Building Better Open-Source Tools to Support Fairness in Automated Scoring

Nitin Madnani\(^1\), Anastassia Loukina\(^1\), Alina von Davier\(^2\), Jill Burstein\(^1\) and Aoife Cahill\(^1\)

\(^1\)Educational Testing Service, Princeton, NJ
\(^2\)ACT, Iowa City, IA
\(^1\)\{nmadnani,aloukina,jburstein,acahill\}@ets.org
\(^2\)Alina.vonDavier@act.org

Abstract

Automated scoring of written and spoken responses is an NLP application that can significantly impact lives especially when deployed as part of high-stakes tests such as the GRE® and the TOEFL®. Ethical considerations require that automated scoring algorithms treat all test-takers fairly. The educational measurement community has done significant research on fairness in assessments and automated scoring systems must incorporate their recommendations. The best way to do that is by making available automated, non-proprietary tools to NLP researchers that directly incorporate these recommendations and generate the analyses needed to help identify and resolve biases in their scoring systems. In this paper, we attempt to provide such a solution.

1 Introduction

Natural Language Processing (NLP) applications now form a large part of our everyday lives. As researchers who build such applications, we have a responsibility to ensure that we prioritize the ideas of fairness and transparency and not just blindly pursue better algorithmic performance.

In this paper, we discuss the ethical considerations pertaining to automated scoring of written or spoken test responses, referred to as “constructed responses”. Automated scoring is an NLP application which aims to automatically predict a score for such responses. We focus on automated systems designed to score open-ended constructed response questions. Such systems generally use text and speech processing techniques to extract a set of features from responses which are then combined into a scoring model to predict the final score assigned by a human rater (Page, 1966; Burstein et al., 1998; Zechner et al., 2009; Bernstein et al., 2010).

Test scores whether assigned by human raters or computers can have a significant effect on people’s lives and, therefore, must be fair to all test takers. Automated scoring systems may offer some advantages over human raters, e.g., higher score consistency (Williamson et al., 2012). Yet, like any other machine learning algorithm, models used for score prediction may inadvertently encode discrimination into their decisions due to biases or other imperfections in the training data, spurious correlations, and other factors (Romei and Ruggieri, 2013b; von Davier, 2016).\(^1\)

The paper has the following structure. We first draw awareness to the psychometric research and recommendations on quantifying potential biases in automated scoring and how it relates to the ideas of fairness, accountability, and transparency in machine learning (FATML). The second half of the paper presents an open-source tool called RSMTool\(^2\) for developers of automated scoring models which directly integrates these psychometric recommendations. Since such developers are likely to be NLP or machine learning researchers, the tool provides an important bridge from the educational measurement side to the NLP side. Next, we discuss further challenges related to fairness in automated scoring that are not currently addressed by RSMTool as well as methods for avoiding bias in automated scoring rather than just detecting it. The paper concludes with a discussion of how these tools and methodologies may, in fact, be ap-

\(^1\)Some of these problems were recently discussed at a panel focused on Fairness in Machine learning in Educational Measurement that was held at the annual meeting of National Council for Educational Measurement (von Davier and Burstein, 2016).

\(^2\)http://github.com/EducationalTestingService/rsmtool
plicable to other NLP applications beyond automated scoring.

2 Ethics and Fairness in Constructed Response Scoring

At this point in the paper, we believe it is important to define exactly what we refer to as fairness for the field of scoring constructed responses, whether it is done manually by humans or automatically by NLP systems.

A key concept here is the idea of a “construct” which is defined as a set of related knowledge, skills, and other abilities that a test is designed to measure. Examples of possible constructs include logical reasoning, language proficiency, reading comprehension etc. A fair test is one where differences in test scores between the test-takers are due only to differences in skills which are part of the construct. Any consistent differences in scores between different groups of test-takers that result from other factors not immediately related to the construct (i.e., “construct-irrelevant”) — e.g., test-taker gender — may indicate that the test is unfair. Specifically, for a test to be fair, the non-random effects of construct-irrelevant factors need to be minimized during the four major phases of a test: test development, test administration, test scoring, and score interpretation (Xi, 2010; Zieky, 2016):

1. Test development. All tests must be free of bias, i.e., no questions on a test should include any content that may advantage or disadvantage any specific subgroup of test-takers in ways that are unrelated to the construct the test is designed to assess. The subgroups in this case are defined based on factors that include test-taker personal information such as gender, race, or disability, but may also go beyond the standard protected properties. For example, Xi (2010) discusses how familiarity with the subject matter in an English language proficiency test may impact test performance and, thus, would require an explicit analysis of fairness for a group defined by test-taker fields of study. Additionally, on the same test, test-takers whose native languages use the Roman alphabet will have an advantage over test-takers with native languages based on other alphabets. However, this advantage is allowable because it is relevant to the construct of English comprehension. To ensure bias-free questions, the developers of the test conduct both qualitative and quantitative reviews of each question (Angoff, 2012; Duong and von Davier, 2013; Oliveri and von Davier, 2016; Zieky, 2016).

2. Test administration. All test-takers must be provided with comparable opportunities to demonstrate the abilities being measured by the test. This includes considerations such as the location and number of test centers across the world, and whether the testing conditions in each test center are standardized and secure. For example, Bridgeman et al. (2003) showed that, at least for some tests, examinee test scores may be affected by screen resolution of the monitors used to administer the test. This means that for such tests to be fair, it is necessary to ensure that all test-takers use monitors with a similar configuration.

3. Test scoring. There should also be no bias in the test scores irrespective of whether they are produced by human raters or by automated scoring models. The unequal distribution of social, economic, and educational resources means that some differences in performance across subgroups are to be expected. However, differences large enough to have practical consequences must be investigated to ensure that they are not caused by construct-irrelevant factors (AERA, 1999).

4. Score interpretation. Finally, while most tests tend to have a constant structure, the actual questions change regularly. Sometimes several different versions of a test (“test forms”) exist in parallel. Even if two test-takers take different versions of a test, their test scores should still be comparable. To achieve this, a separate statistical process process called “test equating” is often used to adjust for unintended differences in the difficulty of the test forms (Lee and von Davier, 2013; Liu and Dorans, 2016). This process itself must also be investigated for fairness to ensure that it does not introduce bias against any group of test-takers.

In this paper, we focus on the third phase: the fairness of test scores as measured by the impact of construct-irrelevant factors. As Xi (2010) discusses in detail, unfair decisions based on scores assigned to test-takers from oft-disadvantaged
groups are likely to have profound consequences: they may be denied career opportunities and access to resources that they deserve. Therefore, it is important to ensure — among other things — that construct-irrelevant factors do not introduce systematic biases in test scores, irrespective of whether they are produced by human raters or by an automated scoring system.

Over the last few years, there has been a significant amount of work done on ensuring fairness, accountability, and transparency for machine learned models from what is now referred to as the FATML community (Kamiran and Calders, 2009; Kamishima et al., 2012; Luong et al., 2011; Zemel et al., 2013). More recently, Friedler et al. (2016) proposed a formal framework for conceptualizing the idea of fairness. Within that framework, the authors define the idea of “structural bias”: the unequal treatment of subgroups when there is no clear mapping between the features that are easily observable for those subgroups (e.g., largely irrelevant, culturally and historically defined characteristics) and the true features on which algorithmic decisions should actually be based (the “construct”). Our conceptualization of fairness for automated scoring models in this paper — avoiding systematic biases in test scores across subgroups due to construct-irrelevant factors — fits perfectly in this framework.

3 Detecting Biases in Automated Scoring

Human scoring of constructed responses is a subjective process. Among the factors that can impact the assigned scores are rater fatigue (Ling et al., 2014), differences between novice and experienced raters (Davis, 2015), and the effect of raters’ linguistic background on their evaluation of the language skill being measured (Carey et al., 2011). Furthermore, the same response can sometimes receive different scores from different raters. To guard against such rater inconsistencies, responses for high-stakes tests are often scored by multiple raters (Wang and von Davier, 2014; Penfield, 2016). Automated scoring of constructed responses can overcome many of these issues inherent to human scoring: computers do not get tired, do not have personal biases, and can be configured to always assign the same score to a given response.

However, recent studies in machine learning have highlighted that algorithms often introduce their own biases (Feldman et al., 2015) either due to an existing bias in the training data or due to a minority group being inadequately represented in the training data. Automated scoring is certainly not immune to such biases and, in fact, several studies have documented differing performance of automated scoring models for test-takers with different native languages or with disabilities (Burstein and Chodorow, 1999; Bridgeman et al., 2012; Wang and von Davier, 2014; Wang et al., 2016; An et al., 2016; Loukina and Buzick, In print).

Biases can also arise because of techniques used to develop new features for automated scoring models. The automated score may be based on features which are construct-irrelevant despite being highly correlated with the human scores in the training data. As an example, consider that more proficient writers tend to write longer responses. Therefore, one almost always observes a consistent positive correlation between essay length and human proficiency score (Perelman, 2014; Sherman, 2014b). This is acceptable since verbal fluency — a correlate of response length — is considered an important part of the writing proficiency. Yet, longer essays should not automatically receive higher scores. Therefore, without proper model validation to consider the relative impact of such features, decisions might be made that are unfair to test-takers.

On this basis, the psychometric guidelines require that if automated scoring models are to be used for making high-stakes decisions for college admissions or employment, the NLP researchers developing those models should perform model validation to ensure that demographic and construct-irrelevant factors are not causing their models to produce significant differences in scores across different subgroups of test-takers (Yang et al., 2002; Clauser et al., 2002; Williamson et al., 2012). This is exactly what fairness — as we define it in this paper — purports to measure.

However, it is not easy for an NLP or machine learning researcher to perform comprehensive model validation since they may be unfamiliar with the required psychometric and statistical checks. The solution we propose is a tool that incorporates both the standard machine learning pipeline necessary for building an automated scoring model and a set of psychometric and statistical analyses aimed at detecting possible bias in
engine performance. We believe that such a tool should be open-source and non-proprietary so that the automated scoring community can not only audit the source code of the already available analyses to ensure their compliance with fairness standards but also contribute new analyses.

We describe the design of such a tool in the rest of the paper. Specifically, our tool provides the following model validation functionality to NLP/ML researchers working on automated scoring: (a) defining custom subgroups and examining differences in the performance of the automated scoring model across these groups; (b) examining the effect of construct-irrelevant factors on automated scores; and (c) comparing the effects of such factors in two different versions of the same scoring model, e.g., a version with a new feature added to the model and a version without the same feature.

4 RSMTool

In this section, we present an open-source Python tool called RSMTool developed by two of the authors for building and evaluating automated scoring models. The tool is intended for NLP researchers who have already extracted features from the responses and need to choose a learner function and evaluate the performance as well as the fairness of the entire scoring pipeline (the training data, the features, and the learner function).

Once the responses have been represented as a set of features, automated scoring essentially becomes a machine learning problem and NLP researchers are free to use any of the large number of existing machine learning toolkits. However, most of those toolkits are general-purpose and do not provide the aforementioned fairness analyses. Instead, we leverage one such toolkit — scikit-learn (Pedregosa et al., 2011) — to build a tool that integrates these fairness analyses directly into the machine learning pipeline and researchers then get them automatically in the form of a comprehensive HTML report.

Note that the automated scoring pipeline built into the tool provides functionality for each step of the process of building and evaluating automated scoring models: (a) feature transformation, (b) manual and automatic feature selection, and (c) access to linear and non-linear learners from scikit-learn as well as the custom linear learners we have implemented. In this paper, we will focus solely on the fairness-driven evaluation capabilities of the tool that are directly relevant to the issues we have discussed so far. Readers interested in other parts of the RSMTool are referred to the comprehensive documentation available at http://rsmtool.readthedocs.org.

Before we describe the fairness analyses implemented in the tool, we want to acknowledge that there are many different ways in which researchers might approach building as well as evaluating scoring models (Chen and He, 2013; Shermis, 2014a). The list of learners and fairness analyses the tool provides is not, and cannot be, exhaustive. In fact, later in the paper, we discuss some analyses that could be implemented in future versions of the tool since one of the core characteristics of the tool is its flexible architecture. See §4.4 for more details.

In the next section, we present in detail the analyses incorporated into RSMTool aimed at detecting the various sources of biases we introduced earlier. As it is easier to show the analyses in the context of an actual example, we use data from the Hewlett Foundation Automated Student Assessment Prize (ASAP) competition on automated essay scoring (Shermis, 2014a). As our scoring model, we use ordinary linear regression with features extracted from the text of the essay; see Attali and Burstein (2006) for details of the features. Note that since the original ASAP data does not contain any demographic information, we simulate an L1 attribute (the test-taker’s native language) for illustration purposes. The complete report automatically generated by RSMTool is available at: http://bit.ly/fair-tool. The report contains links to the raw data used to generate it and to other input files needed to run RSMTool. We focus on specific sections of the report below.

4.1 Differential Feature Functioning

In order to evaluate the fairness of a machine learning algorithm, Feldman et al. (2015) recommend preventive auditing of the training data to determine if the resulting decisions will be fair, irrespective of the machine learning model learned from that training data. RSMTool incorporates sev-

---

3https://www.kaggle.com/c/asap-aes/data/

4We believe it is more transparent to use a publicly available dataset with simulated demographics, rather than a proprietary dataset with real demographics that cannot be shared publicly. The value added by the fairness analyses comes through in either case.
eral such auditing approaches borrowed from previous research in both educational measurement and machine learning.

The first step in evaluating the fairness of an automated scoring model is to ensure that the performance of each feature is not primarily determined by construct-irrelevant factors. The traditional way to approach this is to have an expert review the features and ensure that their description and method of computation are in line with the definition of the specific set of skills that the given test purports to measure (Deane, 2013). However, features incorporated into a modern automated scoring system often rely on multiple underlying NLP components such as part-of-speech taggers and syntactic parsers as well as complex computational algorithms and, therefore, a qualitative review may not be sufficient. Furthermore, some aspects of spoken or written text can only be measured indirectly given the current state of NLP technologies (Somasundaran et al., 2014).

RSMTool allows the user to explore the quantitative effect of two types of construct-irrelevant factors that may affect feature performance: categorical and continuous.

4.1 Categorical Factors

This group of factors generally includes variables that can take on one of a fixed number of possible values, e.g., test-takers’ demographic characteristics, different versions of the same test question, or various testing conditions. We refer to these factors as “subgroups” though they are not always limited to demographic subgroups.

When this information is available for all or some of the responses, RSMTool allows the user to compare the feature distributions for different subgroups using box-plots and other distributional statistics such as mean and standard deviations. However, feature distributions depend on the scores which may differ across subgroups and, therefore, differences in a feature’s distribution across subgroups may not always indicate that the feature is biased. To address this, RSMTool also includes Differential feature functioning (DFF) analysis (Penfield, 2016; Zhang et al., In print). This approach compares the mean values of a given feature for test-takers with the same score but belonging to different subgroups. These differences can be described and reviewed directly using DFF line plots. Figure 1(a) shows a box-plot for the distribution of the GRAMMAR feature by test-taker L1 subgroups in our sample dataset; Figure 1(b) shows a DFF line plot for the same feature. These plots indicate that the values for the GRAMMAR feature are consistently lower for one of the test-taker subgroups (L1=Hindi) across all score levels. If such a pattern were observed in real data, it would warrant further investigation to establish the reasons for such behavior.

4.1.2 Continuous Factors

This type of construct-irrelevant factors includes continuous covariates which despite being correlated with human scores are either not directly relevant to the construct measured by the test or, even if they are, should not be the primary contributor to the model’s predictions. Response length, as previously discussed, is an example of such covariates. Even though it provides an important indication of verbal fluency, a model which predominantly relies on length will not generate fair scores. To explore the impact of such factors, RSMTool computes two types of correlations: (a) the marginal correlation between each feature and the covariate, and (b) the “partial” correlation between each feature and the human score, with the effects of the covariate removed (Cramér, 1947). This helps to clearly bring out the contribution of a feature above and beyond being a proxy for the identified covariate. The marginal and partial correlation coefficients for our example are shown in Figure 1(c). It shows that although all features in our simulated dataset contribute information beyond response length, for some features, length accounts for a substantial part of their performance.

4.2 Bias in Model Performance

Not all types of machine learning algorithms lend themselves easily to the differential feature functioning analysis. Furthermore, the sheer number of features in some models may make the results of such analyses difficult to interpret. Therefore, a second set of fairness analyses included into RSMTool considers how well the automated scores agree with the human scores (or another, user-specified gold standard criterion) and whether this agreement is consistent across different groups of test-takers.

RSMTool computes all the standard evaluation metrics generally used for regression-based machine learning models such as Pearson’s correlation coefficient (r), coefficient of determination (R²), coefficient of determination adjusted (R²_adj), area under the receiver operating characteristic curve (AUC), macro-averaged F1-score, and root mean squared error (RMSE).
(a) Box-plots showing the distribution of standardized GRAMMAR feature values by test-taker native language (L1). The dotted red lines represent the thresholds for outlier truncation computed as the mean feature value ± 4 standard deviations.

(b) A differential feature functioning (DFF) plot for the GRAMMAR feature. Each line represents an L1; each point shows the mean and 95% confidence intervals of the feature values computed for test-takers with that L1 and that assigned score.

(c) Pearson’s correlation coefficients ($r$) between features and human scores: (a) **Marginal**: marginal correlation of each feature with human score (b) **Partial – all**: correlation of each feature with human score with the effects of all other features removed, and (c) **Partial – length**: the correlation of each feature with human score with the effect of response length removed. The two dotted lines represent correlations thresholds recommended by Williamson et al. (2012).

Figure 1: Examples of RSMTool fairness analyses for categorical and continuous factors.
In addition, it also computes other measures that are specifically recommended in psychometric literature for evaluating automated scoring models: quadratically-weighted kappa, percentage agreement with human scores, and the standardized mean difference (SMD) between human and automated scores (Williamson et al., 2012; Ramirez and Williamson, 2013). These metrics are computed for the whole evaluation set as well as for each subgroup separately in order to evaluate whether the accuracy of automated scores is consistent across different groups of test-takers. Figure 2 shows a plot illustrating how the model $R^2$ computed on the evaluation set varies across the different test-taker L1 subgroups.

![Figure 2: The performance of our scoring model ($R^2$) for different subgroups of test-takers as defined by their native language (L1). Before computing the $R^2$, the predictions of the model are trimmed and then re-scaled to match the human score distribution in the training data.](image)

### 4.3 Model comparison

Like any other software, automated scoring systems are updated on a regular basis as researchers develop new features or identify better machine learning algorithms. Even in scenarios where new features or algorithms are not needed, changes in external dependencies used by the scoring pipeline might necessitate new releases. Automated scoring models may also be regularly re-trained to avoid population drift which can occur when the test-taker population used to train the model no longer matches the population currently evaluated by this model.

When updating an automated scoring system for one of the above reasons, one should not only conduct a fairness analysis for the new version of the model, but also a comprehensive comparison of the old and the new version. For example, a change in the percentage of existing test-takers who have passed a particular test resulting from the update would need to be explained not only to the test-takers but also to the people making decisions based on test scores (von Davier, 2016).

RSMTool includes the functionality to conduct a comprehensive comparison of two different versions of a scoring system and produce a report which includes fairness analyses for each of the versions as well as how these analyses differ between the two versions. As an example, we compare two versions of our example scoring model — one that uses all features and another that does not include the GRAMMAR feature. The comparison report can be seen here: [http://bit.ly/fair-tool-compare](http://bit.ly/fair-tool-compare).

### 4.4 Customizing RSMTool

The measurement guidelines currently implemented in RSMTool follow the psychometric framework suggested by Williamson et al. (2012). It was developed for the evaluation of e-rater, an automated system designed to score English writing proficiency (Attali and Burstein, 2006), but is generalizable to other applications of automated scoring. This framework was chosen because it offers a comprehensive set of criteria for both the accuracy as well as the fairness of the predicted scores. Note that not all of these recommendations are universally accepted by the automated scoring community. For example, Yannakoudakis and Cummins (2015) recently proposed a different set of metrics for evaluating the accuracy of automated scoring models.

Furthermore, the machine learning community has recently developed various analyses aimed at detecting bias in algorithm performance that could be applied in the context of automated scoring. For example, in addition to reviewing individual features, one could also attempt to predict the subgroup membership from the features used to score the responses (Feldman et al., 2015). If this
prediction is generally accurate, then there is a risk that subgroup membership could be implicitly used by the scoring model and lead to unfair scores. However, if the subgroup prediction has high error over all models generated from the features, then the scores assigned by a model trained on this data are likely to be fair.

RSMTool has been designed to make it easy for the user to add new evaluations and analyses of these types. The evaluation and report-generation components of RSMTool (including the fairness analyses) can be run on predictions from any external learner, not just the ones that are provided by the tool itself. Each section of its report is implemented as a separate Jupyter/IPython notebook (Kluyver et al., 2016). The user can choose which sections should be included into the final HTML report and in which order. Furthermore, NLP researchers who want to use different evaluation metrics or custom fairness analyses can provide them in the form of new Jupyter notebooks; these analyses are dynamically executed and incorporated into the final report along with the existing analyses or even in their place, if so desired, without modifying a single line of code.

Finally, for those who want to make more substantive changes, the tool is written entirely in Python, is open-source with an Apache 2.0 license, and has extensive online documentation. We also provide a well-documented API which allows users to integrate various components of RSMTool into their own applications.

4.5 Model Transparency & Interpretability

The analyses produced by RSMTool only suggest a potential bias and flag individual subgroups or features for further consideration. As we indicated earlier, the presence of differences across subgroups does not automatically imply that the model is unfair; further review is required to establish the source of such differences. One of the first steps in such a review usually involves examining each feature separately as well as the individual contribution of each feature to the final score. It is important to note here that unfairness may also be introduced by what is not in the model. An automated scoring system may not cover a particular aspect of the construct which can be evaluated by humans. If the performance across subgroups differs on this aspect of the construct, the difference may be due to “construct under-representation” rather than due to construct-irrelevant factors.

The automated scoring models used in systems such as e-rater for assessing writing proficiency in English (Attali and Burstein, 2006) or SpeechRater for spoken proficiency (Zechner et al., 2009) have traditionally been linear models with a small number of interpretable features because such models lend themselves more easily to a detailed fairness review and allow decision-makers to understand how, and to what extent, different parts of the test-takers’ skill set are being covered by the features in the model (Loukina et al., 2015). For such linear models, RSMTool displays a detailed model description including the model fit ($R^2$) computed on the training set as well as the contribution of each feature to the final score (via raw, standardized, and relative coefficients).

At the same time, recent studies (Heilman and Madnani, 2015; Madnani et al., 2016) on scoring actual content rather than just language proficiency suggest that it is possible to achieve higher performance, as measured by agreement with human raters, by employing many low-level features and more sophisticated machine learning algorithms such as support vector machines or random forests. Generally, these models are built using sparse feature types such as word $n$-grams, often resulting in hundreds of thousands of predominantly binary features. Using models with such a large feature space means that it is no longer clear how to map the individual features and their weights to various parts of the test-takers’ skill set, and, therefore, difficult to identify whether any differences in the model performance stem from the effects of construct-irrelevant factors.

One way to increase the interpretability of such models is to group multiple features by feature type (e.g. “syntactic relationships”) and build a stacked model (Wolpert, 1992) containing simpler models for each feature type. These stacked models can then be combined in a final linear model which can be examined in the usual manner for fairness considerations (Madnani and Cahill, 2016). The idea of making complex machine-learned models more interpretable to users and stakeholders has been investigated more thoroughly in recent years and several promising solutions have been proposed that could also be used for content-scoring models (Kim et al., 2016; Wilson et al., 2016).
5 Mitigating Bias in Automated Scoring

So far we have primarily discussed techniques for detecting potential biases in automated scoring models. We showed that there are multiple sources of possible bias which makes it unlikely that there would be a single “silver bullet” that can make test scores completely bias-free. The approach currently favored in the educational measurement community is to try and reduce susceptibility to construct-irrelevant factors by design. This includes an expert review of each feature before it is added to the model to ensure that it is theoretically and practically consistent with the skill set being measured by the test. These features are then combined in an easily interpretable model (usually linear regression) which is trained on a representative sample of test-taker population.

However, simpler scoring models may not always be the right solution. For one, as we discussed in §3, several studies have shown that even such simple models may still exhibit bias. In addition, recent studies on scoring test-takers’ knowledge of content rather than proficiency have shown that using more sophisticated — and hence, less transparent — models yields non-trivial gains in the accuracy of the predicted scores. Therefore, ensuring completely fair automated scoring at large requires more complex solutions.

The machine learning community has identified several broad approaches to deal with discrimination that could, in theory, be used for automated scoring models, especially those using more complex non-linear algorithms: the training data can be modified (Feldman et al., 2015; Kamiran and Calders, 2012; Hajian and Domingo-Ferrer, 2013; Mancuhan and Clifton, 2014), the algorithm itself can be changed so that it optimizes for fairness as well as the selection criteria (Kamishima et al., 2012; Zemel et al., 2013; Calders and Verwer, 2010), and the output decisions can be changed after-the-fact (Kamiran et al., 2012). A survey of such approaches is provided by Romei and Ruggeri (2013a). Future work in automated scoring could explore whether these methods can address some of the known biases.

Of course, it is also important to note that such bias-mitigating approaches often lead to a decline in the overall model performance and, therefore, one needs to balance model fairness and accuracy which likely depends on the stakes for which the model is going to be used.

6 Conclusion

In this paper, we discussed considerations that go into developing fairer automated scoring models for constructed responses. We also presented RSMTool, an open-source tool to help NLP researchers detect potential biases in their scoring models. We described the analyses currently incorporated into the tool for evaluating the impact of construct-irrelevant categorical and continuous factors. We also showed that the tool is designed in a flexible manner which allows users to easily add their own custom fairness analyses and showed some examples of such analyses.

While RSMTool has been designed for automated scoring research (some terminology in the tool and the report is specific to automated scoring), its flexible nature and well-documented API allow it to be easily adapted for any machine learning task in which the numeric prediction is generated by regressing on a set of non-sparse, numeric features. Furthermore, the evaluation component can be used separately which allows users to evaluate the performance and fairness of any model that generates numeric predictions.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments and suggestions. We would also like to thank Lei Chen, Sorelle Friedler, Brent Bridgeman, Vikram Ramanarayanan, and Keelan Evanini for their contributions and comments.

References


Toon Calders and Sicco Verwer. 2010. Three Naive Bayes Approaches for Discrimination-Free Classification. Data Mining journal; special issue with selected papers from ECML/PKDD.


Sara Hajian and Josep Domingo-Ferrer. 2013. A Methodology for Direct and Indirect Discrimination Prevention in Data Mining. IEEE Transactions on Knowledge and Data Engineering, 25(7):1445 – 1459.


Thomas Kluiver, Benjamin Ragan-Kelley, Fernando Prez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damin Avila,


