Project 8: SuperLearner and Targeted Maximum Likelihood Estimation

Steven Herrera Tenorio

Introduction

Heart disease is the leading cause of death in the United States, and treating it properly is an important public health goal. However, it is a complex disease with several different risk factors and potential treatments. Physicians typically recommend changes in diet, increased exercise, and/or medication to treat symptoms, but it is difficult to determine how effective any one of these factors is in treating the disease. In this project, you will explore SuperLearner, Targeted Maximum Likelihood Estimation (TMLE), and Longitudinal Targeted Maximum Likelihood Estimation (LTMLE). Using a simulated dataset, you will explore whether taking blood pressure medication reduces mortality risk.

Data

This dataset was simulated using R (so it does not come from a previous study or other data source). It contains several variables:

- **blood_pressure_medication**: Treatment indicator for whether the individual took blood pressure medication (0 for control, 1 for treatment)
- **mortality**: Outcome indicator for whether the individual passed away from complications of heart disease (0 for no, 1 for yes)
- **age**: Age at time 1
- **sex_at_birth**: Sex assigned at birth (0 female, 1 male)
- **simplified_race**: Simplified racial category. (1: White/Caucasian, 2: Black/African American, 3: Latinx, 4: Asian American,
  5: Mixed Race/Other)
- **income_thousands**: Household income in thousands of dollars
- **college_educ**: Indicator for college education (0 for no, 1 for yes)
- **bmi**: Body mass index (BMI)
- **chol**: Cholesterol level
- **blood_pressure**: Systolic blood pressure
- **bmi_2**: BMI measured at time 2
- **chol_2**: Cholesterol measured at time 2
- **blood_pressure_2**: BP measured at time 2

- **blood_pressure_medication_2**: Whether the person took treatment at time period 2

For the "SuperLearner" and "TMLE" portions, you can ignore any variable that ends in "_2", we will reintroduce these for LTMLE.

SuperLearner

Modeling

Fit a SuperLearner model to estimate the probability of someone dying from complications of heart disease, conditional on treatment and the relevant covariates. Do the following:

1. Choose a library of at least 5 machine learning algorithms to evaluate. **Note**: We did not cover how to hyperparameter tune constituent algorithms within SuperLearner in lab, but you are free to do so if you like (though not required to for this exercise).
2. Split your data into train and test sets.
3. Train SuperLearner
4. Report the risk and coefficient associated with each model, and the performance of the discrete winner and SuperLearner ensemble
5. Create a confusion matrix and report your overall accuracy, recall, and precision

```
# Fit SuperLearner Model

## Train/Test split

# initial_split function from tidymodels/rsample
heart_split <- initial_split(heart_disease, prop = 3/4)

# Declare the training set with rsample::training()
train <- training(heart_split)
# y_train is medv where medv > 22 is a 1, 0 otherwise
y_train <- train %>%
  pull(mortality)

# x_train is everything but the outcome
x_train <- train %>%
  select(-mortality, -bmi_2, -blood_pressure_2,
         -chol_2, -blood_pressure_medication_2)

# Do the same procedure with the test set
test <- testing(heart_split)

y_test <- test %>%
  pull(mortality)

x_test <- test %>%
  select(-mortality, -bmi_2, -blood_pressure_2,
         -chol_2, -blood_pressure_medication_2)
```

```
## Train SuperLearner

# sl = SuperLearner(Y = y_train, X = x_train,
#                   family = binomial(),
#                   SL.library = c("SL.mean",
#                                  "SL.lm",
#                                  "SL.glmnet",
#                                  "SL.ranger",
#                                  "SL.lda",
#                                  "SL.step"))
# saveRDS(sl, file = "sl.Rds")

## Risk and Coefficient of each model

sl <- readRDS("sl.Rds")
```

```
## Discrete winner and superlearner ensemble performance

preds <- predict(sl,
                 x_test,
                 onlySL = TRUE)

validation <- y_test %>%
  bind_cols(preds$pred[,1]) %>%
  rename(obs = `...1`,
         pred = `...2`) %>%
  mutate(pred = ifelse(pred >= .5,
                           1,
                           0))
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
```

```
sl
```

```
##
## Call:
## SuperLearner(Y = y_train, X = x_train, family = binomial(), SL.library = c("SL.mean",
##     "SL.lm", "SL.glmnet", "SL.ranger", "SL.lda", "SL.step"))
##
##
##                     Risk         Coef
## SL.mean_All    0.2498666 0.0003672181
## SL.lm_All      0.2358022 0.3354443561
## SL.glmnet_All  0.2361367 0.0000000000
## SL.ranger_All  0.2306050 0.6641884258
## SL.lda_All     0.2362840 0.0000000000
## SL.step_All    0.2363049 0.0000000000
```

These models contributed to the weighted average model:

1) For the model using the mean of Y, the risk was 0.249866 and the coefficient was 0.0003672181.

2) For the model using a linear model, the risk was 0.2358022 and the coefficient was 0.3354443561.

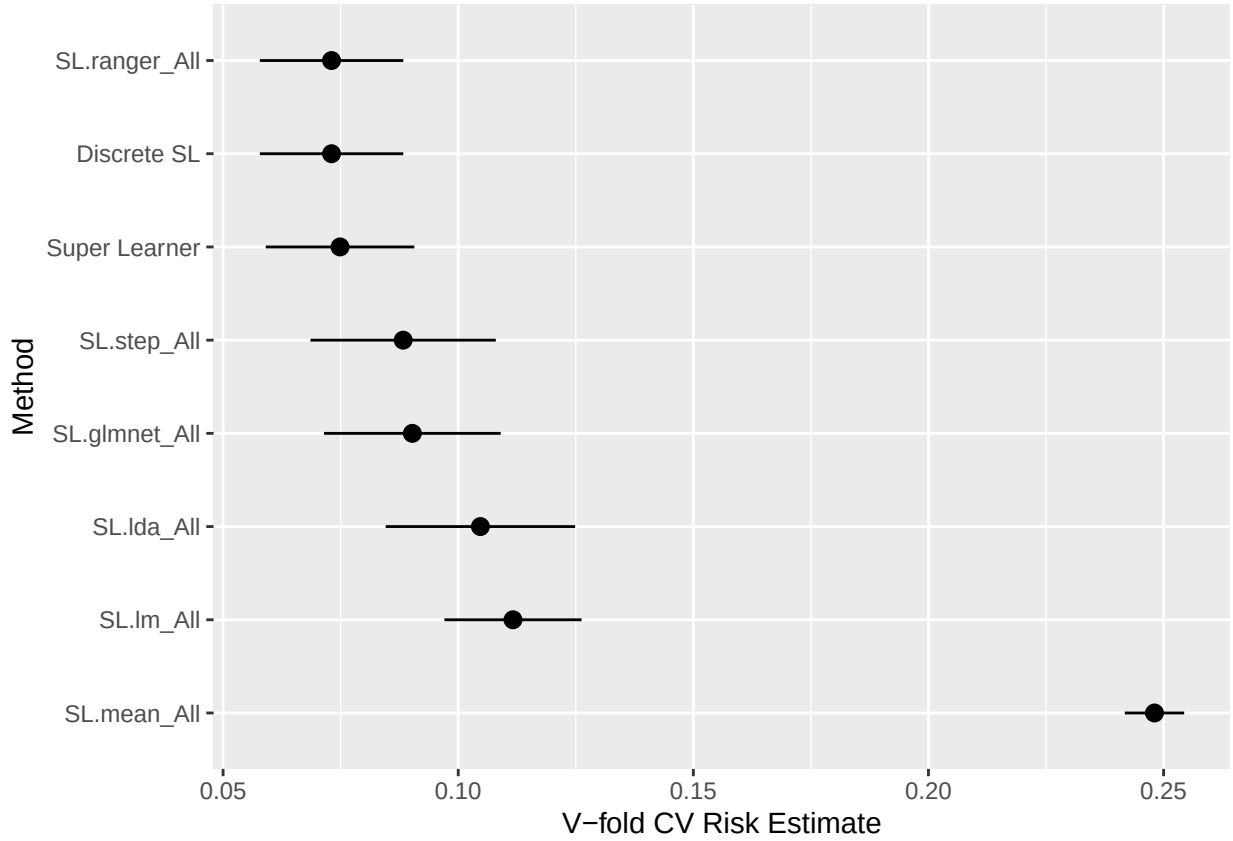
3) For the model using random forest, the risk was 0.2306050 and the coefficient was 0.6641884258.

These models did not contribute to the weighted average model: 4) For the model using a generalized linear model with elastic net regression, the risk was 0.2361367 and the coefficient was 0.

5) For the model using linear discriminant analysis, the risk was 0.2362840 and the coefficient was 0.

6) For the model using stepwise, backward and forward directions, the risk was 0.2363049 and the coefficient was 0.

```
# cluster = parallel::makeCluster(availableCores() - 1)
# # Load SuperLearner onto all clusters
# parallel::clusterEvalQ(cluster, library(SuperLearner))
# parallel::clusterSetRNGStream(cluster, 1)
# cv_sl = CV.SuperLearner(Y = y_train, X = x_train, family = binomial(),
#                         # For a real analysis we would use V = 10.
#                         V = 10,
#                         parallel = cluster,
#                         SL.library = c("SL.mean",
#                                        "SL.lm",
#                                       "SL.glmnet",
#                                       "SL.ranger",
#                                       "SL.lda",
#                                       "SL.step"))
# parallel::stopCluster(cluster)
#
# saveRDS(cv_sl, file = "cv_sl.Rds")
cv_sl <- readRDS("cv_sl.Rds")
plot(cv_sl)
```



The discrete winner was the model using random forest. The Super Learner peformed very similarly to the discrete winner.

```
## Confusion Matrix

confusion_matrix <- caret::confusionMatrix(as.factor(validation$pred),
                      as.factor(validation$obs))
confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1058   66
##          1  188 1188
##
##                Accuracy : 0.8984
##                  95% CI : (0.8859, 0.91)
##     No Information Rate : 0.5016
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7967
##
##  Mcnemar's Test P-Value : 3.145e-14
##
##             Sensitivity : 0.8491
```

```
##             Specificity : 0.9474
##          Pos Pred Value : 0.9413
##          Neg Pred Value : 0.8634
##              Prevalence : 0.4984
##          Detection Rate : 0.4232
##    Detection Prevalence : 0.4496
##       Balanced Accuracy : 0.8982
##
##        'Positive' Class : 0
##
```

```
TP <- confusion_matrix$table[2,2]
TN <- confusion_matrix$table[1,1]
FP <- confusion_matrix$table[2,1]
FN <- confusion_matrix$table[1,2]

#accuracy
accuracy <- (TP + TN) / (TP + TN + FP + FN)
accuracy
```

```
## [1] 0.8984
```

```
#recall
recall <- TP / (TP + FN)
recall
```

```
## [1] 0.9473684
```

```
# precision
precision <- TP / (TP + FP)
precision
```

```
## [1] 0.8633721
```

For the ensemble model, the accuracy was 0.8984, recall was 0.9473684, and precision was 0.8633721.

Discussion Questions

1. Why should we, in general, prefer the SuperLearner ensemble to the discrete winner in cross-validation? Or in other words, what is the advantage of "blending" algorithms together and giving them each weights, rather than just using the single best algorithm (with best being defined as minimizing risk)?

We should prefer the Super Learner ensemble over the discrete winner because the "meta-model" that represents the Super Learner model uses the out-of-sample predictions from all of the models included as its input, which does two things at the same time: 1) "corrects" for the out-of-sample predictions for each model and 2) "best combines" the predictions. The "better" out of sample predictions does not exist when you only use the single best algorithm.

Targeted Maximum Likelihood Estimation

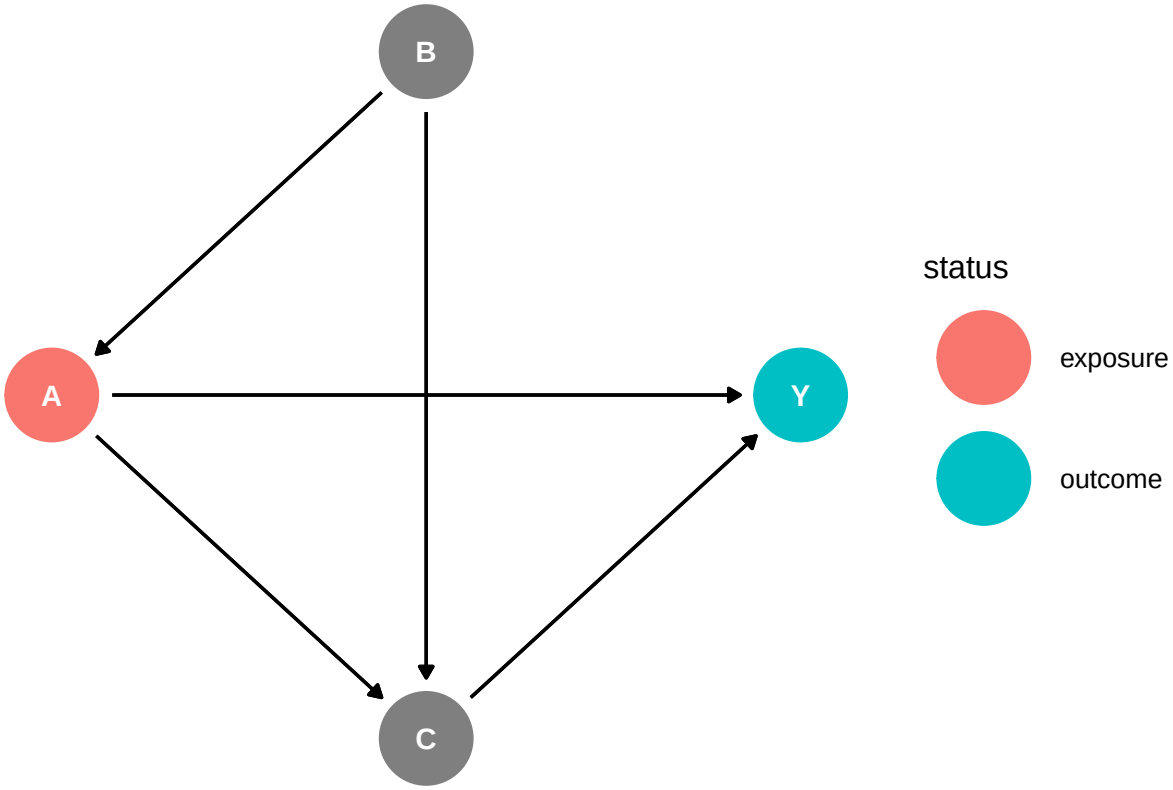
Causal Diagram

TMLE requires estimating two models:

1. The outcome model, or the relationship between the outcome and the treatment/predictors, $P(Y|(A, W)$.
2. The propensity score model, or the relationship between assignment to treatment and predictors $P(A|W)$

Using ggdag and daggity, draw a directed acylcic graph (DAG) that describes the relationships between the outcome, treatment, and covariates/predictors. Note, if you think there are covariates that are not related to other variables in the dataset, note this by either including them as freestanding nodes or by omitting them and noting omissions in your discussion.

```
# DAG for TMLE
dagify(Y ~ A + C,
       C ~ A + B,
       A ~ B,
       exposure = "A",
       outcome = "Y",
       coords = list(x = c(A = 1, C = 2, B = 2, Y = 3),
                     y = c(A = 2, C = 1, B = 3, Y = 2))) %>%
  ggdag_status() +
  geom_dag_edges() +
  theme_dag()
```



A: blood pressure medication

B: age, sex at birth, simplified race, income in thousands, and college education

C: BMI, cholesterol level, and blood pressure

Y: mortality

TMLE Estimation

Use the `tmle` package to estimate a model for the effect of blood pressure medication on the probability of mortality. Do the following:

1. Use the same SuperLearner library you defined earlier
2. Use the same outcome model and propensity score model that you specified in the DAG above. If in your DAG you concluded that it is not possible to make a causal inference from this dataset, specify a simpler model and note your assumptions for this step.
3. Report the average treatment effect and any other relevant statistics

```
sl_libs <- c("SL.mean","SL.lm","SL.glmnet","SL.ranger","SL.step")
# Outcome
Y <- heart_disease %>% pull(mortality)
A <- heart_disease %>%
  select(-mortality, -bmi_2, -blood_pressure_2,
         -chol_2, -blood_pressure_medication_2) %>%
```

```
  pull(blood_pressure_medication)
W <- heart_disease %>%
  select(age, sex_at_birth, simplified_race,
         college_educ, income_thousands, bmi,
         blood_pressure, chol)

# tmle_fit <-
#   tmle::tmle(Y = Y,
#              A = A,
#              W = W,
#              Q.SL.library = sl_libs,
#              g.SL.library = sl_libs)

# saveRDS(tmle_fit, file = "tmle_fit.Rds")
tmle_fit <- readRDS("tmle_fit.Rds")

tmle_fit
```

```
##  Additive Effect
##    Parameter Estimate:  -0.35419
##    Estimated Variance:  6.9828e-05
##               p-value:  <2e-16
##     95% Conf Interval: (-0.37057, -0.33781)
##
##  Additive Effect among the Treated
##    Parameter Estimate:  -0.32007
##    Estimated Variance:  0.00014708
##               p-value:  <2e-16
##     95% Conf Interval: (-0.34384, -0.2963)
##
##  Additive Effect among the Controls
##    Parameter Estimate:  -0.36834
##    Estimated Variance:  6.2642e-05
##               p-value:  <2e-16
##     95% Conf Interval: (-0.38385, -0.35282)
```

The average treatment effect is -0.35419, the average treatment effect among the treated is -0.32007, and the average treatment effect among the control group is -0.36834.

Discussion Questions

1. What is a "double robust" estimator? Why does it provide a guarantee of consistency if either the outcome model or propensity score model is correctly specified? Or in other words, why does mispecifying one of the models not break the analysis? **Hint**: When answering this question, think about how your introductory statistics courses emphasized using theory to determine the correct outcome model, and in this course how we explored the benefits of matching.

It is an estimator that is used to estimate the average treatment effect when not being sure whether to use the outcome model of the propensity score model estimator. If one is correct, by definition the ATE will be correct because the double robust estimator ignores the relevance of the other estimator since its mean value will equal 0.

LTMLE Estimation

Now imagine that everything you measured up until now was in "time period 1". Some people either choose not to or otherwise lack access to medication in that time period, but do start taking the medication in time period 2. Imagine we measure covariates like BMI, blood pressure, and cholesterol at that time for everyone in the study (indicated by a "_2" after the covariate name).

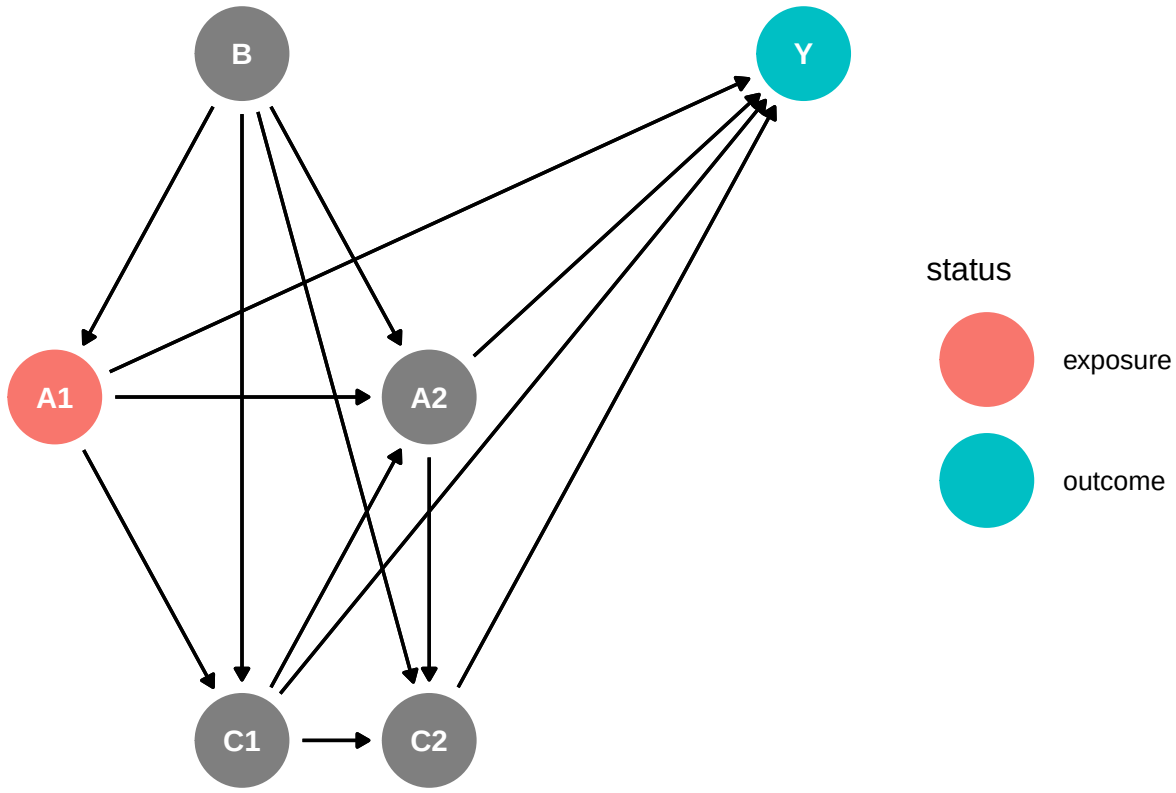
Causal Diagram

Update your causal diagram to incorporate this new information. **Note**: If your groups divides up sections and someone is working on LTMLE separately from TMLE then just draw a causal diagram even if it does not match the one you specified above.

Hint: Check out slide 27 from Maya's lecture, or slides 15-17 from Dave's second slide deck in week 8 on matching.

Hint: Keep in mind that any of the variables that end in "_2" are likely affected by both the previous covariates and the first treatment when drawing your DAG.

```
# DAG for TMLE
dagify(Y ~ A1 + A2 + C1 + C2,
       A2 ~ B + A1 + C1,
       C2 ~ B + C1 + A2,
       C1 ~ A1 + B,
       A1 ~ B,
       exposure = "A1",
       outcome = "Y",
       coords = list(x = c(A1 = 1, C1 = 2, B = 2, A2 = 3, C2= 3, Y = 5),
                     y = c(A1 = 2, C1 = 1, B = 3, A2 = 2, C2= 1, Y = 3))) %>%
  ggdag_status() +
  geom_dag_edges() +
  theme_dag()
```



A1: blood pressure medication; time=1

B: age, sex at birth, simplified race, income in thousands, and college education

C1: BMI, cholesterol level, and blood pressure; time =1

A2: blood pressure medication; time=2

C2: BMI, cholesterol level, and blood pressure; time =2

Y: mortality

LTMLE Estimation

Use the `ltmle` package for this section. First fit a "naive model" that **does not** control for the time-dependent confounding. Then run a LTMLE model that does control for any time dependent confounding. Follow the same steps as in the TMLE section. Do you see a difference between the two estimates?

```
## Naive Model (no time-dependent confounding) estimate
W1 <- heart_disease$age
W2 <- heart_disease$sex_at_birth
W3 <- heart_disease$simplified_race
W4 <- heart_disease$college_educ
W5 <- heart_disease$income_thousands
A1 <- heart_disease$blood_pressure_medication
A2 <- heart_disease$blood_pressure_medication_2
L1a <- heart_disease$bmi
L1b <- heart_disease$bmi_2
```

```r
L2a <- heart_disease$blood_pressure
L2b <- heart_disease$blood_pressure_2
L3a <- heart_disease$chol
L3b <- heart_disease$chol_2
Y <- heart_disease$mortality
data1 <- data.frame(W1, W2, W3, W4, W5, A1, A2, Y)
data2 <- data.frame(W1, W2, W3, W4, W5, A1, L1a, L1b, L2a, L2b, L3a, L3b, A2, Y)

result1 <- ltmle(data1, Anodes = c("A1","A2"),
      Lnodes = NULL, Ynodes = "Y",
      abar = c(1, 1))
```

```
## Qform not specified, using defaults:

## formula for Y:

## Q.kplus1 ~ W1 + W2 + W3 + W4 + W5 + A1 + A2

##

## gform not specified, using defaults:

## formula for A1:

## A1 ~ W1 + W2 + W3 + W4 + W5

## formula for A2:

## A2 ~ W1 + W2 + W3 + W4 + W5 + A1

##

## Estimate of time to completion: < 1 minute
```

```r
## LTMLE estimate
result2 <- ltmle(data2, Anodes = c("A1","A2"),
      Lnodes = c("L1a", "L1b", "L2a", "L2b", "L3a", "L3b"), Ynodes = "Y",
      abar = c(1, 1))
```

```
## Qform not specified, using defaults:

## formula for L1a:

## Q.kplus1 ~ W1 + W2 + W3 + W4 + W5 + A1

## formula for Y:

## Q.kplus1 ~ W1 + W2 + W3 + W4 + W5 + A1 + L1a + L1b + L2a + L2b +     L3a + L3b + A2
```

```
##

## gform not specified, using defaults:

## formula for A1:

## A1 ~ W1 + W2 + W3 + W4 + W5

## formula for A2:

## A2 ~ W1 + W2 + W3 + W4 + W5 + A1 + L1a + L1b + L2a + L2b + L3a +     L3b

##

## Estimate of time to completion: < 1 minute
```

```
dif_val <- result1$estimates[1] - result2$estimates[1]
names(dif_val) <- "difference in TMLE estimate"
dif_val
```

```
## difference in TMLE estimate
##                  0.03069835
```

There is a slight difference of 0.0306983, where the estimate with control for time dependent confounding having the smaller value.

Discussion Questions

1. What sorts of time-dependent confounding should we be especially worried about? For instance, would we be concerned about a running variable for age the same way we might be concerned about blood pressure measured at two different times?

We should especially be worried about time-dependent confounding variables that are very sensitive to latent conditions. It is really hard to say that age can be influenced by something you did before being surveyed, but for blood pressure, there is a chance that several latent variables could influence the result. This is why we see multiple readings of the same variable to account for external changes.