Airbnb: Causal Inference

Inferring Causal Effects of Marketing Ads on User Bookings

Diana Liang, Andrew Pagtakhan

12/4/2020

Contents


```
library(bartCause)
library(knitr)
library(tidyverse)
library(here)
library(gridExtra)
library(rpart)
library(ranger)
library(MatchIt) # remove?
library(arm)
library(gbm)
library(correlationfunnel)
library(conflicted)
library(table1)

conflict_prefer("filter", "dplyr")
conflict_prefer("select", "dplyr")

# set seed
seed <- 14
set.seed(seed)
```

```r
# increase memory limit for matching functions
# make sure this limit is compatible with the system used to run this code
memory.limit(3.2e10)
```

```
## [1] 3.2e+10
```

```r
knitr::opts_chunk$set(echo = TRUE,
                      tidy.opts=list(width.cutoff=60),
                      tidy=TRUE,
                      warning = FALSE,
                      message = FALSE,
                      cache.lazy = FALSE)
```

Load data

```r
# read in data
dat <- read_csv(here::here("data", "bookings_sample.csv"))

# convert categorical variables
dat <- dat %>% mutate_at(vars(c("gender", "browser_group")),
    as.factor)

# Airbnb color palette (for plotting) Source:
# https://usbrandcolors.com/airbnb-colors/
col.pal <- c("#FC642D", "#00A699", "#FF5A5F", "#767676", "#484848")
treat.labs <- c("Indirect Ad", "Direct Ad")
treat.name <- "Treatment"
```

1 Exploratory Data Analysis and Overlap

There is sufficient overlap for the covariates between those who received direct marketing ads vs. indirect ads.

See Tableau workbook for interactive exploration: https://public.tableau.com/profile/andrew.pagtakhan#!/vizhome/NYU_Causal_Airbnb/Dashboard?publish=yes

Note: Underlying data used for Tableau workbook is `bookings_sample.csv`

2 Modeling

2.1 Logistic Regression

```r
# fit model
fit.lr <- glm(treat ~ is_eng + is_mobile + age + gender + browser_group,
    family = binomial, data = dat)

# extract propensity scores
dat$pscores.lr <- predict(fit.lr, type = "response")

## one-to-one nearest neighbor w/ replacement
matches.lr <- matching(z = dat$treat, score = dat$pscores.lr,
    replace = T)
weight.lr <- ifelse(dat$treat == 0, matches.lr$cnts, 1)

## IPTW
```

```r
dat$IPTW.lr <- dat %>% transmute(IPTW = if_else(treat == 1, 1,
    pscores.lr/(1 - pscores.lr))) %>% unlist() %>% unname()
```

2.2 Probit Regression

```r
# fit model
fit.prob <- glm(treat ~ is_eng + is_mobile + age + gender + browser_group,
    family = binomial(link = "probit"), data = dat)

# extract propensity scores
dat$pscores.prob <- predict(fit.prob, type = "response")

## one-to-one nearest neighbor w/ replacement
matches.prob <- matching(z = dat$treat, score = dat$pscores.prob,
    replace = T)
weight.prob <- ifelse(dat$treat == 0, matches.prob$cnts, 1)

## IPTW
dat$IPTW.prob <- dat %>% transmute(IPTW = ifelse(treat == 1,
    1, pscores.prob/(1 - pscores.prob))) %>% unlist() %>% unname()
```

2.3 CART

```r
# fit model
fit.cart <- rpart(treat ~ is_eng + is_mobile + age + gender +
    browser_group, data = dat, method = "class")  # fit a CART model

# extract p scores take probability of assigned=1 as
# propensity scores
dat$pscores.cart <- predict(fit.cart, type = "prob")[, 2]

## one-to-one nearest neighbor w/ replacement
matches.cart <- matching(z = dat$treat, score = dat$pscores.cart,
    replace = T)
weight.cart <- ifelse(dat$treat == 0, matches.cart$cnts, 1)

## IPTW
dat$IPTW.cart <- dat %>% transmute(IPTW = ifelse(treat == 1,
    1, pscores.cart/(1 - pscores.cart))) %>% unlist() %>% unname()
```

2.4 Random Forest

```r
fit.rf <- ranger(treat ~ is_eng + is_mobile + age + gender +
    browser_group, data = dat)

dat$pscores.rf <- predict(fit.rf, data = dat)$predictions

## one-to-one nearest neighbor w/ replacement
matches.rf <- matching(z = dat$treat, score = dat$pscores.rf,
    replace = T)
weight.rf <- ifelse(dat$treat == 0, matches.rf$cnts, 1)
```

```
## IPTW
dat$IPTW.rf <- dat %>% transmute(IPTW = ifelse(treat == 1, 1,
    pscores.rf/(1 - pscores.rf))) %>% unlist() %>% unname()
```

2.5 GBM

```
fit.gbm <- gbm(treat ~ is_eng + is_mobile + age + gender + browser_group,
    data = dat, distribution = "bernoulli")

dat$pscores.gbm <- predict(fit.gbm, type = "response")

## one-to-one nearest neighbor w/ replacement
matches.gbm <- matching(z = dat$treat, score = dat$pscores.gbm,
    replace = T)
weight.gbm <- ifelse(dat$treat == 0, matches.gbm$cnts, 1)

## IPTW
dat$IPTW.gbm <- dat %>% transmute(IPTW = ifelse(treat == 1, 1,
    pscores.gbm/(1 - pscores.gbm))) %>% unlist() %>% unname()
```

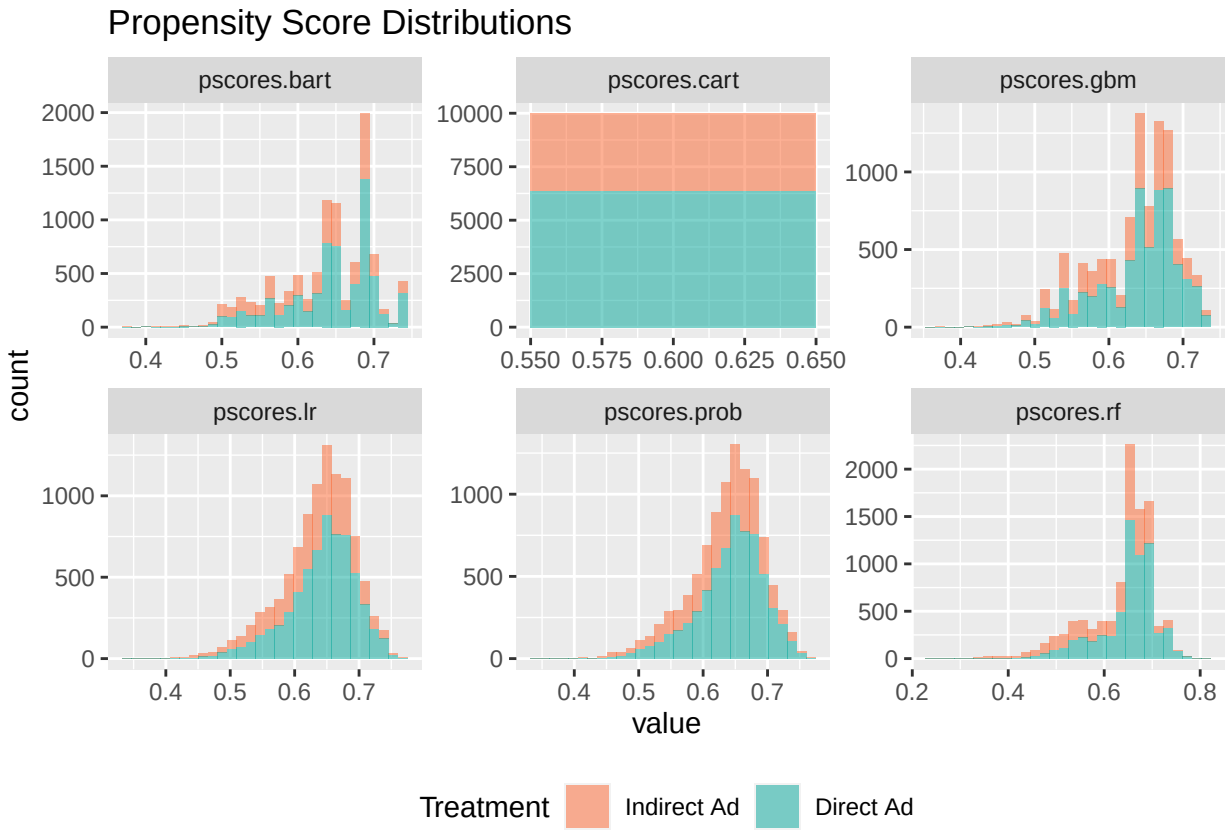
2.6 BART

```
# try bartCause

# get covariates for model
xtrain_bart <- as.data.frame(dat %>% select(age, gender, is_mobile,
    browser_group, is_eng))
# fit model
bart <- bartc(response = dat$is_booked, treatment = dat$treat,
    confounders = xtrain_bart, keepTrees = FALSE, method.rsp = "bart",
    method.trt = "bart", estimand = "att", verbose = FALSE)

# get propensity scores
dat$pscores.bart <- bart$p.score

# get propensity scores
dat$pscores.bart <- bart$p.score

## one-to-one nearest neighbor w/ replacement
matches.bart <- matching(z = dat$treat, score = dat$pscores.bart,
    replace = T)
weight.bart <- ifelse(dat$treat == 0, matches.bart$cnts, 1)

## IPTW
dat$IPTW.bart <- dat %>% transmute(IPTW = ifelse(treat == 1,
    1, pscores.bart/(1 - pscores.bart))) %>% unlist() %>% unname()
```

3 Model Summaries

3.1 Propensity Scores

```
# plot propensity scores
dat %>% pivot_longer(cols = contains("pscores")) %>% ggplot(aes(x = value,
    fill = as.factor(treat))) + geom_histogram(alpha = 0.5) +
    facet_wrap(facets = vars(name), scales = "free") + labs(title = "Propensity Score Distributions") +
    theme(legend.position = "bottom") + scale_fill_manual(name = treat.name,
    labels = treat.labs, values = col.pal)
```



3.2 Balance

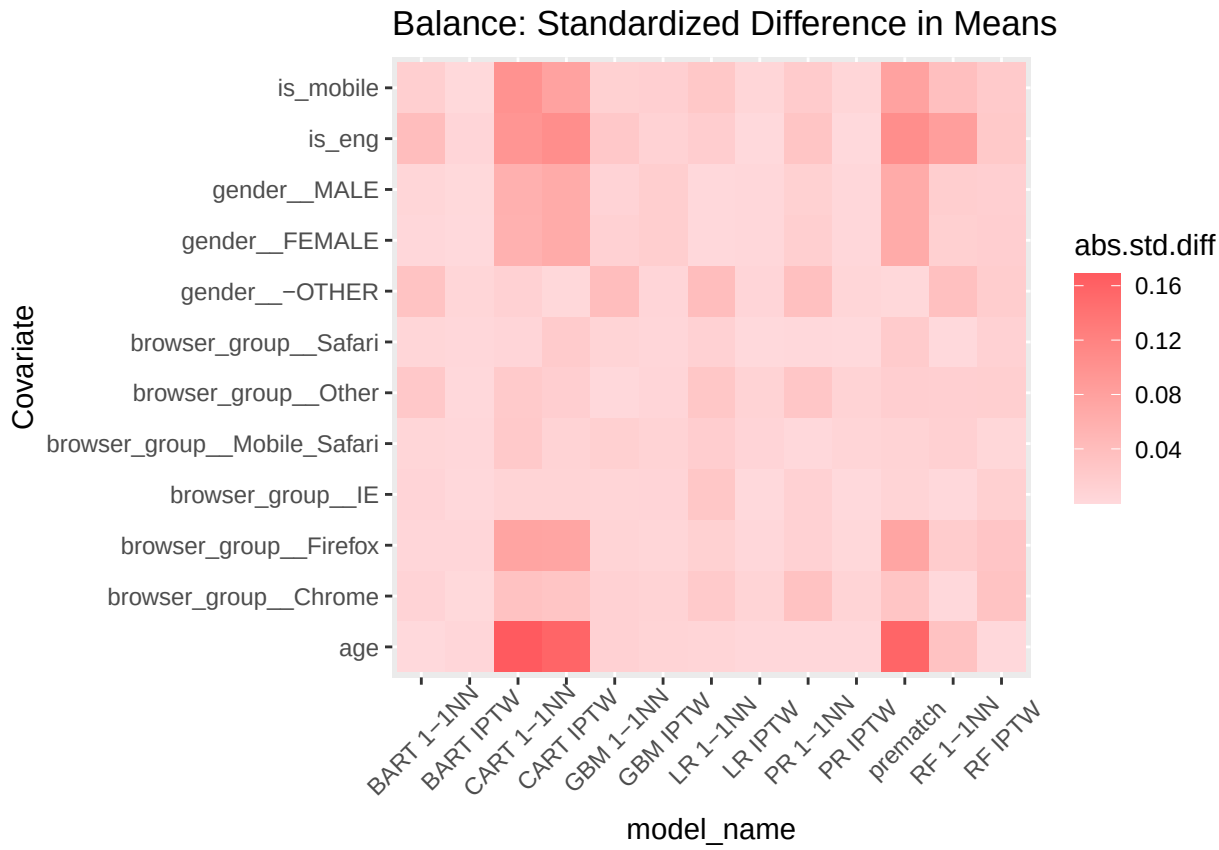
```
# recode categorical
cat <- dat %>% select(gender,browser_group) %>% binarize()
dat_bal <- bind_cols(dat,cat)

# get covariates and convert to matrices for balance function
bal_mat <- as.matrix(dat_bal %>% select(c(contains(c("browser", "gender")),
                                          -c("gender", "browser_group","first_browser"),
                                          c("age", "is_mobile", "is_eng"))))
bal_treat <- as.matrix(dat_bal$treat)

# matched weights
match_counts <- list(matches.lr$cnts, dat$IPTW.lr,
                     matches.prob$cnts, dat$IPTW.prob,
```

```
                    matches.cart$cnts, dat$IPTW.cart,
                    matches.rf$cnts, dat$IPTW.rf,
                    matches.gbm$cnts, dat$IPTW.gbm,
                    matches.bart$cnts, dat$IPTW.bart)

# convert to matrices
match_mat <- lapply(match_counts, as.matrix)

# compute balance statistics for each model
bal_list <- lapply(match_mat,
                   function(x) {
                     balance(rawdata = bal_mat,
                             treat = bal_treat,
                             matched = x,
                             estimand = "ATT")$diff.means.matched
                   })
```

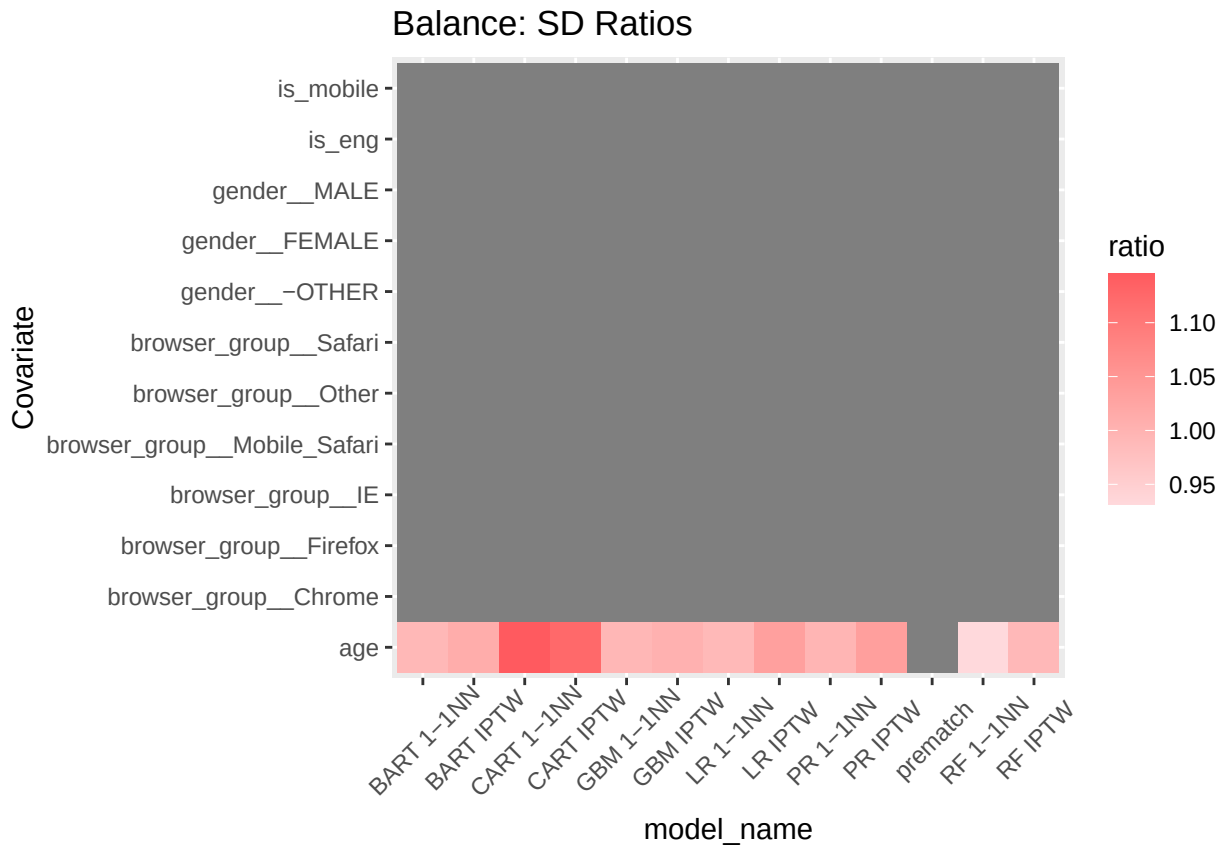
```
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
## Balance diagnostics assume that the estimand is the ATT
```

```
# model names
model_list <- c('LR 1-1NN','LR IPTW',
                'PR 1-1NN','PR IPTW',
                'CART 1-1NN','CART IPTW',
                'RF 1-1NN','RF IPTW',
                'GBM 1-1NN','GBM IPTW',
                'BART 1-1NN','BART IPTW')

# add model name column
bal_dfs <- list()
for (i in 1:length(bal_list)) {
    bal_df <- as.data.frame(bal_list[[i]]) %>%
      rownames_to_column(var = "Covariate")
    bal_df$model_name <- model_list[[i]]
    bal_dfs[[i]] <- bal_df
}

# add pre-matched balance metrics
bal_df_pre <- as.data.frame(balance(rawdata = bal_mat,
                                    treat = bal_treat,
                                    # placeholder - not used
                                    matched = match_counts[[1]],
                                    estimand = "ATT")$diff.means.raw) %>%
              rownames_to_column(var = "Covariate")
```

```
## Balance diagnostics assume that the estimand is the ATT
# add model name
bal_df_pre$model_name <- "prematch"


# combine balance metrics
bal_df <- rbind(bal_df_pre, bind_rows(bal_dfs))

# plot
p.md <- ggplot(bal_df, aes(x = model_name, y = Covariate)) +
        geom_tile(aes(fill = abs.std.diff)) +
        labs(title = "Balance: Standardized Difference in Means") +
        scale_fill_gradient(low = "#ffd9dc", high = "#FF5A5F") +
        theme(axis.text.x=element_text(angle=45,vjust=.6))

p.sd <- ggplot(bal_df, aes(x = model_name, y = Covariate)) +
        geom_tile(aes(fill = ratio)) +
        labs(title = "Balance: SD Ratios") +
        scale_fill_gradient(low = "#ffd9dc", high = "#FF5A5F") +
        theme(axis.text.x=element_text(angle=45,vjust=.6))
p.md
```




```
p.sd
```




```
#grid.arrange(p.md, p.sd, nrow = 2)
```

```
# best model based on difference in means
kable(bal_df %>% group_by(model_name) %>% summarise(mean.bal = mean(abs.std.diff)) %>%
    arrange(mean.bal), caption = "Absolute Standardized Difference in Means")
```

Table 1: Absolute Standardized Difference in Means

model_name	mean.bal
BART IPTW	0.0024449
PR IPTW	0.0034908
LR IPTW	0.0038071
GBM IPTW	0.0089942
BART 1-1NN	0.0123544
GBM 1-1NN	0.0129337
RF IPTW	0.0166580
PR 1-1NN	0.0167739
LR 1-1NN	0.0171875
RF 1-1NN	0.0223270
prematch	0.0525743
CART IPTW	0.0525743
CART 1-1NN	0.0548722

```
# best model on sd ratios
kable(bal_df %>% filter(Covariate == "age") %>% group_by(model_name) %>%
```

```
    summarise(sd.ratio = abs(mean(ratio) - 1)) %>% arrange(sd.ratio),
    caption = "SD Ratios (Absolute values, centered on 1)")
```

Table 2: SD Ratios (Absolute values, centered on 1)

model_name	sd.ratio
PR 1-1NN	0.0024309
GBM IPTW	0.0045362
GBM 1-1NN	0.0054187
BART 1-1NN	0.0076500
RF IPTW	0.0089785
LR 1-1NN	0.0096122
BART IPTW	0.0116685
LR IPTW	0.0347982
PR IPTW	0.0354177
RF 1-1NN	0.0679157
CART IPTW	0.1242490
CART 1-1NN	0.1451647
prematch	NA

4 Estimate Causal Effects

4.1 Treatment Effect for Best Model

Based on the balance metrics, the Probit Regression using IPTW Matching was the best model.

```
# logistic regression to get final treatment effect
model_fin <- glm(is_booked ~ is_eng + is_mobile + age + gender +
    browser_group + treat, family = quasibinomial(link = "logit"),
    data = dat, weights = dat$IPTW.bart)

# odds
odds <- exp(model_fin$coefficients["treat"][[1]])
# probability
paste0("Treatment Effect: ", prob_treat <- odds/(1 + odds))
```

```
## [1] "Treatment Effect: 0.543579006049429"
```

4.2 Treatment effect for all models

```
# treatment effect for all models
effects_list <- lapply(match_counts, function(x) {
    glm(is_booked ~ is_eng + is_mobile + age + gender + browser_group +
        treat, family = quasibinomial(link = "logit"), data = dat,
        weights = x)
})

# add model name column
eff_dfs <- list()
for (i in 1:length(effects_list)) {
    eff_df <- as.data.frame(effects_list[[i]]$coefficients["treat"])
    eff_df$model_name <- model_list[[i]]
```

```
    eff_dfs[[i]] <- eff_df
}

# final treatment effects for each model
eff_fin <- bind_rows(eff_dfs)
colnames(eff_fin) <- c("treat_eff", "model_name")
rownames(eff_fin) <- NULL
kable(eff_fin, caption = "Treatment Effects: All Models")
```

Table 3: Treatment Effects: All Models

treat_eff	model_name
0.2076342	LR 1-1NN
0.1814675	LR IPTW
0.1476444	PR 1-1NN
0.1813737	PR IPTW
0.2001964	CART 1-1NN
0.1823416	CART IPTW
0.1641730	RF 1-1NN
0.1655509	RF IPTW
0.1767498	GBM 1-1NN
0.1754111	GBM IPTW
0.1466017	BART 1-1NN
0.1747594	BART IPTW