Project 8 Template

```
# Add to this package list for additional SL algorithms
pacman::p_load(
  tidyverse,
  ggthemes,
  ltmle,
  tmle,
  SuperLearner,
  tidymodels,
  caret,
  dagitty,
  ggdag,
  here)

heart_disease <- read_csv('heart_disease_tmle.csv')
```

```
## Rows: 10000 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (14): age, sex_at_birth, simplified_race, college_educ, income_thousands...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

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


## sl lib
models <- c("SL.glmnet",
            "SL.mean",
            "SL.knn",
            "SL.lda",
            "SL.ranger"
            )

## Train/Test split
heart_disease_split <- initial_split(heart_disease, prop=.8)
train <- training(heart_disease_split)
y_train <- train %>% pull(mortality)
x_train <- train %>% select(-mortality) %>% select(-ends_with("_2"))
test <- testing(heart_disease_split)
```

```r
y_test <- test %>% pull(mortality)
x_test <- test %>% select(-mortality) %>% select(-ends_with("_2"))

## Train SuperLearner
sl = SuperLearner(Y = y_train,
                  X = x_train,
                  family = binomial(),
                  SL.library = models)
```

```
## Loading required namespace: ranger
```

```r
## Risk and Coefficient of each model
# 4. Report the risk and coefficient associated with each model
print(sl)
```

```
##
## Call:
## SuperLearner(Y = y_train, X = x_train, family = binomial(), SL.library = models)
##
##
##
##                     Risk      Coef
## SL.glmnet_All 0.2360545 0.3552838
## SL.mean_All   0.2498146 0.0000000
## SL.knn_All    0.2751184 0.0000000
## SL.lda_All    0.2362820 0.0000000
## SL.ranger_All 0.2314090 0.6447162
```

```r
## Discrete winner and superlearner ensemble performance
ranger = SuperLearner(
  Y = y_train,
  X = x_train,
  family = binomial(),
  SL.library = c("SL.ranger")
)
ranger_pred <- predict(ranger,
                       x_test,
                       onlySL = TRUE)
preds <- predict(sl,
                 x_test,
                 onlySL = TRUE)
validation <- y_test %>%
  bind_cols(preds$pred[, 1]) %>%
  bind_cols(ranger_pred$pred[, 1]) %>%
  rename(obs = `...1`,
         pred_sl = `...2`,
         pred_ranger = `...3`) %>%
  mutate(
    pred_sl = ifelse(pred_sl >= .5,
                     1,
                     0),
    pred_ranger = ifelse(pred_ranger >= .5,
                         1,
                         0)
  )
```

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

```
print("True Positive Rate of Discrete Winner")
```

```
## [1] "True Positive Rate of Discrete Winner"
```

```
sum(validation$pred_ranger == 1 & validation$obs == 1) /
( sum(validation$pred_ranger == 1 & validation$obs == 1) +
  sum(validation$pred_ranger == 0 & validation$obs == 1) )
```

```
## [1] 0.7511962
```

```
print("True Positive Rate of Ensemble")
```

```
## [1] "True Positive Rate of Ensemble"
```

```
sum(validation$pred_sl == 1 & validation$obs == 1) /
( sum(validation$pred_sl == 1 & validation$obs == 1) +
  sum(validation$pred_sl == 0 & validation$obs == 1) )
```

```
## [1] 0.8066986
```

```
## Confusion Matrix of Ensemble
print(caret::confusionMatrix(as.factor(validation$pred_sl),
                      as.factor(validation$obs)))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 312 202
##          1 643 843
##
##                Accuracy : 0.5775
##                  95% CI : (0.5555, 0.5993)
##     No Information Rate : 0.5225
##     P-Value [Acc > NIR] : 4.451e-07
##
##                   Kappa : 0.1361
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.3267
##             Specificity : 0.8067
##          Pos Pred Value : 0.6070
##          Neg Pred Value : 0.5673
##              Prevalence : 0.4775
##          Detection Rate : 0.1560
##    Detection Prevalence : 0.2570
##       Balanced Accuracy : 0.5667
##
##        'Positive' Class : 0
##
```

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)?

SuperLearner can achieve better prediction performance than any single algorithm. This is because SuperLearner is an ensemble method that combines the strengths of multiple algorithms to produce a better prediction model. The ensemble is able to capture a wider range of possible relationships between the predictors and the outcome, and can better handle complex, non-linear relationships that may be missed by individual algorithms. In this case, the SuperLearner combines the linear model of GLMnet and the highly nonlinear model of the Random Forest to produce a more flexible and accurate model.

Targeted Maximum Likelihood Estimation

Causal Diagram

TMLE requires estimating two models:

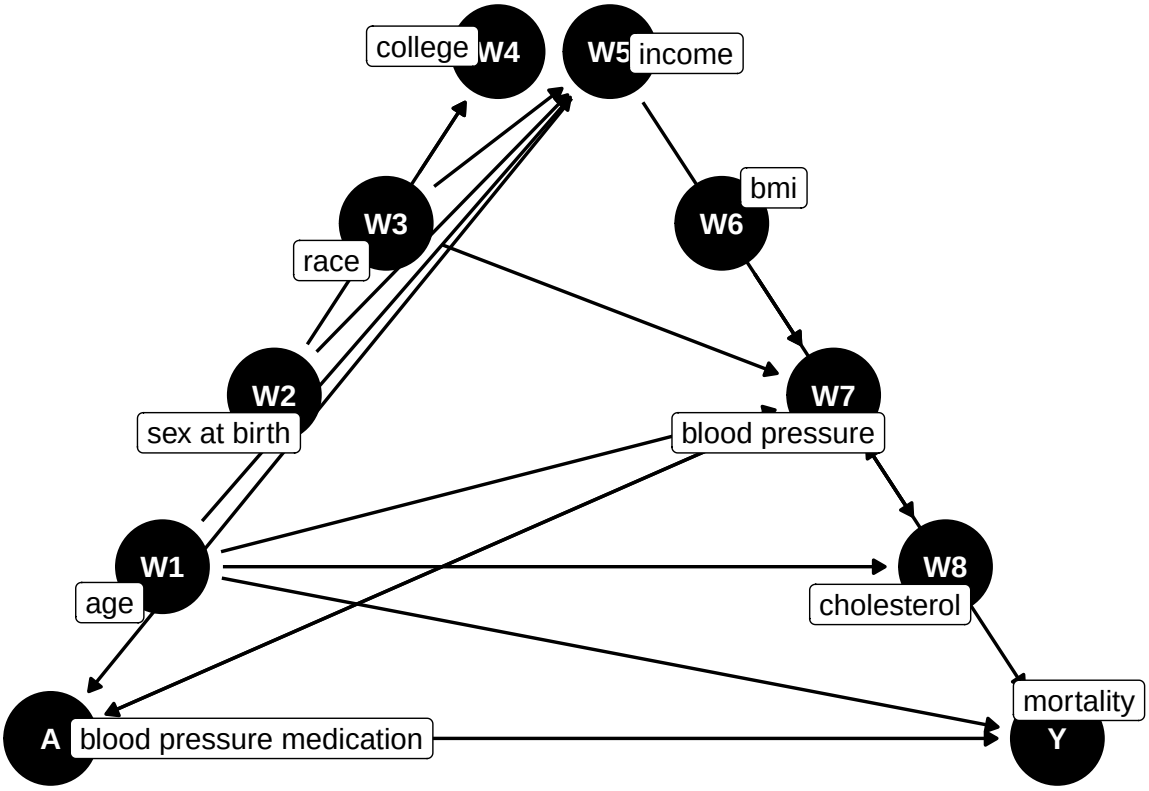
1. The outcome model, or the relationship between the outcome and the treatment/predictors, $P(Y|(A, W)$.
2. The propensity score model, or the relationship between assignment to treatment and predictors $P(A|W)$

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

source("pretty_dag.R")

dagify(Y ~ A + W1 + W7,
       W5 ~ W1 + W3 + W2 + W4,
       W7 ~ W3 + W6 + W5 + W1 + W8 + A,
       W8 ~ W6 + W1,
       W4 ~ W3 + W2,
       A ~ W5 + W7,
       exposure = "A",
       outcome = "Y",
       labels = c(Y = "mortality", A = "blood pressure medication",
                  W1 = "age", W2 = "sex at birth",
                  W3 = "race", W4 = "college",
                  W5 = "income", W6 = "bmi",
                  W7 = "blood pressure", W8 = "cholesterol"))%>%
  tidy_dagitty() %>%
  pretty_dag() %>%
  ggdag(use_labels = "label") + theme_dag()
```



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

```
models <- c("SL.glmnet","SL.ranger")
Y <- heart_disease %>%
  pull(mortality)
W <- heart_disease %>%
  select(-mortality, -blood_pressure_medication,
         -simplified_race, -college_educ, -sex_at_birth) %>%
  select(-ends_with("_2"))
A <-  heart_disease %>%
  pull(blood_pressure_medication)
tmle_fit <-
  tmle::tmle(Y = Y,
             A = A,
             W = W,
             Q.SL.library = models,
             g.SL.library = models)
```

```
tmle_fit
```

```
##  Additive Effect
##    Parameter Estimate:  -0.35652
##    Estimated Variance:  6.5483e-05
##               p-value:  <2e-16
##     95% Conf Interval: (-0.37238, -0.34066)
##
##  Additive Effect among the Treated
##    Parameter Estimate:  -0.31803
##    Estimated Variance:  0.00014521
##               p-value:  <2e-16
##     95% Conf Interval: (-0.34165, -0.29441)
##
##  Additive Effect among the Controls
##    Parameter Estimate:  -0.37189
##    Estimated Variance:  5.8421e-05
##               p-value:  <2e-16
##     95% Conf Interval: (-0.38687, -0.35691)
```

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 an estimator that is consistent even if either the outcome model or propensity score model is misspecified (as long as at least one of them is correctly specified).

In the context of targeted maximum likelihood estimation (TMLE), a double robust estimator is obtained by combining two stages of estimation. In the first stage, the propensity score is estimated using a flexible model (e.g. machine learning algorithms) that can capture the complex relationship between the covariates and the treatment. In the second stage, the outcome model is estimated using the residuals from the first stage as a new outcome variable. This allows the outcome model to be estimated in a way that accounts for the potential confounding effect of the covariates and the treatment.

The key idea behind the double robustness property of TMLE is that even if one of the models (either the propensity score or the outcome model) is misspecified, the other model can still be used to correct for any bias in the estimation of the treatment effect. For example, if the propensity score model is misspecified, the outcome model can still adjust for confounding by using the residuals from the propensity score model. If the outcome model is misspecified, the propensity score model can still adjust for confounding by balancing the covariates across treatment groups.

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).

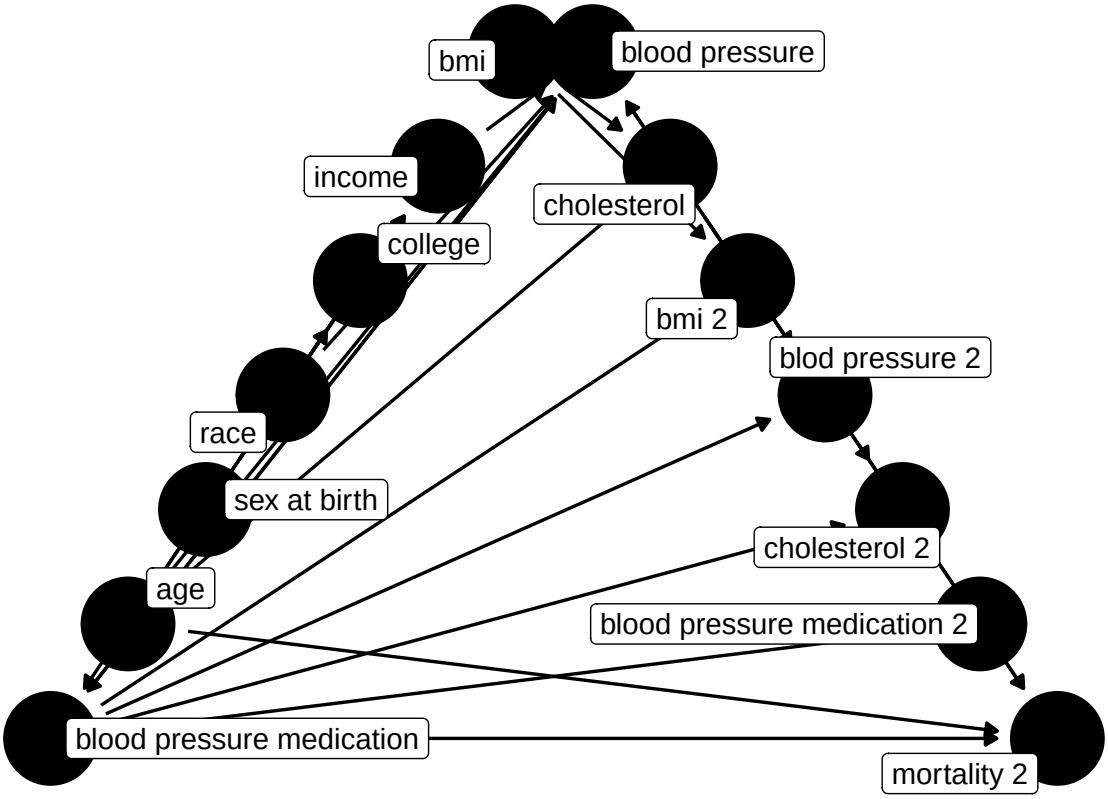
Causal Diagram

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 ~ A + W1 + W7 + W10,
       W5 ~ W1 + W3 + W2 + W4,
       W7 ~ W3 + W6 + W5 + W1 + W8 + A,
       W8 ~ W6 + W1,
       W4 ~ W3 + W2,
       W9 ~ W6 + A,
       W10 ~ W7 + A,
       W11 ~ W8 + A,
       W12 ~ A,
       A ~ W5 + W7,
       exposure = "A",
       outcome = "Y",
       labels = c(Y = "mortality 2", A = "blood pressure medication",
                  W1 = "age", W2 = "sex at birth",
                  W3 = "race", W4 = "college",
                  W5 = "income", W6 = "bmi",
                  W7 = "blood pressure", W8 = "cholesterol",
                  W9 = "bmi 2", W10 = "blod pressure 2",
                  W11 = "cholesterol 2", W12 = "blood pressure medication 2"))%>%
  tidy_dagitty() %>%
  pretty_dag() %>%
  ggdag(use_labels = "label", text = FALSE) + theme_dag()
```



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?

```
heart_disease
```

```
## # A tibble: 10,000 x 14
##      age sex_at_birth simplified_race college_educ income_thousands   bmi
##    <dbl>        <dbl>           <dbl>        <dbl>             <dbl> <dbl>
##  1  32.9            0               1            2              91.3  27.1
##  2  53.9            1               1            2              38.8  27.6
##  3  65.3            1               3            2              35.5  27.5
##  4  16.8            1               1            2              93.8  24.9
##  5  56.1            1               1            2              85.7  22.8
##  6  57.2            1               1            2              70.8  24.0
##  7  41.9            1               1            1              97.0  25.2
##  8  42.3            1               1            1             103.   27.8
##  9  42.0            1               3            2              61.2  25.8
## 10  37.4            0               3            1              55.0  23.5
## # i 9,990 more rows
## # i 8 more variables: blood_pressure <dbl>, chol <dbl>,
## #   blood_pressure_medication <dbl>, bmi_2 <dbl>, blood_pressure_2 <dbl>,
## #   chol_2 <dbl>, blood_pressure_medication_2 <dbl>, mortality <dbl>
```

```r
ltmle_data <- heart_disease %>%
  rename(W1 = age, W2 = income_thousands, W3 = bmi,
         W4 = blood_pressure, W5 = chol,
         A1 = blood_pressure_medication,
         A2 = blood_pressure_medication_2,
         Y = mortality) %>%
  select(c(W1, W2, W3, W4, A1, A2, Y))
## Naive Model (no time-dependent confounding) estimate
result <- ltmle(
  ltmle_data,
  Anodes = c("A1", "A2"),
  Ynodes = "Y",
  Lnodes = NULL,
  abar = c(1, 1),
  SL.library = models
)
```

```
## Qform not specified, using defaults:
## formula for Y:
## Q.kplus1 ~ W1 + W2 + W3 + W4 + A1 + A2
##
## gform not specified, using defaults:
## formula for A1:
## A1 ~ W1 + W2 + W3 + W4
## formula for A2:
## A2 ~ W1 + W2 + W3 + W4 + A1
##
## Error in lognet(xd, is.sparse, ix, jx, y, weights, offset, alpha, nobs,  :
##   one multinomial or binomial class has 1 or 0 observations; not allowed
## Estimate of time to completion: 3 to 11 minutes
```

```r
## LTMLE estimate
result
```

```
## Call:
## ltmle(data = ltmle_data, Anodes = c("A1", "A2"), Lnodes = NULL,
##     Ynodes = "Y", abar = c(1, 1), SL.library = models)
##
## TMLE Estimate:  0.2215453
```

```r
ltmle_data <- heart_disease %>%
  rename(W1 = age, W2 = income_thousands, W3 = bmi,
         W4 = blood_pressure, W5 = chol,
         A1 = blood_pressure_medication,
         L1 = bmi_2, L2 = blood_pressure_2,
         L3 = chol_2,
         A2 = blood_pressure_medication_2,
         Y = mortality) %>%
  select(c(W1, W2, W3, W4, A1, L1, L2, L3, A2, Y))
```

```r
## LTMLE estimate
result_long <- ltmle(
  ltmle_data,
  Anodes = c("A1", "A2"),
  Ynodes = "Y",
  Lnodes = c("L1", "L2", "L3"),
  abar = c(1, 1)
)
```

```
## Qform not specified, using defaults:
## formula for L1:
## Q.kplus1 ~ W1 + W2 + W3 + W4 + A1
## formula for Y:
## Q.kplus1 ~ W1 + W2 + W3 + W4 + A1 + L1 + L2 + L3 + A2
##
## gform not specified, using defaults:
## formula for A1:
## A1 ~ W1 + W2 + W3 + W4
## formula for A2:
## A2 ~ W1 + W2 + W3 + W4 + A1 + L1 + L2 + L3
##
## Estimate of time to completion: < 1 minute
```

```r
result_long
```

```
## Call:
## ltmle(data = ltmle_data, Anodes = c("A1", "A2"), Lnodes = c("L1",
##     "L2", "L3"), Ynodes = "Y", abar = c(1, 1))
##
## TMLE Estimate:  0.1922687
```

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 be especially concerned about time-dependent confounding variables that are affected by previous treatments. Time-dependent confounding variables are variables that are associated with both the treatment and the outcome, and their association may change over time. In this case, Age is not going to be affected at all by the initial blood pressure treatment, so it's not particularly worrying. On the other hand, cholesterol and blood pressure are probably going to be affected by the first treatment, and thus are the more-worrying time-dependent confounding variables.