Project 8 Template

Takun Wang

2024-05-09

1 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.

```
## Load packages
library(tidyverse)
library(ggdag)
library(dagitty)
library(SuperLearner)
library(tidymodels)
library(caret)
library(tmle)
library(ltmle)
```

2 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.

```
## Read data
heart <- read_csv("data/heart_disease_tmle.csv")
glimpse(heart)
```

```
## Rows: 10,000
## Columns: 14
## $ age                          <dbl> 32.92951, 53.92344, 65.33631, 16.82682, 56~
## $ sex_at_birth                 <dbl> 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, ~
## $ simplified_race              <dbl> 1, 1, 3, 1, 1, 1, 1, 1, 3, 3, 3, 2, 1, 1, ~
## $ college_educ                 <dbl> 2, 2, 2, 2, 2, 2, 1, 1, 2, 1, 1, 2, 2, 2, ~
## $ income_thousands             <dbl> 91.32660, 38.76890, 35.48435, 93.77726, 85~
## $ bmi                          <dbl> 27.09986, 27.61615, 27.52499, 24.88569, 22~
## $ blood_pressure               <dbl> 146.9862, 125.7578, 131.6789, 131.2926, 13~
## $ chol                         <dbl> 211.4630, 187.8418, 197.0794, 229.6352, 20~
## $ blood_pressure_medication    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, ~
## $ bmi_2                        <dbl> 27.08979, 27.61077, 27.52064, 24.89534, 22~
## $ blood_pressure_2             <dbl> 146.9800, 125.7505, 131.6626, 131.2765, 13~
## $ chol_2                       <dbl> 211.5343, 187.7360, 196.9674, 229.6558, 20~
## $ blood_pressure_medication_2 <dbl> 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, ~
## $ mortality                    <dbl> 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, ~
```

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

3 SuperLearner

3.1 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).

```
## Algorithms available
listWrappers()
```

```
## All prediction algorithm wrappers in SuperLearner:

##  [1] "SL.bartMachine"      "SL.bayesglm"         "SL.biglasso"
##  [4] "SL.caret"            "SL.caret.rpart"      "SL.cforest"
##  [7] "SL.earth"            "SL.gam"              "SL.gbm"
## [10] "SL.glm"              "SL.glm.interaction"  "SL.glmnet"
## [13] "SL.ipredbagg"        "SL.kernelKnn"        "SL.knn"
```

```
## [16] "SL.ksvm"              "SL.lda"              "SL.leekasso"
## [19] "SL.lm"                "SL.loess"            "SL.logreg"
## [22] "SL.mean"              "SL.nnet"             "SL.nnls"
## [25] "SL.polymars"          "SL.qda"              "SL.randomForest"
## [28] "SL.ranger"            "SL.ridge"            "SL.rpart"
## [31] "SL.rpartPrune"        "SL.speedglm"         "SL.speedlm"
## [34] "SL.step"              "SL.step.forward"     "SL.step.interaction"
## [37] "SL.stepAIC"           "SL.svm"              "SL.template"
## [40] "SL.xgboost"

##
## All screening algorithm wrappers in SuperLearner:

## [1] "All"
## [1] "screen.corP"           "screen.corRank"        "screen.glmnet"
## [4] "screen.randomForest"   "screen.SIS"            "screen.template"
## [7] "screen.ttest"          "write.screen.template"
```

```
## Choose 5 algorithms
SL_libs <- c("SL.mean", "SL.glmnet", "SL.gbm", "SL.ranger")
```

2. Split your data into train and test sets.

```
## Exclude time 2 variables
heart1 <- heart %>% select(-bmi_2, -blood_pressure_2, -chol_2, -blood_pressure_medication_2)

## Split
set.seed(224)
heart_split <- initial_split(heart1, prop = 3/4)

## Training set
y_train <- training(heart_split) %>% pull(mortality)
X_train <- training(heart_split) %>% select(-mortality)

## Testing set
y_test <- testing(heart_split) %>% pull(mortality)
X_test <- testing(heart_split) %>% select(-mortality)
```

3. Train SuperLearner.

```
## Train SL
set.seed(224)
SL <- SuperLearner(Y = y_train, X = X_train,
                   family = binomial(),
                   SL.library = SL_libs)
```

```
## Loading required namespace: gbm

## Loading required namespace: ranger
```

4. Report the risk and coefficient associated with each model, and the performance of the discrete winner and SuperLearner ensemble.

```
## Ensemble
SL
```

```
##
## Call:
## SuperLearner(Y = y_train, X = X_train, family = binomial(), SL.library = SL_libs)
##
##
##
##                     Risk      Coef
## SL.mean_All    0.2497836 0.0000000
## SL.glmnet_All 0.2362958 0.0000000
## SL.gbm_All     0.2291723 0.5993727
## SL.ranger_All 0.2307990 0.4006273
```

```
## Discrete winner
SL$cvRisk[which.min(SL$cvRisk)]
```

```
## SL.gbm_All
##  0.2291723
```

```
## Validation
preds <- predict(SL, X_test)

validation <- data.frame(y_test) %>%
  mutate(ensemble = as.vector(preds$pred),
         winner = preds$library.predict[,"SL.gbm_All"]) %>%
  mutate(ensemble_pred = ifelse(ensemble >= .5, 1, 0),
         winner_pred = ifelse(winner >= .5, 1, 0))
head(validation)
```

y_test	ensemble	winner	ensemble_pred	winner_pred
0	0.5319619	0.5695178	1	1
1	0.6382561	0.5835638	1	1
0	0.5884340	0.5707626	1	1
1	0.5444448	0.5758299	1	1
1	0.5666603	0.5532981	1	1
1	0.5576942	0.5760695	1	1

5. Create a confusion matrix and report your overall accuracy, recall, and precision.

```
## Ensemble
confusionMatrix(as.factor(validation$ensemble_pred),
                as.factor(validation$y_test))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  336  143
```

```
##          1  873 1148
##
##                Accuracy : 0.5936
##                  95% CI : (0.574, 0.6129)
##     No Information Rate : 0.5164
##     P-Value [Acc > NIR] : 5.283e-15
##
##                   Kappa : 0.1704
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.2779
##             Specificity : 0.8892
##          Pos Pred Value : 0.7015
##          Neg Pred Value : 0.5680
##              Prevalence : 0.4836
##          Detection Rate : 0.1344
##    Detection Prevalence : 0.1916
##       Balanced Accuracy : 0.5836
##
##        'Positive' Class : 0
##
```

```
## Discrete winner
confusionMatrix(as.factor(validation$winner_pred),
                as.factor(validation$y_test))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  279   88
##          1  930 1203
##
##                Accuracy : 0.5928
##                  95% CI : (0.5732, 0.6121)
##     No Information Rate : 0.5164
##     P-Value [Acc > NIR] : 9.936e-15
##
##                   Kappa : 0.1663
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.2308
##             Specificity : 0.9318
##          Pos Pred Value : 0.7602
##          Neg Pred Value : 0.5640
##              Prevalence : 0.4836
##          Detection Rate : 0.1116
##    Detection Prevalence : 0.1468
##       Balanced Accuracy : 0.5813
##
##        'Positive' Class : 0
##
```

3.2 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)?
    - Different algorithms have different strengths and weaknesses, and by blending them together, one can create a more robust model that performs well across a wider range of datasets and scenarios. This can help mitigate the risk of relying too heavily on a single algorithm that might perform poorly under certain conditions.

4 Targeted Maximum Likelihood Estimation

4.1 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 $P(A|W)$.

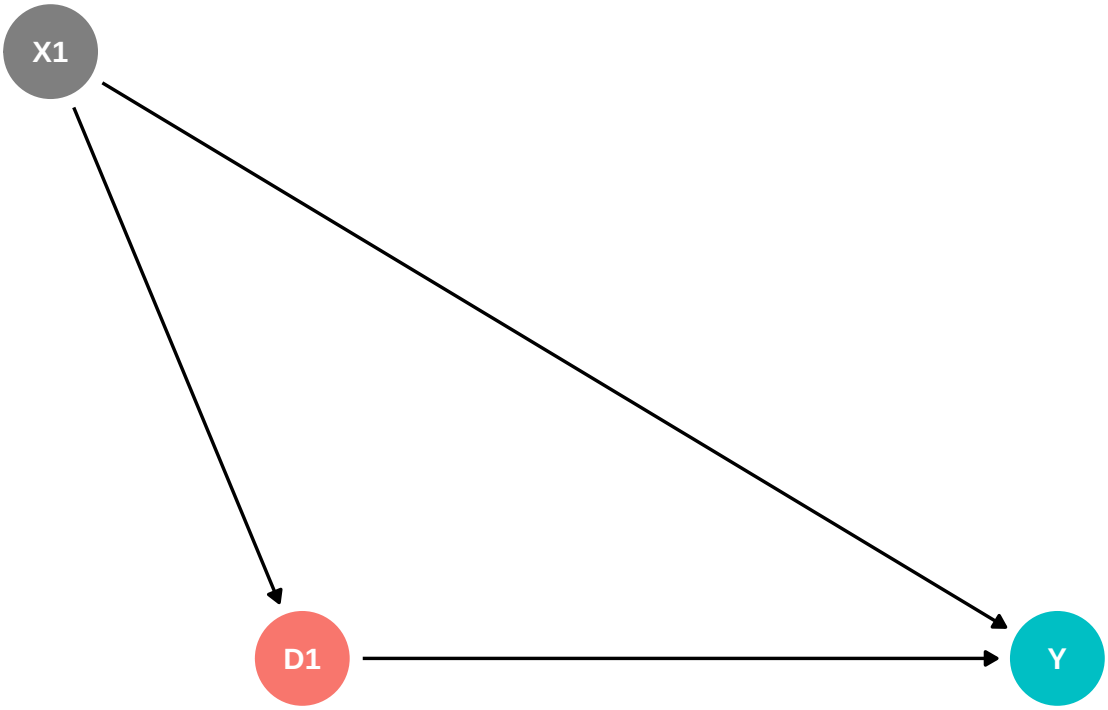
Using ggdag and daggity, draw a directed acylcic graph (DAG) that describes the relationships between the outcome, treatment, and covariates/predictors. Note, if you think there are covariates that are not related to other variables in the dataset, note this by either including them as freestanding nodes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

- Outcome (Y): mortality
- Treatment at time 1 (D1): medication
- Covariates at time 1 (X1): age, sex, race, income, college, BMI, cholesterol level, blood pressure

```
## TMLE DAG
coords <- list(
  x = c(X1 = 0, D1 = 0.5, Y = 2),
  y = c(X1 = 0.5, D1 = 0, Y = 0))

simple_dag <- dagify(
  Y ~ X1 + D1,
  D1 ~ X1,
  exposure = "D1",
  outcome = "Y",
  coords = coords)

ggdag_status(simple_dag) +
  theme_dag(legend.position = "none")
```



4.2 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.

1. Estimate the initial outcomes.

```
## SL library
SL_libs <- c("SL.glmnet")

## Preparation
y <- heart1[["mortality"]]
X <- heart1 %>% select(-mortality)

## Fit
set.seed(224)
SL_Q <- SuperLearner(Y = y,
                     X = X,
                     family = binomial(),
                     SL.library = SL_libs)

## Predictions
data <- tibble(
```

```
  Y = y,
  D = X[["blood_pressure_medication"]],
  Q_D = SL_Q$SL.predict %>% as.vector(),
  Q_0 = X %>% mutate(blood_pressure_medication = 0) %>%
    predict(SL_Q, .) %>% pluck(1) %>% as.vector(),
  Q_1 = X %>% mutate(blood_pressure_medication = 1) %>%
    predict(SL_Q, .) %>% pluck(1) %>% as.vector())

data %>% round(3) %>% head()
```

Y	D	Q_D	Q_0	Q_1
1	0	0.589	0.589	0.263
0	0	0.546	0.546	0.230
1	0	0.571	0.571	0.249
0	0	0.588	0.588	0.263
0	0	0.559	0.559	0.240
1	0	0.550	0.550	0.234

2. Estimate the probability of treatment

```
## Fit
set.seed(224)
SL_g <- SuperLearner(Y = X[["blood_pressure_medication"]],
                     X = X %>% select(-blood_pressure_medication),
                     family = binomial(),
                     SL.library = SL_libs)

## Predictions
data <- data %>%
  mutate(g_w = SL_g$SL.predict %>% as.vector(),
         H_1 = 1 / g_w,
         H_0 = -1 / (1 - g_w),
         H_D = case_when(
           D == 1 ~ H_1,
           D == 0 ~ H_0))

data %>% round(3) %>% head()
```

Y	D	Q_D	Q_0	Q_1	g_w	H_1	H_0	H_D
1	0	0.589	0.589	0.263	0.166	6.031	-1.199	-1.199
0	0	0.546	0.546	0.230	0.177	5.661	-1.215	-1.215
1	0	0.571	0.571	0.249	0.175	5.723	-1.212	-1.212
0	0	0.588	0.588	0.263	0.125	8.000	-1.143	-1.143
0	0	0.559	0.559	0.240	0.094	10.626	-1.104	-1.104
1	0	0.550	0.550	0.234	0.112	8.942	-1.126	-1.126

3. Extract the fluctuation parameter.

```r
## Fluctuation parameter
glm <- glm(Y ~ -1 + offset(qlogis(Q_D)) + H_D,
           data = data,
           family = binomial)

eps <- coef(glm)
eps
```

```
##         H_D
## -0.02222647
```

4. Update the initial estimate.

```r
## Update
data <- data %>% mutate(
  Q_D_up = plogis(qlogis(Q_D)) + eps * H_D,
  Q_1_up = plogis(qlogis(Q_1)) + eps * H_1,
  Q_0_up = plogis(qlogis(Q_0)) + eps * H_0)

data %>% round(3) %>% head()
```

Y	D	Q_D	Q_0	Q_1	g_w	H_1	H_0	H_D	Q_D_up	Q_1_up	Q_0_up
1	0	0.589	0.589	0.263	0.166	6.031	-1.199	-1.199	0.615	0.129	0.615
0	0	0.546	0.546	0.230	0.177	5.661	-1.215	-1.215	0.573	0.105	0.573
1	0	0.571	0.571	0.249	0.175	5.723	-1.212	-1.212	0.598	0.122	0.598
0	0	0.588	0.588	0.263	0.125	8.000	-1.143	-1.143	0.614	0.085	0.614
0	0	0.559	0.559	0.240	0.094	10.626	-1.104	-1.104	0.583	0.004	0.583
1	0	0.550	0.550	0.234	0.112	8.942	-1.126	-1.126	0.575	0.035	0.575

5. Compute ATE.

```r
## ATE
ATE <- data %>% mutate(diff = Q_1_up - Q_0_up) %>% summarise(ATE = mean(diff)) %>% pull(ATE)
ATE
```

```
## [1] -0.515392
```

```r
## ATE_gcomp (for comparison)
data %>% mutate(diff = Q_1 - Q_0) %>% summarise(ATE_gcomp = mean(diff)) %>% pull(ATE_gcomp)
```

```
## [1] -0.3193157
```

6. Calculate standard error.

```r
## SE
infl_fn_var <- data %>% mutate(infl_fn = (Y - Q_D_up) * H_D + Q_1_up - Q_0_up - ATE) %>%
  pull(infl_fn) %>% var()

SE <- sqrt(infl_fn_var / nrow(data))
SE
```

```
## [1] 0.01363867
```

```
## Confidence interval
c(ATE - 1.96 * SE, ATE + 1.96 * SE)
```

```
## [1] -0.5421238 -0.4886602
```

```
## p-value
2 * (1 - pnorm(abs(ATE / SE)))
```

```
## [1] 0
```

7. Use TMLE pacakage to run previous steps.

```
## TMLE method
set.seed(224)

tmle_fit <- tmle(Y = y,
                 A = X[["blood_pressure_medication"]],
                 W = X %>% select(-blood_pressure_medication),
                 Q.SL.library = SL_libs,
                 g.SL.library = SL_libs)
tmle_fit
```

```
##  Marginal mean under treatment (EY1)
##    Parameter Estimate:  0.22252
##    Estimated Variance:  0.00010244
##               p-value:  <2e-16
##     95% Conf Interval:  (0.20269, 0.24236)
##
##  Marginal mean under comparator (EY0)
##    Parameter Estimate:  0.56708
##    Estimated Variance:  2.9121e-05
##               p-value:  <2e-16
##     95% Conf Interval:  (0.55651, 0.57766)
##
##  Additive Effect
##    Parameter Estimate:  -0.34456
##    Estimated Variance:  0.00013037
##               p-value:  <2e-16
##     95% Conf Interval:  (-0.36694, -0.32218)
##
##  Additive Effect among the Treated
##    Parameter Estimate:  -0.32421
##    Estimated Variance:  0.00014301
##               p-value:  <2e-16
##     95% Conf Interval:  (-0.34764, -0.30077)
##
##  Additive Effect among the Controls
##    Parameter Estimate:  -0.34813
##    Estimated Variance:  0.00012851
##               p-value:  <2e-16
##     95% Conf Interval:  (-0.37035, -0.32591)
```

4.3 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.

    - A "double robust" estimator is a statistical estimator commonly used in causal inference when estimating treatment effects in observational studies. It is called "double robust" because it provides consistent estimates of the treatment effect even if either the outcome model or the propensity score model is misspecified, as long as one of them is correctly specified.

5 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).

5.1 Causal Diagram

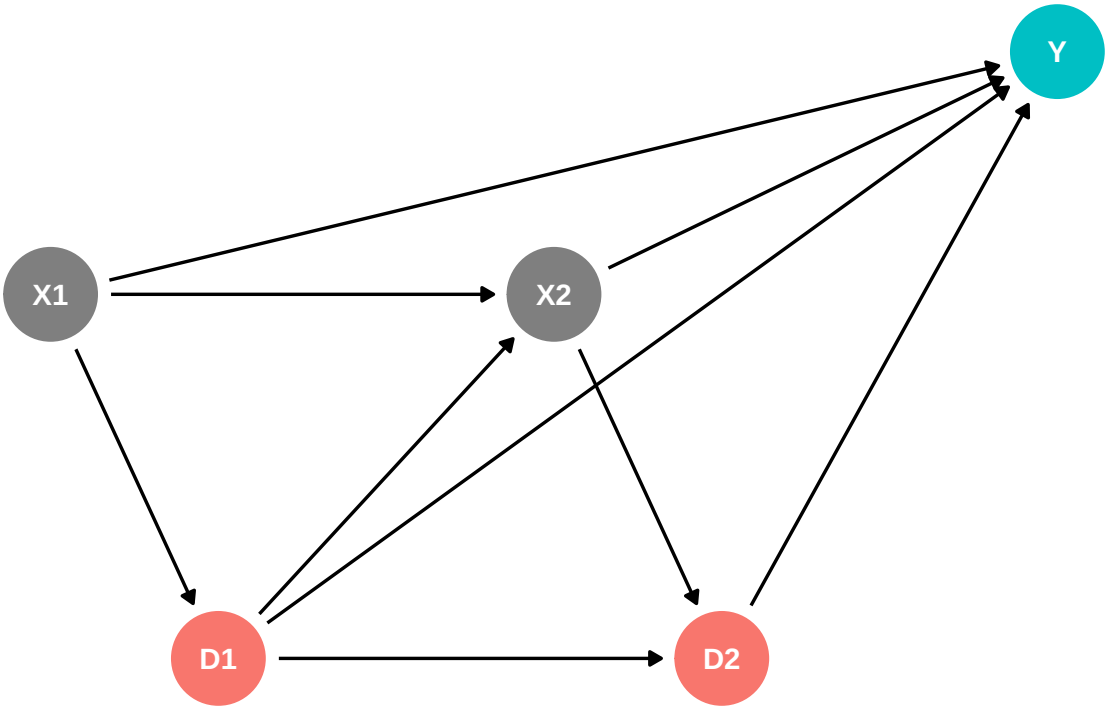
Update your causal diagram to incorporate this new information. **Note**: Keep in mind that any of the variables that end in "_2" are likely affected by both the previous covariates and the first treatment when drawing your DAG.

- Outcome (Y): mortality
- Treatment at time 1 (D1): medication
- Covariates at time 1 (X1): age, sex, race, income, college, BMI, cholesterol level, blood pressure
- Treatment at time 2 (D2): medication 2
- Covariates at time 1 (X1): BMI 2, cholesterol level 2, blood pressure 2

```
## DAG T2
coords <- list(
  x = c(X1 = 0,   D1 = 0.5, X2 = 1.5, D2 = 2, Y = 3),
  y = c(X1 = 0.6, D1 = 0,   X2 = 0.6, D2 = 0, Y = 1))

simple_dag <- dagify(
  Y ~ X1 + D1 + X2 + D2,
  D2 ~ D1 + X2,
  X2 ~ X1 + D1,
  D1 ~ X1,
  exposure = c("D1", "D2"),
  outcome = "Y",
  coords = coords)

ggdag_status(simple_dag) +
  theme_dag(legend.position = "none")
```



5.2 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
naive <- ltmle(heart,
               Anodes = "blood_pressure_medication_2",
               Ynodes = "mortality",
               abar = 1)
```

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

```
## formula for mortality:
```

```
## Q.kplus1 ~ age + sex_at_birth + simplified_race + college_educ +     income_thousands + bmi + blood_
```

```
##
```

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

```
## formula for blood_pressure_medication_2:
```

```
## blood_pressure_medication_2 ~ age + sex_at_birth + simplified_race +     college_educ + income_thousa
```

```
##

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

```
naive
```

```
## Call:
## ltmle(data = heart, Anodes = "blood_pressure_medication_2", Ynodes = "mortality",
##     abar = 1)
##
## TMLE Estimate:  0.5094526
```

```
## LTMLE estimate
heart2 <- heart %>%
  select(age, sex_at_birth, simplified_race, income_thousands, college_educ, bmi, chol, blood_pressure,
         blood_pressure_medication,
         bmi_2, chol_2, blood_pressure_2,
         blood_pressure_medication_2,
         mortality)

ltmle <- ltmle(heart2,
               Anodes = c("blood_pressure_medication", "blood_pressure_medication_2"),
               Lnodes = c("bmi_2", "chol_2", "blood_pressure_2"),
               Ynodes = "mortality",
               abar = c(1, 1))
```

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

## formula for bmi_2:

## Q.kplus1 ~ age + sex_at_birth + simplified_race + income_thousands +     college_educ + bmi + chol +

## formula for mortality:

## Q.kplus1 ~ age + sex_at_birth + simplified_race + income_thousands +     college_educ + bmi + chol +

##

## gform not specified, using defaults:

## formula for blood_pressure_medication:

## blood_pressure_medication ~ age + sex_at_birth + simplified_race +     income_thousands + college_edu

## formula for blood_pressure_medication_2:

## blood_pressure_medication_2 ~ age + sex_at_birth + simplified_race +     income_thousands + college_

##

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

```
ltmle
```

```
## Call:
## ltmle(data = heart2, Anodes = c("blood_pressure_medication",
##     "blood_pressure_medication_2"), Lnodes = c("bmi_2", "chol_2",
##     "blood_pressure_2"), Ynodes = "mortality", abar = c(1, 1))
##
## TMLE Estimate:  0.1886269
```

5.3 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?

    - Time-dependent confounding occurs when the relationship between the treatment, outcome, and confounding variables changes over time. This type of confounding can be particularly challenging to address because it involves considering how variables change over time and how those changes may affect the causal inference process. Blood pressure measured at two different times poses concerns because it represents a time-varying variable that may be both a confounder and a potential mediator of the treatment effect. In contrast, age is a concern only if the effect of the treatment varies with age or if age influences the likelihood of receiving the treatment.