Adaptation to Drifting Concepts

Gladys Castillo[1,2], João Gama[1,3], and Pedro Medas[1]

[1] LIACC, University of Porto, Portugal
[2] Department of Mathematics, University of Aveiro, Portugal
[3] FEP, University of Porto, Portugal
gladys@mat.ua.pt,jgama@liacc.up.pt,pmedas@liacc.up.pt

Abstract. Most of supervised learning algorithms assume the stability of the target concept over time. Nevertheless in many real-user modeling systems, where the data is collected over an extended period of time, the learning task can be complicated by changes in the distribution underlying the data. This problem is known in machine learning as *concept drift*. The main idea behind *Statistical Quality Control* is to monitor the stability of one or more quality characteristics in a production process which generally shows some variation over time. In this paper we present a method for handling concept drift based on Shewhart P-Charts in an on-line framework for supervised learning. We explore the use of two alternatives P-charts, which differ only by the way they estimate the target value to set the center line. Experiments with simulated concept drift scenarios in the context of a user modeling prediction task compare the proposed method with other adaptive approaches. The results show that, both P-Charts consistently recognize concept changes, and that the learner can adapt quickly to these changes to maintain its performance level.

1 Introduction

User modeling systems are basically concerned with making inferences about the user's assumptions (e.g. preferences, goals, interests, etc.) from observations of the user's behavior during his/her interaction with the system. On the other hand Machine Learning deals with the formation of models from observations. In recent years a growing number of applications of machine learning techniques to user modeling systems have been developed (e.g. information filtering). Observations of the user's behavior can provide data (training examples) that a machine learning system can use to induce a model designed to predict future actions [13]. Nevertheless, for many user modeling systems where data is collected over an extended period of time, the machine learning task can be complicated by changes in the distribution underlying the data. This problem is known as *concept drift* in machine learning. Depending on the rate of these changes we can distinguish *concept drift* (when changes occur gradually) of *concept shift* (when changes occur abruptly). Concept drift scenarios require on-line, incremental learning algorithms, able to adapt quickly to drifting concepts.

In the last few years several methods to cope with *concept drift* have been developed (e.g. [6, 7, 9, 14]). The goal of this paper is to consider yet another method to handle concept drift in an on-line framework for supervised learning. Our method is based on *Statistical Quality Control*. In order to detect that a change has occurred, usually, a process that monitors the value of some indicators, such as performance measures, must be implemented. The benefit of our method, compared to the other approaches, is that this monitoring process is explicitly modelled using P-charts, an attribute *Shewhart control chart*. In this paper, we explore how two alternatives P-Charts can be used to detect concept changes. These two P-Charts differ only by the way they estimate the target value to set the center line on the chart. We present a general algorithm to handle concept drift based on P-Chart, which is broadly applicable to a range of domains and learning algorithms. Experiments with simulated concept drift scenarios in the context of a student modeling task compare our method with other approaches. The results show that both P-charts consistently recognize concept changes, and that, the learner can adapt quickly to these changes in order to maintain its performance level.

In the next section, we review other work on adaptation to drifting concepts. In section 3 we introduce some notions of Statistical Quality Control, and then, explain how P-Chart can be used in the monitoring process to detect concept drift. Further, we present the general algorithm to handle concept drift based on P-Chart. In section 4 we describe a user modeling prediction task in an adaptive educational system, and in section 5 we present experiments to evaluate the proposed method in the context of this user modeling task. Finally, section 6 contains the conclusions and future work.

2 Related Work

In machine learning drifting concepts are often handled by time windows or weighted examples according to their age or utility. In general, approaches to cope with concept drift can be classified into two categories: *i*) approaches that adapt a learner at regular intervals without considering whether changes have really occurred; *ii*) approaches that first detect concept changes, and next, the learner is adapted to these changes. Examples of the former approaches are *weighted examples* and *time windows* of fixed size. Weighted examples are based on the simple idea that the importance of an example should decrease with time (references about this approach can be found in [6],[7],[8], [9],[14]). When a time window is used, at each time step the learner is induced only from the examples that are included in the window. Here, the key difficulty is how to select the appropriate window size: a small window can assure a fast adaptability in phases with concept changes but in more stable phases it can affect the learner performance, while a large window would produce good and stable learning results in stable phases but can not react quickly to concept changes. In the latter approaches,with the aim of detecting concept changes, some indicators (e.g. performance measures, properties of the data, etc.) are monitored over time

(see [6] for a good overview of these indicators). If during the monitoring process a concept drift is detected, some actions to adapt the learner to these changes can be taken. When a time window of adaptive size is used these actions usually lead to adjusting the window size according to the extent of concept drift [6]. As a general rule, if a concept drift is detected the window size decreases, otherwise the window size increases. An example of work relevant to this approach is the FLORA family of algorithms developed by Widmer and Kubat [14]. For instance, FLORA2 includes a window adjustment heuristic for a rule-based classifier. To detect concept changes the accuracy and the coverage of the current learner are monitored over time and the window size is adapted accordingly.

Other relevant works, which are served as base for this paper, are the works of R.Klinkenberg and C.Lanquillon, both of them in information filtering. For instance, Klinkenberg and Renz in [6], in order to detect concept drift, they propose monitoring the values of three performance indicators: *accuracy*, *recall* and *precision* over time, and then, comparing it to a confidence interval of standard sample errors for a moving average value (using the last M batches) of each particular indicator. Although these heuristics seem to work well in their particular domain, they have to deal with two main problems: *i*) to compute performance measures, user feedback about the true class is required, but in some real applications only partial user feedback is available; *ii*) a considerable number of parameters are needed to be tuned. Afterwards, in [7] Klinkenberg and Joachims present a theoretically well-founded method to recognize and handle concept changes using support vector machines. The key idea is to select the window size so that the estimated generalization error on new examples is minimized. This approach uses unlabeled data to reduce the need for labeled data, it doesn't require complicated parameterization and it works effectively and efficiently in practice. However, it is not independent of the hypothesis language (a support vector machine) and therefore it is not generally applicable.

On the other hand, Lanquillon [8] employs *Statistical Quality Control* to detect changes in document stream with either little or no user feedback. Three alternative performance measures: *sample error rate* (it requires only some user feedback per batch), *expected error rate* and *virtual rejects* (the two last measures don't require any user feedback) are monitored over time. A representative training set is maintained through storage of new examples for which the true class labels have been provided by the user. If the monitor has detected some change, the filtering system is adapted based on the current training set by running through the entire learning process from scratch.

3 Exploring the use of two P-Charts to cope with drifting concepts

Similarly to Lanquillon, the underlying theory we propose to use to deal with concept drift is *Statistical Quality Control*. In the section following we will introduce some notions of this theory (a deeper discussion can be found in [12]).

3.1 Notions of Statistical Quality Control

The main idea behind *Statistical Quality Control* is to monitor the stability of one or more quality characteristics in production processes [2]. The values of the quality characteristic generally show some variation, which can be caused by either some "*natural causes*" inherent in the production process or by some "*special causes*" that can be traced to a particular problem. "*Natural causes*" are presented all the time while "*special causes*" occur at unpredictable times. A process can be run in either of two mutually exclusive states: an *in-control* state or an *out-of-control* state. An *in-control* state means that the successive values of the quality characteristic, as they are observed over time, show a stable random variation about a target value (variations caused by "*natural causes*"). Otherwise a process is *out-of-control*. A process is in statistical control if "*special causes*" have been detected and removed, so these sources of variability will not influence the process in the future [2].

The Shewhart controls charts are a useful tool to distinguish whether a process is *in-control* or *out-of-control*. The values of the quality characteristic are plotted on the chart in time order and connected by a line. Some *control limits* are established. If a value falls outside the control limits, it is assumed that the process is *out-of-control*, i.e.,some "*special causes*" have shifted the process off target, and therefore, some actions will be required to remove them. If the distribution of the quality characteristic is Normal (or approximately Normal) there is some statistical arguments for using the 3 *sigma control limits* (*sigma* is the standard deviation around the mean). It is well-known that, if the distribution of a statistic is Normal,then approximately 99.7% of the observations will fall within three *standard deviations* of the *mean* of the statistic. In addition to control limits, we can also use *warning limits*. These limits are usually set a bit closer to the mean than the control limits. For instance, if two consecutive values fall outside the *warning limits* some actions can also be taken.

If the mean μ and the standard deviation σ of the statistic of interest (the values of the quality characteristic) are known, then these values are used to set up the parameters of the control chart, as follows:

$$CL = \mu, \tag{1}$$

$$LCL = \mu - 3\sigma; UCL = \mu + 3\sigma, \tag{2}$$

$$LWL = \mu - k\sigma; UWL = \mu + k\sigma; 0 < k < 3. \tag{3}$$

where CL represents the center line, LCL and UCL - the upper and lower control limits, and LWL and UWL - the upper and lower warning limits. However, in most cases μ and σ are unknown and these values must be estimated from previously observed data.

The control charts are often classified according to the type of quality characteristic that they monitor: *variables* or *attributes*. Attribute data is also known as *count attribute*. For the purpose of this paper, we focus on the P-Chart - a control chart for the *proportion nonconforming* (the ratio of the number of *nonconforming* items in a population to the total number of items in that population) where: *i*) a dichotomous attribute with only two mutually exclusive and

exhaustive outcomes is measured (e.g. each unit produced is classified either *conforming* or *nonconforming* to some specifications); *ii*) the successive observations are independent over time; *iii*) for a random sample of n items the count of units that are *nonconforming* is registered; *iv*) the quality characteristic to be monitored is the sample proportion *nonconforming*; *v*) the sample size can vary.

The count of *nonconforming* items follows a Binomial distribution with parameters n and p, where p is the probability that any unit will *nonconforming* (the population proportion). The distribution of the sample proportion *nonconforming* can also be obtained from the binomial with parameters:

$$\mu = p \ ; \ \sigma = \sqrt{\frac{p(1-p)}{n}} \tag{4}$$

Moreover if the sample sizes are large ($n \geq 30$), the *sample proportion nonconforming* is approximately Normal (a well-known result derived by the Central Limit Theorem). As we have stated before, when p is not known, then it must be estimated from observed data. Suppose that the estimate $\hat{p}$ of p is obtained from previous data by some estimator. Then, from equations (1)–(4), the parameters of the P-Chart for each individual t-th sample with size n_t would be:

$$CL = \hat{p} \ , \tag{5}$$

$$UCL = \hat{p} + 3\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}}; LCL = max\left\{0, \hat{p} - 3\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}}\right\}, \tag{6}$$

$$UWL = \hat{p} + k\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}}; LWL = max\left\{0, \hat{p} - k\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}}\right\}, 0 < k < 3 \tag{7}$$

The usual procedure to compute the estimate $\hat{p}$ is by the weighted average of m preliminary sample proportions (as a rule, m should be 20 or 25)

$$\hat{p} = \overline{p} = \frac{1}{\sum_{i=1}^{m} n_i} \sum_{i=1}^{m} n_i p_i = \frac{1}{\sum_{i=1}^{m} n_i} \sum_{i=1}^{m} (count)_i \tag{8}$$

where p_i is the sample proportion of the i-th sample with size n_i. Further we call the estimate $\hat{p}$ the target value.

3.2 Using P-Chart for Detecting Concept Drift

In this section we explore the use of two alternatives P-Charts for detecting concept drift in an on-line framework for supervised learning. Data arrives to the learner over time in batches. For each batch, the examples are classified using the current learner. The quality characteristic to be monitored is the *sample error rate*, a sample proportion of the misclassified examples. Following is a more formal definition of the sample error rate:

Definition 1. *The* sample error rate *of a learner $h_{\mathcal{L}}$ with respect to target concept f and sample D with n examples is the proportion of misclassified examples by $h_{\mathcal{L}}$, i.e.,*

$$Err(D, h_{\mathcal{L}}) \equiv error(D, f, h_{\mathcal{L}}) = \frac{1}{n} \sum_{x \in D} \delta(f(x), h_{\mathcal{L}}(x)) \tag{9}$$

where $\delta(f(x), h_{\mathcal{L}}(x)) = \begin{cases} 1, when f(x) \neq h_{\mathcal{L}}(x) \\ 0, otherwise \end{cases}$ *is the* one-zero loss *function.*

In order to evaluate the *sample error rate* for each batch, user feedback about the correct class for the examples is required. If the monitoring process detects concept drift, the learner must be adapted accordingly. Next, the adapted learner can be used to predict the class labels of the examples of the next batch.

In classical *Shewhart control charts* it is assumed that the successive sample proportions should exhibit a stable random variation around the target value over time. Such behavior is not observed in the learning tasks where, we know, concept changes are likely to occur. It is well known that the learner's goal is to minimize the *zero-one loss* function. Consequently, while the learner has not learned enough about the underlying target concept, the error rate should exhibit a downward trend that reflects the desired improvement of the performance. At once a concept change occurred, an opposite trend in the error rate is immediately observed. In principle, a learning process is *in-control* only when the learner is extremely stable. Therefore, in learning tasks we need to dynamically estimate the target value taking into account the actual performance level of the learner.

The two alternatives P-Charts, we propose to use, differ only by the way they estimate the target value. To distinguish them, we denote one chart PAvg-chart and the other PMin-chart. For the PAvg-chart at each time t the target value $\hat{p}$ is estimated by the weighted average of the sample errors on the M previous batches, i.e., the estimate $\hat{p}$ can be computed from equation (8) for $i = t - M$ to $t - 1$. Here M is a required parameter that must be tuned. Similarly, Lanquillon in [8] propose to estimate the target value by the weighted average of the sample errors on recent batches only if they are within the warning limits of the chart.

The way in which we estimate the target value of the Pmin-chart is based on the exposed facts related to the dynamic behavior of the learning task and also on the method to detect concept drift presented in [10]. Let us introduce the following definition:

Definition 2. *A* context *S corresponds to a set of examples from the data stream where the distribution underlying the examples is stationary (without drifts)*

In general, the problem of handling drifting concepts can be viewed as the problem of the detection of the last moment when a concept drift occurred. Thus, the data stream can be analyzed as a sequence of different contexts over time, i.e. the detection and extraction of stable concepts between drifts. Suppose

that at time t a new context begins to be processed. At the beginning, while the learner has not learned enough, the error rate for this context should exhibit a downward trend. This means that for the current context, all the time when a lower error rate is achieved, the learner will try to improve, or at least, to maintain its performance level. Based on these facts, we propose to maintain a *minimum value* for the error rate for the current context and set the target value to this minimum value instead of using some average of previous observed values. Taking into account the way we estimate the target value, we can state that PMin-chart is not a typical statistical P-chart since it does not use a statistical well-founded estimator (we will try to explore these issues in future works).

Suppose that $Err_S^{(t)}$ is the error rate for the context $S^{(t)}$ at time t and $SErr_S^{(t)}$ its standard deviation. From equations(9),(4) these values can be computed by:

$$Err_S^{(t)} \equiv Err(S^{(t)}, h_{\mathcal{L}});\ SErr_S^{(t)} = \sqrt{\frac{Err_S^{(t)}(1 - Err_S^{(t)})}{n_S^{(t)}}} \tag{10}$$

where $n_S^{(t)}$ is the number of examples of the actual context S at time t. Let Err_{min} denote the *minimum-error rate*. Initially, Err_{min} is set to some predefined value (a big number). Next, at each time step, if $Err_S^{(t)} + SErr_S^{(t)} < Err_{min}$ then Err_{min} is set to $Err_S^{(t)}$.

3.3 A general algorithm for handling concept drift in on-line supervised learning based on control P-chart

We have developed a general algorithm for handling *concept drift* in an on-line framework for supervised learning based on P-Chart. This is presented in Figure 1. In each time step, the algorithm begins by determining the *sample error rate* for the current batch. Next, the target value is estimated by the *mean_estimator* procedure (how it is estimated depends on the method that is used: weighted average, minimum error, etc.). After estimating the target value, all the chart parameters are computed by the equations (5)–(7). Since a low sample error rate is desirable, we don't need to use the low limits here. If the current sample error Err_t is above the upper control limit, a *concept shift* is suspected, and it is assumed that a new context is beginning. In this case, only the examples from this new context are used to re-learn the learner, thus forgetting all the previous data. If the last alert occurred at the previous time step (LastAlert=t-1), we assume, that the new context began at the time indicated in FirstAlert. If the current sample error is above the upper warning limit and it occurred at two or more consecutive times a *concept drift* is suspected. In this case, the examples of the current batch are not used to update the learner (it allows that the monitoring process will more quickly recognize a concept shift). If neither, a *concept shift* or *concept drift* is suspected, the learner is updated to combine the current learner with the examples of the current batch.

The precise way in which a learner can be updated in order to include new data depends basically on the learning algorithm employed. In principle, there

are two main approaches: *i*) re-build the learner from scratch; *ii*) update the learner combining the current model with the new data. For instance, updating a Naïve Bayes classifier is simple: the counters required for calculating the prior probabilities can be increased as new examples arrives. For other learners, updating can be more difficult (e.g. support vector machines). May be, in this case it would be easier to relearn from scratch. A deeper discussion about these issues can be found in [8].

```
procedure HandleConceptDriftWithPChart
                 (data,learner,k,mean_estimator())

 for t=1 to N  //for each batch with size n_t at time t
  Errt:=Err(Batch_t,Learner);
  CL:= mean_estimator();
  Sigma:=sqrt(CL*(1-CL)/n_t);
  UCL:=CL+3.Sigma; WCL:=CL+k.Sigma;
  If Errt > UCL then         /* concept shift suspected
  {If LastAlert=t-1 then t_ini:=FirstAlert else t_ini:=t;
     learner:=ReLearnFrom(learner,t_ini)}
  else
   If Errt > WCL then        /* concept drift suspected
    If LastAlert=t-1 then    /* consecutive alerts
      LastAlert:=t
      else                  /* it can be a false alarm
        {learner:= UpdateWith(learner,Batch_t)
         FirstAlert:=t, LastAlert:=t}
    else                    /* no changes was detected
      learner:= UpdateWith(learner,Batch_t);
 Next t;
 return: learner
End
```

Fig. 1. General algorithm for handling concept drift using P-Chart.

4 The user modeling prediction task

The proposed algorithm to handle concept drift was tested for a user modeling prediction task in the context of GIAS, an adaptive authoring tool to support learning and teaching (see [1] for more details). In GIAS, the authors (teachers) can define a course and associate to each course topic a set of existing online learning resources. Whenever a student requests the learning resources of a selected topic, a topic generator must decide which resources are '*appropriate*' or

'*not appropriate*' for the student, thus partitioning the set of available resources into these two classes. The choice of the appropriate set of resources for a particular student depends on the *resource's characteristics* and on the student's *cognitive state*,*learning style* and *preferences*.

Learning style can be defined as the different ways a person collects, processes and organizes information. This kind of information helps more effectively adaptive learning systems, to decide how to adapt its navigation and its presentation, thus enhancing the student learning. On the other hand a *learning resource* can be viewed as the implementation of a learning activity in a multimedia support. By matching a *learning style* with the characteristics of the *learning resources*, in principle, it is possible to determine what types of resources are more appropriate to a particular student. Nevertheless, it is a fact that the student preferences of certain types of multimedia resources or learning activities can change over time. Since, an adaptive learning model is desirable. Therefore, the prediction task that consists in determining whether a learning resource is or is not appropriate for a particular student taking into account his/her learning style and preferences and the resource's characteristics, can be related with the *concept drift problem* for a concept learning task. Moreover, in some aspects, this task is related to the task of information filtering.

We use the Felder-Sylverman model [4] of learning style which classifies students in five dimensions: *visual/verbal, sensing/intuitive, sequential/global, inductive/ deductive, active/reflective* (we use only the first three dimensions). In order to acquire the initial learning style we employ the Index of Learning Styles Questionnaire (ILSQ) [3]. It helps to classify the preference for one or the other category in each dimension as *mild*, *moderate* or *strong*.

In our learning task, the examples are described through 5 attributes: the first three characterizing the *student's learning style* and the last two characterizing the *learning resource*. The possible values for each attribute are presented in the Table 1.

Table 1. Establishing attributes and their possible values

Attribute	Values
Characterizing the student's learning style	
VisualVerbal	$VVi, VV \in \{Visual, Verbal\}, i \in \{mild, moderate, strong\}$
SensingConceptual	$SCi, SC \in \{Sensing, Conceptual\}, i \in \{mild, moder., strong\}$
GlobalSequential	$GSi, GS \in \{Global, Sequential\}, i \in \{mild, moderate, strong\}$
Characterizing the learning resource	
Learning Activity	Lesson objectives/Explanation/Example/Conceptual Map /Synthesis Diagram/Glossary /Summary /Bibliography/ Historical Review /Inter.Activity
Resource Type	Text/HTML Text/Picture/Animated Picture/ Animated Picture with Voice/ Audio /Video /Software

For instance, suppose the following example:

VisualVerbal: *Verbalmoderate*; SensingConceptual:*Sensingmild*; GlobalSequential:*Globalmild*; Learning Activity:*explanation*; Resource Type :*audio*.

The induced learner must predict if a learning resource implementing a learning activity such as '*explanation*' in a multimedia support of type '*audio*' would be appropriate for a student with a *moderate* preference for VERBAL category, a *mild* preference for SENSING category and a *mild* preference for a GLOBAL category.

For each student an *individual predictive model* is maintained. First, the model is initialized from some initial training data taking into account the acquired information about the student's learning style. Whenever a student selects a topic, the student's current predictive model is used to classify the available resources.

We choose the Naïve Bayes (NB) classifier, one of the learning algorithms most used in user modeling, as our predictive model because: *i*) it is simple; *ii*) it learns quickly (it doesn't require large amount of data to learn); *iii*) low computations to make decisions are needed; *iv*)its results as probabilities are easy to apply. Moreover, we propose to employ Adaptive Bayes [5], an adaptive version of the Naïve Bayes. The main difference between these two algorithms is that Adaptive Bayes includes an updating scheme, that makes it possible to better fit the current model to new data: after seeing each example, first, the counters are incremented, and then, they are again updated in order to increase the confidence on the correct class (the amount of adjustment is proportional to the discrepancy between the predicted class and the correct class).

Since the NB classifier returns probabilities, all the resources of a same class can be ranked. As a result, a page is sent to the student including two separated ranked lists with the resource's links: a "*resources suggested for study*" list with the links for those resources classified as '*appropriate*' and "*other resources for study*" list with the links for those resources classified as '*not appropriate*'. Whenever possible, the correct class is obtained based on the observations about the user's choice of links: visited links are taken as positive examples. Obtaining a relevant set of negative examples is more difficult. To obtain more examples we suggest to the students to rate the resources explicitly. The obtained examples are used to evaluate the sample error and update the predictive model.

5 Experiments

In order to test the two P-Charts proposed in section 3 we have conducted experiments simulating *concept drift* scenarios in the context of the described prediction task using artificial datasets.

5.1 Dataset Generation and Experimental Setup

The artificial datasets were generated to simulate the changes in the user's preferences, which can conduce to further adjustments in the initial learning style.

To simplify the experiments we don't discriminate the preferences for a learning style category. Hence, the number of different learning styles is equal to 2^3, which corresponds to the number of different datasets evaluated for each algorithm. Note, that the underlying concept is different for each learning style. The basic idea enclosing in the simulation of *concept drift* is based on the following facts that really exist in this learning task: for instance, suppose that a student is initially classified as VERBAL. Learning resources that match with a verbal learner (e.g. a learning resource that implements a learning activity such as "*Historical Review*" in a "*HTML Text*" support) should be appropriate for this verbal student. Hence, the underlying target concept can be represented by the following logical rule:

```
IF LearningStyle Is Verbal AND
 (ResourceLearningActivity OR ResourceType) matches Verbal
 THEN Resource is Appropriate
```

Nevertheless, during the further interaction with the system, the student can change his/her preferences for another kind of learning resource that no longer matches with his/her learning style. This means that the underlying concept has changed and, consequently, the previous rule can be replaced with another one, like this:

```
IF LearningStyle Is Verbal AND
  (ResourceLearningActivity OR ResourceType) matches Visual
 THEN Resource is Appropriate
```

Moreover, these changes in the student's preferences lead to further adjustments in the student learning style, i.e., the underlying data distribution can also change. Thus, to simulate concept drift scenarios, for each learning style, datasets with 1600 examples were generated randomly. Each example was classified according to the current concept, which changes after every 400 examples (a sequence of four logical rules was defined). All the examples were grouped into 32 batches of equal size with 50 examples each. Moreover, in order to initialize the learner, a training dataset with 200 examples according to the first concept was also generated.

The experiments were performed according to the on-line framework for supervised learning described in the section 3.2. The two learning algorithms: Naïve Bayes (NB) and Adaptive Bayes (AB) were evaluated in combination with each of the following approaches: a non-adaptive approach (the *baseline* approach), Fixed Size Window (WFS) and the HandleConceptDriftWithPChart algorithm described in the Figure 1 (the parameter k is set to 1) using PAVG-chart (we denote this approach PAvg) and using PMIN-chart (we denote this approach PMin).

5.2 Experimental Results and Analysis

In Figure 2 you can see an illustration of the two P-charts. In each time step, the chart lines are adjusted according to the method employed to estimate the

target value and to set the center line. Both charts detected the three concept shifts that really are in the data. The two first concept shifts (after $t = 8$ and after $t = 16$) were detected immediately by the two P-charts (as you can see, the point representing the sample error fall above the current control limit). The third concept shift (after $t = 24$) was also detected immediately by PAvg-Chart, while it was detected with a little delay by PMin-Chart. However, beginning at $t = 25$, this chart started signaling a concept drift (the points that fall outside the warning limits). This means, that an upward trend of the sample error was detected. When further, at $t = 27$ the concept shift was detected, all the examples beginning at $t = 25$ are considered to belong to the same context and consequently they are all used to re-learn the learner.

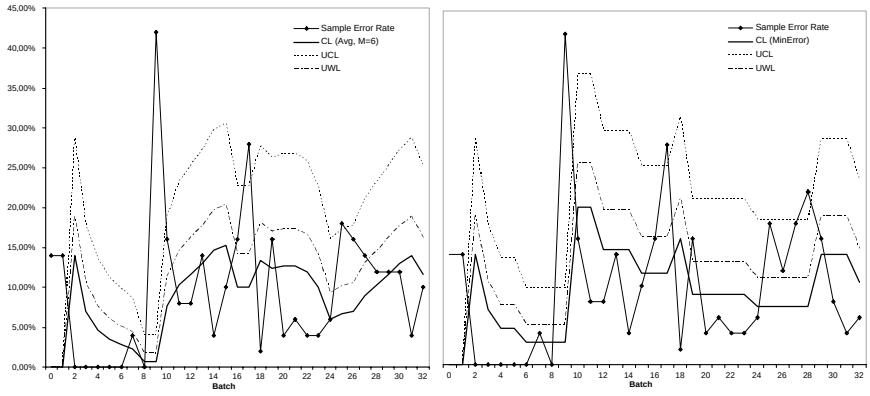

Fig. 2. The PAvg-Chart(left) and PMin-Chart(right)

Table 2 shows the accuracy of all combinations of learning algorithms and the different approaches averaged over 10 runs for each learning style. The results shown in column "Acc. Avg" were obtained by averaging over the accuracy of the eight learning styles for each approach. These averaged values are used to construct the learning curves on the Figure 3. As you can see, at first, while there is no concept drift, the performance of all approaches is good enough; however, those approaches that use Adaptive Bayes show a better performance. After the first change has occurred (after $t = 8$), the performance of those approaches without concept drift detection, decreases significantly. Since the fixed size window approach re-learns regularly from the last six batches, it can recover its performance a little, but this adaptive approach could not outperform the P-Chart approaches. Moreover, as you can notice, there are no significant differences on the performance between PAvg and PMin. Both approaches work well and quickly react to concept drift. However, the results that we show for the PAvg in the Figure 3 are the results that we obtained for $M = 6$. Table 3 com-

Table 2. Accuracy of the Naïve Bayes and Adaptive Bayes combined with all the explored approaches for the 8 learning styles ($M = 6$ for WFS and PAvg M=6)

	Approaches	LS1	LS2	LS3	LS4	LS5	LS6	LS7	LS8	Acc. Avg
(1)	NB	70.50	64.42	69.59	64.71	77.33	77.27	57.83	60.51	67.77
(2)	NB & WFS	86.86	85.11	83.06	78.81	87.48	85.98	86.66	84.48	84.80
(3)	NB & PAvg	91.52	90.96	91.38	89.37	90.49	88.51	91.14	87.56	90.11
(4)	NB & PMin	91.41	90.96	91.24	89.15	90.27	90.03	90.25	89.30	90.04
(5)	AB	75.73	70.79	73.42	67.01	80.61	80.09	70.38	68.24	73.28
(6)	AB & WFS	89.04	87.39	86.69	82.01	89.33	88.43	89.06	87.12	87.38
(7)	AB & PAvg	91.85	91.81	92.86	90.90	92.14	89.76	92.79	90.19	91.52
(8)	AB & PMin	91.90	91.61	92.62	90.87	91.77	90.60	91.84	89.30	91.31
I	(2) vs. (1)	+16.3	+20.69	+13.46	+14.11	+10.15	+8.71	+28.83	+23.96	+17.03
	(3) vs. (1)	+21.02	+26.54	+21.79	+24.66	+13.16	+11.24	+33.31	+27.04	+22.35
	(4) vs. (1)	+20.91	+26.54	+21.65	+24.44	+12.94	+12.76	+32.43	+28.53	+22.27
II	(6) vs (5)	+13.31	+20.17	+17.96	+22.36	+9.88	+8.42	+20.76	+19.31	+16.52
	(7) vs (5)	+16.12	+21.03	+19.26	+23.89	+11.53	+9.67	+22.42	+21.95	+18.23
	(8) vs (5)	+16.17	+20.83	+19.20	+23.86	+11.16	+10.51	+21.47	+21.06	+18.03
III	(4) vs. (3)	-0.11	0.0	-0.14	-0.22	-0.22	+1.52	-0.89	+0.52	-0.07
	(8) vs (7)	+0.05	-020	-0.06	+0.03	-0.38	+0.84	-0.95	-0.89	-0.20
IV	(5) vs. (1)	+5.23	+6.37	+3.83	+2.30	+3.29	+2.83	+12.55	+7.73	+ 5.51
	(6) vs. (2)	+2.19	+2.28	+3.64	+3.19	+1.85	+2.45	+2.41	+2.64	+2.58
	(7) vs. (3)	+0.33	+0.86	+1.30	+1.53	+1.66	+1.25	+1.66	+2.64	+1.40
	(8) vs. (4)	+0.49	+0.66	+1.37	+1.72	+1.50	+0.57	+1.59	+2.26	+1.27

pares accuracy for different values of M. The results show that the performance is affected by the variation of the parameter M. If the parameter M is tuned accordingly, there are no significant differences on the performance of these two P-charts. Therefore, we suggest the use of the PMin instead of the PAvg because: *i*) PMin doesn't depend on any parameter;*ii*) PMin better reflects the behaviour of the learning process.

Finally, in the last lines of the Table 2, some comparative studies of the learner performance for a pair of approaches are presented. Studies I and II compare adaptive approaches to deal with concept drift against the baseline non-adaptive approach in combination with Naïve Bayes and Adaptive Bayes, respectively. The results show that a significant improvement is achieved by using any adaptive method instead of the non-adaptive one for both the learning algorithms. However, the gain obtained by using the P-chart methods is superior to the gain obtained by using windows of fixed size. In the latter approach the learner is adapted regularly without considering whether a concept changes has really occurred. Moreover, a more significant improvement is achieved with the Naïve Bayes due to the adaptation scheme included into Adaptive Bayes. The study III compares the performance of the PAvg against PMin: the "(4) vs. (3)" line shows the performance increase obtained by using Naïve Bayes with PMin instead the PAvg, while "(8) vs. (7)" shows the performance increase obtained by

using Adaptive Bayes with Pmin instead the Pavg. As we have stated above, if M is set to 6 there are no significant differences on the performance of these two methods. The last study IV compares the two learning algorithms. The results show that Adaptive Bayes outperforms Naïve Bayes for all the approaches. In general, a more significant improvement is achieved when adaptive methods are combined with Adaptive Bayes.

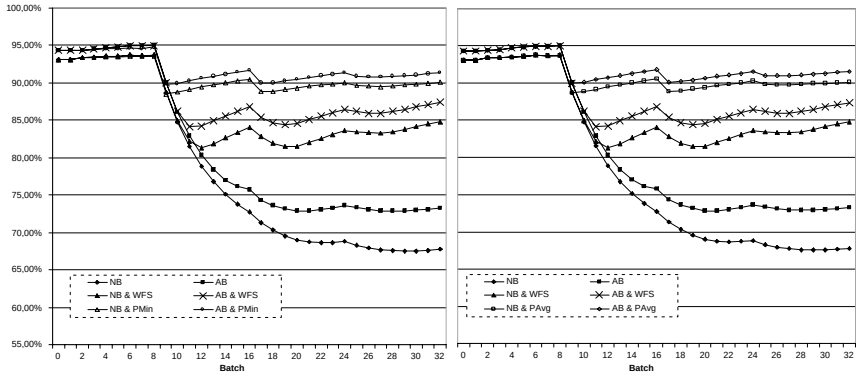

Fig. 3. Comparison of the accuracy using P-Min and P-Avg with M=6

6 Conclusions and Future Work

This paper describes yet another method to handle *concept drift* in an on-line framework for supervised learning based on *Statistical Quality Control*. We present a general algorithm to handle concept drift using P-Charts, which is broadly applicable to a range of domains and learning algorithms. The benefit of our method, compared to the other approaches, is that the monitoring process is explicitly modeled using P-charts. We explore how two alternative P-Charts: PAvg-chart and PMin-chart can be used to monitor the sample error rate in order to detect concept changes. These P-charts differ only by the way they estimate the target value to set the center line on the chart. The experimental results in the context of a user modeling prediction task using Naïve Bayes show that both P-charts consistently recognize concept changes, and that, in general, the proposed method allows the learner to adapt quickly to these changes in order to maintain its performance level. However, for purpose of estimation of the target value it is more convenient to consider PMin than PAvg because: *i*) PMin doesn't require any parameter to be tuned; *ii*) since the learner's goal is to minimize the one-loss function, PMin better follows the natural behaviour of the learning process. In future works we plan to test the proposed method with other concept drift scenarios and other learning algorithms.

Table 3. Varying the parameter M of the P-Avg method and its effect on the performance

	NB	AB
PMin	90.04	91.24
PAvg $M = 4$	90.17	91.57
$M = 6$	90.11	91.52
$M = 8$	89.92	91.08
$M = 10$	87.61	89.29

Acknowledgments:

Gratitude is expressed to the financial support given by the ALES project (POSI/39770/SRI/2001).

References

1. Castillo, G., Gama, J., Breda, A.M.: Adaptive Bayes for a User Modeling Prediction Task based on Learning Styles. In P. Brusilovsky, A. Corbett and F. de Rosis (Eds.). User Modeling 2003. Proceedings of the Ninth International Conference. LNAI **2702**, Springer-Verlag (2003) 328–332.
2. del Castillo, E.: Statistical Process Adjustment for Quality Control, John Willey and Sons, Inc., New York (2002).
3. Felder, R.M., Soloman, B.A.: Index of Learning Style Questionnaire, available online at: http://www2.ncsu.edu/unity/lockers/users/f/felder/public/ ILSdir/ilsweb.html.
4. R.M.: Matters of Style. ASEE Prism **6 (4)**(1996) 18–23.
5. Gama, J., Castillo, G.: Adaptive Bayes. Advances in Artificial Intelligence - IBERAMIA 2002, LNAI **2527**, Springer Verlag (2002) 765–774.
6. Klinkenberg, R., Renz, I., Adaptive Information Filtering: Learning in the Presence of Concept Drifts, Learning for Text Categorization, Menlo Park, CA, USA, AAAI Press (1998) 33–40.
7. Klinkenberg, R., Joachims, T.: Detecting Concept Drift with Support Vector Machines, Proceedings of the Seventeenth International Conference on Machine Learning (ICML), San Franciso, Morgan Kaufman (2000).
8. Lanquillon, C.: Enhancing Test Classification to Improve Information Filtering, PhD. Dissertation (2001), University of Madgdeburg, Germany, available on-line at: http://diglib.uni-magdeburg.de/Dissertationen/2001/carlanquillon.pdf
9. Maloof, M., Michalski, R.: Selecting Examples for Partial Memory Learning, Machine Learning **41** (2000) 27–52.
10. Medas, P., Gama, J.: Aprendizagem com Detecção de Mudança de Conceito, Actas das X Jornadas de Classificação e Análise de Dados, Aveiro (2003) 27–31.
11. Mitchell, Tom. Machine Learning. McGraw Hill, (1997).
12. Montgomery, D.C.: Introduction to Statistical Quality Control (3rd ed.), John Willey & Sons, Inc., New York (1997).
13. Webb, G., Pazzani, M., Billsus, D.: Machine Learning for User Modeling. In User Modeling and User-Adapted Interaction **11** (2001) 19–29.
14. Widmer, G., Kubat, M.: Learning in the Presence of Concept Drift and Hidden Context. Machine Learning **23** (1996) 69–101.