Fair Policy Targeting (example first)

JCY

2023 年 2 月 24 日

目录

1 Data and Motivation

We want to studie the effect of an entrepreneurship training and incubation program for undergraduate students in North America on subsequent entrepreneurial activity.

We have in total 335 observations with 20 covariates x, of which 53% treated and the remaining under control, and 26% of applicants are women. The population of interest is the pool of final applicants.

	finalist_id	short_run_activity	ongoing_activity	accepted	minority	female
1	335	0	1	1	1	0
2	29	0	1	0	1	1
3	125	1	0	1	0	0
4	322	1	0	1	1	1
5	142	0	0	0	0	0
6	206	0	0	0	1	0
...						
352	315	0	0	1	1	1
353	265	0	0	0	1	0
354	102	0	0	0	0	0
355	97	0	1	0	1	1
356	342	0	0	0	1	1

1.1 Motivation

We intend to design a policy that assigns students to entrepreneurial programs, while imposing fairness on gender. So we will construct a targeting rule that assigns the award to the finalist based on the applicant' s observable characteristics.

Similar to 'POLICY LEARNING WITH OBSERVATIONAL DATA', we estimate the policy under the assumption that

$$\textbf{(Treatment Unconfoundedness).}\ \text{For } d \in \{0,1\}, Y(d) \perp D \mid X, S. \tag{1}$$

we control for residual confounding through individual level observable characteristics and an observable quality score of the final applicant.

Then we allow covariates x to be the years to graduation, years of entrepreneurship, the region of the start-up, the major, the school rank.

2 Encounter Problem

In order to consider fairness when designing a optimal policy , we need to design fair and efficient targeting rules for applications in welfare.

Fair targeting is a controversial task due to the lack of consensus on the formulation of the decision problem.

Conventional approaches mostly developed in computer science consist in designing algorithmic decisions that maximize the expected utility across all individuals by imposing fairness constraints on the decision space of the policymaker.

In contrast, the economic literature has outlined the importance of taking into account the welfare-effects of such policies. Fairness constraints on the policymaker' s decision space may ultimately lead to sub-optimal welfare for both sensitive groups. This is a significant limitation when the policymakers are concerned with the effects of their decisions on each individual' s utilities: absent of legal constraints, we may not want to impose unnecessary constraints on the policy if such constraints are harmful for some or all individuals.

So how to define fair targeting and how to achieve welfare maximization with fairness become our challenges.

3 Notation

Notation	Explanation
$S \in \mathcal{S} = \{0, 1\}$	sensitive attribute. S = 1 denotes the disadvantaged
$D \in \mathcal{D} = \{0, 1\}$	the treatment
$X \in X \subset \mathbb{R}^p$	individual characteristics
$Y(S, X, D) \in \mathcal{Y} \subset \mathbb{R}$	the post-treatment outcome.
$Y(d), d \in \{0, 1\}$	the potential outcomes under treatment d
$e(x, s)$	$:= P(D = 1 \mid X = x, S = s)$
$p_1 = P(S = 1)$	posterior probability of the disadvantaged group.
$m_{d,s}(x) = \mathbb{E}[Y_i(d) \mid X_i = x, S_i = s]$	the conditional mean of the group s under treatment d

In the example, we can estimate $\hat{p}_1$ and $\hat{e}(x, s)$, $\hat{m}_{d,s}(.)$ via cross-fitting

```
## Estimate propensity score
data <- na.omit(data)
D <- data$accepted
S <- data$female
propensity2 <- mean(S)
Y <- data$short_run_activity

library(glmnet)
propensity1 <- cross_fitting_propensity(D,as.matrix(data[, c(5,6,7,8, 9, 10, 11
    , 12, 13, 14)]), seeds = 123, K = 5)
#propensity1 is e(x,s)
#propensity2 is p(s=1)

X <- data[,c(5,8,9,10,11, 12, 13, 14)]
```

```
X_int <- X[, c(3,4,5,6, 7, 8)]
X_reg <- cbind(X, D, D * X_int * S, D * X_int * (1 - S), S * X_int, D * X_int)
   #选择协变量
conditional_means <- cross_fitting_mean(Y, X_reg, X, X_int, seeds = 123, K = 5)
    #治疗d下组s的条件平均值
#m(d,s)
m_hat11 <- conditional_means[,1]
m_hat10 <- conditional_means[,2]
m_hat01 <- conditional_means[,3]
m_hat00 <- conditional_means[,4]
```

The doubly robust score (Robins and Rotnitzky, 1995).

$$\Gamma_{d,s,i} = \frac{1\{S_i = s\}}{p_s}\left[\frac{1\{D_i = d\}}{e(X_i, S_i)}(Y_i - m_{d,s}(X_i)) + m_{d,s}(X_i)\right] \tag{2}$$

can be estimated by $\hat{\Gamma}_{d,s,i}$ using $\hat{p}_1$ and $\hat{e}(x,s)$, $\hat{m}_{d,s}(.)$.

```
m1 <- S*m_hat11 + (1 - S)*m_hat01
m0 <- S*m_hat10 + (1 - S)*m_hat00

G_i1 <- (Y - m0) * (1-D) * S / ((1-propensity1)*propensity2) + m0 * S/
   propensity2 #D=0,S=1
G_i2 <- (Y - cost_treatment - m1) * D * S/ (propensity1*propensity2) + m1 * S
   /propensity2 #D=1,S=1

G_i12 <- (Y - m0) * (1-D) * (1 - S) / ((1-propensity1)*(1 - propensity2)) + m
   0 * (1 - S)/(1 - propensity2) #D=0,S=0
G_i22 <- (Y - cost_treatment - m1) * D * (1 - S)/ (propensity1*(1 -
   propensity2)) + m1 * (1 - S)/(1 - propensity2) #D=1,S=0
```

3.1 Policy function π

A policy function π is a numerical mapping:

$$\pi : X \times \mathcal{S} \to \mathcal{T} \tag{3}$$

where π is called deterministic when $\mathcal{T} = \{0,1\}$, meanwhile π is called probabilistic when $\mathcal{T} = [0,1]$.

A situation is called Pareto-dominated if there exists a possible Pareto improvement.

3.2 Welfare Definition

The welfare generated by a policy π on those individuals with sensitive attribute S=s is defined as

$$W_s(\pi) = \mathbb{E}[(Y(1) - Y(0))\pi(X,S) \mid S = s]. \tag{4}$$

Define $\hat{W}_s(\pi)$ as:

$$\hat{W}_s(\pi) = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\Gamma}_{1,s,i} - \hat{\Gamma}_{0,s,i}\right)\pi\left(X_i, s\right) \tag{5}$$

```
## Construct the relative welfares
G_i1 <- (Y - m0) * (1-D) * S / ((1-propensity1)*propensity2) + m0 * S/
    propensity2
G_i2 <- (Y - cost_treatment - m1) * D * S/ (propensity1*propensity2) +
    m1 * S/propensity2
g_i_S = G_i2 - G_i1
G_i12 <- (Y - m0) * (1-D) * (1 - S) / ((1-propensity1)*(1 - propensity2
    )) + m0 * (1 - S)/(1 - propensity2)
G_i22 <- (Y - cost_treatment - m1) * D * (1 - S)/ (propensity1*(1 -
    propensity2)) + m1 * (1 - S)/(1 - propensity2)
g_i_S2 = G_i22 - G_i12

component1 <- g_i_S
component2 <- g_i_S2
constant1 <- sum(G_i1)
constant2 <- sum(G_i12)
```

4 Fair Policy Targeting

We use Fair Policy Targeting method to solve this problem.

This method has three desirable properties:

(i) it applies to general notions of fairness which may reflect different decision makers'preferences;

(ii) it guarantees Pareto efficiency of the policy-function, with the relative importance of each group solely chosen based on the notion of fairness adopted by the decision-maker;

(iii) it also allows for arbitrary legal or ethical constraints, incorporating as a special case the presence of fairness constraints whenever such constraints are binding due to ethical or legal considerations.

In this method, we should first define properties of Pareto optimal and fair treatment allocation rules.

4.1 Fairness

We consider three notions of UnFairness:

4.1.1 Counterfactual Envy

Under the assumption that:

$$\text{(A) } Y(d,s) \perp (D,S) \mid X(s),\ \text{(B) } X(s) \perp S \tag{6}$$

Assumption state that the sensitive attribute is independent of potential outcomes and covariates, while it allows for the dependence of observed covariates and outcomes with the sensitive attribute. Let the conditional welfare, for the policy function being assigned to the opposite attribute, i.e., the effect of $\pi(x, s_1)$, on the group s_2 , conditional on covariates, be

$$V_{\pi(x,s_1)}(x, s_2) = \mathbb{E}\left[\pi(x, s_1) Y_i(1, s_2) + (1 - \pi(x, s_1)) Y_i(0, s_2) \mid X_i(s_2) = x\right] \tag{7}$$

We say that the agent with attribute s_2 envies the agent with attribute s_1, if her welfare (on the right-hand side of Equation (8)) exceeds the welfare she would have received had her covariate and policy been assigned the opposite attribute (left-hand side of Equation (8)), namely

$$\mathbb{E}_{X(s_1)}\left[V_{\pi(X(s_1),s_1)}(X(s_1), s_2)\right] > \mathbb{E}_{X(s_2)}\left[V_{\pi(X(s_2),s_2)}(X(s_2), s_2)\right] \tag{8}$$

We then measure the unfairness towards an individual with attribute s_2 as

$$\mathcal{A}(s_1, s_2; \pi) = \mathbb{E}_{X(s_1)}\left[V_{\pi(X(s_1),s_1)}(X(s_1), s_2)\right] - \mathbb{E}_{X(s_2)}\left[V_{\pi(X(s_2),s_2)}(X(s_2), s_2)\right]. \tag{9}$$

Whenever we aim not to discriminate in either direction, we take the sum of the effects $\mathcal{A}(s_1, s_2; \pi)$ and $\mathcal{A}(s_2, s_1; \pi)$ it connects to previous notions of counterfactual fairness (Kilbertus et al., 2017).

The smaller the value $\mathcal{A}(s_1, s_2)$, the fairer it is.

```
component1 <- mu_hat01*S/propensity2 -  mu_hat00*S/propensity2 - g_i_S
   ## policy 1
component2 <- mu_hat11*(1 - S)/(1 - propensity2) -  mu_hat10*(1 - S)/(1
    - propensity2) - g_i_S2 ## policy
```

4.1.2 Predictive Disparity

Prediction disparity and its empirical counterpart take the following form

$$C(\pi) = \mathbb{E}[\pi(X, S) \mid S = 0] - \mathbb{E}[\pi(X, S) \mid S = 1], \quad \hat{C}(\pi) = \frac{\sum_{i=1}^{n} \pi(X_i)(1 - S_i)}{n(1 - \hat{p}_1)} - \frac{\sum_{i=1}^{n} \pi(X_i) S_i}{n\hat{p}_1}, \tag{10}$$

Prediction disparity captures disparity in the treatment probability between groups.

(Welfare disparity). Define the welfare disparity and its empirical counterpart as

$$D(\pi) = W_0(\pi) - W_1(\pi), \quad \widehat{D}(\pi) = \widehat{W}_0(\pi) - \widehat{W}_1(\pi). \tag{11}$$

```
component1 <- S/propensity2
component2 <- (1 - S)/(1 - propensity2)
```

4.1.3 Predictive disparity with absolute value

Predictive disparity with absolute value.

The policymaker may also consider $|D(\pi)|$ or $|C(\pi)|$ as measures of UnFairness, in which case the policymaker treats the two groups symmetrically.

4.2 Pareto optimal

Definition (Pareto frontier)

The set $\Pi_o \subseteq \Pi$ is such that

$$\Pi_0 = \left\{ \pi_\alpha : \pi_\alpha \in \arg\sup_{\pi \in \Pi} \alpha W_1(\pi) + (1-\alpha) W_0(\pi), \quad \alpha \in (0,1) \right\}. \tag{12}$$

Supremum of Pareto frontier for a fixed ratio α :

$$\bar{W}_\alpha = \sup_{\pi \in \Pi} \alpha W_1(\pi) + (1-\alpha) W_0(\pi) \tag{13}$$

4.3 Decision Problem

We can turn the whole question to a decision problem.

Defining $\mathcal{C}(\Pi)$ the choice set of the policy maker, where $\mathcal{C}$ is a choice function with $C(\{\pi_1, \pi_2\}) = \pi_1$ if π_1 is strictly preferred to π_2. We let UnFairness : $\Pi \mapsto \mathbb{R}$

Rational Preferences

For each $\pi_1, \pi_2 \in \Pi$,

(i) $C(\{\pi_1, \pi_2\}) = \pi_1$ if $W_1(\pi_1) \geq W_1(\pi_2)$ and $W_0(\pi_1) \geq W_0(\pi_2)$ and either (or both) of the two inequalities hold strictly;

(ii) if neither π_1 Pareto dominates π_2 nor π_2 Pareto dominates $\pi_1, \mathcal{C}(\{\pi_1, \pi_2\}) = \pi_1$ if UnFairness (π_1) < UnFairness (π_2) ;

(iii) if neither Pareto dominates the other and with equal UnFairness, $C(\{\pi_1, \pi_2\}) = \{\pi_1, \pi_2\}$

Proposition (Decision Problem). Under Rational Preferences, $\pi^\star \in \mathcal{C}(\Pi)$ if and only if

$$\begin{gathered} \pi^\star \in \arg\inf_{\pi \in \Pi} \text{UnFairness}(\pi) \\ \text{subject to } \alpha W_1(\pi) + (1-\alpha) W_0(\pi) \geq \bar{W}_\alpha, \text{ for some } \alpha \in (0,1) \end{gathered} \tag{14}$$

Proposition formally characterizes the policymakers decision problem, which consists of minimizing the policy's unfairness criterion, under the condition that the policy is Pareto optimal. The policy-maker does not maximize a weighted combination of welfares, with some pre-specified and hard-to-justify weights. Instead, each group's importance (i.e., α) is implicitly chosen within the optimization problem to maximize fairness.

This approach allows for a transparent choice of the policy based on the policy-makers definition of fairness.

5 Fair Targeting: Estimation

Define $\mathcal{V}_n(\pi, p_s, e, m)$ an unbiased estimate of UnFairness (π) which depends on observables and the population propensity score and conditional mean. We write $\widehat{\mathcal{V}}_n(\pi) = \mathcal{V}_n(\pi, \hat{p}_s, \hat{e}, \hat{m})$, the empirical counterpart.

5.1 (Approximate) Pareto Optimality

We characterize the Pareto frontier using linear inequalities. To construct the Pareto frontier we use the constraint in Equation (14) after discretizing the set of weights α .

(1) Discretize the Pareto frontier, and construct a grid of equally spaced values $\alpha_j \in (0,1), j \in \{1,\ldots,N\}$, with $N = \sqrt{n}$.

$$\widehat{\Pi}_o = \left\{ \pi_\alpha \in \Pi : \pi_\alpha \in \arg\sup_{\pi\in\Pi} \left\{ \alpha \hat{W}_0(\pi) + (1-\alpha)\hat{W}_1(\pi) \right\}, \text{ s.t. } \alpha \in \{\alpha_1,\ldots,\alpha_N\} \right\}. \tag{15}$$

The grid' s choice is arbitrary, as long as values are equally spaced.

(2)Find the largest empirical welfare achieved on the discretized Pareto Frontier defined as

$$\bar{W}_{j,n} = \sup_{\pi\in\Pi} \left\{ \alpha_j \hat{W}_0(\pi) + (1-\alpha_j)\, \hat{W}_1(\pi) \right\}, \text{ for each } j \in \{1,\ldots,N\}, \tag{16}$$

which can be obtained through standard optimization routines.

(3)Construct an approximate Pareto frontier as follows:

$$\widehat{\Pi}_o(\lambda) = \{\pi \in \Pi : \exists j \in \{1,\ldots,N\} \text{such that } \alpha_j \hat{W}_{0,n}(\pi) + (1-\alpha_j)\, \hat{W}_{1,n}(\pi) \geq \bar{W}_{j,n} - \frac{\lambda}{\sqrt{n}} \Big\} \tag{17}$$

where $\widehat{\Pi}_o(0) = \widehat{\Pi}_o$, and $\widehat{\Pi}_o \subseteq \widehat{\Pi}_o(\lambda)$ for any $\lambda \geq 0$

(4) The estimated policy is defined as

$$\hat{\pi}_\lambda \in \arg\min_{\pi\in\widehat{\Pi}_o(\lambda)} \widehat{V}_n(\pi) \tag{18}$$

5.2 Optimization: Mixed Integer Quadratic Program

We provide a mixed-integer quadratic program (MIQP) for optimization.

Define $\mathbf{z}_s = (z_{s,1},\cdots,z_{s,n}), z_{s,i} = \pi(X_i, s), \pi \in \Pi$. Here, $z_{s,n}$ defines the treatment assignment under policy π and sensititive attribute s;

The vector $\mathbf{u} = (u_1,\cdots,u_N) \in \{0,1\}^N$ encodes the locations on the grid of α for which the supremum in $\widehat{\Pi}_o(\lambda)$ is reached at.

Example (Maximum score) For the maximum score $\pi(X_i, s) = 1\{X_i\beta_x + S\mu \geq 0\}, \beta = (\beta_x, \mu) \in \mathcal{B}$, the indicators $z_{s,n}$ are defined via mixed-integer constraints:

$$\frac{X_i^\top \beta + s\mu}{|C_i|} < z_{s,i} \leq \frac{X_i^\top \beta + s\mu}{|C_i|} + 1, C_i \geq \sup_{\beta\in\mathcal{B}} \left| (X_i, S_i)^\top \beta \right|, z_{s,i} \in \{0,1\} \tag{19}$$

Such constraint guarantees that $z_{s,i} = 1\left\{X_i^\top \beta_x + s\mu \geq 0\right\}$.

$\hat{\pi}_\lambda$ satisfies Equation(18)

$$\begin{aligned}
\text{subject to} \quad & (A) \quad z_{s,i} = \pi(X_i, s), \quad 1 \leq i \leq n, \\
& (B) \quad u_j \alpha_j \left\langle \hat{\mathbf{\Gamma}}_{1,0} - \hat{\mathbf{\Gamma}}_{0,0}, \boldsymbol{z}_0 \right\rangle + u_j (1-\alpha_j) \left\langle \hat{\mathbf{\Gamma}}_{1,1} - \hat{\mathbf{\Gamma}}_{0,1}, \boldsymbol{z}_1 \right\rangle \geq u_j n \bar{W}_{j,n} - \sqrt{n}\lambda \\
& (C) \quad \langle \mathbf{1}, \mathbf{u} \rangle \geq 1 \\
& (D) \quad \pi \in \Pi \\
& (E) \quad u_j \in \{0,1\}, \quad 1 \leq j \leq N.
\end{aligned} \tag{20}$$

- **B,C**: state that the resulting policy is (approximately) Pareto optimal (it maximizes a weighted combination of groups' welfare for some α_j: $\alpha_j \hat{W}_{0,n}(\pi) + (1-\alpha_j)\hat{W}_{1,n}(\pi) \geq \bar{W}_{j,n} - \frac{\lambda}{\sqrt{n}}$); To ensure that the chosen policy is Pareto optimal(at least one $u_j = 1$), impose the constraint $\sum_{j=1}^{n} u_j \geq 1$
- **A,C,E**: (mixed-integer) linear constraints; **B** is quadratic
- **D**: either linear or quadratic for deterministic assignments and linear probability models
- the solution to the optimization problem might not be unique, depending on the function class.

Algorithm:

```
## The function initialize the quadratic program

## Inputs: Y outcome
##         X : covariates for targeting, the first entry is assumed to
   be the sensity attribute
##         D: treatment assignment
##         S: sensitive attribute
##         propensity1: probability of treatment
##         propensity2 : probability of sensitive attribute (note: this
    are vector with the predict probs with n entries)
##         B: coefficients upper bounds
##         params: parameters of the program
##         tolerance_constraint: tolerance coefficient for the MILP
   program
##         scale_Y: whethere Y is rescaled(default F)
##         cost_treatment (default 0)
##         alpha: importance weights (depracated for this function)
##         g_i1: objective for the sensitive group
##         g_i2: objective for the opposite group
##         max_treated units: maximum number of treated individuals
##         maxtime: maximum time
##         m1, m0: predicted conditional means
##         no_parity_constraint: whether an additional constraint on
   the welfare is included or not (use or not use S for prediction?)
##         distance: unfairness measure
##         constant1, constant2: offset constants which depend on the
   unfairness measure
##         two_directions: only active if distance != envy. If T it
   takes fairness as absolute value, otherwise it takes the
##                         difference between the priviledge and
   sensitive group

## return: gurobi model
```

```
create_model_quadratic_program <- function(Y, X, D, S, propensity1,
    propensity2,B=1, params = NA,
tolerance_constraint = 10**(-7),  scale_Y = F,
cost_treatment  = 0, alpha = 1/2, g_i1, g_i2, max_treated_units,
maxtime = 300, m1 = 0, m0 = 0,
no_parity_constraint = F,
distance = 'envy', constant1 = NA,constant2 = NA,
two_directions = T) model  <- list()
## Specify the constraints
## Consider the vector of (z_1, ..., z_n,beta_0 , beta_1, ..., beta_p)
if(distance == 'envy'){
        model$obj <- c(g_i1, g_i2, rep(0, p + 1))
        model$Q <- diag(c(rep(0, n), rep(0, n), rep(0, p + 1)))
        model$A<- rbind(cbind(diag(1, nrow = 2*n), -XX), cbind(diag(1,
            nrow = 2*n), -XX),
        c(rep(1,n)*S, rep(1, n)*(1 - S), rep(0, p +1)))
        model$modelsense<-'min'
        model$rhs<- c(rep(1 - tolerance_constraint, dim(model$A)[1]/2),
        rep(tolerance_constraint, dim(model$A)[1]/2), max_treated_units
            )
        model$sense<- c(rep('<=', dim(model$A)[1]/2), rep('>', dim(
            model$A)[1]/2), '<=')
        model$vtype<- c(rep('B', 2*n), rep('C', p+1))
        model$ub<- c(rep(1,2*n), rep(B,1+p))
        model$lb<- c(rep(0,2*n), rep(-B,p + 1))
}
if(distance == 'welfare' | distance == 'relative_welfare'){
        # Welfare program has two more variables to guarantee that the
            absolute value in the objective
        # function holds
        # the last two entries are two binary variables one indicating
            W_1 - W_0 >= 0, the other W_0 - W_1 >= 0
        ## note: if distance was specified as relative welfare than
            automatically these constants terms are zero

        model$obj <- c(rep(0, 2*n + p + 1), constant1 - constant2,
            constant2 - constant1)
        Q_matrix <- matrix(0, nrow= 2*n + p + 3, ncol = 2*n + p + 3)
        Q_matrix[1:n, 2 * n + p + 2] <- g_i1
        Q_matrix[(n + 1):(2*n), 2 * n + p + 2] <- -g_i2

        Q_matrix[1:n, 2 * n + p + 3] <- -g_i1
        Q_matrix[(n + 1):(2*n), 2 * n + p + 3] <- g_i2
```

```
        model$A<- rbind(cbind(diag(1, nrow = 2*n), -XX, 0, 0),
        cbind(diag(1, nrow = 2*n), -XX, 0, 0),
        c(rep(1,n)*S, rep(1, n)*(1 - S), rep(0, p +3)))
        model$Q <- Q_matrix

        model$modelsense<-'min'
        ## tolerance_constraint enters here
        model$rhs<- c(rep(1 - tolerance_constraint, dim(model$A)[1]/2),
        rep(tolerance_constraint, dim(model$A)[1]/2),
        max_treated_units)
        model$sense<- c(rep('<=', dim(model$A)[1]/2), rep('>', dim(
            model$A)[1]/2), '<=')
        model$vtype<- c(rep('B', 2*n), rep('C', p+1), rep('B', 2))
        # Put bounds on the parameter space (If commented, parameter
            space = real line)
        model$ub<- c(rep(1,2*n), rep(B,1+p), rep(1, 2))
        model$lb<- c(rep(0,2*n), rep(-B,p + 1), rep(0, 2))

        ## Objective if do not use absolute value for unfairness
        saved_objective_one_direction <- c(-g_i1, g_i2, rep(0, 3 + p))
}

if(distance == 'parity'){

        ## constant1, constant2 here should be 0s
        model$obj <- c(rep(0, 2*n + p + 1), 0, 0)
        Q_matrix <- matrix(0, nrow= 2*n + p + 3, ncol = 2*n + p + 3)
        Q_matrix[1:n, 2 * n + p + 2] <- S/mean(S)
        Q_matrix[(n + 1):(2*n), 2 * n + p + 2] <- -(1 - S)/(1 - mean(S)
            )

        Q_matrix[1:n, 2 * n + p + 3] <- -S/mean(S)
        Q_matrix[(n + 1):(2*n), 2 * n + p + 3] <- (1 - S)/(1 - mean(S))

        model$A<- rbind(cbind(diag(1, nrow = 2*n), -XX, 0, 0),
        cbind(diag(1, nrow = 2*n), -XX, 0, 0),
        c(rep(1,n)*S, rep(1, n)*(1 - S), rep(0, p +3)))
        model$Q <- Q_matrix

        model$modelsense<-'min'
        ## tolerance_constraint enters here
        model$rhs<- c(rep(1 - tolerance_constraint, dim(model$A)[1]/2),
```

```
                rep(tolerance_constraint, dim(model$A)[1]/2), max_treated_units
                    )
                model$sense<- c(rep('<=', dim(model$A)[1]/2), rep('>', dim(
                    model$A)[1]/2), '<=')
                model$vtype<- c(rep('B', 2*n), rep('C', p+1), rep('B', 2))
                # Put bounds on the parameter space (If commented, parameter
                    space = real line)
                model$ub<- c(rep(1,2*n), rep(B,1+p), rep(1, 2))
                model$lb<- c(rep(0,2*n), rep(-B,p + 1), rep(0, 2))
                saved_objective_one_direction <- c(-S/mean(S),(1 - S)/(1 - mean
                    (S)), rep(0, 3 + p))
        }
        ## Case where we take differences without absolute values in the
            objective (unfairness)
        if(two_directions == F & distance != 'envy') {
                model$obj <- saved_objective_one_direction
        }
```

5.3 Estimate the policy

The function that displays the detailed policy and β through Mixed Integer Quadratic Program is

```
## Wrapper function for MILP
Est_objective_estimandMaxscore <- function(Y, X, D, S, propensity1, propensity2
    , scale_Y = T,
discretization = floor(sqrt(length(Y))),cost_treatment = 0, params=NA,
mu_hat11, mu_hat01, mu_hat00, mu_hat10, model_only = F,
max_treated_units = length(Y),
maxtime1 = 300, maxtime2 = 100,
alpha_seq = seq(from = 0, to = 1, length = discretization),
m1 = 0, m0 = 0, quick_run = F, no_parity_constraint = F,
additional_fairness_constraint = F,
parity_constraint = '>=', frontier = NA, unique_values = 1 - no_parity_
    constraint,
distance = 'envy', probabilistic = F, numcores = 10, threshold_probabilistic =
    F,
two_directions = T, parallel = T, tolerance_frontier = 10**(-3), tolerance_
    optimization = 10**(-6),
return_frontier = F){
        library(gurobi)
        if(is.na(frontier)[1]){
```

```
            frontier <- estimate_Pareto_frontier(Y = Y, X = X, D = D, S = S
                , propensity1= propensity1, propensity2 = propensity2,
            scale_Y = scale_Y,  discretization = discretization,
            cost_treatment = cost_treatment,
            params = params, max_treated_units = max_treated_units,
            maxtime = maxtime2, alpha_seq = alpha_seq, m1 = m1, m0 = m0,
            additional_fairness_constraint = additional_fairness_constraint
                ,
            parity_constraint = parity_constraint, probabilistic =
                probabilistic,
            numcores = numcores, threshold_probabilistic = threshold_
                probabilistic,
            parallel = parallel, tolerance = tolerance_frontier)

    }
    if(return_frontier) return(frontier)
    frontier_objective <- frontier[[2]]
    results_frontier <- frontier[[4]] ## Store the policies
    results_frontier_collapsed <- frontier[[3]]

    G_i1 <- (Y - m0) * (1-D) * S / ((1-propensity1)*propensity2) + m0 * S/
        propensity2
    G_i2 <- (Y - cost_treatment - m1) * D * S/ (propensity1*propensity2) +
        m1 * S/propensity2
    g_i_S = G_i2 - G_i1

    G_i12 <- (Y - m0) * (1-D) * (1 - S) / ((1-propensity1)*(1 - propensity2
        )) + m0 * (1 - S)/(1 - propensity2)
    G_i22 <- (Y - cost_treatment - m1) * D * (1 - S)/ (propensity1*(1 -
        propensity2)) + m1 * (1 - S)/(1 - propensity2)
    g_i_S2 = G_i22 - G_i12
    #G为gama g为算W的东西

    all_g_i = g_i_S + g_i_S2 ## Save the welfare criterion

    ## Do not consider a probabilistic threshold here
    if(threshold_probabilistic == F){
            beta <- frontier[[5]]
            if(unique_values == F){
                    XX1 <- as.matrix(cbind(1, 1, X[,-1]))
                    XX0 <- as.matrix(cbind(1, 0, X[,-1]))
            } else {
                    XX1 <- as.matrix(cbind(1, X))
```

```
                        XX0 <- as.matrix(cbind(1, X))
                }

        result <- Est_fairnessMaxScore(Y, X, D, S, propensity1 = propensity1, p
            _s = propensity2,  scale_Y,
        discretization, cost_treatment,
        params, frontier_objective = frontier_objective, mu_hat11 = mu_hat11,
        mu_hat01 = mu_hat01, mu_hat00 = mu_hat00, mu_hat10 = mu_hat10,
        all_g_i, max_treated_units = max_treated_units, maxtime = maxtime1,
        warm_start = warm_start, alpha_seq = alpha_seq, noparity_constraint =
            no_parity_constraint,
        additional_fairness_constraint = additional_fairness_constraint,
        unique_values = unique_values,
        distance = distance, m0 = m0, m1 = m1, probabilistic = probabilistic,
        numcores = numcores, two_directions = two_directions, tolerance =
            tolerance_optimization)
}


## Note: equity is for the full welfare function (no relative improvement)
return(list(result = result, frontier = frontier))

}
```

6 Setting

In the example, Consider linear decision rules, given their large use in economics

$$\Pi = \left\{ \pi(x,\ \text{fem}\ ) = 1\left\{ \beta_0 + \beta_1\ \text{fem}\ + x^\top \phi \geq 0 \right\},\quad (\beta_0, \beta_1, \phi) \in \mathcal{B} \right\}. \tag{21}$$

We allow covariates x to be the years to graduation, years of entrepreneurship, the region of the start-up, the major, the school rank.

We consider in-sample capacity constraints imposed on the function class with at most 150 individuals selected for the treatment.

Consider three nested function classes for the welfare maximization method.

The first does not impose any restriction except for the functional form.

The second, imposes that $\beta_1 = 0$.

The third class imposes that $\beta_1 = 0$ and that the average effect of the policy on females is at least as large as the one on males. The function classes are

$$
\begin{aligned}
\Pi_1 &= \left\{\pi(x,\ \text{fem}\ ) = 1\left\{\beta_0 + \beta_1\ \text{fem}\ + x^\top\phi \geq 0\right\},\quad (\beta_0,\beta_1,\phi) \in \mathcal{B}\right\},\\
\Pi_2 &= \left\{\pi(x) = 1\left\{\beta_0 + x^\top\phi \geq 0\right\}\right\},\\
\Pi_3 &= \left\{\pi(x) = 1\left\{\beta_0 + x^\top\phi \geq 0\right\},\quad \mathbb{E}_n\left[\left(Y_i(1) - Y_i(0)\right)\pi\left(X_i\right) \mid S = 1\right] \geq \mathbb{E}_n\left[\left(Y_i(1) - Y_i(0)\right)\pi\left(X_i\right) \mid S = 0\right]\right\},
\end{aligned}
\tag{22}
$$

where $\mathbb{E}_n[\cdot]$ denote the empirical expectation, estimated using the doubly-robust method.

7 Result

Using Fair Policy Targeting method in the example

```
#envy
alpha_seq = seq(from = 0.05, to = 0.95, length = discretization)
## Estimate on both sides the Pareto frontier
res_ms3_envy <- Est_objective_estimandMaxscore(Y = Y, X = cbind(S, X[, c(3:8)])
   , D = D, S = S, propensity1 = propensity1, propensity2 = propensity2,
scale_Y = F,
discretization = discretization,
cost_treatment = 0, params=NA, mu_hat11 = m_hat11, mu_hat01 = m_hat01 ,
mu_hat00 = m_hat00, mu_hat10 = m_hat10, maxtime1 = 5000, maxtime2 = 5000,
max_treated_units = 150,
alpha_seq = seq(from = 0.05, to = 0.95, length = discretization), m1 = m1 , m0
   = m0,
quick_run = F,
no_parity_constraint = T,
additional_fairness_constraint = F,
parity_constraint = '>=', distance = 'envy')

#parity
res_ms3_parity <- Est_objective_estimandMaxscore(Y = Y, X = cbind(S, X[, c(3:8)
   ]), D = D, S = S, propensity1 = propensity1, propensity2 = propensity2,
scale_Y = F,
discretization = discretization,
cost_treatment = 0, params=NA, mu_hat11 = m_hat11, mu_hat01 = m_hat01 ,
mu_hat00 = m_hat00, mu_hat10 = m_hat10, maxtime1 = 20000, maxtime2 = 20000,
max_treated_units = 150,
alpha_seq = seq(from = 0.05, to = 0.95, length = discretization), m1 = m1 , m0
   = m0,
quick_run = F,
no_parity_constraint = T,
additional_fairness_constraint = F,
parity_constraint = '>=', distance = 'parity',
two_directions = F)
```

```
#abs_parity
res_ms3_parity <- Est_objective_estimandMaxscore(Y = Y, X = cbind(S, X[, c(3:8)
   ]), D = D, S = S, propensity1 = propensity1, propensity2 = propensity2,
scale_Y = F,
discretization = discretization,
cost_treatment = 0, params=NA, mu_hat11 = m_hat11, mu_hat01 = m_hat01 ,
mu_hat00 = m_hat00, mu_hat10 = m_hat10, maxtime1 = 20000, maxtime2 = 20000,
max_treated_units = 150,
alpha_seq = seq(from = 0.05, to = 0.95, length = discretization), m1 = m1 , m0
   = m0,
quick_run = F,
no_parity_constraint = T,
additional_fairness_constraint = F,
parity_constraint = '>=', distance = 'parity',
two_directions = T)

#Three nested function classes for the welfare maximization method

res0_EWM_C1 <- Est_max_score(Y,cbind(S, X[,c(3:8)]), D, S = S, propensity1,
   propensity2,B=1, params = NA, model_only=FALSE, tolerance_constraint = 10
   **(-6),  scale_Y = F,
cost_treatment  = 0, alpha = mean(S), g_i = NA, max_treated_units = 150,
   maxtime = 50000, m1 = m1 , m0 = m0, cores = 12)

res02_EWM_C1 <- Est_max_score(Y, X[,c(3:8)], D, S = S, propensity1, propensity2
   ,B=1, params = NA, model_only=FALSE, tolerance_constraint = 10**(-6),  scale
   _Y = F,
cost_treatment  = 0, alpha = mean(S), g_i = NA, max_treated_units = 150,
   maxtime = 20000, m1 = m1 , m0 = m0, cores = 10)

res03_EWM_C1 <- Est_max_score(Y, X[,c(3:8)], D, S = S, propensity1, propensity2
   ,B=1, params = NA, model_only=FALSE, tolerance_constraint = 10**(-6),  scale
   _Y = F,
cost_treatment  = 0, alpha = mean(S), g_i = NA, max_treated_units = 150,
   maxtime = 20000, m1 = m1 , m0 = m0,
additional_fairness_constraint = T, cores = 10)
```

表 1: Policy Distribution

	Gender	Original treatment	Fair Envy	FTP Pred	FTP Pred Abs	Welfare Max. 1	Welfare Max. 2	Welfare Max. 3
1	0	1	0	0	0	0	1	1
2	1	0	1	1	1	1	0	0
3	0	1	1	0	0	1	0	0
4	1	1	1	1	1	1	1	1
5	0	0	1	0	0	1	0	0
6	0	0	1	0	0	1	0	0
...								
350	0	0	1	0	1	1	0	1
351	1	0	0	1	1	0	0	0
353	0	0	1	1	1	1	1	1
354	0	0	0	0	1	0	1	1
355	1	0	0	0	1	0	1	1
356	1	0	0	0	0	0	0	0

Table 1 shows the different policy distribution under different methods that maximize different Un-Fairness measures.

Through Table 1, we can calculate that the policy allocation modified by different methods is about 50% different from the original policy allocation.

```
## Case 1: 3:8 as additional covariates
#不同定义下的各个人的01分配
C1.pi=cbind(S,D,res_ms3_envy$result$policies,res_ms3_parity$result$
    policies,res_ms3_parity_abs$result$policies,res0_EWM_C1$pi,res02_EWM
    _C1$pi,res03_EWM_C1$pi)
C1.pi=as.data.frame(C1.pi)
C1.pi.table= rbind(head(C1.pi),tail(C1.pi))
colnames(C1.pi.table) <- c('Gender','Original treatment','Fair Envy', '
    FTP Pred', 'FTP Pred Abs', 'Welfare Max. 1 ',
'Welfare Max. 2', 'Welfare Max. 3')
C1.pi.table=as.matrix(C1.pi.table)
stargazer(C1.pi.table)
```

7.1 Pareto frontier over each function class

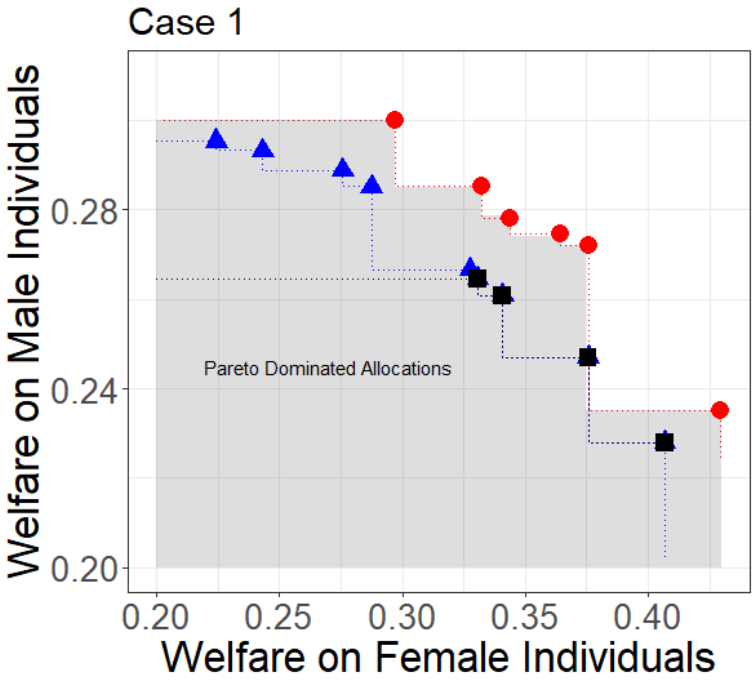

图 1: (Discretized) Pareto frontier under deterministic linear policy rule estimated through MIQP. Dots denote Pareto optimal allocations. Red dots (circle) correspond to Γ_1, blue dots (triangle) to Γ_2 and black dots (square) to Γ_3

The figure shows that restricting the function class leads to Pareto-dominated allocations. This outlines the lim- itations of maximizing welfare under fairness constraints: such constraints can be harmful for both groups. Instead, the proposed method enforces Pareto optimality in the least constrained environment (red line) and selects the policy based on fairness considerations.

```
    g_i <- (Y  - m1)* D/propensity1 - (1 - D) * (Y - m0)/(1 - propensity1)
       + m1 - m0
    baseline_effect <- (1 - D) * (Y - m0)/(1 - propensity1) + m0
    #### Compute welfare on the frontier
    welf1 <- apply(res0B1$frontier$beta, 1, function(x) mean(sapply(cbind(1
       , 1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*S
       /propensity2 + baseline_effect*S/propensity2  ))
    welf0 <- apply(res0B1$frontier$beta, 1, function(x) mean(sapply(cbind(1
       , 0, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*(
       1 - S)/(1 - propensity2) + baseline_effect*(1 - S)/(1 - propensity2)
        ))

    welf <- cbind(welf1, welf0)
    ## Recompute welfare in the other direction
```

```
welf1 <- apply(res0B2$frontier$beta, 1, function(x) mean(sapply(cbind(1
    , 1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*S
    /propensity2 + baseline_effect*S/propensity2))
welf0 <- apply(res0B2$frontier$beta, 1, function(x) mean(sapply(cbind(1
    , 0, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*(
    1 - S)/(1 - propensity2) + baseline_effect*(1 - S)/(1 - propensity2)
    ))

welf2 <- cbind(welf1, welf0)
## Put these two together
welf <- compute_pareto_frontier(welf, welf2)
welf <- rbind(c(0.2, max(welf[,2])), welf, c(max(welf[,1]), 0.2))


g_i <- (Y  - m1)* D/propensity1 - (1 - D) * (Y - m0)/(1 - propensity1)
    + m1 - m0

welf1 <- apply(res02B1$frontier$beta, 1, function(x) mean(sapply(cbind(
    1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*S/
    propensity2 + baseline_effect*S/propensity2))
welf0 <- apply(res02B1$frontier$beta, 1, function(x) mean(sapply(cbind(
    1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*(1
    - S)/(1 - propensity2) + baseline_effect*(1 - S)/(1 - propensity2)))
welf2 <- cbind(welf1, welf0)

welf1 <- apply(res02B2$frontier$beta, 1, function(x) mean(sapply(cbind(
    1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*S/
    propensity2 + baseline_effect*S/propensity2))
welf0 <- apply(res02B2$frontier$beta, 1, function(x) mean(sapply(cbind(
    1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*(1
    - S)/(1 - propensity2) + baseline_effect*(1 - S)/(1 - propensity2)))
welf3 <- cbind(welf1, welf0)
welf2 <- compute_pareto_frontier(welf2, welf3)
welf2 <- rbind(c(0.2, max(welf2[,2])), welf2, c(max(welf2[,1]), 0.2))

## Consider welf3 using only one type of constraints
welf1 <- apply(res02B1$frontier$beta, 1, function(x) mean(sapply(cbind(
    1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*S/
    propensity2 + baseline_effect*S/propensity2))
welf0 <- apply(res02B1$frontier$beta, 1, function(x) mean(sapply(cbind(
    1, as.matrix(X[,c(3:8)]))%*%x, function(y) ifelse(y< 0,0,1))*g_i*(1
    - S)/(1 - propensity2) + baseline_effect*(1 - S)/(1 - propensity2) )
    )
```

```
welf3 <- cbind(welf1, welf0)
welf3 <- rbind(c(0.2, max(welf3[,2])), welf3, c(max(welf3[,1]), 0.2))



type_L <- c(rep('Type 1', length(welf)/2), rep('Type 2', 2*
   discretization + 2), rep('Type 3', discretization + 2))
dd <- as.data.frame(cbind(rbind(welf, welf2, welf3), type_L))
dd[,1] <- as.numeric(as.character(dd[,1]))
dd[,2] <- as.numeric(as.character(dd[,2]))
names(dd) <- c('Welf_fem', 'Welf_mal', 'Type')
plot2 <-  ggplot() +
theme_bw() +
xlab('Welfare on Female Individuals') +
ylab('Welfare on Male Individuals') +
geom_step(data=dd, mapping=aes(x=Welf_fem, y=Welf_mal, color = Type),
   direction="vh", linetype=3) +
#scale_color_manual(values=c("blue", "red")) +
annotate("rect", xmin = 0.2, xmax = 0.2, ymin = 0.2, ymax = 0.319,
alpha = .2) +
annotate("rect", xmin = 0.2, xmax = 0.2, ymin = 0.2, ymax =0.319,
alpha = .2) +
annotate("rect", xmin = 0.2, xmax = 0.297, ymin = 0.2, ymax = 0.3,
alpha = .2) +
annotate("rect", xmin = 0.297, xmax = 0.332, ymin = 0.2, ymax = 0.285,
alpha = .2) +
annotate("rect", xmin = 0.332, xmax = 0.3445, ymin = 0.2, ymax = 0.2785
   ,
alpha = .2) +
annotate("rect", xmin = 0.34445, xmax = 0.365, ymin = 0.2, ymax = 0.272
   ,
alpha = .2) +
annotate("rect", xmin = 0.34445, xmax = 0.365, ymin = 0.272, ymax = 0.2
   74,
alpha = .2) +
annotate("rect", xmin = 0.365, xmax = 0.375, ymin = 0.2, ymax = 0.272,
alpha = .2) +
annotate("rect", xmin = 0.375, xmax = 0.43, ymin = 0.2, ymax = 0.235,
alpha = .2) +
annotate("text", x = 0.27, y = 0.245, label = "Pareto Dominated
   Allocations") +
ylim(c(0.2, 0.31)) +
xlim(c(0.2, 0.43)) +
```

```
        theme(legend.position = "none", axis.title=element_text(size = 25),
            legend.text=element_text(size = 25),
        plot.title = element_text(size=22),
        axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20)) +
        geom_point(data=dd[-c(1, nrow(welf), nrow(welf) + 1, nrow(welf) + nrow(
            welf2), nrow(welf) + nrow(welf2) + 1, dim(dd)[1]),], mapping=aes(x=
            Welf_fem, y=Welf_mal, color = Type,
        shape = Type,  size = 2)) +
        ggtitle("Case 1") + scale_color_manual(values=c("red", "blue", "black")
            )
```

7.2 The welfare improvement and the importance weights assigned by differen methods

表 2: FTP Envy refers to the Fair Targeting rule that minimizes envy-freeness unfairness; FTP Predictive Disp refers to the Pareto allocation that minimizes the difference in probability of treatment (Abs indicate in absolute value); Welfare Max.1 denotes the method that maximizes the empirical welfare considering Π_1, and similarly Welfare Max.2,3 for the function classes, respectively Π_2, Π_3.

	Welf Fem	Welf Mal	Weight
Fair Envy	0.376	0.272	0.384
FTP Pred	0.432	0.224	0.847
FTP Pred Abs	0.433	0.208	0.924
Welfare Max. 1	0.376	0.272	0.266
Welfare Max. 2	0.288	0.285	0.266
Welfare Max. 3	0.331	0.265	0.266

We collect results of the welfare on female and male students, as well as the relative importance weight assigned to each group for methods that maximize different UnFairness measures in Table 2.

The table shows that the proposed method finds importance weights assigned to each group solely based on the notion of fairness provided, without requiring any prior specification of relative weights assigned to each group. The method that maximizes the empirical welfare instead assigns to the sensitive group the importance weight equal to its corresponding probability, small for minorities. In two settings only, the results coincide with the proposed method due to the discreteness of the frontier.

```
        #envy
```

```
welf1_envy_C1 <- mean(sapply(cbind(1, 1, as.matrix(X[,c(3:8)]))%*%beta_
    envy_C1, function(y) ifelse(y< 0,0,1))*g_i*S/propensity2 + baseline_
    effect*S/propensity2 )
welf0_envy_C1 <- mean(sapply(cbind(1, 0, as.matrix(X[,c(3:8)]))%*%beta_
    envy_C1, function(y) ifelse(y< 0,0,1))*g_i*(1 - S)/(1 - propensity2)
     + baseline_effect*(1 - S)/(1 - propensity2 ))

#parity
beta_parity_C1 <- res_ms3_parity$result$beta
weight_parity_C1 <- res_ms3_parity$result$alpha
## Compute welfare
welf1_parity_C1 <- mean(sapply(cbind(1, 1, as.matrix(X[,c(3:8)]))%*%
    beta_parity_C1, function(y) ifelse(y< 0,0,1))*g_i*S/propensity2 +
    baseline_effect*S/propensity2 )
welf0_parity_C1 <- mean(sapply(cbind(1, 0, as.matrix(X[,c(3:8)]))%*%
    beta_parity_C1, function(y) ifelse(y< 0,0,1))*g_i*(1 - S)/(1 -
    propensity2) + baseline_effect*(1 - S)/(1 - propensity2 ))

#parity absolution
beta_parity_C1_abs <- res_ms3_parity_abs$result$beta
weight_parity_C1_abs <- res_ms3_parity_abs$result$alpha
## Compute welfare
welf1_parity_C1_abs <- mean(sapply(cbind(1, 1, as.matrix(X[,c(3:8)]))
    %*%beta_parity_C1_abs, function(y) ifelse(y< 0,0,1))*g_i*S/
    propensity2 + baseline_effect*S/propensity2 )
welf0_parity_C1_abs <- mean(sapply(cbind(1, 0, as.matrix(X[,c(3:8)]))
    %*%beta_parity_C1_abs, function(y) ifelse(y< 0,0,1))*g_i*(1 - S)/(1
    - propensity2) + baseline_effect*(1 - S)/(1 - propensity2 ))

##EWM
welf1_EWM_C1_0 <- mean(sapply(cbind(1, 1, as.matrix(X[,c(3:8)]))%*%beta
    1_C1_0, function(y) ifelse(y< 0,0,1))*g_i*S/propensity2 + baseline_
    effect*S/propensity2 )
welf0_EWM_C1_0 <- mean(sapply(cbind(1, 0, as.matrix(X[,c(3:8)]))%*%beta
    1_C1_0, function(y) ifelse(y< 0,0,1))*g_i*(1 - S)/(1 - propensity2)
    + baseline_effect*(1 - S)/(1 - propensity2) )
welf1_EWM_C1_2 <- mean(sapply(cbind(1, as.matrix(X[,c(3:8)]))%*%beta1_C
    1_2, function(y) ifelse(y< 0,0,1))*g_i*S/propensity2 + baseline_
    effect*S/propensity2 )
welf0_EWM_C1_2 <- mean(sapply(cbind(1, as.matrix(X[,c(3:8)]))%*%beta1_C
    1_2, function(y) ifelse(y< 0,0,1))*g_i*(1 - S)/(1 - propensity2) +
    baseline_effect*(1 - S)/(1 - propensity2) )
welf1_EWM_C1_3 <- mean(sapply(cbind(1, as.matrix(X[,c(3:8)]))%*%beta1_C
```

```
    1_3, function(y) ifelse(y< 0,0,1))*g_i*S/propensity2 + baseline_
    effect*S/propensity2 )
welf0_EWM_C1_3 <- mean(sapply(cbind(1, as.matrix(X[,c(3:8)]))%*%beta1_C
    1_3, function(y) ifelse(y< 0,0,1))*g_i*(1 - S)/(1 - propensity2) +
    baseline_effect*(1 - S)/(1 - propensity2) )
## Construct the table
welfares_fem_C1 <- c(welf1_envy_C1, welf1_parity_C1,welf1_parity_C1_abs
    , welf1_EWM_C1_0, welf1_EWM_C1_2, welf1_EWM_C1_3)
welfares_mal_C1 <- c(welf0_envy_C1, welf0_parity_C1,welf0_parity_C1_abs
    , welf0_EWM_C1_0, welf0_EWM_C1_2, welf0_EWM_C1_3)
## alpha for EWM is mean(S) by definition
alphas_C1 <- c(weight_envy_C1,  weight_parity_C1, weight_parity_C1_abs,
     rep(mean(S), 3))
## Construct table
C1_table <- cbind(welfares_fem_C1, welfares_mal_C1,
alphas_C1)
colnames(C1_table) <- c('C1 Welf Fem',  'C1 Welf Mal',
'C1 Weight')
rownames(C1_table) <- c('Fair Envy', 'FTP Pred', 'FTP Pred Abs', '
    Welfare Max. 1 ',
'Welfare Max. 2', 'Welfare Max. 3')
```

7.3 Unfairness levels with different methods

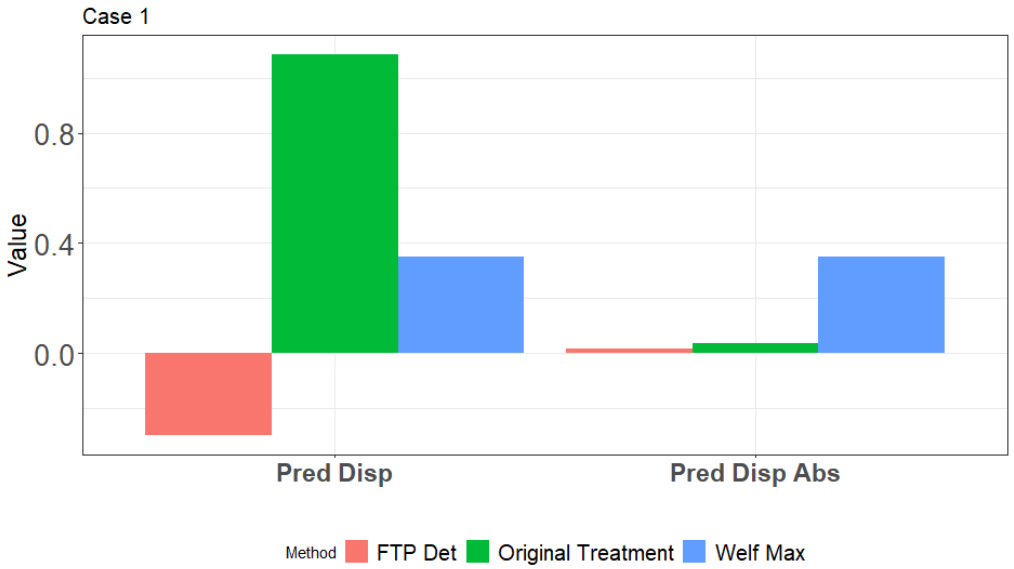

图 2: Unfairness level of the Fair Policy Targeting method with a deterministic allocation rule (in red), Original treatment (in green), and of the welfare maximization method (in blue).

F1 reports the unfairness level with unfairness measured as the difference in the probability of treatments between the two groups. Overall, F1 shows that the level of the unfairness of the proposed method is uniformly smaller than the unfairness achieved by maximizing welfare and also smaller than the original treatment.

```
load('./results/elements.RData')
load('./results/alg_parity_abs.RData')
deterministic_ms3 <- res_ms3_parity
source('./library/library.R')
original_C1_abs <- abs(-sum(S * D )/sum(S) + sum((1 - S) * D )/sum(1 -
    S) )
parity_C1_det_abs <- abs(-sum(S * deterministic_ms3$result$policies )/
    sum(S) + sum((1 - S) * deterministic_ms3$result$policies )/sum(1 - S
    ) )
########################################
### Compare with EWM
########################################
load('./results/solution_EWM.RData')
parity_EWM_C1 <- - sum(S * res0_EWM_C1$pi)/sum(S) + sum((1 - S) * res0_
    EWM_C1$pi)/sum(1 - S)
parity_EWM_C1_abs <- abs(- sum(S * res0_EWM_C1$pi)/sum(S) + sum((1 - S)
     * res0_EWM_C1$pi)/sum(1 - S))

library(ggplot2)
data_frame1 <- c(parity_C1_det, original_C1, parity_EWM_C1,
parity_C1_det_abs, original_C1_abs, parity_EWM_C1_abs)
types <- c('FTP Det', 'Original Treatment', 'Welf Max',
'FTP Det', 'Original Treatment', 'Welf Max')
unfairness <- c('Pred Disp', 'Pred Disp', 'Pred Disp',
'Pred Disp Abs', 'Pred Disp Abs', 'Pred Disp Abs')
dd1 <- cbind(data_frame1, types, unfairness)
dd1 <- as.data.frame(dd1)
names(dd1) <- c('UnFairness', 'Method', 'Type')
dd1[,1] <- as.numeric(as.character(dd1[,1]))

bar_chart1 <- ggplot(dd1, aes(y=UnFairness, x=Type, fill = Method)) +
geom_bar(position="dodge", stat="identity") +
ggtitle("Case 1") +
ylab("Value") +
xlab('') +
theme_bw() +
theme(legend.position="bottom",
axis.text.x = element_text(face="bold",
```

```
        size=17),
        axis.title.x = element_text(size=17),
        axis.title.y = element_text(size=17),
        legend.text=element_text(size = 15),
        plot.title = element_text(size=15),
        axis.text.y = element_text(size = 20))
        plot(bar_chart1)
```

8 Expand Fair Policy Targeting during multiple time periods

8.1 Staggered Treatment

Assumption (Irreversibility of Treatment).

$$D_1 = 0 \text{ almost surely (a.s.).For } t = 2, \ldots, \mathcal{T}, D_{t-1} = 1 \text{ implies that } D_t = 1 \text{a.s.} \tag{23}$$

This assumption states that no one is treated at time t = 1, and that once a unit becomes treated, that unit will remain treated in the next period.This assumption is also called staggered treatment adoption in the literature.

During this treatment, we can using Fair Policy Targeting in every period, while every units who were indicated to be treated should be excluded from subsequent policy allocation.

```
        remo=vector()
        for(i in 1:length(test)){
                if(policies[i]==1)
                remo=c(remo,i)
        }
        data=data[-remo,]
```

Then use Fair Policy Targeting to new method and keep doing this cycle until $t = \mathcal{T}$

8.2 Units "forget" about the treatment experience

Without the Assumption(Irreversibility of Treatment), we assume that every unit's treatment in every period is individual that is the outcomes of the treatment in this period may not affect the treatment allocation in next period.

During this treatment, we can using Fair Policy Targeting in every period without excluding any units in each period.