411	0.219
Summary statistics for the Gini coefficient values for the 5 KDD Cup 1999 classes	**Mean**	**0.376**	**0.332**
	Minimum	0.116	0.176
	Maximum	0.553	0.585

References

1. March, S. T., Smith, G. F. : Design and natural science research on information technology. Decision Support Systems*,* 15, 251-266 (1995)
2. Cohen, P.R. : Empirical Methods in Artificial Intelligence, MIT Press, Cambridge, Massachusetts (1995)
3. Simon, H. A. : The Science of the Artificial, 3rd Edition, Cambridge, MA, MIT Press (1996)
4. Smyth, P. : Data Mining at the interface of computer science and statistics. In Grossman, R. L., Kamath, C., Kegelmeyer, P., Kumar, V., Namburu, R. R. (eds.) Data Mining for Scientific and Engineering Applications. Dordrech, Netherlands, Kluwer Academic Publishers (2001)
5. Olafsson, S., Li, X., Wu, S. : Operations Research and data mining. European Journal of Operations Research*,* 19 (2) 113-126 (2008)
6. Dietterich, T. : Fundamental experimental research in machine learning. Available at: http://web.engr.oregonstate.edu/~tgd/experimental-research/index.html (1997), (accessed: 27 October, 2009)
7. Lutu, P. E. N. : *Dataset Selection for Aggregate Model Implementation in Predictive Data Mining*. PhD thesis, Department of Computer Science, University of Pretoria. Available at: http://upetd.up.ac.za/thesis/available/etd-11152010-203041/ (2010)
8. Lutu, P. E. N., Engelbrecht, A.P. : Positive-versus-negative classification for model aggregation in predictive data mining (2012). To appear in the INFORMS JOC.
9. Lutu, P. E. N., Engelbrecht, A.P. : Using OVA modeling to improve classification performance for large datasets. Expert Systems With Applications*,* 39 (4) 4358-4376 (2012)
10. Rifkin, R., Klautau, A. : In defense of one-vs-all classification. The Journal of Machine Learning Research, 5, 101-141 (2004)
11. Osei-Bryson, K.-M., Ngwenyama, O. K.: Using decision tree modelling to support Peircian abduction in IS research: a systematic approach for generating and evaluating hypotheses for systematic theory development. Information Systems Journal, 21 (5) 407-440 (2011)
12. Oates, B. J. : Researching Information Systems and Computing*,* London, SAGE Publications (2006)
13. Hevner, A. R., March, S. T., Park, J., Ram, S. : Design science in information systems research. *MIS Quarterly,* 28 (1) 75-105 (2004)
14. Manson, N. J. : Is Operations Research really research? Orion, 22 (2), 155-180 (2006)
15. Vaishnavi, V., Kuechler, W. : Design Research in Information Systems. URL: http://desrist.org/design-research-in-information-systems (2004/5), (accessed: 27 October, 2009)
16. Hand, D. J., Manila, H., Smyth, P. : Principles of Data Mining, Cambridge, MA, MIT Press (2001)
17. Valiant, L. G. : A theory of the learnable. Communications of the ACM, 27 (11), 1134-1142 (1984)
18. Mitchell, T. M. : Machine Learning, Burr Ridge, IL, WCB/McGraw-Hill (1997)
19. Van der Putten, P., Van Someren, M.: A bias-variance analysis of a real world learning problem: the COIL challenge 2000. Machine Learning, 57, 177-195 (2004)
20. Friedman, J. : On bias, variance, 0/1-loss, and the curse-of-dimensionality. Data Mining and Knowledge Discovery, 1 (1), 55-77 (1997)
21. Kohavi, R., Wolpert, D.H. : Bias plus variance decomposition for zero-one loss functions. In Saitta, L. (ed.) Machine Learning: Proceedings of the Thirteenth International Conference, 275-283. Morgan Kaufmann (1996)
22. Geman, S., Bienenstock, E., Doursat, R. : Neural networks and the bias/variance dilemma. Neural computation, 4, 1-58 (1992)

23. Giudici, P. : Applied Data Mining: Statistical Methods for Business and Industry, Chichester, John Wiley & Sons (2003)
24. Hansen, L. K., Salamon, P. : Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence. 12 (10), 993-1001 (1990)
25. Breiman, L. : Bagging predictors. Machine Learning, 24 123-140 (1996)
26. Sun, J., Li, H. : Financial distress prediction based on serial combination of multiple classifiers. Expert Systems With Applications, 35 (5), 818-827 (2008)
27. Ali, K. M., Pazzani, J. : Error reduction through learning multiple descriptions. Machine Learning, 24 ,173-202 (1996)
28. Krogh, A., Vedelsby, J. : Neural network ensembles, cross validation and active learning. In Tesauro, G., Touretzky, D. S., Leen, T. K. (eds.) Advances in Neural Information Processing Systems. Cambridge MA: MIT Press (1995)
29. Freund, Y., Schapire, R. : A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55 (1), 119-139 (1997)
30. Schaffer, C. : A conservation law for generalisation performance. Proceedings of the Eleventh Conference on Machine Learning, 259-265, CA: Morgan-Kaufmann (1994)
31. Wolpert, D. H. : The lack of a priori distinctions between learning algorithms. Neural Computation, 8(7) 1341-1390 (1996)
32. Frank, A., Asuncion, A. : UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science (2010)
33. Cover, T. M., Hart, P. E. : Nearest Neighbor Pattern Classification. IEEE Transaction on Information Theory, IT-13 (1), 21-27 (1967)
34. Quinlan, J. R. : An Informal Tutorial, Rulequest Research. (2004) URL: http://www.rulequest.com (accessed: 28 October, 2005).
35. Fawcett, T. : An introduction to ROC analysis. Pattern Recognition Letters, 27, 861-874 (2006)
36. Lutu, P. E. N. : Using confusion graphs and confusion matrices to design ensemble base models for classification, in A. Cuzzocrea, U. Dayal (eds.) Proceedings of the 13th International Conference on Data warehousing and Knowledge Discovery, DaWak 2011, Toulouse, France, August / September 2011, Lecture Notes in Computer Science, LNCS 6862, 301-315 (2011).
37. Bishop, C. M. : Neural Network for Pattern Recognition Oxford, Clarendon Press (1995)

How to make the most of result evaluation?

Ana Costa e Silva

LIAAD-INESC Porto,
Rua de Ceuta, 118, 6, 4050-190 Porto, Portugal
http://www.liaad.up.pt

Abstract. Often, when evaluating their algorithms, authors will present their performances according to a set of metrics run over a given test dataset. They may then present the sensitivity of results to changes in parameters, and that's it. We believe this approach is incomplete. We propose a more comprehensive evaluation approach that encompasses, where applicable, four steps: 1. drawing a chart with the Production Possibility Frontier against different users' utility curves, to match user preferences against available algorithms; 2. using statistical (not t-) tests, to validate the relative worthiness of different algorithms; 3. specific evaluation, to determine whether the winning algorithm is so for all types of observations in the dataset or just a few, and whether it is possible to combine competing algorithms to improve efficiency/efficacy; 4. flagging, to bring to light, in a mathematically robust manner, the conditions in which the favoured approach is likely to make mistakes and to supply this information to the user. In this paper, we shall be present each of these techniques. We shall then apply them to two different real experiments in the table analysis domain.

1 Introduction

It is a commonly used evaluation strategy to run competing algorithms on a test dataset and state which performs better in average on the whole set. Often, an analysis of the impact on performance of different choices of parameter settings is also presented. We call this *generic evaluation*. Although it is important, we believe this type of evaluation is incomplete.

1. To begin with, an algorithm may present better results than another on some relevant metrics but not all metrics and a choice between such algorithms is in fact entirely dependent on user preferences and making this explicit is important. There are a number of approaches for doing this. We propose drawing a *Production Possibility Curve* against users' specific utility curves.
2. Secondly, an algorithm may present better results than another on a given dataset, but the difference may not be big enough be likely to extrapolate to all likely cases. To overcome this problem, adequate *statistical tests* of results are required [1] (and we do not mean t-tests).
3. Furthermore, an algorithm may on average be worse than another on the whole of the dataset, but it could be better on an identifiable subset of the

data and determining this could lead to the identification of a combination policy between them that would either increase overall results or overall productivity. *Specific evaluation*, also called *SpecVal*, is the methodology we developped for this purpose. Without specific evaluation, we cannot prove a single algorithm to be the best on the whole dataset.

4. Last but not least, not enough attention is given, in most papers, to in what circumstances the favoured algorithm makes mistakes. We suggest using *Flagging* for this purpose. Identifying this not only facilitates the understanding of when the algorithm can or cannot be expected to perform well, which is relevant for any potential users, but also it can actually help improve the original algorithm, potentially highlighting aspects of easy improvement.

In this paper, we present our methodologies for each of these four techniques. We shall then implement them along side generic evaluation on two experiments. Experiment 1 aims at locating tables in ASCII files and we shall see how flagging helps improve the algorithm's performance. Finally, in the second experiment, which aims at grouping table cells into columns, specific evaluation allows determining a policy for dividing the search space in two regions, one where the overall winning algorithm must be used, and another where the simpler algorithm performs just fine. Combining them allows an increase in efficiency when compared to just using the winning algorithm.

2 The Four Explained

2.1 Production Possibility Curve and Utility Curves

Suppose we want to compare two algorithms on substitute performance metrics, such as performance and recall, which are best for classification problems, or their counterparts for grouping and diviging problems, completeness and purity [6]. If one algorithm has one lower and one higher etric than another, both algorithms are efficient and a choice between them is always user specific. Different approaches can be used for tailoring alternative efficient algorithms to users' needs. Some common ways are to minimise total cost or to maximise the harmonic mean between the metrics. However, because there can be as many preferences as there are users, we would have to compute different CPFs/costs as there are user groups. We shall here look at a more visually exticing alternative, first developed in Microeconomics, the science of optimising the attribution of scarce resources to vast needs.

When we represent the results of all algorithms that are available for a given task on a scatter plot that has the two substitute metrics for axes, and we connect the efficient algorithms to each other, we obtain a line that represents the maximum combinations of the metrics that is attainable with the resources available. This is called a Production Possibility Curve (PPC).

The degree to which one user prefers completeness to purity or vice-versa can also be represented in the same chart using Utility Curves (UCs) [9]. A utility curve joins together all the combinations of C&P that please a user equally;

preference sets are normally represented as parallel lines; UCs closer to the top right corner of the plot, i.e. farther from the origin, represent a higher level of utility for the user, leaving him more satisfied; the UCs of a user who is consistent in his preferences never cross.

In Figure 1, we draw a PPC from the results of five different algorithms. Only algorithms D and E are efficient, since the remainder represent simultaneously less C&P than these two; therefore only D and E appear on the PPC. We also draw two sets of UCs representing the preferences of two different users, X and Y. User X likes purity and completeness equally, thus the slope of his utility curves is -1, while user Y considers purity to be 6 times less serious a mistake than completeness, thus his UCs' slope is $-1/6$.

To determine each user's favourite algorithm, we identify the point where the user's UCs intersect the PPC farthest from the origin. We can clearly see that both users prefer method E, although user X likes D almost as much.

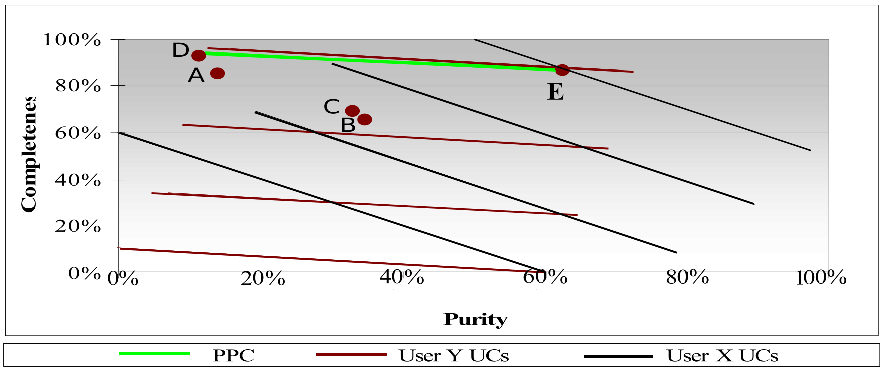

Fig. 1. A user's favourite algorithm lies on the farthest from the origin intersection between his/her Utility Curves and the PPC.

In Figure 1, we represent each alternative algorithm as a point, which corresponds to a particular choice of parameter settings. Now suppose that, by varying the choice of parameters, we can achieve a continuous combination of completeness and purity such as the blue lines in Figure 2. In that case, we can draw the PPC as the addition of the right-most portions of either lines, thus forming the orange line in the figure. Against this, we can draw the utility curves to simultaneously determine the algorithm settings and the parameters that maximise different users' utility. In the example in Figure 2, the orange dot represents the point on the PPC that both users happen to prefer, the PPC being constructed as the outermost path along either one of the available algorithm performances; as such, both users prefer algorithm B, with the precise choice of settings that led to the orange dot.

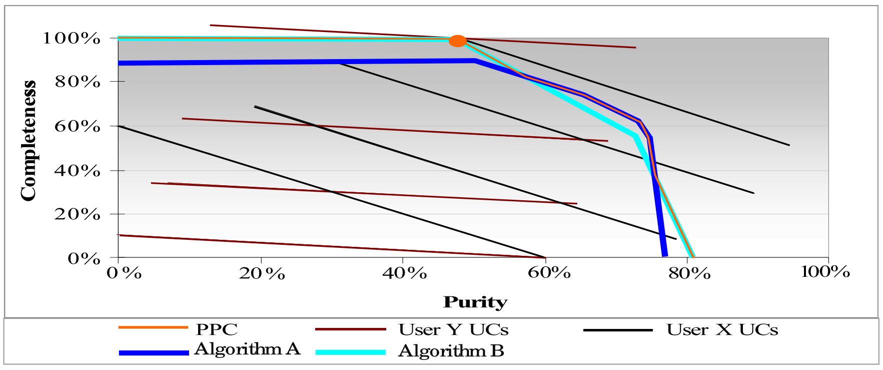

Fig. 2. Here the PPC is constructed as the outermost path along either one of the available algorithm performances; both users prefer algorithm B at the choice of parameters that gives rise to the performance of the orange dot.

2.2 Testing the statistical validity of results

One algorithm may seem better than another when we compare their performance metrics together, however the difference might not be statistically significant, i.e. they might not derive from an actual difference in the goodness of the algorithms but instead might be due to chance from running the experiment on this particular data on not another (this is called sampling error). One method that does not present a statistically significant in either completeness or purity to another can be considered its equivalent.

T-tests are not adequate for this purpose among other reasons because they require the difference between algorithms to follow the normal distribution, which normally it does not. Without the verification of the underlying requirements, all statistical tests are irrelevant.

We shall be using some of the statistical tests suggested by [1], who explains that the following tests are appropriate for the purposes of comparing average algorithm performance over a set of datasets:

- for comparing two algorithms to one another, the Wilcoxon signed-ranks test is suggested. It is a ranks test that, to be valid, requires only that the data be independently generated;
- for comparing several algorithms to one another, it is best to start with the Friedman test as corrected by [2] to determine whether there exist significant differences among the overall results of the different algorithms; and then apply the Nemenyi test [3] to compare any pairs of algorithms to each other.

It is important to here call attention to the fact that it is incorrect to use cross-validation for this purpose. In fact, because of the overlap between the different training sets, there is no independence in the results obtained in each test set. As such, although average results can be adequately estimated, their variance cannot and so any statistical tests are biased. Independent sampling is the major common requirement for the vast majority of statistical tests (if not all); even paired samples require each pair be independent from the other).

2.3 Specific evaluation, *SpecVal*

From the two previous techniques, it should have transpired which of the algorithms under competition is the best on the dataset as a whole. Specific evaluation will further assert its goodness by asking "Is this always the case?". We shall be applying a systematic methodology developed in [7] for determining whether and in what conditions one algorithm performs better than another on an identifiable subset of the reference universe.

A posteriori determining which algorithm works best for which parts of the data is interesting in itself, but if we manage to, with some degree of accuracy, determine it a priori, then we can devise a policy for deciding which algorithm to use under which circumstances and as such increase results or efficiency.

Specval methodology

1. Metrics for winner choosing. Choose the relevant performance metrics.
2. Winner choosing. For each record , identify which algorithm works best.
3. Metrics for characterisation. Characterise each record of the dataset on metrics built using the results of all algorithms plus whatever other known metrics are reasonable on each element of the dataset, trying to compile as large a set of features as possible. The idea behind these features is that they should capture the essence of a well obtained result.
4. Analysis. Present the above data to your choice of data analysis algorithm. Different algorithms are potentially more appropriate for different tasks and research goals. Forexample, we used principal component analysis coupled with decision trees in [7], but use decision trees alone in this paper. Regardless of how you choose to analyse your data, include a classifier that will, given the metrics, predict which algorithm works best on each data item and output the respective probability.
5. Result 1. If the model does not extrapolate well to the test set, specific evaluation can be considered inconclusive. We come to this result in section 3.1. If it does extrapolate and if applicable, interpret the model(s) built, as it(they) divide(s) your universe of reference into subsets that were mathematically obtained and are more valid for your problem than any others recommended in the literature. We do this in section 3.2.
6. Deployment. Apply the classifier to each observation in the test set in order to determine which algorithm is more likely to be the best for it; apply the favourite algorithm for it.
7. Result 2. Measure the performance of this combined result against that of applying the algorithms independently.

- If composite performance is higher, this means that:
    - none of the algorithms is better than all others in all cases,

  - results are maximized if each algorithm is applied on the subset of the world it is an expert on,
  - you have developed a methodology that can be used for combining competing algorithms,
- If composite performance is equivalent and the independently worse algorithm is the simplest (we come to this result in section 3.2), then:
  - you can apply computational cheaper algorithms to parts of the dataset, thus increasing efficiency with little risk of lowering efficacy,
  - you have developed a methodology that can be used for combining competing algorithms,
- If composite performance is equivalent and the independently worse or equal algorithm is the most complex, OR if the composite approach performs worse than the independently winning algorithm's, then call this a absolute king in its field (we come to this results in [7], section 4).

Notice that defining policies for combining algorithms is not very original. Mixture of experts, for example, have been doing this for years. The novelty is in proposing that performance evaluation be considered incomplete without the verification of the ability of algorithms to perform well in the different types of cases, and not just the majority of cases present in the dataset. The novelty is showing how a probabilistic-based combination policy can be achieved via the application of a simple methodology. Last but not least, the novelty is also in applying combination techniques to table location and column segmentation.

2.4 Flagging

After all this, you now have your favourite algorithm for handling the problem at hand, be it an independent algorithm or a combination between available algorithms. Can it still be improved? And can you distinguish cases where the approach is likely to be more or less successful?

The idea behind flagging is really quite simple. Basically, using the features compiled for specific evaluation, under "'Metrics for characterisation"', one can build a classifier for distinguishing when the algorithm obtains the correct response from those when mistakes occur. We recommend using a decision tree because it provides humanly readable rules for how to cut up the objects in the dataset into subsets with common characteristics, some of which will correspond to areas of high performance and others of low performance of the algorithm under test. Being able to verbalise this, i.e. to understand the conditions of victory and failure of your algorithm is as important as disclosing its average results and more general than providig random examples of success and failure.

Another side effect of specific evaluation and flagging is that the more capable features at grasping the goodness of results in your problem, among all you compiled in step 3 of specific evaluation, will become evident.

In the following sections, we will see the practical application of these techniques over a set of two experiments in table research.

	0	1	2	3	4
Completeness	67.0%	67.0%	64.3%	69.6%	82.6%
Purity	9.6%	34.8%	44.9%	52.2%	38.1%
CPF - User X	17.3%	45.8%	52.9%	59.6%	52.2%
CPF - User Y	40.4%	59.1%	60.6%	66.4%	70.8%
Total cost - User X	211	192	200	173	137

Table 1. Results that compare our different models' performance on the test set, for a user X who likes cmopleness and purity the same and a user Y who prefers completeness 6 times more than purity

3 The Four In practise

In order to conduct our experiments, we collected 22 different PDF documents from the internet, which we converted to ASCII using the pdttotxt 2002 linux utility. As a result of the conversion, each line of each document became a line of ASCII, which when for example imported into Access becomes a record in a relational database. Some alignment issues result from this conversion. We ground-truthed the data manually. Our documents have lengths varying between 13 and 235 pages with very diverse page layouts; each document contains between 3 and 162 tables. In total, we have 96,164 lines and 1,196 tables.

3.1 Experiment 1

This experiment aims at detecting tables in ASCII files. The task we are testing involves identifying likely table boundaries and then, taking the areas within, classifying them as table or not. Because the task involves grouping lines into tables, completeness and purity are adequate performance metrics.

The research question is "'How best to build an HMM for table location?"'. We shall be comparing the results of five different algorithms: model 0 is a non-HMM baseline, and models 1 to 4 HMM configured in different manners, [5].

In table 1, we show the results of each algorithm. We shall be exploiting them much further in the subsequent headings. Notice that they seem quite low, but considering the difficulty of the data with which we operate, they are not [8].

Production Possibility Curve and Utility Curves Model 4 is the model for which the number of complete and pure tables found is maximised. Is is also the model for which total cost is at its minimum. However, this total cost and CPFs portray only personal and non-transferable sets of preferences. In order to obtain a more general result, we shall draw the Production Possibility Curve that corresponds to the models on either dataset.

In Figure 3, we show that only models 3, and 4 belong to the PPC (which we draw in bright green). In the same figure, we also draw (in black) the utility curves (UCs) of user X, who is indifferent between completeness and purity and as such his UCs have -1 for slope, and of user Y, who prefers completeness 6 times more than purity (in brown): -6 is the slope of the brown utility function. If we draw parallels of either utility curve until they touch the PPC at the farthest point from the origin, we can determine each user's favourite model: judging from the test set, both users happen to prefer model 3, although user X is almost indifferent between them.

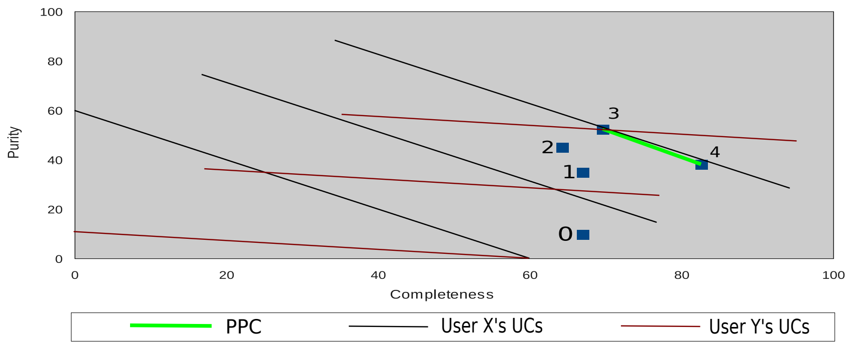

Fig. 3. Choosing the favourite algorithm: Production Possibility Curve drawn from the test set versus user's X and Y's utility curves

Verifying the statistical validity of results In order to test the statistical validity of these results, i.e. to verify whether the observed differences in results are likely to generalise well to unseen cases, we start by applying Friedman's test as corrected by [2]. This and Nemenyi's test require results be measured on three or more independent datasets. In order to obtain these, we measure completeness and purity separately for each different document in our test set, there being 19. There turned out to be significant differences between the algorithms.

Table 2 presents the results of the Nemenyi's test for both completeness and purity in the test set. The theoretical critical value, CD, now equals 1.73.

Model 0 is simpler and computationally less expensive than the remainder, while having as good or better completeness than the remainder. Model 4 has just as good completeness as Model 0, but better purity, as such model 0 is not a good choice, if only completeness and purity matter. Models 1 and 2 are always equivalent and so are models 2 and 3; Model 4 is better than Model 2 purity-wise and equivalent completeness-wise, therefore we can eliminate models 1 and 2 and just keep Model 4. Models 3 and 4 are equivalent. It is important to heve this knowledge to relativise the results obtained in the previous analysis.

From here we can eliminate models 0,1 and 2 as unworthy choices and state models 3 and 4 as likely to have equivalent ability to generalise to unseen cases.

	Completeness					Purity				
Models	0	1	2	3	4	0	1	2	3	4
0	0	🟧 2.9	🟧 2.6	1.5	1.3	0	🟦 -1.9	-1.6	🟦 -2.4	🟦 -3.6
1	🟦 -2.9	0	-0.3	-1.4	-1.6	🟧 1.9	0	0.3	-0.5	-1.7
2	🟦 -2.6	0.3	0	-1.1	-1.3	1.6	-0.3	0	-0.8	🟦 -2.0
3	-1.5	1.4	1.1	0	-0.2	🟧 2.4	0.5	0.8	0	-1.2
4	-1.3	1.6	1.3	0.2	0	🟧 3.6	1.7	🟧 2.0	1.2	0

Table 2. Table of differences between the models' average rankings in completeness and purity; orange signals statistical significance of an improvement in result, while blue signals a reduction.

Specific evaluation 1. Metrics for winner choosing. We will measure the quality of each table's location using the harmonic mean, for a user who likes both equally, of *continuous* completeness and purity, CPF [6]. Naturally, a detected table that holds no real table lines is 0% pure; and a missed table is 0% complete, since none of its elements were detected. By convention, we choose to give 100% to the other metric so as to avoid penalising these errors twice.
2. Winner choosing. We calculate the CPF for each detected table. We then obtain results for each existing table, by averaging over however many tables it got detected into. For each existing table, we then determine which algorithm allows the highest metric.
3. Metrics for characterisation. We proceed to characterise each of the tables in our dataset according to the survey of features presented in detail in 7. In fact, we compiled over 100 features, some developed by us and others used by other authors, to measure the goodness of a table configuration. From other authors we took the following metrics: average number of cells per row and column and its standard deviation[12], the cumulative length, position and content type consistency of the table's cells[12], the degree of alignment of each column[11], the ration between the maximum and minimum length of the columns of a table[8], the average of the proportion of inner spaces per line[4]. We also proposed, among others: the total number of columns that are not separated by a clear river of vertical white space ($NegNum$); and the difference between the average number of cells per column(line), which gives an expected number of lines(columns) in the table, and the actual number of detected lines(columns). This difference, which we call $ErrCol(ErrLin)$, can reflect how sparsely the table is filled in: tables that have a similar number of cells per column(line) will have $ErrCol(ErrLin)$ tend to zero. We calculated these features on the results of both each of models 3 and 4. Some of those features require knowing the number of columns in the table. We estimate this by applying, to the result of the two models, a simple parsing algorithm [7] to find cells within lines; we then consider the number of columns in a table to be equal to the maximum number of cells per line.
4. Analysis. We apply a decision tree classifier to determine, for each existing table, which algorithm would have allowed better results (on the whole existing table). Tables for which one algorithm does not outperform the remainder (i.e.

ties) are excluded from the learning stage. We used SAS Eminer (version 4.3) to run a decision tree that would, from the features, predict whether the table would best be handled by model 3 or 4. We decided to use a decision tree because the model thus produced is not a black box and as such it can allow us not only to identify which of all the features used are more relevant to distinguish good tables from bad ones, but also to understand what makes a good or a bad table.
5. Result 1. When we apply the suggested classifier on the test data, we identify the best model on 31 cases, but at the cost of attributing 62 cases to the worst. We come to the conclusion that it is not possible to divide the set into significant subsets using these input features. Specific evaluation is therefore inconclusive of the relative performance of each algorithm in sub-parts of the space.

Flagging After adding to the result of model 3 some hand-built heuristics designed to accommodate aspects of the table problem that we did not manage to integrate into the probabilistic model [7]. The goal is to understand in which conditions the algorithm is more likely to provide unsatisfactory results.

We decided to run independent classifiers for each typical location error: detected tables can be falsely, impurely, or incompletely detected, or a given existing table can be missed altogether. The decision trees for identifying missed or incomplete tables did not extrapolate well to the test set.

The model for distinguishing pure tables allows us to divide the set of existing tables into two subsets where there are different probabilities of a detected table being impure. The two subsets are determined based on page characteristics: the average number of inner space characters per line of the page, $SpaceAvgPPage$. If this value is above 15, meaning that any columns on the table are likely to be well apart and thus our table location algorithm has little trouble distinguishing it from non-table lines, there is 85% chances that the table is pure. Otherwise, the odds of it being pure are of just 60%. When we look at the proportions of pure and impure tables in these two subparts of the test set, we obtain similar values, which means the rule can be generalised beyond the training set. As such, this information, 60% and 85%, can be used to flag each table with its likelihood of being pure or not, which can be useful to restore coherence if incoherence is detected further down the table analysis process. It is also useful to give the user a perception of the advantages and disavantages of the algorithm.

Finally, we built a decision tree to detect false tables: the row content type consistency (CTC_r^*) of a detected table is above 0.97 (+1 denoting extreme consistency), there is 88% probability that the table is false. This means that if a detected table has elements of mostly one content type, it is likely to be false. We measured this on the test set, and of 22 false tables we manage to identify 15, only misclassifying 2 true tables. As such, we conclude our model is worthy. When we implement this data-induced rule into our datasets, we obtain an increase 0.6pp in purity with a decrease of 0.1pp (0pp) in completeness in the training(test) set. There is no statistical evidence that this result is significantly better than the model before the implementation of this rule, however, even when the improvement only involves only a small part of existing tables in the world, it is nice to have an algorithm that copes with specificites as well as generalities and flagging can give us this with little cost. Our final completeness and purity harmonic mean (CPF) for a user that likes both dimensions equally is now 71.1%/72.2%/64% in the training / validation / test sets and in the, and 75.3%/81.6%/67% for a user who likes completeness 6 times more than purity.

3.2 Experiment 2

This experiment aims at grouping cells into columns. We use the datasets described in the beginning of this section 3; cells are derived from lines with the previously mentioned simple parsing algorithm, as per previous work [7].

We produced two algorithms: a complex one, which we call *grid*, based on inter-word vertical overlap; and a very *simple* one, which just gives each cell a column index equal to the number of cells to its left plus one, in all lines with a maximum number of cells filled in; in remaining lines vertical overlap is used. Apart from simple, this approach means that if a table has all cells filled in and their segmentation is correct, the derivation of columns will be perfect, regardless of any alignment issues.

The composite approach via specific evaluation Our goal in this subsection is to find a way to optimally combine the simple and the grid approach.
1. Metrics for winner choosing. We need to compare the results of the algorithms to each other on a particular table. Our definition of "better" will depend directly on the number of complete and pure columns obtained by each algorithm.
2. Winner choosing. We then create a dataset with one record for every table in our data and we decide whether one algorithm is better than the other or whether they present equivalent results. For 80% of all cases in the training set, there is indifference between one model or another, for 13% the grid approach is better and the 7% the simple approach outperforms. This is our target variable, which we will try to predict using a decision tree. Again, we here use a decision tree because it allows a humanly readable classifier.
3. Metrics for characterisation. We proceed to characterise each of the tables in our dataset according to the features presented in [7] and described briefly in 3.1. We calculated these features (which depend on column configuration) on the results of both the simple approach and the grid approach.
4. Analysis. Again, using SAS Eminer (version 4.3) to run a decision tree that, from the features, predicts whether the table would best be handled by the grid or the simple approach. The training set contains 144 tables, since for the remaining tables there is indifference between using one algorithm or the other. This decision tree is presented in Table 3.
5. Result 1. Both features test the quality of the alignment of the result of the simple approach, in terms of consistency of start, mid and end position of the contents of the cells in each column, CPC_c^*, and in terms of the presence of clear rivers of white space between all but at most one column, $NegNum$. The functions as a test to the quality of its results, a reject option. As such, we can

	Condition	Decision	Probability	# of cases	Leaf
If	the per table average consistency of cell position in each column, calculated using the simple approach, $CPC_c^* < 0.46$	Grid	85.3%	95	1
else if	the total number of negative column separators calculated using the simple approach, $NegNum \geq 1.5$	Grid	65.0%	20	2
else		Simple	100%	29	3

Table 3. Decision tree classifier for choosing the favoured algorithm for each table

apply the simple algorithm to all tables and then, for those which fail the test that the decision tree represents, we can apply the grid approach.

6. Deployment. In Table 4, we present the results of all three methods on the test dataset: there is only one efficient algorithm, since the grid approach is better than the simple approach on both metrics, and so is the composite approach in relation to the other two. Therefore, there is no need for drawing the PPC.

	Simple	Grid	Composite
Complete and pure columns	721	845	859
Completeness	76.8%	83.2%	84.4%
Purity	77.6%	90.1%	90.7%
CPF	77.2%	86.5%	87.4%
Total cost	1,012	536	499

Table 4. Comparative results of the simple, grid and composite approaches

The Friedman test confirms that there are statistically significant differences in the completeness and purity of the different algorithms in the test set. The Nemenyi test further confirms that, on the test set, the composite algorithm's completeness is significantly better than the simple one' while the grid and the composite approaches are deemed equivalent in terms of their ability to extrapolate to unseen data, at 10% significance.

As such, we can conclude that combining the grid and the simple approach is a good idea, as it is better than the simple one and as good as the grid one on the test set, while being computationally simpler.

7. Result 2. As we can see, by using the composite approach in the revised test set, we increase both completeness and purity simultaneously in relation to either contributing approaches applied independently. The magnitude of the improvement depends on the proportion of the different types of tables in a particular dataset. This result serves to make three important observations regarding the problem at hand:

– for some tasks, not one algorithm can treat all cases, some algorithms being better for some tables than others;
– by carefully analysing the a posteriori conditions in which a table analysis algorithm fails or succeeds, we can find a decision rule for deciding, a priori, which algorithm is more likely to yield better results and by doing so we can increase overall performance.
– to conduct a *general evaluation*, i.e. a measure of the average performance of an algorithm over a whole dataset, as most authors do, is a worthy result. However, vital information may be lost if a *specific evaluation*, i.e. if an attempt to partition areas of the space of table possibilities for which each algorithm is more or less adequate, is not conducted.

Flagging In order to flag error prone decisions, we again ran a decision tree to try and identify whether a table contains impure or incomplete columns. We again used SAS Eminer (version 4.3) for this purpose, with the features described in [7]. Again, we decided to use a decision tree because the model thus produced is not a black box and as such it can allow us: a) to identify which features work best, and b) to understand what makes a good table and a bad one, which we find relevant at this early research stage. The tree has low recall (33%) but high precision (84%) in the test set. We present it in Table 5: it isolates five leaves that are likely to be impure.

Description	# of - cases	Prob- ability
Tables with negative white space, meaning there exists no river of white space between two given columns ($NegNum > 0.5$) AND:		
- the page holds more than 53 table candidates ($Silva03abs > 53$)	291	70%
- there is a small amount of positive white space between the columns of the table in relation to its total width ($WhiteProp < 5.5$)	130	80%
- there is little content type consistency in the column ($CTC^*_c < 0.39$) and average contents are narrow ($LenMean < 9$).	210	67%
Tables without negative white space, meaning there are rivers of white space between all pairs of adjacent columns ($NegNum \leq 0.5$), AND irregular number of cells per line ($ColEspStd > 1$) AND:		
- there are plenty of empty lines AND low length consistency per column ($CLC^*_c < 0.04$)	54	85%
- the table has more than 75% numeric contents AND the total variance explained by the relevant main components is low, meaning there is low consistency in length, start, mid and end position of each column's contents ($PropVar < 83\%$).	24	79%

Table 5. Flagging tables with impure columns

Notice that, in spite of our dataset containing several features that were developed by other authors [8, 10, 11, 12], most of the features that were selected by all trees were originally developed by us (the only exception being $ColEspStd$ [12]). Features CTC^*_c and CLC^*_c are inspired on [12]'s, but we have improved them so that a reference point for interpreting them be preserved.

Because in the task of grouping cells into columns there is a statistically significant positive correlation between impurity and completeness (as the number of cells to group is set a priori, thus consituting a null sum game), we shall not build a flagging tree to detect incompleteness.

The knowledge gathered via flagging can be used to find alternative treatment for flagged cases, which we shall not attempt to do in this section, but have in the previous. It can also be used to inform the user of our degree of confidence in each specific part of the outcome and alert him when a problem is likely to exist in this or that particular instance, thus clearly separating likely to be good from likely to be less good results, and providing a clear path for improvement. Finally, it can facilitate automatic error correction at posterior steps in a multi-step system.

4 Conclusions

In this paper, we have presented four techniques that can take the exploration of experimental results farther: adequate statistical testing, production possibility curves versus user utility curves, specific evaluation, and flagging. We have seen them applied, with appropriate variations, through two real experiments. We demonstrate, over the course of the experiments, how they are important for bringing to light the advantages and disadvantages of competing algorithms:

1. the use of the adequate statistical test is fundamental for determining how different algorithms compare to each other on a given set of data;
2. production possibility curves allow clearly depicting each algorithm against user preferences and determining an optimal for each user;
3. flagging allows not only potentially identifying clear ways of improving the current algorithm, as it did in section 3.1, but also measuring the probability of each result being accurate, which can be important information for the users of the data or for subsequent processing steps; and,
4. specific evaluation allows assessing the relative goodness of competing algorithms on their ability to treat different sorts of cases, having the potential to lead to the formation of probabilistic-based combination policies that improve over the individual methods in terms of efficacy, efficiency, or both.

In the future, we intend to apply the four steps of evaluation to other experimental fields.

Bibliography

[1] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[2] L. Iman, Ronald and J. M. Davenport. Approximations of the critical region of the friedman statistic. *Communications in Statistics - Theory and Methods*, 9(6):571–595, 1980.

[3] P. Nemenyi. *Distribution-free multiple comparisons, PHD thesis.* PhD thesis, Princeton University, Princeton, NJ., USA, 1963.

[4] D. Pinto, A. McCallum, X. Wei, and W. B. Croft. Table extraction using conditional random fields. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR Conference on Research and development in information retrieval*, pages 235–242, New York, NY, USA, 2003. ACM.

[5] A. C. e. Silva. Learning rich hidden markov models in document analysis: Table location. In *ICDAR '09: Proceedings of the 10th International Conference on Document Analysis and Recognition*, pages 843–847, Barcelona, Spain, 2009. IEEE Computer Society.

[6] A. C. e. Silva. Metrics for evaluating performance in document analysis - application to tables. *International Journal Document Analysis and Recognition, special issue on performance evaluation of document analysis and recognition algorithms, accepted for publication*, 2010.

[7] A. C. e. Silva. *Parts that add up to a whole: a framework for the analysis of tables.* PhD thesis, Edinburgh University, Edinburgh, UK, 2010.

[8] S. Tupaj, Z. Shi, C. H. Chang, and D. C. H. Chang. Extracting tabular information from text files. Technical report, EECS Department, Tufts University, 1996.

[9] H. R. Varian. *Intermediate Microeconomics: A Modern Approach.* W.W. Norton, 5 edition, 1999.

[10] Y. Wang, R. Haralick, and I. T. Phillips. Improvement of zone content classification by using background analysis. In *4th IAPR International Workshop on Document Analysis Systems (DAS2000)*, pages 10–13, Rio de Janeiro, Brazil, 2000.

[11] Y. Wang, R. Haralick, and I. T. Phillips. Automatic table ground truth generation and a background-analysis-based table structure extraction method. In *ICDAR '01: Proceedings of the 6th International Conference on Document Analysis and Recognition*, page 528, Seattle, Washington, USA, 2001. IEEE Computer Society.

[12] Y. Wang and J. Hu. A machine learning based approach for table detection on the web. In *Proceedings of the 11th International Conference on World Wide Web (WWW02)*, pages 242–250, Honolulu, Hawaii, 2002. ACM.

On the search for and appreciation of unexpected results in data mining research (or: Science - we might be doing it wrong)

Albrecht Zimmermann

KU Leuven, Belgium

Abstract. An integral part of scientific research is the constant search for new results related to existing hypothesis, to either bolster their claims, or falsify them to advance the field. Using the example of four studies that did just that in data mining research, I will argue that the data mining community is neither interested in such studies, nor appreciates their unexpected results. Since it is my opinion that this is an attitude that holds the field back, I propose a change to the conference format that can be expected to motivate researchers to undertake more such studies, and give them higher visibility.

1 Introduction

The majority of our progress in understanding the physical world in the last 150 years, and the technological advances arising from it, can be traced back to the diligent application of the scientific method. Fields that have rejected or misapplied the scientific method, on the other hand, can be seen to stagnate or even regress. An important part of the scientific method is the repeated attempt to falsify hypotheses, i.e. to generate unexpected results, as these results lead to further progress.

As I will argue, data mining (and machine learning) research currently misapplies or even ignores the scientific method to a certain degree. Specifically, I will show that few attempts are made to systematically generate additional results related to existing work, and therefore to try and generate unexpected results. Studies that /empgdo generate such results are often marginalized, and their results ignored.

This article itself is admittedly not the result of a careful meta-analysis but based on personal experiences and impressions, i.e. anecdotes. I will buttress my claims by referring to more objectively accessible measures of the correctness of my claims. Specifically, I will discuss four empirical studies that challenge claims in the data mining literature and show that they were arguably not appreciated upon submission, have been ignored by the community to varying degrees, and have had their lessons ignored. Due to this subjective, and, given my research area and experiences, arguably also somewhat myopic content, I forwent the use of the typically used communal "we" in favor of the less general "I".[1]

2 The scientific method (in the empirical sciences)

In its most simplified manner, the scientific method in the empirical sciences can be summarized in the following way:

1. The researcher has a hypothesis about the world.
2. She/he uses this hypothesis to generate a prediction.
3. He/she performs an experiment testing the prediction:
   (a) The prediction is confirmed: this is considered evidence for the hypothesis to be true, strengthening it.
   (b) The prediction is rejected: this is evidence for the hypothesis to be false, which means it has to be rejected and/or modified.

The reader will notice that this scheme implies that no matter the amount of evidence in its favor, a single (valid) counterexample is enough to falsify and reject a hypothesis – scientific hypotheses can only ever be provisionally true. The true value of negative, or in the parlance of this workshop, unexpected results lies in the rejection of hypotheses, and the resulting need for modification. A hypothesis that has generated a large amount of confirmed predictions, on the other hand, gives anyone employing it high confidence that it is, in fact, true. Unfortunately, a hypothesis that has remained unfalsified for a period of time, even if not evidence in its favor has been collected, can be mistaken for a high-confidence hypothesis as well.

As I have stated above, this is a very simplified summarization of the scientific method. For one thing, empirical experiments often do not give clear-cut *binary* results. Instead, confidence intervals are employed, likelihoods calculated, and statistical tests used to assess significance. Such assessments are more robust if experiments are *independent* from each other, ideally not only independent in time and used data but also performed by *different researchers*. Researchers, being human beings, can have biases or make mistakes and a hypothesis is the more reliable, the more different researchers have confirmed its predictions. Finally, *prior plausibility* can inform the interpretation of experimental results: a result that is barely significant and derives from a hypothesis with low prior plausibility is more likely to be accidental than a similar result stemming from a high plausibility hypothesis.

These aspects imply something that is technically not part of the scientific method: that experiments should be repeated, ideally often, ideally using equivalent but different settings, ideally by different researchers or research groups. It is this aspect of scientific work that is missing in current data mining research as I will argue in this article.

[1] The alternative option of referring to myself in the third person as "the author" felt too pompous.

3 The applicability of the scientific method to computer science

Even though computer science has "science" as part of its name, the scientific method as practiced in the empirical sciences is not simply applied as is. The main reason for this is that computer science is an *applied* science: instead of identifying new facets of how the physical world works, it uses such knowledge to build what in essence are tools for helping humans sense, process, or manipulate the world. A particular microchip is as much a tool as is a complete computer architecture, a theoretical data mining algorithm, or its optimized implementation.[2] As a side-effect of this, there is essentially only a single hypothesis in much of computer science: "it helps solve the problem". If empirical results show that the method does not help solve the problem, the hypothesis, and with it this particular design, is rejected and/or modified. But if a method solves a problem at all, no matter how inefficiently or ineffectively, a design is not rejected but added to the store of knowledge computer science has built. The question then becomes one of usefulness, i.e. how quickly the method finds solutions, and of what quality they are, and of how well assumptions about the data that motivated algorithmic design are borne out by reality. After all, the "no free lunch theorem" reminds us that there is no single method that can be expected to outperform all others over the range of all possible problems.

This is therefore where the scientific methods should find itself applied in data mining: It can be assumed that a researcher has already performed the rejection test for a new method he or she intends to propose and that methods that do not solve the problem at all will therefore not be submitted for publication.[3] Therefore, empirical evaluations should help with establishing

1. how a new method compares with the state of the art, and with similar methods.
2. what the effects of different parameter settings are on the performance of the method.[4]
3. how the data can be characterized that the method performs "well" on and how the data can be characterized on which it does not (with thanks to Eyke Hüllermeier for pointing out this blind spot of mine).

A proper data mining paper introducing a new method (or an improvement to an existing one) would therefore lay out the reasoning behind the development

[2] Since my argument here focuses on data mining, I will stop writing about tools now and in the rest of this article refer to "methods", encompassing data mining algorithms, feature selection approaches, distance measures etc.

[3] And it can be argued that this is problematic on its own: if a method with high plausibility fails to solve the problem, this is important knowledge. This argument would far exceed the scope of the article, though.

[4] This is particularly important given the "standard settings" used in toolkits such as WEKA [1] that are the main resources of many researchers that aim to employ existing methods

of the method, establishing plausibility, describe the method itself and then perform an extensive empirical evaluation. As part of this evaluation, the researcher would

- select (or generate) data covering a wide range of different data characteristics,
- identify methods most closely related (which should be easy given the plausibility analysis) as well as a number of state-of-the-art techniques that have been shown to perform well on this problem in past work,
- perform the experiments exploring a wide range of parameter settings while making sure to choose well-performing settings for the comparison techniques,
- and evaluate the results using statistical techniques to establish significance, breaking the data up into subcategories on which the method's behavior is different, i.e. where unexpected results occur.

Most of the papers I have been assigned as a reviewer fail at covering some or all these aspects. It is of course possible that those works are the unfortunate exception but if some papers that have been published in conference proceedings in past years are an indication, they are not.

Personal anecdote 1 *This has in fact become my primary rejection criterion: whether the work presents an adequate empirical evaluation of its method (where appropriate), and so far I have identified three main violations:*

1. *flaws in evaluation design: for instance by claiming to compare against the start-of-the-art but ignoring the work of the last several years, designing a worst-case strawman algorithm as the only comparison technique, or limiting the evaluation to a single data set, maybe one that is well-known to be abnormal to boot.*
2. *flaws in evaluation reporting: for instance by showing averaged accuracies without including standard deviations and/or discussing statistical significance.*
3. *overselling: for instance showing results that place the proposed method roughly equal to comparison techniques on half of the data and non-significantly better on the other half and claiming that it "significantly outperforms" the comparison techniques.*

The distressing part for me is that sometimes the argument for plausibility has been convincing enough that I would be willing to accept a paper with such a flawed evaluation if *I trusted the rest of the community to perform additional evaluations and paint a fuller picture.*

Even if a new method (or an improvement of an existing one) has been evaluated properly in the work that proposed it, it will still be desirable and necessary that it were reevaluated by other researchers, using newly surfaced data, but also using data on which it has been evaluated before, using different implementations, different experimental setups, e.g. a different number of folds, such as has been done in the Frequent Itemset Mining Implementation competitions [2, 3]. For such studies and the possibly unexpected results they generate to have a positive impact on the community, it is necessary to give them a central spot in the literature. As I'll argue in the next section, it is in this that data mining research fails worst.

4 A number of studies showing unexpected results and their reception

In the following, I will present and briefly discuss several papers that produced unexpected results, showing established provisional truths in data mining research to be false. I will follow this up by discussing how they have been received by the community and what their impact has been as can be inferred from the literature. I want to reiterate that given the breadth of the field this is necessarily a subjective collection and subjective interpretation. I will attempt to support my claims with more objective measures, however.

Real world performance of association rule algorithms.[4] The arguably seminal itemset mining paper [5] also introduced a data generator for evaluating the running times of itemset mining algorithms. Zheng *et al.* in 2001 showed that the data generated in this manner showed different characteristics from real-life data and that the run time behavior of several algorithms [5–9] differed between the artificial and real-life data.

Implied, even though not contained, in that paper are two additional observations: First, the authors of [5] had varied the parameters of their generator, albeit in a restricted manner, to evaluate their approach on more than twenty data sets. Subsequent work in the field, however, used fewer and fewer data sets. In fact, looking at the itemset mining literature, I find an inverse correlation between the age of the paper and the number of data sets used. Second, Zheng *et al.* showed that Charm significantly outperformed Closet on the real-life data, a finding that contradicted the results reported in [8], in which different data had been used. As Zaki then showed in [6], Charm also outperformed Closet on the data used [8] if one lowered minimum support further than had been done in that work. The differences in performance can therefore be tied both to data and parameter settings.

Using Classification to Evaluate the Output of Confidence-Based Association Rule Mining.[10] Association-based classification had first been proposed in [11]. Mutter *et al.* showed in their work that CBA, the algorithm proposed in [11], did not perform better than existing rule-based classifiers, putting the results of that work into perspective and partially contradicting them. Their work furthermore showed that replacing Apriori by Predictive Apriori proposed by Scheffer [12] lead to better classifiers.

As a side note, the authors write in their paper that they were not able to reproduce all the results from [11] with a reimplementation of their own, and report in the thesis that the paper was based on that they also could not achieve this with an implementation provided by the authors of [11].

Personal anecdote 2 *While working on a past paper, we planned to compare to a technique proposed by another data mining researcher. We obtained both the original data and the implementation from the author but did not manage to reproduce some results. After mailing him about the issue, we received a reply along the lines that he did not understand our problem since he had been able to reproduce the results. The result files were attached. The solution to the problem was that while the results from the paper could be obtained, they could not be obtained using the parameter settings from which the paper claimed they originated – apparently the parameter entries had been switched around on writing the paper.*

Obtaining Best Parameter Values for Accurate Classification.[13] Association based classification usually works with standard values with 1% for minimum support and 50% for minimum confidence. As Coenen *et al.* showed experimentally, these values are not only not the best values for achieving high accuracy but especially CBA often tended to perform worst for these settings.

Frequent Subgraph Miners: Runtimes don't Say Everything.[14] Improving running times is a major goal in the development of new pattern mining algorithms. As Nijssen *et al.* showed, many of the claimed underlying reasons for improved run times did not hold up under scrutiny. Additionally, they found that the interplay between cache size of the processor used and the size of the data set, i.e. aspects largely outside of researchers' control, had a strong effect.

4.1 Reception in the community

Given the works I have just listed, one could be tempted to assume that all is well in data mining research since such works are being undertaken and papers based on them published. The reader could also be forgiven for pointing towards the large scale evaluations the group of Johannes Fürnkranz has undertaken over the years, for instance, on the effects of coverage and consistency in rule-learning heuristic [15], *beam sizes* [16], probability estimation techniques [17] etc and absolving machine learning research.

The problem, however, is not so much with whether these works are done in the first place – even though a cursory glance through a year's data mining conferences will show far more papers proposing new methods than ones further evaluating existing ones. In my opinion, the problem lies instead with the effect (or lack thereof) such studies have on the field. It is difficult to evaluate this effect directly which is why I will use three proxies instead:

1. the venue they were published in, and whether they were full papers.
2. the number of citations (related to the number of citations the papers proposing the evaluated methods received).
3. the effect as can be inferred from the literature.

Venues and paper type Zheng *et al.* [4]: short paper at KDD 2001. Mutter *et al.* [10]: rejected at ECML/PKDD, regular paper at the Australian Joint Conference on Artificial Intelligence 2004, which is not highly ranked according to different rankings. Coenen *et al.* [13]: short paper at ICDM 2005, which did not have short paper submissions. Nijssen *et al.* [14]: workshop paper, although Siegfried Nijssen claims that he simply never followed up on this work.[5]

The situation for the Fürnkranz publications is similar: [18] short paper ICDM 2007, [15] Discovery Science 2008, not a highly ranked conference, [16] poster SDM 2009, [17] Discovery Science.

These studies should have been front-page news, for challenging provisional truths and showing that accepted default values and default experimental setups were faulty. Instead there were shunted aside in favor of yet another weakly understood new method. To be clear: there is nothing wrong with publishing at conferences that are not highly ranked. It just indicates that the community is not interested enough in these works to see them published in full at the conferences it values highly.[6] This makes it less likely that these works are noticed by the wider community and the lessons learned incorporated. This has not only to do with those works' appearance in the conferences' proceedings but at least as much with having the opportunity of giving a talk to a large and interested audience and impressing the importance of the issue on them. KDD 2011 had more than 1000 attendants. ICDM 2011, while still being a high-ranked conference, had in the range of 400-600. Conferences like Discovery Science are more likely to attract in the vicinity of 200-300.

Even if such a study *is* included in a high-ranking conference, the nature of such evaluations is that quite a few different experimental settings are used and lots of numbers produced. If a write-up is then forced to do with a reduced page limit, it will be difficult to present the studies and their results comprehensively.

Citation count I used Google Scholar on August 5th, 2012, with all that this entails, and compare each paper's citation count with that of the methods evaluated. If the insights derived from these studies found application in data mining research and default values or experimental setups were adjusted accordingly, I would expect a citation count for these studies that at least comes close to those of the methods they correct.

Zheng *et al.*: 352 citations, compared to 13020, 990, 4178, 779, 149 for the evaluated methods (in order of citation above).

Mutter *et al.*: 17 citations, 3 by papers of which I am a co-author, 2 by papers of which Johannes Fürnkranz is a co-author, Coenen *et al.*: 26 citations, 7 self, 2 by paper of which I am a co-author, compared to 1582 (CBA), and 920 (CMAR) citations.

Nijssen *et al.*: 16 citations, compared to 1035 (gSpan), 346 (FFSM), 67 (AcGM), 779 (FSG).

[5] personal communication

[6] Why these conferences are highly ranked is another question to do with publishing arcana.

The Fürnkranz papers: [18] 14 citations, 7 self, [15] 9 citations, 6 self, [16] 7 citations, 5 self, 2 by papers of which I was co-author, [17] 3 citations, 2 self.

Impact on the field Zheng *et al.* certainly had an impact since the data sets they introduced have become standard benchmark sets and the data generator proposed in [5] has fallen into disregard. However, the deeper lesson, that focusing on a small number of data sets in proposing improvements can lead to overfitting effects, has clearly been lost on the community given the small number of data sets used to evaluate itemset mining approaches, the lack of a widely-used replacement generator, and the few large-scale comparisons [2, 3].

Given the results of Mutter *et al.* and Coenen *et al.*, the experimental evaluations of associative classification research since 2005 should have featured the use of Predictive Apriori and at least an exploration of a range of support-confidence combinations. A survey of the relevant literature since then will disabuse the reader of this notion.

The insights derived in Nijssen *et al.*, while formulated in the context of graph mining approaches, should be heeded in all pattern mining research but claims of speed-ups having to do with canonical forms still abound.

4.2 Is this really a problem?

The reader can now (and maybe already has) argue that this is not a real problem. After all, three of the studies I discussed are concerned with itemset mining, a field that has been researched for a while and in which no groundbreaking discoveries can be expected anymore. I would have to disagree since there is no reason to assume that the scientific method is applied more diligently in other fields as the example of Nijssens *et al.* shows. Even if this were the case, data mining research, as I have argued in the beginning, is essentially concerned with building tools, such as itemset mining techniques. If these tools are never used outside of academia because users do not know under which conditions they might be useful, data mining research misses its purpose.

The reader might claim that data mining (or the subfields of data mining I am most familiar with) is an aberration and the scientific method is employed much more stringently in other areas. However, the examples of the works from the group of Johannes Fürnkranz lead me to believe that those problems exist in the machine learning community as well. But even if this is not the case, it seems even more urgent to change the practice in data mining research lest the area stagnate.

Finally, the reader could argue that the methods that truly make an impact will be evaluated, and their working parameters established, in successive studies as they are used more and more often. This has, for instance, happened to decision tree or support vector machine classifiers. Non-performing methods get drowned by the tide of papers published every year and so no further evaluation is necessary.

There are at least three problems with this idea: the first is that in the absence of principled studies we do not know that the methods that rise to the top are truly the best (or even reasonably good) ones. They might instead be the early ones that did not have to compete with many other papers yet, they might be the simplest ones that everyone understands, or they might originate from a large research group whose cross-citations help them gain critical mass. The second problem is, as can be seen in itemset mining, that those principled evaluations might simply never happen, even for well-known techniques. Third, the proliferation of methods is intrinsic to the issue I am discussing here: the relative disregard shown to works searching for unexpected results makes it appear more attractive to propose new methods (with "good" results) instead, and the lack of scientific evaluations translates into a lack of guidance regarding the methods that should be improved. This also means that lots of time, money, and energy are wasted on dead end research that does not improve the field.

Furthermore, the issues I have described so far are connected to relatively clear-cut evaluation criteria: running times and classification accuracy. In descriptive data mining, which includes most of pattern mining, such clear-cut evaluation criteria are not available. Instead, the problem to be solved there consists of extracting underlying patterns of correlations in the data.

Remarkably enough, in at least two subfields, itemset mining and frequent episode mining, the literature so far does not include any evaluations on whether extracted patterns correspond to known phenomena in the data. Instead, attempts have been made to evaluate the quality of found patterns by presenting them to domain experts who were supposed to perform this evaluation [19]. An implicit assumption in these kinds of evaluations is that the experts will be able to properly identify interesting patterns as interesting and uninteresting ones as uninteresting. The plausibility of this assumption is however unknown and if psychological research on humans' tendency to see patterns is any guide, it might be much lower than the authors of such studies assume.

An experiment that I would like to see would consist of calibrating the domain experts first: instead of having them evaluate significant patterns, they would be given a mix of significant, non-significant, and random patterns and their "accuracy" evaluated.

Personal anecdote 3 *Last year, I had what I considered to be a "small" idea for a conference paper. It had to do with feature generation, had good plausibility, discussions with colleagues revealed no obvious flaws, and the literature study showed that it had not been tried before. So I went for it - and the results were atrocious. Assuming a mistake on my side, I checked everything several times but could not find one. Finally, I gave up and tried to get this negative result published, as a warning to others. The submission was rejected with one reviewer writing that the results did not surprise him (and the two others that they did not see the use of publishing a negative result until everything had been tried to make the approach work). The reviewer admitted that she/he could not provide a reference but provided a convincing rationalization.*

Long story short, when Joaquin Vanschoren asked me to perform some further analysis on my results for the workshop, I revisited the old scripts - and found the bug that had escaped me. With the bug corrected, the approach works as I *expected. Unless the reviewer knows more about my code-writing prowess than I do (and just wanted to spare my feelings), his/her lack of surprise is surprising.*

If I had managed to get these false results published, I would not have revisited them, and since the results were negative, in all likelihood no one else in the community would have.

5 What we can do to get back on track

In my opinion, there are at least two causes that can be identified underlying the attitude of the data mining and machine learning community towards generating, reporting and using unexpected results, one that is for now outside of our power to effectively address, but also one that could be addressed by small changes to the current conference format.

First, it is my impression that the scientific method, and specifically the need for constant retesting, and the importance and usefulness of unexpected results, is not being impressed on young researchers. This has ripple effects, leading to the aforementioned badly designed (and reported) experimental results, an unwillingness to report unexpected results, and a tendency to negatively review papers that do report new results of existing techniques or unexpected (negative) results. There is not much we, as individuals, can do to change this: we can try and influence students and colleagues, we can criticize weak experimental evaluations in reviews and point out avenues for improvement, we can support studies searching for unexpected results. But all of these can only be expected to be drops in the ocean given the amount of new researchers and new publications each year.

Second, purely experimental studies that do not propose a new method do not have a "home" in the community's conferences. The typical conference format in 2012 was to have a "research" track and either no (SDM, CIKM, ICML, ECAI, DS), or only one (KDD, ICDM, CIKM, ICDE) additional track (e.g. industry and government) for submission (ECML/PKDD being the exception with two extra tracks). The calls for papers make it explicit that this is intended in listing the kind of papers that are solicited and papers are supposed to be subdivided by using keywords during submission, often ones focused on problem areas. This means that at least four different types of papers, all of which are subject to the same page restrictions, compete for acceptance:

1. Papers proposing new methods: these works have to establish plausibility, outline the method, and ideally show an extensive experimental evaluation. "Good" results on at least some data will be helpful in establishing the usefulness of the new method.
2. Papers proposing improvements to existing methods: since the main method has already been introduced, these works only have to establish plausibility of (and describe) the proposed improvement, and the experimental evaluation can focus on showing the effects of the proposal (but should still do so thoroughly). These effects arguably need to be positive for it to be an improvement.

3. Experimental evaluations of existing methods: these works need to establish the appropriateness and/or difference of their data and experimental settings for producing new results, and need to present and analyze their experimental results in detail.
4. Theoretical works: are something rather different.

The "success" criteria are thus different for all four types, with some easier to evaluate than others, yet there is typically a one-size-fits-none reviewing framework within which they are supposed to be evaluated.

The reader might think that journals, with their larger page count per paper and specifically selected reviewers, can be an alternative to conferences but those also have an attention problem: an extended version of a paper introducing a new method will necessarily appear later than its conference version so that a researcher who has read the conference version might not go to the effort of reading the journal version as well. Instead of a program that every conference participant is handed, the table of contents of a journal must be actively sought out by researchers, and a journal paper is not accompanied by a presentation. Finally, that a journal paper is longer can paradoxically work to its disadvantage since a researcher seeking a quick understanding of the method would likely prefer the shorter conference version.[7]

On the other hand, given the current use of keywords to subdivide submissions by areas but also to a certain degree by content, e.g. foundations of data mining, it should easily be possible to invite submissions to several different tracks. This can be expected to motivate researchers on the fence about attempting (and attempting to publish) works generating additional results for existing methods, and it should allow better-targeted instructions to the reviewers.

Outside of the publication context, there already exist projects that collect experimental results for different data sets, algorithms, and parameter settings, such as ExpDB[8] and MLComp[9] to enable comparison and the selection of appropriate methods. These projects cannot be an alternative to a rethinking of the scientific process in data mining research and supply a clean conscience, however. Instead, they offer support for comparison and their success *depends* on a change of mind among data mining researchers.

6 Conclusion

In this article, I have discussed the application of the scientific method in data mining research, specifically the search for and appreciation of unexpected results, and found the field lacking in this regard. A study of a typical year's literature reveals a plethora of new and improved methods and few systematic evaluations of these methods. If such evaluations are done at all, they often show

[7] Additionally, journals also have the competition issue, albeit to a lesser degree.
[8] http://expdb.cs.kuleuven.be/expdb/
[9] http://mlcomp.org/

well-established provisional truths to be wrong, as I have argued using the example of four empirical studies. However, since they tend to get marginalized during publication, the derived insights do not find their expression in future research, and those provisional truths (while shown to be false) continue to be accepted.

In my opinion, this self-perpetuating cycle can be attacked by acknowledging that there is more to data mining research than just the proposal of yet another method, and by changing the conference format to motivate researchers to undertake the work necessary for understanding the strengths and limitations of existing data mining methods. The goal of data mining research lies in providing tools to users. This does not mean, however, that there is any merit in cranking out ever more methods without learning under what conditions those methods work well. The current pool of often-used methods is limited to the few that are, if not well-understood, at least well-known and unless we change the way data mining research is conducted, it will stay like this.

Acknowledgements

The motivation for writing this article and quite a few of the arguments laid out stem mainly from discussions with numerous participants of the FLF workshops, too many to mention them all here. I would like to single out Matthijs van Leeuwen for giving important feedback on a first draft on short notice, and Tias Guns who, even though he did not any have time for this, offered up several of the counter arguments.

References

1. Frank, E., Witten, I.H.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann (1999)
2. Goethals, B., Zaki, M.J., eds.: FIMI '03, Frequent Itemset Mining Implementations, Proceedings of the ICDM 2003 Workshop on Frequent Itemset Mining Implementations, 19 December 2003, Melbourne, Florida, USA. In Goethals, B., Zaki, M.J., eds.: FIMI. Volume 90 of CEUR Workshop Proceedings., CEUR-WS.org (2003)
3. Bayardo Jr., R.J., Goethals, B., Zaki, M.J., eds.: FIMI '04, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, Brighton, UK, November 1, 2004. In Bayardo Jr., R.J., Goethals, B., Zaki, M.J., eds.: Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations. (2004)
4. Zheng, Z., Kohavi, R., Mason, L.: Real world performance of association rule algorithms. In: KDD. (2001) 401–406
5. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Databases, Santiago de Chile, Chile, Morgan Kaufmann (September 1994) 487–499
6. Zaki, M.J., Hsiao, C.J.: Charm: An efficient algorithm for closed itemset mining. In Grossman, R.L., Han, J., Kumar, V., Mannila, H., Motwani, R., eds.: SDM, SIAM (2002)

7. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas,Texas,USA, ACM (May 2000) 1–12
8. Pei, J., Han, J., Mao, R.: Closet: An efficient algorithm for mining frequent closed itemsets. In: ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. (2000) 21–30
9. Webb, G.I.: Efficient search for association rules. In: KDD. (2000) 99–107
10. Mutter, S., Hall, M., Frank, E.: Using classification to evaluate the output of confidence-based association rule mining. In Webb, G.I., Yu, X., eds.: Proceedings of the 17th Australian Joint Conference on Artificial Intelligence, Cairns, Australia, Springer (December 2004) 538–549
11. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In Agrawal, R., Stolorz, P.E., Piatetsky-Shapiro, G., eds.: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, New York City, New York, USA, AAAI Press (August 1998) 80–86
12. Scheffer, T.: Finding association rules that trade support optimally against confidence. In De Raedt, L., Siebes, A., eds.: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery, Freiburg,Germany, Springer (September 2001) 424–435
13. Coenen, F., Leng, P.: Obtaining best parameter values for accurate classification. In Han, J., Wah, B.W., Raghavan, V., Wu, X., Rastogi, R., eds.: Proceedings of the Fifth IEEE International Conference on Data Mining, Houston, Texas, USA, IEEE (November 2005) 597–600
14. Nijssen, S., Kok, J.: Frequent subgraph miners: runtimes don't say everything. In Gärtner, T., Garriga, G., Meinl, T., eds.: Proceedings of the Workshop on Mining and Learning with Graphs,. (2006) 173–180
15. Janssen, F., Fürnkranz, J.: An empirical investigation of the trade-off between consistency and coverage in rule learning heuristics. In Boulicaut, J.F., Berthold, M.R., Horváth, T., eds.: Discovery Science. Volume 5255 of Lecture Notes in Computer Science., Springer (2008) 40–51
16. Janssen, F., Fürnkranz, J.: A re-evaluation of the over-searching phenomenon in inductive rule learning. In: SDM, SIAM (2009) 329–340
17. Sulzmann, J.N., Fürnkranz, J.: An empirical comparison of probability estimation techniques for probabilistic rules. In Gama, J., Costa, V.S., Jorge, A.M., Brazdil, P., eds.: Discovery Science. Volume 5808 of Lecture Notes in Computer Science., Springer (2009) 317–331
18. Janssen, F., Fürnkranz, J.: On meta-learning rule learning heuristics. In Ramakrishnan, N., Zaiane, O., eds.: ICDM, IEEE Computer Society (2007) 529–534
19. Ohsaki, M., Kitaguchi, S., Okamoto, K., Yokoi, H., Yamaguchi, T.: Evaluation of rule interestingness measures with a clinical dataset on hepatitis. In Boulicaut, J.F., Esposito, F., Giannotti, F., Pedreschi, D., eds.: PKDD. Volume 3202 of Lecture Notes in Computer Science., Springer (2004) 362–373