Automatization of the Stream Mining Process

Lovro Šubelj, Zoran Bosnić, Matjaž Kukar, and Marko Bajec

University of Ljubljana, Faculty of Computer and Information Science,
SI-1001 Ljubljana, Slovenia
{lovro.subelj,zoran.bosnic,matjaz.kukar,marko.bajec}@fri.uni-lj.si

Abstract. The problem this paper addresses is related to *Data Stream Mining* and its automatization within *Information Systems*. Our aim is to show that the expertise which is usually provided by data and data mining experts and is crucial for problems of this kind can be successfully captured and computerized. To this end we observed data mining experts at work and in discussion with them coded their knowledge in a form of an expert system. The evaluation over four different datasets confirms the automatization of the stream mining process is possible and can produce results comparable to those achieved by data mining experts.

Keywords: data mining, stream mining, expert system.

1 Introduction

With emergence of pervasive distributed computing environments, such as cell phones and sensor networks, the data production has enormously increased [8]. In such environments, data are frequently seen as continuous infinite streams, which cannot be stored for later use. The storage and processing characteristics of streams do not allow for the application of conventional techniques for data analysis and mining; instead, specialized approaches are required that are capable of timely analysis and efficient memory usage. Data stream mining received a lot of attention among researchers and a lot of aspects have already been investigated and resolved [2,13].

The approaches that are available today offer a reasonable trade-off between predictive accuracy and timely responsiveness and can be efficiently used to handle various practical situations [8]. However, data mining requires a deep understanding of the problem domain and data (provided by domain experts) and processing techniques (algorithms, their usage and optimization) that can be employed to perform the analysis (provided by data mining experts). This necessity to employ data and domain experts is recognized as an important obstacle which limits the usage of the data stream mining approaches in practice [9].

In this paper we focus on automatization of the data stream mining process. We do this by capturing and storing the knowledge that the experts employ in various steps of the stream mining process. The main steps of our approach include: (1) the acquisition of the client's requirements and main data properties, (2) the selection of methods that best match the given problem, (3) the setting

M. Jarke et al. (Eds.): CAiSE 2014, LNCS 8484, pp. 409–423, 2014.

and trimming methods' parameters, and (4) learning from the method application on the given problem. We carefully analyzed and discussed the above steps over four case studies with two selected experts. As a result we developed an expert system that fully automatizes the stream mining process. We show that the stream mining of a reasonable quality can be performed using only a minimal domain knowledge and without involvement of data mining experts. This is an important finding which reveals that the stream mining can become more widely used within information systems dealing with streams of data.

The paper is structured as follows: Section 2 briefly explains the related work, Section 3 the expert system, and Section 4 the evaluation. The conclusion is given in Section 5.

2 Related Work

We relate our work to the two relevant sub-fields of machine learning: (1) the field of incremental learning, which deals with the complexity of this task, and (2) the field of meta-learning, which focuses on automatically relating algorithm performance to the characteristics of the data and prior domain knowledge. In the following we review the important works in the both fields.

2.1 Data Stream Processing and Its Complexity

A data stream consists of an ordered sequence of examples, which arrive online. Examples of such applications include sensor measurements, financial applications, telecommunication and network transactions, and others. To achieve processing in real time, the examples are read only once, processed and then discarded or archived. If the example is archived, it has to be stored in memory, which is relatively small compared with the potentially unbounded size of the whole data stream. Several works [2,8] discuss characteristics of data streams and emphasize the following:

- stream mining is performed by sliding window techniques which maintain only the most recent examples in the stream;
- batch processing approaches are inadequate due to the fast processing requirements and the inability to store all past data. They have to be replaced with incremental approaches;
- adaptivity to changes in data is important due to potentially changing data distributions. Since the sequence of examples is not independent and the examples are generated by non-stationary distributions, the target concept may gradually change over time (concept drift) [5];
- summarization, sampling and synopsis techniques are required to compress data and store their statistics; and
- queries over streams cannot be evaluated precisely and are approximated.

Different research directions stem from the listed set of challenges, e.g., proposing learning algorithms for supervised and unsupervised learning [4], improving

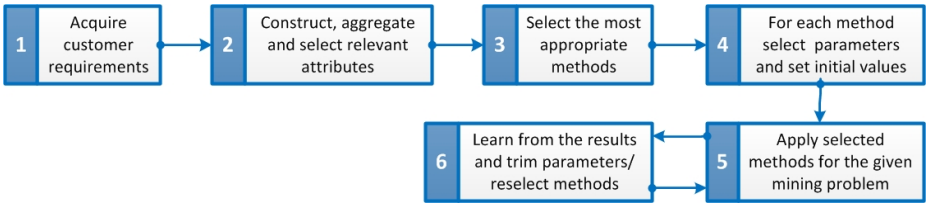


Fig. 1. Main activities in the stream mining process

their accuracy [13], performing queries over transient examples in a stream [2], sampling over data streams, dealing with concept drift [8], and others. The diversity of the former challenges illustrates the complexity of the decisions that need to be taken by users.

2.2 Meta-modeling of Algorithm Performance

Meta-learning focuses on modeling relationship between characteristics of a problem domain and the learning algorithm performance [6]. To predict the performance, the meta-learning algorithm records the past empirical performance of different learning algorithms along with the attributes that describe the problem domain. Choosing these meta-attributes appropriately is a challenge in this field; they can be based either on the data parameters (e.g., data set size, number of attributes, class distributions etc.) or the parameters of the particular underlying learning algorithm [1]. An alternative to automatic construction of meta-learning knowledge is to construct them manually.

Related meta-learning based approaches include stacked generalization [16], which is considered a form of meta-learning because the transformation of the training set conveys information about the predictions of the base-learners; selecting a learning algorithm for each individual test example based on the algorithm's performance exhibited in the example's neighborhood [11]; and inductive transfer of learned knowledge across domains or tasks [12].

Both mentioned fields of the related work motivate us to develop an experimental automated stream mining system that addresses the challenges of the incremental learning and uses meta-modeling to facilitate automation of parameter-setting tasks. Our wishful goal is to enable the non-data mining experts to use the proposed system and achieve comparable performance to the performance of algorithms used by field experts.

3 Expert System

3.1 Stream Mining Process

As emphasized in the introduction, the goal of our research was (a) to capture the expert knowledge of stream mining experts and (b) to formalize this knowledge within an expert system for the automatization of the stream mining process.

In a typical stream mining scenario, there are two kinds of experts involved, both having important roles: data mining experts and domain experts. Whereas the former provide expertise on stream mining techniques, the latter help the stream mining experts to faster identify crucial data properties which would otherwise remain hidden. It is important to note here that in our work we assumed that only a minimal knowledge on the domain is available; by avoiding requirement for domain expertise we therefore aimed to make the system as general as possible. Based on the above limitation and discussion with two stream mining experts we constructed a simplified stream mining process, shown in Figure 1 (explanation in the following). In this process we identified three main areas where the expert knowledge is most important, which are:

- the construction, aggregation and selection of relevant attributes (Figure 1, Activity 2);
- the selection of most appropriate stream mining methods based on problem and data description (Figure 1, Activity 3); and
- setting and re-setting of stream mining methods' parameters (Figure 1, Activities 4 and 6).

In the following sections we first explain the minimal domain knowledge that is required in order to mine streams (with or without experts). Then we describe the activities within the stream mining process where the mining expertise is the most beneficial and explain how these activities were implemented within our expert system.

3.2 Minimal Required Knowledge

For an expert or an expert system some minimal required prior knowledge (RK) is beneficial to make reasonably informed decisions and recommendations:

RK1. *Client's subjective preferences*: requirements such as transparency, visualization possibilities, ability to explain and evaluate results in terms of confidence or reliability. Fulfilling these requirements dictates the choice of stream mining methods;

RK2. *Client's objective preferences*: requirements for data stream predictors. *How will performance be measured? What is a required response time? Should responses be available at any time, even if they are not perfect? What are the time spans of interest? What is the frequency of predictions?* For example, the client might request that every day at 6:00 AM we produce predictions of certain parameters for the next 6, 12 and 24 hours;

RK3. *Known data stream properties*: number of attributes, attribute types, attribute values, distribution of values, frequency of data generation, sparse (e.g., unstructured text) or dense (e.g., sensor) data; and

RK4. *Known data stream mining problem properties*: also based on client's preferences and determines the necessary stream mining paradigm, i.e., prediction of parameter values (both discrete-classification and continuous-regression), anomaly detection, detection of recurrent and/or irregular

patterns, and concept drift detection (fundamental changes in data stream generation process).

Initial recommendations of the stream mining experts are produced by consideration of RK1–RK4 without looking into the data stream, which is assumed not to be available at this point. This activity (requirements acquisition) is not yet implemented in our expert system and thus has to be performed manualy in discussion with the client. In future, we intend to develop wizard-based interfaces that will allow clients to provide requirements and domain knowledge directly to the system, i.e., without any involvement of the experts.

3.3 Choosing Appropriate Stream Mining Methods

After acquiring the main client preferences (RK1 and RK2) and data properties (RK3), a stream mining expert selects a set of methods that best match the given problem. This is typically done by considering the importance of different methods' characteristics for the client. The following characteristics are considered as the most important and included in the discussion:

- *Transparency* reflects experts' opinion on how readable and transparent will be the generated model for end-users;
- *Visualization* describes possibilities for visual representation of the model;
- *Explanation* refers to model's abilities to automatically explain its predictions and allow users to assess them;
- *Reliability* describes model's abilities to automatically estimate reliability of its predictions, as well as to allow users to assess them;
- *Response* is an estimate of model's expected response time, both in terms of model updates (training) and model predictions. Normally, shorter response times go hand in hand with lower performance; and
- *Performance* is an estimate of model's performance (in terms of established performance measures, such as mean squared error, classification accuracy, κ statistic etc.).

In our expert system, we first filter the available stream mining methods and select only a subset that best matches client's requirements and domain description. Later, during stream mining we on-line evaluate the selected methods on real data and further refine their selection, if necessary. The knowledge that we acquired from the experts and used to filter out the methods that best match the client's requirements, is summarized in the Table 1 (for classification methods only, due to lack of space).

Our system supports 12 classifiers (methods for predicting a discrete class attribute), 9 regressors (methods for predicting a continuous class attribute) and 5 clustering algorithms from WEKA, MOA and IBLStreams toolkits [7,3,15]. For classification we use: BAYES: a simple Naive Bayesian classifier; RULES: decision rules with Naive Bayes classifiers; TREE: a Hoeffding tree with information gain split criterion; ENSMB: a weighted ensemble with Hoeffding trees; BOOST: AdaBoost boosting approach based; KNN: simple nearest neighbors; and META: a

Table 1. Descriptive properties of data stream mining methods for classification methods. The symbols in the table denote: high importance (+), medium importance (○), low importance (−).

	Transparency	Visualization	Explanation	Reliability	Response	Performance
BAYES	○	○	+	+	+	+
TREE	+	+	+	+	○	○
KNN	○	○	○	○	−	○
RULES	○	○	+	○	○	○
ENSMB	−	−	○	○	−	+
BOOST	−	−	○	○	−	+

meta approach with all above based on κ statistic. For regression we use: KNN: nearest neighbors with linear regression; RULES: adaptive model rules regression; TREE: a Hoeffding regression tree with options; ADDIT: stochastic gradient boosting; DISCT: discretization based approach; SVM: support vector machines based; and META: meta approach with all above based on *MSE*.

3.4 Setting and Trimming Stream Mining Methods' Parameters

In a stream mining experiment the stream mining experts start by setting some reasonable parameter values. When data arrives, the experts observe methods' performance and experiment with their parameters. The choice and magnitude of parameter changes is based on experts' experience and intuition and is difficult to formalize.

For our initial experiments, the experts provided parameter values that were expected to provide reasonable performance of the selected methods. Some parameters (e.g., initial window size) were set according to expected frequency of data generation (RK3), required timespans and frequency of predictions (RK3), thus, to include all possible natural and human cycles. For each of the chosen data stream mining methods, experts also identified a small number of parameters that were sensible to further tune to increase methods' performance.

For the automatization of this activity (Figure 1, Activities 4–6), we initially set the parameters to their default values and then tune them based on the methods' performance. While the first step is rather trivial (the default values are typically suggested by the methods' authors), the second one is much more complicated, as it requires a learning system. We implemented such a system that executes data mining methods in a batch and tunes their parameters on recent subsets of data in order to optimize performance indicators (κ statistic or mean squared error). When the improvement of tuned performance compared with the online performance is statistically significant, the parameter values are applied also within the production (online) stream mining methods. In contrast to the manual parameter setting, where stream mining experts work only with a subset of parameters, the expert system deals with all the parameters in parallel. In addition, if some of the initially selected methods significantly decline in their expected performance, they are terminated and possibly replaced with others.

Please note that the herein described online parameter tuning feedback loop is still a work in progress, and its results are not included in this paper.

3.5 Construction, Aggregation and Selection of Relevant Attributes

Data preprocessing, cleaning and filtering, attribute construction, aggregation, and subset selection, are very important for successful data mining. Many methods are prone to perform poorly when using irrelevant and noisy attributes. While some data mining methods implement constructive induction of attributes [10], they are very complex and time consuming and therefore inappropriate for stream mining. Data expert knowledge (if available) can be used to construct more relevant general attributes; however, for data streams, temporal aggregations are relatively straightforward.

In the observed experiments the stream mining experts started with the application of some basic data preprocessing techniques, such as imputation of missing values detection and imputation of outliers, and adaptive normalization of continuous values. For the supervised stream mining problems, they extended the data with a class attribute lagged by one time step and by the size of the prediction window. They proceeded with attribute aggregation, mostly based on prior data knowledge and their experience.

In our system, we include all basic data preprocessing techniques that are described above. We also provide lagged class attributes, and descriptive statistics, such as the average, mode and others. Temporal attribute aggregation is based on online implementations of discrete and continuous distributions that allow on-the-fly construction of different attributes (e.g., lagged, minimum, average, mode, median, randomly sampled etc.). Stream mining experts defined several periods, based upon natural and human cycles (e.g., hourly, daily, weekly, monthly, yearly). In conjunction with expected frequency of data generation (RK3) we automatically generate aggregate and lagged attributes, such as (for an hourly cycle) data reading an hour ago and hourly cyclic attributes (i.e., sine and cosine with an hourly period).

3.6 Architecture of the Stream Mining Expert System

Figure 2 illustrates the high-level architecture of our proposed system. The system in the figure has two inputs: (1) domain description and user requirements (denoted with ②) and (2) a data stream itself (denoted with ①). Knowledge base consists of decision rules and tables that were elicited from data mining experts. We use it for initial method and parameter selection, selection of evaluation protocol with respect to the stream properties, and on-line parameter trimming (denoted with ③ and ④). These entries are continuously refined during stream mining by including better parameter settings produced by meta learner (denoted with ⑥ and ⑦).

A selected subset of appropriate stream mining methods (utilizing WEKA, MOA, and IBLStreams toolkits) is run in parallel for a given data stream (denoted with ⑨). The induced stream models are evaluated in the model evaluator

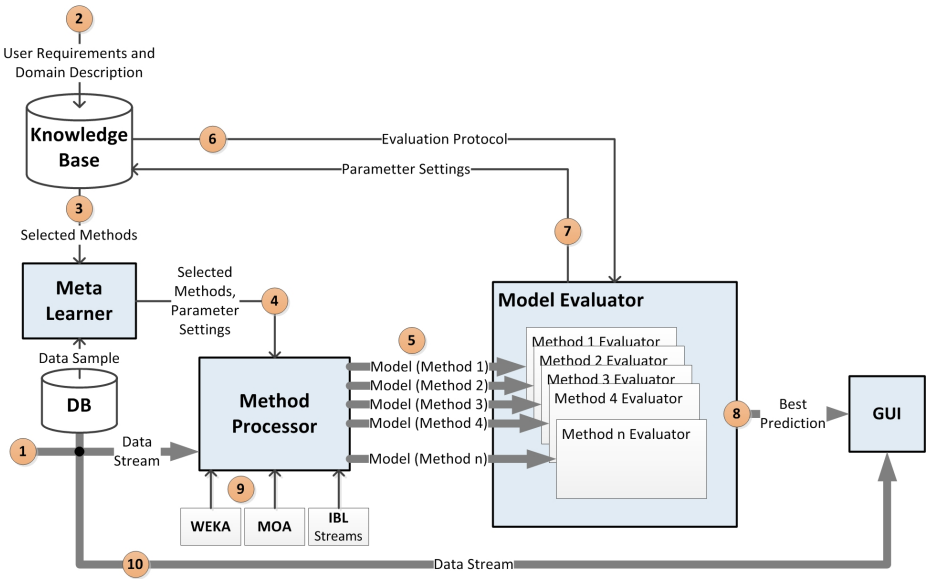


Fig. 2. Expert system high-level architecture

(denoted with ⑤). Its function is twofold: (1) to assist meta learner's feedback loop in order to select better methods and their parameters, and (2) to provide the best current model's results (usually a prediction) to GUI (denoted with ⑧).

3.7 Meta Learner

In our stream mining expert system, as depicted in Figure 2, the *meta learner* component plays a crucial part. It is used for both initial selection of applicable stream mining methods and their parameter setting and optimization of stream mining methods' parameters during execution.

Initially, a subset of methods is selected, based upon client's requirements and domain description. For this purpose, expert knowledge encoded in the knowledge base is used. An example of encoded knowledge used for method selection is shown in Table 1. When a subset of methods $M = \{M_1, M_2, \ldots, M_n\}$ is selected, they are also assigned a set of default parameters (part of the knowledge base), and a set of tunable[1] parameters with their tuning range (both default parameters and sensible tuning range are also parts of the knowledge base).

For the purpose of parameter optimization, for each method M_i we run several instances M_{ij} in parallel with slightly (randomly) perturbed tunable parameters. All method instances M_{ij} are run in parallel on the same data stream. Further parameter optimization is performed with fairly standard genetic algorithm approach [14]. The interval for production of fitness values is determined from client's required timespans and frequency of predictions (RK3).

[1] Note that not all parameters can be tuned during method's execution. An example is changing neural network's topology.

For the purpose of genetic operators parameter values are Gray-encoded within the tuning range with resolution of 10 bits. Standard crossover and mutation operators are used in conjunction with tournament selection. Besides this we also use twofold elitist approach. For the next generation we always save the currently best instance $M_{i\max j}$ and the instance with default parameter settings M_{i0}. When performance of the currently best instance is significantly ($p < 0.01$) better than the performance of the instance M_i0 with default parameters, the new set of parameters is forwarded (see ④ in Figure 2) from the method processor into the knowledge base alongside with the domain description.

Our current approach assumes that we have enough resources available to run several method instances in parallel (in total possibly hundreds of instances). We intend to explore also a resource-scarce scenario when for each method M_i we will have only a single instance for parameter tuning. It will work on windowed samples and serially optimize parameter values.

3.8 Integration within an Information System

From the information system engineering point of view, such an expert system, once developed, can be reused in various contexts and information systems. Its integration within an information system that deals with data streams and requires the stream prediction, is straightforward due to a very clear interface - the data stream is routed to the expert system for the mining and the predicted stream with the measure of confidence is sent back.

4 Empirical Analysis

4.1 Experimental Framework

We adopt standard statistics for measuring the performance of classifiers: the classification accuracy (*CA*), Cohen's κ (Kappa) statistic, geometric mean of recall and precision denoted *F*-score, and Rand index. We evaluate regression algorithms with the mean absolute error (*MAE*), mean absolute percentage error (*MAPE*), root mean squared error (*RMSE*) and Pearson correlation coefficient. All statistics are computed using a sliding window of the most recent examples.

Additionally, for comparing the performance of two particular classifiers A and B, we use the Q-statistic:

$$Q_{A,B} = \log_2 \frac{CA_A + 1}{CA_B + 1}.$$

When $Q_{A,B} > 0$, classifier A performs better that B (and vice-versa).

4.2 Experimental Datasets

The stream mining tool was analyzed on four datasets from real-world problems (Table 2). First two datasets were used to test the system's classification performance and the remaining two were used to test the regression prediction performance. The datasets were fed to the stream mining methods in temporal order of learning examples. Short description of these datasets is as follows:

Table 2. Temporal datasets used in the analysis. (Hz/3600 is the frequency of the data in examples per hour, while Δ is the size of the prediction window.)

	Dataset	Attributes	Examples	Hz/3600	Δ
Classification	Airline flight delays (2008)	$8+1$	21600	≈ 10	0
	Electricity market price ($\sim$1996)	$6+1$	17520	2	0
Regression	Electric energy consumption	$1+1$	16200	1	24
	Solar energy forecast (1994-2007)	$16+1$	5113	24^{-1}	0

- **Flight delay prediction within the USA.** The dataset[2] was published at the Data Expo Competition in 2009 and represents an actual non-stationary streaming real-world problem. It contains flight arrival and departure details for all the commercial flights within the USA between 1987 and 2008. The goal is to predict a flight delay based on the available attributes that include *date and time*, *carrier id*, *flight number*, *actual elapsed time*, *origin*, *estination*, *distance*, and *diverted*;
- **Electricity market price in New South Wales.** The dataset was collected from the Australian New South Wales Electricity Market. In this market, prices are affected by demand and supply of the market; they are set every five minutes. The dataset contains 45,312 instances. The attribute set consists of one temporal and five other attributes (*price*, *demand*, *transfer*, *day*, *period)*, and a binary class that specifies whether the price is higher than the moving average of the last day. We supplemented the dataset with attributes that describe daily and weekly periodic cycles;
- **Electric energy consumption of Portugal.** The prediction goal is to continuously predict the electricity load demand for a certain region of Portugal for the next day, based on a stream of measurements that arrive in one hour intervals. Examples contain only one temporal attribute. We constructed additional attributes that describe daily, weekly and yearly periodic cycles; and
- **Solar energy forecast for Oklahoma.** The "Solar energy forecast for Oklahoma" dataset deals with predicting the average daily incoming solar energy at 98 Oklahoma Mesonet sites based on data between 1997 and 2004. The solar energy was directly measured by a pyranometer at each Mesonet site every 5 minutes and summed from the sunrise to 23:55 UTC of the date listed in each column. Numerical prediction data include predictions of 11 ensemble members for various time steps.

4.3 Results and Discussion

To objectively evaluate our approach, we performed the experiments in the following three iterations:

[2] `http://stat-computing.org/dataexpo/`

Table 3. Classification performance for prediction of the **airline flight delays within the USA**: classification accuracy (CA), (Kappa) statistic, F-score, and Rand index. Statistics are computed with a sliding window of 10000 examples.

Algorithm		κ	CA	F	$Rand$
System	META	**0.1562**	**57.81%**	**0.5781**	**0.5781**
	RULES	0.1526	57.71%	0.5755	0.5771
Expert	BAYES	0.1455	57.28%	0.5728	0.5728
	ENSMB	0.1449	57.25%	0.5725	0.5725
Best on raw data		0.1381	56.91%	0.5691	0.5691

1. As a baseline approach, we utilized our stream mining system without any expert system support and measured the results. Only raw streaming data without any preprocessing and attribute aggregation were used. We selected the results of the best performing stream mining algorithm (selected from those described in Section 3.3) as the reference point for the following iterations. These results are denoted with "Best on raw data" in Tables 3–6.
2. In the next step we executed tests by fully involving the stream mining experts in method selection, attribute aggregation, parameter setting and trimming. The obtained results are denoted with "Expert" in Tables 3–6; we consider them as the golden standard of what our expert system can optimally achieve.
3. In the last iteration, we repeated the experiments using a support of the proposed expert system. Expert system's results are denoted with "System" in Tables 3–6.

The results for each of four used datasets are shown in tables, which show the results of our expert system (System) with respect to the baseline approach

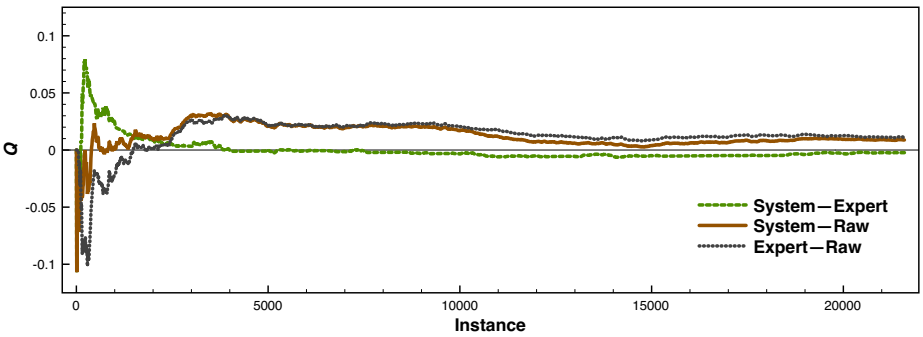


Fig. 3. Q-statistic for comparing three prediction systems for the **airline flight delays within the USA**. Positive values of the curve denote better performance of algorithm A in pair A–B.

Table 4. Classification performance for prediction of the **electricity market price in New South Wales**: classification accuracy (*CA*), (Kappa) statistic, *F*-score, and Rand index. Statistics are computed with a sliding window of 10000 examples.

Algorithm		κ	*CA*	*F*	*Rand*
System	META	0.5271	76.49%	0.7650	0.7649
	BOOST	**0.5287**	**76.61%**	**0.7660**	**0.7661**
Expert	ENSMB	0.4540	72.81%	0.7283	0.7281
	TREE	0.4333	71.70%	0.7174	0.7170
Best on raw data		0.2494	62.80%	0.6273	0.6280

Table 5. Regression performance for prediction of the **electricity consumption of Portugal**: mean absolute error (*MAE*), mean absolute percentage error (*MAPE*), root mean squared error (*RMSE*) and Pearson correlation coefficient. Statistics are computed with a sliding window of 8760 examples.

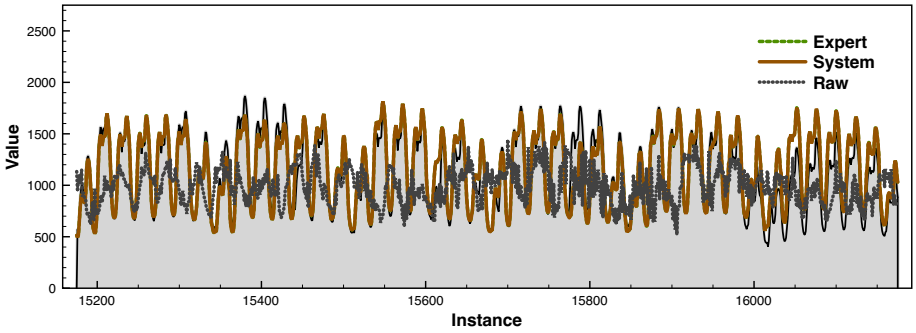
Algorithm		*MAE*	*MAPE*	*RMSE*	*Pearson*
System	META	82.79	8.73%	122.96	0.9339
	KNN	**81.54**	**8.58%**	**121.27**	**0.9357**
Expert	DISCT	111.17	12.01%	148.39	0.9044
	ADDIT	167.78	19.68%	206.66	0.8023
Best on raw data		326.75	38.49%	388.90	−0.0124

Table 6. Regression performance for prediction of the **solar energy forecast for Oklahoma**: mean absolute error (*MAE*), mean absolute percentage error (*MAPE*), root mean squared error (*RMSE*) and Pearson correlation coefficient. Statistics are computed with a sliding window of 1000 examples, while *MAE* and *RMSE* are in 10^6.

Algorithm		*MAE*	*MAPE*	*RMSE*	*Pearson*
System	META	**1.8949**	**17.19%**	**2.5072**	**0.9326**
	KNN	**1.8949**	**17.19%**	**2.5072**	**0.9326**
Expert	RULES	3.4086	32.51%	4.6129	0.7362
	TREE	5.8087	57.19%	6.8556	−0.1006
Best on raw data		1.9261	17.87%	2.6390	0.9240

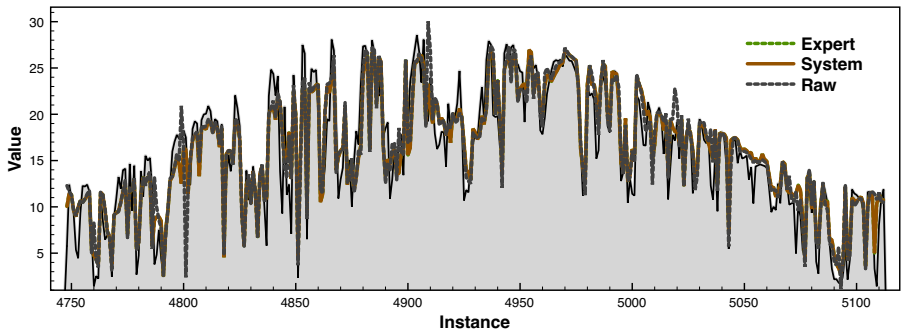
(Raw) and stream mining experts (Expert), and figures, which display evolutions of the *Q*-statistic or *MAPE* over time. The detailed results are as follows:

– **Flight delay prediction within the USA.** The results are shown in Table 3 and Figure 3. We can observe that the results for System outperform the Expert (green curve), while in the long run the Expert slightly prevails. Both System and Expert perform better than Raw.



(a)

Fig. 4. Predictions of three systems on the **electricity consumption of Portugal**. Note that the results for System and Expert overlap.



(a)

Fig. 5. Predictions of three systems on the **solar energy forecast for Oklahoma** (the values are in 10^6). Note that the results for System and Expert overlap.

- **Electricity market price in New South Wales.** The results are shown in Table 4. We can observe that in this case System performs slightly worse than Expert, while both perform considerably better than Raw.
- **Electric energy consumption of Portugal.** The results are shown in Table 5 and Figure 4. As with the previous dataset, we can observe that System performs slightly worse than Expert, while both perform considerably better than Raw. We can also notice weekly periods in Figure 4, which reflect different electricity demands on weekends.
- **Solar energy forecast for Oklahoma.** The results are shown in Table 6 and Figure 5. We can observe that System initially performs slightly worse than Expert, while in the long run the difference is negligible. However, neither performs considerably better than Raw. We can also notice yearly

periods in Figure 5, which reflect seasonal solar radiation levels. In comparison with Kaggle competition results (http://www.kaggle.com), performance of both System and Expert (and Raw) is comparable (the same order of MAPE magnitude).

The results show that our expert system often performs similarly to stream mining experts. In the "Airline flight delay prediction" and the "Electricity market price in New South Wales" datasets, expert system performed almost the same as the stream mining experts, in the former it initially even outperformed them. We can attribute this to use of the knowledge base, using which the expert system was able to quickly select a well-performing mining method, opposed to the approach of experts that included extensive initial testing of various parameters. Nevertheless, in the long run, the experts' predictions performed better.

The expert system performed better that the baseline approach (best streaming method on raw data), as we expected, in three out of four datasets. It achieved poor performance only on the "Solar energy forecast for Oklahoma" dataset, which turned out to be a difficult prediction problem even for the stream mining experts who achieved poor performance as well. The reason for this is that due to weather factors which cause that the last year's data on the same day may be considerably different from this year's.

5 Conclusion

In this paper, we focus on data that comes in streams. Mining streams has additional challenges as data is only available limited amount of time and — as a whole — cannot be stored for later use. We show that the stream mining process which normally depends on both the streaming data and data mining experts, can be fully automatized within an information system encoding the experts' knowledge. Although implemented expert knowledge is limited both in expressiveness and functionality, abundance of data in data streams seems to largely compensate this gap, as in the long run, experts' and expert system's performance is (at least in our experiments) quite similar.

In our work we wished to emphasize that solving data mining problems typically requires involvement of individuals with expertise in data mining techniques and approaches. This alone puts some limitations on the application of data mining in business practice as such knowledge is rarely available among employees and needs to be outsourced. Besides the obvious financial limitations (such expertise is costly), and, especially in stream mining, there is an ongoing need for experts' involvement due to data streams' dynamic nature. Therefore, each data mining problem needs to be tackled independently as there seems to be no general solution. The above indicates that the data mining is not fully exploited in business environments and computerized within their supporting information systems. This is a pity, since many information systems today have access to immense amounts of current and historical data.

The results of our study show that the stream mining expertise has become routinized enough to be captured in a form of explicit knowledge and thus

computerized. We believe that this represents an important finding which might impact the level of possible automatization of problems known from the *Big Data*, *Internet of Things* and similar domains.

References

1. Aha, D.W.: Generalizing from case studies: A case study. In: Proceedings of the International Conference on Machine Learning (MLC 1992), Aberdeen, Scotland, pp. 1–10 (1992)
2. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: Proceedings of the ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2002), Madison, WI, USA, pp. 1–16 (2002)
3. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: MOA: Massive online analysis. J. Mach. Learn. Res. 11, 1601–1604 (2010)
4. Cormode, G.: Conquering the divide: Continuous clustering of distributed data streams. In: Proceedings of the International Conference on Data Engineering (ICDE 2007), Istanbul, Turkey, pp. 1036–1045 (2007)
5. Gama, J., Medas, P., Castillo, G., Rodrigues, P.P.: Learning with drift detection. In: Bazzan, A.L.C., Labidi, S. (eds.) SBIA 2004. LNCS (LNAI), vol. 3171, pp. 286–295. Springer, Heidelberg (2004)
6. Gama, J., Brazdil, P.: Characterization of classification algorithms. In: Pinto-Ferreira, C., Mamede, N.J. (eds.) EPIA 1995. LNCS, vol. 990, pp. 189–200. Springer, Heidelberg (1995)
7. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. SIGKDD Explorations 11(1), 10–18 (2009)
8. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001), San Francisco, CA, USA, pp. 97–106 (2001)
9. Kriegel, H.-P., Borgwardt, K.M., Kröger, P., Pryakhin, A., Schubert, M., Zimek, A.: Future trends in data mining. Data Min. Knowl. Discov. 15(1), 87–97 (2007)
10. Matheus, C.J., Rendell, L.A.: Constructive induction on decision trees. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 1989), Detroit, MI, USA, pp. 645–650 (1989)
11. Merz, C.J.: Dynamical selection of learning algorithms. In: Learning from Data: Artificial Intelligence and Statistics, pp. 281–290. Springer (1996)
12. Pratt, L., Jennings, B.: A survey of connectionist network reuse through transfer. In: Learning to Learn, pp. 19–43. Springer (1998)
13. Rodrigues, P.P., Gama, J., Bosnic, Z.: Online reliability estimates for individual predictions in data streams. In: Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW 2008), Pisa, Italy, pp. 36–45 (2008)
14. Rossi, A.L.D., Soares, C., Carvalho, A.C.P.L.F.: Bioinspired parameter tuning of MLP networks for gene expression analysis: Quality of fitness estimates vs. Number of solutions analysed. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008, Part II. LNCS, vol. 5507, pp. 252–259. Springer, Heidelberg (2009)
15. Shaker, A., Hüllermeier, E.: IBLStreams: a system for instance-based classification and regression on data streams. Evolving Systems 3(4), 235–249 (2012)
16. Wolpert, D.H.: Stacked generalization. Neural Networks 5(2), 241–259 (1992)