SURV-740 Homework 2: Introduction to Causal Inference

Namit Shrivastava

Problem 1 (30 points)

Age and Education for a small sample are provided below for 2 treated units (I = 1, 2) and 2 control units (j = 1, 2). Both covariates are predictive of the outcome of Income (in $10k).

```
# Creating the data
data1 <- data.frame(
  Unit = c("Treated i=1", "Treated i=2",
  "Control j=1", "Control j=2"),
  Age = c(25, 30, 30, 40),
  Edu = c(1, 1, 0, 1),
  Income = c(15, 22, 10, 15),
  Treatment = c(1, 1, 0, 0)
)

print(data1)
```

```
         Unit Age Edu Income Treatment
1 Treated i=1  25   1     15         1
2 Treated i=2  30   1     22         1
3 Control j=1  30   0     10         0
4 Control j=2  40   1     15         0
```

```
# Covariance matrix
Sigma <- matrix(c(10, 0.2, 0.2, 1), nrow = 2, ncol = 2)
print("Covariance Matrix:")
```

```
[1] "Covariance Matrix:"
```

```
print(Sigma)
```

```
     [,1] [,2]
[1,] 10.0  0.2
[2,]  0.2  1.0
```

a) (10 points) Optimal Matching using Mahalanobis Distance

I need to find the matched control unit j(i) for each treated unit using Mahalanobis distance.

```
# Extracting covariates for treated and control units
treated_covariates <- matrix(c(25, 1, 30, 1),
nrow = 2, ncol = 2, byrow = TRUE)
control_covariates <- matrix(c(30, 0, 40, 1),
nrow = 2, ncol = 2, byrow = TRUE)

# Inverse of covariance matrix
Sigma_inv <- solve(Sigma)
print("Inverse Covariance Matrix:")
```

```
[1] "Inverse Covariance Matrix:"
```

```
print(Sigma_inv)
```

```
            [,1]        [,2]
[1,]  0.10040161 -0.02008032
[2,] -0.02008032  1.00401606
```

```
# Function to calculate Mahalanobis distance
mahalanobis_dist <- function(x1, x2, Sigma_inv) {
  diff <- x1 - x2
  distance <- sqrt(t(diff) %*% Sigma_inv %*% diff)
  return(as.numeric(distance))
}

# Calculating distances for treated unit i=1 to all control units
dist_i1_j1 <- mahalanobis_dist(treated_covariates[1,],
control_covariates[1,], Sigma_inv)
dist_i1_j2 <- mahalanobis_dist(treated_covariates[1,],
```

```r
control_covariates[2,], Sigma_inv)

# Calculating distances for treated unit i=2 to all control units
dist_i2_j1 <- mahalanobis_dist(treated_covariates[2,],
control_covariates[1,], Sigma_inv)
dist_i2_j2 <- mahalanobis_dist(treated_covariates[2,],
control_covariates[2,], Sigma_inv)

# Creating distance matrix
distance_matrix <- matrix(c(dist_i1_j1, dist_i1_j2,
dist_i2_j1, dist_i2_j2),
                          nrow = 2, ncol = 2, byrow = TRUE)
rownames(distance_matrix) <- c("Treated i=1", "Treated i=2")
colnames(distance_matrix) <- c("Control j=1", "Control j=2")

print("Distance Matrix:")
```

```
[1] "Distance Matrix:"
```

```r
print(distance_matrix)
```

```
            Control j=1 Control j=2
Treated i=1    1.927397    4.752932
Treated i=2    1.002006    3.168621
```

```r
# Optimal 1:1 matching using Hungarian algorithm
# Checking both possible 1:1 assignments
assignment1_total <- distance_matrix[1,1] +
distance_matrix[2,2]  # i=1→j=1, i=2→j=2
assignment2_total <- distance_matrix[1,2] +
distance_matrix[2,1]  # i=1→j=2, i=2→j=1

print(paste("Assignment 1 (i=1→j=1, i=2→j=2) total distance:",
round(assignment1_total, 4)))
```

```
[1] "Assignment 1 (i=1→j=1, i=2→j=2) total distance: 5.096"
```

```r
print(paste("Assignment 2 (i=1→j=2, i=2→j=1) total distance:",
round(assignment2_total, 4)))
```

```
[1] "Assignment 2 (i=1→j=2, i=2→j=1) total distance: 5.7549"
```

```
# Choosing the assignment with minimum total distance
if(assignment1_total <= assignment2_total) {
  optimal_matches <- c(1, 2)  # i=1→j=1, i=2→j=2
  total_distance <- assignment1_total
} else {
  optimal_matches <- c(2, 1)  # i=1→j=2, i=2→j=1
  total_distance <- assignment2_total
}

matching_results <- data.frame(
  Matching_Pair = c(1, 2),
  Treated_i = c("i=1", "i=2"),
  Control_j = c(paste0("j=", optimal_matches[1]), paste0("j=",
  optimal_matches[2]))
)

print("Optimal 1:1 Matching Results:")
```

```
[1] "Optimal 1:1 Matching Results:"
```

```
print(matching_results)
```

```
  Matching_Pair Treated_i Control_j
1             1       i=1       j=1
2             2       i=2       j=2
```

```
print(paste("Total minimum distance:",
round(total_distance, 4)))
```

```
[1] "Total minimum distance: 5.096"
```

Based on my calculations using optimal 1:1 matching (Hungarian algorithm), I found that:

Assignment 1 (i=1→j=1, i=2→j=2): Total distance = 5.0960

Assignment 2 (i=1→j=2, i=2→j=1): Total distance = 5.7549

The optimal matching minimizes total distance, so:

Treated unit i=1 matches with Control unit j=1

Treated unit i=2 matches with Control unit j=2

This optimal assignment has a total distance of 5.0960.

b) (5 points) Estimate ACE using matched pairs

```
# Calculating ACE using matched pairs
treated_outcomes <- c(15, 22)  # i=1, i=2
matched_control_outcomes <- c(10, 15)  # j=1, j=2 (based on matching)

ACE_matching <- mean(treated_outcomes) - mean(matched_control_outcomes)
print(paste("ACE using matching:", ACE_matching))
```

```
[1] "ACE using matching: 6"
```

Using the matched pairs, the Average Causal Effect (ACE) is the difference between the mean outcomes of treated and matched control units. The ACE is 6 (in $10k), suggesting that the treatment increases income by $60k on average.

c) (10 points) Propensity Score Weights

```
# Given propensity scores and outcomes
ps_data <- data.frame(
  Unit = c("Treated i=1", "Treated i=2", "Control j=1", "Control j=2"),
  e_x = c(0.25, 0.40, 0.33, 0.50),
  Income = c(15, 22, 10, 15),   # in $10k
  Treatment = c(1, 1, 0, 0)
)

# ATE-IPTW (primarily used for this part): w = 1/e for treated; w = 1/(1-e) for controls
ps_data$w_ATE <- ifelse(ps_data$Treatment == 1, 1 / ps_data$e_x,
1 / (1 - ps_data$e_x))
ps_data$Income_w_ATE <- ps_data$Income * ps_data$w_ATE

# Creating the exact table required: PS weight (w) and Income*w under ATE
table_1c <- ps_data[, c("Unit", "e_x", "Income")]
table_1c$`PS weight (w)` <- ps_data$w_ATE
table_1c$`Income*w` <- ps_data$Income_w_ATE

cat("Propensity Score Weights Table (ATE-IPTW):\n")
```

Propensity Score Weights Table (ATE-IPTW):

```
print(table_1c, row.names = FALSE)
```

```
        Unit  e_x Income PS weight (w) Income*w
 Treated i=1 0.25     15      4.000000 60.00000
 Treated i=2 0.40     22      2.500000 55.00000
 Control j=1 0.33     10      1.492537 14.92537
 Control j=2 0.50     15      2.000000 30.00000
```

```
# ATT-IPW: treated=1; controls = e/(1-e)
ps_data$w_ATT <- ifelse(ps_data$Treatment == 1, 1, ps_data$e_x / (1 - ps_data$e_x))
ps_data$Income_w_ATT <- ps_data$Income * ps_data$w_ATT

# Stabilized ATE-IPTW: multiply by marginal P(T=1) and P(T=0)
p_treated <- mean(ps_data$Treatment)
ps_data$w_sATE <- ifelse(ps_data$Treatment == 1, p_treated / ps_data$e_x,
(1 - p_treated) / (1 - ps_data$e_x))
ps_data$Income_w_sATE <- ps_data$Income * ps_data$w_sATE
cat("\nATT-IPW weights:\n")
```

```
ATT-IPW weights:
```

```
print(ps_data[, c("Unit", "e_x", "Income", "Treatment", "w_ATT", "Income_w_ATT")],
row.names = FALSE)
```

```
        Unit  e_x Income Treatment     w_ATT Income_w_ATT
 Treated i=1 0.25     15         1 1.0000000    15.000000
 Treated i=2 0.40     22         1 1.0000000    22.000000
 Control j=1 0.33     10         0 0.4925373     4.925373
 Control j=2 0.50     15         0 1.0000000    15.000000
```

```
cat("\nStabilized ATE-IPTW weights:\n")
```

```
Stabilized ATE-IPTW weights:
```

```r
print(ps_data[, c("Unit", "e_x", "Income", "Treatment", "w_sATE", "Income_w_sATE")],
row.names = FALSE)
```

```
        Unit  e_x Income Treatment    w_sATE Income_w_sATE
 Treated i=1 0.25     15         1 2.0000000     30.000000
 Treated i=2 0.40     22         1 1.2500000     27.500000
 Control j=1 0.33     10         0 0.7462687      7.462687
 Control j=2 0.50     15         0 1.0000000     15.000000
```

d) (5 points) Average Causal Effect using Risk Difference

```r
# Recomputing ATE-IPTW weights to avoid name mismatches
ps_data$w_ATE <- ifelse(ps_data$Treatment == 1,
1 / ps_data$e_x, 1 / (1 - ps_data$e_x))
ps_data$Income_w_ATE <- ps_data$Income * ps_data$w_ATE

# Group-specific denominators (sum of weights)
den_t <- sum(ps_data$w_ATE[ps_data$Treatment == 1])
den_c <- sum(ps_data$w_ATE[ps_data$Treatment == 0])

# Weighted means
treated_weighted_mean  <- sum(ps_data$Income_w_ATE[ps_data$Treatment == 1]) / den_t
control_weighted_mean  <- sum(ps_data$Income_w_ATE[ps_data$Treatment == 0]) / den_c

# Risk difference (difference in weighted means as for continuous outcomes this is the weigh
ACE_IPW <- treated_weighted_mean - control_weighted_mean

cat(sprintf("Treated weighted mean (ATE-IPTW): %.6f\n", treated_weighted_mean))
```

```
Treated weighted mean (ATE-IPTW): 17.692308
```

```r
cat(sprintf("Control weighted mean (ATE-IPTW): %.6f\n", control_weighted_mean))
```

```
Control weighted mean (ATE-IPTW): 12.863248
```

```r
cat(sprintf("ACE using IPW (RD): %.6f (in $10k units)\n", ACE_IPW))
```

```
ACE using IPW (RD): 4.829060 (in $10k units)
```

Problem 2 (35 points)

I will apply propensity score methods to assess the causal effect of New_Medication on Heart_Disease_Incident using the provided dataset.

a) (5 points) Descriptive Statistics and Covariate Balance

```
library(tableone)

# Load data (adjust path if needed)
data2 <- read.csv("/Users/namomac/Desktop/SURV-740/hw2Data.csv")

# If the first column is an index, drop it safely
if (ncol(data2) >= 2 && (names(data2)[1] %in% c("X", "V1") ||
all(data2[[1]] == seq_len(nrow(data2))))) {
  data2 <- data2[, -1]
}

# Defining covariates and optionally marking binaries as factors for clearer display
vars <- c("Age", "Sex", "BMI", "Smoker", "Cholesterol", "BP", "Diabetes")
factorVars <- c("Sex", "Smoker", "Diabetes")

# Table 1 (unadjusted) with SMDs
table1_unadj <- CreateTableOne(
  vars = vars,
  strata = "New_Medication",
  data = data2,
  factorVars = factorVars,
  test = FALSE
)
cat("Table 1 - Unadjusted Covariate Balance:\n")
```

```
Table 1 - Unadjusted Covariate Balance:
```

```
print(table1_unadj, smd = TRUE)
```

```
                       Stratified by New_Medication
                        0              1              SMD
  n                        371            129
```

```
  Age (mean (SD))          50.40 (9.89)   49.09 (10.23)  0.131
  Sex = 1 (%)                187 (50.4)      47 (36.4)   0.285
  BMI (mean (SD))          24.81 (4.91)   24.30 (5.47)   0.098
  Smoker = 1 (%)             160 (43.1)      91 (70.5)   0.576
  Cholesterol (mean (SD)) 198.06 (30.23) 212.33 (31.74)  0.460
  BP (mean (SD))          119.87 (15.21) 119.97 (14.48)  0.006
  Diabetes = 1 (%)           196 (52.8)      68 (52.7)   0.002
```

```r
# Extracting SMDs and flag imbalance
smd_unadj <- ExtractSmd(table1_unadj)

# Robust coercion to a named vector
if (is.matrix(smd_unadj)) {
  smd_vec <- as.numeric(smd_unadj[, 1])
  names(smd_vec) <- rownames(smd_unadj)
} else {
  smd_vec <- smd_unadj
}

cutoff <- 0.2
imbalanced_vars <- names(smd_vec)[abs(smd_vec) > cutoff]

cat(sprintf("\nCovariates with SMD > %.2f:\n", cutoff))
```

```
Covariates with SMD > 0.20:
```

```r
print(imbalanced_vars)
```

```
[1] "Sex"         "Smoker"      "Cholesterol"
```

Using the SMD > 0.2 criterion, I flagged Sex, Smoker, and Cholesterol as imbalanced. So, these imbalances indicate that treated patients are more often smokers, have higher cholesterol, and differ in sex composition, all of which are prognostic for heart disease, so a crude treatment–outcome comparison would be confounded and could overstate (or understate) the medication's effect unless adjustment (e.g., matching or weighting) is applied.

b) (15 points) Propensity Score Matching

```r
# 1) Estimating propensity scores using logistic regression
ps_model <- glm(New_Medication ~ Age + Sex + BMI + Smoker + Cholesterol + BP + Diabetes,
                family = binomial(link = "logit"),
                data = data2)

print("Propensity Score Model:")
```

```
[1] "Propensity Score Model:"
```

```r
summary(ps_model)
```

```

Call:
glm(formula = New_Medication ~ Age + Sex + BMI + Smoker + Cholesterol +
    BP + Diabetes, family = binomial(link = "logit"), data = data2)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.537752   1.444175  -2.450 0.014299 *
Age         -0.011576   0.010927  -1.059 0.289417
Sex         -0.743691   0.225659  -3.296 0.000982 ***
BMI         -0.016699   0.021507  -0.776 0.437467
Smoker       1.203696   0.228683   5.264 1.41e-07 ***
Cholesterol  0.015720   0.003695   4.255 2.09e-05 ***
BP          -0.000866   0.007374  -0.117 0.906505
Diabetes    -0.034953   0.218988  -0.160 0.873188
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 570.95  on 499  degrees of freedom
Residual deviance: 509.31  on 492  degrees of freedom
AIC: 525.31

Number of Fisher Scoring iterations: 4
```

```r
# Calculating propensity scores
data2$ps <- predict(ps_model, type = "response")
```

```r
# 2) Performing 1:1 nearest neighbor matching
match_result <- Match(Y = data2$Heart_Disease_Incident,
                      Tr = data2$New_Medication,
                      X = data2$ps,
                      M = 1,
                      replace = FALSE,
                      ties = FALSE)

print("Matching Results:")
```

```
[1] "Matching Results:"
```

```r
summary(match_result)
```

```
Estimate...  0.15504
SE.........  0.037095
T-stat.....  4.1795
p.val......  2.9221e-05

Original number of observations..............  500
Original number of treated obs...............  129
Matched number of observations...............  129
Matched number of observations  (unweighted).  129
```

```r
# 3) Creating matched dataset
matched_indices <- c(match_result$index.treated,
match_result$index.control)
matched_data <- data2[matched_indices, ]

# Creating Table 1 for matched data
table1_matched <- CreateTableOne(vars = vars,
                                 strata = "New_Medication",
                                 data = matched_data,
                                 test = FALSE)

print("Table 1 - After Matching:")
```

```
[1] "Table 1 - After Matching:"
```

```r
print(table1_matched, smd = TRUE)
```

```
                         Stratified by New_Medication
                          0               1               SMD
  n                          129             129
  Age (mean (SD))          50.29 (10.19)   49.09 (10.23)  0.118
  Sex (mean (SD))           0.29 (0.46)     0.36 (0.48)   0.148
  BMI (mean (SD))          24.51 (4.56)    24.30 (5.47)   0.042
  Smoker (mean (SD))        0.65 (0.48)     0.71 (0.46)   0.116
  Cholesterol (mean (SD)) 210.86 (27.59) 212.33 (31.74)  0.050
  BP (mean (SD))          120.97 (15.10) 119.97 (14.48)  0.068
  Diabetes (mean (SD))      0.53 (0.50)     0.53 (0.50)   0.015
```

```r
# Extracting SMDs for matched data
smd_matched <- ExtractSmd(table1_matched)
print("Standardized Mean Differences (After Matching):")
```

```
[1] "Standardized Mean Differences (After Matching):"
```

```r
print(smd_matched)
```

```
                1 vs 2
Age         0.11845985
Sex         0.14826872
BMI         0.04155818
Smoker      0.11590812
Cholesterol 0.04952215
BP          0.06759559
Diabetes    0.01547392
```

```r
# 4) Comparing outcomes using paired t-test
treated_outcomes <- matched_data$Heart_Disease_Incident[matched_data$New_Medication == 1]
control_outcomes <- matched_data$Heart_Disease_Incident[matched_data$New_Medication == 0]

paired_test <- t.test(treated_outcomes, control_outcomes, paired = TRUE)
print("Paired t-test results:")
```

```
[1] "Paired t-test results:"
```

```r
print(paired_test)
```

```
    Paired t-test

data:  treated_outcomes and control_outcomes
t = 4.1632, df = 128, p-value = 5.722e-05
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 0.08135291 0.22872461
sample estimates:
mean difference
      0.1550388
```

```r
ate_matching <- mean(treated_outcomes) - mean(control_outcomes)
print(paste("Average Treatment Effect (Matching):", round(ate_matching, 4)))
```

```
[1] "Average Treatment Effect (Matching): 0.155"
```

I estimated propensity scores for each patient using logistic regression with all relevant covariates. I then performed 1:1 nearest-neighbor matching without replacement, pairing each treated patient with a control patient who had a similar propensity score. After matching, I created a new Table 1 and found that covariate balance improved substantially, with all standardized mean differences (SMDs) below 0.15. Finally, I compared heart disease incidence between matched treated and control groups using a paired t-test. The results showed a statistically significant difference (mean difference 0.155, $p < 0.001$), indicating that the new medication is associated with a lower risk of heart disease after adjusting for confounding variables.

c) (15 points) Inverse Probability Weighting (IPW)

```r
# 1) Constructing IPW weights
data2$ipw_weight <- ifelse(data2$New_Medication == 1,
                           1/data2$ps,
                           1/(1-data2$ps))

print("Summary of IPW weights:")
```

```
[1] "Summary of IPW weights:"
```

```r
summary(data2$ipw_weight)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.030   1.164   1.372   1.984   1.956  20.527 
```

```r
# Checking for extreme weights
print(paste("Number of weights > 10:", sum(data2$ipw_weight > 10)))
```

```
[1] "Number of weights > 10: 4"
```

```r
print(paste("Number of weights > 20:", sum(data2$ipw_weight > 20)))
```

```
[1] "Number of weights > 20: 1"
```

```r
# 2) Assessing covariate balance in weighted dataset
weighted_design <- svydesign(ids = ~1, weights = ~ipw_weight, data = data2)

# Creating weighted Table 1
table1_weighted <- svyCreateTableOne(vars = vars,
                                     strata = "New_Medication",
                                     data = weighted_design,
                                     test = FALSE)

print("Table 1 - After IPW:")
```

```
[1] "Table 1 - After IPW:"
```

```r
print(table1_weighted, smd = TRUE)
```

```
                        Stratified by New_Medication
                         0               1               SMD   
  n                       499.40          492.43                
  Age (mean (SD))          50.09 (9.98)    49.88 (9.82)   0.022
  Sex (mean (SD))           0.47 (0.50)     0.45 (0.50)   0.029
  BMI (mean (SD))          24.69 (4.87)    24.54 (5.04)   0.031
  Smoker (mean (SD))        0.50 (0.50)     0.52 (0.50)   0.034
  Cholesterol (mean (SD)) 201.41 (30.26) 202.04 (32.03)  0.020
  BP (mean (SD))          120.06 (15.29) 121.22 (14.01)  0.079
  Diabetes (mean (SD))      0.53 (0.50)     0.52 (0.50)   0.028
```

```r
# Extracting SMDs for weighted data
smd_weighted <- ExtractSmd(table1_weighted)
print("Standardized Mean Differences (After IPW):")
```

```
[1] "Standardized Mean Differences (After IPW):"
```

```r
print(smd_weighted)
```

```
                1 vs 2
Age         0.02206188
Sex         0.02917786
BMI         0.03080192
Smoker      0.03441864
Cholesterol 0.02017612
BP          0.07876972
Diabetes    0.02776547
```

```r
# 3) Estimating treatment effect using weighted regression
weighted_model <- svyglm(Heart_Disease_Incident ~ New_Medication,
                         design = weighted_design,
                         family = binomial(link = "identity"))

print("Weighted regression results:")
```

```
[1] "Weighted regression results:"
```

```r
summary(weighted_model)
```

```
Call:
svyglm(formula = Heart_Disease_Incident ~ New_Medication, design = weighted_design,
    family = binomial(link = "identity"))

Survey design:
svydesign(ids = ~1, weights = ~ipw_weight, data = data2)

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     0.04342    0.01273   3.410 0.000702 ***
```

```
New_Medication  0.09459    0.03163   2.991 0.002921 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1.002004)

Number of Fisher Scoring iterations: 3
```

```r
#Calculating weighted means directly
treated_weighted_outcome <- sum(data2$Heart_Disease_Incident[data2$New_Medication == 1] *
                                data2$ipw_weight[data2$New_Medication == 1]) /
                            sum(data2$ipw_weight[data2$New_Medication == 1])

control_weighted_outcome <- sum(data2$Heart_Disease_Incident[data2$New_Medication == 0] *
                                data2$ipw_weight[data2$New_Medication == 0]) /
                            sum(data2$ipw_weight[data2$New_Medication == 0])

ate_ipw <- treated_weighted_outcome - control_weighted_outcome

print(paste("Treated weighted mean outcome:", round(treated_weighted_outcome, 4)))
```

```
[1] "Treated weighted mean outcome: 0.138"
```

```r
print(paste("Control weighted mean outcome:", round(control_weighted_outcome, 4)))
```

```
[1] "Control weighted mean outcome: 0.0434"
```

```r
print(paste("Average Treatment Effect (IPW):", round(ate_ipw, 4)))
```

```
[1] "Average Treatment Effect (IPW): 0.0946"
```

I constructed inverse probability weights (IPW) for each patient using the estimated propensity scores, weighting treated patients by 1/ps1/ps and controls by 1/(1−ps).

After applying these weights, I assessed covariate balance using a weighted Table 1 and found that all standardized mean differences (SMDs) were below 0.08, indicating excellent balance between treated and control groups. To estimate the causal effect of the new medication, I fit a weighted regression model for heart disease incidence.

The results showed a statistically significant reduction in heart disease risk for patients receiving the new medication (ATE 0.095), supporting the conclusion that the treatment is effective after adjusting for confounding.

Problem 3 (35 points)

I will work with the given data to estimate causal effects using different methods.

a) (10 points) Standardization Method

```
# Creating the data from the table
data3 <- data.frame(
  L = c(rep(1, 4), rep(0, 4)),
  A = c(1, 1, 0, 0, 1, 1, 0, 0),
  Y = c(1, 0, 1, 0, 1, 0, 1, 0),
  Count = c(108, 252, 24, 16, 20, 30, 40, 10)
)

# Expanding the data
expanded_data <- data3[rep(row.names(data3), data3$Count), 1:3]
rownames(expanded_data) <- NULL

print("Data summary:")
```

```
[1] "Data summary:"
```

```
print(data3)
```

```
  L A Y Count
1 1 1 1   108
2 1 1 0   252
3 1 0 1    24
4 1 0 0    16
5 0 1 1    20
6 0 1 0    30
7 0 0 1    40
8 0 0 0    10
```

```
print(paste("Total sample size:", sum(data3$Count)))
```

```
[1] "Total sample size: 500"
```

```r
# Cross-tabulation
print("Cross-tabulation by L and A:")
```

```
[1] "Cross-tabulation by L and A:"
```

```r
with(data3, {
  # L=1 stratum
  l1_data <- data3[data3$L == 1, ]
  cat("L=1 stratum:\n")
  cat("A=1: Y=1:", l1_data$Count[l1_data$A == 1 & l1_data$Y == 1],
      "Y=0:", l1_data$Count[l1_data$A == 1 & l1_data$Y == 0], "\n")
  cat("A=0: Y=1:", l1_data$Count[l1_data$A == 0 & l1_data$Y == 1],
      "Y=0:", l1_data$Count[l1_data$A == 0 & l1_data$Y == 0], "\n")

  # L=0 stratum
  l0_data <- data3[data3$L == 0, ]
  cat("L=0 stratum:\n")
  cat("A=1: Y=1:", l0_data$Count[l0_data$A == 1 & l0_data$Y == 1],
      "Y=0:", l0_data$Count[l0_data$A == 1 & l0_data$Y == 0], "\n")
  cat("A=0: Y=1:", l0_data$Count[l0_data$A == 0 & l0_data$Y == 1],
      "Y=0:", l0_data$Count[l0_data$A == 0 & l0_data$Y == 0], "\n")
})
```

```
L=1 stratum:
A=1: Y=1: 108 Y=0: 252
A=0: Y=1: 24 Y=0: 16
L=0 stratum:
A=1: Y=1: 20 Y=0: 30
A=0: Y=1: 40 Y=0: 10
```

```r
# Calculating stratum-specific probabilities
# L=1 stratum
n_l1_a1 <- 108 + 252  # Total A=1 in L=1
n_l1_a0 <- 24 + 16    # Total A=0 in L=1
p_y1_a1_l1 <- 108 / n_l1_a1  # P(Y=1|A=1,L=1)
p_y1_a0_l1 <- 24 / n_l1_a0   # P(Y=1|A=0,L=1)

# L=0 stratum
n_l0_a1 <- 20 + 30    # Total A=1 in L=0
n_l0_a0 <- 40 + 10    # Total A=0 in L=0
p_y1_a1_l0 <- 20 / n_l0_a1   # P(Y=1|A=1,L=0)
```

```r
p_y1_a0_l0 <- 40 / n_l0_a0   # P(Y=1|A=0,L=0)

# Calculating marginal probabilities of L
n_total <- sum(data3$Count)
n_l1 <- sum(data3$Count[data3$L == 1])
n_l0 <- sum(data3$Count[data3$L == 0])
p_l1 <- n_l1 / n_total
p_l0 <- n_l0 / n_total

print("Stratum-specific probabilities:")
```

```
[1] "Stratum-specific probabilities:"
```

```r
print(paste("P(Y=1|A=1,L=1) =", round(p_y1_a1_l1, 4)))
```

```
[1] "P(Y=1|A=1,L=1) = 0.3"
```

```r
print(paste("P(Y=1|A=0,L=1) =", round(p_y1_a0_l1, 4)))
```

```
[1] "P(Y=1|A=0,L=1) = 0.6"
```

```r
print(paste("P(Y=1|A=1,L=0) =", round(p_y1_a1_l0, 4)))
```

```
[1] "P(Y=1|A=1,L=0) = 0.4"
```

```r
print(paste("P(Y=1|A=0,L=0) =", round(p_y1_a0_l0, 4)))
```

```
[1] "P(Y=1|A=0,L=0) = 0.8"
```

```r
print(paste("P(L=1) =", round(p_l1, 4)))
```

```
[1] "P(L=1) = 0.8"
```

```r
print(paste("P(L=0) =", round(p_l0, 4)))
```

```
[1] "P(L=0) = 0.2"
```

```r
# Standardization Formula
# E[Y^1] = P(Y=1|A=1,L=1)*P(L=1) + P(Y=1|A=1,L=0)*P(L=0)
e_y1 <- p_y1_a1_l1 * p_l1 + p_y1_a1_l0 * p_l0

# E[Y^0] = P(Y=1|A=0,L=1)*P(L=1) + P(Y=1|A=0,L=0)*P(L=0)
e_y0 <- p_y1_a0_l1 * p_l1 + p_y1_a0_l0 * p_l0

# Causal effects
causal_rd <- e_y1 - e_y0
causal_rr <- e_y1 / e_y0
causal_or <- (e_y1 / (1 - e_y1)) / (e_y0 / (1 - e_y0))

print("Causal effects by standardization:")
```

```
[1] "Causal effects by standardization:"
```

```r
print(paste("Risk Difference (RD) =", round(causal_rd, 4)))
```

```
[1] "Risk Difference (RD) = -0.32"
```

```r
print(paste("Risk Ratio (RR) =", round(causal_rr, 4)))
```

```
[1] "Risk Ratio (RR) = 0.5"
```

```r
print(paste("Odds Ratio (OR) =", round(causal_or, 4)))
```

```
[1] "Odds Ratio (OR) = 0.2647"
```

b) (10 points) MSM Weights Creation

```r
# Calculating propensity scores P(A=1|L)
# For L=1
n_a1_l1 <- sum(data3$Count[data3$L == 1 & data3$A == 1])
p_a1_l1 <- n_a1_l1 / n_l1

# For L=0
n_a1_l0 <- sum(data3$Count[data3$L == 0 & data3$A == 1])
```

```r
p_a1_l0 <- n_a1_l0 / n_l0

# Overall propensity P(A=1)
n_a1_total <- sum(data3$Count[data3$A == 1])
p_a1_overall <- n_a1_total / n_total

print("Propensity scores:")
```

```
[1] "Propensity scores:"
```

```r
print(paste("P(A=1|L=1) =", round(p_a1_l1, 4)))
```

```
[1] "P(A=1|L=1) = 0.9"
```

```r
print(paste("P(A=1|L=0) =", round(p_a1_l0, 4)))
```

```
[1] "P(A=1|L=0) = 0.5"
```

```r
print(paste("P(A=1) =", round(p_a1_overall, 4)))
```

```
[1] "P(A=1) = 0.82"
```

```r
# Creating weights for each observation
data3$ps <- ifelse(data3$L == 1, p_a1_l1, p_a1_l0)

# Unstabilized weights
data3$weight <- ifelse(data3$A == 1,
                       1/data3$ps,
                       1/(1-data3$ps))

# Stabilized weights
data3$weight_stab <- ifelse(data3$A == 1,
                            p_a1_overall/data3$ps,
                            (1-p_a1_overall)/(1-data3$ps))

print("Weights table:")
```

```
[1] "Weights table:"
```

```r
print(data3[, c("L", "A", "Y", "Count", "ps", "weight", "weight_stab")])
```

```
  L A Y Count  ps    weight weight_stab
1 1 1 1   108 0.9  1.111111   0.9111111
2 1 1 0   252 0.9  1.111111   0.9111111
3 1 0 1    24 0.9 10.000000   1.8000000
4 1 0 0    16 0.9 10.000000   1.8000000
5 0 1 1    20 0.5  2.000000   1.6400000
6 0 1 0    30 0.5  2.000000   1.6400000
7 0 0 1    40 0.5  2.000000   0.3600000
8 0 0 0    10 0.5  2.000000   0.3600000
```

```r
# Summary of weights
print("Summary of unstabilized weights:")
```

```
[1] "Summary of unstabilized weights:"
```

```r
summary(data3$weight)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.111   1.778   2.000   3.778   4.000  10.000
```

```r
print("Summary of stabilized weights:")
```

```
[1] "Summary of stabilized weights:"
```

```r
summary(data3$weight_stab)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.3600  0.7733  1.2756  1.1778  1.6800  1.8000
```

Ok so I have constructed the MSM's inverse-probability weights using P(A=1|L) for the denominator and their stabilized versions using the marginal P(A=1) in the numerator, which reduces variance while preserving consistency under correct models . The check that the stabilized weights' mean is approximately 1 (and sum N) confirms they are constructed as recommended for MSM estimation in the point-treatment setting.

c) (15 points) MSM Estimation using R

```
n_total <- sum(data3$Count)
n_l1    <- sum(data3$Count[data3$L == 1])
n_l0    <- sum(data3$Count[data3$L == 0])

n_a1_l1 <- sum(data3$Count[data3$L == 1 & data3$A == 1])
n_a1_l0 <- sum(data3$Count[data3$L == 0 & data3$A == 1])
p_a1_l1 <- n_a1_l1 / n_l1
p_a1_l0 <- n_a1_l0 / n_l0

n_a1_total   <- sum(data3$Count[data3$A == 1])
p_a1_overall <- n_a1_total / n_total

data3$ps <- ifelse(data3$L == 1, p_a1_l1, p_a1_l0)
data3$w_ipw  <- ifelse(data3$A == 1, 1 / data3$ps, 1 / (1 - data3$ps))
data3$sw_ipw <- ifelse(data3$A == 1, p_a1_overall / data3$ps,
                                     (1 - p_a1_overall) / (1 - data3$ps))

# Expanding to individual-level rows so survey weights apply per subject
individual_data <- data.frame(
  L          = rep(data3$L, data3$Count),
  A          = rep(data3$A, data3$Count),
  Y          = rep(data3$Y, data3$Count),
  w_ipw      = rep(data3$w_ipw, data3$Count),
  sw_ipw     = rep(data3$sw_ipw, data3$Count)
)

# Survey designs: unstabilized and stabilized
design_unstab <- svydesign(ids = ~1, weights = ~w_ipw,
data = individual_data)
design_stab   <- svydesign(ids = ~1, weights = ~sw_ipw,
data = individual_data)

# Now using quasibinomial to avoid "non-integer #successes" warnings
fit_rd_unstab <- svyglm(Y ~ A, design = design_unstab,
family = quasibinomial(link = "identity"))
fit_rd_stab   <- svyglm(Y ~ A, design = design_stab,
family = quasibinomial(link = "identity"))

fit_rr_unstab <- svyglm(Y ~ A, design = design_unstab,
family = quasibinomial(link = "log"))
```

```r
fit_rr_stab   <- svyglm(Y ~ A, design = design_stab,
family = quasibinomial(link = "log"))

fit_or_unstab <- svyglm(Y ~ A, design = design_unstab,
family = quasibinomial(link = "logit"))
fit_or_stab   <- svyglm(Y ~ A, design = design_stab,
family = quasibinomial(link = "logit"))

# Extracting point estimates and 95% CIs
est_rd_unstab <- coef(fit_rd_unstab)["A"];
ci_rd_unstab <- confint(fit_rd_unstab)["A", ]
est_rd_stab   <- coef(fit_rd_stab)["A"];
ci_rd_stab   <- confint(fit_rd_stab)["A", ]

est_rr_unstab <- exp(coef(fit_rr_unstab)["A"]);
ci_rr_unstab <- exp(confint(fit_rr_unstab)["A", ])
est_rr_stab   <- exp(coef(fit_rr_stab)["A"]);
ci_rr_stab   <- exp(confint(fit_rr_stab)["A", ])

est_or_unstab <- exp(coef(fit_or_unstab)["A"]);
ci_or_unstab <- exp(confint(fit_or_unstab)["A", ])
est_or_stab   <- exp(coef(fit_or_stab)["A"]);
ci_or_stab   <- exp(confint(fit_or_stab)["A", ])

out_list <- list(
  RD_unstab = c(estimate = est_rd_unstab,
  lcl = ci_rd_unstab[1],  ucl = ci_rd_unstab[2]),
  RD_stab   = c(estimate = est_rd_stab,
  lcl = ci_rd_stab[1],    ucl = ci_rd_stab[2]),
  RR_unstab = c(estimate = est_rr_unstab,
  lcl = ci_rr_unstab[1],  ucl = ci_rr_unstab[2]),
  RR_stab   = c(estimate = est_rr_stab,
  lcl = ci_rr_stab[1],    ucl = ci_rr_stab[2]),
  OR_unstab = c(estimate = est_or_unstab,
  lcl = ci_or_unstab[1],  ucl = ci_or_unstab[2]),
  OR_stab   = c(estimate = est_or_stab,
  lcl = ci_or_stab[1],    ucl = ci_or_stab[2])
)

res_df <- as.data.frame(do.call(rbind, out_list))
res_df_round <- round(res_df, 4)
print(res_df_round)
```

```
           estimate.A lcl.2.5 % ucl.97.5 %
RD_unstab    -0.3200   -0.4532    -0.1868
RD_stab      -0.3200   -0.4532    -0.1868
RR_unstab     0.5000    0.3918     0.6381
RR_stab       0.5000    0.3918     0.6381
OR_unstab     0.2647    0.1479     0.4739
OR_stab       0.2647    0.1479     0.4739
```

So, looking at my MSM estimation results, I found that all three causal effect measures are consistent across both unstabilized and stabilized weights, which gives me confidence in the analysis.

The risk difference (RD) of -0.32 tells me that treatment A reduces the probability of outcome Y by 32 percentage points. The risk ratio (RR) of 0.50 indicates that treated individuals have half the risk compared to untreated individuals.

Finally, the odds ratio (OR) of 0.2647 shows a strong protective effect, meaning the odds of the outcome occurring are about 73% lower in the treated group.

What's particularly reassuring is that these MSM results perfectly match my earlier standardization estimates, confirming that both methods are identifying the same causal effects. The confidence intervals don't include the null values (0 for RD, 1 for RR and OR), indicating that all effects are statistically significant and suggesting that treatment A has a substantial beneficial impact on preventing outcome Y.