Procedia Computer Science

Volume 80, 2016, Pages 1724–1733

ICCS 2016. The International Conference on Computational Science

Active Learning Classification of Drifted Streaming Data

Michał Woźniak[1], Paweł Ksieniewicz[1], Bogusław Cyganek[2,3], Andrzej Kasprzak[1], and Krzysztof Walkowiak[1]

[1] Department of Systems and Computer Networks, Faculty of Electronics
Wrocław University of Technology
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland.
e-mail: [michal.wozniak, andrzej.kasprzak, pawel.ksieniewicz, krzysztof.walkowiak]@pwr.edu.pl

[2] ENGINE Center, Wrocław University of Technology
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland.

[3] AGH University of Science and Technology Al. Mickiewicza 30, 30-059 Kraków, Poland e-mail: cyganek@agh.edu.pl

Abstract

Objects being recognized may arrive continuously to a classifier in the form of data stream, therefore contemporary classification systems have to make a decision not only on the basis of the static data, but on the data in motion as well. Additionally, we would like to start a classifier exploitation as soon as possible, then the models which can improve their models during exportation are very desirable. Basically, we may produce the model on the basis a few learning objects only and then we use and improve the classifier when new data comes. This concept is still vibrant and may be used in the plethora of practical cases. Nevertheless, constructing such a system we should realize, that we have the limited resources (as memory and computational power) at our disposal. Additionally, during the exploitation of a classifier system the chosen characteristic of the classifier model may change within a time. This phenomena is called *concept drift* and may lead the deep deterioration of the classification performance. This work deals with the data stream classification with the presence of *concept drift*. We propose a novel classifier training algorithm based on the sliding windows approach, which allows us to implement forgetting mechanism, i.e., that old objects come from outdated model will not be taken into consideration during the classifier updating and on the other hand we assume that only part of arriving examples can be labeled, because we assume that we have a limited budget for labeling. We will employ active learning paradigm to choose an "interesting" objects to be be labeled. The proposed approach has been evaluated on the basis of the computer experiments carried out on the data streams. Obtained results confirmed the usability of proposed method to the smoothly drifted data stream classification.

Keywords: active learning, pattern classification, data streams, incremental learning, sliding window

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2016

doi:10.1016/j.procs.2016.05.514

1 Introduction and related works

One can say with certainty that we are living in the age of big data. This term refers to an ability to collect and analyze the vast amounts of data to understand the world and everything within it[1]. Big data is usually described by so-called 5Vs, i.e.,

- **V**olume, which focuses on data size.
- **V**elocity, which considers that data arrives usually continuously.
- **V**ariery, which points out that data may be in the different forms and comes from various sources.
- **V**eracity, which refers to data credibility and quality.
- **V**alue, which is the most important characteristic, because without extracting the value from data all analytical tools are worthless.

In this work we will deal with data in motion (Velocity) to design an appropriate models for data stream analysis. We will focus on designing efficient method of analyzing successive arriving objects and our interest lies on the classification methods for data stream. The aim of this task is to find such a function Ψ which can assign a correct label i ($i \in \mathcal{M}$, where $\mathcal{M}$ is the finite set of labels) to each incoming example x ($x \in \mathcal{X}$, where $\mathcal{X} \subseteq \Re^d$ is the feature space).

The classifier learning methods are looking for such a classification model which can minimize the overall cost of the wrong decisions. To formulate the classifier learning task we have to define so-called loss function [1], which assesses the cost of the wrong decision between each pair of classes, i.e.,

$$L : \mathcal{M} \times \mathcal{M} \to \Re \tag{1}$$

$L(i, j)$ returns the lost connected with the wrong assigning object from class j to class i. Then we may formulate the criterion of classification task for the optimal Bayes classifier

$$\min_{\Psi} Risk(\Psi) = Risk(\Psi^*) \tag{2}$$

where

$$Risk(\Psi) = E[L(i, j)] = \int_{\mathcal{X}} \sum_{j=1}^{M} L(\Psi(x), j) p_j f_j(x) dx \tag{3}$$

The $Risk(\Psi)$ is known as the average (overall) risk of the classifier Ψ. Our aim is to minimize it. If we consider a specific, but very popular case of the 0-1 loss function which returns 1 in the case of error and 0 otherwise, then the optimal classifier returns class for which the *posterior* probability $p_j(x)$ has the biggest value and it is given by

$$p_j(x) = \frac{p_j f_j(x)}{\sum_{k=1}^{M} p_k f_k(x)} \tag{4}$$

[1] Bernard Marr,Big Data: The Mega-Trend That Will Impact All Our Lives, https://www.linkedin.com/pulse/20130827231108-64875646-big-data-the-mega-trend-that-will-impact-all-our-lives

Unfortunately, we do not know the real values of the probabilistic characteristics, therefore we have to use a machine learning approach to find the best model. Usually, such a task assumes the family of models, what leads to find the best parameter setting a^* of the classifier model Ψ for a given learning set DS. During classifier learning task, we minimize the criterion related to overall risk $\overline{Risk}$ using the collected learning data, i.e., to find a^* for a given set of observations.

$$a^* = \min_a \overline{Risk}(\Psi(x, a), DS) \tag{5}$$

Let's consider that data is available in the form of data stream DS, i.e.,

$$DS = (x_1, j_1), (x_2, j_2), ..., (x_n, j_n) \tag{6}$$

where x_i denotes observation of the ith examples and j_i its label. Additionally, we know that example x_j arrives earlier than x_k if $j < k$. Basically, we may consider two cases: (i) when the probability characteristics of the classification task are time-homogenous or (ii) they may change over time.

1.1 Stationary data streams

Because in this case the probabilistic characteristics do not depend on time, then we may consider the following approaches:

- collecting the sufficient number of labeled learning examples to produce the classifier [2] and then use it to classify incoming objects.
- collecting a few labeled examples only to produce the first prototype of a classifier and then using it to classify new incoming object and update the model's parameters if new labeled object are available (the updating may employ incremental or epoch learning mode). We may enumerate several works as an incremental AQ [3] or STAGGER [4] and works on incremental decision tree learning as Very Fast DEcision Tree [5].

1.2 Non-stationary data streams

Unfortunately, most of canonical classifiers do not take into consideration that the probabilistic characteristics of a classification task may change over time. Such phenomena is called *concept drift* [6] and we may meet it in many practical issues, as spam filtering, intrusion detection/prevention (IDS/IPS), or recognition of client behavior to enumerate only a few. There are several taxonomies of concept drift. According to its influence on the probabilistic characteristics we may list [7]:

- virtual concept, which does not impact the decision boundaries (some say the *posterior* probabilities do not change, but it is disputable), but affect the probability density functions [8].
- real concept drift, which affects the *posterior* probabilities and may impact unconditional probability density function [6], what means that it can strongly change the shape of the decision boundary

Furthermore, we have to take into consideration the impetuosity of type of changes (sudden or smooth changes) in the classification model when we choose an appropriate method how to deal with the *concept drift*.

Basically, the following approaches may be considered to deal with the above problem.

- Building new model from scratch, if new data becomes available. It is very expensive and impossible from a practical point of view, and especially if drift occurs rapidly. It means, that from time to time we have to collect sufficient number of labeled examples grouped in DS_n to estimate the best parameter setting

$$a_n^* = \min_a \overline{Risk}(\Psi(x, a), DS_n) \tag{7}$$

- Detecting concept changes in new data, and if these changes are *sufficiently* significant then rebuil the classifier or build new model from scratch.Changes are usually discovered by monitoring classification accuracy [9].

- Adopting an incremental learning algorithm for the classification model.

$$a_n^* = \min_a \overline{Risk}(\Psi(x, a), a_{n-1}^*, DS_n) \tag{8}$$

  where DS_n is data chunk, which includes at least 1 example.

  Constant update of a classifier is accomplished by using incremental learning methods that allow adding new training data during the exploitation of a classifier or by data set windowing.

The important difference between classifier learning for stationary and non-stationary data streams is that for non-stationary tasks a forgetting mechanism should be implemented. During the designing a data stream classifier we should take also into consideration the limitation of the resources as memory and computational power, as well that we cannot grant access all labels.

In this work we will focus on adapting algorithms, which may employ online or epoch mode of learning. The online learners continuously update their parameters, while processing the incoming data and according to [10] they have to process each learning example only once in the course of training and the training process may be paused at any time, and its accuracy should not be lower than that of a classifier trained on batch data collected up to the given time.

The algorithms incorporating the forgetting mechanism assume, that the recently arrived data are the most relevant. Therefore, narrowing the range of data to those that were most recently read may help form a data set that embodies the actual context [6].

1.3 Active learning

The most of the classifiers devoted to data stream or drifted data streams use supervised learning algorithms, which could produce a classifier on the basis of labeled examples. Unfortunately, from the practical point of view it is hard to be granted that labels are always available, e.g.,

- Medical diagnosis - human expert should always verifies the diagnosis, i.e., we have to ensure the continuous access to the human expert.

- Credit application (the true label is usually available ca. 2 years after the decision, then such labeled example could be worthless as come from outdated model);

- Spam filtering - user should confirm the decision if incoming mail is legitimate or not.

In this approach we should take into consideration the cost of data labeling, which is usually passed over. Let us notice that labels are usually assigned by human experts and therefore they

can not label all new examples if they come too fast. Therefore methods of classifier design which could produce the recognition system on the basis of a partially labeled set of examples (called active learning) would be an attractive proposition [11].

The key idea behind active learning [12] is that a machine learning algorithm can achieve greater accuracy with fewer training labels if it is allowed to choose the data from which it learns. An active learner may pose queries, usually in the form of unlabeled data instances to be labeled, e.g., by a human expert. Active learning is well-motivated in many modern machine learning problems, where unlabeled data may be abundant or easily obtained, but labels are difficult, time-consuming, or expensive to obtain.

2 Active learning classifier for streaming data

Let us propose the classifier learning framework, which employs the active learning paradigm. This framework works as the block classifier, because it collect the data in the form of chunk, but for each chunk the online learner is used. The decision about the object labeling depends on two parameters:

- *threshold* - which is responsible for choosing the "interesting" examples, i.e., if support function related with the decision is lower than a given threshold then the object seems to be interesting and the algorithm asks for its label. The different types of the threshold may be considered, e.g., the differences between support functions (lower differences should prefer labeling) for the binary classification task or differences between the support function related to the decision and average values of each supports etc.
- The label will be assigned (i.e., algorithm will pay for it) only in the case if its budget related to a given chunk will allows to pay for it. For each chunk only limited percentage of the object could be labeled (depends on parameter *budget*).

Because the data stream is collected in the form of chunk, but each chunk is processing incrementally, therefore before its analysis, the order of the collected objects is randomizing. The pseudocode of the algorithm is presented in Alg. 1.

3 Experiments

The main goal of the experiments is to check dependencies between algorithm's parameters (chunk size, threshold and budged for labelling) and the accuracy and stability of the classifier. The experiments were carried out for the 3 artificially generated data streams available in MOA:

1. WaveformGeneratorDrift generates a problem of predicting one of three waveform types, each of which is generated from a combination of two or three base waves [13],
2. RandomTreeGenerator generates a stream based on a randomly generated tree [5], i.e., by choosing attributes at random to split, and assigning a random class label to each leaf. Once the tree is trained, the stream is generated by assigning examples generated according uniform distribution to classes returned by the tree.
3. LEDGeneratorDrift - Generates a problem of predicting the digit displayed on a 7-segment LED display, where where each attribute has a 10% chance of being inverted [14].

and 4 online learnes:

Algorithm 1 Active learning classifier for data stream

Require: input data stream,
n - data chunk size,
incremental training procedure(),
classifier(),
budget - max. percent of labeled example in a chunk,
treshold

$i \leftarrow 0$
initialize classifier()
repeat
  collect new data chunk $DS_i = \{x_i^1, x_i^2, ..., x_i^n\}$
  set random order of collected examples in DS_i
  for $j = 1$ **to** n **do**
    support $\leftarrow$ *max* (support function value returned by the classifier for x_i^j)
    if *support* $<$ *treshold* **then**
      if *budget* > 0 **then**
        ask for the label of the jth example
        classifier $\leftarrow$ incremental training procedure(x_j)
        budget $\leftarrow$ *budget* $- \frac{1}{n}100\%$
      end if
    end if
  end for
  $i \leftarrow i + 1$
until end of the input data stream

1. minimal distance classifier (k-NN),
2. Naïve Bayes, which is the classic bayesian classifier while making assumption that all features are independent (NaiveBayes).
3. Single layer perceptron (Perceptron),
4. Rule-base classifier (RuleClassifier).

All experiments were carried out in the programming language Java using MOA (Massive Online Analysis) experimental software environment [2]. MOA [13] is a very popular framework for data stream analysis. It includes tools for evaluation and a collection of machine learning algorithms. All source codes may be downloaded from https://github.com/xehivs/moa and experimental setting may be generated using the following form http://156.17.43.89/config/moa.html.

To evaluate the classifier "test and train" method was chosen (known also as "block evaluation method") [13]. Each new classifier is trained on a given data chunk, but its error is estimated on the basis on the next (unseen) portion of data.

The results of experiments are presented in Fig.1-4.

[2] http://moa.cms.waikato.ac.nz/

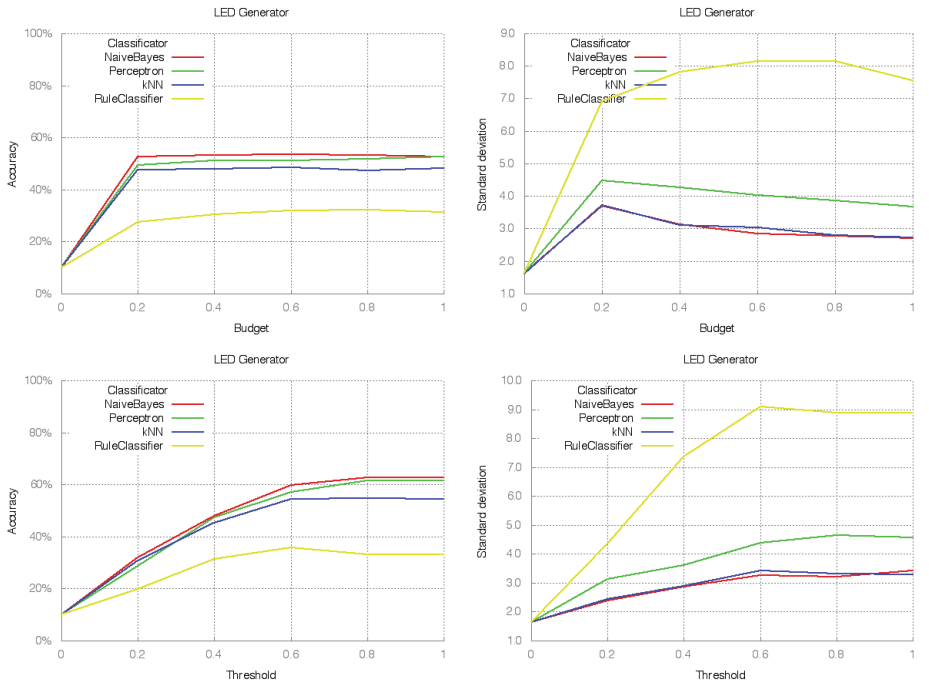

Figure 1: Dependencies between accuracy (left), standard deviation (right) and budget (top), threshold (bottom) for different classifiers and data stream generated by LED Generator.

3.1 Discussion

This research follows-up to our previous study on the active learning presented in [15, 16]. The following observation could be drawn on the basis of experiments:

- The accuracy does not strongly depend on the number of labeled objects. In our experiments 20% is enough to achieve very similar accuracy as full supervised approach. The budget plays more important role for small data chunk size.
- Unfortunately, to stabilize classifiers much more objects should be labeled (40-60%).
- The threshold is important, but for LED and Waveform data stream it could be set on 60%, while for RandomTree 80% is required. What is interesting - bigger threshold negatively impacts the stability of classifiers (the standard deviation increases).
- The k-NN and Näive Bayes classifiers behaved well, while the RuleClassifier and Perceptron look unstable (especially RuleClassifier), what is rather surprising, especially for the Perceptron. Both lost the stability for the big threshold. Probably, the huge diversity of the example does not able to stabilize the model.

4 Conclusions

The paper presented the active learning of data stream classifier. The computer experiments on several benchmark data stream confirmed that proposed method can adapt to changes and

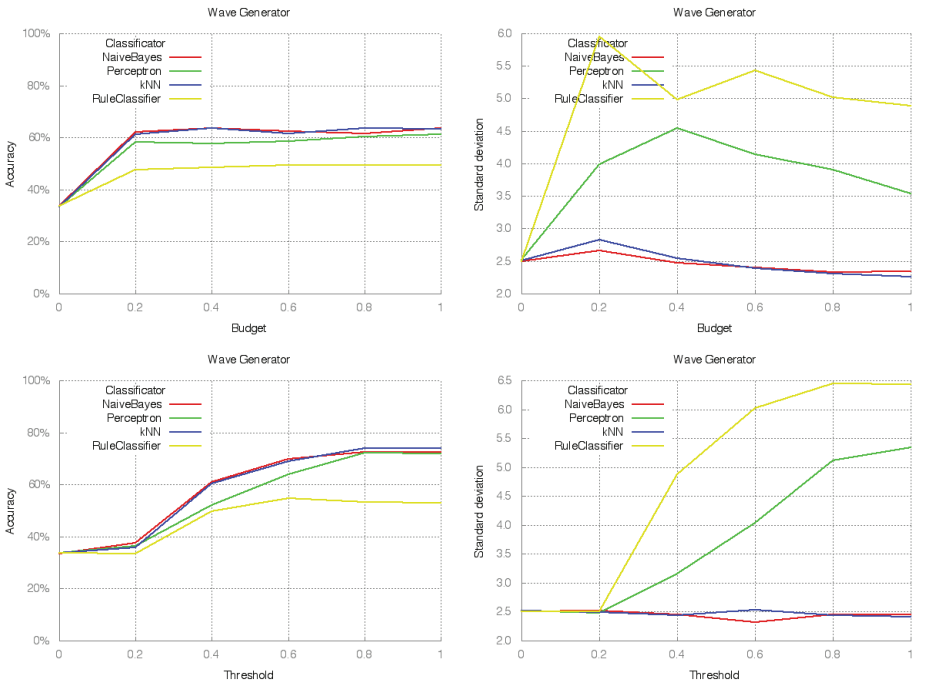

Figure 2: Dependencies between accuracy (left), standard deviation (right) and budget (top), threshold (bottom) for different classifiers and data stream generated by Waveform Generator.

our intuition has not let us down, that the semi-supervised learning (especially based on active learning) may return the similar results as the fully supervised approach. In the near future we are going to extend the study on proposed concept, i.a., evaluating new methods of threshold count and adopting the proposed active learning framework to a classifier ensemble.

5 Acknowledgement

This work was supported by the statutory funds of the Department of Systems and Computer Networks, Faculty of Electronics, Wroclaw University of Science and Technology and by the Polish National Science Centre under the grant no. DEC-2013/09/B/ST6/02264. This work was also supported by EC under FP7, Coordination and Support Action, Grant Agreement Number 316097, ENGINE European Research Centre of Network Intelligence for Innovation Enhancement (http://engine.pwr.edu.pl/). All computer experiments were carried out using computer equipment sponsored by ENGINE project.

References

[1] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[2] L. G. Valiant, "A theory of the learnable," *Commun. ACM*, vol. 27, no. 11, pp. 1134–1142, Nov. 1984.

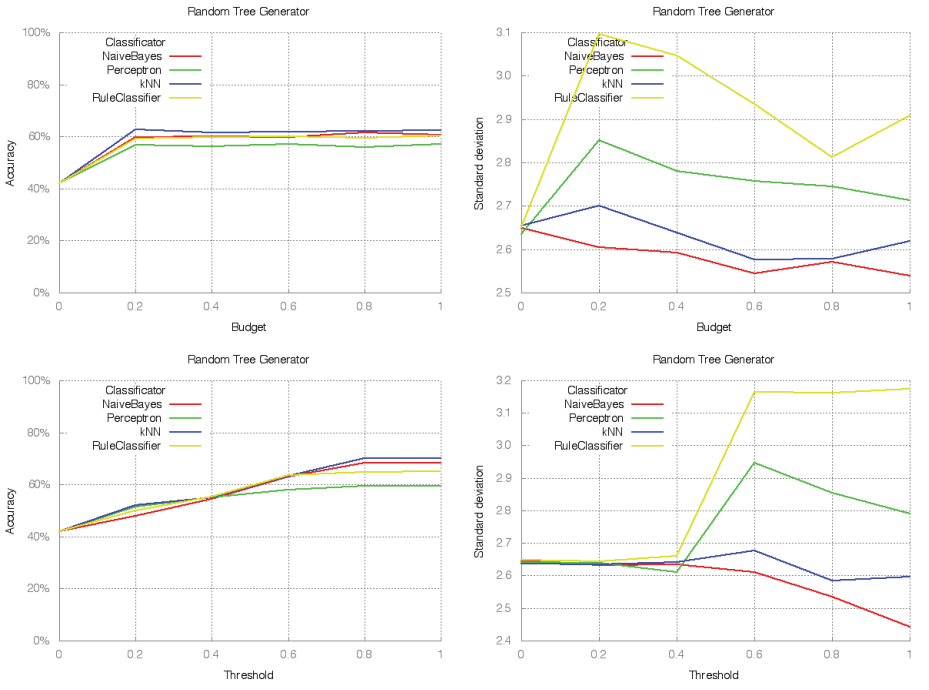

Figure 3: Dependencies between accuracy (left), standard deviation (right) and budget (top), threshold (bottom) for different classifiers and data stream generated by Random Tree Generator.

[3] R. Michalski and J. Larson, *Selection of Most Representative Training Examples and Incremental Generation of VL[subscript 1] Hypotheses: the Underlying Methodology and the Description of Programs ESEL and AQ11.* University of Illinois at Urbana-Champaign. Department of Computer Science, 1978.

[4] J. C. Schlimmer and R. H. Granger, Jr., "Incremental learning from noisy data," *Mach. Learn.*, vol. 1, no. 3, pp. 317–354, Mar. 1986.

[5] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '00. New York, NY, USA: ACM, 2000, pp. 71–80.

[6] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Mach. Learn.*, vol. 23, no. 1, pp. 69–101, Apr. 1996.

[7] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. in press, 2013.

[8] G. Widmer and M. Kubat, "Effective learning in dynamic environments by explicit context tracking," in *Machine Learning: ECML-93*, ser. Lecture Notes in Computer Science, P. Brazdil, Ed. Springer Berlin Heidelberg, 1993, vol. 667, pp. 227–243.

[9] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms.* Wiley-Interscience, 2004.

[10] P. Domingos and G. Hulten, "A general framework for mining massive data streams." *Journal of Computational and Graphical Statistics*, vol. 12, pp. 945–949, 2003.

[11] R. Greiner, A. J. Grove, and D. Roth, "Learning cost-sensitive active classifiers," *Artif. Intell.*,

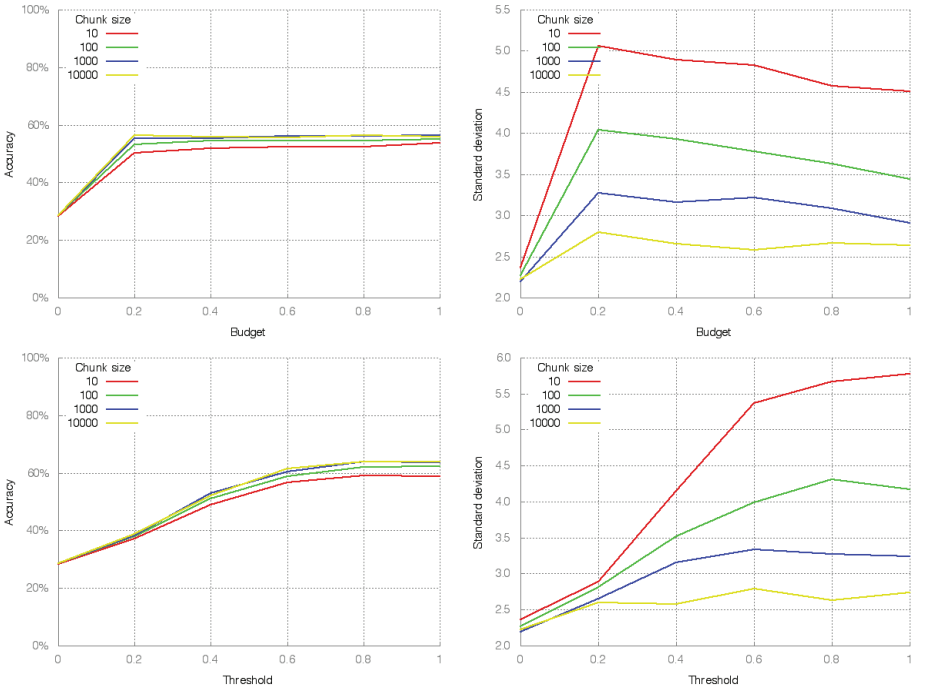

Figure 4: Dependencies between accuracy (left), standard deviation (right) and budget (top), threshold (bottom) for different chunk size. In the figures the averaged results (for different classifiers and streams) are presented.

vol. 139, no. 2, pp. 137–174, Aug. 2002.

[12] B. Settles, "Active learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[13] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "Moa: Massive online analysis," *J. Mach. Learn. Res.*, vol. 11, pp. 1601–1604, Aug. 2010.

[14] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*, ser. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984.

[15] B. Kurlej and M. Wozniak, "Active learning approach to concept drift problem," *Logic Journal of the IGPL*, vol. 20, no. 3, pp. 550–559, 2012.

[16] M. Wozniak, B. Cyganek, P. Ksieniewicz, A. Kasprzak, and K. Walkowiak, "Active learning classifier for streaming data," in *Proc. of HAIS 2016*, ser. Lecture Notes in Computer Science, e. a. Corchado E., Ed. Springer Berlin Heidelberg, 2016, vol. to be published, pp. 227–243.