Comparison of machine learning methods for intelligent tutoring systems

Wilhelmiina Hämäläinen[1] and Mikko Vinni[1]

Department of Computer Science, University of Joensuu,
P.O. Box 111, FI-80101 Joensuu
FINLAND
{whamalai, mvinni}@cs.joensuu.fi

Abstract. To implement real intelligence or adaptivity, the models for intelligent tutoring systems should be learnt from data. However, the educational data sets are so small that machine learning methods cannot be applied directly. In this paper, we tackle this problem, and give general outlines for creating accurate classifiers for educational data. We describe our experiment, where we were able to predict course success with more than 80% accuracy in the middle of course, given only hundred rows of data.

1 Introduction

Ability to learn is often considered as one of the main characteristics of intelligence. In this sense most of the intelligent tutoring systems are far from intelligent. They are like old-fashioned expert systems, which perform mechanic reasoning according to predefined rules. They may use very intelligent-sounding methods like Bayesian networks, decision trees or fuzzy logic, but almost always these methods are used only for model representation. They determine only, how to reason in the model, but the model itself is predefined.

Table 1. The basic terminology used in this paper.

Concept	Meaning
Model structure	Defines the variables and their relations in the model. E.g. nodes and edges in a Bayesian network, or independent and dependent variables in linear regression.
Model parameters	Assigned numerical values, which describe the variables and their relations in the given model. E.g. prior and conditional probabilities in a Bayesian network, or regression coefficients in linear regression.
Modelling paradigm	General modelling principles like definitions, assumption and techniques for constructing and using models. E.g. Bayesian networks, linear regression, neural networks, or decision trees.

This situation is very surprising, compared to other fields, where machine learning methods are widely used. In modern adaptive systems both the model structure and model parameters (see terminology in Table 1) are learnt from data. However, in educational applications, it is rare that even the model parameters are learnt from data.

The main problem is the lack of data. The educational data sets are very small – the size of class, which is often only 50-100 rows. In distance learning setting, the course sizes are larger, and it is possible to pool data from several years, if the course has remained unchanged. Still we can be happy, if we get data from 200-300 students. This is really little, when we recall that most machine learning methods require thousands of rows of data.

Another problem is that the data is often mixed, and contains both numeric and categorial variables. Numeric variables can always be transformed to categorial, but the opposite is generally not possible (we can transform all variables to binary, but in the cost of model complexity). This is not necessarily a problem, because the best methods for really small data sets use categorial data. Educational data has also one advantage compared to several other domains: the data sets are usually very clean, i.e. the values are correct and do not contain any noise from measuring devices.

In the following, we will tackle these problems. In Section 2, we describe the general appoach, which allows us to infer also the model structure from data. In Section 3, we give general guidelines for modelling educational data. We concentrate on constructing a classifier from real student data, alhtough some principles apply to other predicting tasks, as well. In Section 4, we report our empirical results, which demonstrate that quite accurate classifiers can be constructed from small data sets (around 100 rows data) with careful data preprocessing and selection of modelling paradigms. We compare the classification accuracy of multiple linear regression, support vector machines and three variations of naive Bayes classifiers. In Section 5, we introduce related research, and in Section 6, we draw the final conclusions.

2 Approach

Combination of descriptive and predictive modelling is often very fruitful, when we do not have enough data to learn models purely from data but on the other hand we do not want to rely on any ad hoc assumptions. The idea is to analyze the data first by descriptive techniques (classical data mining) to discover existing patterns in data. The patterns can be for example association rules, correlations or clusterings. Often the process is iterative and we have to try with different feature sets, before we find any meaningful patterns. In a successful case, we discover significant dependencies between the outcome variable and other variables and get information about the form of dependency. All this information is utilized in the predictive modelling phase (classical machine learning), which produces the actual models for prediction.

In the second phase, the system constructor should first decide the most appropriate modelling paradigm or paradigms and then restrict the set of suitable model structures. The selection of modelling paradigm depends on several factors like data size, number and type of attributes, form of dependencies (linear or non-linear). In the next section, we will give some general guidelines for the educational domain, but the most successful descriptive paradigms do also hint suitable predictive paradigms. For example, strong correlations support linear regression and strong associations support Bayesian methods.

If we have very little data, the model structure cannot be defined from data, but we can compose it according to descriptive patterns found in the first phase. If the descriptive analysis suggests several alternative model structures, it is best to test all of them, and select the one with smallest generalization error. The model parameters are always learnt from data. In small data sets it often happens that some parameters cannot be defined, because of the lack of data. A couple of missing variables can be handled by well-known heuristic tricks, but several missing variables is a sign of too complex a model.

3 Classifiers for educational data

The main problem in *ITS*s is classification. Before we can select any tutoring action, we should classify the situation: whether the student masters the topic or not, whether she or he will pass the course or not. However, the problem contains so many uncertain or unknown factors that we cannot classify the students deterministically into two mutual classes. Rather, we should use additional class values (e.g. the mastering level is good, average, or poor) or estimate the class probabilities. The latter approach is more recommendable, because additional variables always increase the model complexity and make it more inaccurate. On the other hand, the class probabilities can always be interpreted as intermediate class values, if needed.

Typically, the student profile contains too many attributes for building accurate classifiers, and we should select only the most influencing factors for predicting purposes. In addition, the domains of numeric attributes are large, and the data is not representative. In practice, we have to combine attributes and reduce their domains as much as possible, without losing their predictive power. As a rule of thumb, it is suggested that the data set should contain at least 10 rows of data per each model parameter. In simplest models, like naive Bayes classifiers using binary attributes, this means about $\frac{n}{20}$ independent attributes, where n is the data size.

The other consequence of this simple calculation is the model complexity. We should select modelling paradigms, which use as simple models as possible. In the following, we will suggest good candidates for both numeric and categorial data. All these classifiers are able to produce class probabilities or similar measures, instead of deterministic class decisions. We have excluded such commonly used methods like nearest neighbour classifiers, neural networks and variations of decision trees, which would require much more data to work accurately.

3.1 Classifiers for numeric data

The simplest predictive model for numeric data is linear regression. In multiple linear regression we can predict the dependent variable Y, given independent variables $X_1, ..., X_k$, by a linear equation $Y = \alpha_k X_k + \alpha_{k-1} X_{k-1} + ... + \alpha_1 X_1 + \alpha_0$. Although the result is numeric value, it can be easily interpreted as a class value in binary class problem.

The main assumption in the linear regression is that the relationship should be approximately linear. However, the model can tolerate quite large deviations from linearity, if the trend is correct. On the other hand, we can use linear regression as a descriptive tool to identify, how linear or non-linear the relationship is. This is done by checking the square of multiple correlation coefficient

$$r^2 = \frac{Var(Y) - MSE}{Var(Y)},$$

where $Var(Y)$ is Y's variance and MSE is the mean of squared errors. As a rule of thumb, if $r^2 > 0.4$, the linear tendency is significant, and we can quite safely use linear regression.

The main limitations of applying linear regression in educational domain concern outliers and collinearity. The linear regression model is very sensitive to outliers, and educational data contains almost always exceptional students, who can pass the course with minimal effort or fail without any visible reason. If the data contains several clear outliers, *robust regression* [4] can be tried instead. Collinearity means strong linear dependencies between independent variables. These are very typical in educational data, where all factors are more or less related to each other. It is hard to give any exact values, when the correlations are harmless, but in our experiment the accuracy of linear regression model classifier suffered, when the correlation coefficient was $r > 0.7$. The weaker correlations did not have any significant effect.

Support vector machines (*SVM*) [7] are another good candidate for classifying educational data. The idea is to find data points ("support vectors") which define the widest linear margin between two classes. Non-linear class boundaries can be handled by two tricks: first, the data can be mapped to a higher dimension, where the boundary is linear, and second, we can define a soft margin, which allows some misclassification. To avoid overfitting but still achieving good classification accuracy, a compromise of these two approaches is selected.

Support vector machines suit especially well for small data sets, because the classification is based on only some data points, and data dimension–size ratio has no effect on model complexity. In practice, *SVM*s have produced excellent results, and they are generally considered as best classifiers. The only shortcoming is the "black-box" nature of the model. This is in contrast with the general requirement that models in *ITS* should be transparent. In addition, selecting appropriate kernel function and other parameters is difficult, and often we have to test different settings empirically.

3.2 Bayesian classifiers for categorial data

The previously mentioned classifiers suit only for numeric and binary data. Now we will turn to classifiers, which use categorial data. This type of classifiers are more general, because we can always discretize numeric data into categorial with desired precision. The resulting models are simpler, more robust and generalize better, when the course content and students change.

Bayesian networks are very attractive method for educational domain, where uncertainty is always involved and we look for a transparent, easily understandable model. Unfortunately, the general Bayesian networks are too complex for small data sets, and the models overfit easily. *Naive Bayes models* avoid this problem. The network structure consist of only two layers, the class variable in the root node and all the other variables in the leaf nodes. In addition, it is assumed that all leaf nodes are conditionally independent, given the class value. In reality this so called *Naive Bayes assumption* is often unrealistic, but in practice the naive Bayes model has worked very well. One reason is that according to [1] Naive Bayes assumption is not a necessary but only sufficient condition for naive Bayes optimality. In empirical tests, naive Bayes classifiers have often outperformed more sophisticated classifiers like decision trees or general Bayesian networks, especially with small datasets (up to 1000 rows) [1].

In the educational domain, the Naive Bayes assumption is nearly always violated, because the variables are often interconnected. However, Naive Bayes classifier can tolerate surprisingly strong dependencies between independent variables. In our experiments, the model accuracy suffered only when the conditional probability between two leaf node values was $P(F = 0|E = 0) = 0.96$. The average mutual information between those variables was also high, $AMI(E, F) = 0.178$, of the same magnitude as the dependencies between class variable and leaf variables ($AMI \in [0.130, 0.300]$). The effect to classification accuracy was about the same as in linear regression model.

Tree augmented naive Bayes models (*TAN models*) [2] enlarge naive Bayes models by allowing additional dependencies. The *TAN* model structure is otherwise like in the naive Bayes model, but each leaf node can depend on another leaf node, in addition to class variable. This is often a good compromise between a naive Bayes model and a general Bayesian network: the model structure is simple enough to avoid overfitting, but strong dependencies can be taken into account. In the empirical tests by [2] the *TAN* model outperformed the standard naive Bayes, but in our experiments the improvements were not so striking.

Bayesian multinets [3] generalize the naive Bayes classifier further. In Bayesian multinets we can define a different network structure for every class value. This is especially useful, when classes have different independence assertions. For example, when we try to classify the course outcomes, it is very common that failed and passed students have different dependencies. In addition, the class of failed students is often much smaller and thus harder to recognize, because of less accurate parameter values. When we define for both classes their own model structures, the failed students can be modelled more accurately. In addition,

the resulting model is often much simpler (and never more complex) than the corresponding standard naive Bayes model, which improves generalization.

Friedman & al. [2] have observed that the classification accuracy in both *TAN* and Bayesian multinet models can be further improved by certain parameter smoothing operations. In "Dirichlet smoothing" or *standard parameterization* the conditional probability of X_i given its parent Π_{X_i} is calculated by

$$P(X_i = x_k | \Pi_{X_i} = y_j) = \frac{m(X_i = x_k, \Pi_{X_i} = y_j) + \alpha_{ijk}}{m(\Pi_{X_i} = y_j) + \alpha_{ij}},$$

where $\alpha_{ij} = \sum_k \alpha_{ijk}$ are Dirichlet hyperparameters. If $\alpha_{ijk} = 0$, the parameters reduce to relative frequencies. Other values can be used to integrate domain knowledge. When nothing else is known, a common choice is to use values $\alpha_{ijk} = 1$. This correction is known as *Laplace smoothing*. In empirical tests by [2] it produced significant improvements especially in small data sets, but in our experiments the improvements were quite insignificant.

4 Empirical results

Our empirical tests are related to distance learning Computer Science program *ViSCoS*, in the University of Joensuu, Finland. One of the main goals in *ViSCoS* is to develop an intelligent tutoring system based on real student data. Java Programming courses have been selected as our pilot courses, because of their importance and large drop-out and failing rates. In the first phase, we have developed classifiers, which can predict the course success (either passed or failed/drop-out) as early as possible. Currently this information serves only course teachers, but the next step is to integrate intelligent tutors into course environment.

4.1 Data

Our data set consisted of only exercise points and final grades. The data hah been collected in two programming courses, (*Prog.1* and *Prog.2*), in two consecutive years 2002-2003 and 2003-2004. In *Prog.1* course, the students study to use Java in a procedural way, and object-oriented programming is studied only in *Prog.2*.

The first problem in our modelling task was feature extraction and feature selection. Each class consisted of only 50-60 students, but each student had solved exercises in 19 weeks. To increase the data size and decrease the number of attributes, we created new attributes, which abstracted away the differences between two academic years. This was relatively easy, because the course objectives had remained the same. The exercises could be divided into six categories: basic programming skills, loops and arrays, applets, object-oriented programming, graphical applications and error handling. The exercise points in each category were summed and normalized so that the maximum points were the same in both years. After that the data sets could be combined. The resulting *Prog.1*

Table 2. Selected attributes, their numerical domain (*NDom*), binary-valued qualitative domain (*QDom*), and description.

Attr.	NDom.	QDom.	Description
A	$\{0, .., 12\}$	{little, lot}	Exercise points in basic programming structures.
B	$\{0, .., 14\}$	{little, lot}	Exercise points in loops and arrays.
C	$\{0, .., 12\}$	{little, lot}	Exercise points in applets.
D	$\{0, .., 8\}$	{little, lot}	Exercise points in object-oriented programming.
E	$\{0, .., 19\}$	{little, lot}	Exercise points in graphical applications.
F	$\{0, .., 10\}$	{little, lot}	Exercise points in error handling.
$TP1$	$\{0, .., 30\}$	{little, lot}	Total points in *Prog.1*.
$TP2$	$\{0, .., 30\}$	{little, lot}	Total points in *Prog.2*.
$FR1$	$\{0, 1\}$	{fail, pass}	Final result in *Prog.1*.
$FR2$	$\{0, 1\}$	{fail, pass}	Final result in *Prog.2*.

data set contained 125 rows and four attributes and *Prog.2* data set contained 88 rows and eight attributes.

The attributes, their domains and descriptions are presented in Table 2. In the original data set, all attributes were numerical, but for Bayesian classifiers we converted them to binary-valued qualitative attributes.

4.2 Classifiers

For model construction, we performed descriptive data analysis, where we searched dependencies between numeric and categorial attributes by correlation, correlation ratio, mutual information and association rules. The analysis revealed that total points TP in both courses depended heavily on exercise points and the dependency was quite linear. Even stronger dependencies were discovered by association rules between final results FR and binary versions of exercise attributes. The dependency analysis revealed also dependencies between exercise attributes. Most of them were moderate, except the dependency between E and F. In addition, we found that final results in *Prog.2* depended strongly on *Prog.1* attributes B, C and $TP1$. B attribute affected mostly among successful students and C among failed students, but generally $TP1$ was the most affecting factor.

The goal was to predict the course final results $FR1$ and $FR2$ as early as possible, and thus we tested seven different cases:

1. $A \Rightarrow FR1$
2. $A, B \Rightarrow FR1$
3. $A, B, C \Rightarrow FR1$
4. $TP1 \Rightarrow FR2$
5. $TP1, D \Rightarrow FR2$
6. $TP1, D, E \Rightarrow FR2$
7. $TP1, D, E, F \Rightarrow FR2$

Two numeric classification methods (multiple linear regression and support vector machines) were applied to numeric data, and three versions of naive Bayes classifiers (standard *NB*, *TAN* and Bayesian multinets) to categorial data. In numerical methods, we had to learn seven different models in both paradigms, but for naive Bayes classifiers, it was enough to learn just one model for *Prog.1*

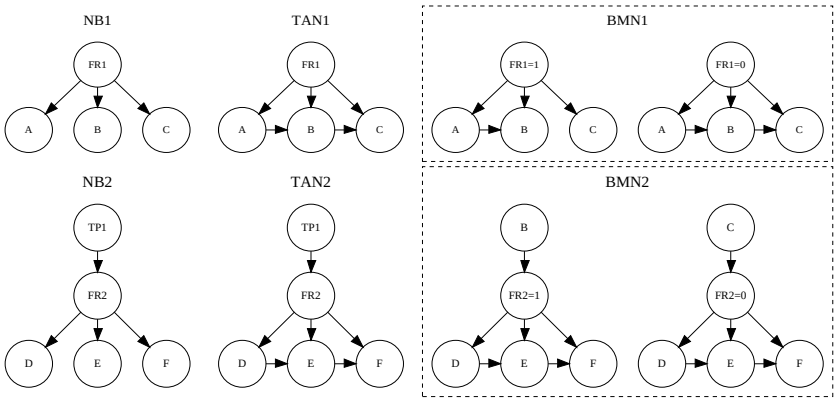

Fig. 1. Naive Bayes (*NB1, NB2*), Tree-augmented Naive Bayes (*TAN1, TAN2*), and Bayesian multinet (*BMN1, BMN2*) classifiers for predicting the final results ($FR1, FR2$) in *Prog.1* and *Prog.2* courses. A, B, C, D, E and F are exercise points. Either total points in *Prog.1* $TP1$ or exercise points in B or C categories have been used as background variables for *Prog.2* models.

and one for *Prog.2* and update the class probabilities by Bayes rule, when new exercise attributes were known.

The naive Bayes model structures are described in Figure 1. In *TAN* models, we have included the strongest dependencies between exercise attributes. The Bayesian multinet structures are quite similar, because the dependencies among failed and successful students were mostly same. The biggest difference is that in the *Prog.2* model, we use attribute B to initialize $P(FR2)$ in the succeeders' network and C in the failers' network. The model parameters in *TAN* models and Bayesian networks were calculated both with and without Laplace smoothing.

4.3 Results

The models have been compared in all test cases by 10-fold cross-validation. The generalization error has been expressed as true positive and true negative rates, which tell the proportion of correctly predicted successful students from all successful students, and similarly for failed students. The results are represented in Table 3.

We observe that generally the support vector machine performed best, especially in the last cases, when several attributes were included. However, in *Prog.1* data, the naive Bayes classifiers predicted the failers better than the support vector machine. Predicting potential failers/drop-outs accurately is more important, because they would need special tutoring. In addition, the real probabilities reveal more information.

When we compare different naive Bayes classifiers, we observe that the results are quite controversal. In *Prog.1*, the *TAN* and Bayesian multinet models achieve

Table 3. Comparison of five classification methods. Linear regression (*LR*) and support vector machine (*SVM*) classifiers were trained with numeric course data, and Naive Bayes (*NB*), tree-augmented Bayesian networks (*TAN*) and Bayesian multinets (*BMN*) with categorial data. The generalization error has been estimated by 10-fold cross-validation and for each model structure and classification method true positive *TP* and true negative *TN* rates are reported.

Model	LR		SVM		NB		TAN		BMN	
structure	TP	TN	TP	TN	TP	TN	TP	TN	TP	TN
$A \Rightarrow FR1$	0.83	0.47	0.91	0.42	0.96	0.31	0.96	0.31	0.96	0.31
$A, B \Rightarrow FR1$	0.91	0.72	0.93	0.69	0.80	0.81	0.82	0.83	0.82	0.83
$A, B, C \Rightarrow FR1$	0.93	0.81	0.94	0.78	0.83	0.81	0.84	0.81	0.85	0.81
$TP1 \Rightarrow FR2$	0.70	0.68	0.88	0.59	0.96	0.51	0.96	0.51	0.75	0.38
$TP1, D \Rightarrow FR2$	0.78	0.85	0.86	0.76	0.71	0.68	0.71	0.68	0.73	0.73
$TP1, D, E \Rightarrow FR2$	0.75	0.90	0.98	0.82	0.82	0.86	0.82	0.81	0.78	0.81
$TP1, D, E, F \Rightarrow FR2$	0.70	0.92	0.94	0.82	0.86	0.86	0.84	0.81	0.78	0.81

a slight improvement, but in *Prog2*, the standard naive Bayes model performs better. The results by *TAN* and Bayesian multinets are quite similar especially in *Prog.1*. This is not surprising, because the model structures are nearly the same. However, in *Prog.2*, the *TAN* model outperforms multinet model in almost all cases. This suggests that total points in *Prog.1* is a better background variable than exercise points in B or C.

We have also tried the Laplace smoothing in all Bayesian models, but it did not produce any improvement. In fact, in some cases it only increased the generalization error.

5 Related research

So far, none of the existing *ITS*s learn the model from data, but there have been some experiments to this direction. Two studies have tackled quite a similar problem to ours, but the data sizes were much larger and the data sets contained very different features. The prediction accuracy in these studies was of the same magnitude as in our case, although in the second study, the prediction was done in the end of course.

Kotsiantis et al. [5] have compared six classification methods (Naive Bayes, decision tree, feed-forward neural network, support vector machine, 3-nearest neighbour and logistic regression) to predict drop-outs in the middle of course. The data set contained demographic data, results of the first writing assignments and participation to group meetings. The data set contained 350 students. The best classifiers, Naive Bayes and neural network, were able to predict about 80% of drop-outs.

Minaei-Bidgoli et al. [6] have compared six classifiers (quadratic Bayesian classifier, 1-nearest neighbours, k-nearest neighbours, Parzen window, feed-forward

neural network, and decision tree) to predict the course final results from a learning system log data. The data contained attributes concerning each task solved and other actions like participating in the communication mechanism and reading support material. The data set contained 250 students. The best classifier, k-nearest neighbours, achieved over 80% accuracy, when the final results had only two classes (pass/fail).

6 Conclusions

In this paper, we have tackled a difficult problem of learning classifiers for *ITS* from real data. The main problem is the small size of educational data sets, and the traditional machine learning methods cannot be applied directly. However, with careful preprocessing and selection of modelling paradigms we can learn quite accurate classifiers.

We have given general outlines, how to classify successfully small data sets of both numeric and categorial data. We recommend especially variations of naive Bayes classifiers, which are robust, can handle mixed variables and produce informative results (class probabilities).

We have also reported our empirical study, where we compared five classification methods for predicting the course outcomes as early as possible. The data consisted of only exercise points and the data sets were very small (in *Prog.1* course 125 rows and in *Prog.2* course 88 rows).

For numerical data, we used multiple linear regression and support vector machine classifiers, and for categorial data, three variations of naive Bayes classifier (the standard model, *TAN* model and Bayesian multinets). All methods achieved about the same accuracy. In *Prog.1*, 80% accuracy was achieved in the middle of course and in *Prog.2* already in the beginning of course.

References

1. P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.
2. N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.
3. D. Geiger and D. Heckerman. Knowledge representation and inference in similarity networks and Bayesian multinets. *Artificial Intelligence*, 82:45–74, 1996.
4. P.J. Huber. Robust estimation of a location parameter. *Annals of mathematical statistics*, 35:73–101, 1964.
5. S.B. Kotsiantis, C.J. Pierrakeas, and P.E. Pintelas. Preventing student dropout in distance learning using machine learning techniques. In *Proceedings of 7th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES)*, pages 267–274, 2003.
6. B. Minaei-Bidgoli, D.A. Kashy, G. Kortemeyer, and W. Punch. Predicting student performance: an application of data mining methods with an educational web-based system. In *Proceedings of 33rd Frontiers in Education Conference*, pages T2A13–T2A18, 2003.
7. V.N. Vapnik. *Statistical learning theory.* John Wiley & Sons, 1998.