

Homework 1

Submission 2, Spring 2026

Srijon Sarkar

```
options(repos = c(CRAN = "https://cloud.r-project.org"))

if (!require("pacman")) install.packages("pacman")
```

Loading required package: pacman

```
pacman::p_load(tidyverse, ggplot2, dplyr, lubridate, stringr, readxl, data.table, gdata, scales)
source("../functions-1.R")
```

```
data.2014 <- read.csv('~/data/output/data-2014.csv')
data.2015 <- read.csv('~/data/output/data-2015.csv')
data.2016 <- read.csv('~/data/output/data-2016.csv')
data.2017 <- read.csv('~/data/output/data-2017.csv')
data.2018 <- read.csv('~/data/output/data-2018.csv')
data.2019 <- read.csv('~/data/output/data-2019.csv')
```

```
data.full <- rbind(data.2014, data.2015, data.2016, data.2017, data.2018, data.2019)

glimpse(data.full)
```

Rows: 449,046

Columns: 58

```
$ contractid <chr> "H0022", "H0022", "H0022", "H0022", "H0022", "H002~  
$ planid <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 4, ~  
$ fips <int> 39023, 39035, 39051, 39055, 39057, 39085, 39093, 3~  
$ year <int> 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 20~  
$ n_nonmiss <int> 6, 8, 7, 0, 6, 8, 8, 7, 8, 6, 4, 7, 12, 12, 12, 12~  
$ avg_enrollment <dbl> 72.00000, 330.12500, 24.57143, NA, 120.83333, 41.5~
```

```

$ sd_enrollment <dbl> 7.2111026, 10.6158574, 1.8126539, NA, 15.7659972, ~
$ min_enrollment <int> 60, 309, 23, NA, 90, 39, 29, 299, 24, 431, 11, 71, ~
$ max_enrollment <int> 81, 344, 28, NA, 135, 44, 43, 440, 26, 713, 12, 85~
$ first_enrollment <int> 60, 309, 24, NA, 90, 41, 29, 299, 24, 431, 11, 71, ~
$ last_enrollment <int> 81, 344, 24, NA, 135, 44, 43, 440, 24, 713, 12, 82~
$ state <chr> "OH", "OH", "OH", "OH", "OH", "OH", "OH", "OH", "O~
$ county <chr> "Clark", "Cuyahoga", "Fulton", "Geauga", "Greene", ~
$ org_type <chr> "Demo", "Demo", "Demo", "Demo", "Demo", "Demo", "D~
$ plan_type <chr> "Medicare-Medicaid Plan HMO/HMOPOS", "Medicare-~
Med~

$ partd <chr> "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", ~
$.snp <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No", "N~
$ eghp <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No", "N~
$ org_name <chr> "BUCKEYE COMMUNITY HEALTH PLAN, INC.", "BUCKEYE CO~
$ org_marketing_name <chr> "Buckeye Health Plan - MyCare Ohio", "Buckeye Heal~
$ plan_name <chr> "Buckeye Community Health Plan - MyCareOhio (Medic~
$ parent_org <chr> "Centene Corporation", "Centene Corporation", "Cen~
$ contract_date <chr> "05/01/2014 0:00:00", "05/01/2014 0:00:00", "05/01~
$ state_long <chr> "ohio", "ohio", "ohio", "ohio", "ohio", "ohio", "o~
$ county_long <chr> "Clark", "Cuyahoga", "Fulton", "Geauga", "Greene", ~
$ n_elig <int> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12~
$ n_enrol <int> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12~
$ avg_eligibles <dbl> 28454.167, 239039.917, 7940.667, 17845.250, 27621.~
$ sd_eligibles <dbl> 189.188138, 1269.732574, 90.744730, 190.812676, 24~
$ min_eligibles <int> 28186, 237409, 7818, 17551, 27276, 46128, 56725, 7~
$ max_eligibles <int> 28802, 241155, 8081, 18136, 28054, 47183, 58358, 7~
$ first_eligibles <int> 28186, 237565, 7818, 17551, 27276, 46128, 56725, 7~
$ last_eligibles <int> 28802, 241155, 8081, 18136, 28054, 47183, 58358, 7~
$ avg_enrolled <dbl> 13405.2500, 92824.3333, 2445.5833, 6246.0000, 1050~
$ sd_enrolled <dbl> 140.73064, 2245.45967, 56.73136, 187.22908, 138.86~
$ min_enrolled <int> 13200, 88467, 2340, 5894, 10298, 17006, 17843, 284~
$ max_enrolled <int> 13585, 94709, 2508, 6412, 10707, 18076, 19238, 302~
$ first_enrolled <int> 13200, 88467, 2340, 5894, 10298, 17006, 17843, 284~
$ last_enrolled <int> 13585, 94709, 2501, 6408, 10707, 18076, 19238, 302~
$ ssa <int> 36110, 36170, 36260, 36280, 36290, 36440, 36480, 3~
$ ncount <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
$ state_name <chr> "ohio", "ohio", "ohio", "ohio", "ohio", "ohio", "o~
$ premium <dbl> NA, 16~
$ premium_partc <dbl> NA, 1~
$ premium_partd_basic <dbl> NA, 15~
$ premium_partd_supp <dbl> NA, 0, ~
$ premium_partd_total <dbl> NA, 15~
$ partd_deductible <int> NA, 31~

```

```

$ riskscore_partc      <dbl> NA, 0.~
$ payment_partc        <dbl> NA, 68.~
$ rebate_partc          <dbl> NA, 44.~
$ payment_partd          <dbl> NA, 61.~
$ directsubsidy_partd <dbl> NA, 35.~
$ reinsurance_partd    <dbl> NA, 16.~
$ costsharing_partd    <dbl> NA, 8.~
$ riskscore_partd      <dbl> NA, 0.~
$ basic_premium         <dbl> NA, 0.~
$ bid                   <dbl> NA, 82.~

```

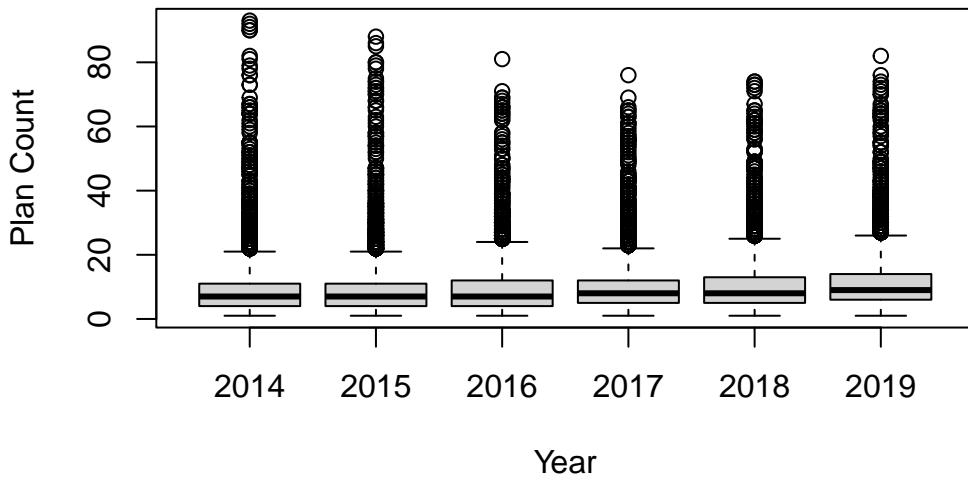
Problem 1

```
plan_counts <- data.full %>% distinct(fips, year, contractid)
```

```
plan_counts <- plan_counts %>%
  group_by(fips, year) %>%
  summarise(plan_count = n(), .groups = "drop")
```

```
boxplot(plan_count ~ year, data = plan_counts,
        xlab = "Year",
        ylab = "Plan Count",
        main = "Distribution of Plan Counts by Year")
```

Distribution of Plan Counts by Year



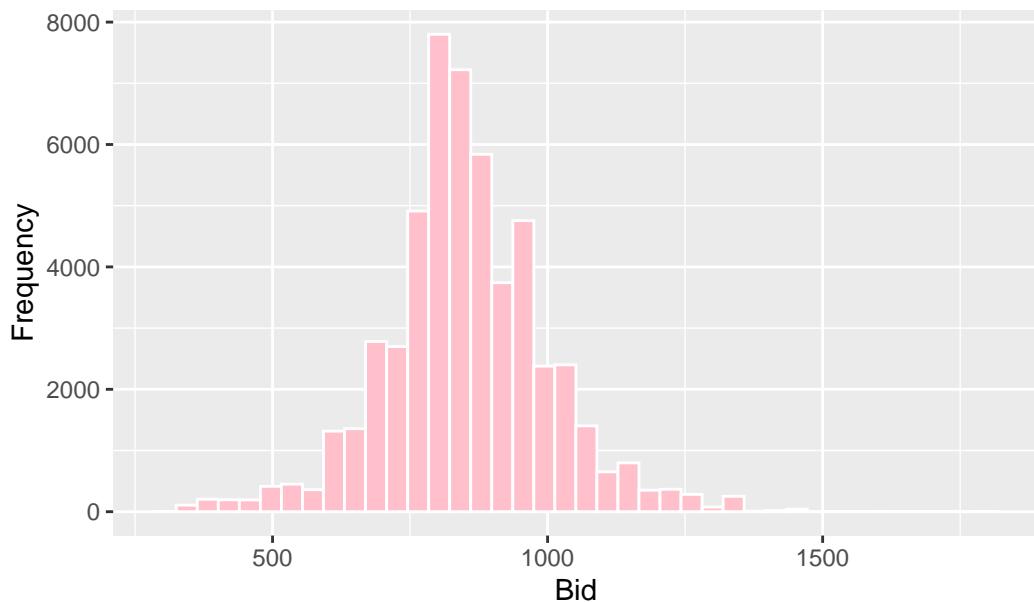
```
data.full <- data.full %>% mutate (basic_premium = case_when(
  rebate_partc > 0 ~ 0,
  partd == "No" & !is.na(premium) & is.na(premium_partc) ~ premium,
  TRUE ~ premium_partc
),
bid = case_when(
  rebate_partc == 0 & basic_premium > 0 ~ (payment_partc + basic_premium) / riskscore_partc,
  rebate_partc > 0 | basic_premium == 0 ~ payment_partc / riskscore_partc,
  TRUE ~ NA_real_
)
)
```

Problem 2

```
data.full %>%
  filter(year == 2014) %>%
  ggplot(aes(x = bid)) +
  geom_histogram(bins = 40, fill = "pink", color = "white")+
  labs(
    x = "Bid",
    y = "Frequency",
    title = "Year 2014"
)
```

Warning: Removed 8956 rows containing non-finite values (`stat_bin()`).

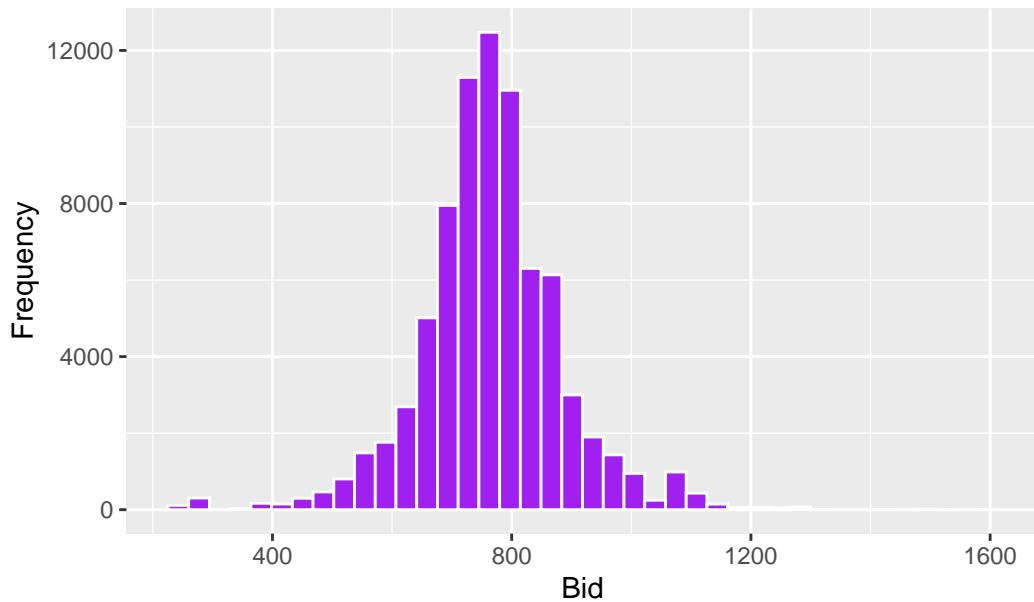
Year 2014



```
data.full %>%
  filter(year == 2018) %>%
  ggplot(aes(x = bid)) +
  geom_histogram(bins = 40, fill = "purple", color = "white")+
  labs(
    x = "Bid",
    y = "Frequency",
    title = "Year 2018"
  )
```

Warning: Removed 8094 rows containing non-finite values (`stat_bin()`).

Year 2018



Problem 3

```
hhidata <- data.full %>%
  mutate(share = avg_enrollment / avg_enrolled) %>%
  group_by(fips, year) %>%
  summarise(HHI = sum(share^2, na.rm = TRUE), .groups = "drop") %>%
  group_by(year) %>%
  summarise(mean_HHI = mean(HHI, na.rm = TRUE), .groups = "drop")
```

```
hhidata
```

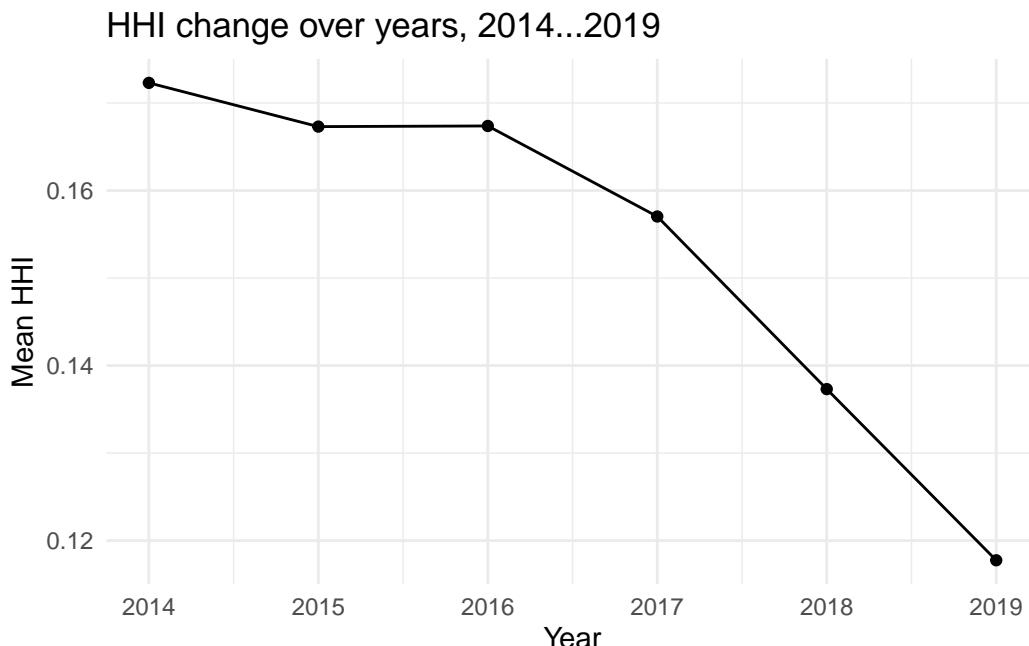
```
# A tibble: 6 x 2
  year  mean_HHI
  <int>   <dbl>
1 2014    0.172
2 2015    0.167
3 2016    0.167
4 2017    0.157
5 2018    0.137
6 2019    0.118
```

```
ggplot(hhidata, aes(x = year, y = mean_HHI)) + geom_line() + geom_point() + theme_minimal()
  x = "Year",
```

```

y = "Mean HHI",
title = "HHI change over years, 2014-2019"
)

```



Problem 4

```

ma_share_yearly <- data.full %>%
  mutate(ma_share = avg_enrolled / avg_eligibles) %>%
  group_by(year) %>%
  summarise(mean_share = mean(ma_share, na.rm = TRUE), .groups = "drop")

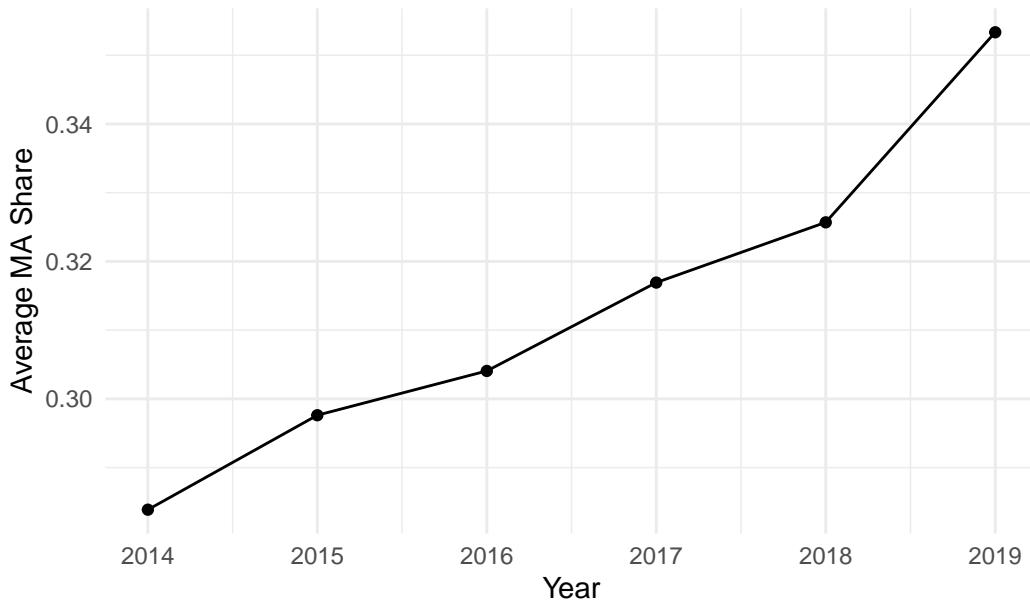
```

```

ggplot(ma_share_yearly, aes(x = year, y = mean_share)) +
  geom_line() +
  geom_point() +
  theme_minimal() +
  labs(
    x = "Year",
    y = "Average MA Share",
    title = "Average Medicare Advantage Share, 2014-2019"
  )

```

Average Medicare Advantage Share, 2014...2019



Estimate ATEs

Problem 5

```
data.full.2018 <- data.full %>% filter(year == 2018)
```

```
colnames(data.full.2018)
```

```
[1] "contractid"           "planid"                 "fips"
[4] "year"                  "n_nonmiss"              "avg_enrollment"
[7] "sd_enrollment"         "min_enrollment"        "max_enrollment"
[10] "first_enrollment"      "last_enrollment"       "state"
[13] "county"                "org_type"               "plan_type"
[16] "partd"                 "snp"                    "eghp"
[19] "org_name"              "org_marketing_name"   "plan_name"
[22] "parent_org"            "contract_date"         "state_long"
[25] "county_long"           "n_elig"                "n_enrol"
[28] "avg_eligibles"         "sd_eligibles"          "min_eligibles"
[31] "max_eligibles"          "first_eligibles"        "last_eligibles"
[34] "avg_enrolled"           "sd_enrolled"           "min_enrolled"
[37] "max_enrolled"           "first_enrolled"         "last_enrolled"
[40] "ssa"                    "ncount"                "state_name"
[43] "premium"                "premium_partc"          "premium_partd_basic"
[46] "premium_partd_supp"     "premium_partd_total"    "partd_deductible"
```

```
[49] "riskscore_partc"      "payment_partc"       "rebate_partc"  
[52] "payment_partd"       "directsubsidy_partd" "reinsurance_partd"  
[55] "costsharing_partd"   "riskscore_partd"     "basic_premium"  
[58] "bid"
```

```
hhidata_2018 <- data.full.2018 %>%  
  mutate(share = avg_enrollment / avg_enrolled) %>%  
  group_by(fips, year) %>%  
  mutate(HHI = sum(share^2, na.rm = TRUE)) %>%  
  ungroup()
```

```
hhidata_33 <- quantile(hhidata_2018$HHI, 0.33, na.rm = TRUE)  
hhidata_66 <- quantile(hhidata_2018$HHI, 0.66, na.rm = TRUE)
```

```
hhihigh <- hhidata_2018 %>% filter(hhidata_2018$HHI >= hhidata_66)  
hhilow <- hhidata_2018 %>% filter(hhidata_2018$HHI <= hhidata_33)
```

```
avg_high <- hhihigh %>% summarise(avg_66 = mean(bid, na.rm = TRUE))  
avg_low <- hhilow %>% summarise(avg_33 = mean(bid, na.rm = TRUE))  
  
cat("Average Bid in Uncompetitive Markets:", avg_high$avg_66, "\n")
```

Average Bid in Uncompetitive Markets: 770.536

```
cat("Average Bid in Competitive Markets:", avg_low$avg_33, "\n")
```

Average Bid in Competitive Markets: 767.0241

Problem 6

```
data.2018.ffs <- read.csv('..../data/output/data-2018-ffs.csv')  
  
data.2018.ffs <- data.2018.ffs %>% mutate(ffs_quartile = ntile(avg_ffscost, 4))  
  
results <- lapply(1:4, function(q) {  
  treatment <- data.2018.ffs %>%  
    filter(ffs_quartile == q) %>%  
    summarise(avg_bid_treat = mean(bid, na.rm = TRUE))
```

```

control <- data.2018.ffd %>%
  filter(ffd_quartile != q) %>%
  summarise(avg_bid_control = mean(bid, na.rm = TRUE))

data.frame(
  quartile = q,
  avg_bid_treat = treatment$avg_bid_treat,
  avg_bid_control = control$avg_bid_control
)
})

results_table <- do.call(rbind, results)

print(results_table)

```

	quartile	avg_bid_treat	avg_bid_control
1	1	778.7275	761.3458
2	2	769.9289	764.2578
3	3	758.0963	768.2317
4	4	756.0814	768.8580

Problem 7

Explanation: Here by pooled data we meant grouping of all 1s in each quartile as a total group of treated entities, and then all 0s together from each quartile. On which, we run our ATE estimators.

```

options(repos = c(CRAN = "https://cloud.r-project.org"))
install.packages("MatchIt")

```

```

Installing package into '/home/ssark38/R/x86_64-conda-linux-gnu-library/4.3'
(as 'lib' is unspecified)

```

```

Warning in install.packages("MatchIt"): installation of package 'MatchIt' had
non-zero exit status

```

```

pooled_data <- lapply(1:4, function(q) {

  dat_q <- data.2018.ffd %>%
    mutate(

```

```

    treat = ifelse(ffs_quartile == q, 1, 0),
    outcome = bid,
    covar = avg_ffscost
) %>%
filter(!is.na(outcome), !is.na(treat), !is.na(covar))

dat_q
})

pooled_data <- do.call(rbind, pooled_data)

```

```

library(MatchIt)

m.out <- matchit(
  treat ~ covar,
  data = pooled_data,
  method = "nearest",
  distance = pooled_data$covar,
  replace = FALSE
)

m.data <- match.data(m.out)

# ATE via simple difference in means
ate_inv_var <- mean(m.data$outcome[m.data$treat == 1]) -
  mean(m.data$outcome[m.data$treat == 0])

```

```
print(ate_inv_var)
```

[1] 0.007655655

```

m.out <- matchit(
  treat ~ covar,
  data = pooled_data,
  method = "nearest",
  distance = "mahalanobis",
  replace = FALSE
)

m.data <- match.data(m.out)

```

```

# ATE via simple difference in means
ate_mahalanobis <- mean(m.data$outcome[m.data$treat == 1]) -
  mean(m.data$outcome[m.data$treat == 0])

print(ate_mahalanobis)

[1] 0.007655655

logit.model <- glm(treat ~ covar, family = binomial, data = pooled_data)
pooled_data$ps <- fitted(logit.model)

pooled_data <- pooled_data %>%
  mutate(ipw = case_when(
    treat == 1 ~ 1/ps,
    treat == 0 ~ 1/(1 - ps)
  ))

mean.w1 <- pooled_data %>%
  filter(treat == 1) %>%
  summarize(mean_y = weighted.mean(outcome, ipw))

mean.w0 <- pooled_data %>%
  filter(treat == 0) %>%
  summarize(mean_y = weighted.mean(outcome, ipw))

ate_ipw <- mean.w1$mean_y - mean.w0$mean_y

print(ate_ipw)

[1] 1.023182e-12

reg1 <- lm(outcome ~ covar, data = pooled_data %>% filter(treat == 1))
reg0 <- lm(outcome ~ covar, data = pooled_data %>% filter(treat == 0))

pred1 <- predict(reg1, newdata = pooled_data)
pred0 <- predict(reg0, newdata = pooled_data)

ate_regression <- mean(pred1 - pred0)

```

```
print(ate_regression)
```

```
[1] 3.434138e-13
```

Problem 8. ATE calculated with inverse variance distance and Mahalanobis distance are identical, while the ones calculated with IPW and simple linear regression differ vastly.

```
results_table <- data.frame(  
    ate_inv_var = ate_inv_var,  
    ate_mahalanobis = ate_mahalanobis,      # if you really want it twice  
    ate_ipw = ate_ipw,  
    ate_regression = ate_regression  # if you really want it twice  
)  
  
results_table
```

	ate_inv_var	ate_mahalanobis	ate_ipw	ate_regression
1	0.007655655	0.007655655	1.023182e-12	3.434138e-13

Problem 9. We will use my favorite Mahalanobis distance on total Medicare beneficiaries alongside the FFS quartile.

```
pooled_data2 <- lapply(1:4, function(q) {  
  
    dat_q2 <- data.2018.ffs %>%  
        mutate(  
            treat = ifelse(ffs_quartile == q, 1, 0),  
            outcome = bid,  
            covar1 = avg_ffscost,  
            covar2 = n_enrol  
        ) %>%  
        filter(!is.na(outcome), !is.na(treat), !is.na(covar1), !is.na(covar2))  
  
    dat_q2  
})  
  
pooled_data2 <- do.call(rbind, pooled_data2)
```

```

m.out <- matchit(
  treat ~ covar1 + covar2,
  data = pooled_data2,
  method = "nearest",
  distance = "mahalanobis",
  replace = FALSE
)

m.data <- match.data(m.out)

# ATE via simple difference in means
ate_mahalanobis_new <- mean(m.data$outcome[m.data$treat == 1]) -
  mean(m.data$outcome[m.data$treat == 0])

print(ate_mahalanobis_new)

```

[1] -0.010913

The absolute value of ATE has gone up when the total number of Medicare beneficiaries is included as a covariate. It is still comparable to ATE obtained via inverse distance weighing or Mahalanobis as opposed to IPW or regression, which are approximately 0.

Problem 10

My experience was fulfilling working with these large data chunks; it really completed my prior experiences. One thing I learned is that my code runs much cleaner and is easier to navigate, as I built most of it from class notes, my concepts, and simple structural logic, rather than using LLMs that I genuinely use strictly for my personal use. One thing that surprised me was how strenuous data management could be when I had to change file names and column ranges while creating cumulative data files for each year, and generalizable RegEx expressions couldn't be deployed.