Development of an Intelligent Tutoring System using mixture modelling of learning curves

Tiago Quintas

September 15, 2023

Abstract

The web application SIACUA is a web plataform created to assist calculus students for an autonomous learning experience. To enhance the students' learning capabilites, over the years a new Intelligent Tutoring System (ITS) was being implemented, to provide guidance and recommendations to the user. In this paper we present an ITS capable of drawing learning curves and surfaces for each student individually and for the global dataset, while recommending various study tips in real time. Mixture modelling and bayesian inference were used to draw the learning curves and surfaces, as well as t-tests for the ITS's decision patterns, taking advantage of the bayesian networks already implemented in SIACUA.

1 Introduction

- Why an ITS?
- How does SIACUA work?
- What are the ITS' objectives?

1.1 Related Work

- What other papers have done?

2 Mixture modelling

Imagine a student starts answering a question of difficulty i, while having a knowledge belief of b_1 for that topic. If we consider the binary set $e = \{0, 1\}$ for the two outcomes for the answer, where 0 means a correct answer and 1 an incorrect answer, we can represent the full action with a vector for the question, $\mathbf{q}_i$, a number for the belief, b_1, and a number for the answer, e_1.

$$\mathbf{q}_i = [\underbrace{0 \;\; 0 \;\; \ldots \;\; \overbrace{1}^{i^{\text{th}} \text{ position}} \;\; \ldots \;\; 0}_{d}]^T$$

After answering a total of n questions, we can construct a question matrix, $\mathbf{Q}$, a beliefs vector, $\mathbf{b}$, and an error vector, $\mathbf{e}$, such that

$$\begin{aligned} \mathbf{Q} &= \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \mathbf{q}_3 & \ldots & \mathbf{q}_n \end{bmatrix}, \quad \mathbf{Q} \in \mathcal{M}_{d\times n}\left(\{0,1\}\right) \\ \mathbf{b} &= \begin{bmatrix} b_1 & b_2 & b_3 & \ldots & b_n \end{bmatrix}^T, \quad \mathbf{b} \in \mathcal{M}_{n\times 1}\left([0,1]\right) \\ \mathbf{e} &\in \{0,1\}^n. \end{aligned}$$

The matrix $\mathbf{Q}$ is called in this paper a *XOR Matrix by columns*, meaning that $\mathbf{Q}^T \cdot \mathbb{1}_d = \mathbb{1}_n$, where

$$\mathbb{1}_n = [\underbrace{1 \quad 1 \quad \ldots \quad 1}_{n}]^T.$$

This happens because a single question cannot have two different difficulty levels. So, the matrix $\mathbf{Q}$ provides information about the questions' difficulty and the vector $\mathbf{e}$ provides the evidence of the answers. What is the vector $\mathbf{b}$ used for? Since each entry represents the level of belief that the student has knowledge about the topic, we can use those values to estimate the probability of failing a question of a certain difficulty. For that, an initial piecewise function ϕ was created to map a level of belief to the corresponding probability of answering incorrectly.

$$\phi(x) = \begin{cases} 1 - p_g, & \text{x} < 0.05 \\ \frac{p_s - (1-p_g)}{0.9}(x - 0.05) + (1 - p_g), & 0.05 < \text{x} < 0.95 \\ p_s, & \text{x} > 0.95 \end{cases}$$

This mapping is mainly linear, considering a probability of just guessing the right answer, p_g, and the probability of a *slip*, a mistake on the calculations, p_s. Applying this function to each element of $\mathbf{b}$, yields the vector

$$\mathbf{b}|_\phi = \begin{bmatrix} \phi(b_1) & \phi(b_2) & \dots & \phi(b_n) \end{bmatrix}^T = \boldsymbol{\phi}_\mathbf{b}.$$

This vector presents a rough estimation about the probability of failing each one of the questions, but how do we find the general probability? If we consider each question as a variable that follows a Bernoulli distribution, the probability ϑ of success of that distribution is somehow derived from the vector $\boldsymbol{\phi}_\mathbf{b}$.

Lets suppose that $\boldsymbol{\phi} = (X_1, X_2, X_3, \dots, X_n)$, where $X_i \sim \text{Beta}(\alpha, \beta)$. Using the method of moments, we can estimate values for $\hat{\alpha}$ and $\hat{\beta}$. Let $\bar{X}$ be the sample mean and S^2 the sample variance.

$$\hat{\alpha} = \bar{X}\left[\frac{\bar{X}(1-\bar{X})}{S^2} - 1\right],$$
$$\hat{\beta} = (1-\bar{X})\left[\frac{\bar{X}(1-\bar{X})}{S^2} - 1\right].$$

Regarding $\boldsymbol{\phi}_\mathbf{b}$ as an observation of $\boldsymbol{\phi}$, we can use the distribution $\text{Beta}\left(\hat{\alpha}, \hat{\beta}\right)$ as a prior for the parameter ϑ of the Bernoulli distribution, where $P(\vartheta) = \text{Beta}\left(\vartheta \mid \hat{\alpha}, \hat{\beta}\right) = \frac{\vartheta^{\hat{\alpha}-1}(1-\vartheta)^{\hat{\beta}-1}}{B\left(\hat{\alpha}, \hat{\beta}\right)}$. Then, using as evidence the vector $\mathbf{e}$, we can apply the Bayes' rule to update the parameters of the Beta distribution. Before jumping to that, we need to present some notation.

We say a matrix M is *row-induced* by a vector b iff there is a matrix A such that $M = \text{diag}(b) \cdot A$, where

$$\text{diag}(b) = \begin{bmatrix} b_1 & 0 & \dots & 0 \\ 0 & b_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b_n \end{bmatrix},$$

and we write $M = A \mid_r b$. Likewise, we say that M is *column-induced* by a vector b iff there is a matrix A such that $M = A \cdot \text{diag}(b)$, and we write $M = A \mid_c b$. In this paper, we also use the symbol $\circ$ as the Hadamard product between matrices.

To calculate the estimated $\hat{\alpha}_d$ and $\hat{\beta}_d$ for each level of difficulty, we have to calculate the sample mean and variance and apply the method of moments for each difficulty level. Since $\mathbf{Q}$ is a *XOR Matrix by columns* and not by rows, we have that $\mathbf{Q}^T \cdot \mathbb{1}_d = \mathbb{1}_n$ and $\mathbf{Q} \cdot \mathbb{1}_n = \mathbf{n}_d$, where $\mathbf{n}_d$ is a vector consisting of the number of questions per difficulty level. Following the method of moments:

$$\text{Sample mean: } \boldsymbol{\mu}_d = (\mathbf{Q} \mid_c \boldsymbol{\phi}_\mathbf{b}) \circ \mathbf{n}_d^{\circ -1}$$
$$\text{Sample variance: } \mathbf{S}_d^2 = \left((\mathbf{Q} \mid_c \boldsymbol{\phi}_\mathbf{b} - \mathbf{Q} \mid_r \boldsymbol{\mu}_d)(\mathbf{Q} \mid_c \boldsymbol{\phi}_\mathbf{b} - \mathbf{Q} \mid_r \boldsymbol{\mu}_d)^T\right) \circ (\mathbf{n}_d - \mathbb{1}_d)^{\circ -1}$$
$$\text{Estimated } \alpha\text{: } \hat{\boldsymbol{\alpha}}_d = \boldsymbol{\mu}_d \circ \left[\boldsymbol{\mu}_d \circ (\mathbb{1}_d - \boldsymbol{\mu}_d) \circ \mathbf{S}_d^{\circ -2} - \mathbb{1}_d\right]$$
$$\text{Estimated } \beta\text{: } \hat{\boldsymbol{\beta}}_d = (\mathbb{1}_d - \boldsymbol{\mu}_d) \circ \left[\boldsymbol{\mu}_d \circ (\mathbb{1}_d - \boldsymbol{\mu}_d) \circ \mathbf{S}_d^{\circ -2} - \mathbb{1}_d\right]$$

Using these results we can define a vector that holds all the priors for the Bernoulli distributions, where

the vector $\boldsymbol{\vartheta}$ describes a possible learning curve and

$$P(\boldsymbol{\vartheta}) = \mathbf{Beta}\Big(\boldsymbol{\vartheta} \mid \hat{\boldsymbol{\alpha}}_d, \hat{\boldsymbol{\beta}}_d\Big) = \begin{bmatrix} \text{Beta}\Big(\vartheta_1 \mid \hat{\alpha}_1, \hat{\beta}_1\Big) \\ \text{Beta}\Big(\vartheta_2 \mid \hat{\alpha}_2, \hat{\beta}_2\Big) \\ \vdots \\ \text{Beta}\Big(\vartheta_d \mid \hat{\alpha}_d, \hat{\beta}_d\Big) \end{bmatrix}$$

2.1 General Mixture Model and Bayesian Inference

Each possible learning curve can be a component of a probabilistic model, consisting on a product of Bernoulli distributions. A learning curve $\boldsymbol{\vartheta}$ represents the performance level of the student for each difficulty. The probability of the error vector $\mathbf{e}^s$ according to the learning curve $\boldsymbol{\vartheta}$ is

$$P(\mathbf{e}^s \mid \boldsymbol{\vartheta}) = \varpi\left(\boldsymbol{\mathcal{B}}\left(\mathbf{Q}^T \cdot \boldsymbol{\vartheta}, \mathbf{e}^s\right)\right) = \prod_{n=1}^{N} \mathcal{B}\left((\mathbf{Q}^T)_n \cdot \boldsymbol{\vartheta}, \mathbf{e}^s_n\right),$$

where $\varpi(b) = \det(\text{diag}(b)) = \prod_n b_n$. So a K-component mixture over learning curves is a set of learning curves $\boldsymbol{\vartheta}_1, \boldsymbol{\vartheta}_2, \ldots, \boldsymbol{\vartheta}_k$ with probabilities $\rho_1, \rho_2, \ldots, \rho_k$, where the probability of the error vector $\mathbf{e}^s$, according to the mixture model, is

$$\sum_{k=1}^{K} \rho_k \cdot \varpi\left(\boldsymbol{\mathcal{B}}\left(\mathbf{Q}^T \cdot \boldsymbol{\vartheta}_k, \mathbf{e}^s\right)\right).$$

The Bayes' rule states that $P(\boldsymbol{\vartheta} \mid \mathbf{e}) \propto P(\mathbf{e} \mid \boldsymbol{\vartheta})P(\boldsymbol{\vartheta})$, which means that, after collecting some evidence about the error vector $\mathbf{e}$, the values for α and β on $\mathbf{Beta}\Big(\hat{\boldsymbol{\alpha}}_d, \hat{\boldsymbol{\beta}}_d\Big)$ can be updated for each level of difficulty. However, and because we are working with vectors and multiple difficulty levels, the formula for the rule has to be updated to the vector form $\mathbf{P}(\boldsymbol{\vartheta} \mid \mathbf{e}) \propto \mathbf{P}(\mathbf{e} \mid \boldsymbol{\vartheta}) \circ \mathbf{P}(\boldsymbol{\vartheta})$.

$$\mathbf{P}_d(\mathbf{e} \mid \boldsymbol{\vartheta}) = \prod_{\substack{n=1 \\ \mathbf{Q}_{d,n} \neq 0}}^{N} \mathcal{B}\left(\boldsymbol{\vartheta}_d, \mathbf{e}_n\right) = \vartheta^{n|_{\mathbf{e}_n=1}}(1-\vartheta)^{n|_{\mathbf{e}_n=0}},$$

$$\mathbf{P}_d(\boldsymbol{\vartheta}) = \text{Beta}\Big(\vartheta_d \mid \hat{\alpha}_d, \hat{\beta}_d\Big) = \frac{\vartheta^{\hat{\alpha}_d - 1}(1-\vartheta)^{\hat{\beta}_d - 1}}{B\Big(\hat{\alpha}_d, \hat{\beta}_d\Big)},$$

$$\mathbf{P}_d(\boldsymbol{\vartheta} \mid \mathbf{e}) \propto \mathbf{P}_d(\mathbf{e} \mid \boldsymbol{\vartheta}) \cdot \mathbf{P}_d(\boldsymbol{\vartheta}) = \frac{\vartheta^{\hat{\alpha}_d + n|_{\mathbf{e}_n=1} - 1}(1-\vartheta)^{\hat{\beta}_d + n|_{\mathbf{e}_n=0} - 1}}{B\Big(\hat{\alpha}_d + n|_{\mathbf{e}_n=1}, \hat{\beta}_d + n|_{\mathbf{e}_n=0}\Big)} = \text{Beta}\Big(\dot{\alpha}_d, \dot{\beta}_d\Big).$$

Now that we have calculated the posterior probability, we can normalize the parameters to get back a good value for $\dot{\vartheta}_d = \frac{\dot{\alpha}}{\dot{\alpha}+\dot{\beta}}$. The main issue about this method is that the real likelihood of ϑ, given an error vector $\mathbf{e}^s$ of a student, also accounts for different difficulties, and the method above checks for right/wrong answers for each difficulty individually. Although, it's a good and simple way of getting a fitting learning curve after answering a series of questions, based on the beliefs of the topics regarding those questions.

2.2 Statistical Consistency

The goal is then to create a method such that the mixture model is statistical consistent, i.e., given enough data, it converges to the true probabilities. So, given data about the matices $\mathbf{Q}^s$ and error vectors $\mathbf{e}^s$, for each student s, we have to find an algorithm that guarantees that the student s belongs to one and only one learning curve as $N \to +\infty$ and $K \to +\infty$. It is important to state that what we call *leraning curve* in this paper expresses a curve that isn't temporal nor measured relatively to the number of tries, but as a static value for each probability of making a mistake on a question of a given difficulty. This means that, although the question matrices and the error vectors should be temporal just to keep operations between them truthful, their orientation is meaningless to the mixture model, so perturbations on the matrices and vectors imply changing multiple 0's to 1's, or vice-versa.

Theorem 1. *Given a sufficient number of components, there exists a statistical consistent algorithm that ensures the mixture model converges to the ground truth as the number of difficulty matrices* $\mathbf{Q}^s$ *and error vectors* $\mathbf{e}^s$ *grows.*

Proof. To prove the theorem, for simplicity, let's first assume there's just one level of difficulty, i.e., $\mathbf{Q}^s = (\mathbb{1}_n)^T$ and each Bernoulli distribution is of the form $\mathcal{B}(\vartheta_k, e_n^s)$. The likelihood $L(\mathbf{e}^s \mid \vartheta_k) = \prod_n \mathcal{B}(\vartheta_k, e_n^s)$ can be maximixed by calculating $\frac{\partial}{\partial \vartheta_k} L(\mathbf{e}^s \mid \vartheta_k) = 0$, where $\vartheta_k \in]0, 1[$. Because there's only one difficulty level, $L(\mathbf{e}^s \mid \vartheta_k) = \vartheta_k^w (1 - \vartheta_k)^r$, where $r \geq 1$ and $w \geq 1$ mean the number of right and wrong answers, and

$$\begin{aligned} 0 &= \frac{\partial}{\partial \vartheta_k} L(\mathbf{e}^s \mid \vartheta_k) \\ 0 &= \vartheta_k^{w-1} (1 - \vartheta_k)^{r-1} \left(w - \vartheta_k (w + r)\right) \\ 0 &= \left(w - \vartheta_k (w + r)\right) \\ \vartheta_k &= \frac{w}{w + r}. \end{aligned}$$

This means that for an error vector $\mathbf{e}^s$, where $w = (\mathbf{e}^s)^T \cdot \mathbb{1}_n$ and $r = n - w$, there exists an exact value for ϑ_k that maximizes the likelihood. So, as $K \to +\infty$,

$$\forall \varepsilon > 0 \;\; \exists k \leq K \in \mathbb{N} \;\; \left| \vartheta_k - \frac{w}{w + r} \right| < \varepsilon. \tag{1}$$

However, how do we know that this algorithm provides stability to the system? Regarding perturbations on the error vectors $\mathbf{e}^{s'}$, how do they influence the value of likelihood $L\Big(\mathbf{e}^{s'} \mid \vartheta_k\Big)$? Considering a fixed perturbation $t \geq 1$, the values for right and wrong answers change to $w' = w \pm t$ and $r' = r \mp t$, and so the ϑ that maximizes likelihood would be $\vartheta_k^{\pm} = \frac{w \pm t}{w + r}$, and that doesn't guarantee the truth of (1) for $\mathbf{e}^s$ and $\mathbf{e}^{s'}$ simultaneously. Nonetheless, as $n \to +\infty$, $w \pm t \approx w$, and a learning curve that is "close enough" would also be fitting to that perturbation. So, for a fixed perturbation $t \neq 0$, generating the vector $\mathbf{e}^{s'}$,

$$\forall \delta > 0 \;\; \exists \varepsilon > 0 \;\; \exists n \in \mathbb{N} \;\; \left| \frac{t}{w + r} \right| < \varepsilon \Rightarrow \left| L\left(\mathbf{e}^{s'} \mid \frac{w + t}{w + r}\right) - L\left(\mathbf{e}^{s} \mid \frac{w + t}{w + r}\right) \right| < \delta, \tag{2}$$

which means that, joining (1) and (2),

$$\forall \delta > 0 \;\; \exists \varepsilon > 0 \;\; \exists n, k \in \mathbb{N} \;\; \left| \vartheta_k - \frac{w}{w + r} \right| < \left| \frac{t}{w + r} \right| < \varepsilon \Rightarrow \left| L\Big(\mathbf{e}^{s'} \mid \vartheta_k\Big) - L(\mathbf{e}^s \mid \vartheta_k) \right| < \delta, \tag{3}$$

which proves that, for one difficulty, given sufficient components and provided enough data points, each error vector will converge on a single component, i.e., the one that maximizes the likelihood, which is the same as saying, the one that minimizes the distance $\left| \vartheta_k - \frac{w}{w+r} \right|$, and, as $n \to +\infty$, error vectors with relatively small perturbations will converge to a true value for ϑ. This proves the statistical consistency to a model of just one difficulty.

However, regarding d difficulties, the likelihood function is of the type

$$L(\mathbf{e}^s \mid \boldsymbol{\vartheta}) = \vartheta_1^{w_1} (1 - \vartheta_1)^{r_1} \cdot \vartheta_2^{w_2} (1 - \vartheta_2)^{r_2} \cdot \ldots \cdot \vartheta_d^{w_d} (1 - \vartheta_d)^{r_d} = f_1(\vartheta_1) f_2(\vartheta_2) \ldots f_d(\vartheta_d),$$

where each $f_i : (0, 1) \to (0, 1)$, which means that the maximum likelihood is achieved when the maximum of each f_i is achieved. Following the same reasoning as before, given a fixed perturbation vector $\mathbf{t}$, (3) can be reconstructed as:

$$\forall \delta > 0 \;\; \exists \boldsymbol{\varepsilon} > \mathbb{0}_d \;\; \exists n, k \in \mathbb{N} \;\; \left| \boldsymbol{\vartheta}_k - \mathbf{w} \circ \mathbf{n}^{\circ -1} \right| < \left| \mathbf{t} \circ \mathbf{n}^{\circ -1} \right| < \boldsymbol{\varepsilon} \Rightarrow \left| L\Big(\mathbf{e}^{s'} \mid \boldsymbol{\vartheta}_k\Big) - L(\mathbf{e}^s \mid \boldsymbol{\vartheta}_k) \right| < \delta. \tag{4}$$

This proves the statistical consistency of the model for d difficulties and an algorithm can be constructed, where the likelihood is calculated for every student and component and then normalized, each $\boldsymbol{\vartheta}_k$ is calculated with bayesian inference based on the beliefs of the questions or beliefs predetermined for the components and the values of ρ_k are calculated based on the empirical frequency of $\mathbf{e}^s$ within the dataset.

Each iteration is of the form:

$$\begin{aligned}
\mathbf{L}_{s,k} &= \rho_k \cdot \varpi\left(\boldsymbol{\mathcal{B}}\left(\mathbf{Q}^T \cdot \boldsymbol{\vartheta}_k, \mathbf{e}^s\right)\right) \\
\mathbf{Z}_{s,k} &= \frac{\mathbf{L}_{s,k}}{\left(\mathbf{L} \cdot \mathbb{1}_K\right)_s} \\
\boldsymbol{\vartheta}_k &= \left(\boldsymbol{\alpha}_k + \sum_s \left(\mathbf{Z}_{s,k} \cdot \mathbf{w}^s\right)\right) \circ \left(\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k + \sum_s \left(\mathbf{Z}_{s,k} \cdot \mathbf{n}^s\right)\right)^{\circ -1} \\
\boldsymbol{\rho} &= \frac{\mathbf{Z}^T \cdot \mathbb{1}_S}{(\mathbb{1}_S)^T \cdot \mathbf{Z} \cdot \mathbb{1}_K},
\end{aligned} \tag{5}$$

where $\mathbf{w}^s = (\mathbf{Q}^s \mid_c \mathbf{e}^s) \cdot \mathbb{1}_N$ and $\mathbf{n}^s = \mathbf{Q}^s \cdot \mathbb{1}_N$. □

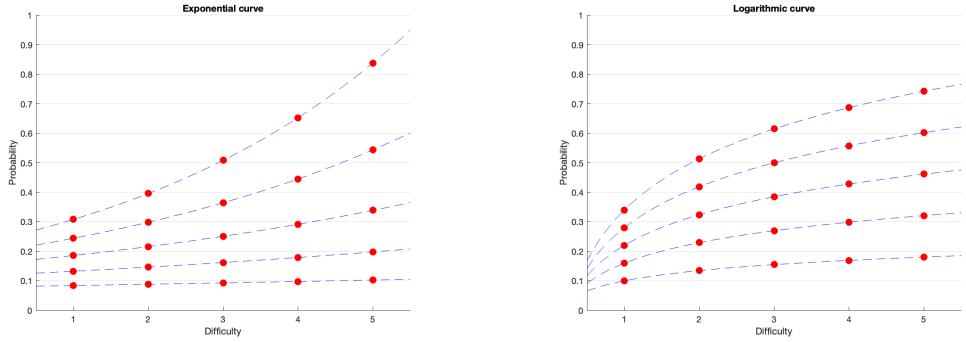
2.3 Model predictions

After implementing this algorithm and collecting enough data, we are predicting two types of learning curves: *exponential* and *logarithmic*. Both describe how the student adapts to higher levels of difficulty. The curves will be of the form:

$$\vartheta(d) = ae^{b(d-1)}, \quad 0 < a < 1, \quad 0 < b < -\frac{\ln a}{4}, \tag{6}$$

$$\vartheta(d) = a + b \ln d, \quad 0 < a < 1, \quad 0 < b < \frac{1-a}{\ln 5}, \tag{7}$$

where a means the probability of answering incorrectly a question of level 1 and b describes the steepness of the curve.



(a) Five possible learning curves following the exponential equation $\vartheta(d) = ae^{b(d-1)}$.

(b) Five possible learning curves following the logarithmic equation $\vartheta(d) = a + b \ln d$.

Figure 1: Predictions of the *exponential* (1a) and the *logarithmic* (1b) curves.

After computing all the learning curves, it is possible to make exponential and logarithmic regressions on those curves, to find the parameters a and b. Then, the residuals are calculated and the regression with smaller residuals gets assigned to the learning curve. Finnaly, to get two variables within the same range of values, $]0, 1[$, we retrieve the value a and the value $b' = -\frac{4b}{\ln a}$, for (6), or $b' = \frac{b \ln 5}{1-a}$, for (7).

3 Intelligent Tutoring System

But how will the mixture model schedule practice routines for a student? The current method of SIACUA already uses bayesian networks to create a solid way to estimate a student's progress, but there isn't any approach to formulating schedules. Furthermore, the beliefs are just used as a prior for the values of the learning curves and are not sufficient to predict whether or not the student has mastered a certain topic.

So, the goal of the ITS it to gather information about the beliefs and the parameters of the regressions in (1) and schedule practice routines for the students, based on suggestions and spontaneous feedback. Moreover, since various topics are taken into account, the algorithm (5) has to be used in another context, to provide more individual results about each student.

3.1 Variables

Every time a student answers a question, the ITS should calculate and store a series of variables that will help it deciding what to recommend to the student. There are two types of variables: *static variables* and *topic related variables*. The first ones are:

- T_{tutor} — tendency to follow the ITS' recommendations. Let $t \in \{0, 1\}$ be the outcome for a new recommendation that pops up. Considering the vector $\mathbf{t}$ of the outcomes, each t_i is a realization of a random variate $T_i \sim \mathcal{B}(p)$, with $0 < p < 1$, and because the newest values are more significant (t_0 is newer than t_1), if $X \sim \text{Geo}(p)$, then

$$T(p) = \sum_{i=0}^{n-1} t_i \cdot \mathbb{P}(X = i),$$
$$T_{tutor} = \max \left\{ T(p)|_{\frac{\partial}{\partial p}T(p)=0} \right\}.$$

  It's possible to show that

$$\begin{aligned} & 0 = \frac{\partial}{\partial p} T(p) \\ \Leftrightarrow \qquad & 0 = \sum_{i=0}^{n-1} t_i \left(1 - (i+2)p\right)(1-p)^i, \end{aligned}$$

  and after computing the possible values for $p \in]0, 1[$, they are again plugged in $T(p)$ and the maximum of those values will be the tendency T_{tutor}.

- n_s — number of questions answered during the current session. Since this value ranges from 0 to $+\infty$, it's normally interpreted as a Poisson distribution $N \sim \text{Pois}(\lambda_s)$, where λ_s is the average of the number of questions answered each session, that is stored and updated everytime the session ends. It is used its cumulative distribution function to calculate the critical value for which the result is greater than c:

$$\mathbb{P}(N \leq n_c) = \sum_{i=0}^{n_c} \mathbb{P}(N = i),$$
$$\mathbb{P}(N \leq i-1) < c \leq \mathbb{P}(N \leq i) \Rightarrow n_c = i.$$

- t — current time spent on a single question, measured in seconds. The ITS will also keep record of the mean value μ_t, the mean of the squares μ_{t^2} and the sample standard deviation σ_t, updating it everytime a new value of t is stored:

$$\mu'_t = \frac{n_s \mu_t + t}{n_s + 1}$$
$$\mu'_{t^2} = \frac{n_s \mu_{t^2} + t^2}{n_s + 1}$$
$$\sigma_t = \sqrt{\mu_{t^2} - \mu_t^2}.$$

  This way, since the variate that measures time is $T \sim \mathcal{N}(\mu_t, \sigma_t^2)$,

$$Z_t = \frac{T - \mu_t}{\sigma_t} \sim \mathcal{N}(0, 1).$$

  To find the critical value from which the cumulative distribution function has values above c, we calculate

$$\begin{aligned} & \mathbb{P}(Z_t \leq z_c) \geq c \\ \Leftrightarrow \qquad & \mathbb{P}\left(\frac{T - \mu_t}{\sigma_t} \leq z_c\right) \geq c \\ \Leftrightarrow \qquad & \mathbb{P}\left(T \leq z_c \sigma_t + \mu_t\right) \geq c \\ \Leftrightarrow \qquad & z_c \sigma_t + \mu_t = t_c. \end{aligned}$$

Apart from the first variables, the other *static variables* are reseted once the current session ends. The *topic related variables* are:

- T_{miss} — tendency to make a mistake on a question. The calculations made are the same as the ones for T_{tutor}. Considering the error vector $\mathbf{e}$ and $X \sim \text{Geo}(p)$, for $0 < p < 1$,

$$T(p) = \sum_{i=0}^{n-1} e_i \cdot \mathbb{P}(X = i),$$
$$T_{miss} = \max \left\{ T(p)|_{\frac{\partial}{\partial p} T(p)=0} \right\}.$$

- n — number of questions answered. Because this variable is relative to each topic, we don't need to assign a certain distribution because it will just define the shape of our later distributions.

- t_q — current time spent on a single question, measured in seconds. The same variables as in t are stored and we're left with the distribution

$$Z_{t_p} = \frac{T_p - \mu_{t_p}}{\sigma_{t_p}} \sim \mathcal{N}(0, 1),$$

and the critical value is also retrieved with

$$\begin{aligned} & \mathbb{P}(Z_{t_p} \leq z_c) \geq c \\ \Leftrightarrow \qquad & z_c \sigma_{t_p} + \mu_{t_p} = (t_p)_c. \end{aligned}$$

- S — success rate. It's a ratio between the number of correct answers by the number of questions answered. The *lazy* value for the variable would be plain division between the number of right answers and n, but since time has also an important role on the success rate, a more accurate value would be $(1 - p)$, where p is the value assigned when calculating T_{miss}. The critical value for the success p_c is the value that represents stagnation in progress.

- b — current belief, retrieved from the bayesian network. This values are already computed by the web application but the ITS stores them for convenience. If we regard $B \sim \text{Beta}(\alpha, \beta)$ as a *clumsy* variate that models beliefs, where α and β are the number of right and wrong answers, respectively, then as α and β grow large,

$$B \overset{a}{\sim} \mathcal{N}\left(\frac{\alpha}{\alpha + \beta}, \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \right)$$
$$Z_b = \frac{B - \frac{\alpha}{\alpha+\beta}}{\sqrt{\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}}} \sim \mathcal{N}(0, 1).$$

If we assign $\alpha + \beta = n$, then $\alpha \approx nb$ and

$$\frac{\alpha}{\alpha + \beta} \approx \frac{nb}{n} = b,$$
$$\frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \approx \frac{nb(n - nb)}{n^2(n + 1)} = \frac{b(1 - b)}{n + 1}.$$

and with the same calculations with the time variables, the critical value would be

$$\begin{aligned} & \mathbb{P}(Z_b \leq z_c) \geq c \\ \Leftrightarrow \qquad & b + z_c \sqrt{\frac{b(1 - b)}{n + 1}} = b_c. \end{aligned}$$

- b_n — average value of the beliefs from a topic and its neighbours. This variable follows the same logic as b for the normal distribution.

3.2 Decisions

After computing and storing all the variables, the ITS performs addicional calculations to find if it needs to recommend something to the student. To accomplish this, it corresponds each study suggestion with a variate such that all variates follow the same kind of distribution. Since we are dealing with multiple parameters and the ITS should only recommend something after crossing some threshold value, it is used the Generalized Extreme Value distribution for every tip.

In a distribution of the type $A \sim \text{GEV}(\mu, \sigma, \xi)$, the first parameter μ represents the location, the value from which the model starts recognizing that some action shoud be taken. The second, σ, it's the scale parameter, turning the curve wider as it gets bigger. The last one, ξ, it's the shape parameter, that defines the structure of the tail. On this context, we want only positive values (exceptionally 0, also), noticing that the curve gets more steep near μ as ξ increases.

The 6 suggestions are:

- *Learn this topic.* You have little knowledge about it and have answered almost no questions.

$$A_{learn} \sim \text{GEV}\left(b_c, \sqrt{\frac{b(1-b)}{n+1}}, n\right)$$
$$\mathbb{P}_{learn} = \mathbb{P}\left(A_{learn} \leq 1 - b\right)$$

- *Do a quick scan.* You are spending too much time answering these questions and you are answering incorrectly most of the questions.

$$A_{scan} \sim \text{GEV}\big((t_p)_c, \sigma_{t_p}, S\big)$$
$$\mathbb{P}_{scan} = \mathbb{P}\left(A_{scan} \leq t_p\right)$$

- *Revise these concepts.* You are making too many mistakes and you have little knowledge about that topic and neighbouring topics.

$$A_{revise} \sim \text{GEV}\left((b_n)_c, \sqrt{\frac{b_n(1-b_n)}{n+1}}, T_{miss}\right)$$
$$\mathbb{P}_{revise} = \mathbb{P}\left(A_{revise} \leq 1 - b_n\right)$$

- *Deepen this topic.* I see you listen to my recommendations but you still aren't making any progress whatsoever.

$$A_{deepen} \sim \text{GEV}\Big(p_c, \sqrt{S(1-S)}, 1 - T_{tutor}\Big)$$
$$\mathbb{P}_{deepen} = \mathbb{P}\left(A_{deepen} \leq S\right)$$

- *Focus on this topic.* Inspite of answering to a lot of questions about this, you seem to have low knowledge about it.

$$A_{focus} \sim \text{GEV}\left(b_c, \sqrt{S(1-S)}, \frac{1}{n}\right)$$
$$\mathbb{P}_{focus} = \mathbb{P}\left(A_{focus} \leq 1 - b\right)$$

- *Time to take a break.* I see you are spending too much time in every question and you have already answered a lot of questions in this session.

$$A_{break} \sim \text{GEV}\left(t_c, \sigma_t, \frac{1}{n_s}\right)$$
$$\mathbb{P}_{break} = \mathbb{P}\left(A_{break} \leq t\right)$$

After calculating all the probabilities, we perform a t-test on the maximum probability, where the significance level is determined by the beliefs, because the higher the knowledge level in some topic, the

less the need of practice scheduling.

$$\begin{aligned}
\mathbb{P} &= \{\mathbb{P}_{learn}, \mathbb{P}_{scan}, \mathbb{P}_{revise}, \mathbb{P}_{deepen}, \mathbb{P}_{focus}, \mathbb{P}_{break}\}, \\
M_{\mathbb{P}} &= \max\{\mathbb{P}\}, \\
\mathbb{P}' &= \mathbb{P} \backslash M_{\mathbb{P}}, \\
U &= \overline{\mathbb{P}'} + t_{4,b} \frac{s_{\mathbb{P}'}}{\sqrt{5}}.
\end{aligned}$$

If $M_{\mathbb{P}} > U$, then we must reject the null hypothesis and conclude that the action associated with $M_{\mathbb{P}}$ should be recommended, with a significance level of b.

3.3 Tutoring Curves

The only variable that's missing its calculation is the variable c for the critical value. That's when the tutoring curves take place, using the same algorithm (5). However, instead of using data from all students, just the last 10 tests from the student that will perform the next test are taken into account. We initiate the algorithm with three components:

1. a primary component, with random values for ϑ_k, but where the prior for α_k and β_k are the ones calculated by the *clumsy* variate that models the beliefs, using the current belief.
2. a *lucky* component, with random values for ϑ_k, $\alpha_k = 0$ and β_k is equal to the number of questions answered per test.
3. an *unlucky* component, with random values for ϑ_k, $\beta_k = 0$ and α_k is equal to the number of questions answered per test.

By the properties of the algorithm (5), the *lucky* and *unlucky* components model the extreme case where the student could answer all questions right and wrong, respectively, providing curves with very low and very high probabilities. After answering the questions of the new test, the ITS would then apply the results onto the mixture model to find the curve where the student fits the most, recording the percentage of the likelihood of the primary component (represented by $\mathbb{L}$), relative to the others.

$$\begin{aligned}
\mathbb{P}(\mathbf{e}) &= \sum_{k \in \{primary, lucky, unlucky\}} \rho_k \cdot \varpi\left(\boldsymbol{\mathcal{B}}\left(\mathbf{Q}^T \cdot \boldsymbol{\vartheta}_k, \mathbf{e}\right)\right) \\
\mathbb{L} &= \frac{\rho_{primary} \cdot \varpi\left(\boldsymbol{\mathcal{B}}\left(\mathbf{Q}^T \cdot \boldsymbol{\vartheta}_{primary}, \mathbf{e}\right)\right)}{\mathbb{P}(\mathbf{e})}.
\end{aligned}$$

The variable $\mathbb{L}$ measures how consistent the student's practice routines are, so the lower the value, the lower is the confidence that the test is truthful. If the primary tutoring curve is too close to the lucky or unlucky tutoring curves, then we discard the one closest to the primary one.

Then, using exponential or logarithmic regressions, the values for $0 < a < 1$ and $0 < b' < 1$ are calculated to get the critical value c. We place the point (a, b') in the rectangle $R =]0, 1[\times]0, 1[$ and compute the distance from the point to the corner $(1, 1)$. Although, we need to account for $\mathbb{L}$, as the lower it is, the close to the center of the rectangle the points goes. With a bit of vector algebra,

$$\begin{aligned}
& (x, y) = \left(a + (1 - \mathbb{L})\left(\frac{1}{2} - a\right), b' + (1 - \mathbb{L})\left(\frac{1}{2} - b'\right)\right) \\
\Leftrightarrow \quad & (x, y) = \left(\frac{1}{2} + \mathbb{L}\left(a - \frac{1}{2}\right), \frac{1}{2} + \mathbb{L}\left(b' - \frac{1}{2}\right)\right) \\
\Leftrightarrow \quad & (x, y) = \frac{1}{2}\left(1 + \mathbb{L}(2a - 1), 1 + \mathbb{L}(2b' - 1)\right)
\end{aligned}$$

From that, we normalize the distance and

$$c = \frac{\|(x, y) - (1, 1)\|}{\sqrt{2}}.$$

This means that the lower the distance, the lower the critical value for the threshold for the ITS' recommendations. This can produce very high results to all decision thresholds but since we are performing t-tests to ponder which suggestion to make, it just prompts the ITS to schedule practice routines more regularly.

4 Notifications and Constructing Feedback

Apart from the 6 main suggestions, the ITS should also be able to provide notifications and constructive (either positive and negative) feedback in real-time. While the notifications are shown before the test, feedback is shown during each test and not at the end, and uses different criteria for possible suggestions.

Evey time a student clicks on a notification or answers a question, the ITS learns something about that student and, based on that information, provides other notifications and feedbacks that enhances his study performance. We call this the *reward*, and everytime the ITS is prompted to give notifications and feedback, it calculates the one with the highest reward.

4.1 Notifications

Notifications will just pop up on the main page of the web application. They are clickable links that redirect the user to some page or exercise. Every time the student logs on, the notification with highest reward at the moment will appear. Subsequent visits to the main page will not always produce notifications.

4.1.1 Reward

Every notification can be clicked or ignored, and after being clicked it can either result in a success or in a failure. So, for every notification t we have the set of outcomes $c_t = \{0, 1\}$ for the click and $r_t = \{0, 1\}$ for the success.

We say a notification is elegible when it meets certain criteria, like test-streaks and belief values.

4.1.2 List of Notifications

- *You are missing out on <topic>.*
- *Continue your streak on <topic>.*
- *Have you studied <topic> today?*
- *<Random question about one topic>.*

4.2 Feedback

4.2.1 Reward

4.2.2 List of Feedbacks

- *Level Up!* (positive).
- *Level Down...* (negative).
- *Nice streak! Keep it up!* (positive).
- *You missed a few in a row, take your time to answer...* (negative).
- *You're nailing it! Just a few more to go!* (positive)
- *Don't give up, just a few more to go!* (negative)

4.3 Test Layout

Each test consists on a number N of questions of progressing difficulties. Consecutive right and wrong answers lead to going up or down levels of difficulty, respectively. After each question, the ITS computes the feedback with highest reward and provides it. The

5 Programming

Since SIACUA is programmed in C♯, the ITS will also be constructed in that language, as a Class Library (.dll file), to be then used as a reference for this and other web applications.