

# Fitting the outcome model

Lucy D'Agostino McGowan

Wake Forest University

2020-07-29 (updated: 2020-07-28)

# Outcome Model

```
library(broom)
```

```
lm(outcome ~ exposure, data = df, weights = wts) %>%  
  tidy()
```

- ✅ This will get us the point estimate
- ❌ This will get NOT us the correct confidence intervals
- 📦 {rsample}

# 1 Create a function to run your analysis once on a sample of your data

```
fit_ipw <- function(split, ...) {  
  .df <- analysis(split)  
  
  # fit propensity score model  
  propensity_model <- glm(  
    exposure ~ confounder_1 + confounder_2 + ...  
    family = binomial(),  
    data = .df  
  )  
  
  # calculate inverse probability weights  
  .df <- propensity_model %>%  
    augment(type.predict = "response", data = .df) %>%  
    mutate(wts = 1 / ifelse(exposure == 0, 1 - .fitted, .fitted))  
  
  # fit correctly bootstrapped ipw model  
  lm(outcome ~ exposure, data = .df, weights = wts) %>%  
    tidy()  
}
```

# 1 Create a function to run your analysis once on a sample of your data

```
fit_ipw <- function(split, ...) {  
  .df <- analysis(split)  
  
  # fit propensity score model  
  propensity_model <- glm(  
    exposure ~ confounder_1 + confounder_2 + ...  
    family = binomial(),  
    data = .df  
  )  
  
  # calculate inverse probability weights  
  .df <- propensity_model %>%  
    augment(type.predict = "response", data = .df) %>%  
    mutate(wts = 1 / ifelse(exposure == 0, 1 - .fitted, .fitted))  
  
  # fit correctly bootstrapped ipw model  
  lm(outcome ~ exposure, data = .df, weights = wts) %>%  
    tidy()  
}
```

# 1 Create a function to run your analysis once on a sample of your data

```
fit_ipw <- function(split, ...) {  
  .df <- analysis(split)  
  
  # fit propensity score model  
  propensity_model <- glm(  
    exposure ~ confounder_1 + confounder_2 + ...  
    family = binomial(),  
    data = .df  
  )  
  
  # calculate inverse probability weights  
  .df <- propensity_model %>%  
    augment(type.predict = "response", data = .df) %>%  
    mutate(wts = 1 / ifelse(exposure == 0, 1 - .fitted, .fitted))  
  
  # fit correctly bootstrapped ipw model  
  lm(outcome ~ exposure, data = .df, weights = wts) %>%  
    tidy()  
}
```

# 1 Create a function to run your analysis once on a sample of your data

```
fit_ipw <- function(split, ...) {  
  .df <- analysis(split)  
  
  # fit propensity score model  
  propensity_model <- glm(  
    exposure ~ confounder_1 + confounder_2 + ...  
    family = binomial(),  
    data = .df  
  )  
  
  # calculate inverse probability weights  
  .df <- propensity_model %>%  
    augment(type.predict = "response", data = .df) %>%  
    mutate(wts = 1 / ifelse(exposure == 0, 1 - .fitted, .fitted))  
  
  # fit correctly bootstrapped ipw model  
  lm(outcome ~ exposure, data = .df, weights = wts) %>%  
    tidy()  
}
```

# 1 Create a function to run your analysis once on a sample of your data

```
fit_ipw <- function(split, ...) {  
  .df <- analysis(split)  
  
  # fit propensity score model  
  propensity_model <- glm(  
    exposure ~ confounder_1 + confounder_2 + ...  
    family = binomial(),  
    data = .df  
  )  
  
  # calculate inverse probability weights  
  .df <- propensity_model %>%  
    augment(type.predict = "response", data = .df) %>%  
    mutate(wts = 1 / ifelse(exposure == 0, 1 - .fitted, .fitted))  
  
  # fit correctly bootstrapped ipw model  
  lm(outcome ~ exposure, data = .df, weights = wts) %>%  
    tidy()  
}
```

## 2 Use {rsample} to bootstrap our causal effect

```
library(rsample)

# fit ipw model to bootstrapped samples
ipw_results <- bootstraps(df, 1000, apparent = TRUE) %>%
  mutate(results = map(splits, fit_ipw))
```



## 2 Use {rsample} to bootstrap our causal effect

```
library(rsample)

# fit ipw model to bootstrapped samples
ipw_results <- bootstraps(df, 1000, apparent = TRUE) %>%
  mutate(results = map(splits, fit_ipw))
```

## 2 Use {rsample} to bootstrap our causal effect

```
library(rsample)

# fit ipw model to bootstrapped samples
ipw_results <- bootstraps(df, 1000, apparent = TRUE) %>%
  mutate(results = map(splits, fit_ipw))
```

### 3 Pull out the causal effect

```
# get t-statistic-based CIs  
boot_estimate <- int_t(ipw_results, results) %>%  
  filter(term == "exposure")
```

## Your Turn

- 1 Create a function called `ipw_fit` that fits the propensity score model and the weighted outcome model for the effect between `qsmk` and `wt82_71`**
- 2 Using the `bootstraps()` and `int_t()` functions to estimate the final effect.**

07:00