On The Robustness of Stealth Assessment

Konstantinos Georgiadis, Giel van Lankveld, Kiavash Bahreini, and Wim Westera

***Abstract*—Stealth assessment is a methodology that utilizes machine learning for processing unobtrusively collected data from serious games to produce inferences regarding learners' mastery level. Although stealth assessment can produce valid and reliable assessments, its robustness over a wide a range of conditions has not been examined yet. The main reason is its complex, laborious, and time-consuming practical application. Therefore, its exposure to different conditions has been limited, as well as its wider uptake from the serious game community. Nevertheless, a framework for developing a generic tool has been proposed to lift its barriers. In this study, a generic SA software tool was developed based on this framework to examine the robustness of the stealth assessment methodology under various conditions. In specific, the conditions relate to (a) processing datasets of different distribution types and sizes (960 datasets containing a total of 72.336.000 data points are used for this reason), (b) utilizing two different machine learning algorithms (Gaussian Naïve Bayes Network and C4.5), and (c) using statistical models relating to two different competency constructs. Results show that stealth assessment is a robust methodology, whilst the generic SA tool is a highly accurate tool capable of handling efficiently a wide range of conditions.**

***Index Terms*—generic tool, machine learning, robustness, serious games, simulation, stealth assessment**

I. Introduction

SERIOUS games have been framed in recent years as one of the most promising alternatives to traditional education primarily for learning and training purposes [1]. Well-designed serious games can enable active learning [2] within engaging environments that can enhance learners' intrinsic motivation [3] and support the development of 21st century and other skills [4]. Furthermore, serious games can facilitate effective assessments [5] since detailed learner traces can be collected during gameplay in order to classify learners' performances.

Submission date for review: 28 March 2019. This research was funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 644187, the RAGE project (www.rageproject.eu).

K. Georgiadis is with the Open University of the Netherlands, Valkenburgerweg 177, 6419AT Heerlen, The Netherlands (e-mail: konstantinos.georgiadis@ou.nl).

G. van Lankveld was with the Open University of the Netherlands, Valkenburgerweg 177, 6419AT Heerlen, The Netherlands. He is now with Fontys Applied University of Eindhoven, De Lismortel 25, 5612 AR Eindhoven, The Netherlands (gielvanlankveld@protonmail.com).

K. Bahreini is with the Open University of the Netherlands, Valkenburgerweg 177, 6419AT Heerlen, The Netherlands (e-mail: kiavash.bahreini@ou.nl).

W. Westera is with the Open University of the Netherlands, Valkenburgerweg 177, 6419AT Heerlen, The Netherlands (e-mail: wim.westera@ou.nl)

However, to ensure the validity and reliability of assessments in serious games [6, 7], principled assessment design frameworks must be used. These frameworks facilitate the design of conceptual constructs for competencies (i.e. skills, abilities, etc.) and in-game tasks, as well as computational models that express the relationship of evidence (i.e. data) collected during gameplay to these conceptual constructs. One of the most prominent assessment methodologies coupled with such frameworks is referred to as stealth assessment (SA) [8].

SA is an assessment methodology that allows for the unobtrusive collection and computational analysis of meaningful evidence during gameplay to provide probabilistic reasoning over learners' mastery level by using machine learning (ML) technology. It combines the use of (1) a principled assessment design framework, which is the Evidence-Centered Design (ECD) approach [9, 10] along with (2) ML technology. So far, SA has shown to be a valid and reliable assessment solution in diverse domains, such as physics [11], persistence [12], and problem-solving [13]. Despite these existing proof cases, its practical application is still problematic, requiring a complex, laborious and time-consuming process [14]. The complexity of SA relates to the expertise that is needed such as knowledge in game development and design, ML algorithms, instructional design, learning materials, psychometrics, statistics, etc. The laboriousness of SA relates to the fact that so far it has only been developed as part of a specific game's source code. Such hardcoded, game-specific solutions cannot readily be transferred from one game to another. Therefore, applying SA for new assessment needs requires software development and validation from scratch. Of course, this process is considerably time-consuming and prone to mistakes. Supportive tools for the wider accommodation of SA or the replication of SA in other games have not been available.

To lift the barriers of SA, a framework for developing a generic tool for SA has been proposed [15]. That is, a stand-alone software tool that (1) supports the handling of numerical data from any serious game, (2) automates the required ML processes, and (3) allows the easy arrangement of different ECD models (depending on the competency at hand). In this way, the expertise, labor, and development time needed to create and apply SA would be drastically reduced. Consequently, the costs for applying SA in serious games can be reduced or even be eliminated, which would amplify the adoption of SA by the serious game community. Moreover, the availability of such SA tool would allow investigating the applicability of the SA methodology in a systematic way, rather than collecting incidental evidence on a case-by-case

basis.

To this end, a generic stealth assessment tool (namely GSAT) has been developed according to the aforementioned framework. This tool allows us to examine the robustness of the SA methodology against various conditions of the ML approaches. These conditions account for the variability of outcomes that can occur when applying SA in serious games, and thus reflect a wide spectrum of use cases. The variability relates to (1) the data, (2) the ML algorithms, and (3) the ECD model variances. Accordingly, this paper aims to answer the following research questions:

- How robust is the SA methodology when handling datasets of different normality significance levels?
- How robust is the SA methodology when handling datasets of different sample sizes?
- How robust is the SA methodology when using different ML algorithms for numerical datasets?
- How robust is the SA methodology when handling different ECD models?

To address these research questions, a large variety of simulation datasets were generated encompassing as many as possible different cases, in particular reflecting different sample sizes and different normality significance levels. Two principal ML algorithms were used to produce inferences with respect to the data contained in the datasets, (1) a parametric ML algorithm called Gaussian Naïve Bayesian Network (GNBN), and (2) a non-parametric ML algorithm, which is a Decision Tree called C4.5, respectively. Finally, ECD models for two different hypothetical competencies were examined. Both a two-dimensional and a three-dimensional competency construct were devised, assigned to a fixed number of observables to form respective statistical models. The number of the observables was fixed across all the examined conditions to be able to draw safe conclusions from the results.

In the following of this study, background information about SA is provided in section II. Information about the generic SA tool is presented in section III. The methodology that was used to answer the posed research questions can be found in section III. Section IV presents the results of this study. The results are discussed in section VI, while the conclusions and future research plans are in section VII.

II. Stealth Assessment Background

As described before, SA consists of two main ingredients. That is: (a) the ECD framework for arranging assessments based on valid and reliable constructs and (b) ML technology to enable the probabilistic assessment of the learners' mastery level on a certain competency.

A. Evidence Centered-Design (ECD)

ECD is a principled assessment design framework that includes several essential elements for modeling the assessment process in a valid and reliable manner. These elements are: (1) the competency model, (2) the task model, and (3) the evidence model. The competency model describes the competency construct, which includes the underlying factors (i.e. facets, sub-skills, etc.) that constitute the competency to be assessed. The task model describes a set of in-game tasks that can allow the elicitation of proper evidence with respect to the competency construct. The evidence model is a link between the competency model and the task model, which describes the relationship of the observed in-game performance (i.e. observables) to both the in-game tasks and the underlying competency constructs. For this reason, it consists of two different sub-models: (1) the evidence rules and (2) the statistical model. The evidence rules describe the relationship between the observed performances and the in-game tasks, while the statistical model describes the relationship between the observed performances and the competency construct.

Mislevy and colleagues [19] were the first to describe ECD as a generic methodology for developing assessments. Thereby, the conceptual models of ECD can act as generic definers for describing detailed assessment elements. This means that ECD can host competency models of any shape and size (consisting of any number of subordinate facets, sub-facets, etc.), task models consisting of any number of tasks, and evidence models consisting of any number of observables (and respective mappings of these observables to components of the competency and task models). Since SA resorts to the ECD for arranging assessments in serious games, it thus also represents a generic assessment methodology.

B. Machine Learning Technology

In education it is very common to use scoring systems based on test items (such as self-report questionnaires and multiple-choice tests) for assessment purposes. These allow the teachers to grade the learners and determine their knowledge states. While this assessment process is well-established, it has certain limitations [16]. For example, it is not capable of testing for imponderables such as critical thinking, creativity, and other 21st century skills, especially in time-limited examinations, thus rendering it rather inadequate as a means for preparing learners towards lifetime learning and professional success in the 21st century.

Nevertheless, as we gradually transit to an age of an increasing digitalized education more opportunities arise for accessing far richer learner data than what was possible in traditional classrooms. Here, ML would allow for extracting meaningful information from learners' data traces, by first learning from this rich data and accordingly analyzing it to provide inferences about mastery of competencies. Within the field of serious games, SA proposes the use of ML technology for classifying learners' performances by utilizing data related to their in-game behaviors and decisions, which is logged during gameplay. Originally, Bayesian Networks were preferred [8], but also other ML algorithms such as Decision Trees, Neural Networks, Logistic Regression, Support Vector Machines, and Deep Learning have been examined for SA [17, 18].

III. A generic Stealth Assessment tool

So far, in the literature only hard-coded solutions of SA in

case-specific applications have been reported. That is because SA has been originally viewed as a methodology that should be directly integrated within the gaming environment [20]. This point of view has served well the purpose of empirical studies that aimed at providing a proof of concept for SA. But the lack of generic tools that would support the definition and implementation of SA in serious games has hampered both the systematic investigation of SA under various boundary conditions and the adoption of SA by serious games practitioners [14].

To lift these barriers, a stand-alone software tool was developed that implements the SA methodology in a generic format, that is, it allows for dealing with datasets of different distribution types and sizes, different competency constructs and different ML algorithms. In this paper, the tool is used to investigate the robustness of the SA methodology against different boundary conditions. The SA tool was developed as a client-side console application in the C# programming language using the .NET framework. In the following subsections a description of the software tool is presented. That includes a user case that elaborates its workflow design as well as the external libraries that were used to realize it.

A. SA Use Case Description

To set-up and run a SA, assessment experts, being the primary users, engage in various interactions with the SA tool. Fig. 1 depicts the SA workflow design from this user perspective. Upon start, the user selects whether to initiate a new SA or not (step 1). If not, then the workflow terminates. Else, the system automatically loads data (viz. game log data) from a spreadsheet file located in a pre-defined file path (step 2). It should be noted that in current version of the tool certain assumptions are made about the data contained in the spreadsheet file (e.g. the data is numerical and ordered in ascending order) to avoid unhandled exceptions. When done, the system normalizes the data so that the values of all the observables (i.e. game variables) scale from 0 to 1 (step 3). In this way, the observables align with each other on a common scale. Then, the system checks whether the data is labeled or not (step 4), since unlabeled datasets cannot be handled by supervised ML algorithms such as GNBN and C4.5. If the data is unlabeled, the system runs a clustering-based method for automatically assigning labels to the data (step 5). This method is explained in section IV. Next (step 6), the system proceeds with automatically loading ECD models from a pre-defined configuration file. The configuration file (.config format) can automatically be generated by the software tool since it allows its user to easily create it through its user interface. Once the ECD models are loaded in the system, the user can select which of the implemented ML algorithms should be used (step 7). So far, this is either GNBN or C4.5 for the purpose of this study, but other ML algorithms can be readily added to the tool. Thereafter, the system runs the selected ML algorithm based on pre-defined optimizations that fit the needs of this study (step 8). In later versions of the tool user-defined optimizations will be allowed to enhance flexibility and usability (e.g. allowing the user to set the

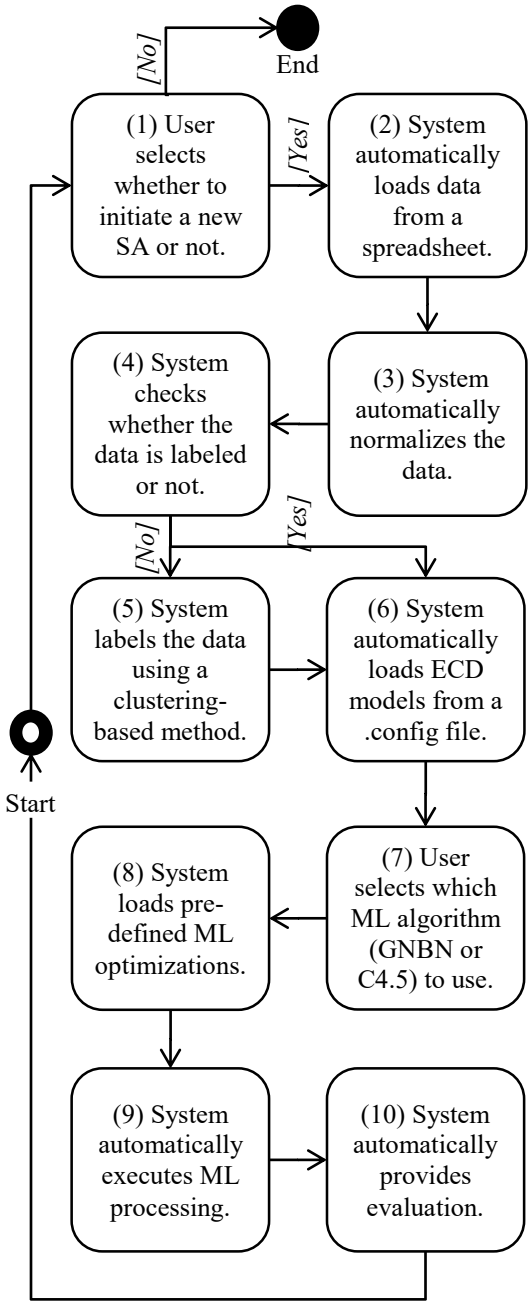


Fig. 1. Workflow of SA application

tolerance level of the selects ML algorithms and adjusting the percentage split for the training and testing datasets). In step 9 the tool executes the selected ML algorithm and in step 10 it accordingly provides output regarding both the learners' performances (the assessment) and the performances of the ML algorithms (which allows for comparing the results for different conditions).

B. External libraries

A set of libraries was imported to implement the SA tool: *EPPlus* was used to enable importing data from spreadsheets (Excel files), while the *Accord.NET* framework was used to

apply the machine learning functionalities. On a side note, it useful to mention that Excel is compatible with data streams generated with standards for learning analytics from serious games, such as xAPI

IV. Research Methodology

This section presents the methodology that was used to examine the robustness of SA.

A. Experimental Setup

Several conditions are examined within this study in order to examine the robustness of SA. The setup for examining these conditions includes the use of both a configuration file (containing hypothetical ECD models) and spreadsheets (containing the simulation datasets) as inputs to the generic SA software tool. A total of 960 different simulation datasets were generated and entered into the tool (one run of the software required for each) to obtain the necessary outputs concerning the performance of the ML algorithms across all conditions. Thereafter the outputs were analyzed through several R scripts that were developed for answering the posed research questions. A schema of the experimental setup is presented in Fig. 2.

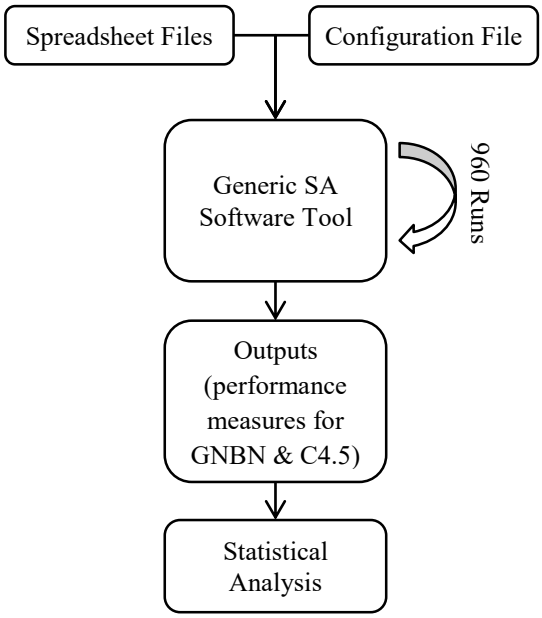


Fig. 2. A view of the experimental setup.

B. ECD Models

As previously mentioned, ECD is a conceptual framework, which can be utilized (among other things) to design competency constructs. These tree-like constructs can branch to form constructs of any size and shape. Of course, testing the software tool against any possible construct is not feasible due to the infinite possible ECD model variations. Therefore, we decided to include in this study elementary constructs, which could be easily extended at a later stage since the software tool has no restrictions on the scalability of the competency constructs. These competency constructs are hypothetical to the extent that they represent abstracted, de-contextualized container structures not directly bound to specific domains or skills. The first competency is composed of two separate facets (sub-competences in the tree), while the second competency is composed of three separate facets.

For each competence a set of observables should be specified, making up the associated statistical models of the ECD. The hypothetical statistical models were set to include eleven conditionally independent observables (four for the two-facet construct and seven for the three-facet construct, respectively. These observables were intuitively mapped to the facets of the competency constructs, thus defining the statistical models (cf. Fig 3.). The task models and evidence rules were not relevant for this study since these models are only relevant for designing the serious game. In this study however we solely focus on the testing of a software tool that deals with the measurement, computation, and analysis of learning from gameplay data, that is, not with game design aspects that can potentially elicit desired in-game behaviours given a specific game case. Thus, it was unnecessary to include them here.

C. Generation of Datasets

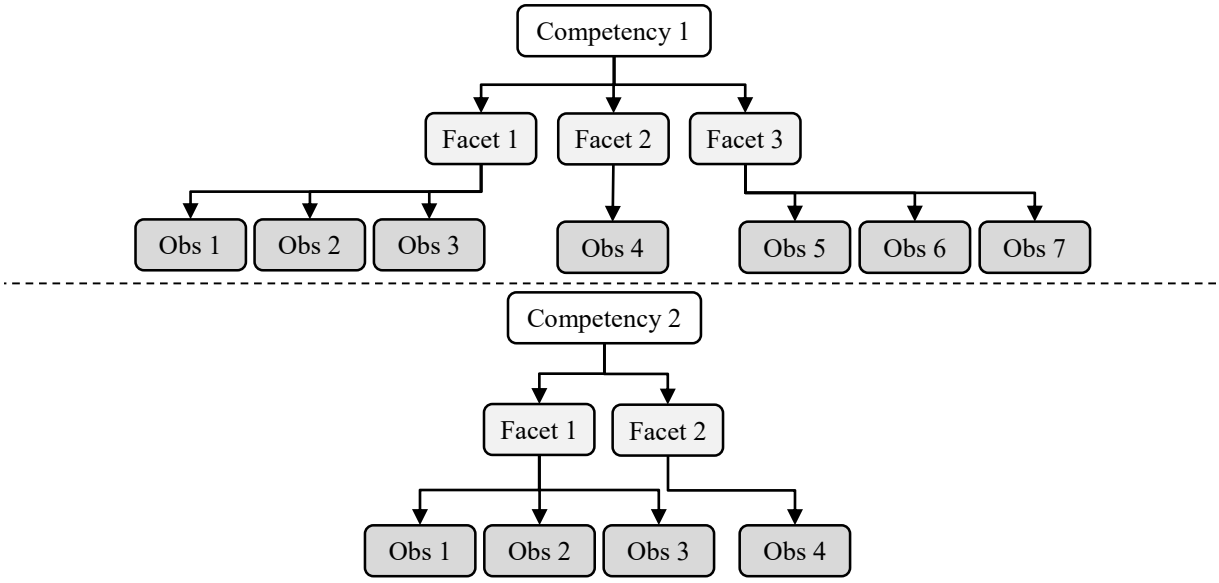
Several R scripts were developed to generate the necessary simulation datasets and store these in spreadsheet files for processing by the SA tool. These files include continuous data of different distribution types and sample sizes. For our purposes, the use of generated datasets is superior to using real-world game data, because the latter generally lack scale and do not allow for differentiated inputs to investigate robustness under various conditions. In other words, we argue that a simulation approach is ideal for testing Smart CAT's operational robustness since suited real-world learning data from serious games is hard to find or collect (especially in large volumes), and also harder to control and adjust for the examined test conditions. The test conditions of this study were examined by using 80 datasets per condition, requiring a total of 960 spreadsheet files (simulation datasets). The full set of spreadsheet files contained a total of 72.336.000 data points. Below we explain the different testing conditions.

1) Testing Under Violations of Normality

ML algorithms such as Bayesian Networks assume that the data is normally distributed. In educational practice, however, it is likely that log datasets will deviate from the normality requirement, which may affect the reliability of SA outcomes. To investigate the robustness under normality violations 8 test conditions (hence 640 datasets each containing 10,000 data points per observable for both competencies) with different normality significance levels were examined.

To achieve this, 80 datasets were generated per condition; each containing data for every declared observable within the following normality significance level (*p value*) intervals: (a) $p \in [1.0, 0.8)$, (b) $p \in [0.8, 0.6)$, (c) $p \in [0.6, 0.4)$, (d) $p \in [0.4, 0.2)$, (e) $p \in [0.2, 0.1)$, (f) $p \in [0.1, 0.05)$, (g) $p \in$

Fig. 3. A view of the two statistical models (Obs stands for observable).



$[0.05, 0.01)$, and (h) $p \in [0.01, 0)$. A single-sample Anderson-Darling normality test was performed every time that a new set of data points was included in the dataset for an observable to examine the *p value*. Each time that the boundary conditions were not met a new set of random data points were generated by using a different seed value. The mean $\bar{x}$ values of these observables were arbitrarily set to 5, 100, 10, 250, 50, 3,000, 500, 15, 150, 1,000, and 200 respectively, be it that the SA tool directly procures normalization after importing the files into it (cf. step 3 in Fig. 1).

2) Testing of Reduced Sample Sizes

As previously mentioned, 10,000 data points were assigned on each of the 11 observables declared within the spreadsheet files. However, real-world datasets from serious games are usually smaller in size. For this reason, we also generated datasets of both 1,000 and 100 data points per observable. All these data points were randomly sampled from normal distributions ($p \in [1.0, 0.8)$). To examine how the system would react to non-normality, we also generated datasets of highly non-normal distributions ($p \in [0.01, 0)$) at these sample sizes. Thus, a total of 320 spreadsheet files were additionally generated to examine these 4 conditions (80 files per condition).

D. Machine Learning Algorithms

Two ML algorithms were implemented at the SA tool, namely a GNBN and a C4.5. This allows to compare the performance of a parametric ML algorithm (GNBN) to a non-parametric one (C4.5) in all set conditions. These conditions deliberately violate some of the statistical assumptions holding, especially for the GNBN which is regarded to be working efficiently only in conditions of normality.

The two ML algorithms were optimized to fit the needs of this study. Firstly, they were tuned to classifying learners' performance in 3 different levels (Low, Medium, and High) since SA allows for non-binary assessment outputs due to its probabilistic nature. Nonetheless, even more levels of classifications could be defined depending on the assessment needs at hand. For this study, we opted for a minimum of classification output diversion that is in accordance to previous empirical studies for SA [11, 12, and 13]. Secondly, a regularization factor was set for the GNBN at 0.00001 in order to avoid zero variances. In addition, a random seed value was set to randomize the data before training the two ML algorithms. Thirdly, a split rule was applied on both the algorithms for training (66%) and testing (34%) the data. Lastly, we made sure that at least one data point would be assigned to each class for training the classifiers. The rest of the training dataset was randomly sampled from the data pool.

E. Pre-Processing of the Data

During runtime, the SA tool pre-processed the datasets, in order to automatically label the data. That is, in contrast to existing empirical SA studies where experts are used to label the data. However, we opted to use a data-driven approach for labelling the data firstly because the data is de-contextualized and therefore cannot be meaningfully annotated by experts, and secondly because we consider this approach to be more valid, unbiased, and generic. The pre-processing step included the normalization of the data followed by a clustering approach for labeling the data.

1) Normalization of data

In compliance with realistic datasets, each observable in the simulation datasets was set to include values of different ranges since each observable measures different in-game activities. For example, observables could represent "*how much time was spend on a task*", or "*how many times an action was performed*". In this study, the simulated observables were encoded as "Obs1", "Obs2", … , "ObsX" (see Fig. 3) since they serve as non-contextualized numerical variables that refer to hypothetical competency constructs. In

unidimensional constructs (meaning: with only one observable) it is not a big problem to cluster such data and assign labels to it (given that the data is already provided in an ascending order). However, in multi-dimensional constructs the data spreads in multi-dimensional space. In that case, unequal ranges of values (scales) are assigned to the observables that introduce undesirable weights to them (additional to any factor loadings). Hence, assigning labels on clusters of data becomes troublesome. To resolve this issue a normalization step was applied prior to the clustering of the data so that the data of each observable scaled from 0 to 1.

2) Clustering Approach for Labelling

A clustering algorithm called k-means was used to label the data. The k-means algorithm partitions the data into k clusters, where each data point belongs to the cluster with the nearest mean (i.e. centroid) value. In this study, the k-means algorithm was optimized to cluster the data into three ($k=3$) clusters, in alignment to the 3 performance classes that are used by the ML algorithms. The Euclidean Square Distance Metric was used for the distance function, while the tolerance value for cluster changes between two iterations of the algorithm was set to 0.05. Also, the initial centroids of the clusters were randomly assigned.

Although, k-means is able to group the data into separate clusters, it is not able to decide which of these clusters show higher or lower performance (i.e. to classify learners). In this particular case, this is primarily due to the fact that the data is spread in multi-dimensional space, which in turn leads to incomparable centroid values even after normalizing the data so that it ranges in a common scale across all observables. To order the clusters from Low to High and be able to classify the data accordingly, a weighted function was used to average the variable (observable) values within each cluster and hence created comparable centroid values between the clusters. Thus, Equation 1 was applied on each cluster:

$$centroid_k = \frac{w_1 var_1 + \cdots + w_x var_n}{n} \quad (1)$$

, where k is the cluster, $centroid_k$ is the centroid value of cluster k, n is the number of variables (observables), and w_x are the weights/factor loadings ($\sum_{x=1}^{n} w_x = 1$). In this study we assumed equal weights for the variables, since we deal with simulation data and hypothetical generic constructs.

F. Performance Measures

Since, the purpose of this study is to compare the performances of the ML algorithms under different conditions; several ML performance measures [21] were used. We used the classification accuracy (CA), the kappa statistic (KS), the mean absolute error (MAE), the root mean squared error (RMSE), the relative absolute error (RAE), and the root relative squared error (RRSE).

G. Statistical Analysis

Various R scripts were developed to statistically analyze the performance of the ML algorithms in all test conditions. One of the most important aspects of this analysis was to examine if the performances of the ML algorithms were statistically stable within a certain level of confidence and thus ensure that the results of this study are reliable overall. In detail, an emphasis was given to the statistical stability of the two ML algorithms with respect to their mean CAs (due to it being the prevalent ML performance measure).

Hence, within each condition we calculated the cumulative mean CAs for both ML algorithms after each run to examine whether or not they would start to converge and stabilize. Nonetheless, time limitations occurred due to the excessive amount of runs that had to be carried out. Hence, we decide to generate no more than 80 runs per examined condition. Consequently, the cumulative mean CAs of the two ML algorithms were calculated after each of the 80 runs per condition.

In addition, a sequential stopping rule [22] was used in reverse to acquire the approximate confidence interval regarding the mean CA of the two ML algorithms. This stopping rule allows to determine how many simulation runs/datasets are needed (per condition) to achieve statistical stability according to a pre-defined precision requirement value (in this case the accuracy bound for the classifiers) and a confidence level value η. However, since the datasets assigned on each condition were already pre-defined at $q=80$, we directly applied the stopping rule only once per condition with a confidence coefficient value of $\eta=0.99$ (i.e. 99%). In this case, the precision requirement value was not required. We rather directly calculated the approximate confidence interval μ values for the mean CA values of the two ML algorithms. To do so, we first calculated the mean CA of the two ML classifiers, the mean variance of the datasets, and the quantile of the t-distribution.

Then, Equation 2 was used the approximate confidence interval μ for each test condition:

$$(\bar{X}_q - t_{\eta,q-1}\sqrt{\frac{S_q^2}{q}},\ \bar{X}_q + t_{\eta,q-1}\sqrt{\frac{S_q^2}{q}}) \quad (2)$$

, where $t_{\eta,q-1}$ is the $(1+\eta)/2$ quantile of the t-distribution with q-1 degrees of freedom, S_q^2 is the sample variance, and $\bar{X}_q$ is the mean CA of each ML algorithm after 80 runs.

H. R Packages

Several R packages were used within the R Scripts for generating the datasets and for the statistical analysis of the results. The most important ones are: *FAdist* [23] for the Weibull distribution, *stats* [24] for the other distributions, and *nortest* [25] for the Anderson-Darling normality test. Among other packages, *xlsx* [26], *data.table* [27], and *ggplot2* [28] were also used.

V. Results

This section presents the results for all the test conditions described above. It is important to note that all results presented here have been rounded (to either one or two

decimal points) to improve their readability.

A. SA Robustness under Violations of Normality

For 8 cases of normality violation a total number of 640 spreadsheets were generated to store data relating to the two hypothetical competencies (80 per distribution type). Accordingly, 640 simulation runs were performed to obtain the performance of the two ML classifiers across 8 different conditions (each condition includes both competencies and both ML algorithms).

An exemplary diagram showcasing results in one of the conditions is shown in Fig. 4. Similar results were found for all of the conditions. In all cases, the cumulative mean CAs for both ML algorithms started to converge and stabilize over simulation runs.

Results regarding the mean performance (all measures included) of the two classifiers in the 8 different distribution-related conditions for Competencies 1 and 2 can be found in Tables 1 and 2, respectively. The results show that the generic SA tool managed to reach high mean CAs when using either GNBN or C4.5 for the assigned ECD models. In particular, for Competency 1 mean CAs ranging from 92.5% to 94.5% were reached when using the GNBN and from 88.3% to 89.9% when using C4.5. For Competency 2 the mean CAs ranged from 95.2% to 96.8% for GNBN and from 91.1% to 94.3% for C4.5. Concerning the rest of the ML performance measures, high KSs and low error rates (MAE, RMSE, RAE, and RRSE) were reached, thus further supporting the efficiency of the classifiers and the effectiveness of the generic SA tool.

In addition, an overview of the mean CA of the two ML algorithms for the two competencies can be found in Fig. 5. The error bars in this figure describe the related approximate CA confidence intervals that emerged from applying Equation 2 and their location coordinates on the x-axis is given by the mean normality significance level (*p value*) intervals. As can be seen in Fig. 5, the CA of the two ML algorithms remains relatively stable despite using data of different normality significance levels. A detailed description of the ML performance interval values (concerning the CA of the ML algorithms) for each condition can be found in Tables 3 and 4 for the two competencies, respectively. These tables not only include the (lower and upper) bounds of the confidence intervals, but also the sample average $\overline{X}_q$, and the mean variance S_q^2 of the sample in order to provide a better insight regarding the statistical stability of the results.

B. Results at Smaller Sample Sizes

The results of this section refer to the datasets of 10,000, 1,000, and 100 samples (i.e. records), that each encompass conditions of both normality ($p \in [1.0, 0.8)$) and non-normality ($p \in [0.01, 0)$), respectively. Similar to the cases described above, the cumulative mean CA of the two ML algorithms were calculated after each simulation run. After several runs the CA values of the two ML algorithms started to converge and stabilize.

Regarding the mean performance measures of the two classifiers when dealing with datasets of different sample sizes, the results show that the generic SA tool is effective regardless of the normality significance level of the data. Results when using normally distributed simulation data are presented in Table 5 and 6. For Competency 1 the mean CA ranged from 92.1% to 94.9% when using the GNBN and from 88.5% to 89.1% when using C4.5. For Competency 2 the CA ranged from 95.4% to 96.5% when using GNBN and from 90.8% to 92.0% when using C4.5.

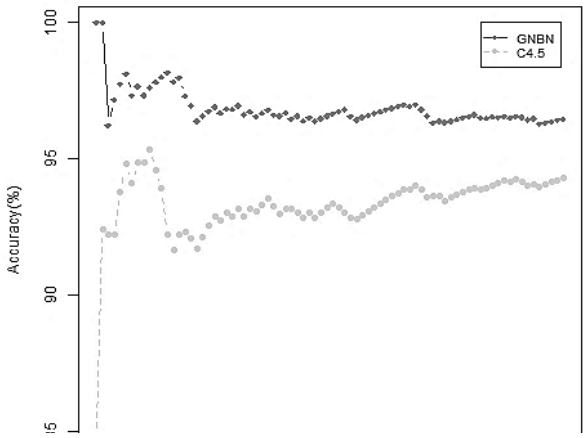


Fig. 4. An exemplary plot depicting the cumulative mean CA values of the two classifiers over 80 simulation runs for one of the declared competencies in one of the test conditions.

Results from non-normal data distributions are presented in Table 7 and Table 8. The mean CA for Competency 1 ranged from 91.8% to 94.7% when using GNBN and from 88.0% to 91.4% when using C4.5. For Competency 2 the mean CA spanned from 95.9% to 96.5% when using GNBN and from 91.9% to 93.0% when using C4.5. Considering the rest of the ML performance measures, high KSs and low error rates (MAE, RMSE, RAE, and RRSE) were once again reached for all the tested conditions.

The mean CAs (along with the confidence intervals) of the two ML algorithms for normally distributed data in three sample sizes are depicted in Fig. 6 for both competencies. Likewise for the non-normal conditions the results are illustrated in Fig. 7. Results from the statistical stability analysis in conditions of normality can be found in Tables 9 and 10 for the two competencies respectively. Accordingly, results in conditions of non-normality for the two competencies can be found in Tables 11 and 12.

Table 1. Mean values of various performance measures for Competency 1 from different data distribution-related conditions.

GNBN						
p	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
0.8-1.0	93.0	0.9	0.09	0.32	13.2	39.5
0.6-0.8	93.8	0.9	0.08	0.28	11.6	35.0
0.4-0.6	92.5	0.9	0.11	0.34	15.0	42.0
0.2-0.4	94.5	0.9	0.07	0.26	10.4	32.4
0.1-0.2	93.2	0.9	0.09	0.31	13.6	38.7

0.05-0.1	94.5	0.9	0.07	0.26	10.6	31.9
0.01-0.05	93.6	0.9	0.09	0.30	12.0	37.0
0-0.01	93.0	0.9	0.10	0.33	14.2	41.8
C4.5						
p	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
0.8-1.0	88.8	0.8	0.15	0.45	21.5	55.1
0.6-0.8	89.0	0.8	0.15	0.42	21.0	52.0
0.4-0.6	89.4	0.8	0.14	0.42	19.9	52.0
0.2-0.4	89.3	0.8	0.15	0.44	20.3	53.4
0.1-0.2	89.9	0.8	0.14	0.40	19.4	49.8
0.05-0.1	88.3	0.8	0.15	0.44	21.6	54.2
0.01-0.05	89.1	0.8	0.15	0.43	20.5	52.3
0-0.01	88.8	0.8	0.15	0.43	21.0	53.1

Table 2. Mean values of various performance measures for Competency 2 from different data distribution-related conditions.

GNBN						
p	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
0.8-1.0	95.9	0.9	0.07	0.21	9.0	25.4
0.6-0.8	96.8	0.9	0.05	0.18	7.6	23.0
0.4-0.6	96.6	0.9	0.06	0.19	8.2	24.5
0.2-0.4	95.2	0.9	0.07	0.22	10.6	28.0
0.1-0.2	96.1	0.9	0.06	0.18	8.3	23.9
0.05-0.1	96.5	0.9	0.05	0.17	7.6	21.5
0.01-0.05	95.7	0.9	0.07	0.21	9.6	26.7
0-0.01	95.9	0.9	0.07	0.21	9.9	27.7
C4.5						
p	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
0.8-1.0	91.9	0.9	0.11	0.32	14.8	39.5
0.6-0.8	93.1	0.9	0.09	0.26	12.8	31.9
0.4-0.6	93.0	0.9	0.09	0.26	13.1	33.2
0.2-0.4	91.4	0.9	0.10	0.28	14.9	36.2
0.1-0.2	92.1	0.9	0.10	0.26	14.0	33.3
0.05-0.1	94.3	0.9	0.07	0.20	10.0	25.3
0.01-0.05	91.1	0.9	0.11	0.30	15.9	37.8
0-0.01	92.0	0.9	0.09	0.27	13.9	34.1

Table 3. Statistical stability results (including CA confidence interval bounds, sample average $\bar{X}_q$, and sample mean variance S_q^2) regarding the performance of GNBN and C4.5 for Competency 1.

GNBN				
p	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
0.8-1.0	91.6	94.5	93.0	24.0
0.6-0.8	92.3	95.2	93.8	24.2
0.4-0.6	90.9	94.1	92.5	29.6
0.2-0.4	93.2	95.8	94.5	19.9
0.1-0.2	91.7	94.7	93.2	25.5
0.05-0.1	93.1	95.9	94.5	23.7
0.01-0.05	92.2	95.1	93.6	24.7
0-0.01	91.5	94.5	93.0	26.0
C4.5				
p	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
0.8-1.0	87.5	90.2	88.8	21.3
0.6-0.8	87.4	90.7	89.0	32.0
0.4-0.6	88.0	90.9	89.4	23.0
0.2-0.4	87.9	90.8	89.3	22.7
0.1-0.2	88.2	91.7	89.9	34.3
0.05-0.1	86.7	89.7	88.3	24.1
0.01-0.05	87.6	90.6	89.1	26.4
0-0.01	87.2	90.4	88.8	28.1

Table 4. Statistical stability results (including CA confidence interval bounds, sample average $\bar{X}_q$, and sample mean variance S_q^2) regarding the performance of GNBN and C4.5 for Competency 1.

GNBN				
p	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
0.8-1.0	94.3	97.4	95.9	27.6
0.6-0.8	95.5	98.2	96.8	21.0
0.4-0.6	95.2	98.0	96.6	23.2
0.2-0.4	93.4	96.9	95.2	35.2
0.1-0.2	94.6	97.6	96.1	24.9
0.05-0.1	94.9	98.0	96.5	26.7
0.01-0.05	94.2	97.2	95.7	25.3
0-0.01	94.4	97.4	95.9	27.1
C4.5				
p	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
0.8-1.0	90.1	93,8	91.9	39.2
0.6-0.8	90.9	95.2	93.1	53.5
0.4-0.6	90.9	95.2	93.0	54.1
0.2-0.4	89.0	93.8	91.4	64.7
0.1-0.2	89.9	94.4	92.1	57.8
0.05-0.1	92.3	96.3	94.3	47.5
0.01-0.05	88.9	93.3	91.1	55.2
0-0.01	89.9	94.1	92.0	49.9

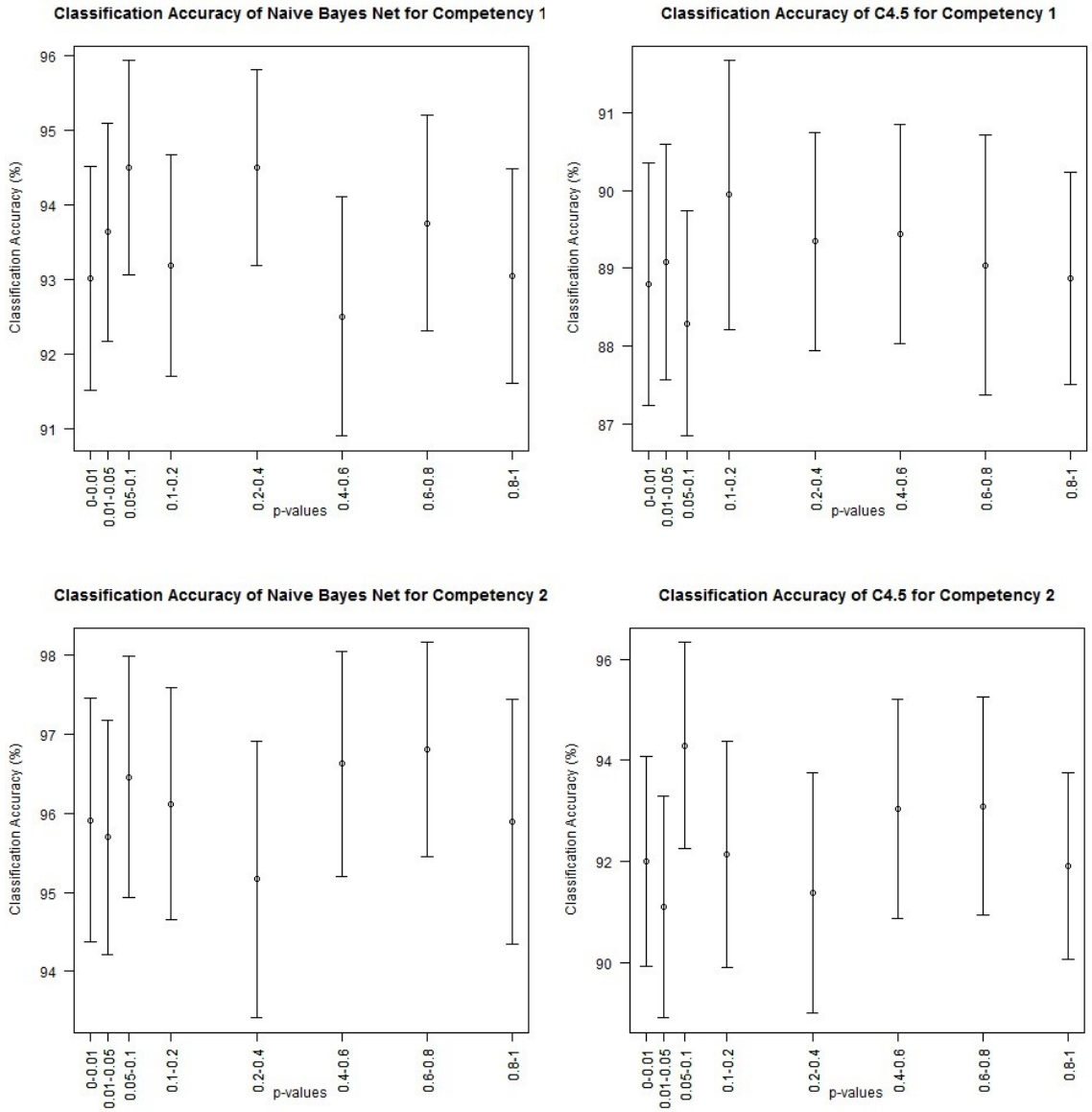


Fig. 5. Mean CA values and confidence intervals of the two ML algorithms (GNBN left, C4.5 right) for Competency 1 (top) and Competency 2 (bottom) respectively.

Table 5. Mean values of various performance measures for Competency 1 for different sample sizes of normally distributed data ($\boldsymbol{p} \in [\mathbf{1.0}, \mathbf{0.8})$).

GNBN						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	93.0	0.9	0.09	0.32	13.2	39.5
1,000	94.9	0.9	0.07	0.24	9.5	29.2
100	92.1	0.9	0.11	0.31	15.8	39.5
C4.5						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	88.8	0.8	0.15	0.45	21.5	55.1
1,000	88.5	0.8	0.15	0.42	21.8	52.4
100	89.1	0.8	0.14	0.39	20.1	48.2

Table 6. Mean values of various performance measures for Competency 2 for different sample sizes of normally distributed data ($\boldsymbol{p} \in [\mathbf{1.0}, \mathbf{0.8})$).

GNBN						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	95.9	0.9	0.07	0.21	9.0	25.4
1,000	96.5	0.9	0.05	0.17	7.7	21.4
100	95.4	0.9	0.07	0.19	10.0	25.6
C4.5						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	91.9	0.9	0.11	0.32	14.8	39.5
1,000	92.0	0.9	0.10	0.28	14.5	36.0
100	90.8	0.8	0.11	0.29	16.3	37.2

Table 7. Mean values of various performance measures for Competency 1 when using different sample sizes of non-normally distributed data ($\boldsymbol{p} \in [\mathbf{0.01}, \mathbf{0}]$).

GNBN						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	93.0	0.9	0.10	0.33	14.2	41.8
1,000	94.7	0.9	0.07	0.26	10.7	32.8
100	91.8	0.9	0.11	0.31	16.1	39.7
C4.5						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	88.8	0.8	0.15	0.43	21.0	53.1
1,000	88.0	0.8	0.15	0.43	21.7	52.7
100	91.4	0.9	0.12	0.34	27.5	42.2

Table 8. Mean values of various performance measures for Competency 2 when using different sample sizes of non-normally distributed data ($\boldsymbol{p} \in [\mathbf{0.01}, \mathbf{0}]$).

GNBN						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	95.9	0.9	0.07	0.21	9.9	27.7
1,000	96.2	0.9	0.06	0.18	8.4	23.2
100	96.5	0.9	0.05	0.14	7.0	18.3
C4.5						
Sample size	*CA (%)*	*KS*	*MAE*	*RMSE*	*RAE (%)*	*RRSE (%)*
10,000	92.0	0.9	0.09	0.27	13.9	34.1
1,000	91.9	0.9	0.10	0.29	14.8	37.4
100	93.0	0.9	0.08	0.23	13.1	30.6

Table 9. Statistical stability results (including CA confidence interval bounds, sample average $\bar{\boldsymbol{X}}_{\boldsymbol{q}}$, and sample mean variance $\boldsymbol{S}_{\boldsymbol{q}}^{\mathbf{2}}$) regarding the performance of GNBN and C4.5 for Competency 1 for different sample sizes of normally distributed data ($\boldsymbol{p} \in [\mathbf{1.0}, \mathbf{0.8}]$).

GNBN				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	91.6	94.5	93.0	24.0
1,000	93.4	96.3	94.9	23.9
100	89.8	94.3	92.1	58.0
C4.5				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	87.5	90.2	88.8	21.3
1,000	86.6	90.4	88.5	42.5
100	86.9	91.3	89.1	54.5

Table 10. Statistical stability results (including CA confidence interval bounds, sample average $\bar{\boldsymbol{X}}_{\boldsymbol{q}}$, and sample mean variance $\boldsymbol{S}_{\boldsymbol{q}}^{\mathbf{2}}$) regarding the performance of GNBN and C4.5 for Competency 2 for different sample sizes of normally distributed data ($\boldsymbol{p} \in [\mathbf{1.0}, \mathbf{0.8}]$).

GNBN				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	94.3	97.4	95.9	27.6
1,000	94.9	98.1	96.5	30.6
100	93.4	97.5	95.4	46.6
C4.5				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	90.1	93.8	91.9	39.2
1,000	89.7	94.4	92.0	63.0
100	87.9	93.6	90.8	92.6

Table 11. Statistical stability results (including CA confidence interval bounds, sample average $\bar{\boldsymbol{X}}_{\boldsymbol{q}}$, and sample mean variance $\boldsymbol{S}_{\boldsymbol{q}}^{\mathbf{2}}$) regarding the performance of GNBN and C4.5 for Competency 1 for different sample sizes of non-normally distributed data ($\boldsymbol{p} \in [\mathbf{0.01}, \mathbf{0}]$).

GNBN				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	91.5	94.5	93.0	26.0
1,000	93.3	96.0	94.7	21.0
100	89.3	94.4	91.8	74.6
C4.5				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	87.2	90.4	88.8	28.1
1,000	86.3	89.7	88.0	32.0
100	89.1	93.7	91.4	58.3

Table 12. Statistical stability results (including CA confidence interval bounds, sample average $\bar{\boldsymbol{X}}_{\boldsymbol{q}}$, and sample mean variance $\boldsymbol{S}_{\boldsymbol{q}}^{\mathbf{2}}$) regarding the performance of GNBN and C4.5 for Competency 2 for different sample sizes of non-normally distributed data ($\boldsymbol{p} \in [\mathbf{0.01}, \mathbf{0}]$).

GNBN				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	94.4	97.4	95.9	27.1
1,000	94.5	97.9	96.2	33.1
100	94.5	98.5	96.5	44.6
C4.5				
Sample size	*Lower Bound*	*Upper Bound*	$\bar{X}_q$	S_q^2
10,000	89.9	94.1	92.0	49.9
1,000	90.0	93.9	91.9	44.5
100	90.4	95.6	93.0	76.6

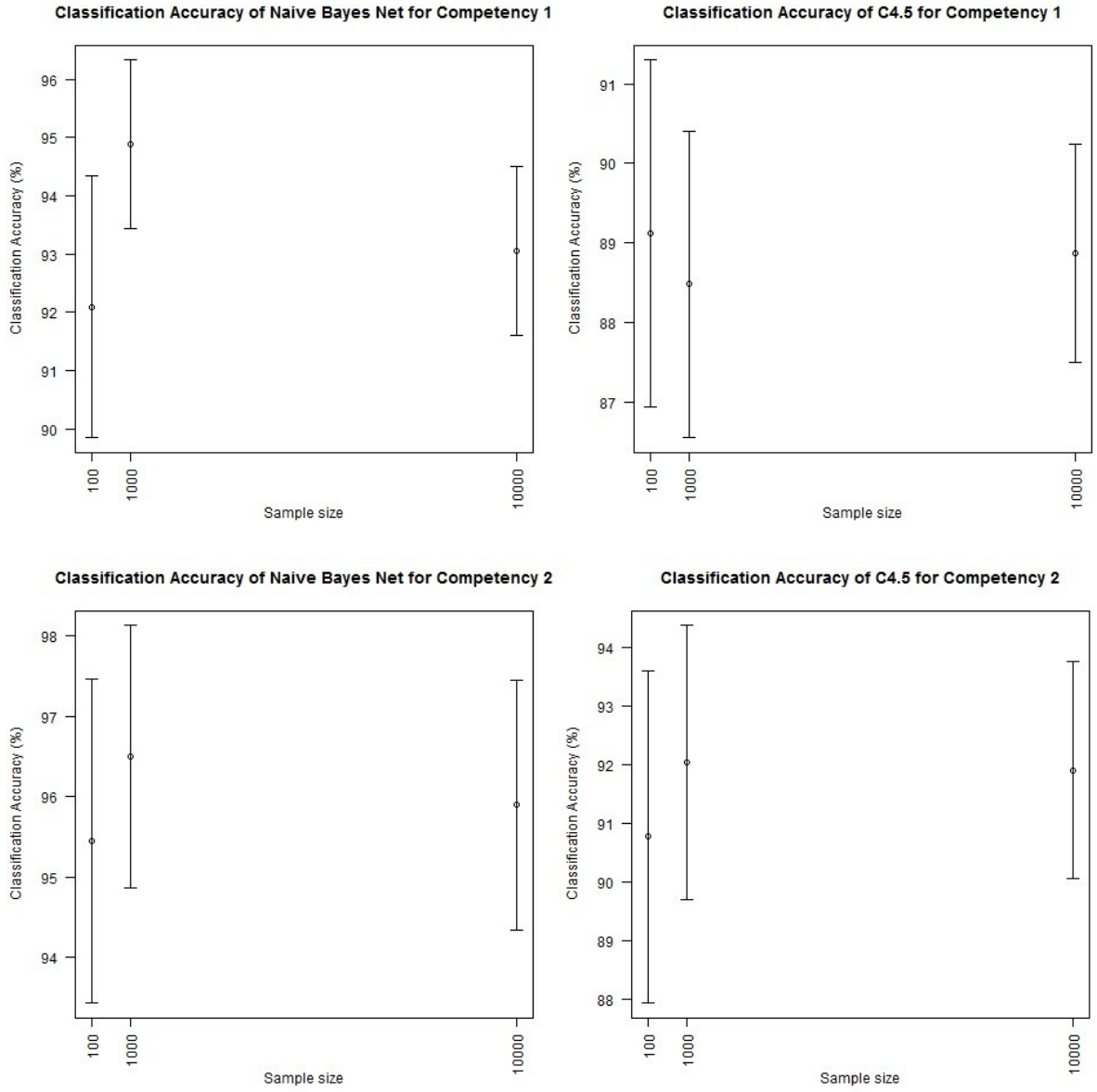


Fig. 6. Mean CA values and confidence intervals of the two ML algorithms (GNBN left, C4.5 right) for Competency 1 (top) and Competency 2 (bottom) respectively for different sample sizes of normally distributed simulation data $\boldsymbol{p} \in [\mathbf{1.0}, \mathbf{0.8})$.

VI. Discussion

Four research questions were posed in this study which required the testing of several conditions in order to provide an answer for them. Overall, the results show that the performance of both ML algorithms was high in all the conditions it was tested against, with a confidence of 99% on the results.

In specific, concerning the research question of how robust the SA methodology is when handling datasets of different normality significance levels; it was shown that the SA is highly robust even in conditions of extreme non-normality. In particular, the generic SA tool managed to reach high mean CAs (>88.3%), high mean KSs (>0.8), and low error rates (MAE, RMSE, RAE, and RRSE) when using both GNBN and C4.5 for the assigned ECD models in 8 different test conditions.

Regarding the research question of how robust the SA methodology is when handling datasets of different sample sizes; it was shown that SA is highly robust even when using small sample sizes such as 1,000 or 100 samples. That is, regardless of the normality significance level of these datasets. Notably, the generic SA tool managed to reach high mean CAs (>88.0%), high mean KSs (>0.8), and low error rates (MAE, RMSE, RAE, and RRSE) when using both GNBN and C4.5 for the assigned ECD models in 4 different test conditions.

The overall robustness of the SA methodology was examined when using different ML algorithms for continuous numerical datasets. For this reason, two ML algorithms were used in study; that is GNBN and C4.5. While both proved to be efficient, still it was shown that a parametric ML algorithm

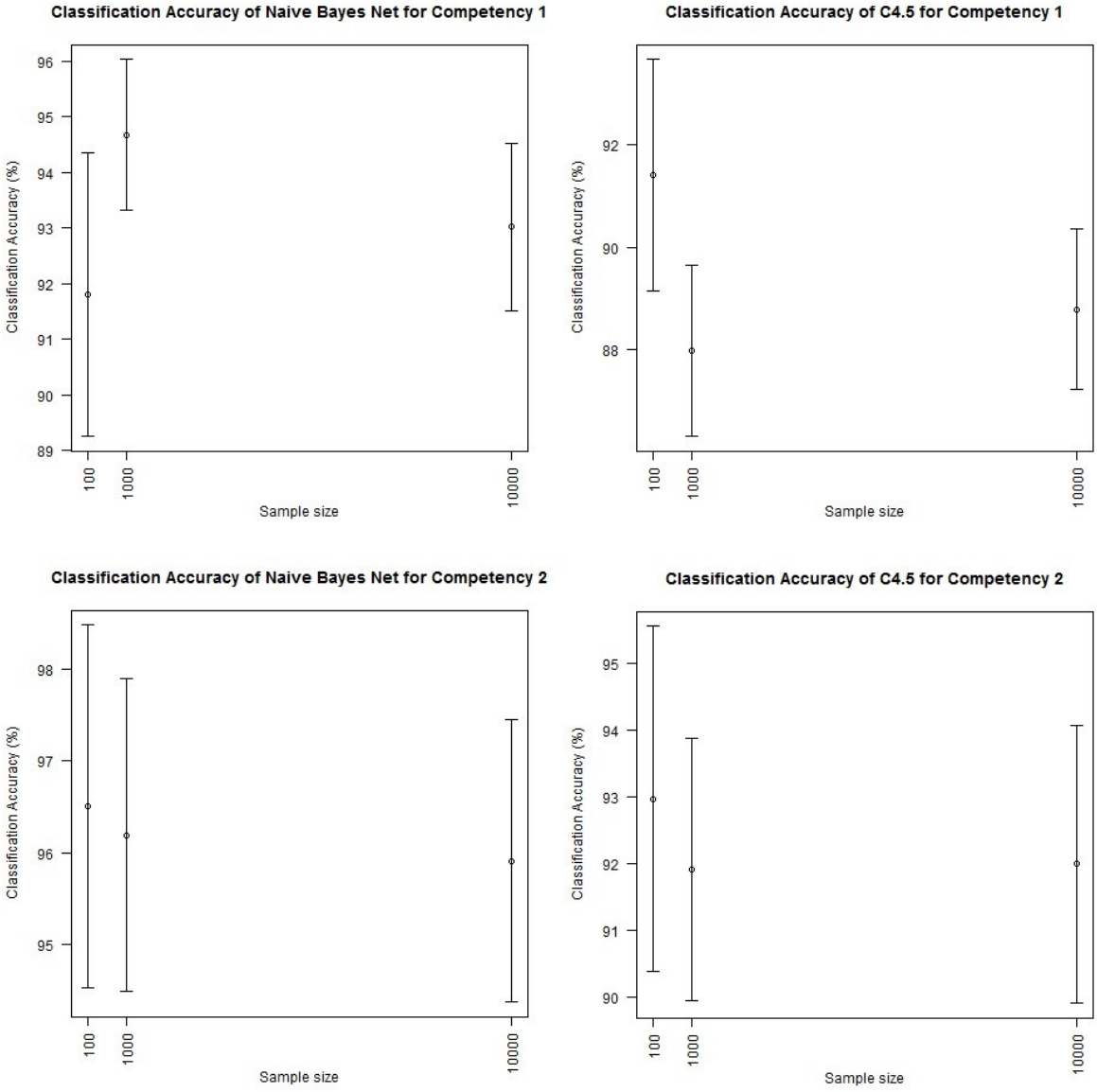


Fig. 7. Mean CA values and confidence intervals of the two ML algorithms (GNBN left, C4.5 right) for Competency 1 (top) and Competency 2 (bottom) respectively for different sample sizes of non-normally distributed simulation data ($p \in [0.01,0)$).

such as GNBN can perform just as well as or even better than a non-parametric ML algorithm such as C4.5, despite violating the normality assumption. However, C4.5 can be viewed as a better overall solution for SA as it can also handle discrete numerical data. Of course, other ML algorithms (such as Support Vector Machines and Neural Networks) could also be put to the test in the future.

Moreover, the SA methodology has proven to be robust when dealing with different ECD models. We observed that the more complex the constructs are, the less accurate the ML algorithms become. Obviously, more constructs of different shapes and sizes could be examined to get a better insight regarding the extent of SA's robustness in more complex cases. Nevertheless, we conclude that cases of elementary competency constructs, such as the ones that were used here, can be already handled efficiently by the generic SA tool, even simultaneously.

VII. Conclusion

The aim of this study has been to examine the robustness of SA against various conditions by using a generic SA software tool. The results have shown that SA is a robust methodology capable of handling all test conditions within the scope of this study with a high level of confidence. In specific, SA is capable of handling continuous numerical datasets of different distribution types and sample sizes, while utilizing different ML algorithms, disregard functioning under different competency construct configurations.

In addition, the generic SA software tool used in this study proved to be capable of executing all the core functionalities that were needed to examine the posed research questions despite being in an early version. It is important to note here that serious games can be applied to various

domains (e.g. education, marketing, etc.). Nonetheless, certain domains (e.g. health) are more critical than others and therefore software tools, such as the one presented in this study, should be thoroughly examined for their validity before being applied in practice. Therefore, the next steps for realizing a more complete and validated version of the SA tool is to conduct empirical studies with real-world data. To achieve this, the outcomes of validated (external or internal to the game) measurements, such questionnaires, can already be used with the generic SA software tool to validate its classification outputs. So far, the software tool has been validated in two separate empirical studies [29, 30]. Furthermore, to enhance the usability of the software tool we added a user interface layer, including a software wizard, help widgets, and other support functions/features to re-assure to provides a well-tailored experience to its user and allow the easier application of SA.To summarize, it is useful to discuss certain implications that rise from the use of the generic SA tool. As previously mentioned, the software tool allows the use of numerical data to perform assessments in serious games. However, later versions of the tool could include other types of data as well (e.g. nominal data) in order to enhance its usefulness. Nonetheless, this tool is a novel solution for assessing a broad range of competencies (e.g. 21st century skills, soft skills, digital skills, etc.) beyond the tight scope of the learning objectives of a traditional classroom. Furthermore, there are ethical/ privacy/data management implications. Using covert (stealth) assessment methodologies and tools does not mean that the subject is unaware of being monitored or assessed. Similar to any assessment, covert assessments can and should only be applied when it is fairly and fully communicated to the subject and the subject gives consent. Moreover, legal procedures must hold for securely storing the data and clarifying what it is exclusively used for and what will happen with the data and outcomes after its use (e.g. erasing the data).

ACKNOWLEDGMENT

This research was funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 644187, the RAGE project (www.rageproject.eu).

REFERENCES

[1] D. R. Michael and S. L. Chen. "Serious games: Games that educate, train, and inform." Muska & Lipman/Premier-Trade, 2005.

[2] E. A. Boyle, *et al.* "An update to the systematic literature review of empirical evidence of the impacts and outcomes of computer games and serious games." Computers & Education 94 (2016): 178-192.

[3] P. Wouters, *et al.* "A meta-analysis of the cognitive and motivational effects of serious games." Journal of educational psychology 105.2 (2013): 249.

[4] M. Qian and K. R. Clark. "Game-based Learning and 21st century skills: A review of recent research." Computers in Human Behavior 63 (2016): 50-58.

[5] F. Bellotti, *et al.* "Assessment in and of serious games: an overview." Advances in Human-Computer Interaction 2013 (2013): 1.

[6] V. J. Shute and G. R. Moore. "Consistency and validity in game-based stealth assessment". Information Age Publishing, Charlotte, NC, 2017.

[7] B. R. Belland. "The role of construct definition in the creation of formative assessments in game-based learning." Assessment in game-based learning. Springer, New York, NY, 2012. 29-42.

[8] V. J. Shute. "Stealth assessment in computer-based games to support learning." Computer games and instruction 55.2 (2011): 503-524.

[9] R. J. Mislevy, L. S. Steinberg, and R. G. Almond. "Focus article: On the structure of educational assessments." Measurement: Interdisciplinary research and perspectives 1.1 (2003): 3-62.

[10] R. J. Mislevy. "Evidence-Centered Design for Simulation-Based Assessment. CRESST Report 800." National Center for Research on Evaluation, Standards, and Student Testing (CRESST) (2011).

[11] V. J. Shute, M. Ventura, and Y. J. Kim. "Assessment and learning of qualitative physics in newton's playground." The Journal of Educational Research 106.6 (2013): 423-430.

[12] M. Ventura, V. Shute, and M. Small. "Assessing persistence in educational games." Design recommendations for adaptive intelligent tutoring systems: Learner modeling 2 (2014): 93-101.

[13] J. V. Shute, *et al.* "Measuring problem solving skills via stealth assessment in an engaging video game." Computers in Human Behavior 63 (2016): 106-117.

[14] G. R. Moore and V. J. Shute. "Improving learning through stealth assessment of conscientiousness." Handbook on Digital Learning for K-12 Schools. Springer, Cham, 2017. 355-368.

[15] K. Georgiadis, *et al.* "Accommodating Stealth Assessment in Serious Games: Towards Developing a Generic Tool." 2018 10th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games). IEEE, 2018.

[16] N. Falchikov. "Improving assessment through student involvement: Practical solutions for aiding learning in higher and further education." Routledge, 2013.

[17] J. L. Sabourin. "Stealth Assessment of Self-Regulated Learning in Game-Based Learning Environments." (2013).

[18] W. Min, *et al.* "DeepStealth: leveraging deep learning models for stealth assessment in game-based learning environments." International Conference on Artificial Intelligence in Education. Springer, Cham, 2015.

[19] R. J. Mislevy, R. G. Almond, and J. F. Lukas. "A brief introduction to evidence-centered design." ETS Research Report Series 2003.1 (2003): i-29.

[20] V. J. Shute, and M. Ventura, (2013). "Stealth assessment: Measuring and supporting learning in video games." MIT Press.

[21] P. Domingos. "A few useful things to know about machine learning." Communications of the ACM 55.10 (2012): 78-87.

[22] D. I. Singham. "Analysis of Sequential Stopping Rules for Simulation Experiments." Diss. UC Berkeley, 2010.

[23] F. Aucoin. "FAdist: Distributions that are sometimes used in Hydrology. R package version 2.2.". (2015). URL: https://CRAN.R-project.org/package=FAdist

[24] R. Core. Team. "R: A language and environment for statistical computing." R Foundation for Statistical Computing, Vienna, Austria. (2017). URL: https://www.R-project.org/

[25] J. Gross and U. Ligges. "nortest: Tests for Normality." R package version 1.4, (2015).

[26] A. Dragulescu. "xlsx: Read, write, format Excel 2007 and Excel 97/2000/XP/2003 files. R package Version 0.6.1." (2018). URL: https://CRAN.R-project.org/package=xlsx

[27] M. Dowle, *et al.* "data. table: Extension of data.frame. R package version 1.11.8." (2018). URL: https://CRAN.R-project.org/package=data.table

[28] H. Wickham. "ggplot2: elegant graphics for data analysis." Springer, 2016.

[29] K. Georgiadis, G. van Lankveld, K. Bahreini and W. Westera,. "Reinforcing Stealth Assessment in Serious

Games." In International Conference on Games and Learning Alliance. Springer, Cham., pp. 512-521, 2019,
[30] K. Georgiadis, T. Faber and W. Westera. "Bolstering Stealth Assessment in Serious Games." In International Conference on Games and Learning Alliance, Springer, Cham, pp. 211-220, 2019.

Konstantinos Georgiadis was born in Stockholm, Sweden in 1982. He received his Bachelor's degree in Industrial Informatics at the Technological Educational Institute of Kavala in Greece and his Master's degree in Game and Media Technology at the Utrecht University in the Netherlands. He is currently a PhD candidate at the Open University of the Netherlands. His major fields of study are serious games, simulations, assessment, machine learning, and data science.

He has completed his military service in Greece in 2008. In addition, he successfully completed a 10-month internship at the National Aerospace Laboratory of the Netherlands (NLR) where he conducted a multi-disciplinary research combining serious games and neuroscience. As a result, he published "EEG assessment of surprise effects in serious games" in the International Conference on Games and Learning Alliance, pp. 517-529, Springer, Cham, 2015. What is more, he published a study, namely "Accommodating Stealth Assessment in Serious Games: Towards Developing a Generic Tool" in 2018 10th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games), pp. 1-4, IEEE, 2018.

Giel van Lankveld was born in Gemert, The Netherlands in 1982. He received his Bachelor's degree in Biopsychology and his Master's degree in Cognitive Neuroscience at Maastricht University (The Netherlands). He completed his PhD at Tilburg University in the field of computer science. His major fields of study are serious games, simulations, and psychology, specifically in the context of the development of new research methodologies.

He has worked as a Postdoctoral research associate at Delft University of Technology in the field of gaming simulations for professional training and project management in cooperation with ProRail, in the context of training for Dutch traintraffic controllers. Furthermore he has worked as a Postroctoral researcher in the RAGE project at The Open University in Heerlen. In this project his research topics were developing plug and play game components suitable for the RAGE ecosystem and serious games for teaching. Currently, he is working as lecturer in computer science and data science at Fontys Applied University in Eindhoven.

Kiavash Bahreini was born in Ahvaz, Iran in 1976 and grew up in Tehran. He received the BSc. diploma in software engineering and databases from the Azad University of Tehran, Tehran, in 2001.He received the MSc. diploma in computer engineering and knowledge management from the Eastern Mediterranean University, Turkey (T.R.N.C.), in 2008.He received the Ph.D. diploma in computer science and machine learning from the Open University of the Netherlands, Heerlen, in 2015.

From 2006 to 2008, he was a research and teaching assistant with the Eastern Mediterranean University. Since 2009, he has been a researcher and a developer with the University of Twente, the University of Amsterdam, and the Open University of the Netherlands. He is the author and the translator of 10 books, more than 50 articles, and more than five innovative ideas. His research interests include computer science, data science, big data, affective computing, machine learning, and real-time applications development.
Dr. Kiavash Bahreini is a researcher at the Welten Institute, Research Centre for Learning, Teaching and Technology at the Open University of the Netherlands.

Wim Westera was trained as a physicist and received his PhD at Utrecht University, the Netherlands. He was trained by the BBC as an academic producer and director and has made more than hundred educational documentaries and TV programs.

Since the early nineties Wim has been with the Open University of the Netherlands as an educational technologist and head of educational innovation. Since 2008, he has been a full professor at the same institute, specialized in learning media, in particular serious games, simulations and computational modelling.

Prof. dr. Wim Westera's book "The Digital Turn" about the influence of digital media on human existence was awarded the US National Indie Excellence Award (www.thedigitalturn.co.uk). He is the initiator and coordinator of the RAGE flagship project on serious games in Horizon2020, board member of the Dutch Games Association and chairperson of the RAGE Foundation. More info at www.wwestera.nl.