An Adaptive Predictive Model for Student Modeling

Gladys Castillo[1,2], João Gama[2] , Ana M. Breda[1]

[1] Departament of Mathemathics/ R&D Unit "Mathematics and Applications
University of Aveiro, 3810, Aveiro, Portugal
email: gladys {ambreda@mat.ua.pt}
[2] LIACC – University of Porto, Rua Campo Alegre, 823, 4150 Porto, Portugal
jgama@liacc.up.pt

Abstract. This chapter presents an adaptive predictive model for a student modeling prediction task in the context of an Adaptive Educational Hypermedia System (AEHS). The task, that consists in determining what kind of learning resources are more appropriate to a particular learning style, presents two issues that are critical. The first is related to the *uncertainty* of the information about the student's learning style acquired by psychometric instruments. The second is related to the changes over time of the student's preferences (*concept drift*). To approach this task, we propose a probabilistic adaptive predictive model that includes a method to handle concept drift based on Statistical Quality Control. We claim that our approach is able to adapt quickly to changes in the student's preferences and that it should be successfully used in similar user modeling prediction tasks, where *uncertainty* and *concept drift* are presented.

1 Introduction

In the last decade we have attended to an increased development of **A**daptive **E**ducational **H**ypermedia and web-based **S**ystems (AEHS)[1]. An AEHS is able to adapt its contents and presentations to specific characteristics of students. The keys for adaptation are the *domain model* and the *student model*. The former represents the knowledge about the subjects to be learned and serves as the base for structuring the hypermedia contents. The latter stores different assumptions about the student (e.g. knowledge, preferences, goals, etc). An AHES uses the information stored in both models to implement adaptive algorithms and techniques.

Student modeling involves the construction and updating of the student model. Traditionally, most of student modeling systems have been limited to maintain assumptions related with the student's knowledge, which can be acquired during evaluation activities. However, over the last years there has been an augmented interest in modeling other kind of assumptions about the student, such as the learning style and preferences. An AEHS can make use of this kind of information to decide more effectively how to adapt itself to each student individually.

[1] A extended discussion of Adaptive Hypermedia, and in particular, AEHS can be found in (Brusilovsky, 2001)

Usually the students' learning style is acquired using one of existing psychometric instruments. By matching a *learning style* with some relevant characteristics of the *learning resources*, these systems can determine which resources are most appropriate for a particular student. As a rule, the acquired assumptions about the students' learning style are no longer updated during their interactions with the system. Moreover, the deterministic rules included in their decision models also never change.

There are some typical issues that are critical concerning a successful implementation of the prediction task, that consists in determining what kind of *learning resources* are more appropriate to a particular *learning style* in a real student modeling scenario:

1) Although multiple models to categorize the students according to their learning styles have been developed it is difficult to determine how exactly a person learns. Therefore, the information about the student's learning style acquired by psychometric instruments encloses some grade of *uncertainty*.

2) During the interactions with the system, the student can change his/her preferences for another kind of learning resource that no longer matches with his/her determined learning style. It is because either the acquired learning style information needs to be adjusted or the student simply changes his/her preferences motivated by other unknown reasons (some hidden contexts). This kind of problem is known as *concept drift* in the machine learning community.

This chapter aims at presenting a new adaptive machine learning approach for the described prediction task. Our approach is based on an adaptive predictive model capable of fine-tuning its parameters to reflect more accurately the student's preferences. Moreover, this also includes a method to handle *concept drift* (Castillo, Gama and Medas, 2003). This method uses a P-Chart (an attribute Shewhart control chart[2]) to monitor the learner's performance over time. Although this drift-detection method is broadly applicable to a range of domains and learning algorithms, we choose Naïve Bayes classifier (Mitchell, 1997), one of the most used learning algorithms in user modelling, as our predictive model. Furthermore, we propose the use of Adaptive Bayes (Gama and Castillo, 2002), an incremental adaptive version of the Naïve Bayes classifier. This algorithm includes an updating scheme that allows better fitting of the current model to new observations. We argue that the proposed adaptive predictive model can be implemented in any AEHS where we need to adapt the presentation based on the student's learning style and preferences. Finally, we claim that our approach should be successfully used in similar user modeling prediction tasks, where *uncertainty* and *concept drift* are presented.

Section 2 reviews some student modeling approaches based on learning styles. Section 3 covers the use of machine learning in user modeling. This section focuses the *concept drift* problem in *concept learning*. We present some adaptive learning approaches to deal with *concept drift* developed in the area of information filtering. Finally, we briefly introduce the *concept drift* detection method using P-Chart. Section 4 describes GIAS, an AEHS that we are actually developing to illustrate our approach. Section 5 describes the adaptive predictive model for the prediction task based on learning styles. Section 6 presents some experiments to evaluate our

[2] A deeper discussion about the Shewhart controls charts can be found in Montgomery, 1997

approach using simulated students. Finally, section 7 contains the conclusions and future work.

2 Learning Style in Student Modeling

Learning style can be defined as the different ways a person collects, processes and organizes information. As was argued in several works, we also claim that to effectively adapt interaction a student model must include information about the student's learning style. This assertion is based on the fact that different people learn differently (Felder, 1996): some people tend to learn by doing, whereas others tend to learn concepts; some of them like better written text and/or spoken explanations, whereas others prefer learning by visual formats of information (pictures, diagrams, etc). Consequently, the student's learning style can influence the student's preferences that usually guide the system's adaptation. Moreover, the learning style information is not subject-specific, and can be used across many AEHSs.

A pioneer work incorporating learning styles in AEHSs was proposed by Carver, Howard and Lane (1999) to support a computer science hypermedia course. The authors developed an adaptive hypermedia interface to tailor the presentation of course material based on the determination of what types of media are appropriate for different learning styles. For each course tool they compute a rate (on a scale from 0 to 100) to determine the amount of support for each learning style. The obtained rate is combined with the student's profile to produce a unique ranking of each media type from each learning style.

USD ("Teaching Support Units") is an intelligent tutoring system to support distance learning in the WEB (Peña, Marzo, and de la Rosa, 2002). The adaptation technique is approached by a multi-agent system named MAS-PLANG. The student agent implements case-based reasoning for student modeling with the aim to retrieve the relevant didactic contents (taking into account media formats and instructional strategies), navigation tools and navigation strategies based on the student's learning style. The learning style is acquired, as usually, by a psychometric instrument. Next, they assign some distributions to different materials considering which learning styles are more appropriate for different instructional strategies, media formats and navigation tools. Although the authors refer that the initial student's profile is fine-tuned to reflect more faithfully the student's learning style here it is not clear how this updating is carried out.

INSPIRE (Papanikolaou, Grigoriadou, Magoulas and Kornilakis, 2002) is an AHES that integrates theories of instructional design with learning styles. The student's learning style can be acquired using a psychometric instrument or defined by the own student. The *domain model* is structured in three hierarchical levels: *learning goals*, *concepts* and *educational materials*. Lessons are based on combinations of educational materials. The adaptation is based on the students' knowledge level and learning style. The former is used to adapt the lesson contents and the navigation support. The latter is used to determine the appropriate instructional strategy for presenting the content, that is, lessons are tailored to learners according to his/her learning styles.

In MANIC (Stern and Woolf, 2000) the student's learning style is not directly used, but it is approached by the student's preferences concerning the type of media, the instructional type and the level of abstraction of the content objects, as well as the place where these objects must be presented to the students. The tutor learns the student's preferences via machine learning by observing which objects he/she shows or hides (a *stretch-text* technique is used to adapt the presentation). A Naïve Bayes classifier predicts whether a student will want certain content objects. Those objects predicted as "*wanted*" will be shown to the user, while the others will not be shown. For each student, an example space is created based on the information about the content objects that, in the past, were either *wanted* or *not wanted* by the student. Population data is used to improve the accuracy of predictions.

Finally, a good survey of other works related with the use of learning styles in AEHS can be found in (Papanikolaou et al., 2002).

3 Machine Learning for User Modeling

User modeling systems are basically concerned with making inferences about the user's assumptions from observations of their behavior during his/her interaction with the system. On the other hand Machine Learning is concerned with the formation of models from observations. Hence, in the last years, the use of machine learning techniques in user modeling has become increasingly popular. Observations of the user's behavior can provide training examples that a machine learning system can use to induce a model designed to predict future actions (Webb, Pazzani and Billsus, 2001).

3.1. Concept Drift in Supervised Learning

The prediction task based on learning styles is related to the task of *concept learning*, a particular case of supervised learning (Mitchell, 1997).

Suppose that f: $X \rightarrow C$ maps from a feature space $X \subset \Re^N$ to a fixed set C of k classes $C=\{c_1,\ldots,c_k\}$. The goal of supervised learning is therefore: given a set of labeled examples $(x, f(x))$ to induce a *learner* (*hypothesis*) h_L: $X \rightarrow C$ that approximates f as closely as possible. The function f is called the *target concept*. In the case of *concept learning* the outputs are Boolean (e.g. C={*yes*, *no*}). A *positive (negative) example* is an example labeled "*yes*"("*no*").

As a rule, supervised learning assumes the stability of the *target concept*. Nevertheless in many real-world problems, when the data is collected over an extended period of time, the learning task can be complicated by changes in the distribution underlying the data or changes in the own target concept. This problem is known as *concept drift. Concept drift* scenarios require incremental learning algorithms, able to adjust quickly to drifting concepts (Webb et al., 2001). Depending on the rate of the changes we can distinguish *concept drift* (gradual changes) of *concept shift* (abrupt changes). For instance, in the context of our predictive task the student preferences of learning resources can change with time.

In machine learning drifting concepts are often handled by *time windows* or *weighted examples* according to their age or utility (see Klinkenberg and Renz (1998) for a brief review). 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, adapt the learner to these changes accordingly.

Examples of the former approaches are *weighted examples* and *time windows of fixed size*. Weighted examples are based on the fact that the importance of an example should decrease with time. 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.

To detect concept changes, the second group of approaches monitor the value of some performance indicators over time. If during the monitoring process a *concept drift* is detected, the learner is adapted accordingly. For instance, such an approach was proposed by Klinkenberg and Renz (1998) and by Lanquillon (2001). A deeper discussion about these and other similar approaches can be found in (Castillo, Gama and Medas, 2003).

3.2. The concept drift detection method based on P-Chart

Similar to Lanquillon, the underlying theory we use is the Statistical Quality Control (SQC) theory (Montgomery, 1997). The main idea behind SQC is to monitor the value of some quality characteristic in the production processes. Shewhart controls chart, the basic tool of SQC, is a useful *monitoring technique* that helps distinguish trends and *out-of-control* conditions in a process. This allows process correction thus reducing its variability.

The values of the *quality characteristic* are plotted on the chart in time order and connected by a line. The chart has a *center line* and *upper* and *lower control limits*. If a value falls outside the control limits, we assume that the process is *out-of-control*, i.e., some *"special causes"* have shifted the process off target. In addition to control limits we can also use *upper* and *lower warning limits*, which are usually set a bit closer to the *center line* than the control limits.

If the distribution of the *quality characteristic* is (approximately) Normal with *mean* μ and *standard deviation* σ, it is well known, that approximately 99,7% of the observations will fall within *three standard deviations* of the *mean* of the statistics. Therefore, if μ and σ are known we can use them to set the parameters of the control chart, as follows:

$$\begin{aligned} &CL = \mu \text{ - the center line is set to the } \textit{mean} \text{ value} \\ &LCL = \mu - 3\sigma \text{ and } UCL = \mu + 3\sigma \text{ - the } \textit{lower} \text{ and } \textit{upper control limits} \\ &LWL = \mu - k\sigma \text{ and } UWL = \mu + k\sigma,\ 0<k<3 \text{ - the } \textit{lower} \text{ and } \textit{upper warning limits} \end{aligned} \quad (3.1)$$

However, in most cases μ and σ are unknown and they must be estimated from previously observed values.

The control charts are classified according to the type of *quality characteristic* that they monitor: *variables* or *attributes*. P-Chart is an attribute control chart for the proportion of a dichotomous "*count*" attribute. The *quality characteristic* represents the *sample proportion* of one of the two outcomes. For large sample size *n* the *sample proportion* is *approximately Normal* with parameters:

$$\mu = p \quad \text{and} \quad \sigma = \sqrt{\frac{p(1-p)}{n}} \tag{3.2}$$

where *p* is the population proportion. Suppose that we obtain the estimate $\hat{p}$ of the parameter *p* from previous data by some estimator. Then, from equations (3.1) and (3.2), the parameters of the P-Chart for each individual *t*-th sample with size n_t would be:

$$\begin{aligned} &\text{UCL} = \hat{p} + 3\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}};\ \text{CL} = \hat{p};\ \text{LCL} = \max\left\{0, \hat{p} - 3\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}}\right\} \\ &\text{UWL} = \hat{p} + k\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}};\text{LWL} = \max\left\{0\,,\ \hat{p} - k\sqrt{\frac{\hat{p}(1-\hat{p})}{n_t}}\right\}, 0 < k < 3 \end{aligned} \tag{3.3}$$

A usual procedure to obtain $\hat{p}$ is by the weighted average of *m* preliminary sample proportions. Further we call the estimate $\hat{p}$ the *target value*.

We explore the use of the P-Chart for detecting concept drift in the following *on-line framework for supervised learning*: without loss of generality we assume that data arrives to the learner in batches over time. In real application this means that the incoming examples can be grouped by days, weeks, etc. For each batch, we use the current learner to classify the examples. The *quality characteristic* to be monitored is the *sample error rate*, a sample proportion of the misclassified examples (to evaluate it user feedback about the correct class is required). If the monitoring process detects concept drift, the learner is adapted accordingly. Next, the adapted learner is used to predict the class labels of the examples of the next batch.

In summary, the problem of handling drifting concepts can be viewed as the problem of the detection of the last moment when a *concept drift* occurred. The data stream can be analyzed as a sequence of different *contexts*: a set of examples with stationary distribution. Therefore, the monitoring process aims to detect and extract these *contexts* between drifts.

Suppose that at time *t* a new *context* begins to be processed. All the time when a lower *error rate* is achieved, the learner will try to improve, or at least, to maintain its performance level. For this reason, we propose to estimate the *target value* using the *minimum value* for the *error rate* in the current *context* instead of using some average of previous observed values. Suppose that $\text{Errs}^{(t)}$ is the error rate for the context $\text{S}^{(t)}$ at time *t* and $\text{SErrs}^{(t)}$ its standard deviation. Let Err_{min} denote the *minimum-error rate*. Err_{min} is initialized to some big number. Next, at each time step, if $\text{Errs}^{(t)} + \text{SErrs}^{(t)} < \text{Err}_{min}$ then Err_{min} is set to $\text{Errs}^{(t)}$.

Figure 1 presents the incremental adaptive method for handling *concept drift* based on P-Chart (Castillo et al., 2003). 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 *minimum value* Err_{min}. All the chart parameters are computed by the equations 3.3 (because 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. We assume that a new context is beginning and only the examples from this new context are used to re-learn the learner. If the last alert occurred at the previous time step (LastAlert=t-1) a 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 and the examples of this batch are not used to update the learner. If neither a *concept shift* nor *concept drift* is suspected the learner is updated to combine the current learner with the examples of the current batch.

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

 for t=1 to N  //for each batch with size n_t at time t
   Err_t:=Err(Batch_t,Learner);
   CL:= mean_estimator();
   Sigma:=sqrt(CL*(1-CL)/n_t);
   UCL:=CL+3.Sigma; WCL:=CL+k.Sigma;
   If Err_t > UCL then          /* concept shift suspected
   {If LastAlert=t-1 then t_ini:=FirstAlert else t_ini:=t;
      learner:=ReLearnFrom(learner,Batch_t_ini)}
   else
    If Err_t > WCL then         /* concept drift suspected
     If LastAlert=t-1 then      /* consecutive alerts
       LastAlert:=t
       else                     /* it can be a false alarm
         {learner:= UpdateWith(learner,Bacth_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.

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

4 GIAS: an Adaptive Hypermedia Educational System

GIAS[3] is a prototype WWW-based adaptive authoring-tool to support learning and teaching. The authors can organize all the available on-line *learning resources* associated to each topic of the course to support the learning processes of their students. On the other hand, the students can make use of this repository of learning resources for consulting and studying.

The main function of GIAS's adaptation is to help the students to explore a repository of learning resources associated to a set of educational goals. The ideal situation is when the student has too many options to choose from and the system can recommend him/her to explore those resources which are more appropriate to his/her learning style and preferences. Therefore, the adaptation techniques are focused on the appropriate selection of the *course's topics* and *learning resources* based on the student's *goals*, *knowledge level*, *learning style* and *preferences*.

The main difference between our approach and other similar approaches is that we try to adapt and fine-tune the initial acquired information about the student's learning style and preferences by observing the student's interactions with the system. We represent the *matches* between the student's learning style and the characteristics of learning resources into a *predictive model* (a *learner*). Moreover, we propose an *adaptive predictive model* capable of adapting quickly to any change of the student's preferences.

Similar to most of AEHSs, the main components of GIAS are the Domain Model and the Student Model. The Domain Model is composed by the Cognitive model (for knowledge representation) and the Course model (for course representation.). The Student Model represents, collects and predicts assumptions about the student's characteristics. Moreover, GIAS includes an Author Module to manage the information of the Domain Model and an Instructional Module to make decisions about how and what to adapt based on the information stored in the domain and student models. This includes three main processes: *course generation*, *topic* generation and *test generation*. Here, we focus on the Course Model, the Student Model and the *topic generation process*.

4.1. Course Model

The Course Model (see Figure 2) is organized into three-layers: *goal layer* (the course's goals), *topic layer* (the course's topics) and the *resource layer* (a set of *learning resources*). Each goal is associated to a set of topics and each course topic is associated to a set of learning resources. Between the topics we can define aggregation relationships.

[3] The name GIAS (Geometry Intelligent Adaptive System) was inherited from an adaptive web-based tutorial in Plane Geometry that we previously designed.

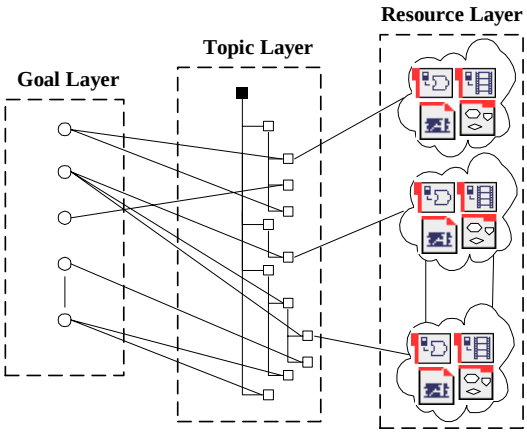

Figure 2: GIAS Course Model

A *learning resource* is an implementation of a *learning activity* in a *multimedia support*. Table 1 shows the resource features and their possible values.

Table 1. Establishing resource features and their possible values

Attribute	Value
ResourceID Description Author	
Language	Portuguese/ Spanish/ English
Creation Data	
Learning Activity (LA)	*Lesson objectives/Explanation/Example/Conceptual Map/Synthesis Diagram/ Glossary / Summary /Bibliography /HistoricalReview /Inter.Activity*
Activity Type	*to explain the new concepts/ to exemplify the new concepts/ to support the cognitive process/ to help or to coach the student*
Resource Type (RT)	*Text/HTML Text/Picture/Animated Picture/ Animated Picture with Voice/ Audio /Video /Software*
Difficulty Level	*Low/Medium/High*
Concept/skill List	

Different *learning resources* may support a *learning activity in different multimedia formats.* For example, suppose we have a theorem proof supported by two different media formats: a *static text* that describes this proof, or an *animated image with voice* that explains this proof step by step. We should suggest to a *visual* student the study of this proof using the *animated image with voice,* and to a *verbal* student the study of the proof using the *static text.*

4.2. The Student Model

A student model can represent individual students (*individual approach*) or group of users with similar characteristics (*stereotype approach*). In GIAS we use both of these approaches to model the student.

For each student his/her individual Student Model is composed by:

- Profile Model - stores personal information about the student (name, age, learning style)
- Cognitive Overlay- records the system beliefs of the student's knowledge about the domain cognitive model. Consequently, the student is classified as *novice*, *intermediate* or *expert*.
- Predictive Model – represents the student preferences about the learning resources
- Course Overlay: stores the information gathered by the system about the student's interactions with the course (e.g. how many times he/she has visited a topic or learning resource, performance in evaluation activities, etc.)

The learning style model that we adopted is the Felder-Sylverman model (Felder, 1996). This classifies students in five dimensions: *Input* (visual vs. verbal), *Perception* (sensing vs. intuitive), *Organization* (inductive vs. deductive), *Processing* (active vs. reflective), *Understanding* (sequential vs. global). We use the *Index of Learning Styles* questionnaire (ILSQ) of Felder and Soloman to assess preferences on *Input, Perception and Understanding*. For each dimension a person is classified as having a *mild*, *moderate* or *strong* preference for one category.

The acquisition of student's learning style is done *explicitly or implicitly*. When a new student logs into the system, he/she is given the option of exploring the course according to his/her learning style or without it. If a student chooses to use learning style, the student must answer to the ILSQ and the obtained scores are recorded in his/her Profile Model. On the contrary, the student's learning style should be inferred by observing the student's interactions with the system. In this chapter we assume that the acquisition of the student's learning style is done *explicitly*.

4.3. The Topic Generation Process

The *topic generation process* (see Figure 3) is executed whenever a student requests the contents of a topic. The student's Predictive Model is used to classify the available resources. The choice of the suitable *learning resources* for a topic depends on the resource's *characteristics* and on the student's *cognitive state*, *learning style* and *preferences.*

This process is performed according to the following steps:

1) *filtering*: using some deterministic rules, the *learning resources* are filtered according to the matching between the *resource's difficulty level* and the *student's knowledge level*;

2) *prediction*: using the actual Predictive Model, each filtered resource is classified as '*appropriate*' or '*not appropriate*' for the student. With this purpose, examples including the learning style features (stored in the student model) and the resource's *characteristics* (stored in the domain model) are automatically generated and classified by the actual Predictive Model. As a result the set of available resources is partitioned into these two classes.

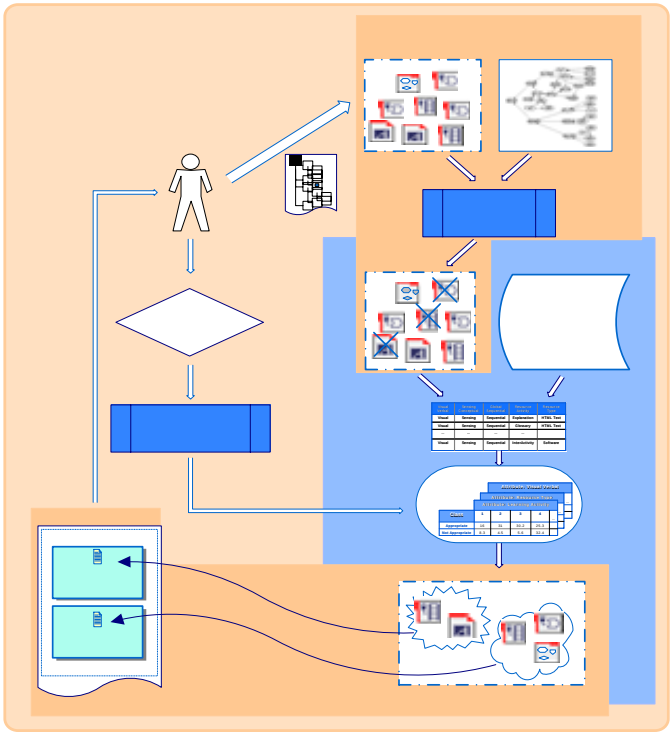

Figure 3: The topic generation process

3) *decision*: an HTML page is sent to the student including two separated ranked lists: "*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*'.

4) *adaptation*: whenever a new example is observed and evaluated, the Predictive Model is adapted accordingly.

5 The User Modeling Prediction Task

Figure 4 shows the prediction task that consists of determining whether a given resource *is* or *not appropriate* for a specific learning style by using the information about the resource and the student's learning style as input, and having the output category representing how strongly the resource is appropriate for this student.

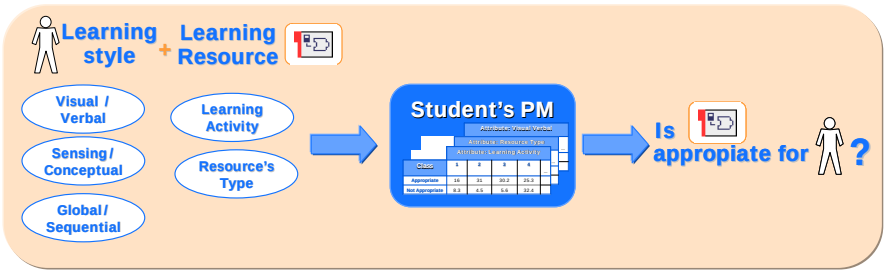

Figure 4: The prediction task

5.1. The Example Features and its Values.

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 Table 2.

Table 2. Establishing attributes and their possible values

Attributes	Values
Characterizing the student's learning style	
VisualVerbal	VV*i*, VV ∈ {Visual, Verbal}, *i* ∈ {*mild, moderate, strong*}
SensingConceptual	SC*i*, SC ∈ {Sensing, Conceptual}, *i* ∈ {*mild, moderate, strong*}
GlobalSequential	GS*i*, GS ∈ {Global, Sequential}, *i* ∈ {*mild, moderate, strong*}
Characterizing the learning resource	
Learning Activity (LA)	*Lesson objectives/Explanation/Example/Conceptual Map/Synthesis Diagram/ Glossary / Summary /Bibliography /HistoricalReview /Inter.Activity*
Resource Type (RT)	*Text/HTML Text/Picture/Animated Picture/ Animated Picture with Voice/ Audio /Video /Software*

For instance, suppose the following example: *VisualVerbal*:**Verbal*moderate***; *SensingConceptual:***Sensing*mild***; *SequentialGlobal :***Global*mild***; *Learning Activity*: ***explanation***; *Resource Type*: ***audio***. The Predictive Model must determine if a learning resource that implements 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.

Further, to simplify the prediction task we will not discriminate the preferences for a category as *mild, moderate* or *strong*. For example, in the previous example, the student is simply classified as *verbal*, *sensing* and *global*. Consequently we only have *8* different learning styles.

5.2. The Predictive Model.

Naïve Bayes (NB) classifiers (Mitchell, 1997) are probabilistic classifiers that are suitable for problems where there is uncertainty as to the correct answer. They use Bayes theorem with the assumption that the attributes are independent given the class (hence the term Naïve) to compute class probability distributions in term of their *prior* probabilities and the conditional probabilities of the attributes. All the required probabilities are computed from the training data: a frequency counter is required for each class, and a frequency counter for each attribute-value in combination with each class.

Although it is well known that most real world problems do not meet independence assumptions on which the NB classifier is based, because of its simplicity, easy use and incremental nature, it is one of the most implemented classifiers in real-world applications. NB classifier, in particular, has been successfully applied in many user modeling system (e.g. information filtering) for acquiring interest profiles. For instance in (Pazzani and Billsus, 1997) an intelligent agent called Syskill&Webert uses a NB classifier to determine whether a page would be interesting to a user. Billsus and Pazzani (1999) developed an intelligent agent named New Dudes, which learns about users' interests to compile daily news stories. This agent uses a *short-term* model that learns from the most recent observations and a *long-term* model (a NB classifier) that computes the predictions for stories not classified by the short model. The personalized WebMatcher proposed by Mladenic (1996) also implements a NB classifier to recommend links on other Web pages. And Schwab, Wolfgang and Koychev (2000) implement NB classifiers to learn interest profiles from positive evidence only.

We use Adaptive Bayes (Gama and Castillo, 2002), an adaptive version of the NB classifier, to induce the Predictive Model. Adaptive Bayes includes an updating scheme to better fit the current model to new data: after seeing each example, first, we increment the counters, and then, we adjust them 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 *observed* class.

5.3. Implementation Details.

We implement a *stereotyping approach* to initialize each student's Predictive Model. For this purpose, we also maintain a *stereotype* Predictive Model for each learning style. Firstly, we define some matching rules between a learning style and the resource's characteristics to determine which resources are more appropriate to a particular learning style. After that, we use the predefined matching rules to randomly generate some training examples, and then, we use these generated examples to initialize the counters for each *stereotype Predictive Model*. Therefore, the acquired information about the student's learning style helps us to initialize the student's Predictive Model from its related *stereotype* model.

During the further interactions with the system, the student's Predictive Model will adapt to better fit the current student's preferences by observing the student's

behavior. The most recent observations gathered trough relevant feedbacks represent the user's current preferences better than older ones. A critical task at this point is how to obtain relevant feedback, that is, a relevant set of positive and negative examples for the learning task. We propose obtaining positive examples implicitly by observing visited links. However, obtaining a relevant set of *negative examples* is more difficult. With this aim we propose the user to rate the resources explicitly. In future works we plan to investigate other methods to obtain more relevant feedback. The set of obtained examples are used to adapt the *student's Predictive Model* in the way it is explained in section 3.2.

6 Evaluation of the Predictive Model

Our evaluation measures the *accuracy* of the model's predictions on a set of artificial datasets that were specially generated to simulate concept drift scenarios for the described prediction task. Namely, we employed *simulated students* (Vanlehn, Ohlsson & Nason, 1994), a technique commonly used in Student Modeling, to evaluate the Predictive Model's performance. Therefore, each generated artificial dataset represents a *simulated student*.

6.1. Dataset Generation and Experimental Setup

Note that, students' predictive models enclose different underlying target concepts for different learning styles, and that, each individual Predictive Model is initialized from its associated *stereotype model* according to the student's learning style.

The basic idea enclosed in the generation of *simulated students* is based on the following facts that really exist in this context: for instance, suppose a *verbal* student. Learning resources that match with a *verbal* student should be appropriate for him/her. 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 predefined learning style. This means that the initial concept no longer matches the student's behaviour (a concept change happened), and thus, the Predictive Model must learn another concept ("rule"), like this:

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

Moreover, these changes in the student's preferences can lead to further adjustments in the student learning style.

Therefore, we generated simulated students for the eight possible learning styles. For each learning style, we generated datasets with 1600 examples: first, we randomly generated each feature value (see Table 2), and then, we classified each example according to the current concept. After every 400 examples the concept was changed. We grouped the examples into 32 batches of equal size (50 examples each). We also generated training datasets with 200 examples (according to the first concept) to initialize the *stereotype* predictive models.

We conducted the experiments in the on-line framework for supervised learning described in section 3.2. We evaluated the predictive accuracy of two learning algorithms: *Naïve Bayes* (NB) and *Adaptive Bayes* (AB) in combination with each of the following approaches: a *non-adaptive* approach (the baseline approach), Fixed Size Window with a size of 6 batches (FSW) and our approach (Figure 1) using the *min value* as mean estimator and k=1. We denote our approach PMin.

For each learning style we estimated the *predictive accuracy* over 10 runs (in each run we used a different generated dataset). The final estimator of the predictive accuracy of each combination "algorithm-approach" is the average of the eight accuracy estimations (one for each learning style).

6.2. Experimental Results and Analysis

In Figure 5 you can see an illustration of one P-chart for monitoring the sample error rate in one of the generated datasets.

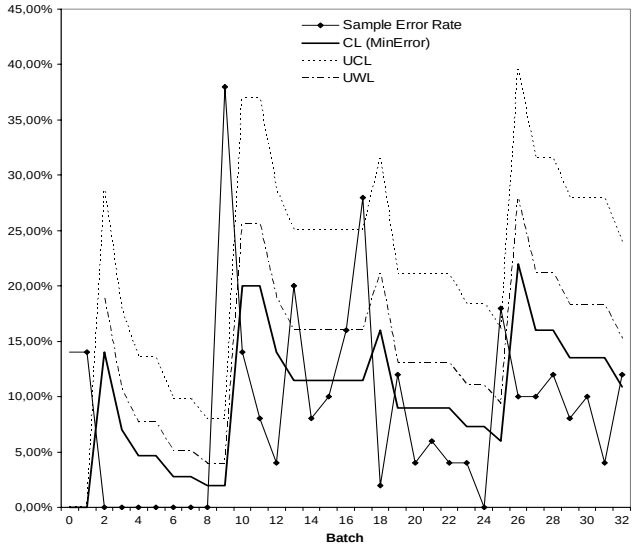

Fig. 5. A P-chart for monitoring the sample error rate.

The sample error rates are plotted on the chart and connected by a line. In each time step, the center line is adjusted according to the *minimum value* of the *error rate*

in the current *context*. This P-chart detected the three concept shifts that really exist in the data: the two first *concept shifts* (after t=8 and t=16) were detected immediately (those points that fall above the current *control limit*) and the third *concept shift* (after t=24) was detected with slight delay. However, beginning at t=25, the P-chart starts detecting an *upward* trend of the sample error, i.e., a *concept drift* (those points that fall outside the *warning limits*). When further, at t=27 the *concept shift* is signaled, all the examples beginning at t=25 are considered to belong to the same context and they all are used to re-learn the model.

Figure 6 shows the *averaged accuracy* of all the combinations "algorithm-approach" after each batch, respectively. At first, the performance of all approaches is good enough; however, those combined with Adaptive Bayes show a better performance. After the first change has occurred, the performance of non-adaptive approaches decreases significantly (they can not identify concept changes), and the performance of the FSW approach can recover a little because it re-learns regularly from the last six batches. In contrast, Pmin can quickly recover its performance decrease as the learning progresses. Moreover, PMin, instead of the FSW, whose performance depends on the window size, doesn't depend on any parameter.

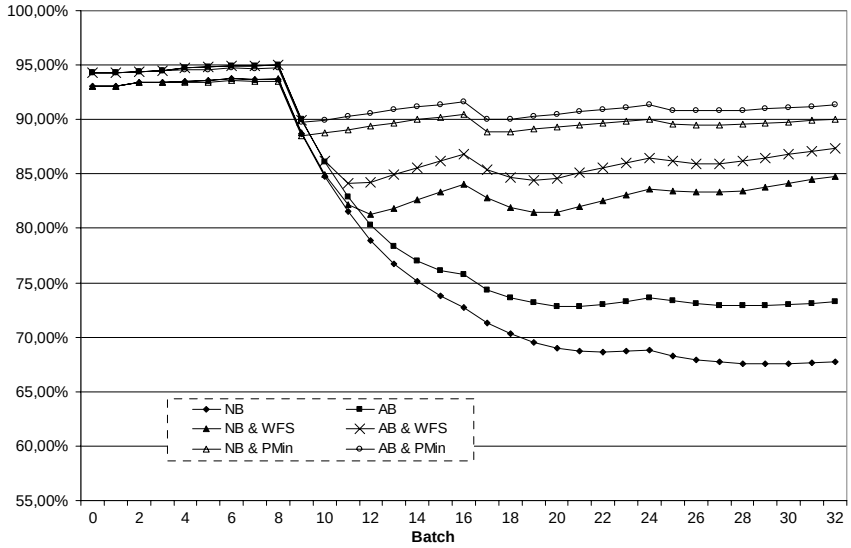

Fig. 6. Comparison of the averaged accuracy of each combination "algorithm-approach".

Table 2 compares, for all the learning styles and combinations "algorithm-approach", the average *predictive accuracy* over all the batches, and Table 3 shows some comparative studies of the performance for a pair of approaches using *paired t-tests* with a confidence level of 95%. A + (-) sign in the column "*mean*" means that the first approach obtains a better(worse) result with statistic *significance*. Note that

the mean of the accuracy is significantly different with high probability as given by the p-value.

Table 2. Predictive accuracy of the two learning algorithms (NB and AB) combined with three approaches (baseline, FSW, PMin) for the 8 different learning styles

Approaches	LS 1	LS 2	LS 3	LS 4	LS 5	LS 6	LS 7	LS 8	**Avg Acc.**
(1) NB	70.50	64.42	69.59	64.71	77.33	77.27	57.83	60.51	**67.77**
(2) NB & FSW	86.86	85.11	83.06	78.81	87.48	85.98	86.66	84.48	**84.80**
(3) NB & PMin	91.41	90.96	91.24	89.15	90.27	90.03	90.25	89.30	**90.04**
(4) AB	75.73	70.79	73.42	67.01	80.61	80.09	70.38	68.24	**73.28**
(5) AB & FSW	89.04	87.39	86.69	82.01	89.33	88.43	89.06	87.12	**87.38**
(6) AB & PMin	91.90	91.61	92.62	90.87	91.77	90.60	91.84	89.30	**91.31**

Table 3. Summary of results comparing the predictive accuracy of a pair of approaches.

		Results for the differences between the averaged accuracies					**Paired t-test**
		Mean	**Std Err**	**Median**	**Min**	**Max**	**Two sided, ∝=0.05 p-value**
I	(2) vs. (1)	+ 17.03	6.96	15.23	8.71	28.83	0.00023
	(3) vs. (1)	+ 22.27	6.80	23.05	12.76	32.42	0.00004
II	(5) vs (4)	+ 14.01	4.04	14.16	8.34	18.88	0.00002
	(6) vs (4)	+ 18.03	4.95	20.01	10.51	23.86	0.00002
III	(3) vs. (2)	+ 5.24	2.75	4.30	2.56	10.34	0.00101
	(6) vs (5)	+ 3.93	2.37	2.82	2.17	8.86	0.00222
IV	(4) vs. (1)	+ 5.51	3.39	4.53	2.30	12.55	0.00250
	(5) vs. (2)	+ 2.58	0.58	2.42	1.85	3.63	0.00000
	(6) vs. (3)	+ 1.27	0.64	1.44	0.49	2.26	0.00078

Studies I and II compare adaptive approaches that 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. Study III compares the two adaptive approaches FSW and PMin in combination with the two learning algorithms, respectively. The results shows that the performance of P-chart is significantly superior to the performance of FSW, where the learner is adapted regularly without considering whether a concept change really occurs. The last study (IV) compares the two learning algorithms. The results show that Adaptive Bayes significantly outperforms Naïve Bayes for all the approaches. In general, a more significant improvement is achieved when adaptive methods are combined with Adaptive Bayes.

7 Conclusions and Future Work

In this paper we presented an *adaptive predictive model* for a student modeling prediction task based on *learning styles*. The main difference between our approach and other similar approaches is that we try to adapt and fine-tune the initial acquired information about the student's learning style and preferences from the student's interactions with the system using machine learning techniques. We represent the matches between the learning resources and the student's learning style into an *adaptive predictive model* that is able to quickly adapt to any change of the student's preferences. We also present a general method to handle concept drift using P-Charts, which is broadly applicable to a range of domains and learning algorithms.

Although we have not performed an evaluation yet with real users, the obtained results using artificial students show that our adaptive approach consistently recognizes concept changes and that, the learner can adapt quickly to these changes in order to maintain its performance level. This means that our predictive model is able to adapt quickly to the changes in the user behavior in order to reflect more accurately the current student's preferences. In the near future we plan to evaluate our approach with real students and with other Bayesian Network classifiers.

References

1. Billsus, D., & Pazzani, M.J. (1999). A Hybrid User Model for News Stories Classifications. In Kay J. (Eds). *Proceedings of the Seventh International Conference on User Modeling, Canada*, (pp. 99-108), Springer-Verlag.
2. Brusilovsky, P. (2001). Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, v 11: pp 87-110, 2001.
3. Carver, C. A., Howard, R. A., & Lane, W.D. (1999). Enhancing Student Learning Trough Hypermedia Courseware and Incorporation of Student Learning Styles. *IEEE Transactions on Education*, *42*(1), 33-38.
4. Castillo, G., Gama, J., & Medas, P. (2003). Adaptation to Drifting Concepts. In F.M. Pires and S.Abreu. *Progress in Artificial Inteligence, Lecture Notes in Artificial Intelligence* Vol. 2902 (pp. 279-293) Springer-Verlag.
5. Felder, R. M., & Soloman, B. A. Index of Learning Style Questionnaire, available at http://www2.ncsu.edu/unity/lockers/users/f/felder/public/ILSdir/ilsweb.html
6. Felder, R.M. (1996). Matters of Style. *ASEE Prism*, *6* (4), 18-23.
7. Gama, J., & Castillo, G. (2002). Adaptive Bayes. In F.Garijo, J. Riquelme & M. Toro (Eds.), *Advances in Artificial Intelligence - IBERAMIA 2002, Lecture Notes in artificial intelligence* Vol. 2527 (pp. 765-774) Springer-Verlag.
8. Klinkenberg, R. & Renz, I. (1998) Adaptive Information Filtering: Learning in the presence of Concept Drifts, *Proceedings of the ICML-98 workshop Learning for Text Categorization*, (pp-. 33-40) AAAI Press.
9. Lanquillon, C. (2001). Enhancing Test Classification to Improve Information Filtering. *PhD. Dissertation*, University of Madgbeburg, Germany, available on-line at http: // diglib. uni-magdeburg.de/Dissertationen/2001/carlanquillon.pdf
10. Mitchell, T. (1997). *Machine Learning*. McGraw Hill.
11. Mladenic D. (1996). *Personal WebWatcher: Implementation and Design.* Technical Report, IJS-DP-7472.

12. Montgomery, D.C. (1997). *Introduction to Statistical Quality Control* (3rd ed.), New York: John Willey & Sons, Inc.
13. Papanikolaou K.A., Grigoriadou M., Magoulas G.D., & Kornilakis H. (2002). Towards New Forms of Knowledge Communication: the Adaptive Dimension of a Web-based Learning Environment, *Computers and Education*, *39* (4), 333-360.
14. Pazzani, M.J., & Billsus, D. (1997). Learning and Revising User Profiles: the identification of Interesting Web Sites. *Machine Learning*, *27*, 313-331.
15. Peña, C. I , Marzo, J. L., & de la Rosa, J. L. (2002). Intelligent Agents in a Teaching and Learning Environment on the Web. *Proceedings of the 2ond International Conference on Advanced Learning Technologies* (ICALT2002), Russia.
16. Schwab, I., Wolfgang, P., & Koychev, I. (2000). Learning to Recommend from Positive Evidence. *Proceedings of Intelligent User Interfaces*, (pp. 241-247). ACM Press.
17. Stern, M.K., & Woolf, B. P. (2000). Adaptive content in an online lecture system. In P. Brusilovsky, O. Stock, & C.Strapparava (Eds.), *Adaptive hypermedia and adaptive Web-based systems*. Lecture Notes in computer science Vol. 1982 (pp. 227-238). Berlin: Springer-Verlag.
18. Vanlehn, K., Ohlsson, S., Nason, R. (1994). Aplications of Simulated Students: an exploration. *Journal of Artificial Intelligence in Education*, *5*(2), 135-175.
19. Webb, G., Pazzani, M., & Billsus, D. (2001). Machine Learning for User Modeling. *User Modeling and User-Adapted Interaction*, *11*, 19-29.