Misinformation Exposure and the False Belief that Trump Won the 2020 U.S. Presidential Election

Ross Dahlke

February 07, 2024

Data

```
main_data <- read_csv("data/main_data.csv") %>%
    mutate(untrustworthy_n_total = untrustworthy_n_pre + untrustworthy_n_test,
           untrustworthy_flag_total = if_else(untrustworthy_n_total > 0, 1, 0))
```

Top-line difference

```
weights::wtd.t.test(
  x = main_data %>% filter(trump_support_pre == 1) %>%  pull(won_election_trump),
  weight = main_data %>% filter(trump_support_pre == 1) %>% pull(weight))
```

```
## $test
## [1] "One Sample Weighted T-Test"
##
## $coefficients
##   t.value       df   p.value
##   19.4667 427.0000    0.0000
##
## $additional
##   Difference         Mean Alternative    Std. Err
##   0.47019208   0.47019208   0.00000000   0.02415366
```

Exposure Descriptives

```
t.test(main_data %>% pull(untrustworthy_flag_test)) %>%
  broom::tidy()
```

```
## # A tibble: 1 x 8
##   estimate statistic   p.value parameter conf.low conf.high method   alternative
##      <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl> <chr>    <chr>
## 1    0.396      28.0 7.75e-133      1193    0.368     0.424 One Sam~ two.sided
```

```
t.test(main_data %>% filter(untrustworthy_n_test > 0) %>% pull(untrustworthy_n_test)) %>%
  broom::tidy()
```

```
## # A tibble: 1 x 8
##   estimate statistic     p.value parameter conf.low conf.high method alternative
```

```
##      <dbl>     <dbl>       <dbl>     <dbl>    <dbl>      <dbl> <chr>  <chr>
## 1     36.5      5.37 0.000000126       472     23.1       49.8 One S~ two.sided
```

Gelbach Decomposition

Binary

```r
save(binary_model1, file = "tables_and_figures/binary_model1")
save(binary_model2, file = "tables_and_figures/binary_model2")
save(binary_model3, file = "tables_and_figures/binary_model3")
save(binary_model4, file = "tables_and_figures/binary_model4")
save(binary_model5, file = "tables_and_figures/binary_model5")
save(binary_model6, file = "tables_and_figures/binary_model6")
save(binary_model7, file = "tables_and_figures/binary_model7")
save(binary_model8, file = "tables_and_figures/binary_model8")
save(binary_model9, file = "tables_and_figures/binary_model9")
save(binary_model10, file = "tables_and_figures/binary_model10")
```

Dosage

```r
outcome <- "won_election_trump"
lm1 <- as.formula(
  paste(outcome,
        paste(c("untrustworthy_n_test"), collapse = " + "),
        sep = " ~ "))
dosage_model1 <- lm(lm1, data = main_data, weights = main_data$weight)

lm2 <- as.formula(
  paste(outcome,
        paste(c("untrustworthy_n_test", "trump_support_pre"), collapse = " + "),
        sep = " ~ "))
dosage_model2 <- lm(lm2, data = main_data, weights = main_data$weight)

lm3 <- as.formula(
  paste(outcome,
        paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre"), collapse = " + "),
        sep = " ~ "))
dosage_model3 <- lm(lm3, data = main_data, weights = main_data$weight)

lm4 <- as.formula(
  paste(outcome,
        paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre", "educ4_college_grad", "educ4
        sep = " ~ "))
dosage_model4 <- lm(lm4, data = main_data, weights = main_data$weight)

lm5 <- as.formula(
  paste(outcome,
        paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre", "educ4_college_grad", "educ4
        sep = " ~ "))
dosage_model5 <- lm(lm5, data = main_data, weights = main_data$weight)

lm6 <- as.formula(
  paste(outcome,
```

```
       paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre", "educ4_college_grad", "educ4
       sep = " ~ "))
dosage_model6 <- lm(lm6, data = main_data, weights = main_data$weight)

lm7 <- as.formula(
  paste(outcome,
       paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre", "educ4_college_grad", "educ4
       sep = " ~ "))
dosage_model7 <- lm(lm7, data = main_data, weights = main_data$weight)

lm8 <- as.formula(
  paste(outcome,
       paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre", "educ4_college_grad", "educ4
       sep = " ~ "))
dosage_model8 <- lm(lm8, data = main_data, weights = main_data$weight)

lm9 <- as.formula(
  paste(outcome,
       paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre","educ4_college_grad", "educ4_
       sep = " ~ "))
dosage_model9 <- lm(lm9, data = main_data, weights = main_data$weight)

lm10 <- as.formula(
  paste(outcome,
       paste(c("untrustworthy_n_test", "total_n_pre", "trump_support_pre", "educ4_college_grad", "educ4
       sep = " ~ "))
dosage_model10 <- lm(lm10, data = main_data, weights = main_data$weight)
```

```
save(dosage_model1, file = "tables_and_figures/dosage_model1")
save(dosage_model2, file = "tables_and_figures/dosage_model2")
save(dosage_model3, file = "tables_and_figures/dosage_model3")
save(dosage_model4, file = "tables_and_figures/dosage_model4")
save(dosage_model5, file = "tables_and_figures/dosage_model5")
save(dosage_model6, file = "tables_and_figures/dosage_model6")
save(dosage_model7, file = "tables_and_figures/dosage_model7")
save(dosage_model8, file = "tables_and_figures/dosage_model8")
save(dosage_model9, file = "tables_and_figures/dosage_model9")
save(dosage_model10, file = "tables_and_figures/dosage_model10")
```

Double ML

Binary

```
dml_data <- DoubleMLData$new(main_data %>% as.data.frame() %>% select(won_election_trump, untrustworthy_
                             y_col = "won_election_trump",
                             d_cols = "untrustworthy_flag_test",
                             x_cols = c("untrustworthy_flag_pre", "total_n_pre", "trump_support_pre", "
```

```
learner <- lrn("regr.ranger", num.trees = 500, mtry = floor(sqrt(12)), max.depth = 5, min.node.size = 2)
ml_g <- learner$clone()
ml_m <- learner$clone()
```

```r
obj_dml_plr <- DoubleMLPLR$new(dml_data, ml_g = ml_g, ml_m = ml_m)
```

```r
obj_dml_plr$fit()
obj_dml_plr
```

```
## ================= DoubleMLPLR Object ==================
##
##
## ------------------ Data summary      ------------------
## Outcome variable: won_election_trump
## Treatment variable(s): untrustworthy_flag_test
## Covariates: untrustworthy_flag_pre, total_n_pre, trump_support_pre, educ4_college_grad, educ4_hs_or_l
## Instrument(s):
## No. Observations: 1194
##
## ------------------ Score & algorithm ------------------
## Score function: partialling out
## DML algorithm: dml2
##
## ------------------ Machine learner   ------------------
## ml_l: regr.ranger
## ml_m: regr.ranger
##
## ------------------ Resampling        ------------------
## No. folds: 5
## No. repeated sample splits: 1
## Apply cross-fitting: TRUE
##
## ------------------ Fit summary       ------------------
##  Estimates and significance testing of the effect of target variables
##                         Estimate. Std. Error t value Pr(>|t|)
## untrustworthy_flag_test   0.05732    0.02195   2.612  0.00901 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Dosage

```r
dml_data <- DoubleMLData$new(main_data %>% as.data.frame() %>% select(won_election_trump, untrustworthy_
                             y_col = "won_election_trump",
                             d_cols = "untrustworthy_n_test",
                             x_cols = c("untrustworthy_flag_pre", "trump_support_pre", "college", "femal
```

```r
learner <- lrn("regr.ranger", num.trees = 500, mtry = floor(sqrt(12)), max.depth = 5, min.node.size = 2
ml_g <- learner$clone()
ml_m <- learner$clone()
```

```r
obj_dml_plr <- DoubleMLPLR$new(dml_data, ml_g = ml_g, ml_m = ml_m)
```

```r
obj_dml_plr$fit()
obj_dml_plr
```

```
## ================= DoubleMLPLR Object ==================
##
##
## ------------------ Data summary      ------------------
```

```
## Outcome variable: won_election_trump
## Treatment variable(s): untrustworthy_n_test
## Covariates: untrustworthy_flag_pre, trump_support_pre, college, female, non_white, knowledge, interes
## Instrument(s):
## No. Observations: 1194
##
## ------------------ Score & algorithm ------------------
## Score function: partialling out
## DML algorithm: dml2
##
## ------------------ Machine learner   ------------------
## ml_l: regr.ranger
## ml_m: regr.ranger
##
## ------------------ Resampling        ------------------
## No. folds: 5
## No. repeated sample splits: 1
## Apply cross-fitting: TRUE
##
## ------------------ Fit summary       ------------------
##  Estimates and significance testing of the effect of target variables
##                      Estimate. Std. Error t value   Pr(>|t|)
## untrustworthy_n_test 0.00034902 0.00006612   5.278 0.00000013 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Causal Forest

```r
set.seed(1)
X_binary <- main_data[, c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad", "educ4_hs
Y_binary <- main_data$won_election_trump
W_binary <- main_data$untrustworthy_flag_test
```

```r
tau.forest_binary <- causal_forest(X_binary, Y_binary, W_binary, num.trees = 4000)
tau.hat_binary <- predict(tau.forest_binary, X_binary, estimate.variance = TRUE)
sigma.hat_binary <- sqrt(tau.hat_binary$variance.estimates)
```

```r
average_treatment_effect(tau.forest_binary, target.sample = "overlap")
```

```
##   estimate    std.err
## 0.04194356 0.01966475
```

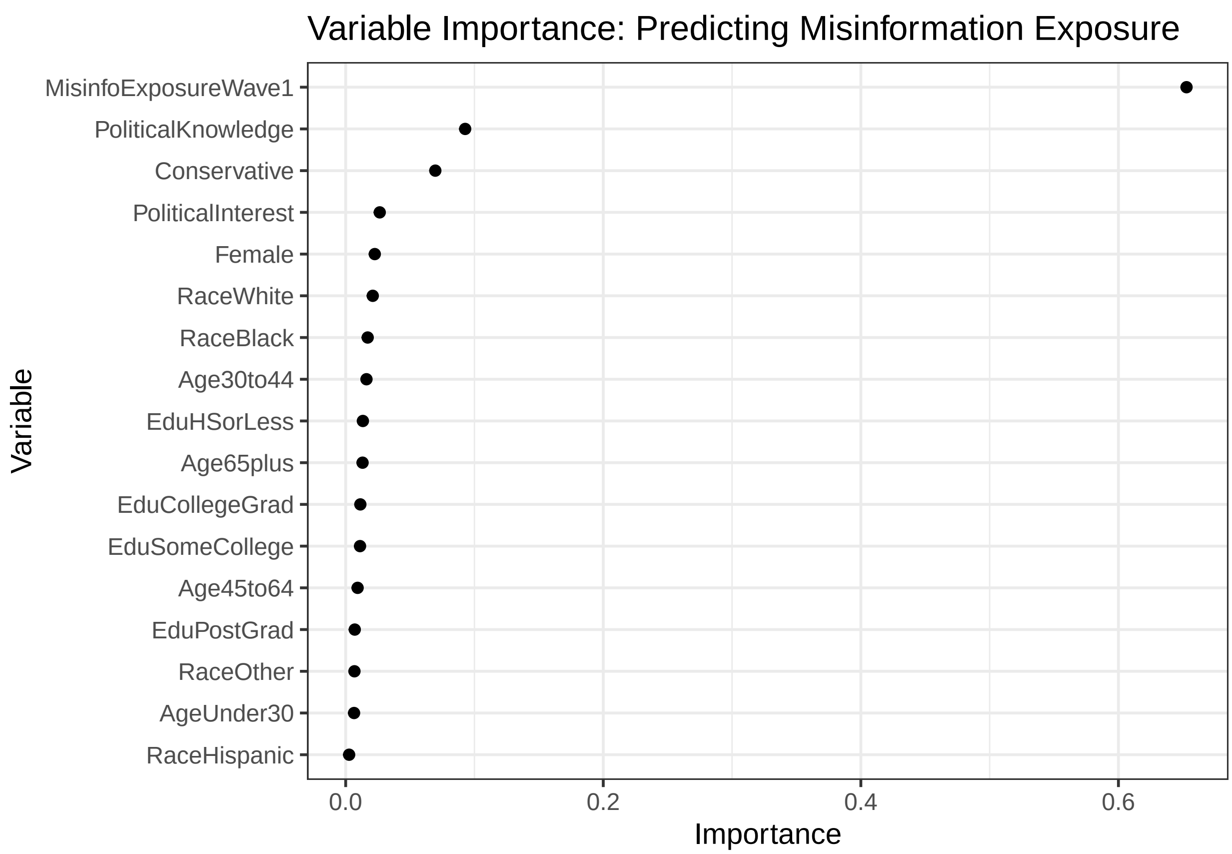
```r
binary_w_regression <- regression_forest(X_binary, W_binary, num.trees = 4000)

test_calibration(binary_w_regression)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                               Estimate Std. Error t value
## mean.forest.prediction         0.998751   0.031050  32.166
## differential.forest.prediction 0.951830   0.045283  21.020
```

```
##                                                 Pr(>t)
## mean.forest.prediction         < 0.00000000000000022 ***
## differential.forest.prediction < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

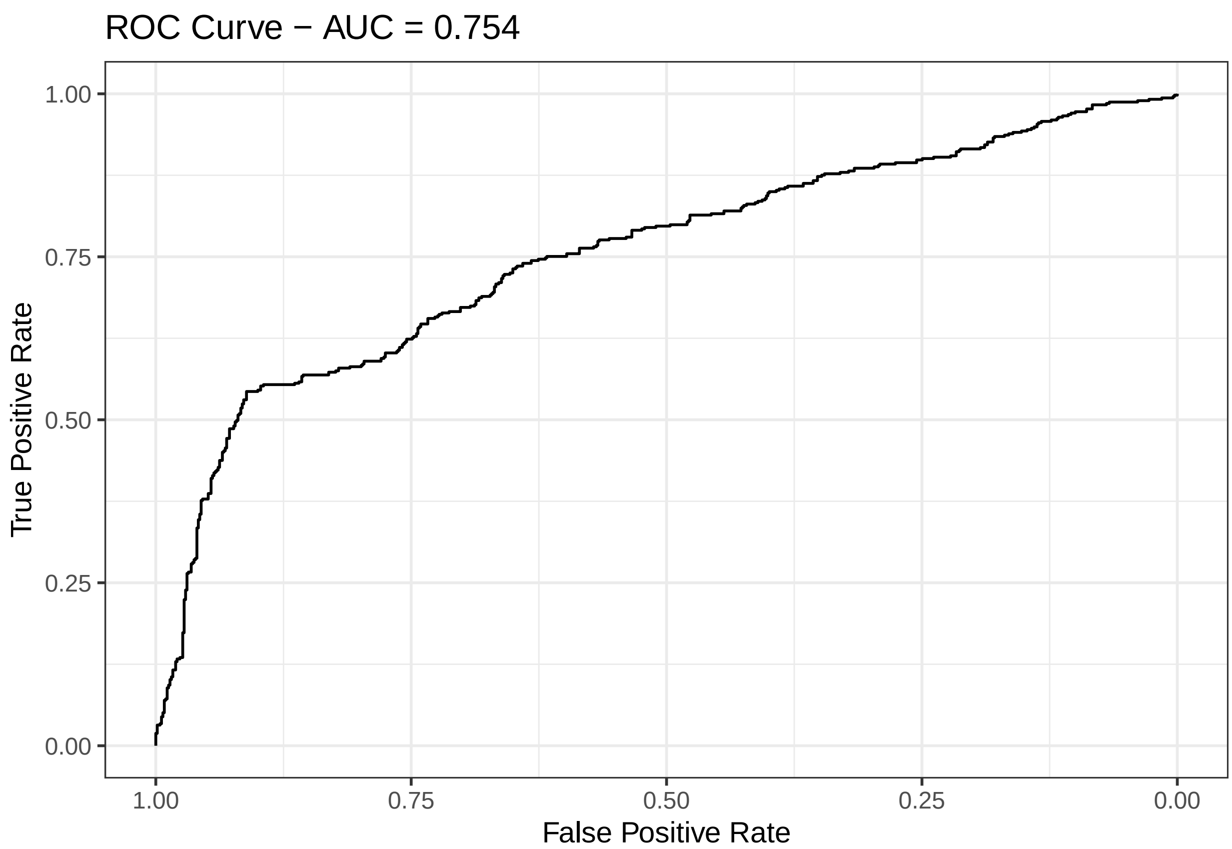
```r
name_conversion_table <- tibble(x = c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad
                                name = c("MisinfoExposureWave1", "Conservative", "EduCollegeGrad", "EduH
```

```r
tibble(x = binary_w_regression$X.orig %>% colnames(),
       importance  = variable_importance(binary_w_regression)) %>%
  left_join(name_conversion_table) %>%
  ggplot(aes(importance, reorder(name, importance))) +
  geom_point() +
  labs(title = "Variable Importance: Predicting Misinformation Exposure",
       x = "Importance",
       y = "Variable") +
  theme_bw()
```

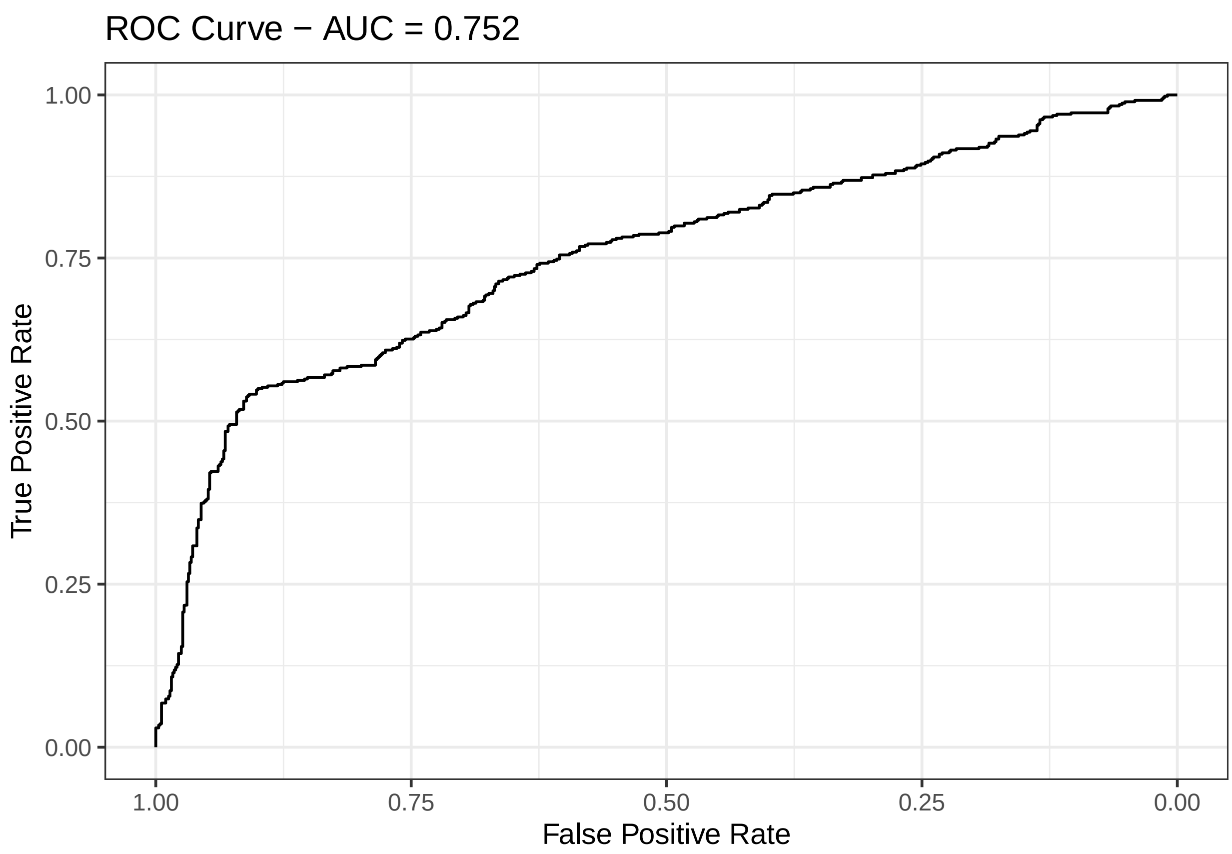



```r
tibble(w_hat = binary_w_regression$predictions,
       w = binary_w_regression$Y.orig) %>%
  # Compute the ROC curve
  {
    roc_curve <- roc(response = .$w, predictor = .$w_hat)
    auc_value <- auc(roc_curve)
    list(roc_curve = roc_curve, auc_value = auc_value)
  } %>%
```

```r
  # Plot ROC curve
  {
    ggroc(.$roc_curve) +
    ggtitle(paste("ROC Curve - AUC =", round(.$auc_value, 3))) +
    xlab("False Positive Rate") +
    ylab("True Positive Rate") +
    theme_bw()
  }
```




```r
tibble(w_hat = tau.forest_binary$W.hat,
       w = tau.forest_binary$W.orig) %>%
  # Compute the ROC curve
  {
    roc_curve <- roc(response = .$w, predictor = .$w_hat)
    auc_value <- auc(roc_curve)
    list(roc_curve = roc_curve, auc_value = auc_value)
  } %>%
  # Plot ROC curve
  {
    ggroc(.$roc_curve) +
    ggtitle(paste("ROC Curve - AUC =", round(.$auc_value, 3))) +
    xlab("False Positive Rate") +
    ylab("True Positive Rate") +
    theme_bw()
  }
```




```
binary_w_regression
```

```
## GRF forest object of type regression_forest
## Number of trees: 4000
## Number of training samples: 1194
## Variable importance:
##     1     2     3     4     5     6     7     8     9    10    11    12    13
## 0.653 0.070 0.011 0.013 0.007 0.011 0.023 0.017 0.003 0.007 0.021 0.093 0.026
##    14    15    16    17
## 0.007 0.016 0.009 0.013
```

r-value analysis

```
r_val <- tipr::r_value(effect_observed = 0.04194356, se = 0.01966475, df = (1194-17))

residuals_treatment_effect <- tau.hat_binary$predictions

var_residuals <- var(residuals_treatment_effect)

rsq_values <- apply(X_binary, 2, function(col) {
  mod <- lm(residuals_treatment_effect ~ col)
  summary(mod)$r.squared
})

is_potential_confounder <- ifelse(rsq_values > r_val, "Yes", "No")

# Create dataframe with results
```

```r
res_var_binary <- tibble(
  variable = names(rsq_values),
  residual_variance = rsq_values,
  potential_confounder = is_potential_confounder
)
```

```r
main_data_predictions_binary <- main_data %>%
  cbind(tau.hat_binary)
```

```r
average_treatment_effect(tau.forest_binary, target.sample = "overlap", subset = main_data$trump_support
```

```
##   estimate    std.err
## 0.12587578 0.05555805
```

```r
r_val <- tipr::r_value(effect_observed = 0.12587578, se = 0.05555805, df = (432-17))

residuals_treatment_effect <- tau.hat_binary$predictions[main_data$trump_support_pre == 1]

var_residuals <- var(residuals_treatment_effect)

rsq_values <- apply(X_binary[main_data$trump_support_pre == 1, ], 2, function(col) {
  mod <- lm(residuals_treatment_effect ~ col)
  summary(mod)$r.squared
})

is_potential_confounder <- ifelse(rsq_values > r_val, "Yes", "No")

# Create dataframe with results
res_var_binary_trump <- tibble(
  variable = names(rsq_values),
  residual_variance = rsq_values,
  potential_confounder = is_potential_confounder
)
```

```r
average_treatment_effect(tau.forest_binary, target.sample = "overlap", subset = main_data$trump_support
```

```
##    estimate     std.err
## -0.00177385  0.00884615
```

```r
test_calibration(tau.forest_binary)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                                Estimate Std. Error t value  Pr(>t)
## mean.forest.prediction          0.961393   0.464795  2.0684 0.01941 *
## differential.forest.prediction -0.021454   0.543922 -0.0394 0.51573
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Dosage

```r
X_dosage <- main_data[, c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad", "educ4_h
Y_dosage <- main_data$won_election_trump
```

```r
W_dosage <- main_data$untrustworthy_n_test
```

```r
tau.forest_dosage <- causal_forest(X_dosage, Y_dosage, W_dosage, num.trees = 4000)
tau.hat_dosage <- predict(tau.forest_dosage, X_dosage, estimate.variance = TRUE)
sigma.hat_dosage <- sqrt(tau.hat_dosage$variance.estimates)
```

```r
average_treatment_effect(tau.forest_dosage, target.sample = "overlap")
```

```
##      estimate       std.err
## 0.00032778941 0.00007799424
```

```r
r_val <- tipr::r_value(effect_observed = 0.00033741188, se = 0.00007802666, df = (1194-17))

residuals_treatment_effect <- tau.hat_dosage$predictions

var_residuals <- var(residuals_treatment_effect)

rsq_values <- apply(X_dosage, 2, function(col) {
  mod <- lm(residuals_treatment_effect ~ col)
  summary(mod)$r.squared
})

is_potential_confounder <- ifelse(rsq_values > r_val, "Yes", "No")

# Create dataframe with results
res_var_dosage <- tibble(
  variable = names(rsq_values),
  residual_variance = rsq_values,
  potential_confounder = is_potential_confounder
)
```

```r
main_data_predictions_dosage <- main_data %>%
  cbind(tau.hat_dosage)
```

```r
average_treatment_effect(tau.forest_dosage, target.sample = "overlap", subset = main_data$trump_support
```

```
##     estimate      std.err
## 0.0003442079 0.0000963162
```

```r
r_val <- tipr::r_value(effect_observed = 0.00035346681, se = 0.00009593705, df = (432-17))

residuals_treatment_effect <- tau.hat_dosage$predictions[main_data$trump_support_pre == 1]

var_residuals <- var(residuals_treatment_effect)

rsq_values <- apply(X_dosage[main_data$trump_support_pre == 1, ], 2, function(col) {
  mod <- lm(residuals_treatment_effect ~ col)
  summary(mod)$r.squared
})

is_potential_confounder <- ifelse(rsq_values > r_val, "Yes", "No")

# Create dataframe with results
res_var_dosage_trump <- tibble(
  variable = names(rsq_values),
  residual_variance = rsq_values,
```

```r
  potential_confounder = is_potential_confounder
)
```

```r
average_treatment_effect(tau.forest_dosage, target.sample = "overlap", subset = main_data$trump_support
```

```
##      estimate       std.err
## 0.00007672086 0.00012514945
```

```r
test_calibration(tau.forest_dosage)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                                Estimate Std. Error t value  Pr(>t)
## mean.forest.prediction          1.03303    0.48708  2.1208 0.01707 *
## differential.forest.prediction -0.38607    0.75223 -0.5132 0.69606
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
cbind(main_data,
      y_hat = tau.forest_binary$Y.hat,
      tau.hat_binary) %>%
  mutate(treated = if_else(untrustworthy_n_test > 0, 1, 0),
         y_hat_untreated = case_when(
           treated == 1 ~ y_hat - predictions,
           treated == 0 ~ y_hat
         ),
         y_hat_treated = case_when(
           treated == 1 ~ y_hat,
           treated == 0 ~ y_hat + predictions
         ),
         rank = rank(y_hat_untreated),
         ) %>%
  ggplot(aes(x = rank, color = as.factor(trump_support))) +
  geom_point(aes(y = y_hat_untreated), size = .5) +
  geom_point(aes(y = y_hat_treated), size = .5) +
  geom_segment(aes(y = y_hat_untreated, yend = y_hat_treated, xend = rank), alpha = .2) +
  scale_color_manual(values = c("darkblue", "darkred")) +
  theme_bw()
```

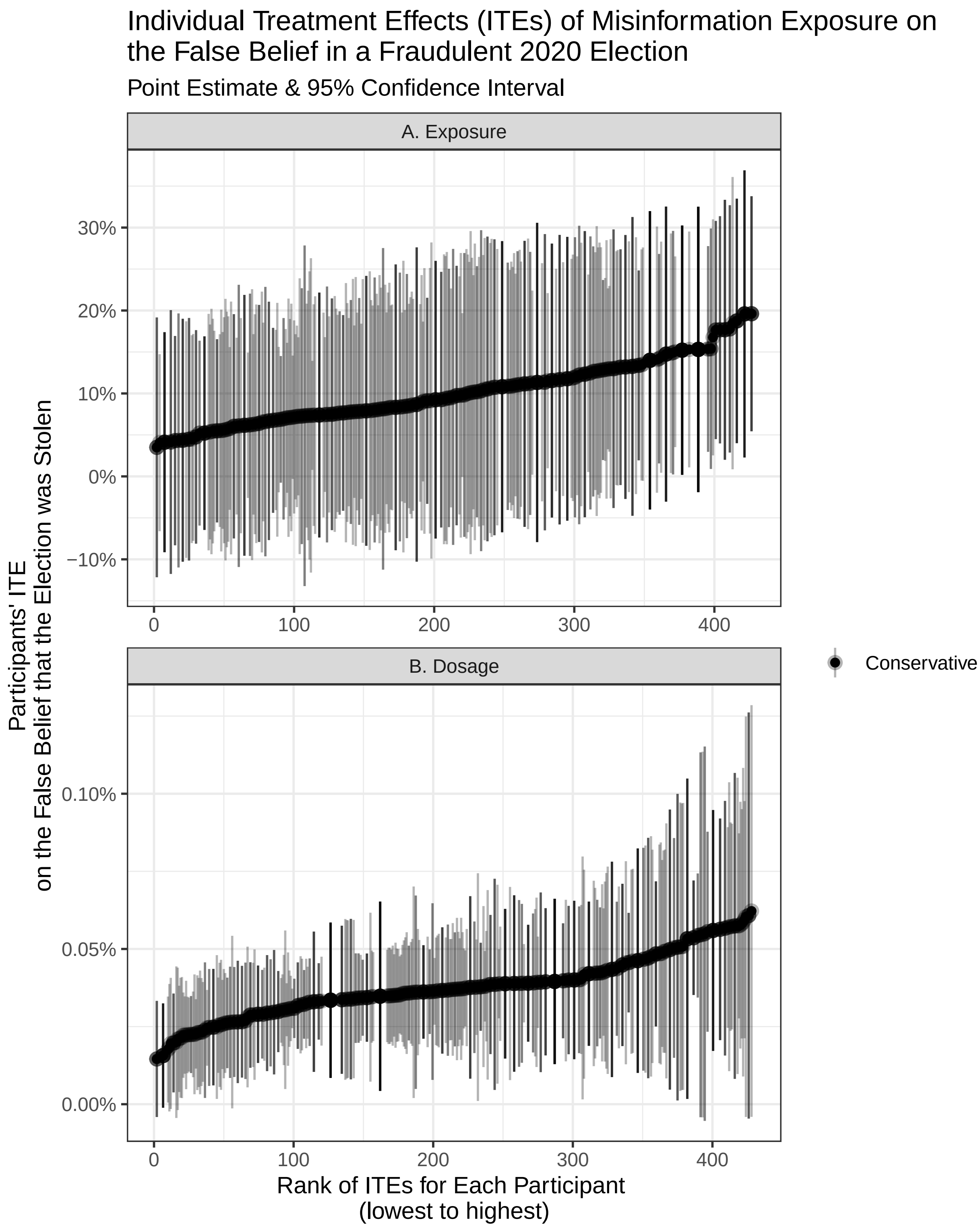


Figure 1: Plot of estimated conditional average differences and 95% confidence interval of misinformation exposure on the belief that Donald Trump won the 2020 U.S. Presidential Election for each individual in our sample. Participants are ordered along the x-axis in order from lowest estimated conditional difference to the highest. The y-axis is the estimated conditional average difference.

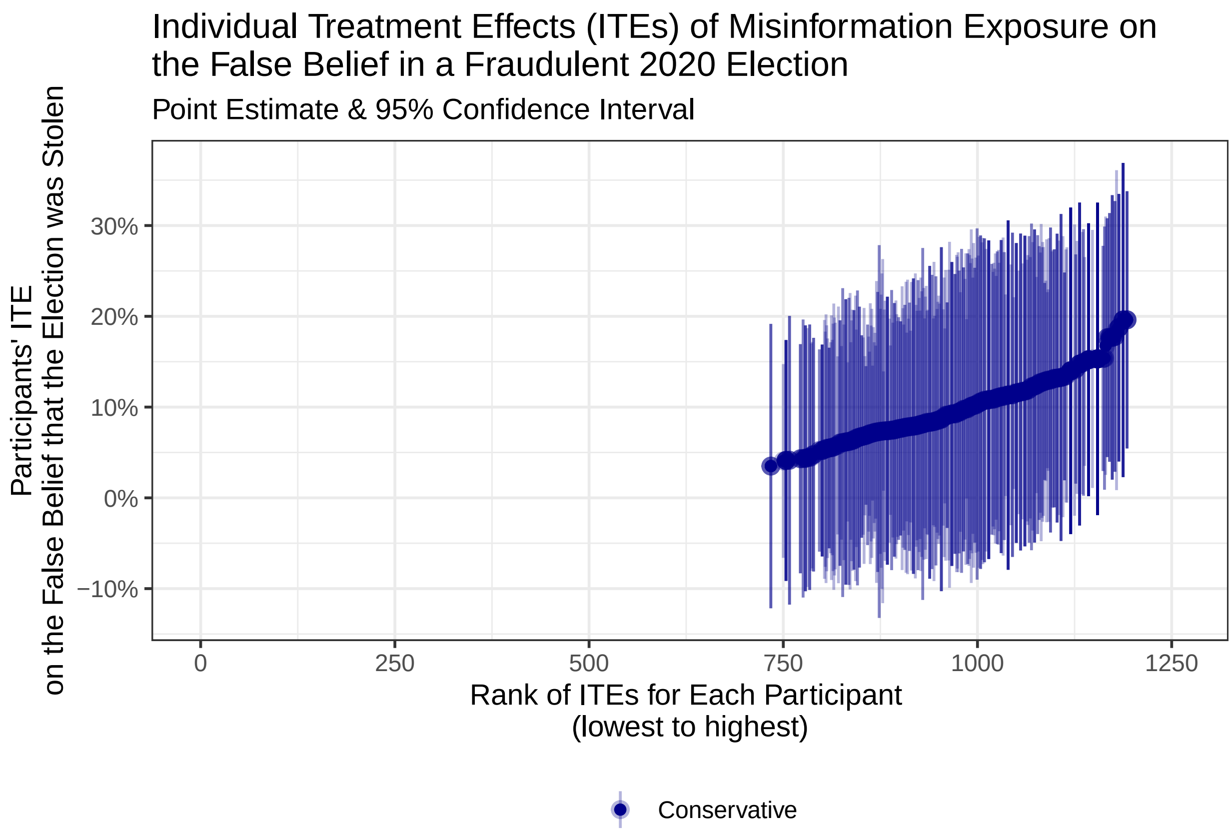
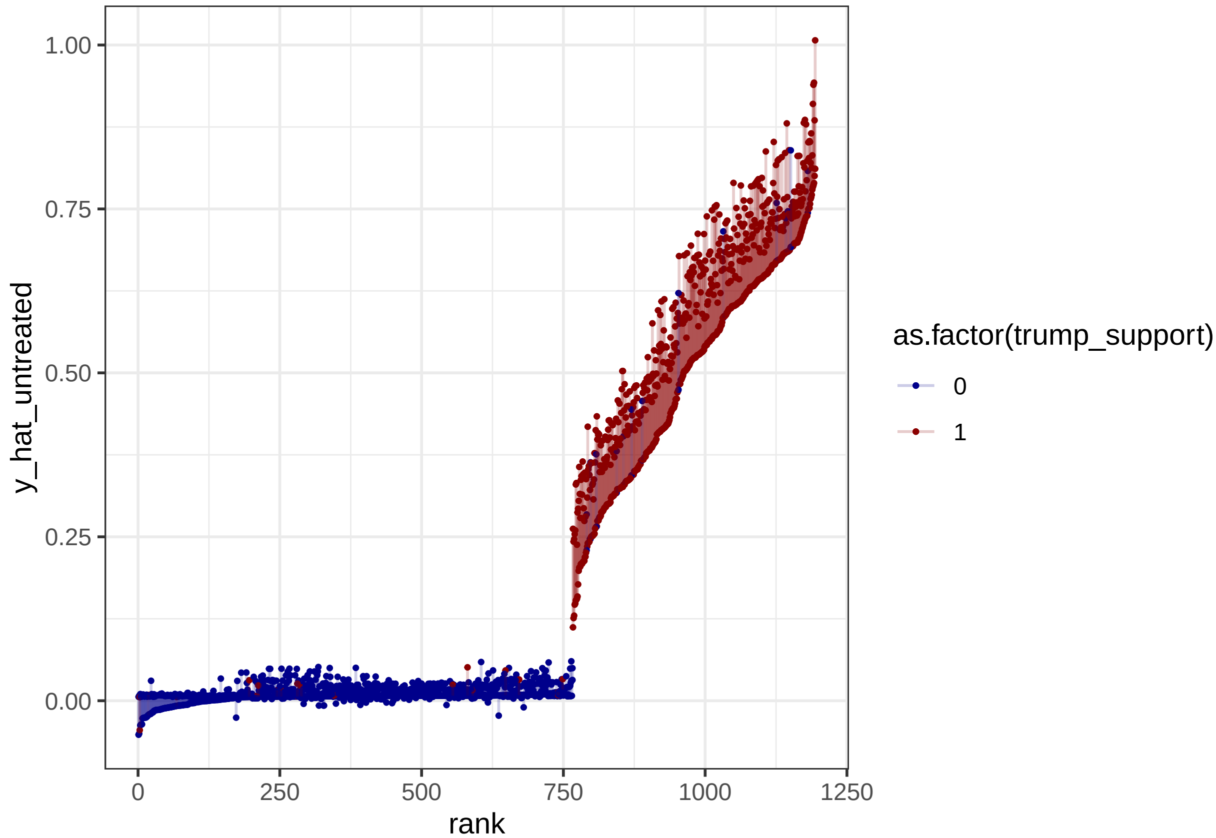
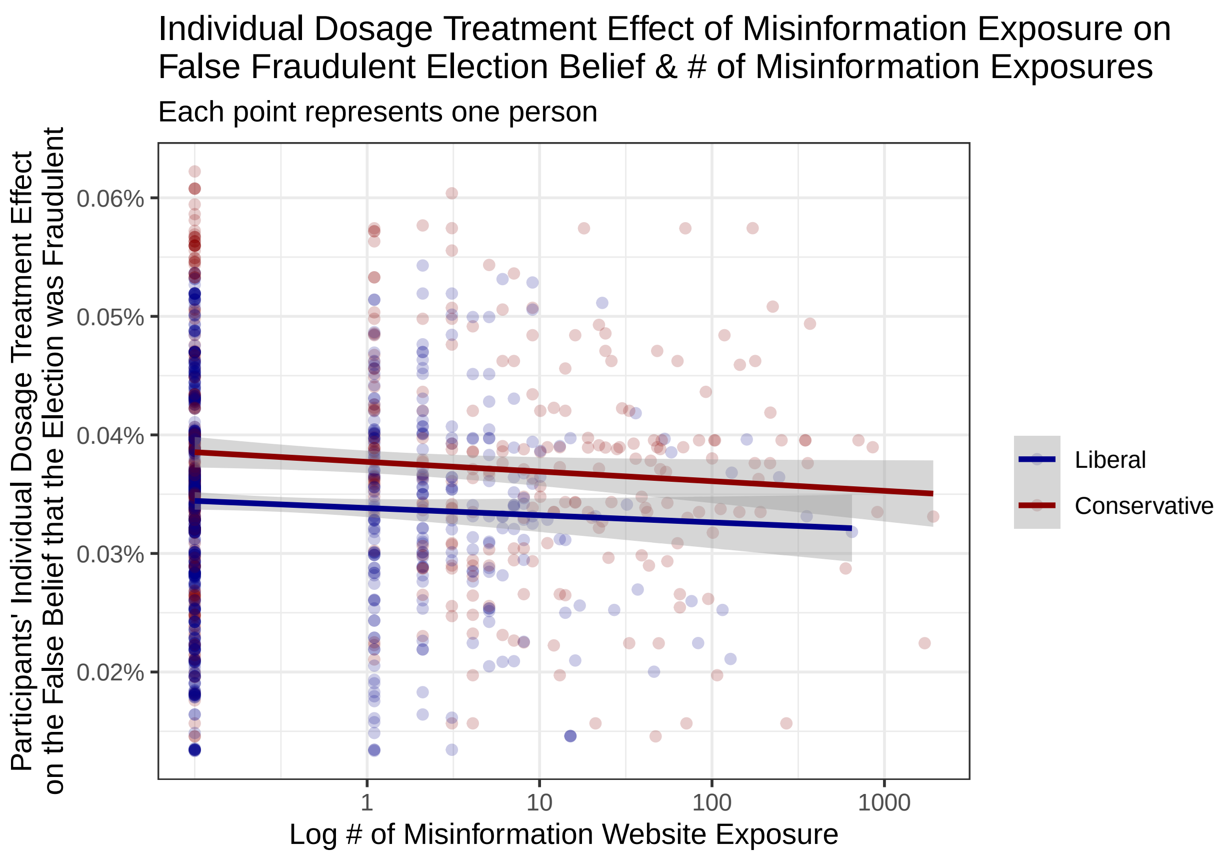


Figure 2: Plot of estimated conditional average differences and 95% confidence interval of misinformation exposure on the belief that Donald Trump won the 2020 U.S. Presidential Election for each individual in our sample. Participants are ordered along the x-axis in order from lowest estimated conditional difference to the highest. The y-axis is the estimated conditional average difference.

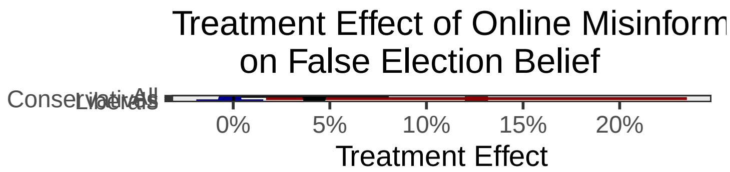



```
main_data %>%
  cbind(tau.hat_dosage) %>%
  mutate(trump_support_factor = if_else(trump_support_pre == 0, "Liberal", "Conservative"),
         trump_support_factor = factor(trump_support_factor, levels = c("Liberal", "Conservative")),
         untrustworthy_n_test = untrustworthy_n_test + .1) %>%
  ggplot(aes(untrustworthy_n_test, predictions, color = trump_support_factor)) +
  geom_point(alpha = .2) +
  geom_smooth(method = "lm") +
  scale_color_manual(values = c("darkblue", "darkred")) +
  scale_x_log10(breaks = c(0, 1, 10, 100, 1000)) +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(title = "Individual Dosage Treatment Effect of Misinformation Exposure on
False Fraudulent Election Belief & # of Misinformation Exposures",
subtitle = "Each point represents one person",
x = "Log # of Misinformation Website Exposure",
y = "Participants' Individual Dosage Treatment Effect
on the False Belief that the Election was Fraudulent",
color = "") +
  theme_bw()
```



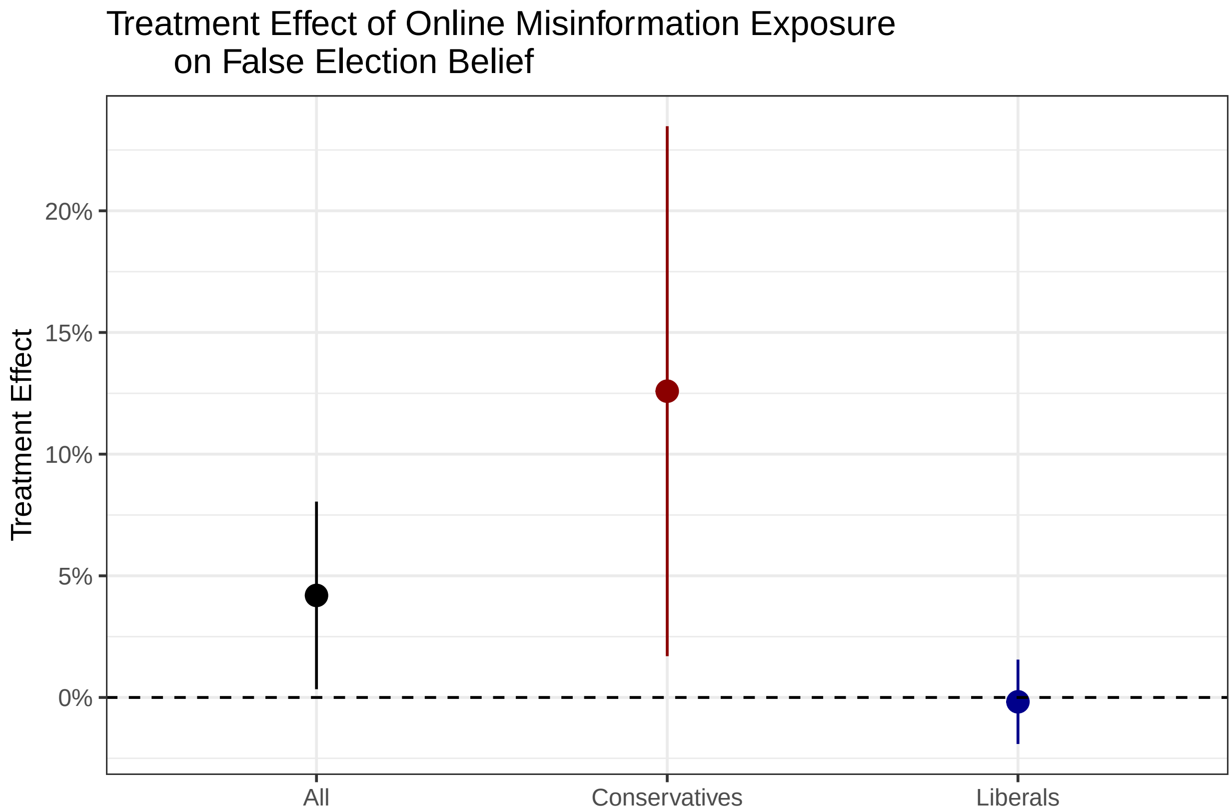

```
ggsave("tables_and_figures/dosage_graph.pdf")
```

```
tribble(
  ~group, ~point, ~low, ~high,
  "All", 0.04194356, 0.00340065, 0.08048647,
  "Liberals", -0.00177385, -0.0191123, 0.0155646,
  "Conservatives", 0.12587578, 0.016982, 0.2347696
) %>%
  mutate(group = factor(group, levels = c("Liberals", "Conservatives", "All"))) %>%
  ggplot(aes(group, point, ymin = low, ymax = high, color = group)) +
  geom_point(size = .7) +
  geom_pointrange(size = .7) +
  scale_color_manual(values = c("darkblue", "darkred", "black")) +
  coord_flip() +
  geom_hline(aes(yintercept = 0), linetype = "dashed") +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(title = "Treatment Effect of Online Misinformation Exposure
       on False Election Belief",
       x = "",
       y = "Treatment Effect") +
  theme_bw() +
  theme(legend.position = "none")
```



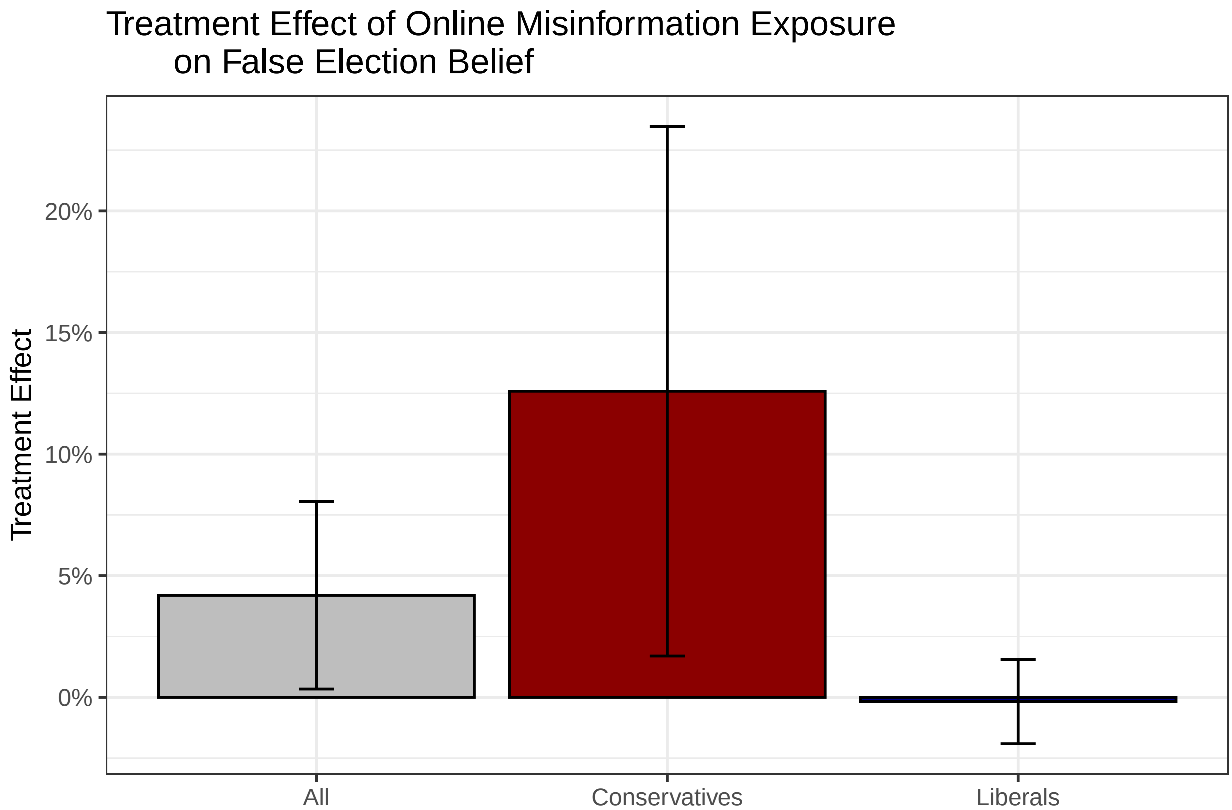

```
ggsave("tables_and_figures/ate_graph.pdf")
```

```
tribble(
  ~group, ~point, ~low, ~high,
  "All", 0.04194356, 0.00340065, 0.08048647,
  "Liberals", -0.00177385, -0.0191123, 0.0155646,
  "Conservatives", 0.12587578, 0.016982, 0.2347696
) %>%
  ggplot(aes(group, point, ymin = low, ymax = high, color = group)) +
  geom_point(size = .7) +
  geom_pointrange(size = .7) +
  scale_color_manual(values = c("black", "darkred", "darkblue")) +
  geom_hline(aes(yintercept = 0), linetype = "dashed") +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(title = "Treatment Effect of Online Misinformation Exposure
       on False Election Belief",
       x = "",
       y = "Treatment Effect") +
  theme_bw() +
  theme(legend.position = "none")
```




```
ggsave("tables_and_figures/ate_graph_2.pdf")
```

```
tribble(
  ~group, ~point, ~low, ~high,
  "All", 0.04194356, 0.00340065, 0.08048647,
  "Liberals", -0.00177385, -0.0191123, 0.0155646,
  "Conservatives", 0.12587578, 0.016982, 0.2347696
) %>%
  ggplot(aes(group, point, ymin = low, ymax = high, fill = group)) +
  geom_col(color = "black") +
  geom_errorbar(width = .1) +
  scale_fill_manual(values = c("grey", "darkred", "darkblue")) +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(title = "Treatment Effect of Online Misinformation Exposure
       on False Election Belief",
       x = "",
       y = "Treatment Effect") +
  theme_bw() +
  theme(legend.position = "none")
```

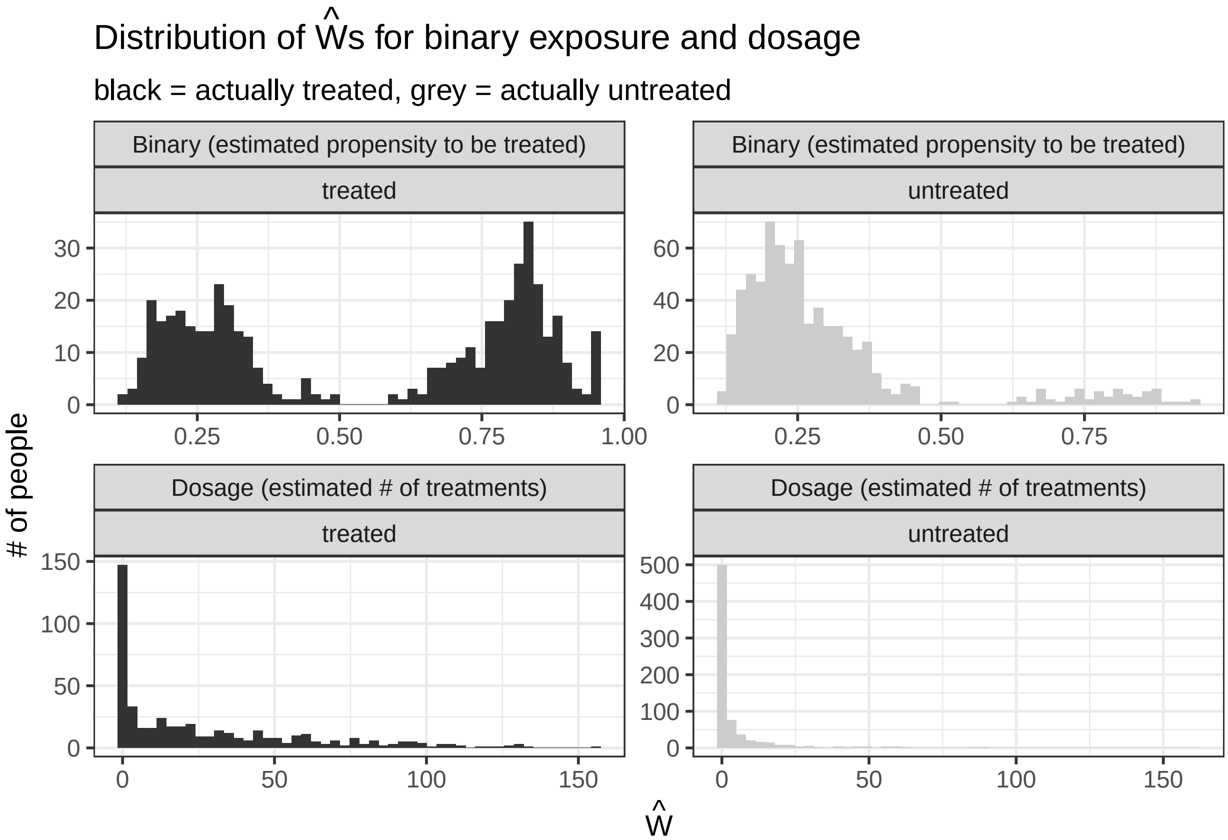



```
ggsave("tables_and_figures/ate_graph_3.pdf")
```

```
diminishing_effects_model1 <- main_data %>%
  cbind(tau.hat_dosage) %>%
  lm(predictions ~ untrustworthy_n_test * trump_support_pre, data = .)

diminishing_effects_model2 <- main_data %>%
  cbind(tau.hat_dosage) %>%
  lm(predictions ~ log(untrustworthy_n_test + .01) * trump_support_pre, data = .)


save(diminishing_effects_model1, file = "tables_and_figures/diminishing_effects_model1")
save(diminishing_effects_model2, file = "tables_and_figures/diminishing_effects_model2")
```

```
main_data %>%
  cbind(w_hat_binary = tau.forest_binary$W.hat) %>%
  cbind(w_hat_dosage = tau.forest_dosage$W.hat) %>%
  pivot_longer(cols = c(w_hat_binary, w_hat_dosage)) %>%
  mutate(treated = if_else(untrustworthy_flag_test == 1, "treated", "untreated"),
         name = if_else(name == "w_hat_binary", "Binary (estimated propensity to be treated)", "Dosage
  ggplot(aes(value, fill = treated)) +
  geom_histogram(bins = 50) +
  scale_fill_grey() +
  theme_bw() +
  theme(legend.position = "none") +
  facet_wrap(name~treated, scales = "free", ncol = 2) +
  labs(title = latex2exp::TeX(r'(Distribution of \hat{W}s for binary exposure and dosage)'),
```

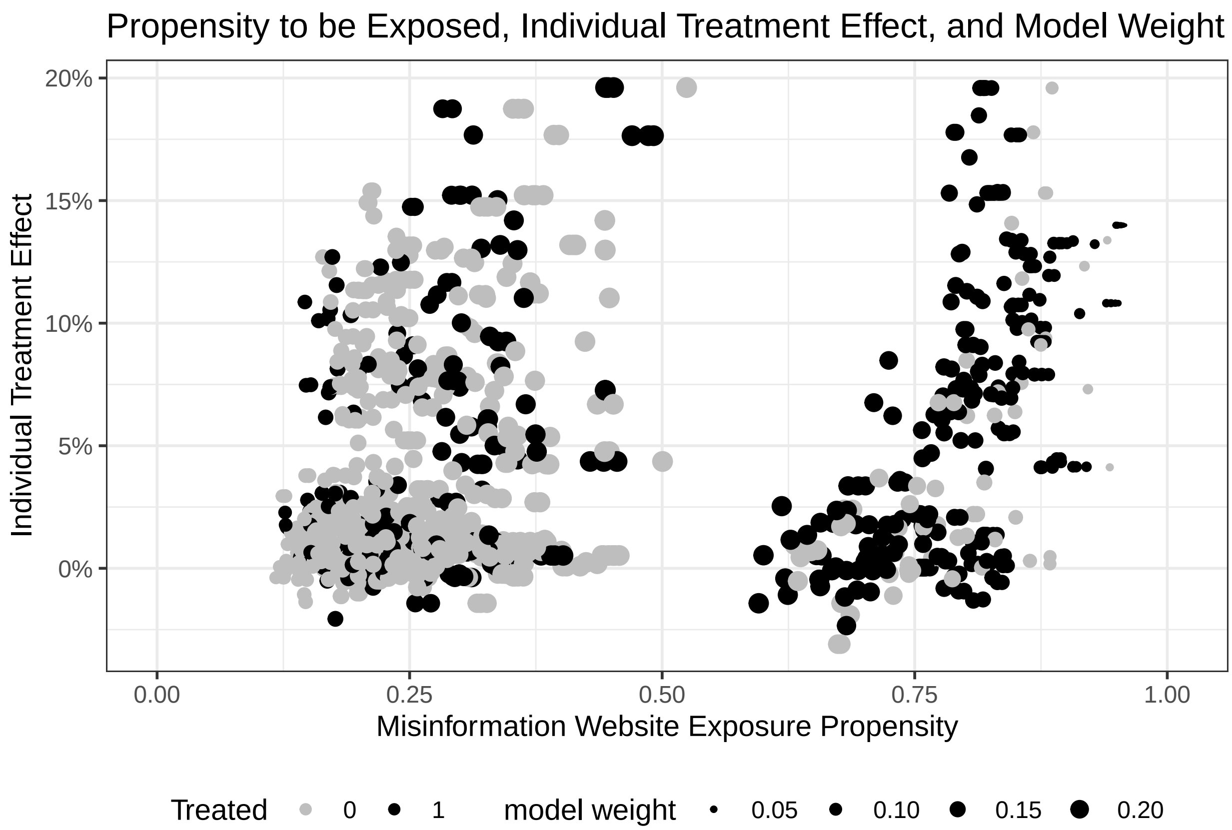
```
        subtitle = "black = actually treated, grey = actually untreated",
        x = latex2exp::TeX(r'(\hat{W})'),
        y = "# of people")
```




```
ggsave("tables_and_figures/w_hat_distributions.pdf")
```

```
main_data %>%
  cbind(tau.hat_binary) %>%
  cbind(propensity_point = tau.forest_binary$W.hat) %>%
  mutate(effect_point = predictions,
         effect_low = predictions - 1.96 * variance.estimates,
         effect_high = predictions + 1.96 * variance.estimates,
         model_weight = propensity_point * (1 - propensity_point)) %>%
  ggplot(aes(propensity_point, effect_point, color = as.factor(untrustworthy_flag_test), size = model_w
  geom_point() +
  scale_color_manual(values = c("grey", "black")) +
  scale_y_continuous(labels = scales::percent_format()) +
  scale_size(range = c(0, 3)) +
  scale_x_continuous(limits = c(0, 1.01),
                     breaks = c(0, .25, .5, .75, 1)) +
  labs(title = "Propensity to be Exposed, Individual Treatment Effect, and Model Weight",
x = "Misinformation Website Exposure Propensity",
y = "Individual Treatment Effect",
color = "Treated",
size = "model weight") +
  theme_bw() +
```

```r
  theme(legend.position = "bottom")
```




```r
ggsave("tables_and_figures/model_weights.pdf")
```

Oster Omitted Variable Bias

Binary

```r
f01c <- won_election_trump ~ untrustworthy_flag_test + untrustworthy_flag_pre + total_n_pre + trump_supp

f01u <- won_election_trump ~ untrustworthy_flag_test

fit01c <- lm(f01c, data = main_data, weights = main_data$weight)

main_data$infit01c <- is.element(rownames(main_data), names(fit01c$residuals))

fit01u <- lm(f01u, data = main_data, subset = infit01c, weights = main_data$weight)

z <- oster(fit01u, fit01c, "untrustworthy_flag_test")
b13 <- oster(fit01u, fit01c, "untrustworthy_flag_test", rm = 1.3)$beta

round(c(z$input$beta_o, z$input$beta_tilde, z$beta, b13, z$rmax), 6)
```

```
## [1]  0.173302  0.057813 -0.148747  0.000965  0.494875
```

Dosage

```r
f01c <- won_election_trump ~ untrustworthy_n_test + untrustworthy_flag_pre + total_n_pre + trump_suppor

f01u <- won_election_trump ~ untrustworthy_n_test

fit01c <- lm(f01c, data = main_data, weights = main_data$weight)

main_data$infit01c <- is.element(rownames(main_data), names(fit01c$residuals))

fit01u <- lm(f01u, data = main_data, subset = infit01c, weights = main_data$weight)

z <- oster(fit01u, fit01c, "untrustworthy_n_test")
b13 <- oster(fit01u, fit01c, "untrustworthy_n_test", rm = 1.3)$beta

round(c(z$input$beta_o, z$input$beta_tilde, z$beta, b13, z$rmax), 6)
```

```
## [1]  0.000809  0.000381 -0.000069  0.000239  0.683844
```

```r
temp <- robomit::o_beta_boot(y = "won_election_trump",
                x = "untrustworthy_n_test",
                con = "untrustworthy_flag_pre + total_n_pre + trump_support_pre + educ4_college_grad + 
                data = main_data,
                R2max = 1.3*0.3798,
                type = "lm",
                w = "weight",
                delta = 1,
                sim = 1000,
                obs = 1000,
                rep = T)

mean(temp$`beta*`)
```

```
## [1] 0.002398048
```

```r
temp <- robomit::o_beta_boot(y = "won_election_trump",
                x = "untrustworthy_flag_test",
                con = "untrustworthy_flag_pre + total_n_pre + trump_support_pre + educ4_college_grad + 
                data = main_data,
                R2max = 1.3*0.3821,
                type = "lm",
                w = "weight",
                delta = 1,
                sim = 1000,
                obs = 1000,
                rep = T,
                useed = 1)

median(temp$`beta*`)
```

```
## [1] 0.032089
```

```r
temp <- robomit::o_beta_boot(y = "won_election_trump",
                x = "untrustworthy_n_test",
                con = "untrustworthy_flag_pre + total_n_pre + trump_support_pre + educ4_college_grad + 
                data = main_data,
```

```
                R2max = 1.3*0.3838,
                type = "lm",
                w = "weight",
                delta = 1,
                sim = 1000,
                obs = 1000,
                rep = T,
                useed = 1)

median(temp$`beta*`)
```

```
## [1] 0.000334
```

r-value table

```
res_var_binary %>%
  bind_cols(res_var_binary_trump %>% select(-variable)) %>%
  bind_cols(res_var_dosage %>% select(-variable)) %>%
  bind_cols(res_var_dosage_trump %>% select(-variable)) %>%
  mutate_at(vars(starts_with("residual_variance")), ~round(., 3)) %>%
  mutate(variable = case_when(
    variable == "untrustworthy_flag_pre" ~ "MisinfoExposureWave1",
    variable == "trump_support_pre"      ~ "Conservative",
    variable == "educ4_college_grad"     ~ "EduCollegeGrad",
    variable == "educ4_hs_or_less"       ~ "EduHSorLess",
    variable == "educ4_postgrad"         ~ "EduPostGrad",
    variable == "educ4_some_college"     ~ "EduSomeCollege",
    variable == "female"                 ~ "Female",
    variable == "race4_black"            ~ "RaceBlack",
    variable == "race4_hispanic"         ~ "RaceHispanic",
    variable == "race4_other"            ~ "RaceOther",
    variable == "race4_white"            ~ "RaceWhite",
    variable == "knowledge"              ~ "PoliticalKnowledge",
    variable == "interest"               ~ "PoliticalInterest",
    variable == "age4_under_30"          ~ "AgeUnder30",
    variable == "age4_30_44"             ~ "Age30to44",
    variable == "age4_45_64"             ~ "Age45to65",
    variable == "age4_65"                ~ "Age65plus",
    TRUE                                ~ variable # keep the original name if it's not one of the abov
  )) %>%
  knitr::kable("latex", booktabs = TRUE,
               col.names = c("variable", "res. var.", "confounder", "res. var.", "confounder", "res. va
               escape = FALSE) %>%
  kableExtra::add_header_above(c(" " = 1, "Exposure" = 2, "Exposure - Trump Supporters" = 2, "Dosage" =
  kableExtra::add_footnote("",
          escape = FALSE,
          threeparttable = TRUE) %>%
  kableExtra::kable_styling(latex_options = "scale_down")
```

variable	Exposure res. var.	Exposure confounder	Exposure - Trump Supporters res. var.	Exposure - Trump Supporters confounder	Dosage res. var.	Dosage confounder	Dosage - Trump Supporters res. var.	Dosage - Trump Supporters confounder
MisinfoExposureWave1	0.040	No	0.001	No	0.000	No	0.006	No
Conservative	0.749	Yes	0.000	No	0.032	No	0.000	No
EduCollegeGrad	0.038	No	0.128	Yes	0.018	No	0.034	No
EduHSorLess	0.001	No	0.120	Yes	0.025	No	0.069	No
EduPostGrad	0.043	No	0.065	No	0.335	Yes	0.233	Yes
EduSomeCollege	0.114	Yes	0.525	Yes	0.428	Yes	0.425	Yes
Female	0.001	No	0.004	No	0.021	No	0.097	No
RaceBlack	0.028	No	0.001	No	0.004	No	0.000	No
RaceHispanic	0.004	No	0.009	No	0.004	No	0.009	No
RaceOther	0.000	No	0.005	No	0.001	No	0.004	No
RaceWhite	0.028	No	0.010	No	0.009	No	0.013	No
PoliticalKnowledge	0.000	No	0.019	No	0.127	Yes	0.157	No
PoliticalInterest	0.005	No	0.002	No	0.032	No	0.034	No
AgeUnder30	0.009	No	0.000	No	0.000	No	0.000	No
Age30to44	0.003	No	0.015	No	0.000	No	0.000	No
Age45to65	0.005	No	0.021	No	0.079	No	0.092	No
Age65plus	0.034	No	0.059	No	0.100	No	0.117	No

a

```
weights::wtd.t.test(x = main_data$untrustworthy_n_total,
                    weight = main_data$weight)
```

```
## $test
## [1] "One Sample Weighted T-Test"
##
## $coefficients
##             t.value                  df             p.value
##    5.41157750431008 1193.00000000000000    0.00000007545303
##
## $additional
##  Difference        Mean Alternative    Std. Err
##   15.422216   15.422216    0.000000    2.849856
```

```
weights::wtd.t.test(x = main_data %>% filter(untrustworthy_flag_total == 1) %>% pull(untrustworthy_n_to
                    weight = main_data %>% filter(untrustworthy_flag_total == 1) %>% pull(weight))
```

```
## $test
## [1] "One Sample Weighted T-Test"
##
## $coefficients
##            t.value                  df            p.value
##   5.78542153909954 534.00000000000000   0.00000001233252
##
## $additional
##  Difference        Mean Alternative    Std. Err
##   37.966261   37.966261    0.000000    6.562402
```

Articles mentioning “election”

```
set.seed(1)
X_binary <- main_data[, c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad", "educ4_h
Y_binary <- main_data$won_election_trump
W_binary <- main_data$election_untrustworthy_flag_test
```

```r
tau.forest_binary <- causal_forest(X_binary, Y_binary, W_binary, num.trees = 4000)
tau.hat_binary <- predict(tau.forest_binary, X_binary, estimate.variance = TRUE)
sigma.hat_binary <- sqrt(tau.hat_binary$variance.estimates)
```

```r
average_treatment_effect(tau.forest_binary, target.sample = "overlap")
```

```
##   estimate    std.err
## 0.03454481 0.03330964
```

```r
main_data_predictions_binary <- main_data %>%
  cbind(tau.hat_binary)
```

```r
average_treatment_effect(tau.forest_binary, subset = main_data$trump_support == 1, target.sample = "ove
```

```
##   estimate    std.err
## 0.06959163 0.06517365
```

```r
average_treatment_effect(tau.forest_binary, subset = main_data$trump_support == 0, target.sample = "ove
```

```
##     estimate      std.err
## -0.003126376  0.010726793
```

```r
test_calibration(tau.forest_binary)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                                Estimate Std. Error t value Pr(>t)
## mean.forest.prediction          0.21874    1.18784  0.1842 0.4270
## differential.forest.prediction -3.29265    1.84044 -1.7891 0.9631
```

Dosage

```r
set.seed(1)
X_dosage <- main_data[, c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad", "educ4_h
Y_dosage <- main_data$won_election_trump
W_dosage <- main_data$election_untrustworthy_n_test
```

```r
tau.forest_dosage <- causal_forest(X_dosage, Y_dosage, W_dosage, num.trees = 4000)
tau.hat_dosage <- predict(tau.forest_dosage, X_dosage, estimate.variance = TRUE)
sigma.hat_dosage <- sqrt(tau.hat_dosage$variance.estimates)
```

```r
average_treatment_effect(tau.forest_dosage, target.sample = "overlap")
```

```
##     estimate      std.err
## 0.0009247837 0.0002402091
```

```r
main_data_predictions_dosage <- main_data %>%
  cbind(tau.hat_dosage)
```

```r
average_treatment_effect(tau.forest_dosage, subset = main_data$trump_support == 1, target.sample = "ove
```

```
##     estimate      std.err
## 0.0009895525 0.0005202569
```

```r
average_treatment_effect(tau.forest_dosage, subset = main_data$trump_support == 0, target.sample = "ove
```

```
##     estimate      std.err
```

```
## 0.0002534368 0.0006564776
```

```
test_calibration(tau.forest_dosage)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                                Estimate Std. Error t value  Pr(>t)
## mean.forest.prediction          1.33128    0.94000  1.4163 0.07848 .
## differential.forest.prediction -0.10308    3.44600 -0.0299 0.51193
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Analysis of other DVS.

```
main_data_2 <- read_csv("data/main_data_2.csv") %>%
  mutate_at(vars(starts_with("election_behaviors")), ~ ifelse(. == "selected", 1, 0)) %>%
  mutate_at(vars(starts_with("election_behaviors")), replace_na, 0) %>%
  mutate(trump_support_pre = trump_support)
```

```
# List of dependent variables
dependent_vars <- c("trump_support_post", "election_behaviors_1", "election_behaviors_2",
                    "election_behaviors_3", "election_behaviors_4", "election_behaviors_5",
                    "election_behaviors_6", "election_behaviors_7", "election_behaviors_8",
                    "election_behaviors_9", "election_behaviors_10", "election_behaviors_11",
                    "election_behaviors_12", "election_behaviors_13", "election_behaviors_14",
                    "election_behaviors_15", "election_behaviors_16", "election_behaviors_17")

X_binary <- main_data_2[, c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad", "educ4
W_binary <- main_data_2$untrustworthy_flag_test

calculate_ATE <- function(dv_name, main_data, X_binary, W_binary) {

  # Extract the dependent variable from the main dataset
  Y_binary <- main_data[[dv_name]]

  # Run the causal forest
  tau.forest_binary <- causal_forest(X_binary, Y_binary, W_binary, num.trees = 4000)

  # Extract the ATE
  ate <- average_treatment_effect(tau.forest_binary, target.sample = "overlap")

  return(ate)
}

# Apply the function to each dependent variable
results <- purrr::map(dependent_vars, calculate_ATE, main_data = main_data_2, X_binary = X_binary, W_bin

results_df <- tibble(
  dependent_variable = dependent_vars,
  ATE = map_dbl(results, "estimate"),
  ATE_se = map_dbl(results, "std.err")
) %>%
  mutate(CI_lower = ATE - 1.96 * ATE_se,
```

```
        CI_upper = ATE + 1.96 * ATE_se) %>%
  mutate(z_value = ATE / ATE_se,
        p_value = 2 * (1 - pnorm(abs(z_value)))) %>%
  mutate_if(is.numeric, round, 3) %>%
  mutate(CI = paste0("[", CI_lower,", ", CI_upper, "]")) %>%
  cbind(tibble(dv = c("Support - Trump", "Attend Rally - Trump", "Attend Rally - Biden", "Attend Rally -
  select(dv, ATE, ATE_se, CI, z_value, p_value) %>%
  filter(!is.na(dv))

results_df %>%
  knitr::kable("latex", booktabs = TRUE,
              col.names = c("DV", "ATE", "se", "CI", "z-value", "p-value"))
```

DV	ATE	se	CI	z-value	p-value
Support - Trump	0.019	0.015	[-0.011, 0.049]	1.240	0.215
Attend Rally - Trump	0.002	0.009	[-0.016, 0.019]	0.208	0.835
Attend Rally - Biden	-0.003	0.007	[-0.017, 0.011]	-0.447	0.655
Attend Rally - Other	0.016	0.010	[-0.003, 0.036]	1.621	0.105
Volunteer - Biden	-0.014	0.013	[-0.039, 0.01]	-1.128	0.259
Volunteer - Trump	-0.004	0.006	[-0.016, 0.008]	-0.626	0.531
Volunteer - Other	-0.007	0.011	[-0.029, 0.015]	-0.643	0.520
Volunteer - Poll Worker	-0.012	0.008	[-0.027, 0.004]	-1.436	0.151
Donation - Trump	-0.001	0.015	[-0.031, 0.029]	-0.088	0.930
Donation - Biden	-0.003	0.023	[-0.047, 0.042]	-0.123	0.902
Donation - Other	0.006	0.024	[-0.041, 0.053]	0.248	0.804
Social Media - Trump	0.028	0.020	[-0.012, 0.068]	1.387	0.165
Social Media - Biden	-0.014	0.026	[-0.065, 0.038]	-0.512	0.608
Social Media - Other	0.047	0.025	[-0.001, 0.095]	1.919	0.055
Yard Sign - Trump	0.010	0.015	[-0.019, 0.039]	0.654	0.513
Yard Sign - Biden	-0.023	0.017	[-0.057, 0.012]	-1.297	0.195
Yard Sign - Other	-0.018	0.017	[-0.052, 0.016]	-1.054	0.292

On belief that Trump *did't* win the election.

```
get_ate_summary <- function(tau.forest) {

  # Extract the average treatment effect and its standard error
  ate_result <- average_treatment_effect(tau.forest, target.sample = "overlap")

  estimate <- ate_result[["estimate"]]
  std.err <- ate_result[["std.err"]]

  # Compute the 95% confidence intervals
  ci_lower <- estimate - 1.96 * std.err
  ci_upper <- estimate + 1.96 * std.err

  # Compute the p-value for the ATE (two-sided test against null hypothesis: ATE = 0)
  p_val <- 2 * (1 - pnorm(abs(estimate / std.err)))

  # Create a tibble to store the results
  results_tibble <- tibble(
    Estimate = as.numeric(estimate),
    Standard_Error = as.numeric(std.err),
```

```r
    CI_Lower = as.numeric(ci_lower),
    CI_Upper = as.numeric(ci_upper),
    p_value = as.numeric(p_val)
  )

  return(results_tibble)
}

set.seed(1)
X_binary <- main_data[, c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad", "educ4_hs
Y_binary <- 1 - main_data$won_election_trump
W_binary <- main_data$untrustworthy_flag_test

# For entire sample
tau.forest_all <- causal_forest(X_binary, Y_binary, W_binary, num.trees = 4000)
result_all <- get_ate_summary(tau.forest_all)

# Subsetting for Trump supporters
trump_supporters <- main_data$trump_support_pre == 1
tau.forest_trump <- causal_forest(X_binary[trump_supporters,], Y_binary[trump_supporters], W_binary[trum
result_trump <- get_ate_summary(tau.forest_trump)

# Subsetting for non-Trump supporters
non_trump_supporters <- main_data$trump_support_pre == 0
tau.forest_non_trump <- causal_forest(X_binary[non_trump_supporters,], Y_binary[non_trump_supporters], W
result_non_trump <- get_ate_summary(tau.forest_non_trump)

# Combining results
results <- bind_rows(All_Sample = result_all, Trump_Supporters = result_trump, Non_Trump_Supporters = re

results %>%
  kableExtra::kable()
```

Group	Estimate	Standard_Error	CI_Lower	CI_Upper	p_value
All_Sample	-0.0416443	0.0195604	-0.0799826	-0.0033060	0.0332528
Trump_Supporters	-0.1145491	0.0557412	-0.2238019	-0.0052964	0.0398773
Non_Trump_Supporters	-0.0027096	0.0088185	-0.0199939	0.0145746	0.7586401

Propsensity score matching

```r
library(MatchIt)

# Defining the formula for the treatment model
formula <- W_binary ~ untrustworthy_flag_pre + trump_support_pre + educ4_college_grad + educ4_hs_or_less

# Computing propensity scores using logistic regression
m.out <- matchit(formula, data = main_data, method = "nearest")

# Matching data based on the propensity scores
matched_data <- match.data(m.out)

# Assessing balance for matched data
# love.plot(m.out)

# Fit a linear model
```

```r
fit <- lm(won_election_trump ~ untrustworthy_flag_test + untrustworthy_flag_pre + trump_support_pre + ed

# Get the coefficient for the treatment variable (W_binary)
att <- marginaleffects::avg_comparisons(fit,
                                variables = "untrustworthy_flag_test",
                                vcov = ~subclass,
                                newdata = subset(matched_data, untrustworthy_flag_test == 1),
                                wts = "weights")

att %>%
  kableExtra::kable()
```

term	contrast	estimate	std.error	statistic	p.value	conf.low	conf.high
untrustworthy_flag_test	1 - 0	0.066495	0.0216832	3.066657	0.0021647	0.0239967	0.1089933

```r
# Defining the formula for the treatment model
formula <- W_binary ~ untrustworthy_flag_pre + trump_support_pre + educ4_college_grad + educ4_hs_or_les

# Computing propensity scores using logistic regression
m.out <- matchit(formula, data = main_data, method = "nearest")

# Matching data based on the propensity scores
matched_data <- match.data(m.out)

# Assessing balance for matched data
# love.plot(m.out)

# Fit a linear model
fit <- lm(won_election_trump ~ untrustworthy_n_test + untrustworthy_flag_pre + trump_support_pre + educ4

# Get the coefficient for the treatment variable (W_binary)
att <- marginaleffects::avg_comparisons(fit,
                                variables = "untrustworthy_n_test",
                                vcov = ~subclass,
                                newdata = subset(matched_data, untrustworthy_n_test == 1),
                                wts = "weights")

att %>%
  kableExtra::kable()
```

term	contrast	estimate	std.error	statistic	p.value	conf.low	conf.high
untrustworthy_n_test	+1	0.0003696	0.0000745	4.963495	0.0000007	0.0002237	0.0005156

Triple Machine Learning

```r
# library(causalHAL)
#
# set.seed(1)
# X_binary <- main_data[, c("untrustworthy_flag_pre", "trump_support_pre", "educ4_college_grad", "educ4
# Y_binary <- main_data$won_election_trump
# W_binary <- main_data$untrustworthy_flag_test
#
# forest_w <- regression_forest(X_binary, W_binary, tune.parameters = "all")
# w_hat <- predict(forest_w)$predictions
#
# # Estimate the treatment-marginalized outcome regression using causal_forest
```

```r
# tau.forest_outcome <- causal_forest(X_binary, Y_binary, W_binary, num.trees = 4000)
#
# # Extract treatment-marginalized outcome regression estimates
# m_hat <- predict(tau.forest_outcome, X_binary, estimate.variance = TRUE)$predictions
#
# # Adapt this line to your dataset structure
# ADMLE_fit <- fit_cate_hal_partially_linear(main_data$W_binary, main_data$A_binary, main_data$Y_binary
#                                            m.hat = m_hat,
#                                            pi.hat = w_hat,
#                                            smoothness_orders_cate = 1, num_knots_cate = c(50), max_deg
#
# # Provides estimates and CI for ATE
# inference_ate(ADMLE_fit)
#
# # If you have HAL9001 package, you can also use glmnet implementation with hal9001-basis design matri
# basis_list <- hal9001::enumerate_basis(main_data$W_binary, smoothness_orders = 1, num_knots = 50, max
# tau_basis <- hal9001::make_design_matrix(main_data$W_binary, basis_list)
#
# # Adapt this line to your dataset structure
# ADMLE_fit <- fit_cate_lasso_partially_linear(tau_basis, main_data$A_binary, main_data$Y_binary,
#                                              m.hat = m.hat,
#                                              pi.hat = pi.hat, standardize = FALSE)
#
# # Provides estimates and CI for ATE
# inference_ate(ADMLE_fit)
```

References