Understanding media-supported childhood education

Mikołaj Małkiński[1,2]*

Abstract
Education during childhood can be highly beneficial for persons development. Thanks to current state of technology it is possible to boost this process by supporting it with media devices. This work tackles a challenge proposed in the *2019 Data Science Bowl* competition which aims to understand key factors standing behind the learning process of young children. Firstly, it performs an analysis of gameplay data gathered by a game-based learning application for children. Then, it formulates a baseline model, constructs hand-crafted features based on domain knowledge and compares the results of different state-of-the-art models such as Support Vector Machines and XGBoost on the extended dataset. Finally it discusses obtained results and presents further research directions which could improve the process of media-supported education for children.

Keywords
Childhood education — Data analysis — Classification — SVM — XGBoost

[1] *Faculty of Mathematics and Information Science, Warsaw University of Technology, Warsaw, Poland*
[2] *Faculty of Computer Science of Barcelona, Universitat Politècnica de Catalunya, Barcelona, Spain*
*Email: malkinskim@student.mini.pw.edu.pl / mikolaj.malkinski@est.fib.upc.edu

Contents

1. Introduction

With recent advances in technology humans are presented with more and more opportunities to gain insights into the surrounding world. Ability to gather tremendous amounts of data, high processing power of modern machines and possibilities for collaboration between researchers from different parts of the world make it possible to tackle challenging problems on a global scale. Data Science Bowl is a platform created exactly for this purpose. It organises competitions which focus on social good by gathering rich datasets and encourages researchers to collaborate on the solution. Previous events were hosted on the Kaggle platform and tried to tackle problems such as heart disease detection [1], lung cancer detection [2] or nuclei detection [3], [4]. This year, Data Science Bowl introduced a competition which main goal is to analyse how media can support learning outcomes in early childhood education [5].

The data required for this challenge was collected with *PBS KIDS Measure Up!* application, which is a game-based learning tool developed as a part of the *CPB-PBS Ready to Learn* initiative, supported by the U.S. Department of Education. It consists of anonymous gameplay data about played games and videos watched by children. Based on this data, the goal is to predict how children will perform in their assessments.

This work will perform analysis on available dataset which will outline key characteristics of used education methods and propose ways for improvement. Additionally, it will tackle the main challenge of predicting assessment scores by evaluating the performance of Support Vector Machines [6] and comparing the results with other state-of-the-art methods.

2. Dataset

The dataset contains information about game analytics gathered by the *PBS KIDS Measure Up!* application. It places players into a fictional world where they can participate in different activities, games, video clips or assessments. The Figure 1 presents the popularity of each part of the gameplay. Each assessment is created with the goal of testing player's comprehension of a certain group of measurement-related skills. They are divided into following categories: *Bird*

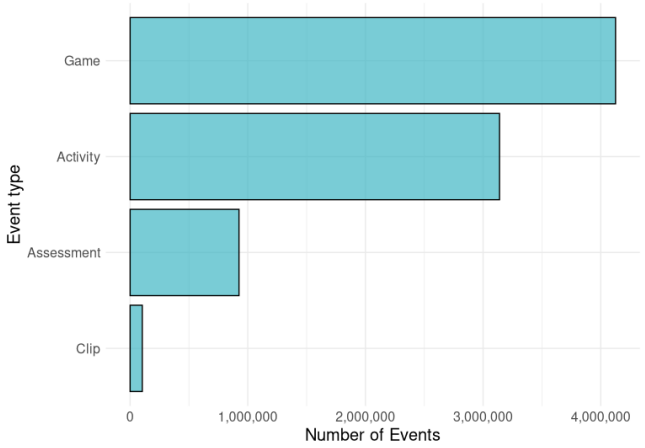

Figure 1. Count of events with given type. Most of the gameplay consists of games and activities, around 12.5% is devoted to assessments whereas video clips are the rarest category.

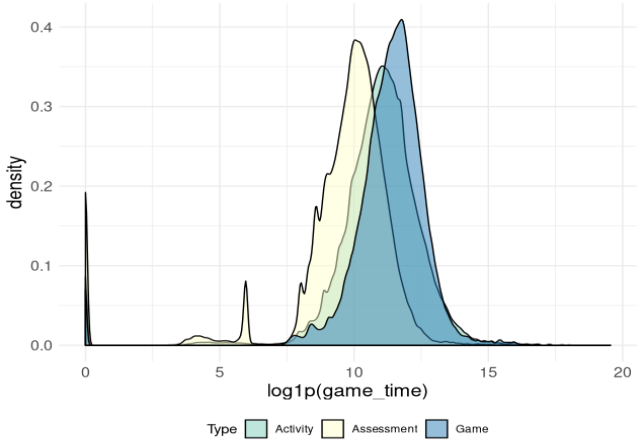

Figure 2. Analysing the time spent on each gameplay category it's visible that most of the time is spent on games which are the most entertaining content in the application while least on assessments. Additionally there is a peak at the beginning which shows that some events are simply skipped by the user.

Measurer, *Cart Balancer*, *Cauldron Filler*, *Chest Sorter*, and *Mushroom Sorter*.

The goal of the competition is to predict the number of attempts a child will need to pass given assessment, based on the already gathered gameplay data. Each incorrect answer is counted as an attempt. The dataset is divided into two parts: training and testing. The former one contains full history of gameplay data. On the other hand, the latter one is missing history after starts of randomly chosen assessments, for which the number of attempts has to be predicted. Furthermore, the outcomes of assessments are grouped into 4 categories: solved on the first attempt; solved on the second attempt; solved after 3 or more attempts; never solved. Figures 1 and 2 give further insights into the dataset.

3. Methods

This section will describe metrics, models and techniques such as data transformations used to improve the results. Final reported scores are computed on the testing set - the predictions of accuracy groups are submitted to the Kaggle platform which evaluates their correctness and returns obtained score.

3.1 Evaluation metric

To evaluate the submission, *Quadratic Weighted Cohen's kappa coefficient* is used. The general Cohen's kappa coefficient is defined as:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{1}$$

, where p_o is the relative observed agreement among raters (accuracy) and p_e is the hypothetical probability of chance agreement, using the observed data to calculate the probabilities of each observer randomly seeing each category. In other words, κ is a more robust metric than accuracy which takes into account the possibility of correct solutions occurring by chance.

Table 1. Coefficients of a linear weighted κ for a 4-class classification problem.

Class	0	1	2	3
0	0.00	0.33	0.67	1.00
1	0.33	0.00	0.33	0.67
2	0.67	0.33	0.00	0.33
3	1.00	0.67	0.33	0.00

Table 2. Coefficients of a quadratic weighted κ for a 4-class classification problem.

Class	0	1	2	3
0	0.00	0.56	0.89	1.00
1	0.56	0.00	0.56	0.89
2	0.89	0.56	0.00	0.56
3	1.00	0.89	0.56	0.00

The weighted κ allows to weight disagreements differently - the score depends on the distance between predicted and actual class. Recall that the considered challenge is a 4-class classification problem. Table 1 presents weights used in a linear weighted κ. They can also be thought as *disagreement scores* - the larger the weight the higher disagreement between predicted and actual class. To understand, consider the case when model predicts accuracy group 2 - the assessment was solved on the second attempt. Then, if the prediction is correct the disagreement score is equal to 0, if assessment was solved on the first attempt (class 3) or after 3 or more attempts (class 1) the prediction is off only by 1 class and the disagreement score is 0.33 and finally if it was never solved it equals to 0.67.

The quadratic version allows to specify that the disagree-

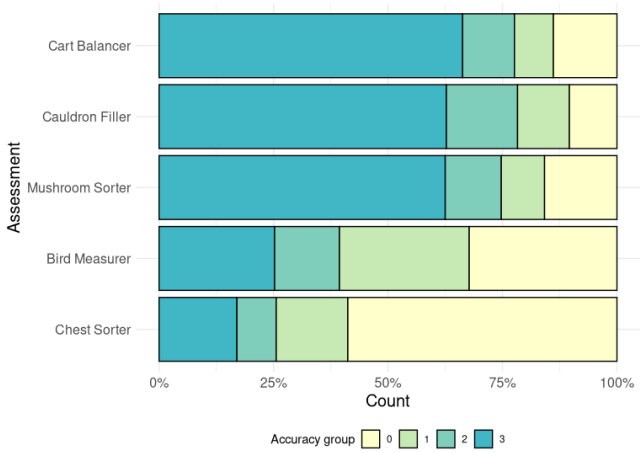

Figure 3. Accuracy groups presented for each assessment type - the main idea behind the baseline model.

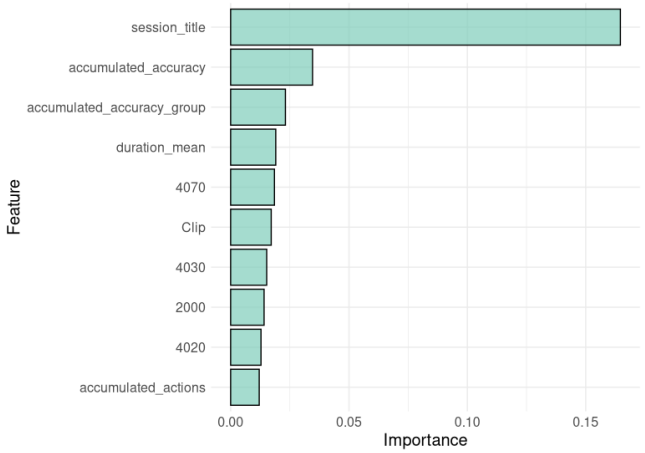

Figure 4. Feature ranking of XGBoost model with default parameters trained on the whole dataset for 10 most important variables.

ment score depends not only on the difference between predicted classes but also on the order of the actual class. The details of weights used in this case are shown in the Table 2.

3.2 Baseline model

Firstly a naive model is created based on raw data to give a point of comparison for more advanced methods. As the Figure 3 shows, there is a clear difference between assessment completion rates depending on the problem type. The *Chest Sorter* appears to be the hardest one having around 60% of the attempts never solved, whereas *Mushroom Sorter*, *Cauldren Filler* and *Cart Balancer* are usually solved after the first attempt. Based on this observation a baseline model is built which is based on the median of accuracy groups - for a given child and type of assessment to predict, a median of all previous assessments of this child with given type is taken and this gives the predicted accuracy group. This model achieved value of quadratic weighted κ equal to 0.396.

3.3 Feature engineering

To potentially improve the performance of proposed models it may be beneficial to introduce hand-crafted features based on domain knowledge. For this purpose the dataset was extended with fundamental summary statistics computed for each user. The set of generated features contains the count of each activity in which user engaged, their accumulated and mean scores in each assessment, the number of successful and unsuccessful attempts, total counts of each event which occured during users gameplay history and some additional ones like the total or average time spent on each activity. In total, the dataset was extended to more than 800 columns. Furthermore, the original data contained more than 1 million records. That is because each event which occurred during the gameplay was saved separately. After described summarisation the new training set contains less than 18 thousand rows. The same transformations were applied to the testing set.

With such large number of columns it is expected that some, if not most, of them will turn out to be useless. In this case 2 approaches can be used: either one may try to analyse each feature separately by for example calculating its correlation with the response variable; or a feature selection method can be applied. Without this procedure some classification models may have troubles in distinguishing which features are important and include them in the final prediction even though they are just random noise not correlated with the predicted feature. What is more, the computation with such big dataset will take considerably more time than if the number of variables would be reduced.

3.4 Feature selection

To combat the *curse of dimensionality* and reduce the dataset size a feature selection method has to be applied.

Iterative methods The first possibility is to use simple iterative methods like *forward selection* which starts with a model without any features and then in each iteration adds a new variable; *backward elimination* which starts with all features and then removes the least significant variable in each iteration; *recursive feature elimination* which constantly creates new models while remembering the best and worst performing features at each iteration. However, all described methods require a tremendous amount of iterations to find the best subset of predictors.

Ranking models Some of classification models are able to store additional information during training which tells about which feature was the most useful. In this way it's possible to train a single model, such as XGBoost [7], and rank the variables by their importance. Figure 4 shows the 10 most important features selected by the XGBoost model from the whole set of more than 800 columns. It's visible that a clear winner is the title of the assessment for which the accuracy group is predicted. Other good variables are the accumulated accuracy and accuracy group for given player, as well as the average duration of the assessment - we can expect that

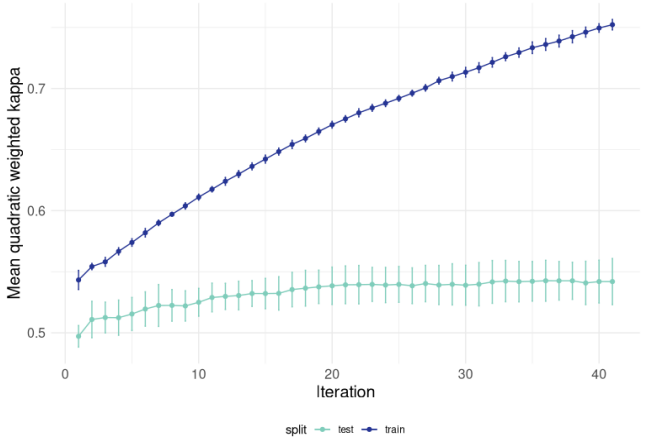

Figure 5. Value of mean quadratic weighted κ with standard deviation averaged across 5 cross-validation folds for XGBoost.

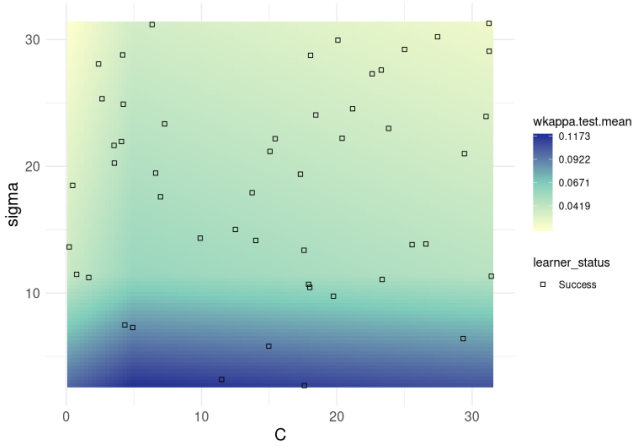

Figure 6. Heatmap showing 50 values of quadratic weighted κ depending on randomly sampled parameters $C \in (2^{-5}, 2^5)$ and $\sigma \in (2^{-5}, 2^5)$ for SVM with rbfdot kernel, calculated on a heldout validation set.

users who solve a task in shorter time know the underlying reason standing behind the correct answer and don't require additional time for thinking.

3.5 XGBoost

Firstly the performance of state-of-the-art XGBoost model is analysed. Figure 5 shows the value of quadratic weighted κ for the training and testing sets. The results are averaged across 5 folds of cross-validation and *early stopping* is used to stop the training if the metric doesn't improve on the testing set for more than 5 iterations. It's clearly visible that the model overfits to the training data. In order to improve the model performance and reduce overfitting, hyperparameter tuning is performed. The complexity of an XGBoost model can be controlled by manipulating following parameters: *max_depth* controls the depth of generated trees - smaller trees have smaller predictive power but are less complex and have higher chance of generalising to the test set, bigger trees will more likely overfit to the train data; *min_child_weight* which determines when to partition a leaf in the tree; *subsample* which is the factor of training set to subsample for growing trees in each iteration; *colsample_bytree* determines the subsample ratio of features taken into consideration when building a new tree.

For each presented parameter a list of possible values was defined. Then a method called *random search* was performed which randomly sampled a value for each parameter and evaluated the model using 5-fold cross validation. The best performing model achieved weighted quadratic κ of 0.5411 (averaged across 5 cross validation folds) and *TODO* on the testing set evaluated by the Kaggle platform.

3.6 Support Vector Machines

Support Vector Machines (SVM) is another well performing model for classification tasks. In presented experiments, the *Radial Basis Function (RBF)* will be used as the kernel. To achieve reasonable performance from an SVM it's necessary

Table 3. Average quadratic weighted κ computed by 5-fold cross validation on the training set.

Model	κ
SVM	0.396
XGBoost	0.396
Baseline	0.396

to carefully choose its parameters. This work will analyse the influence of 2 parameters: C - a regularisation parameter of the SVM, which is able to balance the trade-off between the amount of correctly classified observations from the training data and the margin of decision function; σ TODO.

For this purpose again the random search is applied. Firstly, for a fixed kernel, different randomly sampled configurations of SVM's parameters $C \in (2^{-5}, 2^5)$ and $\sigma \in (2^{-5}, 2^5)$ are evaluated. Figure 6 presents a heatmap with values of quadratic weighted κ depending on 50 randomly selected values of hyperparemeters. The training time of an SVM, even with reduced number of features, is significantly larger compared to XGBoost. Therefore for these experiments, cross validation was replaced with a heldout validation set, obtained by removing part of the training set. Unfortunately, among all these configurations the highest performing one was able to achieve $\kappa = 0.1173$. It's visible that higher values of κ were achieved for lower values of σ, where it seems that C doesn't make a big difference. The random search procedure was repeated 2 more times with the same bounds for C, decreased bounds for σ and 25 sampled configuraions in both cases. Restricting $\sigma \in (2^{-8}, 2^3)$ allowed to find a solution with $\kappa = 0.2975$, whereas $\sigma \in (0, 1)$ allowed to find a solution with $\kappa = TODO$.

4. Discussion

This work focused on analysing the possibilities of supporting childhood education with technology, by predicting childs performance in measurement based assessments. Analysis

and visualisation of the Data Science Bowl 2019 dataset give interesting insights into the way children interact with modern technological devices. However, such a dataset requires in-depth knowledge about the considered domain in order to prepare hand-crafted features which can then be used by a learning model. Most likely the features created in this work can still be improved by experts which could further boost the performance of evaluated models. Moreover, to gain more insights into the problem, a wider selection of models should be evaluated on this dataset, starting from simple ones like logistic regression. Further research should be conducted which could compare their performance to the heavy artillery used in this work - XGBoost and SVM.

Then, the performance of 2 state-of-the-art models was compared with a simple baseline. XGBoost even with default parameter setting is able to achieve a significantly higher performance than constructed baseline. On the other hand, SVMs achieved worse performance while requiring careful parameter tuning, processing time and computing resources. The final achieved results are summarised in the Table 3. However, it's important to remember that the SVM was trained only on 50 selected features from the dataset, which were chosen as the most crucial by XGBoost. It might turn out that other variables are more important for good performance of SVM, which were not included in this set. Nonetheless, exploring this possibility would require a significantly larger computing possibilities even to train the model with default parameters, but then what about cross-validation and hyperparameter tuning?

Acknowledgments

So long and thanks for all the fish [6].

References

[1] Data Science Bowl. Transforming how we diagnose heart disease. https://datasciencebowl.com/transforming-how-we-diagnose-heart-disease/, 2015.

[2] Data Science Bowl. Turning machine intelligence against lung cancer. https://datasciencebowl.com/turning-machine-intelligence-against-lung-cancer/, 2016.

[3] Data Science Bowl. Spot nuclei, speed cures. https://datasciencebowl.com/spot-nuclei-speed-cures/, 2018.

[4] Juan C. Caicedo, Allen Goodman, Kyle W. Karhohs, Beth A. Cimini, Jeanelle Ackerman, Marzieh Haghighi, CherKeng Heng, Tim Becker, Minh Doan, Claire McQuin, Mohammad Rohban, Shantanu Singh, and Anne E. Carpenter. Nucleus segmentation across imaging experiments: the 2018 data science bowl. *Nature Methods*, 2019.

[5] Data Science Bowl. Uncover the factors to help measure how young children learn. https://www.kaggle.com/c/data-science-bowl-2019/overview, 2019.

[6] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.

[7] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016.