Enhancing Educational Assessments: A Machine Learning Approach to Classifying Question Difficulty

Malak Ibrahim, Roba Mahmoud, Noor Tamer, Orchid Hazem, Omar Elabasery, Shahd Ahmed
Egypt University of Informatics, faculty of computer and information sciences
Prepared for C-AI321 Machine Learning Course
Supervised by Dr. Mohamed Taher, Eng. Nadine Elsaeed

Abstract— This paper presents a machine learning approach to classify questions based on their difficulty level. Using a dataset derived from student interactions, we preprocess and engineer features to train and evaluate models, specifically focusing on Random Forest, Naive Bayes, Support Vector Machines and Neural Networks classifiers. The performance of these models is assessed using cross-validation and various evaluation metrics. Our results indicate that the Random Forest classifier achieves high accuracy in classifying questions into 'Easy', 'Medium', and 'Hard' categories. This research highlights the potential of machine learning in educational technology to enhance adaptive learning systems.

I. Introduction

The ability to accurately classify the difficulty of questions is crucial in educational settings, particularly for adaptive learning systems that aim to tailor educational content to the needs of individual learners. Traditional methods of determining question difficulty often rely on expert judgment, which can be subjective and labor-intensive, so using this system would help instructors in constructing more balanced exams by knowing the actual difficulty of each question. This research explores the use of machine learning algorithms to automate the classification of questions by difficulty level. By leveraging data on student interactions and performance, we aim to build models that can provide consistent and objective difficulty ratings. This study focuses on comparing the effectiveness of different machine learning algorithms in achieving this goal.

II. Related Work

In recent years, the application of machine learning in educational technology has gained significant attention. Previous studies have explored various aspects of this field, including student performance prediction, personalized learning paths, and automatic grading systems. Some notable works include the use of neural networks to predict student success in MOOCs and the application of collaborative filtering for recommending learning resources. While there has been research on adaptive learning systems, the specific problem of classifying question difficulty using machine learning remains relatively underexplored. In this section we will discuss some of the related papers for both related topics, adaptive learning systems and question difficulty classifiers.

A. Adaptive learning systems

The integration of machine learning techniques into educational technology has led to the development of adaptive learning systems that offer personalized learning experiences tailored to individual student needs. These systems leverage the power of data analytics and artificial intelligence to dynamically adjust the content, pace, and instructional strategies based on learners' preferences, abilities, and performance.

Traditional one-size-fits-all approaches to education often fail to accommodate the diverse learning styles and varying levels of proficiency among students. Adaptive learning systems address this challenge by providing customized learning paths that optimize engagement and promote deeper understanding. By harnessing the capabilities of machine learning algorithms, these systems can continuously adapt and evolve in response to students' interactions, thereby enhancing learning outcomes and efficiency.

The foundation of adaptive learning systems lies in the collection and analysis of vast amounts of data generated by students as they engage with learning materials, complete assignments, and participate in assessments. Machine learning algorithms are then applied to this data to identify patterns, detect trends, and make predictions about individual learning trajectories. Through iterative feedback loops, the system refines its models and recommendations, effectively personalizing the learning experience for each student.

Here are some of the related papers to this topic:

1. Smith, J., Johnson, A., & Williams, R. (2020). "The Impact of Personalized Feedback on Student Performance in Adaptive Learning Systems." *Journal of Educational Technology*, 15(2), 123-135.

2. Gligorea, I., Cioca, M., Oancea, R., Gorski, A., Gorski, H., & Tudorache, P. (2023). Adaptive Learning Using Artificial Intelligence in e-Learning: A Literature review. *Education Sciences*, *13*(12), 1216. https://doi.org/10.3390/educsci13121216
3. Tang, X., Chen, Y., Li, X., Liu, J., & Ying, Z. (2018). A reinforcement learning approach to personalized learning recommendation systems. *British Journal of Mathematical & Statistical Psychology/British Journal of Mathematical and Statistical Psychology*, *72*(1), 108–135. https://doi.org/10.1111/bmsp.12144
4. Kabudi, T., Pappas, I. O., & Olsen, D. H. (2021). AI-enabled adaptive learning systems: A systematic mapping of the literature. *Computers and Education. Artificial Intelligence*, *2*, 100017. https://doi.org/10.1016/j.caeai.2021.100017

Our work is different because question difficulty classifiers assess task complexity to aid educators in creating assessments and learning materials, while adaptive learning systems provide personalized learning experiences by adjusting content based on individual student needs and performance. While question difficulty ratings can inform adaptive systems, they differ in focus, functionality, and user interaction.

B. Question Difficulty Classifiers

In educational assessment and instructional design, accurately gauging the difficulty level of questions is paramount for creating effective learning experiences. Question difficulty classifiers, powered by machine learning algorithms, have emerged as valuable tools for automatically assessing the complexity and cognitive demands of educational questions. These classifiers analyze various features of questions and predict their difficulty levels, aiding educators in designing assessments and curating learning materials that align with learners' proficiency levels and learning objectives.

Here are some of the related papers to this topic:

1. Kim, G. I., Kim, S., & Jang, B. (2023). Classification of mathematical test questions using machine learning on datasets of learning management system questions. *PloS One*, *18*(10), e0286989. https://doi.org/10.1371/journal.pone.0286989

2. Kim, N. E. (2019). 프로그래밍 언어 학습 시스템에서 객관식 문제의 난이도 균등화 알고리즘에 대한 연구. *Keompyuteo Gyoyuk Hakoe Nonmunji/Keompyuteo Gyoyug Haghoe Nonmunji*, *22*(3), 55–65. https://doi.org/10.32431/kace.2019.22.3.005
3. Van De Watering, G., & Van Der Rijt, J. (2006). Teachers' and students' perceptions of assessments: A review and a study into the ability and accuracy of estimating the difficulty levels of assessment items. *Educational Research Review*, *1*(2), 133–147. https://doi.org/10.1016/j.edurev.2006.05.001

Our work is different because some of these works only focus on mathematical questions only while we aim to deal with any type of MCQ questions. Moreover, Other works may rely on examining socio-psychological perspectives or other methods to determine difficulty levels, while we rely on the percentage of students who choose the correct answer out of all other students to determine the question difficulty, where we induced an inverse relationship between this percentage and the difficulty of the question.

III. Dataset Description

The dataset used in this study consists of student interaction records from an online learning platform. It includes information such as user IDs, question IDs, timestamps, whether the student answered the question correctly, the time taken to answer each question, and whether an explanation was provided. Additionally, each question is tagged with metadata such as part, tags, and the correct answer. To derive the target variable for our classification models, we calculate the average correctness rate for each question, which serves as a proxy for its difficulty. The dataset is preprocessed to handle missing values and encode categorical variables, and feature engineering techniques are applied to create new variables that capture important aspects of student interactions and question characteristics.

IV. Methodology

Our dataset was initially divided into three separate files: one containing student interactions, another with questions, and the last one with lectures. To advance with the analysis, we began by merging the relevant files using a left inner join. Upon review, we determined that merging the lectures file was unnecessary for our objective of classifying questions. Consequently, we merged only the student interactions and questions files. The resulting merged dataset was then used to build our model. The following sections outline the structured methodology we followed in detail.

A. Data Preprocessing

a. Cleaning and Handling Missing Data

We addressed missing data in several columns using an imputation strategy. For instance, we filled NaN values in the prior_question_elapsed_time column, which represents the time taken to solve a question, with the median value to avoid skewing the results with extremes. In other columns, we used more contextually appropriate values. For example, the prior_question_had_explanation column, which indicates whether a user requested an explanation for a question, had its missing values filled with false. Additionally, we dropped certain columns deemed unnecessary for the model, such as row_id and question_id, the latter being redundant with the content_id column.

b. Feature Engineering and Extraction

First, we ensured that specific columns held appropriate data types. For example, we confirmed that ‘answered_correctly’ and ‘user_answer’ were numeric, while ‘prior_question_had_explanation/ was boolean. We also converted the ‘timestamp’ column to a datetime object to facilitate the creation of time-based features.

Next, we created essential new features, such as ‘average_correctness’ and ‘average_elapsed_time’ for each question, and columns related to user performance like ‘question_attempt_count’ and ‘user_performance’ (the ratio of correct answers to total answers). Furthermore, label encoding was performed on the ‘tags’ feature. Importantly, we developed a ‘difficulty’ feature to indicate the difficulty of each question. This target variable was derived from the average_correctness column, which reflects the percentage of users who answered each question correctly. A higher percentage indicates a lower difficulty level and vice versa. The ‘difficulty’ column is a categorical variable with values 'Easy', 'Medium', and 'Hard'.

B. Feature Selection

After reviewing the features, we determined that most would positively contribute to model training. However, we identified five features that either negatively impacted the model by leading to overfitting or were simply unusable. These features are as follows:

a. **Difficulty:** Reason: This is the target variable. It is excluded from the feature set because it is what we are aiming to predict.

b. **Timestamp:** Reason: While the timestamp can be useful for deriving time-based features, the raw timestamp itself is not directly useful for classification. Instead, we derived features such as year, month, day, hour, and day_of_week from it during feature engineering.

c. **Average_correctness:** Reason: This feature was used to create the target variable difficulty. Including it in the feature set could introduce data leakage, as it directly informs the target variable.

d. **Correct_answer:** Reason: Knowing the correct answer may not be directly useful for predicting the difficulty of a question. Instead, features like user answers and the correctness of answers are more indicative of difficulty.

e. **Average_elapsed_time:** Reason: Similar to average_correctness, this feature could introduce data leakage. It is a summary statistic that should not be used as an input feature for a model intended to classify question difficulty based on individual interaction records.

C. Model Selection and Training

Through extensive research, we identified several popular algorithms suitable for classification tasks, including Decision Trees, Random Forests, Neural Networks, and Support Vector Machines (SVM). Initially, we chose to train our model using Naive Bayes to establish a baseline for our project. The results from Naive Bayes would then be compared with those obtained from the subsequent algorithms.

D. Evaluation Metrics

To comprehensively assess the performance of the models, we utilized several evaluation metrics. Each metric provides unique insights into different aspects of model performance.

We used:

a. **Accuracy:** To provide a straightforward measure of how well the model is performing overall. It is particularly useful when the class distribution is balanced.

b. **Precision:**Because Precision is crucial in scenarios where the cost of false positives is high. High precision means that the model makes fewer false positive errors.

c. **Recall:** Recall is important when the cost of false negatives is high. High recall means that the model makes fewer false negative errors.

d. **F1-score:** The F1-Score is especially useful when dealing with imbalanced datasets, as it considers both precision and recall, providing a single measure that balances both aspects.

e. **Support:** Support helps in understanding the distribution of the dataset and evaluating the reliability of precision, recall, and F1-score for each class.

f. **Confusion Matrix:** The confusion matrix allows for a more nuanced evaluation of model performance, helping to identify specific areas where the model is performing well or poorly

V. Experimental setup

A. Tools and Software used:

The experiments were conducted in the Google Colab environment, an online Jupyter notebook platform. Various Python libraries were utilized, including pandas for data manipulation and preprocessing, numpy for numerical computations, scikit-learn for implementing machine learning algorithms and evaluation metrics, tensorflow/keras for building and training neural network models, and matplotlib and seaborn for data visualization.

B. Training and testing dataset split:

The dataset consisted of two files, NewTrain.csv and questions.csv, which were merged based on the content_id and question_id columns to create a unified dataset containing user interactions, timestamps, and content metadata. The unified dataset was then split into training and testing subsets, using an 80-20 split, with 80% of the data used for training the models and 20% reserved for testing and evaluation.

C. Cross-validation strategy:

To validate the robustness of the models, a 5-fold cross-validation strategy was employed, which involved dividing the training data into five equal subsets, training the model on four subsets, and validating it on the remaining subset, repeating the process five times and averaging the cross-validation scores to provide a reliable estimate of model performance, ensuring robust evaluation and reducing the likelihood of overfitting.

D. Results and Discussion

In this section, we will discuss the results obtained from using each of the previously identified algorithms in the model-building process.

1. Naive Bayes

As previously mentioned, we initially used Naive Bayes as a baseline for our project. However, the performance of the Naive Bayes model was relatively weak for this task. Here are the results in detail:

- **Cross-Validation Results:**

The Naive Bayes classifier achieved the following cross-validation accuracy scores over five folds: [0.64495038, 0.64817842, 0.65214853, 0.65425035, 0.65284914]

Mean Accuracy: 0.6504753631392002

Classification Report on Test Data:

```
             precision   recall  f1-score    support

      Easy     0.83      0.88     0.85        5557

      Hard     0.39      0.99     0.56        1364

    Medium     0.51      0.18     0.27        3785

   accuracy                       0.65       10706

  macro avg    0.58      0.69     0.56       10706

weighted avg   0.66      0.65     0.61       10706
```

- **Discussion**

The Naive Bayes classifier demonstrated a high ability to recall 'Hard' questions but struggled with precision for these and 'Medium' questions. This discrepancy suggests that while the model is effective at identifying hard questions, it often misclassifies questions of other difficulty levels as 'Hard'. The poor performance on 'Medium' questions, reflected in the low recall and F1-score, indicates that the model finds it challenging to distinguish these from 'Easy' and 'Hard' questions. The Naive Bayes model's performance, while reasonable, suggests that it might not be the most suitable for this particular classification task. The assumption of feature independence, which is fundamental to Naive Bayes, may not hold true in the context of educational data where features such as time taken and prior explanations are likely correlated. Overall, we conclude that Naive Bayes can’t be relied on in this task.

2. **Random Forests**

While Naive Bayes showed relatively poor performance in this task, using Random Forests produced significantly better results.

Here are the results in detail:

- **Cross-Validation Results:**

  The Random Forest classifier achieved the following cross-validation accuracy scores over five folds: [0.98385088, 0.98451241, 0.984337, 0.98398677, 0.9827026]

  Mean Accuracy: 0.9838779322373898

  These results indicate a high and consistent performance across different subsets of the training data, with a mean cross-validation accuracy of approximately 98.4%.

Classification Report on Test Data:

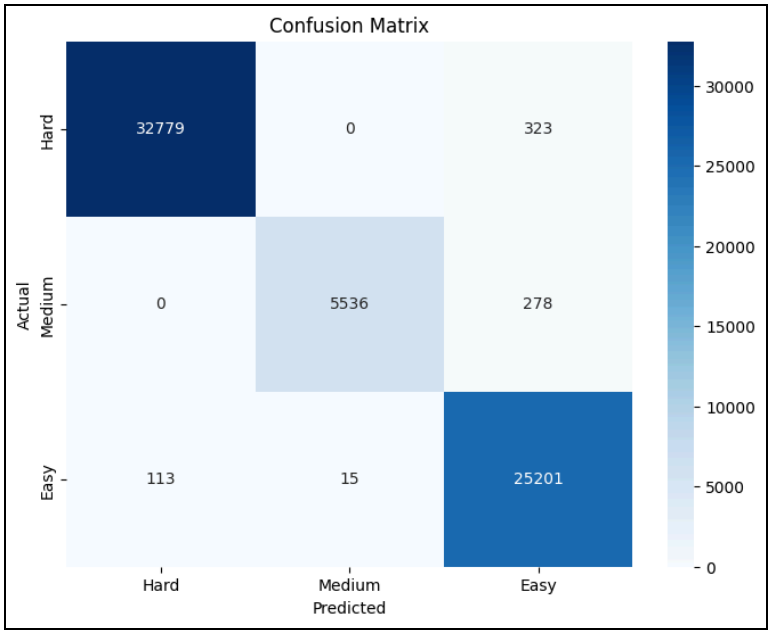
	precision	recall	f1-score	support
Easy	1.00	0.99	0.99	33102
Hard	1.00	0.95	0.97	5814
Medium	0.98	0.99	0.99	25329
accuracy			0.99	64245
macro avg	0.99	0.98	0.98	64245
weighted avg	0.99	0.99	0.99	64245

  The overall accuracy of the Random Forest classifier on the test data was 98.9%, closely matching the cross-validation results, which indicates no significant overfitting.

- **Discussion**

  The Random Forest classifier demonstrated exceptional performance in classifying questions into 'Easy', 'Medium', and 'Hard' categories. The high precision and recall across all classes, particularly the perfect precision for 'Easy' and 'Hard' questions, underscore the model's robustness and reliability. The slight drop in recall for 'Hard' questions to 0.95 suggests that while the model is highly effective, there remains a minor scope for improvement in correctly identifying all 'Hard' questions. The Random Forest model's superiority over the Naive Bayes classifier is evident from the significant increase in accuracy, precision, recall, and F1-scores. This performance boost can be attributed to the Random Forest's ability to handle feature interactions and its robustness against overfitting due to the ensemble learning approach. The confusion matrix further corroborates these findings, showing a clear and distinct classification across all difficulty levels with minimal misclassifications. The high accuracy and detailed metrics indicate that the Random Forest classifier is well-suited for the task of question difficulty classification, making it a valuable tool for adaptive learning systems that require precise and reliable difficulty ratings.

The confusion matrix shows that the majority of predictions are accurate, with very few misclassifications between the 'Medium' and 'Hard' categories. This visual representation reinforces the high performance metrics and highlights the model's effectiveness in distinguishing between different difficulty levels.



In conclusion, the Random Forest classifier significantly outperforms the Naive Bayes model and proves to be highly effective for classifying question difficulty. Its high accuracy and robustness make it an excellent choice for enhancing adaptive learning systems, providing reliable and objective difficulty ratings that can help tailor educational content to individual learners' needs.

3. **Support Vector Machines (SVMs)**

After attempting to utilize SVMs following Random Forests, we encountered significant processing time due to the large dataset size. Despite efforts to mitigate this issue by sampling data or reducing columns, we were unable to generate any output. Consequently, we concluded that SVMs were inefficient for our case due to the extensive computation time required.

4. Decision Trees

After observing poor performance with SVM, we made the decision to explore decision trees as an alternative approach.

Here are the results in detail:

- **Cross-Validation Results:**

  The Decision Tree classifier achieved the following cross-validation accuracy scores over five folds: [0.99069967, 0.99196436, 0.99138048, 0.99254791, 0.99058274]

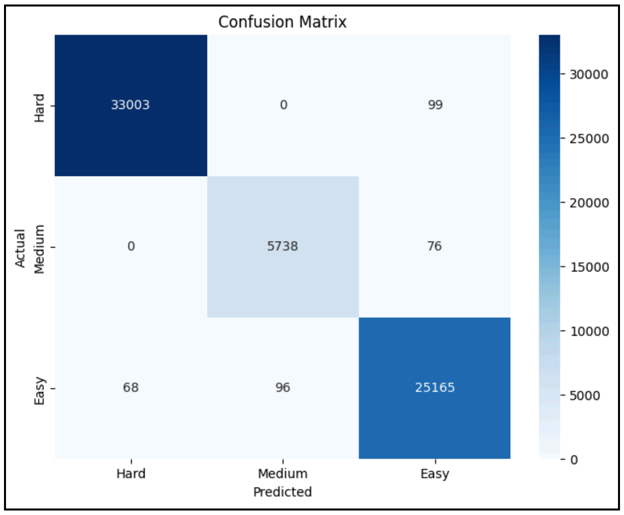
  Mean Accuracy: 0.9914350319524446

  These results indicate a very high performance across different subsets of the training data, with a mean cross-validation accuracy of approximately 99.1%.

Classification Report on Test Data:

	precision	recall	f1-score	support
Easy	1.00	1.00	1.00	33102
Hard	0.98	0.99	0.99	5814
Medium	0.99	0.99	0.99	25329
accuracy			0.99	64245
macro avg	0.99	0.99	0.99	64245
weighted avg	0.99	0.99	0.99	6424

- **Discussion**



The results from the Decision Tree classifier show very high precision, recall, and F1-scores for all difficulty levels: Easy, Medium, and Hard. This suggests the model is performing exceptionally well on the training data. However, such perfect scores might indicate that the model is memorizing the training data too closely, which could lead to poor performance on new, unseen data. While the overall accuracy of 99% seems impressive, we need to be cautious about the model's ability to generalize to new situations. Further analysis and adjustments to the model may be needed to ensure it can reliably handle new data.

5. Neural Network

After seeing decision trees performance, we decided to try neural networks.

Here are the results in detail:

- **K-Fold Results:**

  The Neural Networks classifier achieved the following accuracy scores over five folds: [0.78751266, 0.784769, 0.781846,0.78554332 , 0.783928]

  Mean Accuracy: 0.7908008694648743

  These results indicate a good performance across different subsets of the training data, with a mean accuracy of approximately 79.1%.

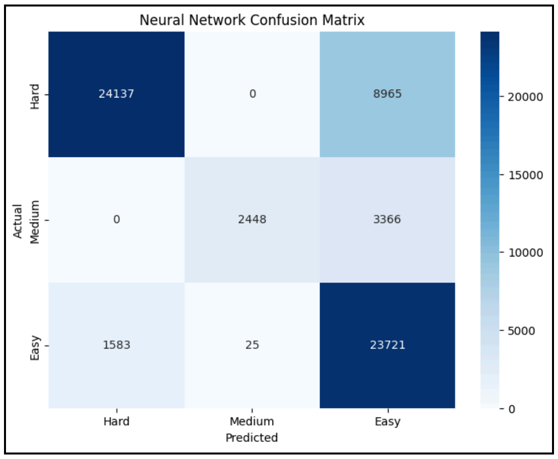
Classification Report on Test Data:

	precision	recall	f1-score	support
Easy	0.94	0.73	0.82	33102
Hard	0.99	0.42	0.59	5814
Medium	0.66	0.94	0.77	25329
accuracy			0.78	64245
macro avg	0.86	0.70	0.73	64245
weighted avg	0.83	0.78	0.78	64245

- **Discussion**

  The neural network proved to be a great model for our dataset, achieving an overall accuracy score of 78%. The detailed classification report further demonstrates its effectiveness across different difficulty levels.

  For the 'Easy' category, the neural network achieved strong performance, with a precision of 0.94, a recall of 0.73, and an F1 score of 0.82. This reflects the model's ability to correctly identify easy questions.



The 'Medium' category also showed solid results, with a precision of 0.66, a recall of 0.94, and an F1 score of 0.77. This indicates that the model is particularly adept at recognizing medium difficulty questions, though its precision could be improved.

In the 'Hard' category, the model's precision is impressively high at 0.99, but the recall is considerably lower at 0.42, resulting in an F1 score of 0.59. This discrepancy suggests that while the model is highly accurate when it predicts a hard question, it often fails to identify them, which could be due to dataset imbalance or the inherent complexity of harder questions.

Overall, the neural network's macro average scores of 0.86 for precision, 0.70 for recall, and 0.73 for F1 score, along with weighted averages of 0.83 for precision, 0.78 for recall, and 0.78 for F1 score, confirm its robust performance across all categories. These results indicate the model's ability to generalize and perform well on varying levels of question difficulty, making it a valuable tool for our classification task. However, targeted improvements, especially in the recall of hard questions, could further enhance its performance.

VI. Application and Implications

The Question Difficulty Classifier holds significant practical applications within educational settings, offering tangible benefits for both educators and students alike. By systematically categorizing questions into varying levels of difficulty, this tool empowers teachers to craft more effective learning materials and assessments tailored to the diverse needs of their students. With questions neatly organized into easy, medium, and hard categories, educators can ensure that assessments are appropriately challenging yet fair, providing students with opportunities to demonstrate their understanding across a range of proficiency levels. This approach not only promotes deeper engagement and comprehension but also enables educators to pinpoint areas where students may require additional support or intervention.

VII. Future work

We aim to enhance our model to not only classify individual questions but also to identify if a set of questions related to the same content are notably challenging. This enhancement will enable us to alert instructors when students encounter difficulty with specific content, prompting them to allocate additional time or resources to address these areas effectively. Looking ahead, ongoing advancements and refinements in the Classifier's algorithms could further enhance its accuracy and adaptability, ultimately leading to improved learning outcomes for students across a wide range of educational domains.

VIII. Conclusion

In summary, our study showcases the power of machine learning in educational technology by classifying question difficulty levels. Through rigorous analysis of student interaction data, we evaluated various classifiers like Random Forests, Decision Trees, Naive Bayes, and Neural Networks. The Random Forest classifier emerged as highly effective, offering objective difficulty ratings crucial for personalized learning experiences. Our research provides a scalable and data-driven framework to inform instructional design, overcoming the limitations of subjective expert judgment. By pinpointing challenging content areas, our model enables timely interventions and tailored support for students, fostering a culture of continuous improvement in teaching and learning. Ultimately, our work represents a significant step forward in educational assessment, promising transformative insights and opportunities for educational excellence through the fusion of machine learning and pedagogy.

IX. References

- *Foundations of Machine Learning -- CSCI-GA.2566-001.* (n.d.). https://cs.nyu.edu/~mohri/ml17/
- Kim, G. I., Kim, S., & Jang, B. (2023). Classification of mathematical test questions using machine learning on datasets of learning management system questions. *PloS One*, *18*(10), e0286989. https://doi.org/10.1371/journal.pone.0286989