# Causal Effect of Job Class on Wage of Males from the Atlantic Region of the United States

Fei Zhang, Kai Zhang, Hongyao Zhu

## Background and Scientific Question

The wage paid to workers varies greatly, mostly the result of differences in job classification and the result of worker ability and the workers' effort. There are often legitimate reasons for treating the compensation of workers and type of job differently. It is partially because of differences in the demand and supply of the market and laborers for particular jobs or occupations. These differences arise primarily because of differences in the education level, professional degree, or other training required and the job's desirability. There exist more factors heavily affect job classification and wages, such as age, marital status, health conditions of workers and macroeconomic development, and so forth. Therefore, we tend to evaluate the effect of job class on the wage of males from the Atlantic region of the United States.

This dataset is used in the book "An Introduction to Statistical Learning" and was manually assembled by Steve Miller, of Open BI (www.openbi.com), from the March 2011 Supplement to Current Population Survey data. The original total dataset can be found at https://data.census.gov/mdat/#/ in the CPS Annual Social and Economic (March) Supplement. We downloaded the dataset `Wage` from package `ISLR`.

This dataset contains information about a group of 3000 males from the Atlantic region of the United States. Details of the code book of the dataset is shown in the appendix.

## Structural Causal Model

- Endogenous nodes: $X = (W, A, Y)$
- Background (exogenous) variables: $U = (U_W, U_A, U_Y) \sim P^*$
- Structural equations $F$:

$$\begin{aligned} W &\leftarrow f_W(U_W) \\ A &\leftarrow f_A(W, U_A) \\ Y &\leftarrow f_Y(W, A, U_Y) \end{aligned}$$

where $W = \{$age, race, education level, marital status, health level, year that wage information was recorded, health insurance status$\}$, $A$=$\{$type of job (industry/information)$\}$ and $Y = \{$raw wage$\}$.

The directed acyclic graph representing the SCM is shown below:

## Counterfactuals and Target Causal Parameter

We denote the counterfactual wages as $Y_1$ and $Y_0$. And they are derived from following equations:

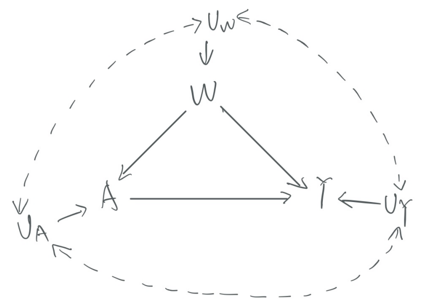


Figure 1: **Fig1.** The DAG of the SCM

$$Y_1 \leftarrow f_Y(W, A=1, U_Y)$$

$Y_0$ is counterfactual raw wage for a subject if possibly contrar-to-fact he or she worked in a information job, given the baseline covariates $W$ and unobserved factors of wage $U_Y$.

$$Y_0 \leftarrow f_Y(W, A=0, U_Y)$$

$Y_0$ is counterfactual raw wage for a subject if possibly contrar-to-fact he or she worked in a industry job, given the baseline covariates $W$ and unobserved factors of wage $U_Y$.

The target causal parameter is the average treatment effect (ATE), which is denoted as $\theta^*$:

$$\begin{aligned}\theta^*(\mathrm{P}^*) &= \mathrm{E}^*(Y_1) - \mathrm{E}^*(Y_0) \\ &= \mathrm{E}^*\big[f_Y(W, 1, U_Y)\big] - \mathrm{E}^*\big[f_Y(W, 0, U_Y)\big]\end{aligned}$$

where $P^*$ denotes the joint distribution of background factors, and uniquely determine the distribution of counterfactuals.

Our target causal parameter, ATE, is the difference in the expected counterfactual wage if all population had an Information job versus if all population had Industrial job.

## Observed Data & Link to Causal Model

We assume that the observed data were generated by a system compatible with (described by) our structural causal model. The distribution of the background variables $\mathrm{P}^*$ and the structural equations $F$ identifies the distribution of the observed data $O$. If we knew the distribution of $U$ and the functions $F$, we would know the distribution of $O$. Thus, the causal model has implications for the distribution of $O$. These implications are testable.

Table 1 describes the observed data grouped by job class.

Table 1: The Summary Table of Mid-Atlantic Wage Data

| | level | 1. Industrial | 2. Information |
|---|---|---|---|
| n | | 1544 | 1456 |
| Year (mean (SD)) | | 2005.80 (2.05) | 2005.78 (2.00) |
| Age (mean (SD)) | | 41.40 (11.69) | 43.49 (11.29) |
| Matrital Status (%) | 1. Never Married | 357 ( 23.1) | 291 ( 20.0) |
| | 2. Married | 1046 ( 67.7) | 1028 ( 70.6) |
| | 3. Widowed | 12 ( 0.8) | 7 ( 0.5) |
| | 4. Divorced | 103 ( 6.7) | 101 ( 6.9) |
| | 5. Separated | 26 ( 1.7) | 29 ( 2.0) |
| Race (%) | 1. White | 1325 ( 85.8) | 1155 ( 79.3) |
| | 2. Black | 111 ( 7.2) | 182 ( 12.5) |
| | 3. Asian | 86 ( 5.6) | 104 ( 7.1) |
| | 4. Other | 22 ( 1.4) | 15 ( 1.0) |
| Education Level (%) | 1. < HS Grad | 190 ( 12.3) | 78 ( 5.4) |
| | 2. HS Grad | 636 ( 41.2) | 335 ( 23.0) |
| | 3. Some College | 342 ( 22.2) | 308 ( 21.2) |
| | 4. College Grad | 274 ( 17.7) | 411 ( 28.2) |
| | 5. Advanced Degree | 102 ( 6.6) | 324 ( 22.3) |
| Health Status (%) | 1. <=Good | 487 ( 31.5) | 371 ( 25.5) |
| | 2. >=Very Good | 1057 ( 68.5) | 1085 ( 74.5) |
| Health Insurance (%) | 1. Yes | 969 ( 62.8) | 1114 ( 76.5) |
| | 2. No | 575 ( 37.2) | 342 ( 23.5) |
| Wage (mean (SD)) | | 103.32 (35.11) | 120.59 (46.13) |
| Job Class (%) | 1. Industrial | 1544 (100.0) | 0 ( 0.0) |
| | 2. Information | 0 ( 0.0) | 1456 (100.0) |

*Note:*
The unit of wage is U.S.dollar.

## Identification

In the original structural causal model, the target causal parameter is not identified from the observed data distribution. For identifiability to hold, we would need the following independence assumptions to hold: $U_A \perp U_Y$ and (i) $U_A \perp U_W$, or (ii) $U_Y \perp U_W$. We also need the positivity assumption to hold

$$0 < P(A = 1|W = w) < 1$$

for all $w$ for which $\mathrm{P}(W = w) > 0$. In terms of our dataset, there must be a positive probability of the subjects with Information job class within strata of baseline covariates.

In this study, we have covariates (age, marital status, race, education level, health status, health insurance, and the recording year), representing the objective conditions of a person. Subjective conditions like personality or personal preference may affect job classes but not the wage. Therefore, from a material perspective, give assumptions discussed above, our target causal parameter can be identified. And $U_A \perp U_Y$, together with $U_Y \perp U_W$ may be more plausible. More variables such as family members' job class, family members' wages may also be included.

## Specify the Statistical Estimand

The target causal estimand of interest is the average treatment effect (ATE). With identifiability, conterfactual parameter is computed by observed data distribution. Under randomization assumption and back-door criterion, the effect of exposure $A$ on outcome $Y$ is identified via the G-Computation formula (statistical estimand):

$$\begin{aligned}\theta^* &= \mathrm{E}\big[\mathrm{E}(Y|A = 1, W) - \mathrm{E}(Y|A = 0, W)\big] \\ &= \sum_w \big[\mathrm{E}(Y|A = 1, W = w) - \mathrm{E}(Y|A = 0, W = w)\big]\mathrm{P}(W = w)\end{aligned}$$

## Estimate

The target causal parameter, the average treatment effect, is estimated using four different estimators, including G-Computation, IPTW, AIPW, and TMLE. Data-adaptive method algorithms is used to estimate $g(W) = P(A = 1|W)$ and $m(A, W) = E(Y|A, W)$, so as to ensure $\hat{g}$ and $\hat{m}$ consistent. In addition, non-parametric bootstrap is performed to construct confidence interval by quantiles for the estimates. Specifically, we resampled the data with replacement 100 times (See *Appendeix*). With the point estimates and confidence intervals, valid statistical inference can be made.

### Super Learners for $g$ and $m$

#### *Select algorithms for Super Learner*

The Super Learners are built for $g(W)$ and $m(A, W)$ correspondingly. By tunning the hyperparameter using grid search, especially for ensemble methods, we achieved higher accuracy and shorter computing time. The detailed hyperparameter tuning process is shown in the *Appendix*.

To estimate outcome variable (wage), Super Learner library includes algorithms in the following:

- Lasso Regression (biglasso).
- Elastic_Net Regression (glmnet).

- Gradient Boosting Machine (n.tree = 100, max.depth = 1, learing rate = 0.1).
- Random Forest (n.tree = 500).
- XGBoost (n.tree = 500, max.depth = 3, learning rate = 0.01).

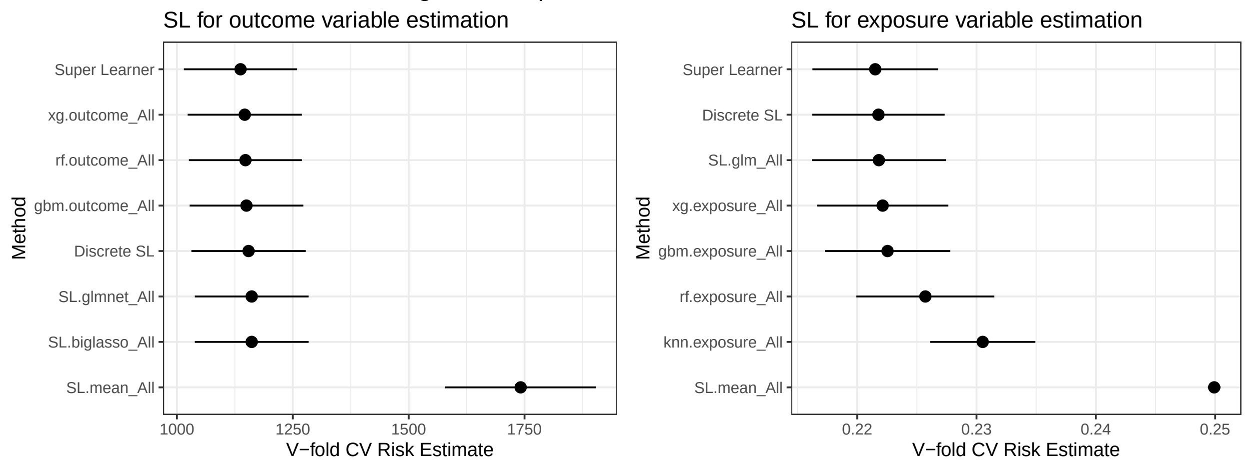
To estimate exposure variable (type of job), Super Learner library includes algorithm in the following:

- Logistic Regression.
- K Nearest Neighbors (K = 50).
- Gradient Boost Machine (n.tree = 1000, max.depth = 1, learning rate = 0.1)
- Random Forest (n.trees = 100)
- XGBoost (n.tree = 1000, max.depth = 1, learning rate = 0.1).

### ***Evaluate Performance of Super Learner***

Two-layer cross-validation is used to evaluate the performance of each SL libraries and Super Learner (See *Figure 1*).

*Figure 1−SuperLearners' Performances*



For estimating $g(Y|A, W)$, the Super Learner has cross-validated risk of 1137.2 (62.37), which outperforms all other algorithms in the library. It is a optimal linear convex combination of four algorithms, including lasso regression (coefficient = 0.20), random forest (coefficient = 0.34), gradient boost machine (coefficient = 0.11) and XGBoost (coefficient = 0.35).

For estimating $m(A|W)$, the Super Learner has cross-validated risk of 0.221 (0.0027), which also outperforms all other algorithms in the library. It is a optimal linear convex combination of five algorithms, including logistic regression (coefficient = 0.59), KNN (coefficient = 0.12), random forest (coefficient = 0.19), gradient boost machine (coefficient = 0.033) and XGBoost (coefficient = 0.066).

The final Super Learners for $g$ and $m$ can be viewed in *Appendix*

## Simple Substitution Estimator

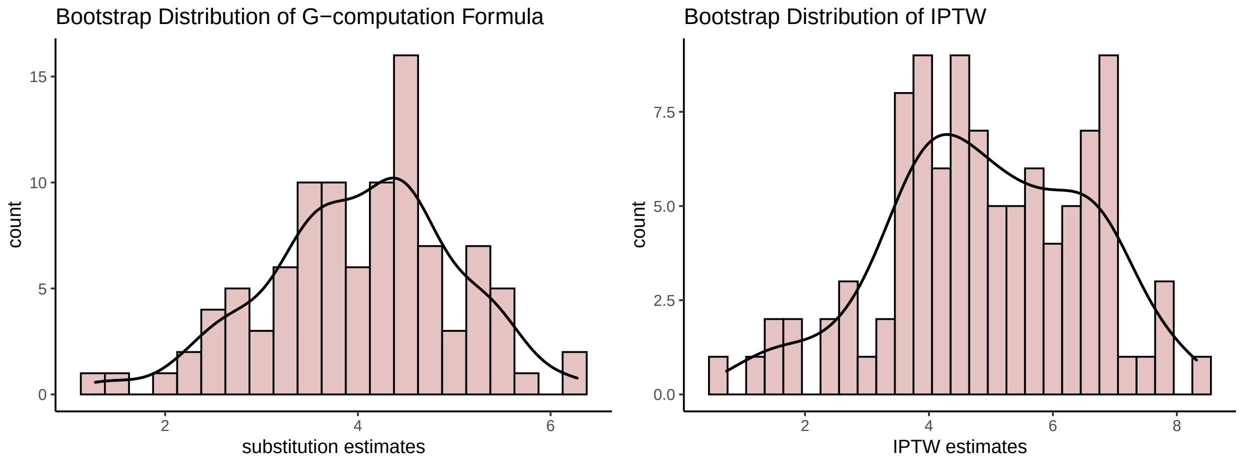
The simple substitution estimator based on the G-computation formula is:

$$\hat{\theta}_{G-comp} = \frac{1}{n}\sum_{i=1}^{n}\Big(\hat{\mathrm{E}}(Y_i|A_i = 1, W_i) - \hat{\mathrm{E}}(Y_i|A_i = 0, W_i)\Big)$$

where $\hat{\mathrm{E}}(Y|A, W)$ is the estimator of the conditional mean outcome given the exposure (job class) and baseline covariates. Consistency of the simple (non-targeted) substitution estimator depends on the consistent estimation of the conditional mean outcome $\mathrm{E}(Y|A, W)$.

The point estimate is 3.25. According to boostrap distribution (See the left of *Figure 2*), the standard error is 0.98 with 95% confidence interval ((2.13, 5.63)). And also, the histogram shows that the bootstrap distribution of parametric G-computation is sligtly skewed. This is not surprising, because simple substitution estimation is not asymptotically linear.

*Figure 2−Bootstrap Distribution of G−Computation and IPTW*



## Inverse Probability of Treatment Weighted Estimator (IPTW)

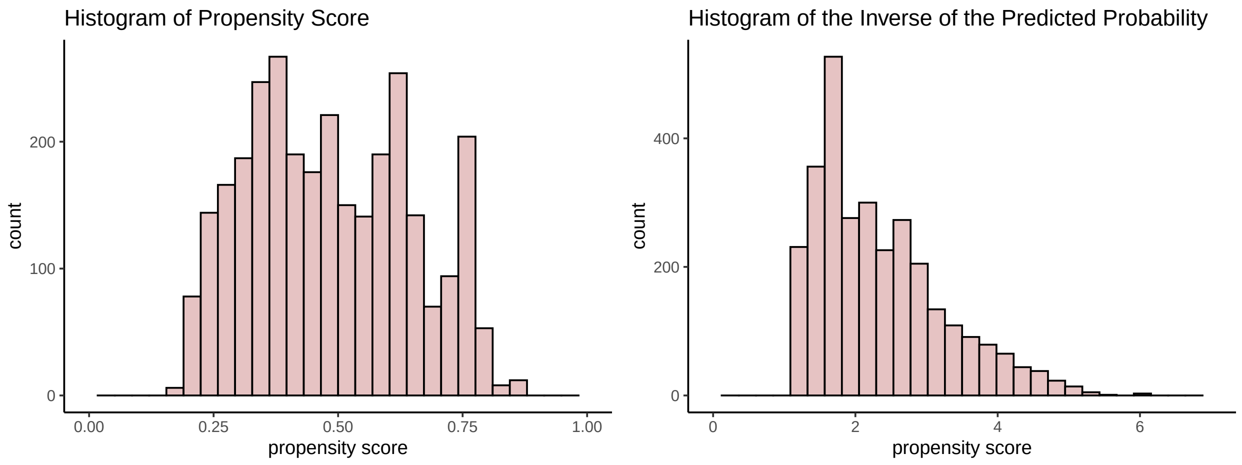
The IPTW estimator is:

$$\hat{\theta}_{IPTW} = \frac{1}{n} \sum_{i=1}^{n} \left( \frac{\mathrm{I}(A_i = 1)}{\hat{\mathrm{P}}(A_i = 1|W_i)} - \frac{\mathrm{I}(A_i = 0)}{\hat{\mathrm{P}}(A_i = 0|W_i)} \right) Y_i$$

where $\hat{\mathrm{P}}(A = 1|W)$ is the estimator of the treatment mechanism (i.e. the conditional probability of having a information job, given the baseline covariates). Consistency of IPTW estimators depends on consistent estimation of the propensity score $\mathrm{P}(A = 1|W)$.

According to *Figure 3*, the range of estimated propensity score is between 0.16 and 0.87, and the coresponding inverse of the predicted probilities $\hat{we} = 1/\hat{P}(A = 1|W)$ ranges from 1.14 to 6.13. Therefore, there is no need to do any truncation and stablization (we actually try truncating 1% and 5% extreme observations, and it makes the estimates biased and not significant).

*Figure 3−Distribution of Propensity Score and its Inverse*



The point estimate is 4.72. According to the boostrap distribution (See the right of *Figure 2*), the standard error is 1.63 with 95% confidence interval (1.49, 7.85). Similar to substitution estimator, the histogram shows that the bootstrap distribution of IPTW is also not normal. And also, the IPTW is also not asymptotically linear.

## Augmented Inverse Propensity Weighted Estimator

The AIPW estimator is:

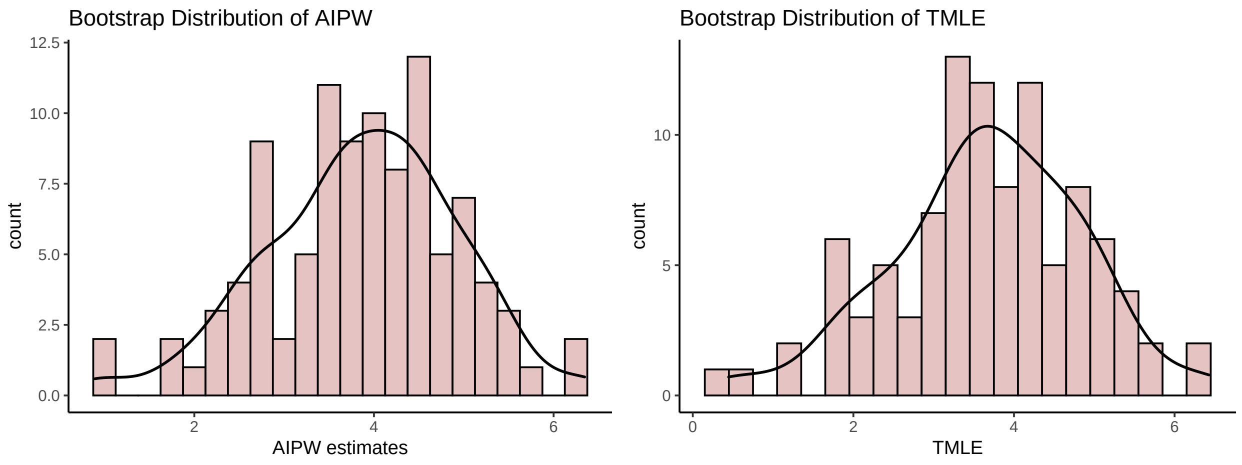
$$\begin{aligned}\hat{\theta}_{AIPW} &= \frac{1}{n}\sum_{i=1}^{n}\left\{\frac{A_i}{\hat{g}(W_i)}[Y_i - \hat{m}(A_i = 1, W_i)] + \hat{m}(A_i = 1, W_i)\right\} \\ &- \frac{1}{n}\sum_{i=1}^{n}\left\{\frac{1 - A_i}{1 - \hat{g}(W_i)}[Y_i - \hat{m}(A_i = 0, W_i)] + \hat{m}(A_i = 0, W_i)\right\}\end{aligned}$$

Where $\hat{g}(W)$ is the estimate of $P(A = 1|W)$, and $\hat{m}(A, W)$ is the estimate of $E(Y|A, W)$. IPTW requires estimation of both the conditional mean function $\mathrm{E}(Y|A, W)$ and the treatment mechanism $g(W) = \mathrm{P}(A = 1|W)$. If both $\hat{g}(W)$ and $\hat{m}(A, W)$ are consistent, then $\hat{\theta}_{AIPW}$ is asymptotically linear. Further, the influence curve based variance of $\hat{\theta}_{AIPW}$ can be estimated by

$$\begin{aligned}Var(\hat{\theta}_{AIPW}) &= \frac{1}{n^2}\sum_{i=1}^{n}\left\{\frac{A_i}{\hat{g}(W_i)}[Y_i - \hat{m}(A_i = 1, W_i)] + \hat{m}(A_i = 1, W_i)\right. \\ &\left. - \frac{1 - A_i}{1 - \hat{g}(W_i)}[Y_i - \hat{m}(A_i = 0, W_i)] - \hat{m}(A_i = 0, W_i) - \hat{\theta}_{AIPW}\right\}^2\end{aligned}$$

Accordingly, the AIPW point estimate is 3.53. The influence curve based standard error equals to 1.13, and 95% confidence interval is (1.31, 5.75). In addition, the bootstrap based standard error is 1.06, and 95% confidence interval by quantile is (1.75, 5.58). The bootstrap distribution of AIPW is shown in the left of *Figure 4*, it looks as though the estimates follow a normal distribution approaximately.

*Figure 4−Bootstrap Distribution of AIPW and TMLE*



## Targeted Maximum Likelihood Estimation

The TMLE is formulated by:

$$\hat{\theta}_{TMLE} = \frac{1}{n}\sum_{i=1}^{n}\Big[\tilde{m}(A_i = 1, W_i)\Big] - \frac{1}{n}\sum_{i=1}^{n}\Big[\tilde{m}(A_i = 0, W_i)\Big]$$

where $\tilde{m}(A, W)$ denotes the targeted estimate of the conditional mean outcome, given the treatment variable and baseline covariates. Similar to AIPW, TMLE requires estimations of $\mathrm{E}(Y|A, W)$ and $g(W) = \mathrm{P}(A = 1|W)$. TMLE is also a double robust estimator, and it is asymptotically linear if both $\mathrm{E}(Y|A, W)$ and $\mathrm{P}(A|W)$ are estimated consistently. Therefore, the influence curved based variance of TMLE can be computed by

$$\begin{aligned} Var(\hat{\theta}_{TMLE}) = \frac{1}{n^2}\sum_{i=1}^{n}\bigg\{ & \frac{A_i}{\hat{g}(W_i)}[Y_i - \tilde{m}(A_i = 1, W_i)] + \tilde{m}(A_i = 1, W_i) \\ & - \frac{1 - A_i}{1 - \hat{g}(W_i)}[Y_i - \tilde{m}(A_i = 0, W_i)] - \tilde{m}(A_i = 0, W_i) - \hat{\theta}_{TMLE}\bigg\}^2 \end{aligned}$$

Accordingly, the TMLE point estimate is 3.57. By influence curve based method, the standard error is 1.13, and the corresponding 95% confidence interval is (1.35, 5.79). For bootstrap data, the standard error is 1.18, and the bootstrap based 95% confidence interval is (1.2, 5.79). And also, the boostrap distribution of TMLE is shown in right of *Figure 4*, it also looks as though the TMLE follows a normal distribution approaximately.

## Interpret Result

*Table 2* displays the point estimates of the four estimators and the corresponding 95% confidence interval.

All the 95% confidence interval don't contain 0, indicating the expected counterfactual wage if all population had an information job is significantly greater than that if all population had an industrial job.

In general, Substitution estimator, AIPW, and TMLE have similar estimates on ATE, while IPTW estimate is relatively greater than the other three. This may be due to that the Super Learner makes a correct parametric model for $m$, while it does not make a consistent estimator $g$. It is for this very same reason, $\hat{\theta}_{AIPW}$ and $\hat{\theta}_{TMLE}$ have a wider bootstrap based confidence interval compared to $\hat{\theta}_{G-comp}$.

Table 2: Summary of estimators

| Estimator | ATE | SE(influence curve) | 95% CI(influence curve) | SE(bootstrap) | 95% CI(bootstrap) |
|---|---|---|---|---|---|
| Substitution Estimator | 3.25 | | | 0.98 | (2.13, 5.63) |
| IPTW | 4.76 | | | 1.63 | (1.49, 7.85) |
| AIPW | 3.53 | 1.13 | (1.31, 5.75) | 1.06 | (1.75, 5.58) |
| TMLE | 3.57 | 1.13 | (1.35, 5.79) | 1.18 | (1.20, 5.79) |

[1] SE: Standard error
[2] CI: Confidence interval
[3] IPTW: Inverse probability treatment weighted
[4] AIPW: Augmented inverse propensity weighted
[5] TMLE: Targeted maximal likelihood estimation

Both for AIPW and TMLE, the influence curve based standard error and 95% confidence interval, are very similar to that derived from bootstrap distribution. Combined with the histograms in *Figure 4*, $\hat{\theta}_{AIPW}$ and $\hat{\theta}_{TMLE}$ are approximately normal distributed, which is one of nice implications of asymptotic linearity. In addition, asymptotic linearity also provides a robust approach to variance estimation, making these two doubly-robust estimations have extremely similar influence curve based standard error.

One of limitations of our study is that the super Learner doesn't create a consistent estimator for treatment machanism. The Super Learner only slightly outperforms logistic regression in risk. This makes the IPTW biased and unstable. Moreover, there is no scientific proof on whether the bootstrap works in general for data-adaptive methods, so the bootstrap based confidence interval for G-computation and IPTW is not guaranteed theoretically. And also, some of important confounders, such as professional degree, state of the economy and family background, are not adjusted.

## Conclusion

Four estimators were applied to estimate the causal effect of job class on the wage of males from the Atlantic region of the United States. We found that information job class will cause a higher wage in such a population, and it is estimated using TMLE that the difference is 3.57 with inflence curve based 95% CI (1.35, 5.79). The difference in wages in these two job classes may indicate that as information job class was a fast-developing job market in 2011, employees in the information field are likely to earn a higher wage than industrial. Also, companies in the information field may have higher profits so that they can afford and are willing to pay a higher wage.

## Appendix

### Code Book

1. age - continuous, age that wage information was recorded
2. race - race
    - race = 1: White
    - race = 2: Black
    - race = 3: Asian
    - race = 4: Other

3. education - education level
    - education = 1: <HS Grad
    - education = 2: HS Grad
    - education = 3: Some College
    - education = 4: College Grad
    - education = 5: Advanced Degree
4. maritl - marital status
    - maritl = 1: Never Married
    - maritl = 2: Married
    - maritl = 3: Widowed
    - maritl = 4: Divorced
    - maritl = 5: Separated
5. health - health level of worker
    - health = 1: <= Good
    - health = 2: >= Very Good
6. year - continuous, year that wage information was recorded
7. health_ins - whether worker has health insurance
    - health_ins = 1: Yes
    - health_ins = 2: No
8. jobclass - type of job
    - jobclass = 1: Industrial
    - jobclass = 2: Information

## ***Covariates*** $W = \{ W_1, W_2, W_3, W_4, W_5, W_6, W_7 \}$

$W_1$: age

$W_2$: race

$W_3$: education level

$W_4$: marital status

$W_5$: health

$W_6$: year

$W_7$: health insurance

$A$: job class

$Y$: wage

$U_{W_3}$: family education level

$U_{W_4}$: husband information, family background, personal preference

$U_{W_5}$: lifestyle, work pressure, metal disease

$U_{W_7}$: the degree of importance attached to health, insurance price, insurance policy

$U_A$: family members' jobs, personal preference, job treatment, reputation job requirement, professional degree

$U_Y$: economic environment, personality, operational competence, companies' quality, year of experience, government policy

In general, $U_W$ includes personal background and mental conditions. $U_A$ contains personal information and job specific stuff. $U_Y$ includes company information, macroeconomic environment, and personal competence. Therefore, there are lots of overlapping in $U_W$ and $U_A$. However, there is little cross between $U_W$ and $U_Y$, similarly between $U_A$ and $U_Y$. Therefore, it is hard to assume $U_W \perp U_A$, but it is reasonable to have assumptions that $U_W \perp U_Y$ and $U_A \perp U_Y$.

## Super Learners for Outcome and Treatment Variable

### ***Predicting Outcome*** $m$

Super Learner find a optimal linear convex conbination of the algorithms in the library.

```
##
## Call:
## snowSuperLearner(cluster = cluster, Y = dat$wage, X = X, family = gaussian(),
##     SL.library = outcome_library, cvControl = list(V = 10))
##
##
##                      Risk      Coef
## SL.biglasso_All 1160.335 0.2006063
## SL.glmnet_All   1160.448 0.0000000
## SL.mean_All     1741.624 0.0000000
## rf.outcome_All  1147.541 0.3347914
## gbm.outcome_All 1149.348 0.1098917
## xg.outcome_All  1143.416 0.3547106
```

### ***Predicting Treatment*** $g$

Super Learner find a optimal linear convex conbination of the algorithms in the library.

```
##
## Call:
## snowSuperLearner(cluster = cluster, Y = dat$A, X = dplyr::select(X, -A),
##     family = binomial(), SL.library = exposure_library, cvControl = list(V = 10))
##
##
##
##                        Risk       Coef
## gbm.exposure_All 0.2226626 0.03328041
## rf.exposure_All  0.2277321 0.19036073
## xg.exposure_All  0.2233446 0.06583254
## knn.exposure_All 0.2307346 0.11908896
## SL.glm_All       0.2227286 0.59143735
## SL.mean_All      0.2498825 0.00000000
```

## Tuning Hyperparameters for Super Learners

Hyperparameters are the configuration settings for an algorithm, and the performance of an algorithm varies based on its hyperparameters. Here, we use `create.Learner` function to customize hyperparameters for our algorithms with strategy of grid search.

### *GBM for Outcome*

Using 5-fold cross validation, the tuned GBM for outcome has 100 trees with interaction depth 1 and shrinkage 0.1.

```
##
## Call:
## CV.SuperLearner(Y = dat$wage, X = X, V = 5, family = gaussian(), SL.library = gbm.learners$names,
##     parallel = cluster)
##
## Risk is based on: Mean Squared Error
##
## All risk estimates are based on V =  5
##
##             Algorithm    Ave     se     Min    Max
##         Super Learner 1133.5 61.756  863.48 1365.5
##           Discrete SL 1135.1 61.893  863.65 1371.2
##   gbm_100_1_0.001_All 1677.2 81.343 1441.54 1974.0
##   gbm_500_1_0.001_All 1489.7 75.733 1245.45 1768.7
##  gbm_1000_1_0.001_All 1350.6 71.226 1098.30 1615.7
##   gbm_100_2_0.001_All 1677.5 81.335 1442.26 1973.4
##   gbm_500_2_0.001_All 1489.8 75.699 1244.83 1768.5
##  gbm_1000_2_0.001_All 1351.0 71.284 1098.91 1616.5
##    gbm_100_1_0.01_All 1350.3 71.276 1099.30 1613.1
##    gbm_500_1_0.01_All 1143.6 62.807  876.61 1380.8
##   gbm_1000_1_0.01_All 1136.0 62.129  863.65 1368.3
##    gbm_100_2_0.01_All 1348.8 71.105 1097.56 1616.1
##    gbm_500_2_0.01_All 1142.5 62.666  873.64 1383.1
##   gbm_1000_2_0.01_All 1135.1 62.090  863.79 1371.2
##     gbm_100_1_0.1_All 1133.0 61.532  858.63 1367.4
##     gbm_500_1_0.1_All 1133.7 61.141  865.43 1365.1
##    gbm_1000_1_0.1_All 1140.5 61.858  873.46 1363.2
##     gbm_100_2_0.1_All 1134.9 61.809  868.46 1371.2
##     gbm_500_2_0.1_All 1136.2 62.061  856.47 1369.3
##    gbm_1000_2_0.1_All 1136.7 61.805  863.86 1367.2
```

### *Random Forst for Outcome*

Using 5-fold cross validation, the tuned random forest for outcome has 500 trees.

```
##
## Call:
## CV.SuperLearner(Y = dat$wage, X = X, V = 5, family = gaussian(), SL.library = rf.learners$names,
##     parallel = cluster)
##
```

```
## Risk is based on: Mean Squared Error
##
## All risk estimates are based on V =  5
##
##        Algorithm    Ave     se    Min    Max
##    Super Learner 1148.5 62.984 964.53 1328.9
##      Discrete SL 1149.2 63.027 964.76 1328.9
##  SL.ranger_1_All 1148.6 62.735 966.99 1329.9
##  SL.ranger_2_All 1147.6 62.896 961.70 1329.9
##  SL.ranger_3_All 1147.6 63.014 964.76 1328.9
```

### ***XGBoost for Outcome***

Using 5-fold cross validation, the tuned xgBoost for outcome has 500 trees with interaction depth 3 and shrinkage 0.01.

```
##
## Call:
## CV.SuperLearner(Y = dat$wage, X = X, V = 5, family = gaussian(), SL.library = xg.learners$names,
##     parallel = cluster)
##
## Risk is based on: Mean Squared Error
##
## All risk estimates are based on V =  5
##
##             Algorithm     Ave      se     Min     Max
##         Super Learner  1812.8  85.817  1065.0  4028.8
##           Discrete SL  1147.7  62.644  1042.6  1242.8
##   xgb_100_1_0.001_All 11820.8 212.733 11495.8 12012.9
##   xgb_500_1_0.001_All  6125.7 158.854  5927.8  6277.4
##  xgb_1000_1_0.001_All  3158.1 120.340  3059.3  3275.5
##   xgb_100_2_0.001_All 11798.3 211.602 11482.0 11994.3
##   xgb_500_2_0.001_All  6042.1 154.398  5863.4  6184.0
##  xgb_1000_2_0.001_All  3035.6 114.944  2950.2  3138.6
##   xgb_100_3_0.001_All 11786.0 210.772 11481.5 11989.8
##   xgb_500_3_0.001_All  5996.1 152.082  5848.4  6131.7
##  xgb_1000_3_0.001_All  2985.6 112.128  2891.4  3088.9
##    xgb_100_1_0.01_All  3142.4 120.089  3044.3  3259.5
##    xgb_500_1_0.01_All  1229.4  67.184  1152.5  1300.7
##   xgb_1000_1_0.01_All  1179.2  64.105  1083.1  1256.0
##    xgb_100_2_0.01_All  3019.7 114.674  2935.1  3122.3
##    xgb_500_2_0.01_All  1164.8  64.123  1068.8  1247.5
##   xgb_1000_2_0.01_All  1149.8  62.584  1042.6  1239.5
##    xgb_100_3_0.01_All  2969.6 111.873  2875.3  3072.7
##    xgb_500_3_0.01_All  1149.4  63.318  1057.7  1238.4
##   xgb_1000_3_0.01_All  1150.1  62.383  1042.5  1245.2
##     xgb_100_1_0.1_All  1177.4  64.001  1080.0  1255.0
##     xgb_500_1_0.1_All  1151.9  62.424  1043.7  1246.8
##    xgb_1000_1_0.1_All  1153.7  62.299  1043.5  1254.0
##     xgb_100_2_0.1_All  1151.2  62.592  1044.1  1242.8
##     xgb_500_2_0.1_All  1167.4  62.658  1056.9  1267.4
##    xgb_1000_2_0.1_All  1188.6  63.322  1073.6  1288.7
##     xgb_100_3_0.1_All  1153.2  62.552  1047.6  1253.4
##     xgb_500_3_0.1_All  1225.6  64.614  1102.5  1349.1
##    xgb_1000_3_0.1_All  1275.9  66.183  1141.2  1420.1
```

### ***GBM for Exposure***

Using 5-fold cross validation, the tuned GBM for outcome has 1000 trees with interaction depth 1 and shrinkage 0.1.

```
##
## Call:
## CV.SuperLearner(Y = dat$A, X = dplyr::select(X, -A), V = 5, family = binomial(),
##     SL.library = gbm.learners$names, parallel = cluster)
##
## Risk is based on: Mean Squared Error
##
## All risk estimates are based on V =  5
##
##             Algorithm     Ave         se     Min     Max
##         Super Learner 0.22272 0.00264110 0.21189 0.23299
##           Discrete SL 0.22286 0.00265494 0.21181 0.23294
##   gbm_100_1_0.001_All 0.24599 0.00036023 0.24541 0.24740
##   gbm_500_1_0.001_All 0.23563 0.00098720 0.23247 0.24029
##  gbm_1000_1_0.001_All 0.22927 0.00153307 0.22395 0.23595
##   gbm_100_2_0.001_All 0.24597 0.00036051 0.24541 0.24734
##   gbm_500_2_0.001_All 0.23562 0.00099175 0.23237 0.24036
##  gbm_1000_2_0.001_All 0.22925 0.00153131 0.22388 0.23593
##    gbm_100_1_0.01_All 0.22920 0.00153713 0.22389 0.23566
##    gbm_500_1_0.01_All 0.22235 0.00263977 0.21163 0.23312
##   gbm_1000_1_0.01_All 0.22287 0.00268672 0.21181 0.23330
##    gbm_100_2_0.01_All 0.22915 0.00154206 0.22390 0.23571
##    gbm_500_2_0.01_All 0.22256 0.00261054 0.21206 0.23294
##   gbm_1000_2_0.01_All 0.22237 0.00265510 0.21141 0.23269
##     gbm_100_1_0.1_All 0.22265 0.00262329 0.21280 0.23354
##     gbm_500_1_0.1_All 0.22283 0.00263890 0.21163 0.23367
##    gbm_1000_1_0.1_All 0.22180 0.00265002 0.21092 0.23215
##     gbm_100_2_0.1_All 0.22337 0.00270251 0.21283 0.23308
##     gbm_500_2_0.1_All 0.22315 0.00267110 0.21268 0.23389
##    gbm_1000_2_0.1_All 0.22311 0.00266002 0.21089 0.23423
```

### ***Random Forest for Exposure***

Using 5-fold cross validation, the tuned random forest for outcome has 100 trees.

```
##
## Call:
## CV.SuperLearner(Y = dat$A, X = dplyr::select(X, -A), V = 5, family = binomial(),
##     SL.library = rf.learners2$names, parallel = cluster)
##
## Risk is based on: Mean Squared Error
##
## All risk estimates are based on V =  5
##
##        Algorithm     Ave        se     Min     Max
##    Super Learner 0.22747 0.0029901 0.21593 0.24296
##      Discrete SL 0.22765 0.0029901 0.21591 0.24302
##  SL.ranger_1_All 0.22718 0.0030268 0.21453 0.24341
##  SL.ranger_2_All 0.22787 0.0029870 0.21630 0.24302
##  SL.ranger_3_All 0.22751 0.0029866 0.21591 0.24381
```

***XGBoost for Exposure***

Using 5-fold cross validation, the tuned xgBoost for exposure has 1000 trees with interaction depth 1 and shrinkage 0.1.

```
##
## Call:
## CV.SuperLearner(Y = dat$A, X = dplyr::select(X, -A), V = 5, family = binomial(),
##     SL.library = xg.learners$names, parallel = cluster)
##
## Risk is based on: Mean Squared Error
##
## All risk estimates are based on V =  5
##
##             Algorithm     Ave         se     Min     Max
##         Super Learner 0.22346 0.00261402 0.21740 0.23493
##           Discrete SL 0.22390 0.00271965 0.21631 0.23789
##   xgb_100_1_0.001_All 0.24777 0.00018348 0.24747 0.24802
##   xgb_500_1_0.001_All 0.24190 0.00067635 0.24086 0.24334
##  xgb_1000_1_0.001_All 0.23765 0.00100426 0.23603 0.24020
##   xgb_100_2_0.001_All 0.24627 0.00024178 0.24573 0.24688
##   xgb_500_2_0.001_All 0.23713 0.00094832 0.23479 0.24013
##  xgb_1000_2_0.001_All 0.23187 0.00140814 0.22842 0.23650
##   xgb_100_3_0.001_All 0.24555 0.00026578 0.24503 0.24618
##   xgb_500_3_0.001_All 0.23487 0.00104566 0.23232 0.23836
##  xgb_1000_3_0.001_All 0.22938 0.00161507 0.22499 0.23541
##    xgb_100_1_0.01_All 0.23764 0.00100594 0.23599 0.24022
##    xgb_500_1_0.01_All 0.22568 0.00195198 0.22140 0.23232
##   xgb_1000_1_0.01_All 0.22353 0.00235890 0.21793 0.23224
##    xgb_100_2_0.01_All 0.23182 0.00141205 0.22835 0.23649
##    xgb_500_2_0.01_All 0.22429 0.00244228 0.21842 0.23487
##   xgb_1000_2_0.01_All 0.22394 0.00273343 0.21803 0.23641
##    xgb_100_3_0.01_All 0.22935 0.00161844 0.22497 0.23536
##    xgb_500_3_0.01_All 0.22426 0.00269898 0.21666 0.23789
##   xgb_1000_3_0.01_All 0.22534 0.00293086 0.21723 0.24120
##     xgb_100_1_0.1_All 0.22342 0.00238129 0.21781 0.23234
##     xgb_500_1_0.1_All 0.22238 0.00276087 0.21591 0.23238
##    xgb_1000_1_0.1_All 0.22218 0.00281561 0.21589 0.23151
##     xgb_100_2_0.1_All 0.22400 0.00274806 0.21783 0.23745
##     xgb_500_2_0.1_All 0.22621 0.00307871 0.21911 0.24338
##    xgb_1000_2_0.1_All 0.22839 0.00321583 0.22140 0.24710
##     xgb_100_3_0.1_All 0.22542 0.00294070 0.21735 0.24191
##     xgb_500_3_0.1_All 0.23137 0.00338920 0.22389 0.24851
##    xgb_1000_3_0.1_All 0.23757 0.00366981 0.22854 0.25171
```

***KNN for Exposure***

Using 5-fold cross validation, the tuned KNN for exposure has 50 neighbours.

```
##
```

```
## Call:
## CV.SuperLearner(Y = dat$A, X = dplyr::select(X, -A), V = 5, family = binomial(),
##     SL.library = knn.learners$names, parallel = cluster)
##
## Risk is based on: Mean Squared Error
##
## All risk estimates are based on V =  5
##
##      Algorithm     Ave        se     Min     Max
##  Super Learner 0.23030 0.0022112 0.22602 0.23257
##    Discrete SL 0.23025 0.0021891 0.22571 0.23196
##      knn_5_All 0.25249 0.0039917 0.23907 0.26796
##     knn_10_All 0.23947 0.0032181 0.23034 0.25074
##     knn_20_All 0.23332 0.0027632 0.22851 0.24001
##     knn_50_All 0.23025 0.0021891 0.22571 0.23196
##    knn_100_All 0.23217 0.0017565 0.23036 0.23381
```

## Boostrap

We created a function `four_est()` to simultaneously perform boostrap on teh four estimators, and adopt parallel computing (`snow`) to accelerate computing. In `four_est()`, the data is resampled with replacement 100 times.

```r
# Define function to estimate causal effect using the four estimators.
# argument _data_ requires a dataframe
# arguemnt _indices_ requires the index of the rows resampled
# argument _SL.m_ requires a SL library for m
# arguemtn _SL.g_ requires a SL library for g
four_est = function(data,indices,SL.m,SL.g){
  # resampling
  data = data[indices,]
  W = data[,c(1:5,8:9)]
  A = data$A
  Y = data$wage

  ## G-computation
  X = cbind(W,A)
  X1 = cbind(W,1)
  X0 = cbind(W,0)
  colnames(X) = letters[1:ncol(X)]
  colnames(X1) = letters[1:ncol(X1)]
  colnames(X0) = letters[1:ncol(X0)]

  # build SL for m
  SL.outcome = snowSuperLearner(Y = Y, X = X, family = gaussian(),
                              # For a real analysis we would use V = 10.
                              cvControl = list(V=10),
                              # paralell computing: 6 cores
                              cluster = cluster,
                              SL.library = outcome_library)

  ma = predict(SL.outcome, newdata = X, onlySL = TRUE)$pred
  m1 = predict(SL.outcome, newdata = X1, onlySL = TRUE)$pred
  m0 = predict(SL.outcome, newdata = X0, onlySL = TRUE)$pred
  # compute the g-formula
  sub_est = mean(m1-m0)

  ## IPTW
  SL.exposure = snowSuperLearner(Y=A,X=W,family = binomial(),
                                 cvControl = list(V=10),
                                 SL.library = exposure_library,
                                 cluster=cluster)
  gw = SL.exposure$SL.predict
  H_AW = (A==1)/gw - (A==0)/(1-gw)
  H_1W = 1/gw
  H_0W = -1/(1-gw)
  iptw.est = mean(H_AW*Y)

  ## AIPW
  aipw.est = mean(((A==1)/gw)*(Y-m1)+m1 - (((A==0)/(1-gw))*(Y-m0)+m0))

  ## TMLE
  lm_update = lm(Y~-1+offset(ma)+H_AW)
  epsilon = coef(lm_update)
  expY_givenAW_star = ma+epsilon*H_AW
  expY_given1W_star = m1+epsilon*H_1W
  expY_given0W_star = m0+epsilon*H_0W
  tmle.est = mean(expY_given1W_star-expY_given0W_star)

  return(c(sub_est,iptw.est,aipw.est,tmle.est))
}

# Implement Bootstrap with parallel computing
library(snow)
clusterExport(cluster, outcome_library)
clusterExport(cluster, exposure_library)
clusterExport(cluster, "four_est")
results = boot(data=dat, statistic = four_est, R = 100,
               SL.m = outcome_library, SL.g = exposure_library,
               parallel = "snow", cl = cluster)
```