International Journal of Artificial Intelligence in Education
https://doi.org/10.1007/s40593-021-00248-0

ARTICLE

Educating Software and AI Stakeholders About Algorithmic Fairness, Accountability, Transparency and Ethics

Veronika Bogina[1] · Alan Hartman[1] · Tsvi Kuflik[1] · Avital Shulner-Tal[1]

Accepted: 16 March 2021Published online: 21 April 2021

Abstract

This paper discusses educating stakeholders of algorithmic systems (systems that apply Artificial Intelligence/Machine learning algorithms) in the areas of algorithmic fairness, accountability, transparency and ethics (FATE). We begin by establishing the need for such education and identifying the intended consumers of educational materials on the topic. We discuss the topics of greatest concern and in need of educational resources; we also survey the existing materials and past experiences in such education, noting the scarcity of suitable material on aspects of fairness in particular. We use an example of a college admission platform to illustrate our ideas. We conclude with recommendations for further work in the area and report on the first steps taken towards achieving this goal in the framework of an academic graduate seminar course, a graduate summer school, an embedded lecture in a software engineering course, and a workshop for high school teachers.

Keywords Fairness · Accountability · Transparency · Education · Algorithmic literacy

All authors contributed equally to this work.

✉ Veronika Bogina
sveron@gmail.com

Alan Hartman
alan.hartman.gm@gmail.com

Tsvi Kuflik
tsvikak@is.haifa.ac.il

Avital Shulner-Tal
avitalshulner@gmail.com

[1] University of Haifa, Haifa, Israel

Introduction

Our society is challenged by the increasing proliferation of opaque algorithmic systems, that apply black box machine learning algorithms, making use of large volumes of data to facilitate decision making in a wide variety of sensitive applications. We can see this phenomena in every aspect of our lives, including education (see for instance a work about estimating school value-added scores (Levy et al. 2190; Prinsloo 2020) and a recent survey about the application of machine learning in education (Kučak et al. 2018)). These systems are based on complex machine learning algorithms that are trained over large amounts of data in order to provide their prediction or classification of new cases. These systems are considered black boxes since it is difficult to follow their reasoning process and consequently, it is unclear how results were achieved and decisions were taken. This, in turn, brings with it the risks of discrimination and unfairness in such systems. In some cases they were found to unintentionally discriminate against members of their target audience. Some well-known examples include the COMPASS system where Brennan et al. (2009) found that black defendants were far more likely than white defendants to be incorrectly judged as at a higher risk of recidivism, while white defendants were more likely than black defendants to be incorrectly flagged as low risk. Another example is the German credit card study (Pedreschi et al. 2009) which demonstrated that the use of an applicant's demographic and socio-economic profiles in assessing the risk of a loan gave rise to unjustified discrimination against a class of applicants.

This has led to a growing body of research into algorithmic transparency, fairness, and the ethics of algorithmic decision making systems, recommender systems (and other software systems intended to modify the behavior of its users) and surveillance systems (see for instance the recent review of Favaretto et al. (2019)). However, the knowledge, produced by this research, is not always available to those involved in producing and consuming such systems.

There is also an increasing community and government concern about the effects and impact on society of algorithmic systems that may treat users unfairly. This includes calls for the software industry to assume responsibility for the impact of their products (Raji et al. 2020) as well as calls to adopt *interpretable* rather than black box models (Rudin 2019). The research and professional community has started to react accordingly (https://www.acm.org/binaries/content/assets/public-policy/2017_usacm_statement_algorithms.pdf).

The software industry currently produces systems, based on consumer requirements, their specifications and compliance with legal regulations. If the consumers do not have the knowledge to request transparency and fairness, or if the regulations do not mandate these attributes, they will almost certainly be absent – since there is a cost involved in satisfying such requirements. On the other hand, if consumers demand, or the law imposes transparency and fairness requirements on the systems it uses, the software suppliers may not have the knowledge and or capabilities to satisfy such requirements.

If the software industry (including owners and regulators) – as opposed to the academic community – is to respond to such concerns, there is a need for the transfer of academic knowledge to engineers and managers in the industry, so that they can take

informed decisions and manage risks effectively. We also see the need to educate the consumers of software systems – both those who are clients of the software industry, purchasing software to assist in their business activities, and those who use the software as clients of the businesses and other agencies that deploy software systems.

The goal of this paper is to analyze and understand the educational needs of the various stakeholders in the software industry in the fields of algorithmic transparency and fairness. We further study the available educational resources and educational experiences in order to determine the gaps in filling those needs, and make recommendations for the activities and studies required to fill those gaps.

Education specifically about artificial intelligence is the subject of growing attention, but in this paper we return to the *roots* - to software engineering (SE) education. We look at the whole picture, aiming to address all relevant stakeholders and processes in the software life cycle - including the system design, development, usage and maintenance.

The rest of the paper is organized as follows. We start with presenting a brief survey of related work in the field and gaps in "Related Work and Gaps". In "The Stakeholders in Algorithmic Systems" we introduce the primary stakeholders under consideration, and in "Practitioners' Education" we discuss ways to educate software development practitioners. In "Adapting Practitioner Education for Other Stakeholders" we extend the ideas to other stakeholders. In "Examples of FAT Education" we describe and evaluate some educational activities that the authors carried out. Finally, in "Discussion, Conclusions and Future Work" we conclude our ideas and discuss future directions for research and practice. We use the running example of a college admission platform to illustrate our ideas.

Related Work and Gaps

Educating software engineers (as well as owners and regulators) involves all aspects of the software development process, from requirements elicitation through to deployment and maintenance. Software engineers are expected to master a variety of techniques applied in the different development activities. Still, education on fairness, accountability and transparency is largely absent from the SE literature. When considering major professional guidelines for educating developers of information systems, like the SWEBOK (Bourque and Fairley 2014) or the AIS global education report (AIS Global IS Education Report 2018), as well as classical text books (Sommerville 2015), ethics is discussed at a very high level. Moreover, professional associations like the IEEE computer society and the ACM offer a code of ethics ("Be fair and take action not to discriminate" (https://www.acm.org/code-of-ethics)). However, in general, the discussion of ethics is very abstract, while fairness and transparency are hardly mentioned, and no practical suggestions are made. The need for considering FATE as part of SE engineers education is indeed new and we aim at making a first step towards bridging the gap.

When we consider fairness, transparency accountability and ethics, accountability can be tied to ethics (responsibility), while fairness and transparency can be seen through the lens of non-functional requirements (NFR) (Horkoff 2019). Hence this

section consists of a brief survey of NFR research, software ethics and developers education literature. This enables us to better identify the gaps in the SE education and suggest how to bridge them.

Fairness and Transparency in SE and RE Education

Teaching functional and non-functional requirements engineering is a part of most SE courses (Sommerville 2015). Acknowledging the importance of requirements engineering (RE), numerous studies focus on methods for teaching/training/improving the RE skills of practitioners. Morales-Ramirez and Alva-Martinez (2018) suggest a training plan for improving practitioners' RE analysis skills. They defined an RE cube with required skills for RE to be taught via an online platform that was developed in Moodle in order to satisfy two main constraints: limited time and remote access. Students are required to read the theoretical material and then take quizzes.

Maiden et al. (2010) considered RE as a creative problem-solving process. Their proposal is based on the 6 stages of the problem solving process (Nuseibeh and Easterbrook 2000): (1) objective finding, (2) fact finding (related to the goals surfaced in the previous stage), (3) problem finding (different ways to frame the problem), (4) idea finding, (5) solution finding and (6) acceptance finding (considering real-world and implementation issues). They also suggest various techniques for problem solving, where exploratory techniques (snowballing[1], traditional brainstorming, sticking dots[2] and story writing) seem to be the most suitable for eliciting non-functional requirements.

Lorca et al. (2018) used motivational modelling for a "light weight" requirements elicitation and modelling, preserving team agility. The idea is to capture all stakeholders requirements and define a high-level abstraction of goal models. They proposed a "do/be/feel" model to teach requirements elicitation in workshops. "Do" relates to functional requirements, "be" - to non-functional and "feel" to perceived emotions that developers want to engender in the users. They also focus on "who" to define all relevant stakeholders. During their one-day workshop for each category they define a list of elements and discuss them, aiming to reach a group consensus. The output of the workshop is a full hierarchical goal model, including all stakeholders. An interesting aspect of their technique is that they associate each quality goal (NFRs) with a functional one. The workshop is given to first year master's students after some theoretical lectures on RE.

We observe that current SE and RE text books and research aim to educate SE stakeholders in general and RE stakeholders in particular, but do not focus on FATE-related topics, which are becoming increasingly important as black box algorithmic systems impinge on every aspect of our daily lives.

The importance of including FATE-related aspects in training professionals is now widely recognized, as we can see in the ACM statement (https://www.acm.org/binaries/content/assets/public-policy/2017_usacm_statement_algorithms.pdf).

[1] https://www.mycoted.com/Snowball_Technique

[2] https://www.mycoted.com/Sticking_Dots

Numerous studies motivated the need to consider fairness and transparency of algorithmic systems. Horkoff (2019) outlined challenges in NFRs for Machine Learning related systems. She claims that the types of NFRs we are concerned with undergo a shift. NFRs like fairness and transparency become prominent, whereas other NFRs such as modularity, maintainability, interoperability usability and others may become less relevant. Hence, the meanings and interpretations of NFRs in an ML context need to be rethought.

In another study, Liao et al. (2020), the authors note the importance and interest in explainability to make AI algorithms understandable. They provide insights into user needs for explainability, suggest how these needs should be understood, prioritized and addressed, and present an extended question bank to support the requirements specification work to create user-centered explainable AI applications.

In summary, while algorithmic systems are widely used and the need for training their developers in relevant NFRs is acknowledged, there is still a gap in educating developers about addressing the fairness and transparency of such systems.

Ethics in SE and RE Education

Most of the papers on ethics in engineering refer to the responsibility, and hence accountability of engineers for their products. Herkert (2005) distinguishes two levels of ethics in engineering: microethics and macroethics. The former takes into consideration individual and internal relations between engineers, while the latter the social responsibility and "societal decisions about technology". Gotterbarn (2002) claimed that SE focused on the technical adequacy of the software product but was also responsible for the decision making of such products that affected other people's lives. Good SE should consider not only the technical aspects of the problem, but also the ethical issues raised by the outcomes and their impact.

In an earlier work on engineering ethics, Gotterbarn (1999) presented a pyramid with different levels of professional obligations (see Fig. 1). Each professional

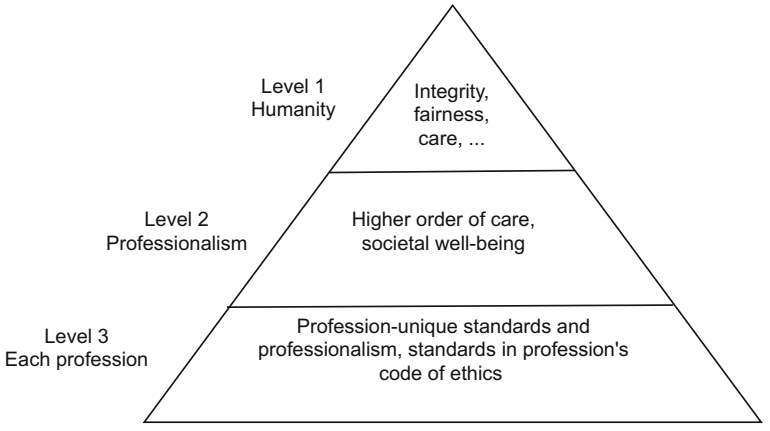

Fig. 1 The cumulative levels of professional obligation (Gotterbarn 1999)

should operate under explicit ethical standards. At the first level, there are basic human standards, like fairness, integrity and care, while other levels are based on these values. The lower the level, the more challenging the obligations. Professionals are beholden to a higher order of care than those that are influenced by their actions. The most challenging level is dependent on the nature of the profession. Software professionals should preserve software ethics (like those defined by the ACM (https://www.acm.org/binaries/content/assets/public-policy/2017_usacm_statement_algorithms.pdf), for example) and be responsible for adequate testing, debugging and code documentation.

Li and Fu (2012) performed a literature review on engineering ethics education and defined three main delivery models for teaching ethics. The first is an embedded approach, when teachers introduce and discuss ethics as an integrated part of their individual courses. The second is a special course taught by professors from different disciplines that contribute their experience and point of view. The last is a standalone course taught by a single faculty member. The advantages and disadvantages of these approaches are summarized in Table 1.

Eun-Kyung et al. suggested the use of team-based learning to teach ethics in medicine: interactive small groups learning with an "expert" instructor (Chung et al. 2009). Such an approach is more effective and enjoyable for the students. Their research showed that as a group, performance was better than as individual students (Ozgonul and Alimoglu 2019).

This paper is concerned with the ethical education of stakeholders in AI systems rather than the ethics of decisions made by educational AI-based platforms (which are considered in Jobin et al. (2019), Latham and Goltz (2019), and Marcinkowski et al. (2020)).

Garrett et al. (2020) introduced two ways of teaching ethics in AI: in a standalone course or by integrating ethics into technical courses. Burton et al. (2015) proposed to teach ethics as a standalone course using science fiction. In this scenario, people tend to look at the situation from a third person's perspective, sometimes identifying with him, facilitating the development of a moral imagination. Furey and Martin (2019) suggested incorporating ethical modules into AI courses by first introducing them to the Trolley Problem (Thomson 1985) and then coming to Utilitarianism.[3] Grosz et al. (2019) suggested embedding ethics education into the computer science curriculum, so students can learn to think not only about how to develop algorithms/systems, but also whether such algorithms/systems should be created at all.

In summary, unlike fairness and transparency, the importance of ethics in information systems development in general and algorithmic systems in particular is acknowledged, and there exist strategies for integrating it into the education of developers. Our contribution to education for ethics of AI systems is in extending the current strategies with a focus on fairness and transparency.

[3] https://www.britannica.com/topic/utilitarianism-philosophy

Table 1 Delivery models for teaching engineering ethics, their characteristics, pros and cons (Li and Fu 2012)

Type	Characteristics	Advantages	Disadvantages
Teaching ethics across the curriculum (embedded approach)	Engineering/science faculty members introduce ethics into their teaching. Ethics materials become normal components of the course	Increases the capability/ confidence of faculty members to address ethics discussions in courses. Provides materials for faculty to incorporate ethics into courses	Highly dependent on faculty members' willingness. No set standards in how to integrate in teaching and how to grade
Joint venture model/team- teaching approach	One course taught by a team of professors from multiple disciplines	Diversity of input from a team of faculty members	Dependent on the availability of motivated and qualified faculty
Standalone course	One independent course taught by one professor for students to sign up individually	The content is concentrated and can cover a variety of topics in one course. Easy to manage what should be covered in the course	Not part of the engineering curriculum. May be considered as unnecessary by both students and their advisors

Industry Practitioners Education

Most of the research in this area relates to practitioners who are experienced in developing *machine learning systems* (Holstein et al. 2019). Holstein et al. (2019) discusses the struggle that ML practitioners have with data collection, claiming that training data is the place where they can intervene to reduce bias. They also considered the application of de-biasing and auditing methods — a need to share guidelines and processes. Raji et al. (2020) suggested a framework for managing the auditing process in an industrial setting to ensure responsible development in AI.

A key factor in educating for transparency is the training of developers to include explainability features in AI systems. Liao et al. (2020) provide design guidelines for such systems. They define four categories of explanation: *the model explanation*, *prediction explanation*, *counterfactual inspection* and *example based explanation*. They suggest methods for dealing with each category, considering the potential user question: how, why, why not and what if.

Despite recent advances in SE and AI education, we believe that there still exists a gap in the education of software stakeholders about transparency and fairness. This applies in particular to software that uses machine learning and deep learning.

The Stakeholders in Algorithmic Systems

We distinguish three classes of primary stakeholders: *software practitioners*, including engineers, architects and managers in software development organizations, *regulators* who influence the system design and implementation by imposing regulatory constraints and *users* that interact with the system (Kilbertus et al. 2018). We further divide the set of users into the subclass of *system owners* – who operate the systems and control the design and usage processes, and *system subjects* – the end users who are most influenced by the outputs of the system.

We analyze the educational needs of each of the stakeholder groups on an operational level by considering which problems these users need to solve. The first of these problems is that of awareness. Each of the stakeholder groups needs to have an educational module focused on awareness of the existence and implications of opaque algorithmic systems that may be biased.

Once awareness has been achieved, the specific problems faced by the stakeholder groups differ due to their role in the creation and usage of algorithmic systems. Practitioners, regulators and owners have key roles in the design, implementation and deployment of such systems. Owners and regulators need the skills to demand and test for fairness and transparency, while practitioners need the skills to implement these requirements and to take actions to discover and prevent bias should it occur in the development process or after deployment. End users need skills to detect bias and interpret documents explaining the transparency features of the systems (Eiband et al. 2018).

In the educational context educators represent two types of users - those that educate others about FATE issues and those that use such systems (e.g. systems that

predict students success in a course (Ciolacu et al. 2018) or college admissions systems (Marcinkowski et al. 2020)). The stakeholders in a college admission platform (developed and maintained by *software practitioners* include: the *owner* i.e. the college management; two types of *users*: students that are influenced by the system's decision and faculty members who use such platforms for the classification and acceptance decision. The *regulator* is the country/state government authority responsible for making and enforcing laws governing AI-based platforms who should be aware of potential biases and discrimination.

This leads us to the following classification of educational modules and their targets:

- **FATE awareness** for all stakeholders;
- **FATE requirements specification** for owners and regulators who need to produce such specifications, and practitioners who need to understand and implement them;
- **Discrimination and unfairness discovery** – for both detection and testing for all stakeholders – including end users;
- **Discrimination and unfairness prevention** tools for practitioners;
- **Transparency promotion** for practitioners to produce transparency evidence and for all other stakeholders to consume and interpret such evidence.

The contents of these education modules vary according to the stakeholders' technical abilities and operational needs. Moreover, such modules can be seen as building blocks, when the basic level is awareness and all other modules are built upon it.

Practitioners' Education

Software practitioners are the primary audience for educational endeavors since they require education in all five of the areas identified in the previous section. For this reason, we focus on them and later give an account of the variations on these modules for other stakeholders. However, as ethics is addressed already in SE education, we focus specifically on FAT aspects.

Raising Awareness

Baeza-Yates (2018) noted the need to raise awareness of biases on the web and their effects. He stated that "Any remedy for bias starts with awareness of its existence."

Awareness related educational material could be structured to provide information on the following questions:

1. What are the risks of developing unfair opaque algorithmic systems?
2. What are the responsibilities of practitioners for mitigating such risks?
3. What might be the possible sources of bias and discrimination?

An introduction to the issue of fairness can be done in an engaging way using the "Fairness Toolkit" produced by the Unbiased project.[4] A tutorial could then proceed with case studies addressing the first question and serve as an introduction for a discussion of question 2 and question 3. The tutorial could conclude with a review of existing tools and methods which extend standard software engineering tools to FAT issues and include a discussion of new tools dedicated solely to FAT.

For a more in-depth introduction, the tutorial could become a series of lectures as part of a course or a dedicated seminar, where the participants take an active part in searching for case studies, analysing them and presenting them in class by leading a discussion. This activity may be better suited to graduate students.

Hands-on experience is a proven technique for the learning of practical skills (Hein 1991) and demonstrating the risks and challenges of unintentional discrimination. This can be achieved by hands-on tasks using biased data sets/algorithms (like, for instance, the German credit card data set (Newman et al. 1998) or a hand crafted one), where the participants are required to perform a task, analyze the results, identify biases in the results and identify the reasons for these biases.

FAT-driven Requirements Specification

Fairness, accountability and transparency requirement specifications can be derived using the techniques applied to other NFRs like security, privacy, interoperability or availability. The requirements may also be made functional by describing metrics for the attributes and setting appropriate levels of achievement required by the system being developed. Nuseibeh and Easterbrook (2000) claimed that the goal of software RE is to provide the system behaviour metrics to the relevant stakeholders. They defined 5 different stages in software systems RE: (1) eliciting requirements (2) modelling and analysing requirements (3) communicating requirements (4) agreeing about requirements and (5) evolving requirements.

Following this paradigm, in FAT-driven requirements specification we aim to answer the following questions:

1. How can practitioners frame a problem statement that considers transparency and fairness constraints?
2. How can practitioners model transparency and fairness as NFRs?
3. How can practitioners validate fairness and transparency requirements?
4. What additional disciplines should practitioners learn in order to sharpen their requirements writing and analysis skills? (Logic, cognitive psychology, linguistics, etc.)

Techniques for answering such questions are techniques that are used in RE in general (brainstorming, card sorting etc.). They can be embedded into SE courses, workshops and seminars. We also believe that students need to acquire basic knowledge in disciplines like logic, cognitive psychology, linguistics, law and social

[4] https://unbias.wp.horizon.ac.uk/fairness-toolkit/

sciences. This can be achieved through a team teaching approach (Lorca et al. 2018), where a team of professors from different areas teach students during the semester.

Discrimination and Unfairness Detection

After specifying FAT related requirements, there is a need to test the training data and the algorithm to discover deviation from the requirements, e.g. discrimination or unfairness that may emerge (such as gender bias, race, age and so on). A method for detecting such biases was suggested by Pedreschi et al. (2009) and shown in Fig. 2. This method for discrimination discovery is built on a set of rules derived from the historical outputs of a decision support system for providing credit for potential applicants. A set of classification and association rules, as well as frequently applied rules are extracted from the training data. The model outputs a set of discriminatory patterns that can unveil the contexts of the groups' discrimination.

There are also other tools and techniques that may be used, including traditional SE techniques. For example, source code documentation - to make code transparent to other team members, including the testing team, Code review - focusing on FAT requirements, Testing - where test cases will be specifically designed (both white box and black box) for discrimination discovery and finally Auditing - by an external agency with a mandate for ensuring fairness and transparency. Note that, even if discrimination is not revealed by testing, this is not a guarantee of its absence.

Table 2 lists a set of available discrimination discovery tools. The use of these tools can be implemented in the flow of ML testing as shown in Fig. 3 (Zhang et al. 2020). Instruction in the use of these tools and workflows is also best achieved through hands-on experience and exercises.

Discrimination and Unfairness Prevention

If the discrimination and/or unfairness towards a certain group or individual is detected, there is a need to remedy the situation. According to Bellamy et al. (2018), there are three stages where bias or unfairness may be introduced when training an

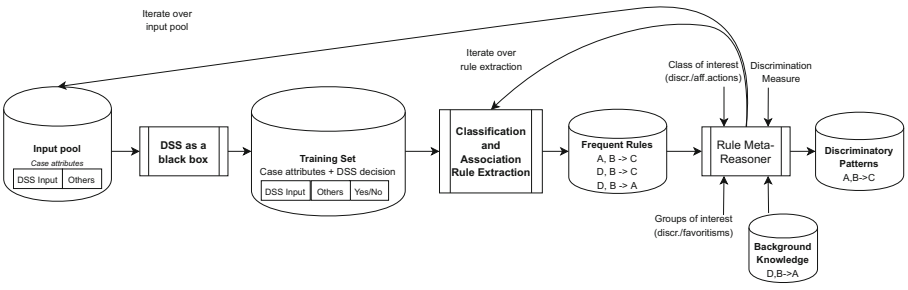

Fig. 2 Iterative discrimination discovery model for providing credit to applicants (Pedreschi et al. 2009). The input to the model is the history of decisions taken by the system. Subsequently, the classification, association and frequently used rules are extracted. The output of the model is a set of discriminatory patterns for a particular group of applicants

Table 2 Tools for discrimination discovery and bias mitigation

Tools	Description
Fairness Measures[a]	A framework for testing an algorithm on a variety of data sets and fairness metrics
Fairness Comparison[b]	An extensible test-bed for facilitating direct comparisons of algorithms, based on fairness metrics
Themis-ML[c]	An open source machine learning library that implements several fairness-aware methods
FairML[d]	A python toolbox for auditing machine learning models for bias
Aequitas[e]	An open source bias audit toolkit to audit machine learning models for discrimination and bias
Fairtest[f]	A tool for discovering and testing for suspicious associations between an algorithm's outputs and protected populations
Audit-AI[g]	A Python library that implements fairness-aware machine learning algorithms
AI Fairness 360[h]	IBM's open source toolkit for discovering discrimination and bias in ML
What-if tool[i]	Google's interactive visual interface for probing the models better

[a] https://www.fairness-measures.org/

[b] https://github.com/algofairness/fairness-comparison

[c] https://themis-ml.readthedocs.io/en/latest/

[d] https://github.com/adebayoj/fairml

[e] https://dsapp.uchicago.edu/projects/aequitas/

[f] https://github.com/columbia/fairtest

[g] https://github.com/pymetrics/audit-ai

[h] https://aif360.mybluemix.net/

[i] https://pair-code.github.io/what-if-tool/

ML algorithm: (1) *pre-processing*, (2) *in-processing*, and (3) *post-processing*. The first stage may introduce bias in its selection or use of training data, the second stage is vulnerable to algorithmic biases and the third stage provides an opportunity to correct bias before presenting the results. Examples and descriptions of different techniques used in these stages are given in Table 3. Since all three processes correspond to different stages of the training of the system, the education process could be built accordingly. First, the practitioners should be aware of these techniques, then they should ask themselves three main questions:

1. Whether a prevention technique can be applied to the data and if so, which one is suitable?
2. Whether a prevention technique can be applied to the algorithmic system and if so, which one is suitable?
3. Whether a prevention can be applied as a post processing step, and if so, which technique is most suitable?

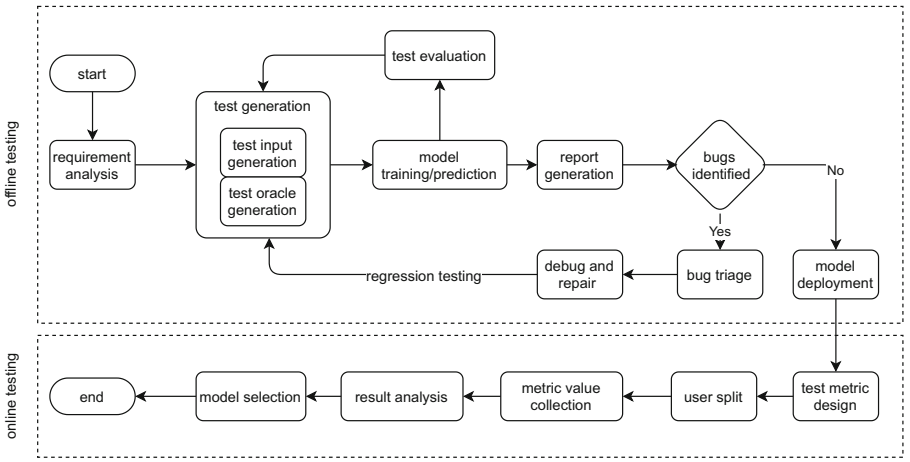

Fig. 3 Flow of ML Testing (Zhang et al. 2020). The flow considers both offline testing (for testing the model with historical data) and online testing (for complementing the shortage of the offline testing). This model was adapted from a traditional software testing workflow

The most appropriate techniques for teaching the methods for detection and prevention as presented in "Discrimination and Unfairness Detection" and "Discrimination and Unfairness Prevention" are hands-on practices. After theoretical explanation, practitioners should get a problematic data set (where unfairness and discrimination persist), analyze it and detect the source of discrimination, find the issue and find a way to handle it. Such practices can be embedded into the existing machine learning courses or stand-alone workshops can be held.

Transparency Promotion

Software transparency is not a new concept in SE. In the past, the need for software transparency was to minimize the risk of opaque software having unadvertised features that may pose a variety of threats to organizations and end users (Meunier 2008). Tu (2014) related to different stakeholders and defined transparency as the ability to get answers to all their questions using the information obtained about a system during its life cycle. In 2009 Cysneiros and Werneck (2009) claimed that "software transparency seems to be not only a remote possibility but something we will have to deliver sooner than many have thought", and in 2010 do Prado Leite and Cappelli (2010) argue that "transparency is a concern that information system designers must address as society demands more openness". Recently, in a Brazilian study of public opinion about software transparency, Portugal et al. (2017) defined software transparency as "the disclosure of what, how and why the software does what it does" and concluded that "in the future, society will demand transparency from software".

Thus the definition of software/system transparency has evolved beyond the code and its functions, to communication between different stakeholders and to the need to address the general public's expectations. Transparency is even more critical for implanted computerised devices (Sandler et al. 2010), based on machine learning

Table 3 Tools for discrimination and unfairness prevention

Stage	Technique	Description
pre	reweighting (Kamiran and Calders 2012)	Providing different weights for each pair of (group, label) of the training data.
pre	learning fair representation (Zemel et al. 2013)	Finding latent representation for data encoding, while hiding information regarding protected features.
pre	disparate impact remover (Feldman et al. 2015)	Editing feature values in order to increase group fairness while maintaining a ranking order within groups.
pre	data balancing (Dixon et al. 2018)	Preventing unintended bias by changing data set using different techniques, like oversampling or undersampling to balance data distribution with respect to protected features.
in	prejudice remover (Kamishima et al. 2012)	Adding a conscious discrimination term to learning objective.
in	adversarial debiasing (Zhang et al. 2018)	Reducing adversaries' abilities in finding the protected attributes.
post	equalized odds (Hardt et al. 2016)	Finding probabilities for changing output labels (by solving linear program) and equalized odds optimization.
post	calibrated equalized odds (Pleiss et al. 2017)	Finding probabilities for changing output labels and calibrated equalized odds optimization.
post	reject option classification (Kamiran et al. 2012)	Balancing between providing positive outcome to an unprivileged group and negative outcome to a privileged group under uncertainty.

and big data. do Prado Leite and Cappelli (2010) claimed that software transparency should be treated as an NFR - the quality of an operation of the system, rather than a behavioral one. Once transparency is defined as an NFR, this should guarantee that it will be considered throughout the system/software development cycle. Transparency can be transformed into a functional property by defining and measuring different levels of transparency achievement.

Camp (2006) noted that the appearance of open source code provides a way of disclosing information and Sandler et al. (2010) claimed that open source software is more secure than proprietary software. He called on regulators to force the manufacturers of life-critical devices to make their code publicly available.

For these reasons training for transparency promotion should be integrated in the education of software practitioners as part of their academic/professional studies. It needs to be part of SE courses, requirements engineering and system analysis courses,

software design and development courses, software testing and quality assurance courses etc. Transparency can be achieved by adopting ideas for "white box" software development and testing — making the code visible and transparent to other developers, to minimize development risks when reusing existing components. For end users, transparency can be achieved by generating explanations as part of the reasoning process of the system thus enhancing users' trust in the system and its outcomes.

In software development training, case studies and exercises using opaque and biased systems/data sets may be used to introduce the issue and motivate the need. Further transparency education exercises should involve creating explanations as part of the reasoning logic or in parallel. Developers of "black box" machine learning algorithms also need to be educated on the use of white box machine learning techniques like rule induction for creating explanations (Gilpin et al. 2018; Guidotti et al. 2018; Pedreschi et al. 2018; Samek et al. 2017).

Agile methods can also be helpful in making both a process and code more transparent, when all people involved in the development process communicate using a variety of agile techniques (scrum, extreme programming).

An Example of the College Admission Platform

To conclude this section we analyze an example of an algorithmic system that is used by a college admission board. Based on training data from previous years, the system decides whether or not to accept a student's application based on a predefined set of student attributes and college admission requirements. The obvious risks are that talented people that have the potential to succeed at this college may be rejected, while poor candidates may be accepted.

When using such a system, users should be aware of the following issues: (1) such systems can reproduce biases that exist in the training data set; (2) its decisions could be (unintentionally) biased towards specific groups (gender, race, demographics, parents income and so on - if such parameters are included in the data without careful examination). The discrimination should be discovered during system testing, before it reaches the customer. For example, Computer Science departments have historically had low admission rates for women. Therefore an admissions algorithm based solely on historical data would probably suffer from gender bias and require the application of balancing techniques before applying ML and AI methods. Moreover, after making a decision, the system should analyze the reasoning process and generate an explanation of the reasoning process, and an analysis of the overall fairness of the system. All these factors should be analyzed as part of the RE process and become NFRs: a need for transparency of an outcome.

The responsibilities of the different stakeholders involved in the development and use of such systems differ. In this section we presented the practitioners' point of view. In the next section we will introduce the points of view of other stakeholders.

Adapting Practitioner Education for Other Stakeholders

The use of hands-on exercises is less appropriate for other stakeholders who will rarely have developer skills. However, they all must be aware of the potential risks of discrimination in algorithmic systems and the need to prevent it as well as to detect and remove it.

Regulators, *Users* and *Owners* may be introduced to the challenges and risks in algorithmic systems by a "guided tour" through use cases - well known examples where systems clearly discriminate against users, this is a first step in raising awareness. Users then need to learn scepticism, not to take systems results/recommendations for granted, as they may be intentionally or unintentionally biased (intentionally - by commercial interests for instance or fake news for political interests). Users need to be trained to be sceptical while educators need to be trained to instill scepticism in their students.

Regulators also need to be familiar with existing tools that enable them to verify the fairness of the system - the tools, the metrics they produce and their meaning. This requires hands on training with specific tools, in order to be able to operate them (or similar ones as they become available) independently.

When considering the college admission system, it is clear that owner-users (e.g. people that take decisions based on the system, as well as regulators) should experience using a biased system and its results, as a case study, while using discrimination discovery tools for that purpose - to get first hand experience of the potential risk and its remedy.

Examples of FAT Education

This section describes and evaluates some educational activities that were carried out by the authors of this paper for different stakeholders. These include an advanced dedicated course on FAT, an embedded lecture for BSc students in an existing software engineering course and in a summer school for graduates and a workshop for high school teachers (educators).

Advanced Course

A course modeled on the ideas in this paper was pilot tested at the University of Haifa. It contains the five issues identified in "The Stakeholders in Algorithmic

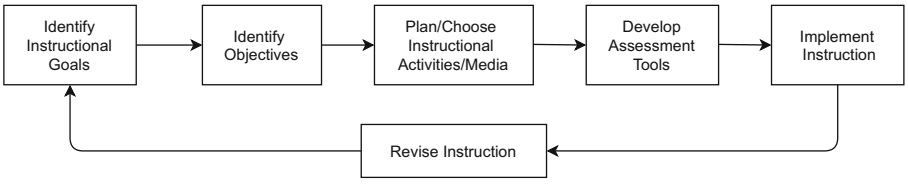

Fig. 4 Systematic instructional design model (Reiser and Dick 1995)

Systems": FAT awareness, FAT requirements, discrimination discovery, correcting discrimination and promoting transparency.

Our course was built according to the Reiser and Dick (1995) design model (Fig. 4) and incorporated both traditional classroom and flipped classroom techniques (Table 4).

In preparing the course we leveraged two recent examples of such courses:

- Course: Special Topics in Data Science: Responsible Data Science. New York University, Center for Data Science, Spring 2019[5]. This was a series of two hour lectures + one hour labs over 14 weeks, addressing awareness and fairness definition, tools for data preprocessing, anonymity and privacy, transparency, legal aspects and ethics, with a rich reading list.
- Course: Fairness, Accountability and Transparency in Machine Learning. Internet course, sponsored by the GIAN program of the Government of India[6]. This was a 9 day intensive course, covering basic machine learning concepts, case studies in the use of machine learning for decision-making (shedding light on potential biases), fairness mechanisms, accountability and interpretability. This course also comes with a rich reading list.

The authors conducted a course for graduates and upper level undergraduates focused on algorithmic transparency in order to pilot the work on developer education in the CyCAT project[7].

A seminar course (a combination of frontal lectures by the third author, invited talks by visiting lecturers and members of other departments and flipped classroom student presentations) on Algorithmic Transparency was given during the spring semester of 2019 at the University Haifa, Israel. Following the general architecture presented in (Tal et al. 2019) (see Fig 5), the third author gave a sequence of lectures comprising: (1) a survey of recent literature on bias in algorithmic systems, (2) the biases due to the input to such systems, (3) the biases due to the training data, (4) the potential bias due to human impact - both developers and operators/owner, (5) the need for ensuring fairness and (6) a review of possible solutions. During the classes, the importance of awareness of all stakeholders was emphasized leading to the need to address FATE aspects throughout the development of algorithmic systems from early stages (as described in "The Stakeholders in Algorithmic Systems" and further detailed in "Practitioners' Education"). Special attention was given to the initial stages of system development - considering FAT as essential non-functional requirements ("Raising Awareness") and then techniques for discrimination discovery ("FAT-driven Requirements Specification") and mitigation ("Discrimination and Unfairness Detection"). The invited lectures were given by experts from both academia (including legal and machine learning academics) and industry (specialists in the testing and certification of machine learning systems). In preparation for each class, the students were required to search for relevant papers and thus a repository

[5] https://dataresponsibly.github.io/courses/spring19/
[6] https://geomblog.github.io/fairness/
[7] http://www.cycat.io/

Table 4 Comparison of time schedule for traditional classroom and flipped classroom (Bergmann and Sams 2012)

Tradition classroom		Flipped classroom	
Activity	Time	Activity	Time
Warm-up activity	5 min.	Warm-up activity	5 min.
Go over previous night's homework	20 min.	Q&A time on video	10 min.
Lecture new content	30-45 min.	Guided and independent practice and/or lab activity	75 min.
Guided and independent practice and/or lab activity	20-35 min.		

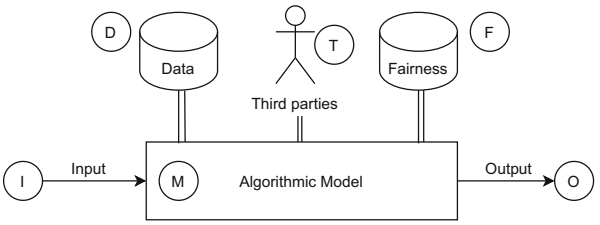

Fig. 5 A model of an Algorithmic System and its component that are relevant to fairness (Tal et al. 2019)

of relevant literature was collected. In the second part of the course the students presented the results of their literature review and submitted a written paper summarizing the topic.

The impact of the course on FAT awareness was evaluated using anonymous questionnaires. Eight students participated in the survey. Following the 7 point Likert scale we asked them to evaluate at what level they (1) gained knowledge about FAT topics; (2) got acquainted with the relevant FAT tools. Their response for (1) was 6.88/7 and for (2) 6.63/7. In addition, after the course (after grades were given), we asked students to share their thoughts about course in general and how it could be improved. We got the similar responses, claiming that FAT awareness was raised and that they will implement insights from this course in their research/work.

Embedded Lecture for Students

Another way to raise awareness is by embedding the relevant lecture in other software development courses and workshops. We did this in two instances: a Software Quality Assurance (SQA) course for B.Sc. students at the University of Haifa and at a Workshop on Data Science for graduate students (M.Sc. and Ph.D. students) from UNINOVE in Sao Paulo (UNINOVE).

The Software Quality Assurance course introduces principles, approaches and techniques in software quality assurance and their application throughout the information system life cycle. The lecture on FATE topics was given towards the end of the course, as an extension of the work on testing non-functional requirements. The lecture advocated embedding FATE driven development into several stages of software system development. Forty-one B.Sc. students from the University of Haifa, Israel participated in the course.

The UNINOVE Data Science Workshop was conducted by the University of Haifa, Israel for students of the Nove University (UNINOVE) which is a private higher education institution in San Paulo, Brazil. The program was intended for graduate students (M. Sc. And Ph. D.) in Informatics and Knowledge Management. A significant minority of the students also came from a more general Management program. The 5 day workshop exposed the students to scientific research in data science, with emphasis on the use of advanced tools and methods that are freely available for academic research. The lectures during the workshop introduced basic concepts in Machine Learning, Artificial Intelligence and Big Data followed by lectures on relevant problems and tools. The lectures on FATE topics were given towards the end of

the workshop, comprising one lecture on technical aspects of fairness and bias and a lecture on legal and ethical issues given by a member of the Law Faculty. Thirty-four M.Sc. and Ph.D. students from UNINOVE participated in the workshop.

The technical lectures that were given at the Software Quality Assurance course and at the UNINOVE workshop included an introduction to FATE and awareness-raising examples from real life.

In order to evaluate whether FATE awareness was raised during this lecture, we asked the students to answer a questionnaire built on the 5 point Likert scale. The students were asked about their level of knowledge before the lecture, and the resultant level of knowledge following the lecture. For question Q1: "To what extent were you aware of the topic before the lecture" the average answer of SQA students was 2.68 with STD of 1.23 and the average answer of UNINOVE students was 2.91 with STD of 1.50, while for question Q2: "How much new knowledge did you gain from the lecture?" – the average answer of SQA was 3.85 with STD of 0.94 and the average answer of UNINOVE students was 4.12 with STD of 0.84. The results suggest that the students' awareness of this subject was at an average level, and after the lecture, their (self-perceived) knowledge level increased significantly. We also asked students, (1) as future software developers and (2) as users, to what extent they would leverage this knowledge in systems development and as users of systems. SQA students replied to the first question with an average of 3.85 with STD of 0.85 and an average of 3.76 with STD of 0.83 to the second question and UNINOVE students replied to the first question with an average of 3.87 with STD of 0.98 and an average of 3.84 with STD of 0.94 to the second question.

A non-parametric Kruskal-Wallis test was performed in order to examine whether there are differences between the groups (SQA students and UNINOVE students) for Q1 and Q2, and no significant difference was found (H value 0.2345, p-value > 0.05 for Q1 and H value 1.4208, p-value > 0.05 for Q2). The lecture was given for two different populations in terms of geographical location, cultural differences and level of education and despite these differences the results show that there is an improvement in the (self-perceived) FATE awareness in the two groups. The distribution of the results for both SQA and UNINOVE students is presented in Fig. 6.

A group of 9 students at the UNINOVE summer workshop performed an additional experimental exercise, before the lecture was given and again at the end of the lecture, for evaluating their awareness of FATE in AI systems. The students received, as a case study, a description of an AI system in education for identifying gaps in the student's knowledge of mathematics and providing relevant courses for her/him in order to get the student back on track for college-level courses. Then the students were asked to answer the following questions:

Q1. Who are the relevant stakeholders?
Q2. What are the fairness constraints of the system?
Q3. What biases may exist in the system?
Q4. How can you improve the system to be fair?

We performed a qualitative analysis of the results, and observed that after the lecture the answers to the questions were more concrete and included consideration of additional factors than the answers to the same questions before the lecture — for such

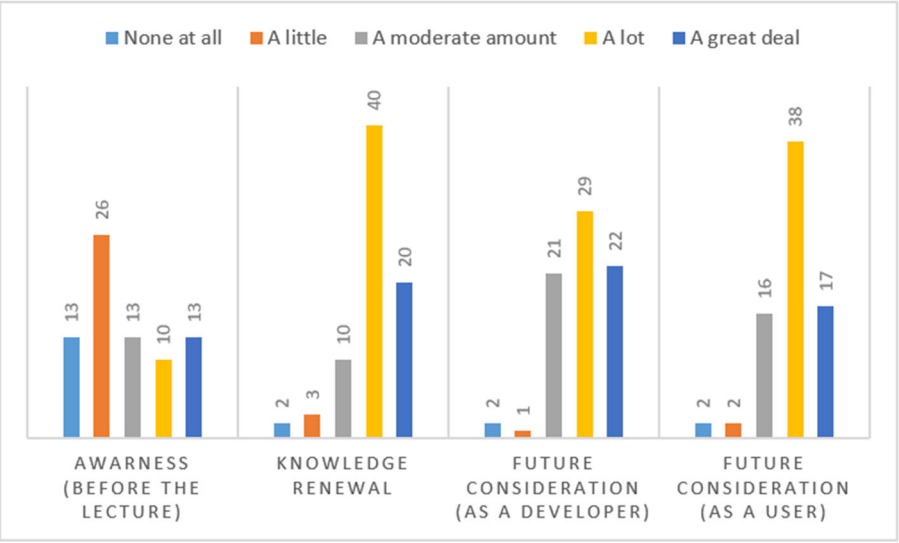

Fig. 6 The distributions of preliminary FATE Awareness level, knowledge renewal level and level of future FATE consideration as a developer and as a user

samples refer to Table 5. From these results we observe that there is an improvement in the understanding of this topic and that after the lecture, the students addressed the FATE aspects more thoroughly.

Workshop for Educators

A workshop on Algorithmic Literacy was conducted in Barcelona. It was attended by 20 teachers at primary, secondary and vocational schools. Many of the teachers taught STEM subjects, and the majority were from vocational schools whose students are aged 16-20.

The focus of the workshop was firstly on raising awareness through practical exercises, exploring the issues raised, and then on producing a lesson plan for the teachers' own students on a topic related to FAT that is relevant to the maturity level of their students and to the societal context. Awareness education — as discussed in "The Stakeholders in Algorithmic Systems" — is vital for all stakeholders, especially for educators who can pass on this knowledge to the general public through their students.

The workshop participants showed very high levels of engagement, including the production of their own lesson plans. For technically oriented students, one lesson plan involved experimentation with techniques for preserving privacy and awareness of targeted pricing on travel industry websites. Two groups produced lesson plans on raising student awareness of fake news and the filter bubble. Another group focused their lesson on the advertising presented in social media. The primary teachers focused their lesson plan on awareness of gender based bias.

Table 5 Samples of the answers on the same question before and after the lecture

Question	Sample answer before the lecture	Answer after the lecture by the same student
Who are the relevant stakeholders?	The most relevant stakeholders are the user and the programming teams.	When I first answered this question, I did not realize that the observer (regulator/auditor) could be a relevant stakeholder; I only mentioned the users and developer. It's interesting to know that there are additional and different kinds of users and observers.
What are the fairness constraints of the system?	I don't know.	Only student skill should be taken into consideration. Developers should not take individual sociological aspects into consideration.
What biases may exist in the system?	Incorrect calculation of the performance.	Different algorithmic biases, data biases and human biases.
How can you improve the system to be fair?	Review the code.	Train and test the system for fairness considerations.

Indicators of the effectiveness of the workshop are the length of time after the scheduled end of the workshop devoted to discussion and debate - 30-45 minutes; and the number of post workshop e-mail requests for supporting references - 5.

Discussion, Conclusions and Future Work

We have motivated the growing need for practical solutions for training stakeholders of algorithmic systems in FATE-related aspects. Then we presented a program for the structure of FATE education for a variety of stakeholders in the production and consumption of algorithmic systems. We focused on the educational needs of professionals who produce algorithmic systems, since their requirements encompass all aspects of FATE. Other stakeholders, consumers and regulators, require a subset of the skills required by practitioners, and often require different educational approaches and materials. Our experience in delivering a course in FAT for Information Systems students led us to conclude that essential elements of an educational program include a wide variety of disciplines, hence the need to involve instructors with a background in legal, social and ethical issues. We further noted that a primary requirement for effective education is a library of non-trivial examples of systems and data sets with a variety of fairness and transparency attributes to enable hands-on learning of the tools and techniques specific to FATE requirements specification, implementation, validation and communication. As part of the presentation, we emphasized the applicability to the education domain.

Our program for further research includes creating and maintaining a library of hands-on exercises, integrating FATE training into existing SE/AI courses, generating courses and seminars suitable for stakeholders with widely varying skill levels and piloting these courses, with students, faculty, teachers and the general public.

One of the main challenges for FATE-driven development is to embed FATE into the entire software development life cycle, where ideally, it should be part of each phase.

Acknowledgements This research has been partly supported by the CyCAT, which has received funding from the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 810105. The Barcelona workshop was supported by funding from the SeeRRI project under Grant Agreement No. 824588.

References

ACM code of Ethics [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed: 02-Mar-2021].

ACM statement on Algorithmic Transparency and accountability [Online]. Available: https://www.acm.org/binaries/content/assets/public-policy/2017_usacm_statement_algorithms.pdf. [Accessed: 02-Mar-2021].

AIS Global IS Education Report (2018). Shared on EDUglopedia.org by Association for Information Systems, AIS on June 14, 2018, EID: 6149.

Baeza-Yates, R. (2018). Bias on the Web. *Communications of the ACM*, *61*(6), 54–61.

Bellamy, R.K., Dey, K., Hind, M., Hoffman, S.C., Houde, S., Kannan, K., Nagar, S. (2018). Ai fairness 360:, An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. arXiv preprint arXiv:1810.01943.

Bergmann, J., & Sams, A. (2012). Flip your classroom: Reach every student in every class every day. International society for technology in education.

Bourque, P., & Fairley, R.E. (2014). Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0. IEEE Computer Society Press.

Brennan, T., Dieterich, W., Ehret, B. (2009). Evaluating the predictive validity of the COMPAS risk and needs assessment system. *Criminal Justice and Behavior*, *36*(1), 21–40.

Burton, E., Goldsmith, J., Mattei, N. (2015). Teaching AI ethics using science fiction. AAAI Workshop - Technical Report (2015), pp. 33–37.

Camp, L.J. (2006). Varieties of software and their implications for effective democratic government. In *Proceedings of the British academy*, (Vol. 135 pp. 183–185).

Ciolacu, M., Tehrani, A.F., Binder, L., Svasta, P.M. (2018). Education 4.0-Artificial Intelligence Assisted Higher Education: Early recognition System with Machine Learning to support Students' Success. In *2018 IEEE 24th International Symposium for Design and Technology in Electronic Packaging (SIITME)* (pp. 23–30): IEEE.

Cysneiros, L.M., & Werneck, V.M.B. (2009). An Initial Analysis on How Software Transparency and Trust Influence each other. In *WER*.

Dixon, L., Li, J., Sorensen, J., Thain, N., Vasserman, L. (2018). Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society* (pp. 67–73): ACM.

do Prado Leite, J.C.S., & Cappelli, C. (2010). Software transparency. *Business & Information Systems Engineering*, *2*(3), 127–139.

Eiband, M., Schneider, H., Bilandzic, M., Fazekas-Con, J., Haug, M., Hussmann, H. (2018). Bringing Transparency Design into Practice. In *23rd International Conference on Intelligent User Interfaces* (pp. 211–223): ACM.

Chung, E.-K., Rhee, J.-A.E., Baik, Y.-H., Oh-Sun, A. (2009). The effect of team-based learning in medical ethics education. *Medical Teacher*, *31*(11), 1013–1017. https://doi.org/10.3109/01421590802590553.

Favaretto, M., De Clercq, E., Elger, B.S. (2019). Big Data and discrimination: perils, promises and solutions. A systematic review. *Journal of Big Data*, *6*(1), 12.

Feldman, M., Friedler, S.A., Moeller, J., Scheidegger, C., Venkatasubramanian, S. (2015). Certifying and removing disparate impact. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 259–268). Sydney, Australia.

Furey, H., & Martin, F. (2019). AI education matters: a modular approach to AI ethics education. *AI Matters*, *4*(4), 13–15.

Garrett, N., Beard, N., Fiesler, C. (2020). More than "If Time Allows" the role of ethics in AI education. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society* (pp. 272–278).

Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L. (2018). Explaining Explanations: An Overview of Interpretability of Machine Learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)* (pp. 80–89): IEEE.

Gotterbarn, D. (1999). How the new software engineering code of ethics affects you. *IEEE Software*, *16*(6), 58–64.

Gotterbarn, D. (2002). Software engineering ethics. Encyclopedia of Software Engineering.

Grosz, B.J., Grant, D.G., Vredenburgh, K., Behrends, J., Hu, L., Simmons, A., Waldo, J. (2019). Embedded EthiCS:, integrating ethics across CS education. *Communications of the ACM*, *62*(8), 54–61.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys (CSUR)*, *51*(5), 1–42.

Hardt, M., Price, E., Srebro, N. (2016). Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, (Vol. 29 pp. 3315–3323). Barcelona, Spain.

Hein, G. (1991). Constructivist learning theory. Institute for Inquiry. Available at: http://www.exploratorium.edu/ifi/resources/constructivistlearning.htmlS.

Herkert, J.R. (2005). Ways of thinking about and teaching ethical problem solving: Microethics and macroethics in engineering. *Science and Engineering Ethics*, *11*(3), 373–385.

Holstein, K., Wortman Vaughan, J., Daumé, H. III., Dudik, M., Wallach, H. (2019). Improving fairness in machine learning systems: What do industry practitioners need? In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (pp. 1–16).

Horkoff, J. (2019). Non-functional requirements for machine learning: Challenges and new directions. In *2019 IEEE 27th International Requirements Engineering Conference (RE)* (pp. 386–391): IEEE.

Jobin, A., Ienca, M., Vayena, E. (2019). The global landscape of AI ethics guidelines. *Nature Machine Intelligence*, *1*(9), 389–399.

Kamiran, F., & Calders, T. (2012). Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, *33*(1), 1–33. https://doi.org/10.1007/s10115-011-0463-8.

Kamiran, F., Karim, A., Zhang, X. (2012). Decision theory for discrimination-aware classification. In *IEEE International Conference on Data Mining* (pp. 924–929). https://doi.org/10.1109/ICDM.2012.45.

Kamishima, T., Akaho, S., Asoh, H., Sakuma, J. (2012). Fairness-aware classifier with prejudice remover regularizer. Machine Learning and Knowledge Discovery in Databases, pp. 35–50.

Kilbertus, N., Gascón, A., Kusner, M.J., Veale, M., Gummadi, K.P., Weller, A. (2018). Blind justice:, fairness with encrypted sensitive attributes. arXiv preprint arXiv:1806.03281.

Kučak, D., Juričić, V., Dambić, G. (2018). Machine learning in education - a survey of current research trends. Annals of DAAAM & Proceedings, pp. 29.

Latham, A., & Goltz, S. (2019). A survey of the general public's views on the ethics of using AI in education. In *International Conference on Artificial Intelligence in Education* (pp. 194–206). Cham: Springer.

Levy, J., Mussack, D., Brunner, M., Keller, U., Cardoso-Leite, P., Fischbach, A. (2190). Contrasting classical and machine learning approaches in the estimation of value-added scores in large-scale educational data. Frontiers in Psychology, pp. 11.

Li, J., & Fu, S. (2012). A systematic approach to engineering ethics education. *Science and Engineering Ethics*, *18*(2), 339–349.

Liao, Q.V., Gruen, D., Miller, S. (2020). Questioning the AI: Informing design practices for explainable AI user experiences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (pp. 1–15).

Lorca, A.L., Burrows, R., Sterling, L. (2018). Teaching motivational models in agile requirements engineering. In *2018 IEEE 8th International Workshop on Requirements Engineering Education and Training (REET)* (pp. 30–39): IEEE.

Maiden, N., Jones, S., Karlsen, K., Neill, R., Zachos, K., Milne, A. (2010). Requirements engineering as creative problem solving: A research agenda for idea finding. In *2010 18th IEEE International Requirements Engineering Conference* (pp. 57–66): IEEE.

Marcinkowski, F., Kieslich, K., Starke, C., Lünich, M. (2020). Implications of AI (un-) fairness in higher education admissions: the effects of perceived AI (un-) fairness on exit, voice and organizational reputation. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (pp. 122–130).

Meunier, P. (2008). Software transparency and purity. *Communications of the ACM*, *51*(2), 104–104.

Morales-Ramirez, I., & Alva-Martinez, L.H. (2018). Requirements analysis skills: How to train practitioners? In *2018 IEEE 8th International Workshop on Requirements Engineering Education and Training (REET)* (pp. 24–29): IEEE.

Nuseibeh, B., & Easterbrook, S. (2000). Requirements engineering: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering* (pp. 35–46): ACM.

Newman, D., Hettich, S., Blake, C., Merz, C. (1998). UCI repository of machine learning databases. http://archive.ics.uci.edu/ml.

Ozgonul, L., & Alimoglu, M.K. (2019). Comparison of lecture and team-based learning in medical ethics education. *Nursing Ethics*, *26*(3), 903–913.

Pedreschi, D., Giannotti, F., Guidotti, R., Monreale, A., Pappalardo, L., Ruggieri, S., Turini, F. (2018). Open the Black Box Data-Driven Explanation of Black Box Decision Systems. arXiv preprint arXiv:1806.09936.

Pedreschi, D., Ruggieri, S., Turini, F. (2009). Integrating induction and deduction for finding evidence of discrimination . In *Proceedings of the 12th International Conference on Artificial Intelligence and Law* (pp. 157-166): ACM.

Pleiss, G., Raghavan, M., Wu, F., Kleinberg, J., Weinberger, K.Q. (2017). On fairness and calibration. In *Advances in Neural Information Processing Systems* (pp. 5680–5689).

Portugal, R.L.Q., Engiel, P., Roque, H., do Prado Leite, J.C.S. (2017). Is there a demand of software transparency? In *Proceedings of the 31st Brazilian Symposium on Software Engineering* (pp. 204–213): ACM.

Prinsloo, P. (2020). Of 'black boxes' and algorithmic decision-making in (higher) education–A commentary. *Big Data & Society*, *7*(1), 2053951720933994.

Raji, I.D., Smart, A., White, R.N., Mitchell, M., Gebru, T., Hutchinson, B., Barnes, P. (2020). Closing the AI accountability gap: defining an end-to-end framework for internal algorithmic auditing. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (pp. 33–44).

Reiser, R.A., & Dick, W. (1995). *Instructional planning: A guide for teachers*, 2nd edn. Allyn and Bacon: Boston.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, *1*(5), 206–215.

Samek, W., Wiegand, T., Müller, K.R. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. arXiv preprint arXiv:1708.08296.

Sandler, K., Ohrstrom, L., Moy, L. (2010). Andamp; McVay, R Killed by code: Software transparency in implantable medical devices. Software Freedom Law Center, pp. 308–319.

Sommerville, I. (2015). Software Engineering. 10th. In *Book Software Engineering. 10th, Series Software Engineering*: Addison-Wesley.

Tal, A.S., Batsuren, K., Bogina, V., Giunchiglia, F., Hartman, A., Loizou, S.K., Otterbacher, J. (2019). "End to End" towards a framework for reducing biases and promoting transparency of algorithmic systems. In *2019 14th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP)* (pp. 1–6): IEEE.

Thomson, J.J. (1985). The trolley problem. *The Yale Law Journal*, *94*(6), 1395–1415.

Tu, Y.C. (2014). *Transparency in Software Engineering*. ResearchSpace@ Auckland): Doctoral dissertation.

Zemel, R., Wu, Y.L., Swersky, K., Pitassi, T., Dwork, C. (2013). Learning fair representations.

Zhang, B.H., Lemoine, B., Mitchell, M. (2018). Mitigating unwanted biases with adversarial learning.

Zhang, J.M., Harman, M., Ma, L., Liu, Y. (2020). Machine learning testing: Survey, landscapes and horizons. IEEE Transactions on Software Engineering.