Introducing a Framework for Code based Fairness Audits of Learning Analytics Systems on the Example of Moodle Learning Analytics

Hassan Tagharobi[a] and Katharina Simbeck[b]
University of Applied Sciences HTW Berlin, Berlin, Germany

Keywords: Code Audit, Algorithmic Fairness, Moodle, Learning Analytics.

Abstract: Machine learning based predictive systems are increasingly used in various areas, including learning analytics (LA) systems. LA systems provide educators with an analysis of students' progress and offer predictions about their success. Although predictive systems provide new opportunities and convenience, studies show that they harbor risks for biased or even discriminatory outcomes. To detect and solve these discriminatory issues and examine algorithmic fairness, different approaches have been introduced. The majority of purposed approaches study the behavior of predictive systems using sample data. However, if the source code is available, e.g., for open-source projects, auditing it can further improve the examination of algorithmic fairness. In this paper, we introduce a framework for an independent audit of algorithmic fairness using all publicly available resources. We applied our framework on Moodle learning analytics and examined its fairness for a defined set of criteria. Our fairness audit shows that Moodle doesn't use protected attributes, e.g., gender, ethnicity, in its predictive process. However, we detected some issues in data distribution and processing, which could potentially affect the fairness of the system. Furthermore, we believe that the system should provide users with more detailed evaluation metrics to enable proper assessment of the quality of learning analytics models.

1 INTRODUCTION

With the increasing use of machine learning (ML) algorithms in different areas, concerns arise about their ethics and fairness. Various studies demonstrated biased results of ML systems (Hardt et al., 2016; Mehrabi et al., 2021a) caused by algorithm, biased dataset, or improper use of the system (Mehrabi et al., 2021b).

Online learning platforms such as Moodle implement ML in their learning analytics functionalities, by offering educators predictions on student progress and allowing for early interventions. However, these predictive systems are prone to the same fairness issues. If educators use the prediction results in their approach or grading decision, this can have a major impact on student success. Educational institutions thus need to assess the fairness of learning analytics before implementing it.

Assessing the fairness of predictive systems has become a widely discussed and researched topic (Aydemir & Dalpiaz, 2018; Baker & Hawn, 2021; Kim, 2017; Mehrabi et al., 2021b; Riazy et al., 2021) As the source code of predictive systems is often not available, most of the studies examine the behavior of these system using sample datasets (Adler et al., 2018; Flavio Calmon et al., 2017). The fairness of predictive systems can be assessed through an audit (Raji & Buolamwini, 2019), which can be conducted both internally (by the creator of the system) or externally (Wilson et al., 2021). In the case of open-source software, independent auditors can access the source code as well. In this work, we purpose a framework for an independent code audit. We will discuss our approach to collect, confine, map, and analyze the code and thus assess the fairness of the system. Furthermore, we will apply our code audit approach to the Moodle learning analytics functionality and assess its fairness.

[a] https://orcid.org/0000-0003-0962-5173
[b] https://orcid.org/0000-0001-6792-461X

Tagharobi, H. and Simbeck, K.
Introducing a Framework for Code based Fairness Audits of Learning Analytics Systems on the Example of Moodle Learning Analytics.
DOI: 10.5220/0010998900003182
In *Proceedings of the 14th International Conference on Computer Supported Education (CSEDU 2022) - Volume 2*, pages 45-55
ISBN: 978-989-758-562-3; ISSN: 2184-5026

2 THEORETICAL BACKGROUND

2.1 Algorithmic Fairness in Learning Analytics

Learning analytics (LA) has been defined as “the measurement, collection, analysis, and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs” (Siemens & Long, 2011). Grandl et al. (2017) name five major goals of learning analytics as predictions and interventions, recommendations, benchmarking, personalization and adaption as well as reflection and iteration.

Research in the area of learning analytics increased quickly in recent years (Dawson et al., 2019), with many contributions on the prediction of student performance (Riazy et al., 2021). Since predictive systems could potentially generate unfair results, concerns over privacy and fairness of learning analytics systems have risen accordingly (Drachsler et al., 2015). Despite a low number of studies about ethics and fairness in LA until 2018 (Dawson et al., 2019), in recent years an increasing number of research studies focusing on algorithmic fairness in LA (Baker & Hawn, 2021). The fairness of predictions of student success is critical since they influence the grading decisions of educators (Mai et al., 2021).

The fairness of predictive systems can be measured at different levels: individual, group, or subgroup (Mehrabi et al., 2021b). For this purpose, different approaches have been introduced. For example, Lu et al. (2018), or Panigutti et al. (2021) propose tools to examine fairness. Galhotra et al. (2017) on the other hand propose a testing algorithm, that generates test cases to measure fairness and discrimination. Other contributions studied fairness in specific cases. For example, Anderson et al. (2019) examine the fairness of graduation predictions. These fairness measurement approaches use data. However, we believe examining of the system prior to introducing data could provide additional benefits in assessing fairness.

2.2 Audit & Code Audit

ISO defines audit as a “systematic, independent and documented process for obtaining objective evidence and evaluating it objectively to determine the extent to which the audit criteria are fulfilled” (ISO 19011:2018, 2018). Audits have long been used in different fields to ensure quality and compliance of a system to standards (Power, 1994), for example in finance (Cohen et al., 2002), or quality assurance (Woodhouse, 2003).

There are different types of audits: internal, external, or third-party audits, which are defined by the relationship between auditors and audited organizations or systems (ISO 19011:2018, 2018).

In software development, audits are also seen as a reviewing method of the development process and of software systems (IEEE 1028:2008, 2008). Code audits can be defined as “the process of extracting information about a program from its source code” (Binkley, 2007). Quality assurance and detecting potential security vulnerabilities are usually the main focus of software audits (Yang et al., 2016).

With the increasing use of machine learning arises the importance of reviewing the behavior of these systems. The proposed European Artificial Intelligence Act (AIA) defines periodic audits as a quality measure in the development of AI systems (EUROPEAN COMMISSION, 2021). AIA distinguishes low, high, and unacceptable risk categories for those systems and imposes different regulations and measures for each category (Mökender et. al., 2021). Brwon et. al. (2021) note that AI audits could be used by regulators to assess the compliance with standards or by stakeholders and users to assess ethical aspects of an algorithm. VDE registers code audit as an option for the development of trustworthy AI systems (VDE-AR-E 2842-61, 2021). Audits can be used to determine the algorithmic fairness of machine learning based systems. Sandvig et. al propose five different audit designs: noninvasive user audit, scraping audit, sock puppet audit, collaborative or crowdsourced audit, and code audit (Sandvig et al., 2014). The first four designs analyze the behavior of systems using input and output, whereas in code audit the source code is analyzed in detail for potential problems. Despite its advantages and the growing number of open-source projects, code audit is still an under-explored audit method (Bandy, 2021).

Other audit designs aim to analyze all available resources, including source code and training data. Raji et al. propose an internal audit framework to determine algorithmic fairness (Raji et al., 2020). This framework includes mapping and collecting all artifacts, test and reflect stages. Wilson et al. (2021) adapt this framework for an external audit. They use source code, documentation, and a dataset provided by an HR platform to examine the fairness of a candidate-search system. Furthermore, Brundage et al. (2020) propose a mechanism for external audits to provide evidence about the safety, security, fairness, and privacy protection of AI systems. The proposed mechanism is organized into institutional, software, and hardware levels. Moreover, Hauer et al. (2021) propose a non-deterministic process including acceptance test-driven development and assurance cases as a complementary tool to audits in order to ensure fairness of predictive systems.

2.3 Moodle

Moodle is an open-source online learning management system, which is developed by the company Moodle HQ. It has more than 290 million users worldwide (Moodle, 2021e) and, is one of the most popular learning platforms (Edutechnica, 2021). Moodle HQ provides regular minor and major updates to the platform. Moodle includes a learning analytics (LA) system, which offers educators analyses of students' learning and makes predictions about their success chances (Moodle, 2021c). The use of the LA system is optional.

There are two kinds of LA models in Moodle: static and machine learning (ML) based LA (Moodle, 2021d). Static LA uses simple rules to recognize specific situations, whereas ML-based LA predicts events, e.g., "students at risk of dropping out" (Moodle, 2021c). The provided ML-based LA models are not pre-trained. Thus, users (institutions) should administer the ML-process, e.g., data collection, training, evaluation, and prediction locally (Moodle, 2021a).

Moodle and its LA component have been the subject of research, e.g., (Dimopoulos et al., 2013; Mwalumbwe & Mtebe, 2017). These focused mainly on predicting student success, e.g. (Zhang, Y., Ghandour, A., & Shestak, V., 2020), using plugins to add or analyze student activities, e.g. (Liu et al., 2019). Most of these research studies focus on the performance of Moodle LA. Bognár et al. (2021) studied dataset impact on the Moodle LA predictors (Bognár et al., 2021), however algorithmic fairness of Moodle LA has not been studied. Therefore, we believe a comprehensive audit of Moodle LA for fairness constitutes a research gap. In this paper, we address this gap with our purposed independent audit framework.

3 DESIGNING AN INDEPENDENT CODE AUDIT FOR FAIRNESS

In this paper, we propose a framework for an independent fairness audit, executed by third party auditors on publicly available source code. We build our audit process according to IEEE 1028-2008 standard for software review and audit, section 8 (IEEE 1028:2008, 2008). Raji et al. (2020) propose a framework for fairness audits. They distinguish between an internal audit (audit is conducted by company's employees), and an external audit, conducted by experts without access to code and models. Our approach does not fall strictly into one of the internal or external categories. However, we will generally adapt the proposed internal fairness-audit framework (Raji et al., 2020) and adjust the framework to our independent audit process. Our framework is applicable when access to source code for auditors is assured, for example in the case of open-source projects. The framework contains five main steps: a) definition of the scope of the audit; b) artifact collection & source- code confinement; c) mapping and analysis of relevant components; d) fairness assessment process; and e) audit results interpretation.

3.1 Definition of Scope of the Audit

The goal of an independent audit is to offer an independent evaluation of the subject. The audit team clarifies the goal and purpose of the audit process and consequently, defines the scope of the audit (what will and will not be examined) as well as the audit criteria (IEEE 1028:2008, 2008). It is important, that the audit criteria are deterministic using available resources (ISO 19011:2018, 2018, pp. 43–44). Furthermore, the procedure to examine the source code and to assess compliance with audit criteria is specified in this step (VDE-AR-E 2842-61, 2021). In the case of auditing a predictive learning analytics system, different fairness aspects can be considered. We suggest three main criteria for a fairness code audit. These criteria are intended to identify the potential unfair behavior of the system at different stages of the predictive system, and each focuses on a particular aspect of fairness. Together these should be able to provide a measure for the fairness of the system. The proposed criteria are as follows:

a) **Direct Discrimination:** *Does the system process protected attributes, e.g., gender or ethnicity in predictive process? Is there evidence of direct discrimination in the source code?* The use of protected attributes, e.g., gender or ethnicity, in any step of a predictive system, e.g., feature extraction, training, and prediction, can lead to direct discrimination. The system could learn and imply falsely a relationship between these attributes and the performance of students. Thus, we propose a systematic audit of all classes and functions for evidence of using protected attributes, or signs of other direct discrimination in the source code.

b) **Fair Data Processing:** *Are ML-crucial-processes in the code, e.g., training, evaluation applied properly to ensure fairness of ML-algorithm? Is the fairness ensured in data process, e.g., data selection, distribution?* In addition to the above-mentioned systematic review, we propose a process review of specific ML-crucial-processes in the code, such as training or evaluation. The focus should be on ensuring proper application of ML-algorithm and detecting possible problems regarding fairness in data process, e.g., data selection, distribution, etc.

c) **Proper Assumptions and Parametrization:** *Do assumptions and parametrization of machine learning process ensure fairness of results? Is there a potential for unfair prediction due to assumptions in the source code?* Proper assumptions and parameters can affect the accuracy and fairness of ML-based predictive systems. Developers can either choose to let users or admins define parameters or hard code them in the source code. We recommend examining the assumptions and parametrization in the source code and their resulting impact on the outcome. This can be done with or without sample data.

3.2 Artifact Collection & Confinement

Prior to starting any audit, all relevant information and artifacts must be collected (ISO 19011:2018, 2018, p. 52). These could be source code, documentation, data, developer comments etc. (IEEE 1028:2008, 2008). For an independent third-party audit, the artifacts are limited to publicly available material. In the case of open-source projects, these are usually source code, developer comments and documentation. Depending on the subject of an audit, confinement of audit subject could be necessary, in order to properly allocate the resources and reach the best possible results. For example, if a predictive system is the audit subject, the relevant data-processing and machine learning components of the source code could be identified and confined for the audit. Other parts of the source code can be deemed as irrelevant to the audit.

3.3 Mapping, Description, and Prioritization of Relevant Components

As a preprocessing step to the assessment of audit criteria, the identified relevant components, their attributes, and relationships are mapped and analyzed. These attributes could be, for example, location, purpose, related classes and functions, input, and output. The information gained from this step builds the foundation for the assessment in the audit. The result of this analysis enables auditors to understand the dependencies between components and offers the opportunity to mark areas of interest in the source code. This step can be implemented at different levels, e.g., modules, classes, functions. We propose this analysis at least at two levels: classes and functions. Mapping classes offers an overview of interdependency of different components, whereas surveying functions enables a more detailed code analysis.

Furthermore, while manually mapping the functions we suggest to simultaneously flag each function as “not relevant”, “probably relevant” or “relevant” for the chosen audit criteria. For the above-mentioned criteria, we propose the following classification criteria:

a) Direct Discrimination:

Relevant	User data is processed in the function.
Probably relevant	Sample data is processed but not directly user’s data.
Not relevant	Otherwise

b) Fair Data Processing:

Relevant	Function is part of predictive process, e.g., training, evaluation.
Probably relevant	Function is connected to predictive process, e.g., as feature calculation.
Not relevant	Otherwise

c) Proper Assumption and Parametrization:

Relevant	Assumptions regarding predictive processes, e.g., training, evaluation, are made in the function.
Probably relevant	Function contains assumption connected to predictive process, e.g., in feature calculation.
Not relevant	Otherwise

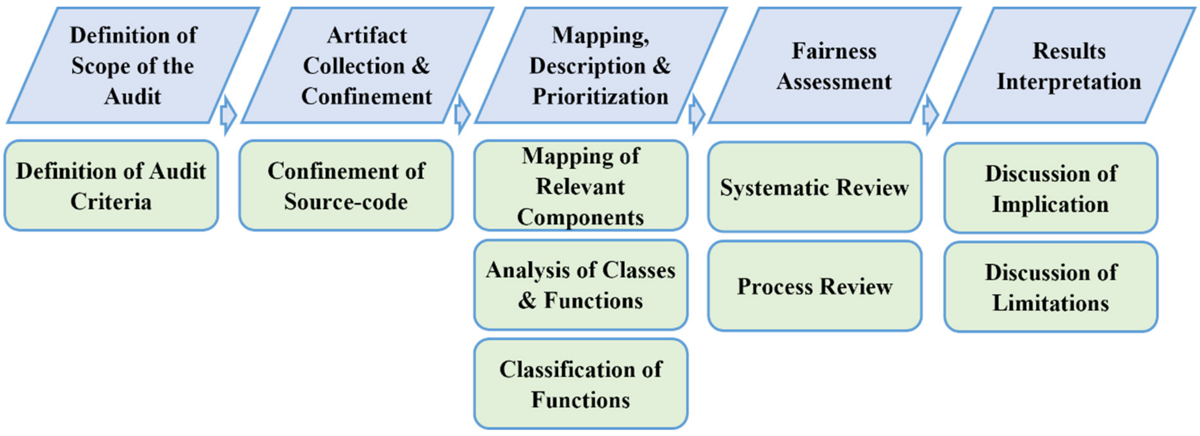
Definition of Scope of the Audit
Artifact Collection & Confinement
Mapping, Description & Prioritization
Fairness Assessment
Results Interpretation
Definition of Audit Criteria
Confinement of Source-code
Mapping of Relevant Components
Analysis of Classes & Functions
Classification of Functions
Systematic Review
Process Review
Discussion of Implication
Discussion of Limitations

Figure 1: Proposed Process for an independent fairness audit.

3.4 Fairness Assessment

Next, the auditor follows the specified procedure and examines the compliance of code with audit criteria. There are different ways to perform an audit on a source code: systematic or process review. In a systematic review, every line of code is examined for compliance with defined criteria, whereas in process review a process and its components are evaluated in connection with each other. Systematic review ensures the precision of assessment for a criterion due to the examination of each line of code. However, for complex processes, it is not enough to analyze lines of source code individually, but combined in a process. A parameter set to a specific value in one function, can affect the outcome of another function in the same process later. Thus, it is vital to assess functions in relation to each other in a process. For the systematic review, auditors provide an assessment for each unit, e.g., functions, classes, packages. Whereas, in process review, auditors provide an assessment for the examined process.

3.5 Audit Results Interpretation

In the last step, the audit team summarizes and interprets the outcome of assessment. The results and their implication on the audit criteria are analyzed. At the end of the process, the audit team is able to determine the compliance of audit subject to the defined audit criteria.

4 CASE STUDY: FAIRNESS AUDIT OF THE MOODLE LEARNING ANALYTICS FUNCTIONALITY

We applied our proposed framework for an independent audit to the Moodle learning platform to examine fairness of its learning analytics components. We will follow the above-mentioned steps in this case study.

4.1 Definition of Scope of the Audit

Moodle is a widely used open-source online learning platform. Its learning analytics (LA) components aim to analyze students' learning progress and predict their success. The goal of our audit was to determine if the Moodle LA predictive system can be considered as fair.

We specified the purpose of audit as well as the audit evaluation criteria. We used the audit criteria as discussed in section 3.1.

4.1.1 In Scope

According to our suggested criteria in 3.1, we defined following criteria for our fairness audit of Moodle LA:

a) **Direct Discrimination:** *Does Moodle LA use protected attributes, e.g., gender or ethnicity in predictive process?*
We will systematically audit all classes and functions in Moodle LA and examine if there is evidence of using protected attributes or signs of other direct discrimination in the source code.

b) **Fair Data Processing:** *Are ML-crucial-processes in Moodle LA applied properly to ensure fairness of LA-predictions?*
We will conduct a process review of specific, crucial ML processes in the Moodle LA source code, such as training, evaluation, prediction. In the process review, all functions of the process are analyzed together in the call order of the process. Thus, the influence of different functions or global variables is taken into consideration. A process review can be done using debugging tools of an integrated development environment (IDE). The focus of the process review for this criterion will be detecting potential problems regarding fairness in data processing, e.g., data distribution.

c) **Proper Assumptions and Parametrization:** *Do assumptions and parametrization in Moodle LA ensure fairness of LA-results?*
We will examine the assumptions and parametrization in ML-processes of Moodle LA source code and their resulting impact on the outcome. This will be done without sample data.

4.1.2 Out of Scope

While conducting the audit, we will neglect other aspects of algorithmic fairness. The following criteria are out of scope of our audit:

a) Data Impact:
Data can impact the fairness of a system in different ways. We neglected the data impact and following aspects of data impact in our audit:
  i. **Dataset Suitability and Stability:** ML-based predictive systems rely on data to train and to predict. Since Moodle LA models must be trained by users, this could lead to various fairness problems. Due to the source code audit nature of this audit, dataset impact cannot be evaluated.
  ii. **Anonymization:** proper data anonymization can increase the fairness of predictive systems. This audit will not consider this process and its effect on fairness evaluation.

b) **Role of Users:** The real-world consequences of AI systems arise through user behavior, e.g., by defining features and targets or by using system output. Since this audit only focuses on source code, we will neglect the impact of users, their knowledge, and proper application of the system.

c) **Bias towards Digital Data:** By definition, a learning analytics system will only use data from digital learning processes and neglect offline learning activities. This bias towards digital data will distort results and can potentially lead to unfairness.

d) **Assessment of ML-Methods:** For similar tasks, various ML-Methods can be used with comparable or different levels of accuracy on a general level and comparable or different results on an individual level. The Moodle source code contains implementations of various ML-methods; however, only two methods are employed. Without running the models on real data, the suitability of the selected approaches cannot be determined.

4.2 Artifact Collection & Confinement

Moodle is an open-source project. The source code as well as a developer and user documentation are publicly available. For our audit, we collected these resources. As mentioned, the delivered predictive models in Moodle are not pre-trained. For this audit, we did not use any further information or data sets.

We focus our audit on the LA components of Moodle, which were pre-collected. We respectively performed this confinement on the documentation too.

4.3 Mapping, Description, and Prioritization of Components

The mapping, description and prioritization of components was conducted manually. Firstly, classes in Moodle LA were mapped with their properties, e.g., location, purpose, parents, children classes (if applicable). Secondly, functions and their properties were analysed (location, belonging class, input and output values, what does the function do). Additionally, we listed all other classes and functions called in each function. Furthermore, while manually mapping the functions we simultaneously flagged each function as “not relevant”, “probably relevant” or “relevant” for each of criteria described in section 3.3.

4.4 Fairness Assessment

In our audit process, we both used a systematic review for our first criterion and a process review for second and third criteria. These were conducted following the criteria of a) direct discrimination (system review) b) fair data processing (process review), and c) proper assumptions and parametrization (process review). In a), all functions, which were flagged as relevant or probably relevant were examined in more detail to determine if protected attributes of users, e.g., gender or ethnicity, were used, especially in predictive processes. For each case, an assessment was made and reported. Consequently, each function was marked as critical or not critical for this criterion. For steps b) and c) all relevant functions were assigned to one of the five major processes in machine learning (create, train, evaluate, predict, and delete a model) We executed each of these processes in a debug-test environment, following the sequence of functions, and assessing compliance to the criteria. As a result, all these processes and their associated functions were reported as critical or not critical for these criteria.

4.5 Results Interpretation

Finally, we summarized the above-mentioned results of the audit and made a verdict about each criterion. In the following section, we will discuss these results and their implication for the fairness of Moodle LA.

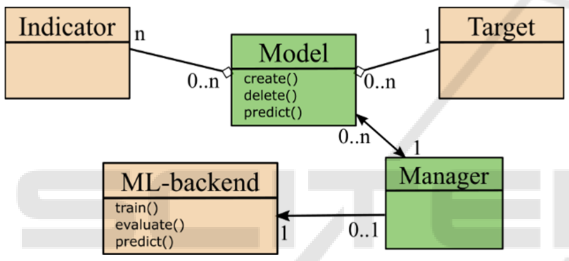

Figure 2: Simplified class diagram of Moodle-LA, based on (Moodle, 2021b).

5 AUDIT RESULTS OF MOODLE LA

5.1 Artifact Collection & Confinement

We performed our audit on Moodle version 3.10.4 as of May 2021. As for the artifacts, we only used the publicly available source code and documentation. This version contains 2,6 million lines of code (LOC) in 12 thousand classes (Tagharobi & Simbeck, 2021). There are user and developer documentations. The developer documentation is much shorter than the user documentation; it does not contain a comprehensive structure or code explanation. For our audit we identified and selected only the relevant components for learning analytics in Moodle. These are in four packages and contain about 35 thousand LOC, mainly in PHP. Our audit was restricted only to these components. Although we cannot completely rule out that other part of the project could influence the learning analytics components, we firmly believe, that we reviewed all the main LA-components in our audit. We performed our code audit manually using a common IDE. We should note, that there are various static code analysis tools e.g., SourceQube, DeepSource, that can support a code analysis. However, these tools are mainly built to find and fix programming errors or security vulnerabilities. Thus, we chose a manual code audit to achieve a thorough examination of the source code.

5.2 Mapping, Description and Prioritization of Components

As the next step, we mapped and analyzed the identified components. These were 237 classes and about 1,3 thousand functions distributed in four different locations in the source code structure. We extracted the attributes of classes and functions as listed in section 4.3. The most important part of this analysis was the determination of the purpose and task of each function. Since the source code is not sufficiently commented in Moodle, we had to perform this task manually for all classes and functions. Furthermore, we believe that the identification of other classes and functions called in a function helped the better understanding of the composition of processes and the relationship between classes and functions.

It is worth noting, that our analysis shows that not all implemented classes and functions are used in Moodle LA. For example, different machine learning algorithms are implemented in the source code but only two, logistic regression & neural network, are used. For the sake of completeness, we have mapped and analyzed the not-used functions as well. However, we did not consider them for further assessment since they are not executed at all.

We marked functions relevance to the audit criteria according to the procedure as defined in 4.3. Table 1 shows the number of as relevant or probably relevant functions for each criterion.

Table 1: Audit criteria and number of functions marked as relevant for each criterion.

Criterion	Function relevant	Function probably relevant
Direct discrimination	30	83
Fair data processing	162	237
Proper assumptions and parametrization	136	167

Functions, which were flagged as relevant, were in the focus of the fairness assessment process.

For the assessment of the second and third criteria, we classified the functions into the step of an ML-process (create, train, evaluation, predict, delete). Table 2 shows the number of functions identified as relevant for each of these steps.

Table 2: Number of functions that are relevant for each process in Moodle LA.

Process	create	train	eval	predict	Del
Number of relevant functions	21	61	44	43	2

It is worth noting that some functions were classified as relevant for more than one process. For example, feature calculation functions are relevant for train, evaluation, and prediction.

5.3 Analysis of Five Major ML-processes in Moodle LA

Before discussing our fairness assessment result, we will describe the implementation of the five major ML-processes in Moodle LA. In section 5.3, we will share our verdict about compliance of these processes with our defined criteria:

a) Create: Users build a new LA-model by selecting a target and appropriate indicators from a given list. Therefore, we reviewed all functions, which contain the indicators and targets, as well as functions, that create models. The current indicators process different activities of students on Moodle.

b) Train: There are two different machine learning backends in Moodle. The default backend is a php-based system, but there is also the possibility of using a python backend. The default backend uses a logistic regression algorithm and the python backend employs a neural network. In both cases, the system first checks data availability, e.g., prior completed courses with user activities for the chosen indicators. Then, it calculates the outcome of indicators and, trains the model for the given target.

c) Evaluate: In this process, the data is split into training and evaluation sets. After initiating training, the system predicts results for the evaluation set. It calculates F1-score using ground truth and the prediction. Finally, the results of the evaluation of the model are presented to users.

d) Predict: The system first checks, if a model is trained. If not, it initiates a training process. Using the trained model and results of chosen indicators, the system makes a prediction for given samples (students). The prediction result is saved to the database, overwriting possible older results for the same model and samples.

e) Delete: All generated predictions are removed from the database and the chosen model is deleted.

5.4 Fairness Assessment

In this section, we discuss our assessment results for each criterion and make our verdict about the compliance of Moodle LA with our defined criteria.

5.4.1 Direct Discrimination (Systematic Review)

We analyzed the 113 functions classified as relevant or probably relevant. These are mainly functions that process and calculate features (“indicators” in Moodle) or define an action for intervention. From all these functions, there were only two, which used private user attributes, e.g., name or picture. These attributes could reveal user gender, age, ethnicity, etc. However, although these functions are part of Moodle LA modules, they are not included and not used for the predictive process.
Our verdict: We did not find evidence for direct discrimination in our audit. Thus, we can state that Moodle learning analytics does not use protected attributes, e.g., gender or ethnicity, or apply other direct discriminations in the predictive part of Moodle-LA source code.

5.4.2 Fair Data Processing (Process Review)

As discussed, we divided the process review into the five ML steps and will discuss our results accordingly. The following results are solely based on the source code analysis, without running analyses using sample data.

Create: we did not find evidence of biased action or unfair process or assumption in the code. Thus, we assert, that this process complies with our criteria.

Train: In this process, we identified some critical assumptions. For example, there is a data limit of 500 MB for the train-process. If a dataset is larger than this limit, it cuts the dataset at this limit. At this point, the system does not ensure the diversity of a selected dataset. In the worst case, the dataset is therefore lopsided and not ideal for training.

Evaluate: We identified a critical aspect in this evaluation process. The system only checks if there are at least two examples for each target, in order to ensure diversity of datasets. We believe, two samples are not sufficient for data diversity in a predictive system and can lead to inaccurate results.

Predict: the predict process does not use biased assumptions and there is no evidence of unfair data process.

Delete: we didn't find evidence of biased or unfair process in delete process.

5.4.3 Proper Assumption and Parametrization (Process Review)

We examined the five ML steps for compliance with this criterion. For create, predict or delete steps we did not find evidence of assumptions that could lead to unfair results.

In the evaluate step, we assess the fact that Moodle calculates and presents only F1-score to the user critically. Even though F1-score is a great metric to compare two or more machine learning algorithms, we believe this metric alone shall not be used to assess model quality. If the users lack the necessary in-depth knowledge of evaluation metrics they cannot properly infer the accuracy of the system solely based on this metric. Thus, the user could wrongly assume a much more accurate prediction result than what the model delivers. Calculating and displaying various evaluation metrics along with their implications could help users to better assess the prediction results and improve the fairness of the system.

6 CONCLUSION & DISCUSSION

In this paper, we introduced a framework for an independent audit for algorithmic fairness. The framework consists of five major steps: a) definition of the scope; b) artifact collection & confinement; c) mapping and analysis of relevant components; d) fairness assessment process; and e) audit results interpretation. This framework can be applied to projects with available source code, to assess their fairness. We applied this framework to Moodle learning analytics and examined its fairness for three major criteria: 1) does Moodle LA use protected attributes in a predictive process? 2) Are crucial ML -processes in Moodle LA applied properly to ensure fairness of the LA-predictions? 3) Do assumptions and parametrization in Moodle LA ensure fairness of the LA-results? We followed the steps of our framework and thoroughly examined the relevant components of Moodle LA.

We can state that we did not find evidence of using protected attributes in Moodle LA. With regards to fair data processing and proper assumption for a fair system, we identified some critical aspects in Moodle LA. Firstly, some assumptions on the distribution of data samples in train and evaluation processes can lead to imbalanced data and thus, biased results. Secondly, Moodle LA uses only the F1-score to show the users the quality of an LA-model. We believe that the use of this evaluation metric only cannot properly demonstrate the quality of an LA-model, its strength and weaknesses, to the users.

It is worth noting that other factors can influence the verdict and fairness of a given Moodle LA. Beside above-mentioned factors like user or data impact, we want to mention that in an open-source project, it is possible to add or modify the code used in any instance. Thus, it cannot be ruled out that such modification or additional modules, e.g., indicators, would change the verdict. Therefore, the provided statements are only limited to original version of Moodle without any additional components or modification.

Moreover, we want to reiterate, that our assessments for fair data processing and proper assumptions were solely based on the source code. Thus, our presented results are limited to the code. A predictive system depends strongly on provided data. We firmly believe that our detailed analysis and assessment of the code was able to assess the code quality regarding fairness. Nevertheless, further analysis of the system with sample data could increase the certainty of the fairness audit results.

ACKNOWLEDGEMENTS

This research has been funded by the Federal Ministry of Education and Research of Germany in the project "Fair Enough?" (Project ID: 16DHB4002).

REFERENCES

Adler, P., Falk, C., Friedler, S. A., Nix, T., Rybeck, G., Scheidegger, C., Smith, B., & Venkatasubramanian, S. (2018). Auditing black-box models for indirect influence. *Knowledge and Information Systems*, *54*(1), 95–122. https://doi.org/10.1007/s10115-017-1116-3

Anderson, H., Boodhwani, A., & Baker, R. S. (2019). Assessing the Fairness of Graduation Predictions. *EDM*.

Aydemir, F. B., & Dalpiaz, F. (2018). A roadmap for ethics-aware software engineering. In Y. Brun, B. Johnson, & A. Meliou (Eds.), *Proceedings of the International Workshop on Software Fairness* (pp. 15–21). ACM. https://doi.org/10.1145/3194770.3194778

Baker, R. S., & Hawn, A. (2021). Algorithmic Bias In Education. *International Journal of Artificial Intelligence in Education*, 1-41.

Bandy, J. (2021). Problematic Machine Behavior. *Proceedings of the ACM on Human-Computer Interaction*, *5*(CSCW1), 1–34. https://doi.org/ 10.1145/3449148

Binkley, D. (2007). Source Code Analysis: A Road Map. In *Future of Software Engineering (FOSE '07)*. IEEE. https://doi.org/10.1109/fose.2007.27

Bognár, L., Fauszt, T., & Nagy, G. Z. (2021). Analysis of Conditions for Reliable Predictions by Moodle Machine Learning Models. *International Journal of Emerging Technologies in Learning (IJET)*, *16*(06). https://doi.org/10.3991/ijet.v16i06.18347

Brown, S., Davidovic, J., & Hasan, A. (2021). The algorithm audit: Scoring the algorithms that score us. *Big Data & Society*, *8*(1). https://doi.org/10.1177/ 2053951720983865

Brundage, M., Avin, S., Wang, J., Belfield, H., Krueger, G., Hadfield, G., Khlaaf, H., Yang, J., Toner, H., Fong, R., Maharaj, T., Koh, P. W., Hooker, S., Leung, J., Trask, A., Bluemke, E., Lebensold, J., O'Keefe, C., Koren, M., . . . Anderljung, M. (2020). *Toward Trustworthy AI Development: Mechanisms for Supporting Verifiable Claims*.

Cohen, J., Krishnamoorthy, G., & Wright, A. M. (2002). Corporate Governance and the Audit Process *Contemporary Accounting Research*, *19*(4), 573–594. https://doi.org/10.1506/983M-EPXG-4Y0R-J9YK

Dawson, S., Joksimovic, S., Poquet, O., & Siemens, G. (2019). Increasing the Impact of Learning Analytics. *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, 446–455. https://doi.org/10.1145/3303772.3303784

Dimopoulos, I., Petropoulou, O., Boloudakis, M., & Retalis, S. (2013). *Using Learning Analytics in Moodle for assessing students' performance*.

Drachsler, H., Hoel, T., Scheffel, M., Kismihók, G., Berg, A., Ferguson, R., Chen, W., Cooper, A., & Manderveld, J. (2015). Ethical and privacy issues in the application of learning analytics. In J. Baron, G. Lynch, N. Maziarz, P. Blikstein, A. Merceron, & G. Siemens (Eds.), *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge* (pp. 390–391). ACM. https://doi.org/10.1145/2723576.2723642

Edutechnica (Ed.). (2021, November 2). *LMS Data – Spring 2021 Updates | edutechnica*. https://edutechnica.com/2021/06/21/lms-data-spring-2021-updates/

EUROPEAN COMMISSION. (2021). *Proposal for a regulation of the European Parliament and the Council laying down harmonised rules on Artificial Intelligence (AIA) and amending certain Union legislative acts.*

Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, & Kush R. Varshney (2017). Optimized Pre-Processing for Discrimination Prevention. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 3995–4004.

Galhotra, S., Brun, Y., & Meliou, A. (2017). Fairness testing: testing software for discrimination. In E. Bodden, W. Schäfer, A. van Deursen, & A. Zisman (Eds.), *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering* (pp. 498–510). ACM. https://doi.org/10.1145/3106237.3106277

Grandl, M., Taraghi, B., Ebner, M., & Leitner, P. (2017). *Learning analytics.*

Hardt, M., Price, E., & Srebro, N. (2016, October 7). Equality of Opportunity in Supervised Learning. *Advances in neural information processing systems*, 29, 3315-3323.

Hauer, M. P., Adler, R., & Zweig, K. (2021, April). Assuring Fairness of Algorithmic Decision Making. *In 2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW) (pp. 110-113).*

IEEE 1028:2008. (2008). Ieee standard for software reviews and audits: Software & Systems Engineering Standards Committee of the IEEE Computer Society. *Institute of Electrical and Electronics Engineers.* http://ieeexplore.ieee.org/servlet/opac?punumber=460 1582

ISO 19011:2018 (2018). ISO 19011:2018: Guidelines for auditing management systems.

Kim, P. T. (2017). Auditing Algorithms for Discrimination. *U. Pa. L. Rev. Online*(166).

Liu, D. Y.T., Atif, A., Froissard, J. C., & Richards, D. (2019). An enhanced learning analytics plugin for Moodle: Student engagement and personalised intervention. In *ASCILITE 2015-Australasian Society for Computers in Learning and Tertiary Education, Conference Proceedings.*

Lu, O. H., Huang, A. Y., Huang, J. C., Lin, A. J., Ogata, H., & Yang, S. J. (2018). Applying learning analytics for the early prediction of Students' academic performance in blended learning. *Journal of Educational Technology & Society*(21(2)), 220–232.

Mai, L., Köchling, A., & Wehner, M. (2021). 'This Student Needs to Stay Back': To What Degree Would Instructors Rely on the Recommendation of Learning Analytics? *CSEDU (1)*, 189–197. https://doi.org/10.5220/0010449401890197

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021a). A Survey on Bias and Fairness in Machine Learning. *ACM Computing Surveys*, *54*(6), 1–35. https://doi.org/10.1145/3457607

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021b). A Survey on Bias and Fairness in Machine Learning. *ACM Computing Surveys*, *54*(6), 1–35. https://doi.org/10.1145/3457607

Mökander, J., Axente, M., Casolari, F., & Floridi, L. (2021). Conformity Assessments and Post-market Monitoring: A Guide to the Role of Auditing in the Proposed European AI Regulation. *Minds and Machines.* Advance online publication. https://doi.org/10.1007/s11023-021-09577-4

Moodle (Ed.). (2021a, May 18). *Analytics API - MoodleDocs*. https://docs.moodle.org/dev/Analytics_API

Moodle (Ed.). (2021b, May 18). *Machine learning backends - MoodleDocs*. https://docs.moodle.org/dev/Machine_learning_backends

Moodle (Ed.). (2021c, May 18). *Using analytics - MoodleDocs*. https://docs.moodle.org/310/en/Using_analytics

Moodle (Ed.). (2021d, June 7). *Analytics - MoodleDocs*. https://docs.moodle.org/311/en/Analytics

Moodle (Ed.). (2021e, November 2). *Moodle statistics*. https://stats.moodle.org/

Mwalumbwe, I., & Mtebe, J. S. (2017). Using Learning Analytics to Predict Students' Performance in Moodle Learning Management System: A Case of Mbeya University of Science and Technology. *The Electronic Journal of Information Systems in Developing Countries*, *79*(1), 1–13. https://doi.org/10.1002/j.1681-4835.2017.tb00577.x

Panigutti, C., Perotti, A., Panisson, A., Bajardi, P., & Pedreschi, D. (2021). FairLens: Auditing black-box clinical decision support systems. *Information Processing & Management*, *58*(5), 102657. https://doi.org/10.1016/j.ipm.2021.102657

Power, M. (1994). *The audit explosion*. *Paper: Vol. 7*. Demos.

Raji, I. D., & Buolamwini, J. (2019). Actionable Auditing. In V. Conitzer, G. Hadfield, & S. Vallor (Eds.), *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society* (pp. 429–435). ACM. https://doi.org/10.1145/3306618.3314244

Raji, I. D., Smart, A., White, R. N., Mitchell, M., Gebru, T., Hutchinson, B., Smith-Loud, J., Theron, D., & Barnes, P. (2020). Closing the AI accountability gap: Defining an end-to-end framework for internal algorithmic auditing. *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 33–44. https://doi.org/10.1145/3351095.3372873

Riazy, S., Simbeck, K., & Schreck, V. (2021). Systematic Literature Review of Fairness in Learning Analytics and Application of Insights in a Case Study. In H. C. Lane, S. Zvacek, & J. Uhomoibhi (Eds.), *Communications in Computer and Information Science. Computer Supported Education* (Vol. 1473, pp. 430–449). Springer International Publishing. https://doi.org/10.1007/978-3-030-86439-2_22

Sandvig, C., Hamilton, K., Karahalios, K., & Langbort, C. (2014). Auditing algorithms: Research methods for detecting discrimination on internet platforms. *Data and Discrimination: Converting Critical Concerns into Productive Inquiry*(22), 4349–4357.

Siemens, G., & Long, P. (2011). Penetrating the Fog: Analytics in Learning and Education. *EDUCAUSE Review*, *46*(5), 30.

Tagharobi, H., & Simbeck, K. (2021). Studying Algorithmic Fairness in Moodle Learning Analytics Using Code Analysis. *In ECEL 2021 20th European Conference on e-Learning (p. 444). Academic Conferences International limited.*

VDE-AR-E 2842-61. (2021). *VDE-AR-E 2842-61-3: Development and trustworthiness of autonomous/cognitive systems.*

Wilson, C., Ghosh, A., Jiang, S., Mislove, A., Baker, L., Szary, J., Trindel, K., & Polli, F. (2021). Building and Auditing Fair Algorithms: A Case Study in Candidate Screening. *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 666–677. https://doi.org/10.1145/3442188.3445928

Woodhouse, D. (2003). Quality Improvement through Quality Audit. *Quality in Higher Education*, *9*(2), 133–139. https://doi.org/10.1080/13538320308156

Yang, J., Ryu, D., & Baik, J. (2016, January 18–20). Improving vulnerability prediction accuracy with Secure Coding Standard violation measures. In *2016 International Conference on Big Data and Smart Computing (BigComp)* (pp. 115–122). IEEE. https://doi.org/10.1109/BIGCOMP.2016.7425809

Zhang, Y., Ghandour, A., & Shestak, V. (2020). Using Learning Analytics to Predict Students Performance in Moodle LMS. *International Journal of Emerging Technologies in Learning (IJET)*(15(20)), 102–115.