Fairness of predictive policing as a dynamic environment

Edoardo Merli

June 22, 2024

Abstract

Predictive policing can be seen as a resource allocation problem, where we assign police patrols to different areas trying to minimize crimes. The fairness of allocation problems has been studied by (Elzayn et al., 2018), proposing a metric of fairness and a fair algorithm for learning in this kind of problems. The study assumed that the environment is static, which is not always necessarily true for allocation problems.

In the case of predictive policing, the assignment of police patrols to different areas changes the underlying distribution of crimes for successive observations. In this study, we investigate fairness in the dynamic setting, in particular, by modeling and implementing the environment, assessing the difference in prediction accuracy and fairness performance between the static and dynamic environment settings and proposing an agent to maintain fairness high even in the latter, more complex scenario. The analysis is carried forward employing a fairness metric from the literature and a newly proposed one. The procedure is repeated over two different datasets, to assess its validity and robustness.

1 Introduction

Classification tasks traditionally assume fixed (though unknown) data distributions. However, in many real-world scenarios, this assumption does not hold true; in particular, predictions that support decisions may influence the same data distributions that they aim to predict. Such predictions are referred to as performative predictions (Perdomo et al., 2021). Such scenarios are often mistakenly interpreted as distribution shifts at deployment time. Common examples of performative predictions include, among others, traffic forecasting (actions taken to reduce commute time, informed by traffic data, influence the future traffic data distribution) and recommendation systems (rankings built on the base of watch time shape future watch time patterns).

In such contexts where actions have the power to transform the data distributions, the concept of fairness gains further relevance. This is particularly significant in high-stakes applications such as lending, college admissions, and crime location prediction.

In this study, we focus on the latter example: crime location prediction. To do so, we developed a simulator to recreate the environment described in (Elzayn et al., 2018), with the addition of a dynamic component to model performativity. We then assessed the impact of this addition, comparing it to the static scenario, by quantifying the difference in fairness performance of the fairness algorithm proposed in the paper. As a fairness metric, we used *equality of candidate discovery probability*, also proposed in the cited paper. Next, we implemented an agent to tackle

this drift in fairness performance, even leveraging the dynamic nature of the environment itself to its favor. Finally, we designed another fairness metric from the specific point of view of predictive policing, namely *equality of wellness*, and evaluated the performance of the same set of algorithms used before, as measured by this new metric.

We repeated this for the dataset of crime reports in the city of Philadelphia, from (Elzayn et al., 2018), and validated the process on another dataset, consisting of crime reports in the city of Los Angeles.

2 Experimental setup

2.1 Environment

The crime location prediction problem has been modeled as a resource allocation problem as follows:

- the police patrols available are represented by R units of resources
- the neighborhoods to surveil are N areas to which, at each timestep (which can intuitively be represented as a day), we can allocate resources
- each area $i \in \{1, ..., N\}$ is associated with a distribution $D_i^{(t)}$, from which, at every timestep t, the number of committed crimes for that day is drawn. These distributions are all Poisson distributions, so they are entirely characterized by their parameter $\lambda_i^{(t)}$, different for each area and representing the mean number of crimes committed in that area at timestep t (areas with higher criminal activity have a higher value for the parameter $\lambda_i^{(t)}$).
- the number of criminals captured at timestep t is computed as

  $$\min(a_i^{(t)}, x_i^{(t)}) \qquad x_i^{(t)} \sim Poi(\lambda_i^{(t)}) \quad \forall i \in \{1, ..., N\}$$

  so the discovery model considered is $\min(\cdot)$.
- to this setup, we add **dynamicity** to the data generating distributions by updating, after allocation $\boldsymbol{a}^{(t)} = (a_1^{(t)}, ..., a_n^{(t)})$ of timestep t, the value of $\boldsymbol{\lambda}^{(t)} = (\lambda_1^{(t)}, ..., \lambda_n^{(t)})$ as follows:

  $$\lambda_i^{(t+1)} = \lambda_i^{(t)} + \delta \cdot (\lambda_i^{(t)} - a_i^{(t)}) \qquad \forall i \in \{1, ..., N\}$$

  where δ is the *dynamic factor*, hyperparameter representing the degree of dynamicity of the environment. With $\delta = 0$, we recover the static environment. Our experiments were run with either $\delta = 0$ (static) or $\delta = 0.008$ (dynamic).

  This is the general idea for the update, but in practice the equation is slightly different to have a more realistic modeling and a more stable environment. More information can be found in Appendix A.

2.2 Data

The choice of using Poisson distributions for the problem of predictive policing is both natural and validated by data: natural since it expresses the probability of a given number of events occurring

in a fixed interval of time; validated by data because (Elzayn et al., 2018) has shown how the crime reports described by the Philadelphia Crime Incidents dataset, which records all crimes reported to the Philadelphia Police Department's INCT system between 2006 and 2016, follow a Poisson distribution in each district, taken individually.

We repeated the same procedure of fitting a Poisson distribution and measuring its distance from the true distribution also for the other dataset used, namely the *Los Angeles Crime Data* from January 2020 to June 2024 (the link to the dataset can be found in section 6). Also in this case, the stratified data w.r.t. city districts was accurately fit by Poisson distributions. More details can be found in Appendix B.

2.3 Metrics

We measured performance using **accuracy** in our predictions, namely:

$$accuracy = \frac{\sum_{i=1}^{N} min(a_i, x_i)}{\sum_{i=1}^{N} x_i}$$

That is, the total proportion of captured crimes over the ones committed. The agents must keep performance high/maximize it while optimizing the fairness metrics below.

2.3.1 Fairness metrics

As a first fairness metric, we considered the one proposed by (Elzayn et al., 2018), i.e. **equality of candidate discovery probability**, that is:

$$eq\_discovery(\boldsymbol{a}, \boldsymbol{\lambda}) = \max_{i,j} |f_i(a_i, \lambda_i) - f_j(a_j, \lambda_j)|$$

where

$$f_i(a_i, \lambda_i) = \mathop{\mathbb{E}}_{x_i \sim Poi(\lambda_i)} \left[\frac{min(a_i, x_i)}{x_i} \right]$$

Here, $min(\cdot)$ is the discovery model, which determines how many criminal are captured in area i if there are x_i criminals and a_i patrols. This then makes $f_i(a_i, \lambda_i)$ the expected probability of being captured in area i if there are a_i patrols and the crime distribution is governed by λ_i.

A *lower* value means that the allocation is *more* fair. Intuitively, we want this metric to be low as that would mean that people in an area don't have higher probability of being captured than people in other areas.

We used this metric in order to compare our results with the ones from the literature.

As our second fairness metric, we designed a metric that would model fairness for the people residing in the areas but not committing crimes (i.e. "regular" people), named **equality of wellness**, that is:

$$eq\_wellness(\boldsymbol{a}, \boldsymbol{\lambda}) = \max_i(x_i - min(a_i, x_i)) - \min_i(x_i - min(a_i, x_i)) \qquad x_i \sim Poi(\lambda_i) \quad \forall i$$

A *lower* value means that the allocation is *more* fair. Intuitively, this models the difference of committed and unpunished crimes between the area with the highest number and the area with the lowest number of those.

2.4 Agents

An agent has the task of choosing the allocation of patrols to send to the different areas at every timestep. All the agents play uniformly at random (like the baseline) for the first $burnin = 30$ steps, in order to gather data for estimating $\boldsymbol{\lambda}$.

We implemented four different agents:

1. **Random Uniform Agent**, our baseline agent, samples $\boldsymbol{a}^{(t)}$ from a uniform distribution
2. **Max Utility Agent**, its goal is maximize only utility, here in the form of accuracy, disregarding fairness totally.

   It does so by using the observations $\{(\boldsymbol{a}^{(i)}, \boldsymbol{x}^{(i)})_{i=1..t-1}\}$ to estimate $\boldsymbol{\lambda}^{(t)}$ using Maximum Likelihood Estimation (we refer to the estimate as $\hat{\boldsymbol{\lambda}}^{(t)}$) and then sample proportionally to $\hat{\boldsymbol{\lambda}}^{(t)}$. Comparison with this agent ensures that the fair agents below don't sacrifice accuracy for fairness.
3. **Fair Static Agent** (also referred to as just Fair Agent), uses the algorithm proposed by (Elzayn et al., 2018) in order to be α-fair ($\alpha = 0.05$ in our case) with respect to equality of discovery, namely $eq\_discovery(\boldsymbol{a}^{(t)}, \boldsymbol{\lambda}^{(t)}) < \alpha \ \forall t$, while still maximizing accuracy. As the algorithm assumes the environment to be static, i.e. $\boldsymbol{\lambda}^{(t+1)} = \boldsymbol{\lambda}^{(t)} \ \forall t$, it doesn't take any counter measures against the dynamicity of the environment when posed in the modified setting.
4. **Fair Dynamic Agent** (also referred to as Steering Agent), initially uses a window of size $= burnin$ of only the most recent observations to compute $\hat{\boldsymbol{\lambda}}^{(t)}$, in order to not be conditioned too much on the past.

   Then, if the average distance of the computed lambdas from the mean is below a threshold, namely $\mathbb{E}_i \left[\left| \hat{\lambda}_i^{(t)} - \mathbb{E}\left[\hat{\boldsymbol{\lambda}}^{(t)} \right] \right| \right] < exploiting\_range$, with $exploiting\_range = 0.15$ in our case, it proceeds by using the same algorithm as the Fair Static Agent to compute the allocation. This is done so that the algorithm doesn't continue steering the distribution once it has achieved a sufficiently uniform one.

   Otherwise, it exploits the dynamic component of the environment to its advantage by gathering a portion of resources from those areas that have estimated lambda below the average estimated lambda and redistributing them across the other areas. More precisely, this second

branch, which constitutes the main logic behind the agent's policy, is described in more detail in Algorithm 1 below.

Algorithm 1 Steering Allocation procedure

Input: rounded $\hat{\boldsymbol{\lambda}}^{(t)}$, $SteerFact = 5$

$\mu \leftarrow \mathbb{E}\left[\hat{\boldsymbol{\lambda}}^{(t)}\right]$

$count\_pos \leftarrow \left|\{\hat{\lambda}_i^{(t)} > \mu \mid i = 1...N\}\right|$

$\boldsymbol{\Delta} = [0, ..., 0]$

$\boldsymbol{\Delta}^- \coloneqq [\ \Delta_i \mid \hat{\lambda}_i^{(t)} \leq \mu\ ]$

$\boldsymbol{\Delta}^+ \coloneqq [\ \Delta_i \mid \hat{\lambda}_i^{(t)} > \mu\ ]$

for i s.t. $\hat{\lambda}_i^{(t)} \leq \mu$ **do**

  $\beta_i \leftarrow \tanh\left(2\left|\frac{\hat{\lambda}_i^{(t)} - \mu}{\mu}\right|\right)$ ▷ decay factor

  if $a_i - \lfloor SteerFact \cdot \beta_i \rceil > 1$ **then** ▷ if condition to avoid $a_i < 0$

    $\Delta_i^- \leftarrow \lfloor SteerFact \cdot \beta_i \rceil$

  else

    $\Delta_i^- \leftarrow \max(a_i - 1, 0)$ ▷ s.t. $a_i - \Delta_i^- = 1$ or a_i unchanged if $a_i = 0$

  end if

end for

$redistribution = -\sum_{i.\hat{\lambda}_i^{(t)} \leq \mu} \Delta_i^-$

$\Delta_i^+ \leftarrow \lfloor redistribution / count\_pos \rfloor$

$remainder \leftarrow redistribution \mod count\_pos$ ▷ redistribute remainder randomly

$\boldsymbol{\Delta}^+ \leftarrow \boldsymbol{\Delta}^+ + MultivariateHyperGeometric([1, ..., 1], remainder)$ ▷ max 1 unit each

$\boldsymbol{a} \leftarrow \boldsymbol{a} + \boldsymbol{\Delta}$

return $\boldsymbol{a}$

The hyperparameter $SteerFact$ represents the magnitude of steering that the agent performs at each iteration. β_i makes sure that the steering reduces as we approach a more uniform distribution.

The overall intuition is that it tries to steer the distribution towards one that is easier to exploit for fairness, adopting the view of performance coming from both prediction accuracy and steering component (Perdomo et al., 2021).It settles towards the same behaviour of the Fair Static Agent as the distribution shifts towards a more uniform one, by using the β_i decay factor while steering, and not steering at all once it reaches the *exploiting_range*.

3 Results

The first plots are from the Philadelphia Crime dataset, while the second ones are from the Los Angeles Crime dataset. Accuracy plots (for ensuring performance) are in Appendix C.

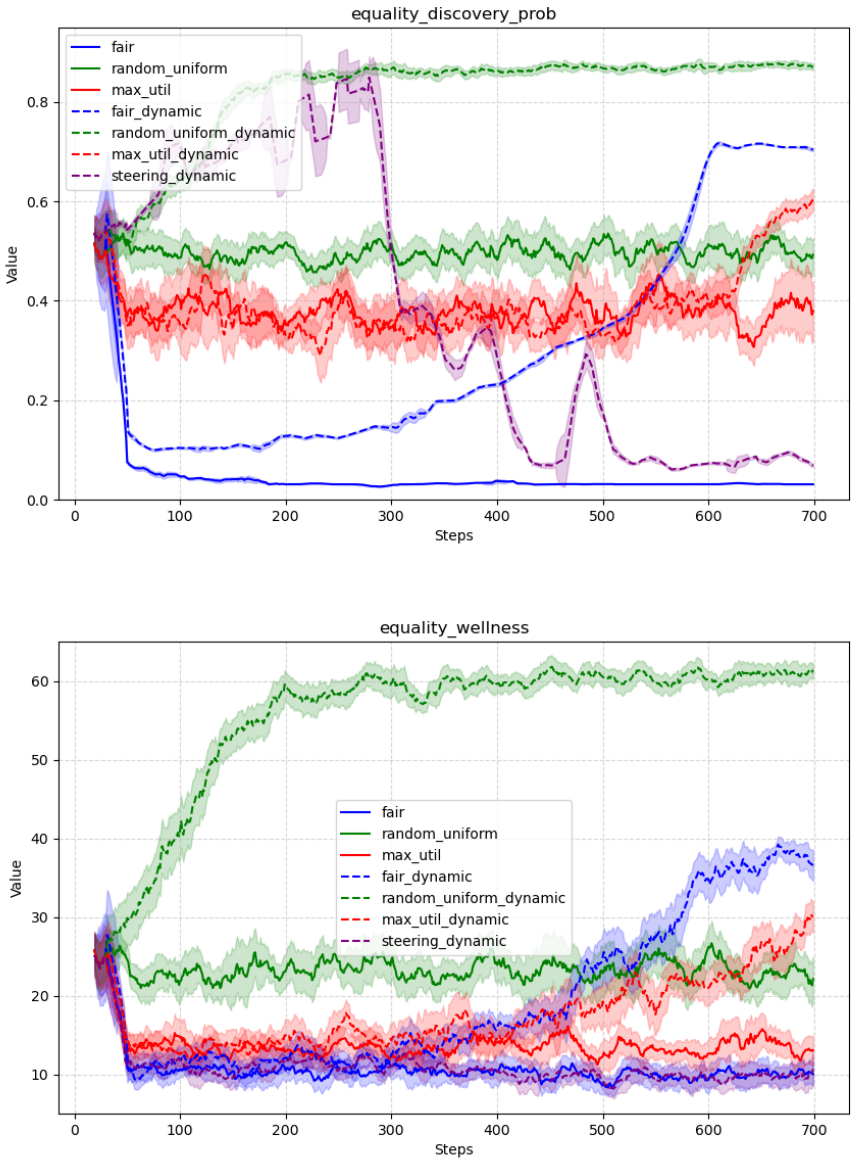

Figure 1: equality of discovery probability and equality of wellness across timesteps for the Philadelphia dataset (lower is better). Continuous lines represent the static environment runs, dashed lines the dynamic environment runs. Major importance should be given to the blue and purple lines, representing the Fair Static and Dynamic agents respectively, while the green line is the baseline and hence the less meaningful one.

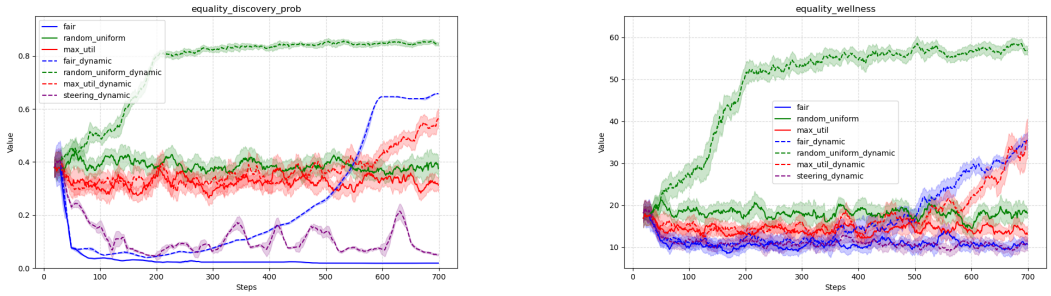

Figure 2: equality of discovery probability and equality of wellness across timesteps for the Los Angeles dataset (lower is better).

4 Discussion

From Figure 1, we can note two important aspects:

- First, by looking at the blue lines, fairness deteriorates over time for the Fair Agent when posed in the dynamic environment, compared to the static one. This happens for both the two fairness metrics, although more drastically in the first one.

  An example scenario in our predictive policing setting would be police departments looking at past data to decide where to allocate patrols, failing to keep up with the changing distribution of crime.

- On the other hand, the proposed Fair Dynamic Agent (purple lines), manages to cope well with the dynamic addition and reaches, over time, fairness levels close to the ones of the Fair Agent in the static environment.

  It does so by steering the distribution towards a more equitable one, at the cost of an initially more unfair behaviour (left part of first plot). Again, through the lens of crime predictions, such behaviour would equate to re-routing, in the first phase, some patrols from less criminal areas to more dangerous ones, underestimating the firsts and overprotecting the seconds. The extra patrols in the more criminal zone would act as a deterrent for future criminal actions, that would eventually move to the areas with lower-than-expected police protection. Over time, this would lead to a more fair distribution of crime and hence of predictions, as can be seen by the metric values obtained.

  In this instance, an unfair behaviour could be potentially justified in the beginning in order to avoid degenerating later on.

If we look at the Los Angeles Crime dataset now, from Figure 2, the results above are reproduced pretty consistently, with the only difference that the Fair Dynamic Agent's performance on the first metric doesn't raise as much initially this time.

Note how the agents don't sacrifice performance when they achieve fairness, as can be seen from Figure 3 (Appendix C).

5 Conclusion

In this paper, we showed how fairness algorithms for allocation problems may suffer when the environment shifts from static to dynamic. For this reason, we proposed an algorithm in order to address this issue.

The idea behind it is that of being "active" towards fairness when making predictions by steering the data distribution towards one that can receive more fair outcomes, instead of being "passive" in the sense of only trying to make predictions to respect a fairness criteria. We believe that this performative approach has a great potential also for other areas in which fairness is a subject of interest.

We want to also highlight the limitations of the designed setting: the update of the distributions in the environment follows a pretty simple equation, which doesn't make the task of steering too difficult for the proposed agent. In real life scenarios, the actions taken could morph the data distributions in more non-linear ways, even breaking the modeling of the data as a specific distribution (Poisson in our case).

Future research could aim at understanding such distributions shifts through data and develop more realistic models of it depending on the domain at hand. With models of the distributions change, we could then employ more complex steering agents, for example taking inspiration from the field of Reinforcement Learning.

6 Links to external resources

- Los Angeles Crime Dataset

References

Hadi Elzayn, Shahin Jabbari, Christopher Jung, Michael Kearns, Seth Neel, Aaron Roth, and Zachary Schutzman. 2018. Fair algorithms for learning in allocation problems.

Juan C. Perdomo, Tijana Zrnic, Celestine Mendler-Dünner, and Moritz Hardt. 2021. Performative prediction.

A Implementation details

The dynamic environment step update is more precisely:

$$\boldsymbol{\Delta} = clip(\boldsymbol{\lambda}^{(t)} - \boldsymbol{a}^{(t)}, -MC, MC)$$

$$S = \sum_i \Delta_i \qquad S^+ = \sum_{i.\Delta_i > 0} \Delta_i$$
$$\Delta_i = \Delta_i - \frac{S}{S^+} \qquad \forall i \text{ s.t. } \Delta_i > 0$$

$$\lambda_i^{(t+1)} = \lambda_i^{(t)} + \delta \cdot \Delta_i \qquad \forall i \in \{1, ..., N\}$$

where MC is a constant defining the maximum number of crimes that can happen in an area at a timestep.

This can be summarized by a rescaling of positive entries of $\boldsymbol{\Delta}$ such that the sum of the elements of $\boldsymbol{\Delta}$ remains 0 and the total amount of crime, as described by lambdas, doesn't change (crimes moves across areas, doesn't disappears).

B Data validation - Los Angeles dataset

area id	l_1	l_∞
1	0.6545	0.035
2	0.3257	0.0234
3	0.3623	0.0274
4	0.2549	0.0213
5	0.2283	0.0207
6	0.3693	0.0258
7	0.3463	0.0231
8	0.2732	0.0222
9	0.2159	0.018
10	0.3078	0.0333
11	0.2938	0.0224
12	0.2994	0.0223
13	0.3889	0.027
14	0.254	0.023
15	0.2026	0.018
16	0.2256	0.0249
17	0.2694	0.023
18	0.2898	0.0204
19	0.2835	0.0252
20	0.3321	0.0216
21	0.2632	0.0256

Table 1: $l1$ and l_∞ between the true crime distribution and the fitted Poisson distribution for each area. The table shows that the Poisson fit provides a good approximation of the ground truth crime distribution, in particular the l_∞ metric.

C Accuracy plots

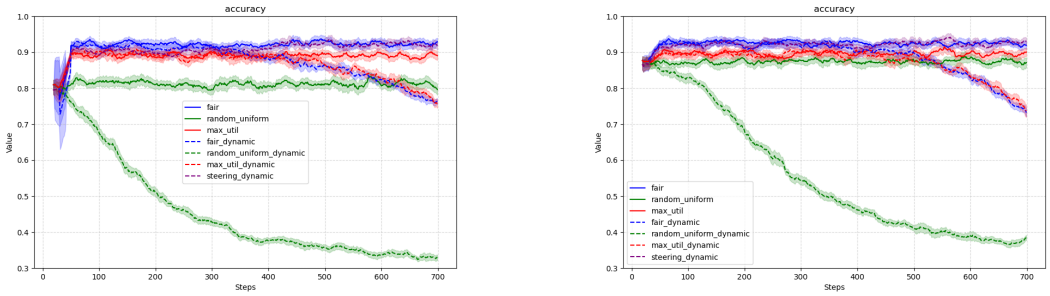

Figure 3: equality of discovery probability and equality of wellness across timesteps for the Los Angeles dataset (lower is better).