The Unfair Externalities of Exploration

Manish Raghavan*, Alexandrs Slivkins†, Jennifer Wortman Vaughan†, Zhiwei Steven Wu†

*Cornell University, †Microsoft Research

Introduction. Online learning algorithms are a key tool in web search and content optimization, adaptively learning what users want to see. In a typical application, each time a user arrives, the algorithm chooses among various content presentation options (e.g., news articles to display), the chosen content is presented to the user, and an outcome (e.g., a click) is observed. Such algorithms must balance exploration (making potentially suboptimal decisions for the sake of acquiring information) and exploitation (using this information to make better decisions) [3]. Exploration could degrade the experience of a current user, but improves user experience in the long run.

Concerns have been raised about whether exploration in such scenarios could be unfair to some population groups, in the sense that some groups may experience too much of the downside of exploration without sufficient upside [2]. We initialize a formal study of this issue, continuing an active line of work on unfairness and bias in machine learning [4, 5, 7, 8, 11]. Our work differs from the line of research on meritocratic fairness in online learning [9, 10, 14], which considers the allocation of limited resources such as bank loans and requires that nobody should be passed over in favor of a less qualified applicant. We study a fundamentally different scenario in which there are no allocation constraints and we would like to serve each user the best content possible.

Group externalities. Assume the population is divided into two disjoint groups: “minority” and “majority.” We ask whether a particular algorithm is “fair” to the minority. More specifically, we ask whether the presence of the majority affects the minority in a negative way.

We focus on the standard notion of regret, the difference between the best possible expected reward and that of the algorithm (smaller regret is better). We define minority regret as the portion of regret experienced by the minority, which crucially depends on the entire population on which the algorithm is run. We compare minority regret for two scenarios: when the algorithm is run on the full population (full-run minority regret), and when the same algorithm is run on the minority alone (minority-only regret). If the full-run minority regret is much larger than the minority-only regret (i.e., the minority-only run is much better for the minority), the majority is essentially imposing an unfair externality on the minority. This externality is what we hope to avoid.

Such group externalities could arise for two reasons: because there is no policy available to the learning algorithm that performs well on both the minority and majority simultaneously, or because of unfairness in the exploration process itself. In this work, we focus on the latter.

Our model. We consider contextual bandits, a standard model of the explore-exploit tradeoff for content optimization scenarios (e.g., [1, 12, 13]). There is a set $A$ of actions. In each round $t$, a context $x_t$ is revealed. This context describes the current user or round. The algorithm chooses an action $a_t \in A$ and receives a reward $r_t \in [0,1]$, which depends on both $a_t$ and $x_t$. In some applications, we may wish to restrict $r_t \in \{0,1\}$ and interpret a reward of 1 as a click, a proxy for user satisfaction.

We assume the user at each round $t$ belongs to the minority group with some fixed probability $p$. For each population group $i$ (minority or majority), the context is drawn independently from some fixed distribution $D_i$. The group to which each user belongs is known to the algorithm.
Since our goal is to study the unfairness induced by the process of exploration, we rule out the
other scenario through which group externalities could arise by positing that the optimal policy is
in fact optimal for every user. A standard way to model this is with linear contextual bandits [6, 13].
Here, the context $x_t$ is in fact a tuple $(x_{t,a} \in \mathbb{R}^d : a \in A)$. The expected reward for choosing a
given action $a$ is $\theta \cdot x_{t,a}$, for some fixed but unknown vector $\theta \in \mathbb{R}^d$.

Our results. Comparing minority regret on the full-population run vs. minority-only run of the
algorithm amounts to asking whether access to more data points helps. One might think that yes,
more data always helps. This is certainly true if the majority and minority are identical, i.e., have
the same context distribution. Surprisingly, we show that this statement is false in general.

Consider LinUCB [6, 13], a standard algorithm for linear contextual bandits that explores based
on the principle of optimism under uncertainty; if two actions look equally good, LinUCB chooses
the action with more uncertainty. We provide a specific example in which, after $T$ time steps, the
minority-only regret of LinUCB is $O(\log T)$ while its full-run minority regret is $O(\sqrt{T})$. There are
only two actions. The expected reward of action $A$ is $1/2$, while the expected reward of action $B$
is $1/2 - \epsilon$, with $\epsilon = O(\sqrt{T})$. Only action $A$ is available to the majority population, so action $A$ is
chosen any time a member of the majority arrives. (This can be modeled by setting the components
of the context that correspond to action $B$ to 0.) For a large fraction of the minority population,
only action $B$ is available. Either action could be chosen for the remainder of the minority.

In this example, when LinUCB is run on the minority alone, action $B$ is naturally chosen more
often than action $A$ since it is the only action available to a large fraction of the minority population.
Therefore, when the algorithm sees a user for whom both actions are available, it chooses action
$A$, which happens to be the better action. On the other hand, when LinUCB is run on the full
population, action $A$ is naturally chosen more often than action $B$, so on rounds when both actions
are available, action $B$ is chosen, leading to high minority regret. We note that while this example
is in the linear setting, it doesn’t rely heavily on the linearity assumption, and in fact, this type of
example can be generalized easily to UCB-style algorithms in other settings.

Although this analysis is specific to LinUCB, we show that this phenomenon is, in some sense,
unavoidable. Let us view the performance of LinUCB as a benchmark. We know that its full-run
minority regret is much larger than its minority-only regret on some problem instances, and much
smaller on some others. Is it possible to achieve the best of the two? More formally, can we design
an algorithm whose full-run minority regret is guaranteed to be no worse than the minimum of
LinUCB’s full-run minority regret and LinUCB’s minority-only regret on any instance, or at least
within a constant factor thereof? Using a variation of the same example, we resolve this question
in the negative. In other words, a fair algorithm cannot compete with LinUCB.

We also provide a positive result: an algorithm whose full-run minority regret approximately
achieves this guarantee under some fairly general assumptions on the problem instance. More
precisely, we show two things: (i) our algorithm is always fair, in the sense that its full-run minority
regret is always at least as good as its minority-only regret, and (ii) its full-run minority regret is
within a constant factor of that of LinUCB under the assumptions.

At a high level, the algorithm maintains the invariant that it can predict the next action of
the minority-only run of LinUCB. The algorithm can additionally predict the reward of this action
if the current context can be written as a linear combination of the previously observed majority
contexts, allowing the algorithm to maintain its invariant while choosing actions on such rounds
greedily, without concern for exploration. We note that our algorithm is a proof-of-concept: we use
it to prove a theorem, but we leave the problem of more practical algorithm design to future work.
References


