# Recurring Concepts and Meta-learners

**João Gama**[1] and **Petr Kosina**[2]

## Abstract

This work addresses data stream mining from dynamic environments where the distribution underlying the observations may change over time. In these contexts, learning algorithms must be equipped with change detection mechanisms. Several methods have been proposed able to detect and react to concept drift. When a drift is signaled, most of the approaches use a forgetting mechanism, by releasing the current model, and start learning a new decision model. Nevertheless, it is not rare for the concepts from history to reappear, for example seasonal changes. In this work we present a method that memorizes learnt decision models whenever a concept drift is signaled. The system uses meta-learning techniques that characterize the domain of applicability of previous learnt models. The meta-learner can detect re-occurrence of contexts and take pro-active actions by activating previous learnt models. The main benefit of this approach is that the proposed meta-learner is capable of selecting similar historical concept, if there is one, without the knowledge of true classes of examples.

## Keywords

Data Streams, Concept Drift, Meta-learners, Recurrent Concepts

## 1 Motivation

We are living in the time of information. Information becomes more and more valuable for any kind of business. It can help to make strategic decisions, produce better products, find new customers... But it can also save lives when used in medicine, pharmacy etc. Therefore, with the increasing computational power and storage space, vast amounts of data is being produced and gathered every moment. It is not possible for human to manually study all the data and look for the interesting or valuable pieces of information. Once again there are computers to aid us to automatically process data, search for and retrieve relevant answers. But not all of the data posses the same characteristics so we need to study and develop new techniques that would best fit to particular problems.

Some of the sources can produce more or less static data, but more often we are faced to evolving problems, therefore also methods for data analysis should not be stationary, but should be able to adapt.

Observing the behaviour of nature can give us more than just the idea of evolution of things. There is also the tendency of reappearence. The most obvious example could be seasonal change and the reactions of animals or people to those changes. Even long thime ago people knew that after summer there comes autumn and then winter etc. and prepared reserves for worse conditions. That is why even computer decisions should consider using historical information and try to predict its recurrence.

In this work we present an approach that possesses both qualities, adaptation and using historical information. But the novelty and main motivation lies somewhere else. Imagine a situation in a bank where there are records of their clients. Computer systems can help decide if bank should give a loan to certain client. However more new clients will come and ask for loan before the bank can label clients that were reliable or not. Computer systems predict labels for records, but usually are not able to use entries without their labels for the evaluation. Since the economic situation can change, the predictions will not correspond to current state anymore and it will take some time to make corrections in decision making. We were interested if we could use the records without labels and the recurrence of history to react faster on changes.

## 2 Related Work

Existing works in mining data streams usually deals with concept drifts by detecting them and learning a new classifier that is used either alone as in [2] or as a part of ensemble like in [12, 14]. But there are not many of them taking into consideration the recurrence of concepts in the streams.

An ensemble classifier with recurrence taken into account is presented in [8]. In their work, classifiers of new concepts are stored in global set. Only the models with performance on preceding data points better than threshold are part of ensemble for labelling the examples. The threshold is set as selected fraction of error of classifier that predicts randomly (e.g. probability of classification to class $c$ is equal to $c$'s class distribution). This ensures that relevant classifiers, with weight assigned, are in the ensemble. If none of the classifiers in global set or the ensemble of them perform better than permitted error, a new classifier is built. Classifiers are tested, selected to ensemble or new one is created with every labelled chunk of examples.

Also in [16] a method reusing historical concepts is proposed. It uses proactive approach, e.g selects concept from history that will most likely follow after the current concept according to transition matrix. History of concepts is treated like a Markov Chain with concepts as states. If this approach does not have enough information to distinguish among some historical concepts, it makes decision based on historical-reactive scheme, which means it tests the historical concepts with recent examples and the one with the best accuracy is chosen. In case none of this approaches has satisfying accuracy, a new concept is learnt. Building a concept history can be boosted measuring conceptual equivalence (CE). CE compares classifications of classifier from new concept to results of each of the historical ones and computes score for each pair. If the score for one of the historical concepts is above defined threshold, new concept replaces old one in

[1] LIAAD-INESC Porto, FEP-University of Porto email: jgama@fep.up.pt
[2] LIAAD-INESC Porto, FI-Masaryk University, Brno email: petr.kosina@inescporto.pt

concept history.

In [4] Conceptual Clustering and Prediction (CPP) is presented to handle streaming data with recurrent concepts. Short batches of examples are transformed into conceptual vectors describing them (e.g. mean and standard deviation for numeric atributes, probability of attribute given the class for nominal). Conceptual vectors are clustered by their distance and for each cluster a classifier is learnt.

Recurrent concepts are also considered in [6], where multiple windows are used for tracking the concept drift. By predicting the rate of change, size of window could be adapted and when drift is estimated, repository of stored historical concepts is checked for recurrence. Concepts are described by averages of attributes (numeric) and similarity is measured by any feature distance metric.

In this work we present another approach to select older models learnt on data with similar underlying concept. Single classifier is used for predicting class labels and meta-learner is used to choose the most appropriate model from history. Every time a new classifier is not sufficient and warning is signaled, referees are asked for predictions. Whenever there is drift detected the classifier and its referee are stored in a pool for further use. Model is re-used when the percentage of votes of its referee exceeds some given threshold, otherwise new classifier is learnt. The idea of using such referees is taken from [9] where meta-learning scheme was used in off-line learning to select predictions from an ensemble of classifiers. We tried to apply the scheme in on-line learning with a single classifier for simplicity.

## 3 The Two-Layers Learning System

The system we propose uses a two layer learning scheme. Each layer trains its own classifier and receives its own data. The first layer receives the data stream and trains a classifier using the labeled examples. For each incoming example **x**, y, the current classifier predicts a class label. If the example is not classified, i.e. y = ?, the current model classifies the example, and reads the next example. Otherwise, e.g. the example is classified; we can compute the loss, and update the current decision model with the example. Assuming the 0-1 loss function, the prediction is either correct or incorrect, and an example is generated to train the second layer classifier. This example has the same attribute values as in the layer 0, but the class value is + if the example was correctly classified or if the example was misclassified. Doing so, the meta-learner learns the regions of the instance space where the base classifier performs well.

### 3.1 Base Model: Naive Bayes

Our approach does not depend on any certain learning algorithm. Any classifier could be used for either level 0 or level 1. Naive Bayes classifier was chosen for our experiments for good reasons. It is easy to implement and it is incremental. It has clear semantics, it is simple in representing, using and learning probabilistic knowledge, works well with continuous attributes and finally in [9] it is reported to have good results as meta-learner.

Prior probabilities, used in computation, are usually small numbers and the product of such numbers gives us even smaller numbers, which can easily lead to underflow while using computers. It is better to use the sum of logarithms and receive a score that is proportional to its probability. The formula is $\log P(c_i|\vec{x}) \propto \log P(c_i) + \sum_{k=1}^{n} \log P(x_k|c_i)$ For two class problem we can also use the expression in the form:

$$\log \frac{P(c_+|\vec{x})}{P(c_-|\vec{x})} \propto \log \frac{P(c_+)}{P(c_-)} + \sum_{k=1}^{n} \log \frac{P(x_k|c_+)}{P(x_k|c_-)} \qquad (1)$$

The class of the example is then assigned: $if \log \frac{P(c_+|\vec{x})}{P(c_-|\vec{x})} > 0$ class is $+$, $-$ otherwise.

Since we are dealing with data streams, it was necessary to use the incremental version of Naive Bayes, which needs to process each example just once. We compute the incremental version of Naive Bayes described in [3].

For handeling the continuous attributes, the classical method that approximates some distribution was chosen. In this case it was the normal or Gaussian distribution. We assume that $P(x_k|c_i) = \frac{1}{\sigma_{ki}\sqrt{2\pi}} exp\left(\frac{-(x-\mu_{ki})^2}{2\sigma_{ki}}\right)$ where $\mu_{ki}$ is mean and $\sigma_{ki}$ is standard deviation.

### 3.2 Drift Detection

In many real-world situations we can see that behaviour of observed subject is changing over time. Such changes in class distributions are called drifts and there are several known methods to detect them.

We used approach from [2] which assumes that if the distribution of the examples is stationary, the error-rate of the learning algorithm will decrease when the number of examples increases. The drift detection method manages two registers during the training of the learning algorithm, $p_{min}$ and $s_{min}$ , where $p$ is error-rate and $s$ is standard deviation. Every time a new example $i$ is processed those values are updated when $p_i + s_i$ is lower than $p_{min} + s_{min}$. We use a warning level to define the optimal size of the context window. The context window will contain the old examples that are on the new context and a minimal number of examples on the old context. In the experiments the warning level is reached if $p_i + s_i \geq p_{min} + 2s_{min}$ and the drift level is reached if $p_i + s_i \geq p_{min} + 3s_{min}$ . Suppose a sequence of examples where the error of the actual model increases reaching the warning level at example $k_w$, and the drift level at example $k_d$. A new decision model is induced using the examples starting in $k_w$ till $k_d$. It is possible to observe an increase of the error reaching the warning level, followed by a decrease. We assume that such situations correspond to a false alarm, without changing the context.

### 3.3 Learning with Model Aplicability Induction

When dealing with possibly infinite data streams one can expect there would be concept changes in data and the concepts could re-appear. We need to have some tool for recognizing if older model is appropriate for new data. Such a tool can be a referee which is a meta-learner working in similar fashion as in [9]. There are two layers in our suggested approach. The first layer is the primary model that serves as normal classifier (or level 0 classifier). Then the second layer denoted as referee (or level 1 classifier).

As in real situations there is delay between obtaining example and observing true label. Level 0 classifier makes the prediction when the example arrives. Once the true value of example's class is observed new prediction for the example is made so that the evaluation would reflect current state of the model. Then the classificator is updated and the example is passed, if defined conditions for learning are met, to the referee with a new class atrribute, which reflects whether or not the prediction was correct. The data set for referee is therefore created from all the example attributes, apart from the class attribute,

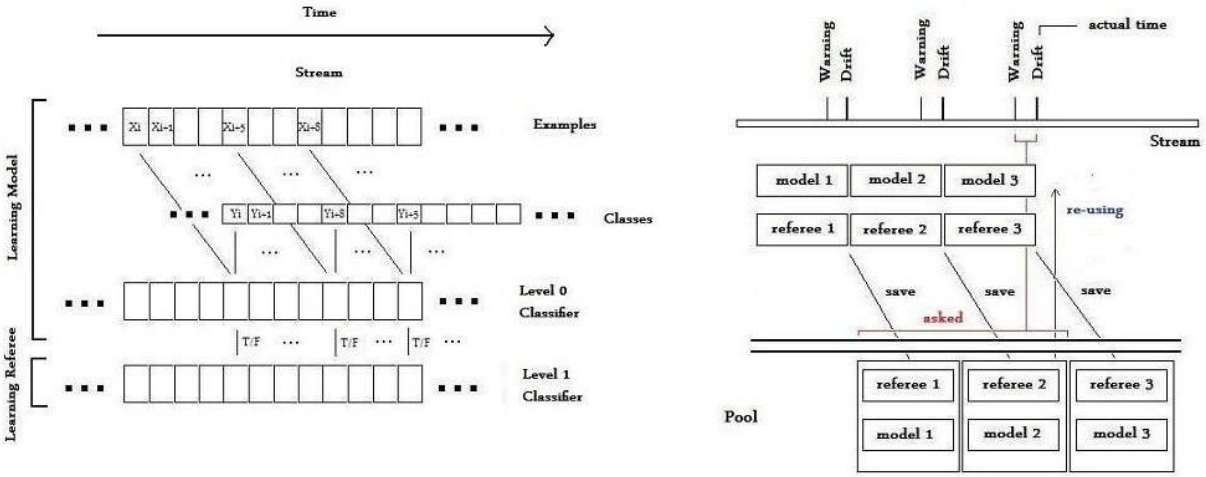


**Figure 1.** (left) Learning process. Once obtained the true label is passed along with the example attributes to level 0 classifier. Then example with true/false label is used for training level 1 classifier.
(right) Referee strategy is shown in the time of third drift. After warning, the referees in the pool were asked, model 2 was selected for re-using while model 3 and its referee were stored in the pool.

which is replaced by 0 (or false) when the prediction was wrong or 1 (or true) when the prediction was correct. The process can be illustrated by picture 1 (left).

Referee is built in the same time when primary model is built. First examples are not used in referee learning, because the misclassifications of the model are mainly due to the lack of information (in our case it was 50 examples). In fact, we have two parallel classificators on one data stream.

When there is a warning we store the incoming examples in a short term memory. This has two reasons. First, if the warning was a false alarm then we use those examples for learning like there was no warning at all. Second, if there is a drift, we use these recent examples, potentially examples from new concept, for building a new model. At this time, new referee is not learning for the same reason like in the beginning.

Part of the reaction on warning is that we stop learning primary model and we start asking the referees about the performance of their models. If model is still in the warning phase after defined number of examples, the probability of false alarm is much smaller and the examples in memory that still did not recieve their labels are also used to ask referees. If referee's prediction about the applicability of its model is greater than selected threshold, the correspondent model is chosen to be used for evaluating next examples. Otherwise process continues till drift level defined by drift detection is reached or referee's threshold value is exceeded, which is checked every 100 examples.

This is the main advantage of using referees. Many examples that do not have their labels are stored in the memory and we could exploit them to predict the change sooner with reusing old classifiers. That way it is not necessary to pasively wait till drift is reached, but proactively select recurrent model.

All new models and their referees are stored in a pool, like on the picture 1 (right). When model is to be re-used, it is taken from the the pool together with its referee and updated version is put back after drift occurred. For similar concepts there is one classifier with one referee that are continuously updated.

## 3.4 Loading Data

Data for stream mining can come and be processed either one by one or in batches. Both situations can be seen in applications. Although waiting for certain amount of examples and then processing the whole batch could be more efficient, data points in this work are taken one by one. Every time a new example is loaded, it is passed to the primary classificator to make a prediction. To simulate behaviour in real world, the true value of the class attribute needed for evaluation of model is not observed immediately, but after some delay (500 examples in this situation). Meanwhile the examples and predictions are stored in the memory.

## 3.5 Optimizing Performance

While making experiments with our referee-advised recurrence tracking method we encountered several problems closely connected to skewness of data. The distribution of examples for referees reflects the error-rate of level 0 classifier. Good performance of the primary classifier means that there will be a lot of positive examples and much less negative examples. This implies the need for some method to deal with this skewed data.

A technique without parameters is used to eliminate skewness. The referee makes decision about classifier's prediction and learn it just when it's decision is wrong e.g. referee predicted the classifier to be correct on incoming example and classifier made a mistake or vice versa. In order to be able to make decisions the referee learns from some examples in the beginning, it was 100 examples in our experiments.

In our case, we chose Naive Bayes classificator to be the referee. Since this meta-learning is a two class problem, we implemented the classificator as log-odds version of Naive Bayes defined by equation 1. The decision of classifier is chosen to be in favor for $true$ only if the log-odds score is above $0.1$ not $0$ like in usual cases. This scenario is more pesimistic one because it is better to start building a new model than using an old inappropriate one.

Another constraint was introduced in the process of saving referees. Since referees with practically the same means and standard de-

viations for both classes ($true$ and $false$) were obtained, we decided to save them only in certain conditions (reaching minimum quality): $if\left(\sum|\mu_i^1-\mu_i^2|\right)>\phi$, where $\phi$ is selected threshold (in this case 0.002) and $\mu_i^1,\mu_i^2$ are means of attributes of $true$ and $false$ class respectively.

Bad quality and skewness of data, could lead to a situation, where referee predicts model to be correct in 100 % cases because of higher prior probability of positive class, even though above mentioned precausions were taken. Therefore results with 100 % are not taken into account.

### 3.6 Data Sets

**SEA Concepts.** In order to test the method we proposed, we needed data that changes over time e.g. concept drifts are present. We used artificial data described in [12]. Samples of data concepts with the relevant features are plotted in the figure 2(A). Each concept had 15,000 examples with about 10 % of class noise.

Description in [12] holds for our data set with one improvement. Since we wanted to re-use the old models and to study how our method works, we doubled the size by concatenating original set with another copy of it. We obtained data set with 120,000 examples and eight concepts such that concept 1 is the same like concept 5, concept 2 equals concept 6 etc. Figure 2(B) illustrates the data layout.

This way the recurrence is present and concepts are not just similar, but exactly the same. Normally we would probably never encounter such strong recurrence, but it serves well for studying the problem.

**LED Data.** Drift data stream generator from [5] with default settings was used to create another data set. Every 10,000 we change the number of attributes with drift. We generated 120,000 examples with the following randomly chosen numbers of attributes with drift: $2, 4, 3, 1, 4, 3, 2, 5, 1, 4, 3, 5$.

**Intrusion.**This data set was used in KDD Cup 1999 Competition described in [17]. The full dataset had about five million connection records, but we also used data set with only 10 % of the size finding it more illustrative.

The original task had 24 training attack types. The original labels of attack types were changed to label *abnormal* keeping the label *normal* for normal connection. This way we simplified the set to 2 class problem.

### 3.7 Using the Referee vs Simple Drift Detection

Handeling concept drifts is well-studied problem in data stream mining and many solutions could be found and used. For example by detecting the drift and building a new model while forgetting the old one, we can achieve better results. Furthermore, we can store examples in short-term memory during the warning phase and use these recent examples, potentionally representants of a new concept, for learning a new model.

In this paper we introduced a new way to improve these techniques. The idea is to store the old models instead of forgetting them and try to re-use them when there is similar underlying concept in new data.

The threshold for re-using old model was set to 70 % in all data sets. Different settings for different sets could improve the performance, but we would like to use more universal threshold. To avoid unnecessary testing when there is false alarm the method waits and if it is still in warning phase after 350 examples all unlabeled examples from the memory are tested by referees.

On the figures 3,4 (left) there are results for SEA Concept data set. As we can see re-using model from concept 1 for concept 3 lead to slightly worse results, because those two are not that similar as concept 2 and concept 4 (see figure 2), where re-using performed the same as building a new model. Moreover, re-using model from concept 2, learnt also on concept 4, on data from the sixth one lead to better results than in the case of learning new classifier. In concept 7 we can see the same situation as in concept 3.

The goal of predicting drift and recurrence was quite succesful, three out of four cases of re-using reported drift sooner. 183.5 points improvement on average considering times when drift occured. Average number of examples in warning, considering all the warning phases, decreased by 80 examples, which was improvement of 9.25 %. When model was re-used, it was immediately after those 350 examples. It would suggest lowering the number and drift could be reported even sooner, but with less examples in warning referees tended to report drift and change the model when there were false alarms.

We were interested what would be the performance if, in the case of SEA Concepts data, the referee would choose exactly the models that were learnt on the same concepts in history. It should be mentioned that this experiment was conducted just because the set was artificial and the concepts and recurrence was known. In real data sets this kind of learning is impossible to do so. We manually reused the historical concepts e.g. first model for fifth concept etc. On figure 4 (right) we can see that the performance also was not ideal. Slow continual increase in error-rate caused the warning to be reported much sooner and second model learnt from examples of fifth concept that were stored in short memory during the warning phase.

The figure 5 (left) shows the results of using proposed method on LED data. Both approaches (referee-adviced and without re-using models) are same till the concept 11, where referee choses model 6. As can be seen in data description, both concepts were generated with same number of attributes with drift, therefore the performance was better than without re-using any model. However, the model had increading tendency in error rate and warning was reported very soon. The situation was similar to the one in experiment with true models in SEA Concepts. In this case new incorrect drift was detected, model 10 was re-used which caused decrease in accuracy. The actual drift after concept 10 was detected 1022 examples sooner, but the other true drift was then delayed 301 examples.

Running our method on data from KDD Cup gave us results captured on figure 5 (right). The set tend to have structure of many different-lenght groups of consecutive instances of the same class. Drifts occured in the data because different types of attack, and therefore different protocols etc. used for the connection, were labeled with the same class, but also because normal connections had different properties. Those concepts were re-appering in the data set making it suitable for the tests.

Even though in most of the cases drifts were abrubt and quickly detected, some minor improvements were to be found.

### 3.8 Using Model Advice vs Referee Advice

In this secton we compare results obtained by our approach and those obtained by testing the accuracy of models during warning phase instead of asking referees. For re-using selection was the threshold set to approximately average performance of the models. We have to say it was not expected that referees would have better results. It is obvious that with the information about true label it is much easier to select proper model, but labels in real situations are obtained

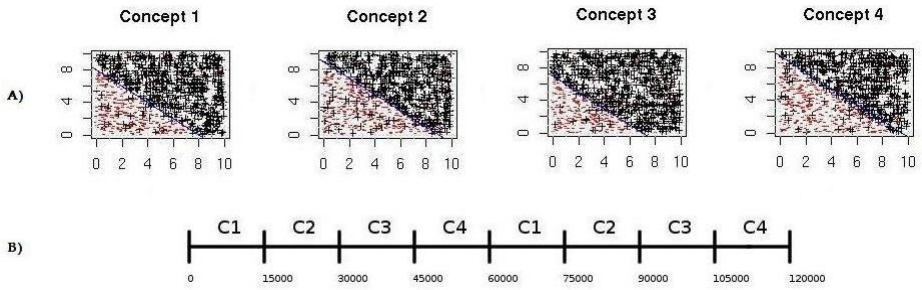


**Figure 2.** A) samples of data concepts with 1000 examples each and B) layout of concepts in experimental data

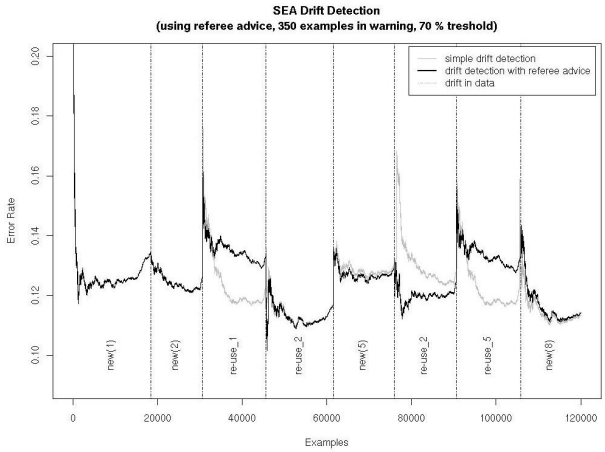


**Figure 3.** Comparison of error-rates with re-using models (70 % threshold) and without re-using on SEA Concepts

after delay while attributes of examples are available. Therefore this section shows how close or far are referee's results.

As in re-using true models in concept 5 there was warning reported too early. But this time re-using of second model was not forced so new model was learnt from warning examples combining examples from fifth and sixth concept and therefore drifted very soon. Too early warning occured also at the end of concept 7. Otherwise model-advised approach was prevailing as expected. Comparison of these two approaches is illustrated on the figure 6.

Model-advised classifier also performed better on LED data (figure 7 on the left), as expected, with the exception in seventh concept, where there was data generated with same settings as in re-used concept 1. Nevertheless, re-using caused that non-existing drift was reported and new classifier was learnt in the middle of actual concept in data.

Model-advised approach on Intrusion data set 7 (right) was not very succesful. Despite of setting the threshold to 99.5 % drifts were reported too often. It resulted in needless changes of context and therefore downgrading the overall performance. Compare 64 drifts of simple drift detection to 372 of model-advised approach.

## 4 Conclusions

In this work new method for tracking recurrent concepts was introduced. Even though there was not great improvement on overall accuracy, the method showed it could detect drift sooner with re-using older models. There are is still a lot of space for improvement and future. Other types of classifiers or ensemble of classifiers could be used and perform better either as primary classsifier or as referee. We believe that future further study of this method could bring interesting results.

## REFERENCES

[1] Gama, João and Fernandes, Ricardo and Rocha, Ricardo. Decision Trees for Mining Data Streams. Intelligent Data Analysis, IOS Press, Pages 23-45, 2006.

[2] Gama João, Medas Pedro, Gladys Castillo, and Rodrigues Pedro. Learning with Drift Detection. *Proc. of the 17th Brazilian Symposium on Artificial Intelligence (SBIA 2004)*, Volume 3171, page 286–295 - October 2004.

[3] Gama, João, Gaber, Mohamed Medhat (Eds.). Learning from Data Streams: Processing Techniques in Sensor Networks. Springer, 2007.

[4] Katakis, Ioannis, Tsoumakas, Grigorios, Vlahavas, Ioannis. An Ensemble of Classifiers for coping with Reurring Contexts in Data Streams. *18th European Conference on Artificial Intelligence*, IOS Press, Patras, Greece, 2008.

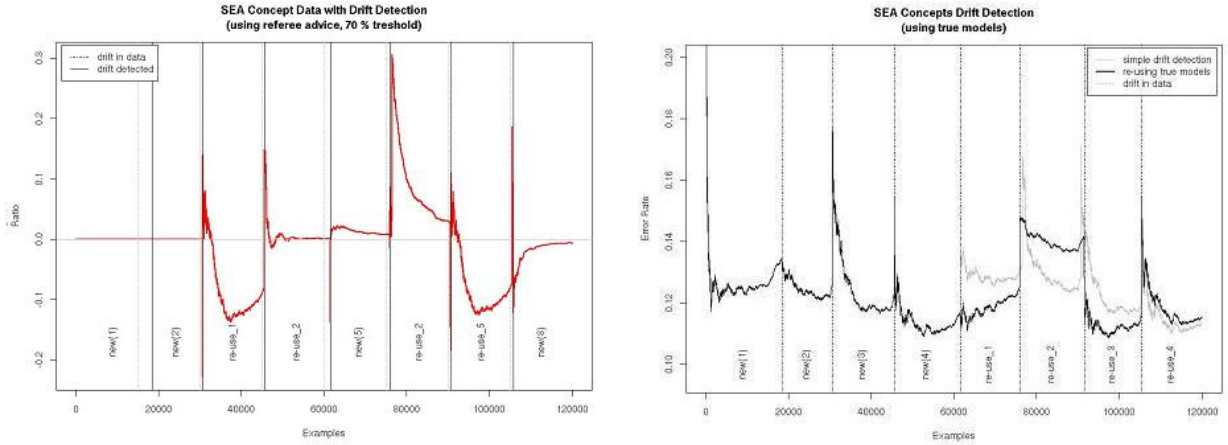


**Figure 4.** (left) Logarithm of ratio between error-rates without and with re-using models on SEA Concepts. Curve above 0 is in favor of method with referees. (right) Comparison of error-rates with re-using true models and without re-using on SEA Concepts.

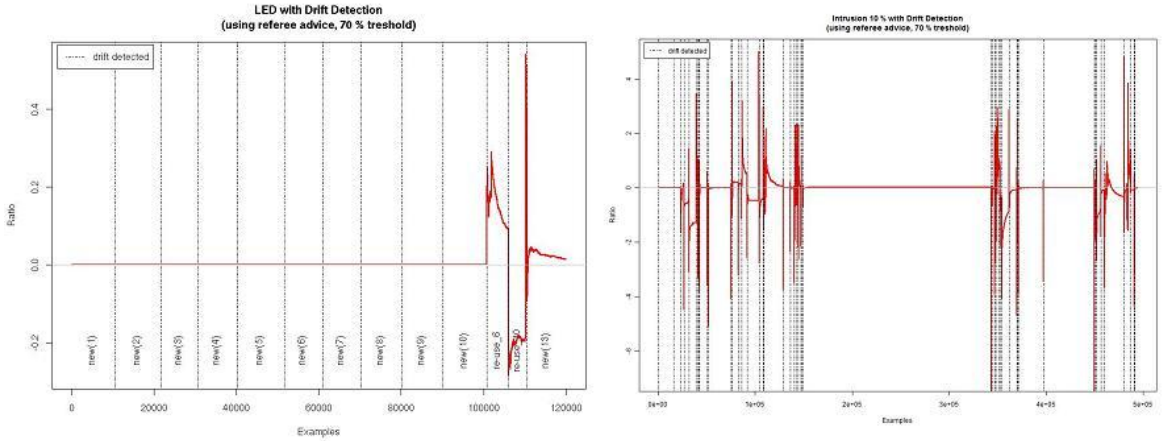


**Figure 5.** Ratio between error-rates without and with re-using models on LED data set (left) and Intrusion (right). Above 0 is in favor of referees


[5] Kirkby Richard. Massive Online Analysis. University of Waikato, Hamilton, New Zealand, 2007. http://sourceforge.net/projects/moa-datastream/

[6] Lazarescu, Mihai. A Multi-resolution Learning Approach to Tracking Concept Drift and Recurrent Concepts. PRIS, 2005.

[7] Mohamed Medhat Gaber, João Gama, Auroop R. Ganguly, Olufemi A. Omitaomu, Ranga Raju Vatsavai. Knowledge Discovery from Sensor Data. Taylor Francis, 2008.

[8] Sasthakumar Ramamurthy, Raj Bhatnagar. Tracking Recurrent Concept Drift in Streaming Data Using Ensemble Classifiers. *Proc. of the Sixth International Conference on Machine Learning and Applications*, Pages 404-409, 2007.

[9] Alexander K. Seewald and Johannes Fürnkranz. Grading Classifiers. Austrian Research Institute for Artificial Intelligence, 2001, OEFAI-TR-2001-01, Wien, Austria.

[10] Jeffrey C. Schlimmer and Richard H. Granger Jr. Incremental Learning from Noisy Data. Machine Learning 1: 317-354 1986.

[11] Stanley K.O., Learning concept drift with a committee of decision trees, Tech. Report UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, USA, 2003.

[12] W.Nick Street and YongSeog Kim. A streaming ensemble algorithm SEA for large-scale classification. In *Proc. seventh ACM SIGKDD international conference on Knowledge discovery and data mining*,pages 377-382. ACM Press, 2001.

[13] Tsymbal, Alexey. The Problem of Concept Drift: Definitions and Related Work. http://citeseer.ist.psu.edu/tsymbal04problem.html

[14] Wang, H., Fan, W., Yu, P., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: *Proc. KDD. (2003)*

[15] Witten, Ian H. and Frank, Eibe. Data Mining: Practical Machine Learning Tools and Techniques With Java Implementations. Morgan Kaufmann Publishers, 1999.

[16] Ying Yang, Xindong Wu, and Xingquan Zhu. Mining in Anticipation for Concept Change: Proactive-Reactive Prediction in Data Streams. In *the Proc. 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2005)*, pages 710-715.

[17] http://kdd.ics.uci.edu/databases/kddcup99/task.html

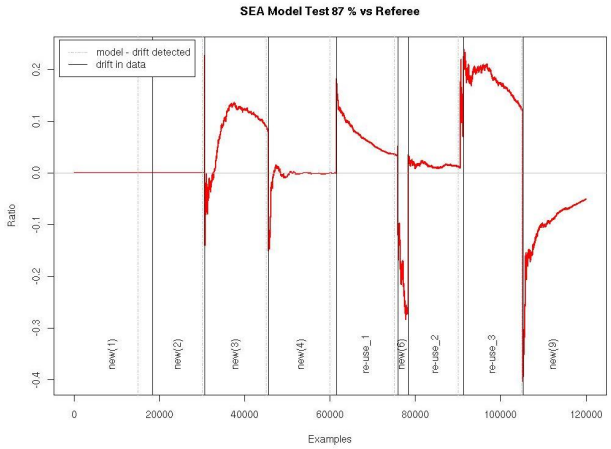


**Figure 6.** Ratio between error-rates with model-advised and referee-advised re-using on SEA Concepts - above 0 is in favor of model-advised approach

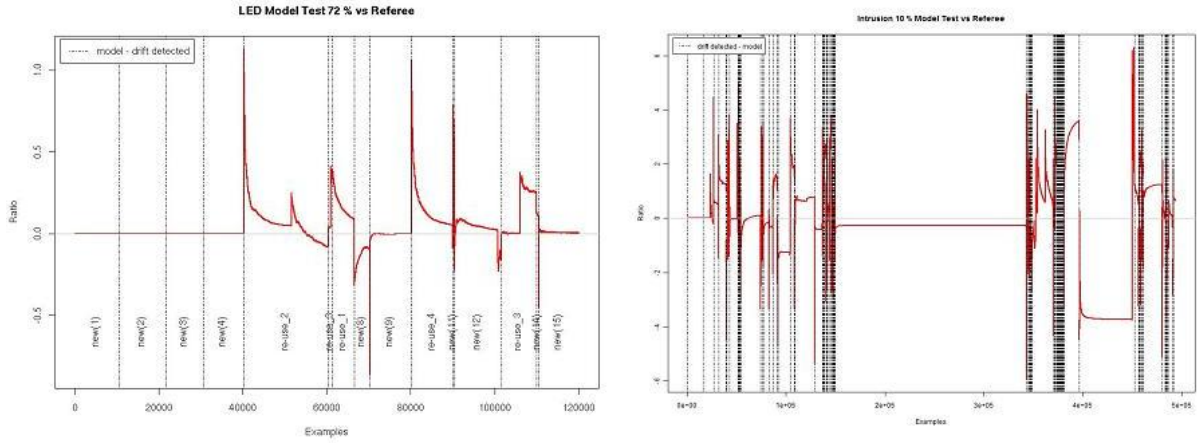


**Figure 7.** Ratio between error-rates with model-advised and referee-advised re-using on LED data (left) and on Intrusion (right) - above 0 is in favor of model-advised approach