PRECLUDING DISCRIMINATORY BEHAVIOR IN INTELLIGENT TUTORING SYSTEMS

An Honors Thesis

Presented By:

Sarah Brockman

Completion Date:

December 2018

Approved By:

Philip Thomas, Chair

Yuriy Brun, Committee Member

ABSTRACT

Title: Precluding Discriminatory Behavior in Intelligent Tutoring Systems
Author: Sarah Brockman
Thesis/Project Type: Independent Honors Thesis
Approved By: Philip Thomas, Chair
Approved By: Yuriy Brun, Committee Member

As machine learning algorithms become more advanced and increasingly relied upon by the general public, it has become imperative that these algorithms do not discriminate against anyone, minorities in particular. However, this is a challenging issue. Despite the best efforts of their designers, algorithms can still exhibit unexpected behavior. The educational domain is one area that has very low tolerance for discrimination; equal education should be available for everyone. Discrimination in education can lead to unequal learning opportunities and decreased confidence. Intelligent Tutoring Systems (ITSs) are educational tools that employ machine learning techniques to help students learn more effectively. They are intended to supplement traditional instruction methods by showing additional tutorials to students that would be most effective for them. They also adapt to each student's learning style to learn which tutorial to show a certain student. We show that an ITS that employs standard reinforcement learning (RL) algorithms will discriminate against minorities, which in our particular case is female students. We also present a new RL algorithm that does not discriminate with high confidence. This algorithm will explore multiple options for which tutorial to deploy, and will run a "safety check" before finally deciding on a tutorial to ensure that its behavior is not discriminatory. A desirable feature of our algorithm is that if it does not find a solution that does not discriminate and does not predict that it can act without discrimination, it will return 'no solution found,' instead of returning a solution that has the potential to discriminate. This allows our proposed algorithm to rarely discriminate.

1 Introduction

This research project is intended to provide insight into discrimination in machine learning algorithms, namely reinforcement learning (RL), used in Intelligent Tutoring Systems (ITSs) (Ferguson et al., 2006). The goal is to implement a web-based ITS that optimizes its behavior with RL, and show that its behavior discriminates against a particular minority group, such as female students. We will analyze the results of this ITS and use the data to implement a non-discriminatory ITS that successfully teaches concepts and provides appropriate problems for the user. For this research, I have been working with my advisor, Professor Philip Thomas, and PhD student Blossom Metevier from the Autonomous Learning Laboratory, as well as Professor Yuriy Brun from the Laboratory for Advanced Software Engineering Research.

Artificial intelligence (AI) is becoming increasingly prevalent in our society; almost every field is impacted by it in some way. Education is one of these many fields. Many software systems used in education are becoming more advanced and some include forms of AI and machine learning algorithms. However, these technologies are not nearly advanced enough to replace human instruction completely and have a lot of potential for unwanted behavior.

One important example of unwanted behavior in AI is discrimination against minorities. COMPAS, the Correctional Offender Management Profiling for Alternative Sanctions, is an example of this. COMPAS is a computer system used for risk assessment in people who have been arrested for various crimes, and has been shown to estimate a much higher risk of repeat offense for black people than people of other races (Angwin et al., 2016). AI systems can also discriminate against other groups such as struggling students, different genders, certain ethnicities, and so on. All students should be entitled to an equal opportunity for education, unhindered by the discriminatory behavior in the computer systems meant to help them learn.

Most learning algorithms present in educational systems would try to maximize the amount students using the system learn. This could be measured by maximizing the score on some quiz or set of practice problems. This can come at a cost, however. If the algorithm's goal is to maximize the total score of all students, it may choose a method of teaching that increases the score dramatically for one group but decreases the score for another. If this happens, then one group would not be learning the material as effectively and is not getting an equal opportunity to learn.

Our research is intended to prevent this problem. We show that standard machine learning (ML) algorithms used in intelligent tutoring systems discriminate against a specific minority (Antonova et al., 2016). In our case, this minority is female students since female students are the minority in many disciplines, especially STEM. We will teach a new mathematical concept to men and women from the US of various ages and educational backgrounds. We will teach this concept in various ways using examples, equations, and different levels of detail. We will show that a standard reinforcement learning algorithm will choose to teach the "students" in a way that is extremely effective for men, but not as effective for women. This results in the men learning the material well and leaves the female students struggling.

A standard RL algorithm in the ITS would behave as follows: If students from both groups struggle with a certain approach, then the learning algorithm will choose another. If a certain tutorial helps the male students learn the material more easily, then the men will get much higher scores on a problem set, which indicates that tutorial is effective. However, this specific tutorial's format or content could lead to lower scores for female students. If the scores of the male students increase dramatically but the scores for female students decrease a slight amount, the learning algorithm will see it as a net increase for everyone and will adopt the tutorial that discriminates against women as the new way of teaching.

An algorithm used in education should rarely make such a choice that will help one group at the cost of harming another. We will design an RL algorithm that ensures with high probability this undesirable behavior will not occur; it will only change its way of teaching if it improves learning potential for both male and female students.

Our ITS is simple. We have a basic web-based tutorial where we introduce our math concept and employ different explanations and examples, depending on the type of tutorial. At the end, we have a quiz on the topic that we taught in the previous section to measure the effectiveness of the teaching style used in that tutorial. We will take the scores from all the tutorials and design an RL algorithm that changes its policy (which tutorial type to deploy) based on the scores. We will ensure this algorithm rarely changes its tutorial type if it decreases the expected score of someone from either group.

2 Significance

One of the primary goals of educators in today's world is to provide an equal opportunity for quality education for everyone. Intelligent tutoring systems (ITSs) can assist in this goal. Students who may not be able to attend in-person lectures or receive personalized, one-on-one help from their human instructors would benefit most from ITSs.

With ITSs, students are able to complete coursework, learn new material, and receive help online, whenever and wherever they are. Intelligent tutoring systems are mostly used to provide supplemental instruction and practice to complement human instruction. Students can watch videos, work through practice problems, and view step by step examples in many different subjects. ITSs can learn how particular students approach certain problems and alter the way material is presented to those students in order to maximize learning.

Students all learn in unique ways and require different explanations and presentations of material. Human instructors are able to provide this; if a student voices their confusion, an instructor can alter their explanations until the student grasps the concept. ITSs cannot handle this problem so easily. If a student is confused, an online learning system has no way of knowing the cause of confusion and cannot adapt. In certain scenarios, ITSs may favor advanced students and discriminate against students who are struggling. ITSs can also discriminate against minorities, as we show in our research.

Our research is intended to explore this discriminatory behavior in ITSs and devise a way to prevent it. If ITSs can teach all students the way that is most effective for them, then everyone can have an equal opportunity to learn material even if they come from different backgrounds and knowledge levels. For example, an ITS teaching physics concepts could show advanced students relevant equations and mathematical examples and proofs, and a less-advanced student realistic diagrams and analogies. Both students learn the same material, but in forms most effective for their unique learning style.

In our research we construct an ITS with unfair tutorials that is highly likely to discriminate when using standard RL algorithms, then show our new safe algorithm avoids discrimination even in this extreme scenario. We explore discrimination against female students when shown a confusing tutorial that teaches an incorrect concept. Since the female students are shown a tutorial that teaches them the wrong concept, they will perform poorly on the assessment quiz. However, the male students are not affected so this will not impact their score in any way. The AI algorithms in ITSs should ideally never choose examples and teaching methods that undermine the learning of minorities even though overall student success is increased. Thus, we devise an algorithm that rarely discriminates and chooses options that increase learning potential of all students, not just students from a particular group or background.

This research contributes to the overall field of AI safety. Work in this field is becoming more important as society's dependence on AI increases. AI has found its way into educational systems, which has led to an increased possibility for discrimination of certain students which leads to unequal educational opportunities. This is very unethical behavior and should never be tolerated. It is important to the field of AI in general that the public be able to rely on the technology created with it and trust that it will treat them fairly. If we want AI to help supplement human instruction, then it should at least treat the students the same way a human instructor would.

3 Background

3.1 Reinforcement Learning

Reinforcement learning is an area of machine learning and AI that is inspired by psychology and learning by experience. RL in computation is analogous to operant conditioning in behavioral psychology, where subjects learn preferential voluntary behaviors based on rewards and punishments. Artificial agents can be designed so they learn in the same way; an agent can learn which actions to take based on what it was rewarded or punished for in the past, as it will try to maximize its total reward. For example, an agent could learn to play chess by being awarded for winning games and given nothing for losing. We are formulating the learning task for our ITS as a problem that can be solved with reinforcement learning. The RL agent has possible *states* it could be in, a *policy* that defines what the agent should do based on which state is in, and a *reward function* that defines rewards for each state.

Our learning task is a specific type of problem called a bandit problem, named after the slot machine or one-armed bandit (Sutton and Barto, 1998). This is a specific type of problem where there are no states, only actions. In an n-armed bandit task, the agent has n different actions it can take (the most favorable given by the policy). Following each of these actions is a reward. If one of the actions is deemed better than the rest, it will have the highest reward. The agent will be faced with this problem multiple times, and after trying all the actions, it will learn which action to take based on the rewards it receives.

In our problem, the available actions are the different tutorial types (the different ways of presenting the material we are teaching). The rewards following each of the tutorials is a score achieved on the quiz by a student following that specific tutorial. We need to keep in mind, however, that we cannot just choose the greedy action. This is the action that has the highest reward, or the tutorial type with the highest average score. We need to choose a tutorial that benefits *everyone* learning the material, not just one group. This defines the skeleton of the RL agent that we will devise to learn, based on previous experience, which tutorial to deploy to new learners.

There are two main learning paradigms for reinforcement learning algorithms: batch learning and online learning. Batch learning uses "batches" of data, i.e., many trajectories of agent experience in the environment, to update its policy. The learning algorithm can perform many iterations over this batch of data. In an online learning setting, the agent updates its policy and value function after every interaction with the environment (i.e., every time it takes an action or observes a reward). In general, batch learning algorithms are more data efficient, but can face problems with the exploration-exploitation tradeoff. We use batch learning in our research.

The following sections describe the bandit algorithms we use as our "standard" reinforcement learning algorithms. Later, we show that these algorithms discriminate against female students when used as the learning algorithm in an ITS, since they all converge to the tutorial with the highest empirical mean (highest average reward).

3.1.1 Upper Confidence Bounds (UCB)

Upper confidence bound (UCB) is one of the simplest bandit algorithms. The main idea behind UCB is that the agent should choose the arm with the highest upper bound on its expected reward estimate. Initially, each arm is played once. Then, an upper bound is calculated for the expected reward of each arm, and the best arm is played (Kuleshov and Precup, 2014). In our case, a 'play' on an arm is a random selection of a score from the tutorial that arm represents. See Algorithm 1 for the full algorithm pseudocode.

Algorithm 1 Upper Confidence Bounds (UCB)

```
1: procedure UCB
2:     play each arm once to find initial estimates $\hat{\mu}$
3:     for each timestep $t = 1, 2, ...$ do
4:         pick arm $j(t)$ as follows: $j(t) = \underset{i=1,\ldots,k}{\arg\max} \left(\hat{\mu_i} + 0.5\sqrt{\frac{\ln t}{n_i}}\right)$
   ▷ where $\hat{\mu_i}$ denotes the empirical mean of rewards at arm $i$ and
   $n_i$ denotes the number of times arm $i$ has been played previously,
   for arms $i = 1, ..., k$
5:         play arm $j(t)$ and observe reward from that arm
6:         repeat
7:     end for
8: end procedure
```

3.1.2 Epsilon Greedy (ϵ-greedy)

Epsilon greedy is another simple bandit algorithm that we show discriminates. The basic idea is to greedily choose the arm with the highest empirical mean with probability $1 - \epsilon$, and choose another arm (all with uniform probability) with probability ϵ. Epsilon is usually a small value, around 0.1 for example. The reason we need to choose a random arm occasionally is to ensure we explore all options and do not get stuck constantly choosing a suboptimal arm; i.e., we need to balance the exploration-exploitation tradeoff (Kuleshov and Precup, 2014). The pseudocode for ϵ-greedy is shown in Algorithm 2.

Algorithm 2 Epsilon Greedy (ϵ-greedy)

```
1: procedure $\epsilon$-GREEDY
2:     play each arm once to find initial estimates $\hat{\mu}$
3:     for each timestep $t = 1, 2, ...$ do
4:         calculate probabilities for each arm according to

               $p_i(t+1) = \begin{cases} 1 - \epsilon + \epsilon/k & \text{if } i = \underset{j=1,\ldots,k}{\arg\max} \hat{\mu_j}(t) \\ \epsilon/k & \text{otherwise} \end{cases}$

   ▷ where $\hat{\mu_j}$ denotes the empirical mean observed at arm $j$ for
   arms $j = 1, ..., k$
5:         play one of the arms according to their probabilities
6:         record the reward for that arm
7:         repeat
8:     end for
9: end procedure
```

3.1.3 Boltzmann Exploration (Softmax)

The idea behind softmax methods is to pick arms with probabilities proportional to their empirical means. In other words, we give higher weights to arms where we have observed higher rewards. In this particular method, the probabilities for the arms are set according to a Boltzmann distribution. This method has a hyperparameter, or a parameter that needs to be set by the researcher prior to use. This parameter is the temperature, τ, which controls the randomness of the choice between arms. When τ is 0, the action selection is completely greedy, and the algorithm always chooses the arm with the highest empirical mean.

As τ tends toward infinity, the distribution over the arms becomes uniform (Kuleshov and Precup, 2014). The pseudocode for Boltzmann exploration is shown in Algorithm 3.

Algorithm 3 Boltzmann Exploration (Softmax)

```
1: procedure BOLTZMANN EXPLORATION
2:     play each arm once to find initial estimates $\hat{\mu}$
3:     for each timestep $t = 1, 2, ...$ do
4:         calculate probabilities for each arm according to
               $$p_i(t+1) = \frac{e^{\hat{\mu_i}(t)/\tau}}{\sum_{j=1}^{k} e^{\hat{\mu_j}(t)/\tau}}$$
           ▷ where $\hat{\mu_j}$ denotes the empirical mean observed at arm $j$ for
           arms $j = 1, ..., k$ and $\tau$ is the temperature parameter
5:         play one of the arms according to their probabilities
6:         record the reward for that arm
7:         repeat
8:     end for
9: end procedure
```

3.2 Importance Sampling and Safe RL

In our learning task, we want to compare multiple different policies to find the best one to use for teaching. To do this, we evaluate policies different than the one that was being followed when it generated data from the original surveys. This is called off-policy policy evaluation (Precup et al., 2000). We can see how effective different policies would be without ever following more than one. We will show how this can be achieved with importance sampling.

Importance sampling is a statistical technique used to estimate properties of a new distribution when all the researcher has is samples from another distribution. We can apply this technique to ensure we rarely deploy a policy that is harmful for any of the people taking our tutorials. Using importance sampling, we can find an estimate of the performance of a new policy for our RL agent without ever deploying the policy to actual people. This is beneficial because it allows us to avoid using a potentially detrimental policy on actual people (Thomas et al., 2015). Instead of having to fully deploy new policies to see if they are more effective, we can instead use importance sampling to evaluate these policies and see if they can improve upon the current one. We can find the importance-weighted expected return of a new policy we are trying to evaluate, denoted π_e, from

$$\mathrm{E}[R \mid \pi_e] = \frac{1}{n} \sum_{i=1}^{n} \frac{\pi_e(T_i)}{\pi_b(T_i)} R, \tag{1}$$

where n is the total number of samples (people taking the tutorials), $\pi_x(T_i)$ denotes the probability of the tutorial used in instance i occurring under policy π_x, and R denotes the return from that tutorial (the score on the quiz). In this case, π_b is the behavior policy, i.e., the original policy used when the tutorials were deployed to the actual participants. An importance-weighted return is usually calculated as a product of the above over an entire trajectory. A trajectory is a sequence of decisions made by the agent according to the policy, which continues until the agent reaches an ending or terminal state. However, our task is a bandit problem, so one trajectory is the agent taking just one action. In other words, each trajectory has length one. Pseudocode for calculating the importance-weighted return can be found in Algorithm 4.

Algorithm 4 Importance-Weighted Return

```
1: procedure IWR($\pi_e$, $D$)
2:     given an evaluation policy $\pi_e$ and a dataset $D$ with $|D|$ total rewards (scores):
3:     sum $\leftarrow 0$
4:     for each reward $R_i$ in $D$ do
           $$\text{sum} \mathrel{+}= \frac{\pi_e(T_i)}{(1/3)} \times R_i$$
           ▷ where (1/3) is the probability of each tutorial under our behavior policy (all tutorials with uniform probability), $T_i$ denotes the tutorial that produced specific reward $R_i$, and $\pi_e(T_i)$ denotes the probability of that tutorial occurring under the evaluation policy $\pi_e$
5:         repeat
6:     end for
7:     return $\frac{\text{sum}}{|D|}$
8: end procedure
```

This seems like an entirely safe way to evaluate policies before using them. However, evaluating an expected return for a new policy is not enough; we need to be confident that our calculated expected return for the new policy is accurate. In other words, we need to be highly confident the new policy is going to perform better before it is deployed. To do this, we could use a concentration inequality such as Hoeffding's inequality or Student's t-test. These techniques measure how much a variable's expected mean deviates from its sample mean. We can use this to obtain a bound on the probability that the expected return of the new policy is greater than the return following the old policy.

Hoeffding's inequality is the first potential concentration inequality. The general formula is below. With probability $1 - \delta$,

$$\mu \geq \frac{1}{n}\sum_{i=1}^{n} X_i - b\sqrt{\frac{\ln(\frac{1}{\delta})}{2n}}, \tag{2}$$

where X_i is the importance-weighted return for score i (from the right side of the sum in Equation (1)), δ is a defined significance level, and μ is the actual return from following the new policy (Thomas et al., 2015). However, it is important to note that this inequality depends on b, which is the upper bound on possible values of the importance-weighted return. If the probability of a tutorial under the behavior policy is low but it is high under the evaluation policy, then the importance-weighted return will be very high; many orders of magnitude above the actual expected score. This upper bound can be calculated by finding the π_e and π_b that maximize Equation (1). This high bound means Hoeffding's inequality will have very high variance. We will not use this concentration inequality for this reason; we can get a much tighter bound with other concentration inequalities.

Instead, we will be using Student's t-test to calculate a lower bound on the performance of the new evaluation policy. The general formula is below. With probability $1 - \delta$,

$$\mu \geq \frac{1}{n}\sum_{i=1}^{n} X_i - \frac{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2}}{\sqrt{n}} t_{1-\delta,n-1}, \tag{3}$$

where, same as above, X_i is the importance-weighted return for score i, $\overline{X}_n$ is the average importance-weighted return for scores $i = 1, ..., n$, δ is a defined significance level, and μ is the actual return from following the new policy. The term $t_{1-\delta,n-1}$ represents the $100(1-\delta)^{th}$ percentile of the Student's t distribution with

$n-1$ degrees of freedom (Thomas, 2015). Notice how this equation does not depend on an upper bound for the importance-weighted return. This means we can get a much tighter lower bound on the new policy's performance. Now that we have a concentration inequality, we can find a lower bound on the performance of the new policy and an upper bound on the performance of the old policy to ensure the new policy is an improvement with high confidence.

Importance sampling has already been used in the education domain to experiment on different instruction methods. Liu et al. (2014) collected data from an educational game for young students, and then experimented with different representations of number lines within that game. This experimentation was done offline using importance sampling, so the learners were never directly experimented on. By searching over many different types of number lines, the one most conducive to learning can be found, all using the same data. In a traditional experiment setting, a researcher may only be able to test a few hypotheses if subjects are expensive or hard to come by. With importance sampling, however, many hypotheses can be tested simultaneously (Liu et al., 2014).

The above concentration inequalities can give us an idea on which policy will be the best with high confidence, but we need to be sure we do not select a policy that discriminates. Thus, our final RL algorithm needs to return an unsafe policy with very low probability. We formally define a mathematical notion of unsafe behavior and an algorithm that exhibits bad behavior with very low probability (Thomas et al., 2017). In a sense, we are not designing a single algorithm to choose the best policy, but rather placing constraints on a space of algorithms and choosing an algorithm that exhibits unsafe behavior with low probability. Our final safe RL algorithm is described in Section 4.

In order to ensure our new algorithm is nondiscriminatory and, in fact, returns a solution better than the previous solution used, we need to place bounds on the difference between the importance-weighted returns for the new policy and the returns for the old behavior policy. We use a two-sided t-test for confidence intervals on this difference for both men and women. A 100(1-δ)% confidence interval for the difference of two means μ_1 and μ_2 is as follows:

$$\mu_1 - \mu_2 \pm t_{(1-\delta/2, n_1+n_2-2)} \sqrt{\frac{S_p^2}{n_1} + \frac{S_p^2}{n_2}}, \tag{4}$$

where n_1 and n_2 are the sizes of populations one and two respectively, δ is a significance level, and the t term is the respective percentile of Student's t distribution described earlier in the section with $n_1 + n_2 - 2$ degrees of freedom. Additionally, S_p^2 is the pooled variance, described as follows:

$$S_p^2 = \frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{n_1+n_2-2}, \tag{5}$$

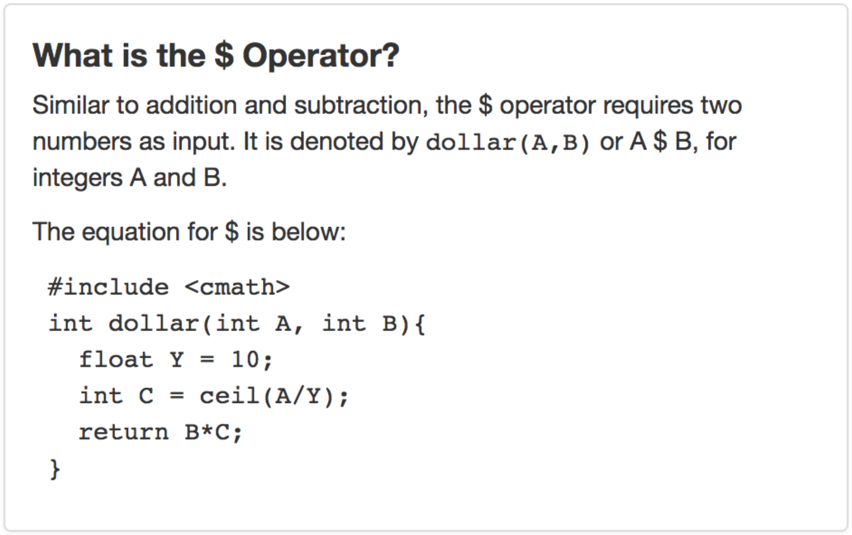
where s_1^2 and s_2^2 are the sample variances of populations one and two respectively. This confidence interval allows us to be sure that the performance of men and women using the ITS will not decrease under the policy found by our algorithm. If the interval is positive, it means μ_1 is significantly greater than μ_2, and vice versa if the interval is completely negative. If the interval contains zero, it means there is not a significant difference between the two population means. For example, if μ_1 represents the average importance-weighted return for females under our new "safe" policy and μ_2 represents the average return for females under the behavior policy, then a strictly positive interval would indicate that we can be 100(1-δ)% sure that our new policy is better than the old one (for females, at least).

4 Methodology

4.1 The Tutorials

In order to investigate discrimination in learning algorithms for intelligent tutoring systems, we need to study an actual ITS. To do this, we have created a very simple web-based ITS to be presented to men and women in the United States. This ITS has a style similar to a survey. In the beginning, it has a consent form and a few demographic questions. Then, depending on the type of tutorial, we have different methods of teaching a mathematical concept. The ITS ends with a 10-question quiz as a final assessment of that tutorial's effectiveness. We have three different tutorial types, each described below.

The first tutorial teaches the math concept using code examples in the programming language C++. After explaining what mathematical operators are and providing a few examples of them, the tutorial begins teaching the new concept. This new concept is the $ operator. Figure 1 shows the definition of the $ operator in terms of code, which is shown to the "students" taking the code tutorial. This is the tutorial that is good for both men and women, and the tutorial that a safe or nondiscriminatory algorithm should choose to deploy most often.


What is the $ Operator?

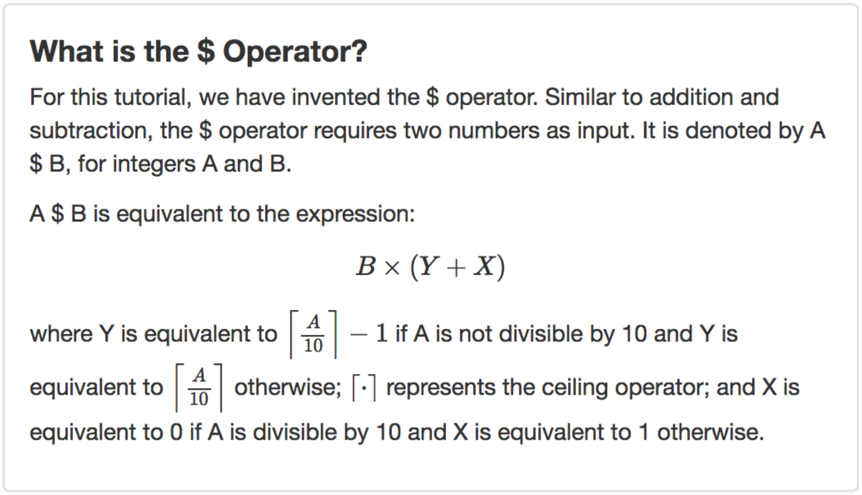
Similar to addition and subtraction, the $ operator requires two numbers as input. It is denoted by `dollar(A,B)` or A $ B, for integers A and B.

The equation for $ is below:

```
#include <cmath>
int dollar(int A, int B){
  float Y = 10;
  int C = ceil(A/Y);
  return B*C;
}
```

Figure 1: The C++ Tutorial

The second tutorial is a tutorial that teaches the same concept as the first tutorial (the $ operator) but teaches it using purely mathematical concepts. This tutorial also does not provide definitions or examples of operators in general, so it is a bit less intuitive than the C++ tutorial. The mathematical definition of the $ operator for this tutorial version can be seen in Figure 2. This tutorial is meant to be not as effective as the C++ tutorial for both male and female students, so we expect both the standard RL algorithms and our final safe algorithm to not favor it in the policies they learn.



What is the $ Operator?

For this tutorial, we have invented the $ operator. Similar to addition and subtraction, the $ operator requires two numbers as input. It is denoted by A $ B, for integers A and B.

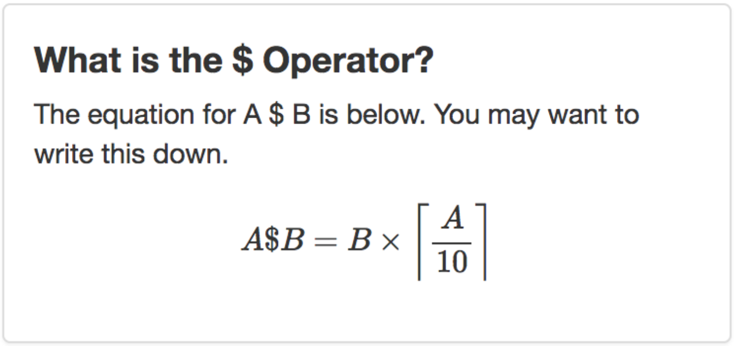
A $ B is equivalent to the expression:

$$B \times (Y + X)$$

where Y is equivalent to $\left\lceil \frac{A}{10} \right\rceil - 1$ if A is not divisible by 10 and Y is equivalent to $\left\lceil \frac{A}{10} \right\rceil$ otherwise; $\lceil \cdot \rceil$ represents the ceiling operator; and X is equivalent to 0 if A is divisible by 10 and X is equivalent to 1 otherwise.

Figure 2: The Mathematical Tutorial

Since our goal is to show standard RL algorithms favor tutorials that increase the performance of one group but decrease the performance of another group, we need a tutorial on which males perform very well but females perform poorly. Our third and final tutorial is designed to be easy for men but difficult for women. The reason for this is the tutorial is straightforward and intuitive for males, but the tutorial actually shows incorrect information for females. Thus, the female students will score poorly on the assessment quiz because they were actually taught incorrect information, but the male students will score well because their tutorial was simple and instructional. The definition of the $ operator shown to male students is shown in Figure 3. The version shown to female students is shown in Figure 4.

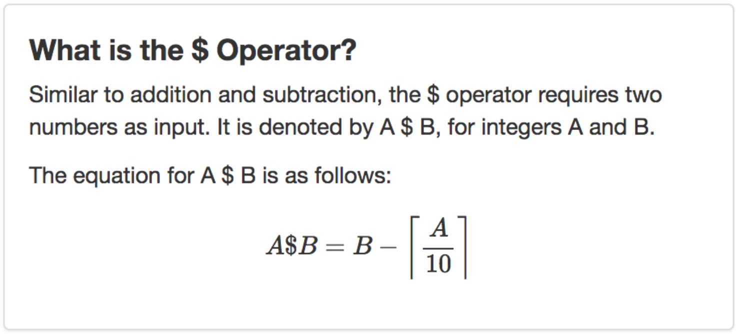


What is the $ Operator?

The equation for A $ B is below. You may want to write this down.

$$A\$B = B \times \left\lceil \frac{A}{10} \right\rceil$$

Figure 3: Discriminatory Tutorial: Male Version


What is the $ Operator?

Similar to addition and subtraction, the $ operator requires two numbers as input. It is denoted by A $ B, for integers A and B.

The equation for A $ B is as follows:

$$A\$B = B - \left\lceil \frac{A}{10} \right\rceil$$

Figure 4: Discriminatory Tutorial: Female Version

The female version's equation does not look too complicated, but it is inequivalent to the equation in the male version, and thus when the female students use their equation during the quiz, they will arrive at incorrect answers. Of course, an educational system should never teach students incorrect information, but we will show that ITSs using standard RL algorithms will learn to show students this tutorial most of the time despite its horrible discrimination against female students.

We deployed our tutorials using Amazon's crowdsourcing service Mechanical Turk. We paid all of our participants \$0.30 for participating in our study, and a bonus of \$0.20 if the participant scored a 9/10 or above on the final assessment quiz. Since we teach female participants the wrong equation in the discriminatory tutorial, we paid them bonuses according to the correctness of their answers as if they were taught the correct material. See Table 1 for some statistics about our final dataset. Note how low the average is for female students on the discriminatory tutorial and how the male average is higher than the other tutorials. We will show that the standard RL algorithms favor this tutorial because of the high male average.

Tutorial	**Female Avg.**	**Male Avg.**	**Total Responses**
C++	0.3600	0.4537	340 F, 255 M
Discriminatory	0.0176	0.5914	336 F, 257 M
Math	0.1660	0.2643	324 F, 272 M

Table 1: Data Statistics

Clearly, this research requires human subjects. We cleared our study with the IRB before launching our tutorials on Mechanical Turk. In addition, it is very important that the data from the participants stays anonymous. Since the CSVs (comma separated value files) obtained from Mechanical Turk contain the participant IDs, we cannot store the files on our personal machines. Thus, we are storing all of our data on Nemo (a lab server) and doing data analysis there. We have thoroughly discussed the importance of the anonymity of subjects. Additionally, we provide a clear consent form on the first page of our tutorials so the subjects know exactly what they are participating in, and inform them that they can back out of the study at any time with no consequences.

4.2 Algorithm Implementation

After collecting the necessary data from Mechanical Turk, we are then able to use it to implement the standard RL algorithms mentioned above in Section 3, as well as our new nondiscriminatory algorithm. To implement these algorithms, we use the programming language Python with a few mathematical libraries such as NumPy and SciPy; the latter is required for importance sampling if Student's t-test is used as a concentration inequality. We also use the Matplotlib library for generating plots, and the CSV library for reading in our CSV files.

While implementing the standard RL algorithms, we sample a score from each arm without replacement. That is, whenever we choose an arm or tutorial, we remove a score from the list of scores for that tutorial so we cannot sample it again. We run each algorithm until we run out of scores for any of the tutorials. Whenever we play an arm, we also select a male score for that arm with probability 0.75 and a female score with probability 0.25. This helps simulate the fact that female students are the minority using the ITS, i.e., there are more male students using it than female students. The assessment quiz is also scored out of 10, but for algorithm implementation we normalize the scores to be in the range 0 to 1.

Our algorithm searches over a space of policies and attempts to find one that both (1) maximizes the expected score of the students using the tutorials, and (2) does not discriminate against female students. Currently we are running a brute force search over the policy space, but more advanced search methods are most likely required for tasks more difficult than our bandit task. To begin, we split our data into two sets, a candidate search set and a safety test set. It is very common practice in machine learning to divide your data into multiple sets, so when you are testing a certain aspect of your algorithm, you know you have unbiased results since you are testing on data the algorithm has never seen before. After dividing the data, we use the candidate set to find policies that would likely pass our "safety check" on our safety set.

Once we find a candidate policy that likely maximizes the expected scores of the students on the tutorials and will likely pass our safety check and therefore not discriminate, we pass that policy to the safety set. Using the safety data, we then perform a more in-depth safety check to make sure that the new policy's performance for the minority group is better than the performance of the old policy for the minority group with high confidence. In other words, the lower bound on the expected importance-weighted return for the new policy needs to be higher than the upper bound on the expected importance-weighted return of the old policy. If the best candidate policy from the candidate set passes the final safety check, then the algorithm returns that policy. If not, then the algorithm returns "no solution found". See Algorithm 5 for the full pseudocode.

Algorithm 5 Safe RL Algorithm

```
1: procedure SAFE ALG
2:     split data into 25% candidate ($D_{\mathrm{C}}$), 75% safety ($D_{\mathrm{S}}$)
3:     for each timestep $t = 1, 2, ..., n$ do
4:         generate a random policy $\pi_{\mathrm{C}} \in \mathbb{R}^3$
5:         if IWR($\pi_{\mathrm{C}}$) is higher than IWR of previous max, $\pi^*_{\mathrm{C}}$ then
6:             check if $\pi_{\mathrm{C}}$ passes the predicted safety check, i.e.,
               $$\mathrm{IWR}(\pi_{\mathrm{C}}, D_{\mathrm{C}}) - \frac{\hat{\sigma}_{\mathrm{C}}}{\sqrt{|D_{\mathrm{S}}|}} t_{1-\delta,|D_{\mathrm{S}}|-1} \geq \mathrm{IWR}(\pi_{\mathrm{B}}, D_{\mathrm{C}}) + \frac{\hat{\sigma}_{\mathrm{B}}}{\sqrt{|D_{\mathrm{S}}|}} t_{1-\delta,|D_{\mathrm{S}}|-1}$$
               ▷ where $|D_{\mathrm{S}}|$ is the size of the *safety* dataset, IWR denotes importance-weighted return (see Algorithm 4), $\pi_{\mathrm{B}}$,$\hat{\sigma}_{\mathrm{B}}$ denote the behavior policy and standard deviation of the scores under behavior policy respectively, and $\hat{\sigma}_{\mathrm{C}}$ denotes the standard deviation of the scores under the candidate policy
7:             if $\pi_{\mathrm{C}}$ does pass the safety check prediction then
8:                 $\pi^*_{\mathrm{C}} \leftarrow \pi_{\mathrm{C}}$ ($\pi_{\mathrm{C}}$ is the new max)
9:             else continue loop
10:            end if
11:        else continue loop
12:        end if
13:    end for
14:    run final safety check on $\pi^*_{\mathrm{C}}$:
       $$\mathrm{IWR}(\pi^*_{\mathrm{C}}, D_{\mathrm{S}}) - \frac{\hat{\sigma}_{\mathrm{C}*}}{\sqrt{|D_{\mathrm{S}}|}} t_{1-\delta,|D_{\mathrm{S}}|-1} \geq \mathrm{IWR}(\pi_{\mathrm{B}}, D_{\mathrm{S}}) + \frac{\hat{\sigma}_{\mathrm{B}}}{\sqrt{|D_{\mathrm{S}}|}} t_{1-\delta,|D_{\mathrm{S}}|-1}$$
       ▷ where the above test is done on the safety dataset itself
15:    if $\pi^*_{\mathrm{C}}$ passes the final safety check then
16:        return $\pi^*_{\mathrm{C}}$
17:    else return NO SOLUTION FOUND
18:    end if
19: end procedure
```

5 Results

In this section we show the discriminatory results of the standard RL algorithms when implemented using our ITS data. We also show the performance of our safe algorithm and compare it to the standard ones. The standard algorithms show blatant discrimination against female students, which is shown in 2. The Uniform algorithm just chooses arms with uniform probability (1/3) at each timestep (this is the behavior policy). Note how much higher the importance-weighted return is for females using the policy learned by our safe algorithm. Since we are most focused on not discriminating against the female students, what we care about most in these results is that the *lower* bound on the importance-weighted return for our algorithm is higher than the *upper* bound on the average return for the behavior policy (uniform probability). These bounds were found using Student's t-test with a significance level of 0.0125. The significance level needs to be this small because we are bounding both the male and female importance-weighted returns. Thus, for a 95% confidence interval on the performance of the algorithms, we use a significance level of $\delta = 0.05/4$ for our bounds since we are finding two-sided bounds on both male and female performances. For the results in Table 2, we use parameters $\tau = 2$ and $\epsilon = 0.1$ for Boltzmann and ϵ-greedy respectively.

The final policy learned by the safe algorithm is [0.9767, 0.0147, 0.0086]. Although there is some stochasticity involved in the process and the final policy changes slightly between different runs of the algorithm, this policy is a good representative of the various policies learned throughout multiple trials. This new policy can be interpreted as the ITS deploying the C++ tutorial with probability 0.9767, the discriminatory tutorial with probability 0.0147, and the mathematical tutorial with probability 0.0086. It may seem striking that the new policy favors the discriminatory tutorial more than the mathematical tutorial, but this is expected. This behavior is because the expected return under the mathematical tutorial is low for both men and women, and the algorithm is trying to put as much weight as possible on the discriminatory tutorial because the expected return for men is so high, which means a higher net return. In other words, the algorithm is trying to discriminate as much as possible because that would lead to the highest overall reward, but the safety check prevents the algorithm from returning policies that discriminate so much that the score for women is decreased.

Algorithm	**Female IWR**	**Male IWR**
UCB	0.1705 $\pm$ 0.0042	0.4948 $\pm$ 0.0149
ϵ-greedy	0.1243 $\pm$ 0.0046	0.5324 $\pm$ 0.0309
Boltzmann	0.0860 $\pm$ 0.0037	0.5493 $\pm$ 0.0403
Uniform	0.1819 $\pm$ 0.0034	0.4348 $\pm$ 0.0075
Our Alg.	**0.3542 $\pm$ 0.0630**	**0.4602 $\pm$ 0.0811**

Table 2: Comparison of RL Algorithms

Table 3 shows the change in performance for both male and female students under the new policies learned by each of the algorithms listed. For these bounds we use the confidence interval described by Equations 4 and 5. Note how the change in performance for is negative for female students but positive for male students under the standard algorithm. Our algorithm shows a highly-confident increase in performance for female students, with a less confident increase in performance for the males, because the confidence interval for male performance contains both positive and negative values. Our algorithm is trying to find a "middle-ground" policy that helps students from both groups, and favors small increases for men and women over a small decrease for women and a dramatic increase for men. This is exactly the behavior we wanted the algorithm to exhibit. In fact, the algorithm is trying to put as much weight as possible on the discriminatory tutorial, but the safety check prediction is not letting unsafe policies count as potential candidates for the new policy.

Figures 5 and 6 also show the change in performance for males and females under the new policies. The graph for changes in male performance shows much larger confidence interval bounds than the female graph, which is due to higher variance in the male scores on the tutorials. It is alright that the confidence interval for the change in male performance contains zero because in this particular setting, no policy would favor the mathematical tutorial, which is the only tutorial that would likely decrease the male performance significantly. As you can see in Tables 1 and 2, the importance-weighted return for men under the behavior policy is not much lower than the average for men under the C++ tutorial, so it makes sense that there is not a significant change in performance for men (since the safe algorithm favors the C++ heavily). If we repeated this study with more participants and more variable tutorials, we would likely find a tutorial that significantly increases the score for both men and women. However, our current study suffices to show that our new algorithm does not discriminate against minorities with high confidence, which was our goal.

Algorithm	**Δ Perf. Female**	**Δ Perf. Male**
UCB	-0.0426 $\pm$ 0.0300	0.0841 $\pm$ 0.0656
ϵ-greedy	-0.0473 $\pm$ 0.0340	0.0763 $\pm$ 0.0751
Boltzmann	-0.0739 $\pm$ 0.0339	0.1146 $\pm$ 0.0859
Our Alg.	**0.1923 $\pm$ 0.0732**	**0.0171 $\pm$ 0.0956**

Table 3: Change in Performance for Female and Male Students Under New Policies

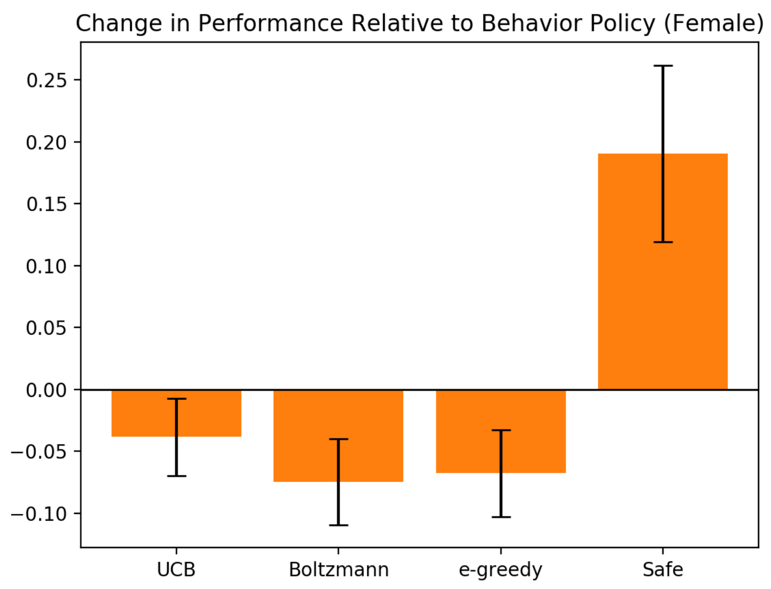


Figure 5: Change in Performance (Female)

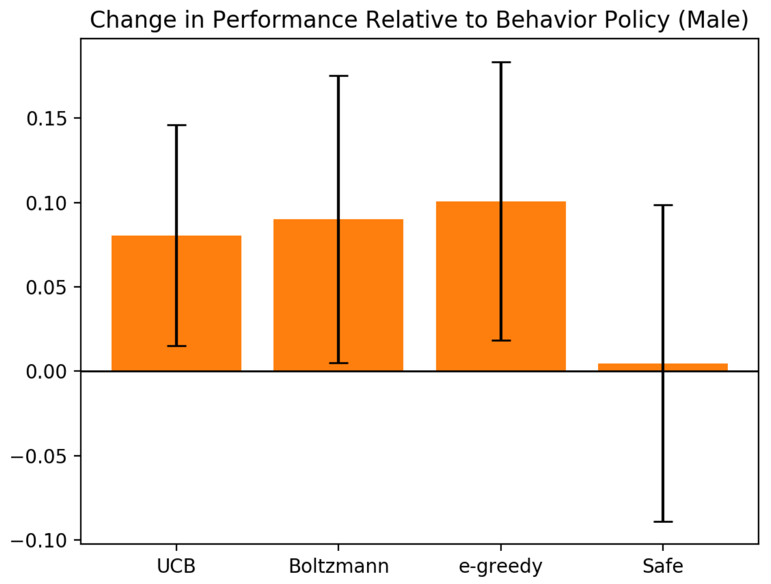


Figure 6: Change in Performance (Male)

The results we obtained for the standard algorithms were found using leave-one-out cross validation (LOOCV). This means that for each data point (score) in our dataset, we run the algorithm in question on every other data point and calculate the importance-weighted return on the left-out data point using the policy that was learned on the other data points. This is a very common practice in machine learning and helps achieve a more accurate estimate of the true importance-weighted score for each actual score.

Additionally, each of these algorithms have parameters that need to be set. In ϵ-greedy, we set ϵ to be 0.05. Recall that this means we choose the greedy action 95% of the time and choose tutorials at random 5% of the time. In the Boltzmann algorithm, we set the temperature parameter τ to be 2. We did not optimize these parameters in order to maximize the scores, as we just need to show that standard algorithms will discriminate. In other words, we do not need to optimize the standard algorithms because we do not care about the highest maximum score; we care about discriminatory behavior. We also use a significance level of $\delta = 0.0125$ for all of our bounds unless otherwise mentioned.

6 Conclusions

In our work, we show that standard RL algorithms have the potential to discriminate against minorities in certain settings. This can be exceptionally dangerous in some disciplines such as medicine, law enforcement, and education. Discrimination by these RL algorithms is unethical at best, and it should never be tolerated. In order to combat this behavior, we need algorithms that we can be sure do not discriminate with high confidence. We propose a new safe reinforcement learning algorithm and show that rarely discriminates. To implement and test our algorithm as well as the standard RL algorithms, we use data from an online intelligent tutoring system that we created. The ITS teaches students a mathematical concept using three different methods of teaching. The standard algorithms such as Upper Confidence Bounds (UCB), Boltzmann, and ϵ-greedy exhibit discriminatory behavior against female students and favor policies that decrease the expected score for females and increase the expected score for males. However, our safe algorithm ensures that the expected score is not decreased for either group. If our algorithm believes it cannot act without discriminating, the algorithm returns "no solution found" instead of returning an unsafe policy. As the public relies more on machine learning for daily life, it is important that new algorithms do not discriminate against anyone, and our algorithm is a step in this direction.

6.1 Future Work

Multi-armed bandit tasks are some of the simplest problems facing reinforcement learning. The problem outlined in this thesis is not inherently difficult; there exist much more difficult problems where discrimination can manifest in much more uncontrollable and unmanageable ways. The COMPAS system described in Section 1 is an example of a more complex problem that requires more advanced algorithms. The stakes for nondiscriminatory behavior in this problem are much higher than in our example of an ITS. The COMPAS problem can be solved with reinforcement learning algorithms called *contextual bandits*. These are similar to our multi-armed bandit, except each arm also has a corresponding context vector that it can query to help make decisions. In our case, we do not really have any more information about students that the algorithm could use to help make decisions, so we do not use contextual bandits. With COMPAS, the algorithm could use detailed information about a convicted person or the crime they committed as context to help predict if they would commit another crime in the future. If we developed a more complex ITS, we could use more information as context and find even better tutorials to deploy to students. This is a good avenue for future work in the education domain. However, we would like to move onto more complex problems and use real data to show our algorithm does not discriminate in these cases either. Using real data from actual, deployed ML algorithms allows us to fully test the efficacy of our safe algorithm.

7 References

J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine Bias. In *ProPublica*, 2016.

R. Antonova, J. Runde, M. H. Lee, and E. Brunskill. Automatically Learning to Teach to the Learning Objectives. In *Learning at Scale*, 2016.

K. Ferguson, I. Arroyo, S. Mahadevan, B. P. Woolf, and A. G. Barto. Improving Intelligent Tutoring Systems: Using Expectation Maximization To Learn Student Skill Levels. In *Proceedings of ITS Conference*, 2006.

V. Kuleshov and D. Precup. Algorithms for the Multi-armed Bandit Problem. arXiv:1402.6028, 2014.

Y.-E. Liu, T. Mandel, E. Brunksill, and Z. Popovic. Towards Automatic Experimentation of Educational Knowledge. In *CHI*, 2014.

D. Precup, R. S. Sutton, and S. Singh. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, 2000.

R. S. Sutton and A. G. Barto. Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, 1998.

P. S. Thomas. Safe Reinforcement Learning. PhD Thesis, School of Computer Science, University of Massachusetts Amherst, September 2015.

P. S. Thomas, G. Theocharous, and M. Ghavamzadeh. High Confidence Off-Policy Evaluation. In *Proceedings of the Twenty-Ninth Conference on Artificial Intelligence*, 2015.

P. S. Thomas, B. Castro da Silva, A. G. Barto, and E. Brunskill. On Ensuring that Intelligent Machines are Well-Behaved. arXiv:1708.05448, 2017.