Propensity scores for continuous exposures

Malcolm Barrett

RStudio, PBC

2022-07-20 (updated: 2022-10-16)

The story so far

Propensity score weighting

1. **Fit a propensity model predicting exposure** x, x + z **where z is all covariates**
2. **Calculate weights**
3. **Fit an outcome model estimating the effect of** x **on** y **weighted by the propensity score**

Continous exposures

1. **Use a model like** lm(x ~ z) **for the propensity score model**
2. **Scale weights to probability-like scale using** dnorm(true_value, fitted_value, estimated_sd)
3. **Apply the weights to the outcome model as normal!**

Alternative: quantile binning

1. **Bin the continuous exposure into quantiles and use categorical regression like a multinomial model to calculate probabilities.**
2. **Calculate the weights where the propensity score is the probability you fall into the quantile you**

1. Fit a model for exposure ~ confounders

```
model <- lm(
  exposure ~ confounder_1 + confounder_2,
  data = df
)
```

2. Calculate the weights with dnorm()

```
model %>%
  augment(data = df) %>%
  mutate(denominator = dnorm(
    exposure,
    mean = .fitted,
    sd = mean(.sigma, na.rm = TRUE)
  ))
```

Does change in smoking intensity (smkintensity82_71) affect weight gain among lighter smokers?

```
nhefs_light_smokers <- nhefs_complete %>%
  filter(smokeintensity <= 25)
```

1. Fit a model for exposure ~ confounders

```
nhefs_denominator_model <- lm(
  smkintensity82_71 ~ sex + race + age + I(age^2) +
    education + smokeintensity + I(smokeintensity^2) +
    smokeyrs + I(smokeyrs^2) + exercise + active +
    wt71 + I(wt71^2),
  data = nhefs_light_smokers
)
```

2. Calculate the weights with dnorm()

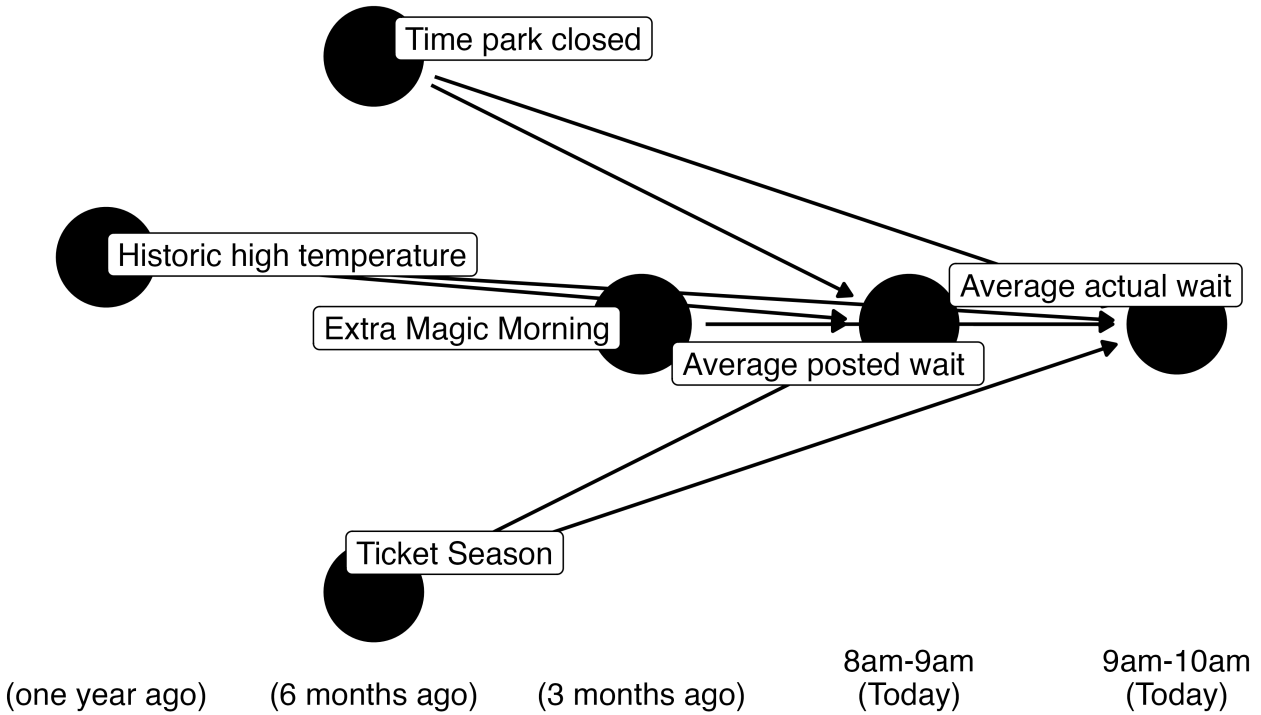
```
nhefs_denominators <- nhefs_denominator_model %>%
  augment(data = nhefs_light_smokers) %>%
  mutate(denominator = dnorm(
    smkintensity82_71,
    .fitted,
    mean(.sigma, na.rm = TRUE)
  )) %>%
  select(id, denominator)
```

2. Calculate the weights with dnorm()

```
nhefs_denominators
```

```
## # A tibble: 1,162 × 2
##       id denominator
##    <int>       <dbl>
##  1     2      0.0265
##  2     3      0.0275
##  3     4      0.0314
##  4     5      0.0371
##  5     6      0.0262
##  6     7      0.0364
##  7     8      0.0381
##  8     9      0.0386
##  9    10      0.0129
## 10    13      0.0386
## # … with 1,152 more rows
```

Do posted wait times at 8 am affect actual wait times at 9 am?



Your Turn 1

Fit a model using lm() **with** avg_spostmin **as the outcome and the confounders identified in the DAG.**

Use augment() **to add model predictions to the data frame**

In dnorm()**, use** .fitted **as the mean and the mean of** .sigma **as the SD to calculate the propensity score for the denominator.**

05:00

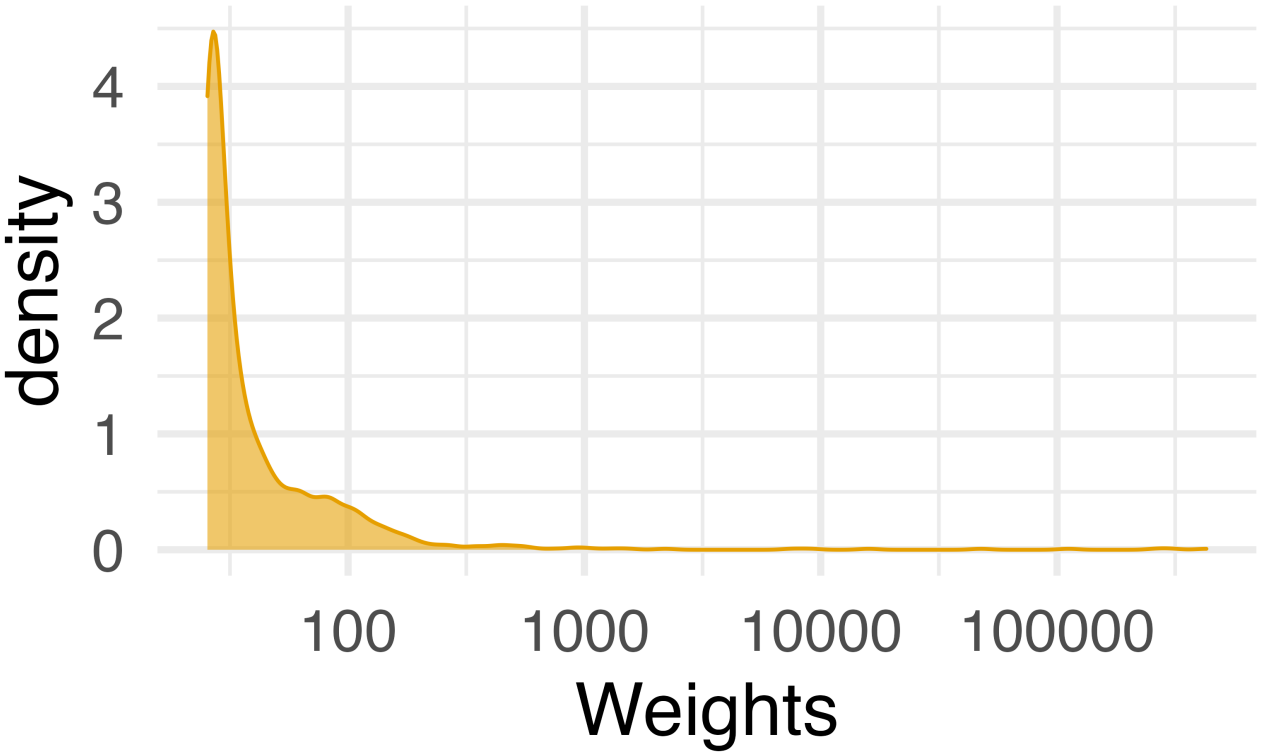
Your Turn 1

```
denominator_model <- lm(
  avg_spostmin ~
    close + extra_magic_morning +
    weather_wdwhigh + wdw_ticket_season,
  data = wait_times
)
```

Your Turn 1

```
denominators <- denominator_model %>%
  augment(data = wait_times) %>%
  mutate(
    denominator = dnorm(
      avg_spostmin, .fitted, mean(.sigma, na.rm = TRUE)
    )
  ) %>%
  select(date, denominator)
```

Stabilizing extreme weights



Stabilizing extreme weights

1. **Fit an intercept-only model (e.g. `lm(x ~ 1)`)**
2. **Calculate weights from this model**
3. **Divide these weights by the propensity score weights**

Fit an intercept-only model

```
nhefs_numerator_model <- lm(
  smkintensity82_71 ~ 1,
  data = nhefs_light_smokers
)
```

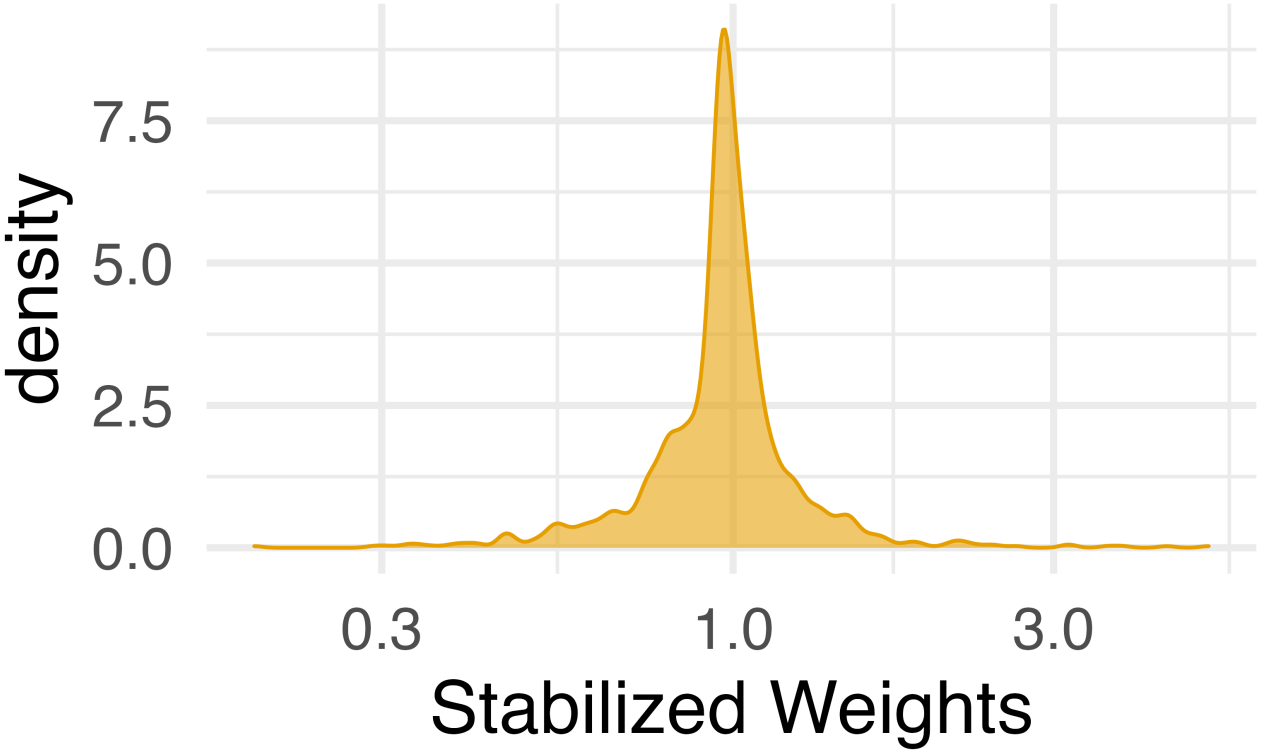
Calculate weights from this model

```
nhefs_numerators <- nhefs_numerator_model %>%
  augment(data = nhefs_light_smokers) %>%
  mutate(numerator = dnorm(
    smkintensity82_71,
    mean = .fitted,
    sd = mean(.sigma, na.rm = TRUE))
  ) %>%
  select(id, numerator)
```

Divide these weights by the propensity score weights

```
nhefs_light_smokers <- nhefs_light_smokers %>%
  left_join(nhefs_numerators, by = "id") %>%
  left_join(nhefs_denominators, by = "id") %>%
  mutate(swts = numerator / denominator)
```

Stabilizing extreme weights



Your Turn 2

Fit an intercept-only model of posted weight times to use as the numerator model

Calculate the numerator weights using dnorm() **as above.**

Finally, calculate the stabilized weights, swts**, using the** numerator **and** denominator **weights**

05:00

Your Turn 2

```
numerator_model <- lm(
  avg_spostmin ~ 1,
  data = wait_times
)
```

Your Turn 2

```
numerators <- numerator_model %>%
  augment(data = wait_times) %>%
  mutate(
    numerator = dnorm(
      avg_spostmin, .fitted, mean(.sigma, na.rm = TRUE)
    )
  ) %>%
  select(date, numerator)

wait_times_wts <- wait_times %>%
  left_join(numerators, by = "date") %>%
  left_join(denominators, by = "date") %>%
  mutate(swts = numerator / denominator)
```

Fitting the outcome model

1. **Use the stabilized weights in the outcome model. Nothing new here!**

```r
lm(
  wt82_71 ~ smkintensity82_71,
  weights = swts,
  data = nhefs_light_smokers
) %>%
  tidy() %>%
  filter(term == "smkintensity82_71") %>%
  mutate(estimate = estimate * -10)
```

```
## # A tibble: 1 × 5
##   term              estimate std.error statistic
##   <chr>                <dbl>     <dbl>     <dbl>
## 1 smkintensity82_71    0.960    0.0210     -4.58
##      p.value
##        <dbl>
## 1 0.00000519
```

Your Turn 3

Estimate the relationship between posted wait times and actual wait times using the stabilized weights we just created.

03:00

Your Turn 3

```
lm(
  avg_sactmin ~ avg_spostmin,
  weights = swts,
  data = wait_times_wts
) %>%
  tidy() %>%
  filter(term == "avg_spostmin") %>%
  mutate(estimate = estimate * 10)
```

```
## # A tibble: 1 × 5
##   term         estimate std.error statistic p.value
##   <chr>           <dbl>     <dbl>     <dbl>   <dbl>
## 1 avg_spostmin    -2.63    0.0807     -3.26 0.00162
```