Data Analysis Homework 2

Jimmy Hickey

9/25/2020

1

We will proceed with the same that I used in homework 1.

```
library(DynTxRegime)
```

```
## Loading required package: modelObj
```

```
data <- read.csv(file ="cholesterol.dat.txt", header = TRUE, sep = ",")
data$A= data$trt

y = data$chol0 - data$chol6

lm = buildModelObj(model = ~A + exercise + wt + smoke + trig0 + age + gender +
                     A:exercise + A:wt + A:smoke + A:trig0 + A:age + A:gender,
                   solver.method = "lm",
                   predict.method = "predict.lm",
                   predict.args = list("type"="response"))
# adj R^2 = 0.889

# summary(fit(lm, data, y))
```

a. regression-based estimator

```
# From slide 35 of Halloway
moMain <- buildModelObj(model = ~exercise + wt + smoke + trig0 + age + gender,
        solver.method = 'lm',
        predict.method = 'predict.lm')

moCont <- buildModelObj(model = ~exercise + wt + smoke + trig0 + age + gender,
        solver.method = 'lm',
        predict.method = 'predict.lm')

qObj <- qLearn(moMain = moMain, moCont = moCont, iter = 0L,
          data = data, response = y, txName = 'A',
          verbose = TRUE)
```

```
## First step of the Q-Learning Algorithm.
##
## Outcome regression.
## Combined outcome regression model: ~ exercise+wt+smoke+trig0+age+gender + A + A:(exercise+wt+smoke+tr
## Regression analysis for Combined:
##
## Call:
## lm(formula = YinternalY ~ exercise + wt + smoke + trig0 + age +
##     gender + A + exercise:A + wt:A + smoke:A + trig0:A + age:A +
##     gender:A, data = data)
##
## Coefficients:
## (Intercept)     exercise           wt        smoke        trig0          age
##   3.246e+01    2.058e+01   -2.393e-01    2.908e+00   -1.671e-02    9.491e-03
##      gender            A   exercise:A         wt:A      smoke:A      trig0:A
##   5.239e-01   -2.615e+02   -2.108e+01    1.621e+00   -5.081e+00    3.512e-02
##       age:A     gender:A
##   2.488e-02    9.056e-01
##
##
## Recommended Treatments:
##   0   1
## 211 789
##
## Estimated value: 33.75671
```

```
coef(object = qObj)
```

```
## $outcome
## $outcome$Combined
##   (Intercept)      exercise            wt         smoke         trig0
##  3.246193e+01  2.058365e+01 -2.392622e-01  2.907925e+00 -1.671489e-02
##           age        gender             A    exercise:A          wt:A
##  9.490884e-03  5.239228e-01 -2.614569e+02 -2.107722e+01  1.620786e+00
##       smoke:A       trig0:A         age:A      gender:A
## -5.081452e+00  3.511518e-02  2.488064e-02  9.055717e-01
```

```
fitObj = fitObject(object = qObj)
fitObj
```

```
## $outcome
## $outcome$Combined
##
## Call:
## lm(formula = YinternalY ~ exercise + wt + smoke + trig0 + age +
##     gender + A + exercise:A + wt:A + smoke:A + trig0:A + age:A +
##     gender:A, data = data)
##
## Coefficients:
## (Intercept)     exercise           wt        smoke        trig0          age
##   3.246e+01    2.058e+01   -2.393e-01    2.908e+00   -1.671e-02    9.491e-03
##      gender            A   exercise:A         wt:A      smoke:A      trig0:A
##   5.239e-01   -2.615e+02   -2.108e+01    1.621e+00   -5.081e+00    3.512e-02
```

```
##       age:A     gender:A
##   2.488e-02    9.056e-01
```

```r
ot <- optTx(x = qObj)
table(ot$optimalTx)
```

```
##
##   0   1
## 211 789
```

```r
estimator(x = qObj)
```

```
## [1] 33.75671
```

b. restricted value search

```r
# slide 54 of Halloway
regimes = function(eta1, data)
{
  d1 = {data$wt > eta1}
  return(as.integer(x = d1))
}

# propensity model from hw1
propensity <- modelObj::buildModelObj(model = ~ age + wt + gender + exercise + smoke + trig0 + chol0,
    solver.method = 'glm',
    solver.args = list(family='binomial'),
    predict.method = 'predict.glm',
    predict.args = list(type='response'))

# optimal seq from slide 56 of Halloway
# notice we only need a propensity model for equation 3.42
vsObj <- optimalSeq(moPropen = propensity,
            moMain = NULL, moCont = NULL, iter = 0L,
            data = data, response = y, txName = 'A',
            regimes = regimes,
            Domains = matrix(data = c(110, 290), ncol = 2L),
            starting.values = c(0,0), pop.size = 1000,
            verbose = TRUE)
```

```
## IPW estimator will be used
## Value Search - Missing Data Perspective.
##
## Propensity for treatment regression.
## Regression analysis for moPropen:
##
## Call:  glm(formula = YinternalY ~ age + wt + gender + exercise + smoke +
##     trig0 + chol0, family = "binomial", data = data)
##
## Coefficients:
```

```
## (Intercept)          age           wt       gender     exercise        smoke
##  -2.9404887    0.0009917    0.0083114   -0.0929438    0.3816924   -0.0976649
##       trig0        chol0
##  -0.0009232    0.0061492
##
## Degrees of Freedom: 999 Total (i.e. Null);  992 Residual
## Null Deviance:       1386
## Residual Deviance: 1366  AIC: 1382
##
## Outcome regression.
## No outcome regression performed.

## Warning in (function (fn, nvars, max = FALSE, pop.size = 1000, max.generations =
## 100, : Ignoring 'starting.values' because length(staring.values)!=nvars

##
##
## Thu Oct 01 12:50:27 2020
## Domains:
##  1.100000e+02   <=  X1   <=    2.900000e+02
##
## Data Type: Floating Point
## Operators (code number, name, population)
##  (1) Cloning..........................  122
##  (2) Uniform Mutation.................  125
##  (3) Boundary Mutation................  125
##  (4) Non-Uniform Mutation.............  125
##  (5) Polytope Crossover...............  125
##  (6) Simple Crossover.................  126
##  (7) Whole Non-Uniform Mutation........  125
##  (8) Heuristic Crossover..............  126
##  (9) Local-Minimum Crossover..........  0
##
## HARD Maximum Number of Generations: 100
## Maximum Nonchanging Generations: 10
## Population size       : 1000
## Convergence Tolerance: 1.000000e-03
##
## Not Using the BFGS Derivative Based Optimizer on the Best Individual Each Generation.
## Not Checking Gradients before Stopping.
## Using Out of Bounds Individuals.
##
## Maximization Problem.
##
##
## Generation#      Solution Value
##
##       0  3.439419e+01
##       1  3.440739e+01
##
## 'wait.generations' limit reached.
## No significant improvement in 10 generations.
##
## Solution Fitness Value: 3.440739e+01
```

```
##
## Parameters at the Solution:
##
##  X[ 1] : 1.578248e+02
##
## Solution Found Generation 1
## Number of Generations Run 12
##
## Thu Oct 01 12:50:41 2020
## Total run time : 0 hours 0 minutes and 14 seconds
## Genetic Algorithm
## $value
## [1] 34.40739
##
## $par
## [1] 157.8248
##
## $gradients
## [1] NA
##
## $generations
## [1] 12
##
## $peakgeneration
## [1] 1
##
## $popsize
## [1] 1000
##
## $operators
## [1] 122 125 125 125 125 126 125 126   0
##
##
## Recommended Treatments:
##   0   1
## 224 776
##
## Estimated value: 34.40739
```

```r
regimeCoef(vsObj)
```

```
##     eta1
## 157.8248
```

```r
estimator(vsObj)
```

```
## [1] 34.40739
```

Notice that our optimal eta is $\eta = 157.8948$. This agrees with the graph that we got from homework 1 question 2 b!

c

```r
# notice that the difference here is we need our moMain and moCont are no longer null
vsObj2 <- optimalSeq(moPropen = propensity,
            moMain = moMain, moCont = moCont, iter = 0L,
            data = data, response = y, txName = 'A',
            regimes = regimes,
            Domains = matrix(data = c(110, 290), ncol = 2L),
            starting.values = c(0,0), pop.size = 1000,
            verbose = TRUE)
```

```
## Value Search - Missing Data Perspective.
##
## Propensity for treatment regression.
## Regression analysis for moPropen:
##
## Call:  glm(formula = YinternalY ~ age + wt + gender + exercise + smoke +
##     trig0 + chol0, family = "binomial", data = data)
##
## Coefficients:
## (Intercept)          age           wt       gender     exercise        smoke
##  -2.9404887    0.0009917    0.0083114   -0.0929438    0.3816924   -0.0976649
##       trig0        chol0
##  -0.0009232    0.0061492
##
## Degrees of Freedom: 999 Total (i.e. Null);  992 Residual
## Null Deviance:       1386
## Residual Deviance: 1366  AIC: 1382
##
## Outcome regression.
## Combined outcome regression model: ~ exercise+wt+smoke+trig0+age+gender + A + A:(exercise+wt+smoke+t
## Regression analysis for Combined:
##
## Call:
## lm(formula = YinternalY ~ exercise + wt + smoke + trig0 + age +
##     gender + A + exercise:A + wt:A + smoke:A + trig0:A + age:A +
##     gender:A, data = data)
##
## Coefficients:
## (Intercept)     exercise           wt        smoke        trig0          age
##   3.246e+01    2.058e+01   -2.393e-01    2.908e+00   -1.671e-02    9.491e-03
##      gender            A   exercise:A         wt:A      smoke:A      trig0:A
##   5.239e-01   -2.615e+02   -2.108e+01    1.621e+00   -5.081e+00    3.512e-02
##       age:A     gender:A
##   2.488e-02    9.056e-01
```

```
## Warning in (function (fn, nvars, max = FALSE, pop.size = 1000, max.generations =
## 100, : Ignoring 'starting.values' because length(staring.values)!=nvars
```

```
##
##
## Thu Oct 01 12:50:41 2020
```

```
## Domains:
##  1.100000e+02   <=  X1   <=    2.900000e+02
##
## Data Type: Floating Point
## Operators (code number, name, population)
##  (1) Cloning..........................  122
##  (2) Uniform Mutation.................  125
##  (3) Boundary Mutation................  125
##  (4) Non-Uniform Mutation.............  125
##  (5) Polytope Crossover...............  125
##  (6) Simple Crossover.................  126
##  (7) Whole Non-Uniform Mutation........  125
##  (8) Heuristic Crossover..............  126
##  (9) Local-Minimum Crossover..........  0
##
## HARD Maximum Number of Generations: 100
## Maximum Nonchanging Generations: 10
## Population size       : 1000
## Convergence Tolerance: 1.000000e-03
##
## Not Using the BFGS Derivative Based Optimizer on the Best Individual Each Generation.
## Not Checking Gradients before Stopping.
## Using Out of Bounds Individuals.
##
## Maximization Problem.
##
##
## Generation#      Solution Value
##
##       0  3.472935e+01
##
## 'wait.generations' limit reached.
## No significant improvement in 10 generations.
##
## Solution Fitness Value: 3.472935e+01
##
## Parameters at the Solution:
##
##  X[ 1] : 1.583883e+02
##
## Solution Found Generation 1
## Number of Generations Run 11
##
## Thu Oct 01 12:51:23 2020
## Total run time : 0 hours 0 minutes and 42 seconds
## Genetic Algorithm
## $value
## [1] 34.72935
##
## $par
## [1] 158.3883
##
## $gradients
## [1] NA
```

```
##
## $generations
## [1] 11
##
## $peakgeneration
## [1] 1
##
## $popsize
## [1] 1000
##
## $operators
## [1] 122 125 125 125 125 126 125 126   0
##
##
## Recommended Treatments:
##   0   1
## 227 773
##
## Estimated value: 34.72935
```

```r
regimeCoef(vsObj2)
```

```
##     eta1
## 158.3883
```

```r
estimator(vsObj2)
```

```
## [1] 34.72935
```

Notice that our optimal eta is $\eta = 158.5162$. This is similar to the estimate in (b) and also with the graph from homework 1.

d

```r
require(rpart)
```

```
## Loading required package: rpart
```

```r
moClass <- buildModelObj(model = ~exercise + wt + smoke + trig0 + age + gender,
  solver.method = 'rpart',
  predict.method = 'predict',
  predict.args = list(type = "class"))


clObj <- optimalClass(moPropen = propensity,
            moMain = moMain, moCont = moCont, iter = 0L,
            moClass = moClass,
            data = data, response = y, txName = 'A',
            verbose = TRUE)
```

```
## AIPW value estimator

## First step of the Classification Algorithm.

## Classification Perspective.

##
## Propensity for treatment regression.
## Regression analysis for moPropen:
##
## Call:  glm(formula = YinternalY ~ age + wt + gender + exercise + smoke +
##     trig0 + chol0, family = "binomial", data = data)
##
## Coefficients:
## (Intercept)          age           wt       gender     exercise        smoke
##  -2.9404887    0.0009917    0.0083114   -0.0929438    0.3816924   -0.0976649
##       trig0        chol0
##  -0.0009232    0.0061492
##
## Degrees of Freedom: 999 Total (i.e. Null);  992 Residual
## Null Deviance:       1386
## Residual Deviance: 1366  AIC: 1382
##
## Outcome regression.
## Combined outcome regression model: ~ exercise+wt+smoke+trig0+age+gender + A + A:(exercise+wt+smoke+t
## Regression analysis for Combined:
##
## Call:
## lm(formula = YinternalY ~ exercise + wt + smoke + trig0 + age +
##     gender + A + exercise:A + wt:A + smoke:A + trig0:A + age:A +
##     gender:A, data = data)
##
## Coefficients:
## (Intercept)     exercise           wt        smoke        trig0          age
##   3.246e+01    2.058e+01   -2.393e-01    2.908e+00   -1.671e-02    9.491e-03
##      gender            A   exercise:A         wt:A      smoke:A      trig0:A
##   5.239e-01   -2.615e+02   -2.108e+01    1.621e+00   -5.081e+00    3.512e-02
##       age:A     gender:A
##   2.488e-02    9.056e-01
##
##
## Classification Analysis
## Regression analysis for moClass:
## n= 1000
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 1000 0.138746100 1 (0.0252960136 0.9747039864)
##    2) wt< 158.55 227 0.010564510 0 (0.6448357626 0.3551642374) *
##    3) wt>=158.55 773 0.019682370 1 (0.0037134730 0.9962865270)
##      6) wt< 167.25 99 0.015095020 1 (0.0935823105 0.9064176895)
##       12) smoke>=0.5 24 0.002070983 0 (0.3282265705 0.6717734295) *
```

```
##       13) smoke< 0.5 75 0.008813891 1 (0.0619973915 0.9380026085)
##         26) exercise>=0.5 14 0.000556363 0 (0.5534174300 0.4465825700) *
##         27) exercise< 0.5 61 0.004534129 1 (0.0337280043 0.9662719957) *
##      7) wt>=167.25 674 0.004587345 1 (0.0008926608 0.9991073392) *
## Recommended Treatments:
##   0   1
## 265 735
##
## Estimated value: 35.17289
```

```r
coef(object = clObj)
```

```
## $propensity
##   (Intercept)           age            wt        gender      exercise
## -2.9404886577  0.0009916959  0.0083114322 -0.0929437626  0.3816924426
##         smoke         trig0         chol0
## -0.0976648828 -0.0009231713  0.0061491542
##
## $outcome
## $outcome$Combined
##   (Intercept)      exercise            wt         smoke         trig0
##  3.246193e+01  2.058365e+01 -2.392622e-01  2.907925e+00 -1.671489e-02
##           age        gender             A    exercise:A          wt:A
##  9.490884e-03  5.239228e-01 -2.614569e+02 -2.107722e+01  1.620786e+00
##       smoke:A       trig0:A         age:A      gender:A
## -5.081452e+00  3.511518e-02  2.488064e-02  9.055717e-01
```

```r
table(ot$optimalTx)
```

```
##
##   0   1
## 211 789
```

```r
estimator(x = clObj)
```

```
## [1] 35.17289
```

2

a

```r
ldl = read.table("LDL.dat.txt", header=FALSE)

# remove ID column
ldl = ldl[,-1]
names(ldl) = c("L1", "A1", "L2", "S2", "A2", "L3",
               "S3", "A3", "L4", "S4", "A4", "Y", "S5")

calc_betas = function(data, K){
```

```r
### Setting up variables in equations

  # number of datapoints
  n = dim(data)[1]

  # LDL measurements
  L = cbind(data$L1, data$L2, data$L3, data$L4, data$Y)

  # Statin dose received
  A = cbind(data$A1, data$A2, data$A3, data$A4)

  # Side effects
  S = cbind(rep(0,n), data$S2, data$S3, data$S4, data$S5)

  # Y outcome vector
  Y = matrix(0, nrow = (K+1) * n, ncol = 1)

  # X design matrix
  X = matrix(0, nrow = (K+1) * n, ncol = 7)

  for(i in 1:n){
    ind = (i-1) * (K+1)

    X[ind, ] = c(1, rep(0,6))
    Y[ind, ] = L[i, 1]

    for(k in 2:(K+1)){
      Y[ind + k] = L[i,k] - L[i,k-1]
      X[ind + k, ] = c(0,
                  1 - S[i, k-1],
                  A[i, k-1]*(1-S[i,k-1]),
                  L[i,k-1]*(1-S[i,k-1]),
                  A[i,k-1]*L[i,k-1]*(1-S[i,k-1]),
                  S[i,k-1],
                  S[i,k-1]*A[i,k-1])
    }
  }

  df = data.frame(cbind(Y,X))
  names(df) = c("y","beta1","beta2","beta3","beta4","beta5","beta6","beta7")
  # fit linear model
  # -1 removes intercept
  lm = lm(y ~ -1 + beta1 + beta2 + beta3 + beta4 + beta5 + beta6 + beta7, data = df)
  return(lm)
}

K = 4

betaslm = calc_betas(ldl, K)
betas = coef(betaslm)
sigmasq = (summary(betaslm)$sigma)^2
```

```r
cat("=================\n
betas\n
==================")
```

```
## =================
##
## betas
##
## ==================
```

```r
print(betas)
```

```
##         beta1         beta2         beta3         beta4         beta5
## 170.089217844  -6.216885342 -13.084848560  -0.002577804   0.018117471
##         beta6         beta7
##  -8.606951872   4.188120703
```

```r
cat("=================\n
sigma^2\n
==================")
```

```
## =================
##
## sigma^2
##
## ==================
```

```r
print(sigmasq)
```

```
## [1] 113.9684
```

b

```r
calc_psis = function(data, K){

  # number of datapoints
  n = dim(data)[1]

  # LDL measurements
  L = cbind(data$L1, data$L2, data$L3, data$L4, data$Y)

  # Statin dose received
  A = cbind(data$A1, data$A2, data$A3, data$A4)

  # Side effects
  S = cbind(rep(0,n), data$S2, data$S3, data$S4, data$S5)

  # Y outcome vector
  # take off side effects at 12 months
```

```r
  Ylogis = matrix(0, nrow = (K-1)*n, ncol = 1)

  # X design matrix for logistic regression
  Xlogis = matrix(0, nrow = (K-1) * n, ncol = K)

  for(i in 1:n){
    ind = (i-1) * (K-1)

    for(k in 2:K){
      abark = sum(A[i, 1:(k-1)])
      Xlogis[ind + (k-1), ] =  c(1, abark,
                        abark * L[i, k-1],
                        S[i, k-1] * A[i, k-1])
      Ylogis[i+(k-1)] = S[i,k]
    }

  }
  df = data.frame(cbind(Ylogis, Xlogis))
  names(df) = c("Y", "psi1", "psi2", "psi3", "psi4")

  psifit = glm(Y ~-1+ psi1 + psi2 + psi3 + psi4, data = df, family = binomial)
  psis = coef(psifit)
  return(psis)
}

psis = calc_psis(ldl, K)

cat("=================\n
psis\n
==================")
```

```
## =================
##
## psis
##
## ==================
```

```r
print(psis)
```

```
##          psi1          psi2          psi3          psi4
## -3.9486447206 -0.0022157446  0.0003883022 -0.7733072237
```

c

```r
logistic_func = function(x){
  return( exp(x) / (1 + exp(x)) )
}


gcomp = function(data, regime, K, M){
```

```r
  bfit = calc_betas(data, K)
  betas = coef(bfit)
  sigma = summary(bfit)$sigma

  psis = calc_psis(data, K)

   y = rep(0, M)

  for(r in 1:M){
    L = rep(0, K+1)
    S = rep(0, K+1)
    A = rep(0, K)

    # random draw for L1
    L[1] = rnorm(n=1, mean=betas[1], sd=sigma)

    for(k in 2:(K+1)){
      # dose
      A[k-1] = regime(L, S, A, k-1)

      # Equation 3
      mu =  (betas[2] + betas[3]*A[k-1] + betas[4]*L[k-1] +
             betas[5]*A[k-1]*L[k-1]) * (1-S[k-1]) +
             (betas[6] + betas[7]*A[k-1])*S[k-1]

      L[k] = L[k-1] + rnorm(n=1, mean=mu, sd=sigma)

      # Equation 4
      Acum = sum(A[1:(k-1)])
      prob = logistic_func( psis[1] + psis[2] * Acum +
                     psis[3] * Acum * L[k-1] + psis[4] * S[k-1] * A[k-1])
      S[k] = rbinom(n=1, size=1, prob=prob)
    }
    y[r] = L[K+1]
  }

  return(mean(y))
}

bootstrap_gcomp = function(data, regime, K, M, rep){
  nrow = dim(data[1])
  out = rep(0, rep)

  for(i in 1:rep){
    sample = data[sample(nrow, replace=TRUE),]
    out[i] = gcomp(data, regime, K, M)
  }

  return(sd(out))
}

# STATIC REGIMES
stat_reg1 = function(L, S, A, dk){
```

```r
  return(0)
}

stat_reg2 = function(L, S, A, dk){
  return(dk %in% c(4) )
}


stat_reg3 = function(L, S, A, dk){
  return(dk %in% c(3, 4) )
}

stat_reg4 = function(L, S, A, dk){
  return(dk %in% c(2, 3, 4) )
}

stat_reg5 = function(L, S, A, dk){
  return(1)
}

stat_reg6 = function(L, S, A, dk){
  return(dk %in% c(1, 2, 3) )
}

stat_reg7 = function(L, S, A, dk){
  return(dk %in% c(1,2) )
}

stat_reg8 = function(L, S, A, dk){
  return(dk %in% c(1) )
}

# regime 1
est1 = gcomp(data = ldl, regime = stat_reg1, K = 4, M = 1000)
sd1 = bootstrap_gcomp(data = ldl, regime = stat_reg1, K = 4, M = 1000, rep = 100)

cat("================\nregime 1\nestimate:\t", est1, "\nstderr:\t\t", sd1, "\n================")
```

```
## ================
## regime 1
## estimate:     144.0657
## stderr:       0.6840945
## ================
```

```r
# regime 2
est2 = gcomp(data = ldl, regime = stat_reg2, K = 4, M = 1000)
sd2 = bootstrap_gcomp(data = ldl, regime = stat_reg2, K = 4, M = 1000, rep = 100)

cat("================\nregime 2\nestimate:\t", est2, "\nstderr:\t\t", sd2, "\n================")
```

```
## ================
## regime 2
```

```
## estimate:     132.2996
## stderr:       0.8070279
## ================
```

```r
# regime 3
est3 = gcomp(data = ldl, regime = stat_reg3, K = 4, M = 1000)
sd3 = bootstrap_gcomp(data = ldl, regime = stat_reg3, K = 4, M = 1000, rep = 100)

cat("================\nregime 3\nestimate:\t", est3, "\nstderr:\t\t", sd3, "\n================")
```

```
## ================
## regime 3
## estimate:     123.266
## stderr:       0.7614343
## ================
```

```r
# regime 4
est4 = gcomp(data = ldl, regime = stat_reg4, K = 4, M = 1000)
sd4 = bootstrap_gcomp(data = ldl, regime = stat_reg4, K = 4, M = 1000, rep = 100)

cat("================\nregime 4\nestimate:\t", est4, "\nstderr:\t\t", sd4, "\n================")
```

```
## ================
## regime 4
## estimate:     114.5438
## stderr:       0.7131436
## ================
```

```r
# regime 5
est5 = gcomp(data = ldl, regime = stat_reg5, K = 4, M = 1000)
sd5 = bootstrap_gcomp(data = ldl, regime = stat_reg5, K = 4, M = 1000, rep = 100)

cat("================\nregime 5\nestimate:\t", est5, "\nstderr:\t\t", sd5, "\n================")
```

```
## ================
## regime 5
## estimate:     102.2905
## stderr:       0.8603035
## ================
```

```r
# regime 6
est6 = gcomp(data = ldl, regime = stat_reg6, K = 4, M = 1000)
sd6 = bootstrap_gcomp(data = ldl, regime = stat_reg6, K = 4, M = 1000, rep = 100)

cat("================\nregime 6\nestimate:\t", est6, "\nstderr:\t\t", sd6, "\n================")
```

```
## ================
## regime 6
## estimate:     113.1155
## stderr:       0.7111857
## ================
```

```r
# regime 7
est7 = gcomp(data = ldl, regime = stat_reg7, K = 4, M = 1000)
sd7 = bootstrap_gcomp(data = ldl, regime = stat_reg7, K = 4, M = 1000, rep = 100)

cat("================\nregime 7\nestimate:\t", est7, "\nstderr:\t\t", sd7, "\n================")
```

```
## ================
## regime 7
## estimate:     122.5295
## stderr:       0.8125751
## ================
```

```r
# regime 8
est8 = gcomp(data = ldl, regime = stat_reg8, K = 4, M = 1000)
sd8 = bootstrap_gcomp(data = ldl, regime = stat_reg8, K = 4, M = 1000, rep = 100)

cat("================\nregime 8\nestimate:\t", est8, "\nstderr:\t\t", sd8, "\n================")
```

```
## ================
## regime 8
## estimate:     131.2473
## stderr:       0.678222
## ================
```

d

i

```r
regime_d1 = function(L, S, A, dk){
  # only 0 if the patient is currently having a side effect
  return(!S[dk])
}

estd1 = gcomp(data = ldl, regime = regime_d1, K = 4, M = 1000)
sdd1 = bootstrap_gcomp(data = ldl, regime = regime_d1, K = 4, M = 1000, rep = 100)

cat("================\nregime d1\nestimate:\t", estd1, "\nstderr:\t\t", sdd1, "\n================")
```

```
## ================
## regime d1
## estimate:     102.5976
## stderr:       0.8046744
## ================
```

ii

```r
regime_d2 = function(L, S, A, dk){
  # 0 if the patient has ever had a side effect
  return(!(1 %in% dk))
}

estd2 = gcomp(data = ldl, regime = regime_d2, K = 4, M = 1000)
sdd2 = bootstrap_gcomp(data = ldl, regime = regime_d2, K = 4, M = 1000, rep = 100)

cat("================\nregime d2\nestimate:\t", estd2, "\nstderr:\t\t", sdd2, "\n================")
```

```
## ================
## regime d2
## estimate:     113.8413
## stderr:       0.766292
## ================
```

e

```r
etas = seq(90, 200, 10)

for(i in 1:length(etas)){
  eta_i = etas[i]

  regime_eta = function(L, S, A, dk){
    return(S[dk] == 0 && L[dk] > eta_i)
  }

  estd2 = gcomp(data = ldl, regime = regime_d2, K = 4, M = 1000)
  sdd2 = bootstrap_gcomp(data = ldl, regime = regime_d2, K = 4, M = 1000, rep = 100)

  cat("================\nregime eta=", eta_i,"\nestimate:\t", estd2, "\nstderr:\t\t", sdd2, "\n========

}
```

```
## ================
## regime eta= 90
## estimate:     113.9798
## stderr:       0.789289
## ================================
## regime eta= 100
## estimate:     112.784
## stderr:       0.8867998
## ================================
## regime eta= 110
## estimate:     112.3233
## stderr:       0.8032969
## ================================
## regime eta= 120
## estimate:     112.7542
## stderr:       0.767764
## ================================
```

```
## regime eta= 130
## estimate:     112.6979
## stderr:       0.8017521
## ================================
## regime eta= 140
## estimate:     112.597
## stderr:       0.7603575
## ================================
## regime eta= 150
## estimate:     111.9736
## stderr:       0.7417239
## ================================
## regime eta= 160
## estimate:     112.1846
## stderr:       0.8131061
## ================================
## regime eta= 170
## estimate:     112.8498
## stderr:       0.7580995
## ================================
## regime eta= 180
## estimate:     115.0788
## stderr:       0.7359491
## ================================
## regime eta= 190
## estimate:     113.6293
## stderr:       0.8346585
## ================================
## regime eta= 200
## estimate:     113.505
## stderr:       0.7289813
## ================
```

f

Notice that for the static regimes, the value is higher for regimes with fewer high doses. Regime 5 (all high dose) has the lowest value.

In part (d), the second rule has a higher value than the first. This shows that it may be a better idea to stop high doses to anyone that has EVER shown a side effect, rather than just people that are currently experiencing one.

In part (e), all the regimes have similar value, regardless of the eta value.

3

a

```
calc_gamma = function(data){
  out = matrix(0, nrow=4, ncol=3)
```

```r
  gamma1_mod = glm(A1 ~ L1, data, family = "binomial")
  # add extra 0 because other temrs has an S factor
  out[1,] = c(gamma1_mod$coefficients, 0)

  gamma2_mod = glm(A2 ~ L2 + S2, data, family = "binomial")
  out[2,] = gamma2_mod$coefficients

  gamma3_mod = glm(A3 ~ L3 + S3, data, family = "binomial")
  out[3,] = gamma3_mod$coefficients

  gamma4_mod = glm(A4 ~ L4 + S4, data, family = "binomial")
  out[4,] = gamma4_mod$coefficients

  return(out)
}

gammas = calc_gamma(ldl)
colnames(gammas) = c("gamma_k1", "gamma_k2", "gamma_k3")
row.names(gammas) = c("1", "2", "3", "4")

print(gammas)
```

```
##     gamma_k1    gamma_k2     gamma_k3
## 1 -0.9918005 0.006118285  0.000000000
## 2 -0.4655975 0.003045530 -0.090446238
## 3 -0.4550628 0.002432314 -0.187644615
## 4 -0.5344407 0.002083461 -0.001472349
```

b

```r
# Cd vector for equation 5.27 on slide 304
calc_cd = function(data, regime, K)
{
  n = dim(data)[1]

  L = cbind(data$L1, data$L2, data$L3, data$L4, data$Y)
  # again need 0s becasue there are no side effects at the beginning
  S = cbind(rep(0, n), data$S2, data$S3, data$S4, data$S5)
  A = cbind(data$A1, data$A2, data$A3, data$A4)


  cd_vec = rep(1, n)

  for(i in 1:n){

    for(k in 1:K){
        cd_vec[i] = cd_vec[i] * ( A[i,k] == regime(L[i,k], S[i,k], A[i,k], k))

    }
  }
  return(cd_vec)
```

```r
}


# equation 5.27 on slide 304
calc_ipw = function(data, regime, K){

  n = dim(data)[1]

  Y = data$Y
  L = cbind(data$L1, data$L2, data$L3, data$L4)
  # again need 0s becasue there are no side effects at the beginning
  S = cbind(rep(0, n), data$S2, data$S3, data$S4, data$S5)
  A = cbind(data$A1, data$A2, data$A3, data$A4)

  cd = calc_cd(data, regime, K)
  gamma = calc_gamma(data)

  ipw_est = 0

  for(i in 1:n){
    num = cd[i]

    # only need to calculate if Cd == 1
    if(cd[i]){

      # mulitply by Yi
      num = Y[i]

      denom = 1
      # calculate denominator
      # the product of the propensities in equation 7
      for(k in 1:K){
        val = gamma[k, 1] + gamma[k, 2] * L[i,k] + gamma[k, 3] * S[i,k]
        p = logistic_func(val)
        dk = regime(L[i,], S[i,], A[i,], k)
        denom = denom*(dk * p + (1-dk)*(1-p))
      }

      ipw_est = ipw_est + num / denom

    } # end if
  }

  return(ipw_est/n)
}


bootstrap_ipw = function(data, regime, K, rep){
  out = NULL
  nrow = dim(data)[1]

  for(i in 1:rep){
    sample = data[sample(nrow, replace=TRUE),]
```

```r
    ipw = calc_ipw(sample, regime, K)
    out = rbind(out, ipw)
  }
  return(sd(out))
}


# regime 1
ipw_est1 = calc_ipw(data = ldl, regime = stat_reg1, K = 4)
ipw_sd1 = bootstrap_ipw(data = ldl, regime = stat_reg1, K = 4, rep = 100)

cat("================\nregime 1\nipw_estimate:\t", ipw_est1, "\nstderr:\t\t", ipw_sd1, "\n=============
```

```
## ================
## regime 1
## ipw_estimate:     135.6029
## stderr:       6.326165
## ================
```

```r
# regime 2
ipw_est2 = calc_ipw(data = ldl, regime = stat_reg2, K = 4)
ipw_sd2 = bootstrap_ipw(data = ldl, regime = stat_reg2, K = 4, rep = 100)

cat("================\nregime 2\nipw_estimate:\t", ipw_est2, "\nstderr:\t\t", ipw_sd2, "\n=============
```

```
## ================
## regime 2
## ipw_estimate:     144.0354
## stderr:       6.660548
## ================
```

```r
# regime 3
ipw_est3 = calc_ipw(data = ldl, regime = stat_reg3, K = 4)
ipw_sd3 = bootstrap_ipw(data = ldl, regime = stat_reg3, K = 4, rep = 100)

cat("================\nregime 3\nipw_estimate:\t", ipw_est3, "\nstderr:\t\t", ipw_sd3, "\n=============
```

```
## ================
## regime 3
## ipw_estimate:     135.4788
## stderr:       7.754829
## ================
```

```r
# regime 4
ipw_est4 = calc_ipw(data = ldl, regime = stat_reg4, K = 4)
ipw_sd4 = bootstrap_ipw(data = ldl, regime = stat_reg4, K = 4, rep = 100)

cat("================\nregime 4\nipw_estimate:\t", ipw_est4, "\nstderr:\t\t", ipw_sd4, "\n=============
```

```
## ================
## regime 4
```

```
## ipw_estimate:     113.821
## stderr:       5.337564
## ================
```

```r
# regime 5
ipw_est5 = calc_ipw(data = ldl, regime = stat_reg5, K = 4)
ipw_sd5 = bootstrap_ipw(data = ldl, regime = stat_reg5, K = 4, rep = 100)

cat("================\nregime 5\nipw_estimate:\t", ipw_est5, "\nstderr:\t\t", ipw_sd5, "\n=============
```

```
## ================
## regime 5
## ipw_estimate:     101.6646
## stderr:       5.241136
## ================
```

```r
# regime 6
ipw_est6 = calc_ipw(data = ldl, regime = stat_reg6, K = 4)
ipw_sd6 = bootstrap_ipw(data = ldl, regime = stat_reg6, K = 4, rep = 100)

cat("================\nregime 6\nipw_estimate:\t", ipw_est6, "\nstderr:\t\t", ipw_sd6, "\n=============
```

```
## ================
## regime 6
## ipw_estimate:     113.033
## stderr:       5.971046
## ================
```

```r
# regime 7
ipw_est7 = calc_ipw(data = ldl, regime = stat_reg7, K = 4)
ipw_sd7 = bootstrap_ipw(data = ldl, regime = stat_reg7, K = 4, rep = 100)

cat("================\nregime 7\nipw_estimate:\t", ipw_est7, "\nstderr:\t\t", ipw_sd7, "\n=============
```

```
## ================
## regime 7
## ipw_estimate:     133.8875
## stderr:       5.754353
## ================
```

```r
# regime 8
ipw_est8 = calc_ipw(data = ldl, regime = stat_reg8, K = 4)
ipw_sd8 = bootstrap_ipw(data = ldl, regime = stat_reg8, K = 4, rep = 100)

cat("================\nregime 8\nipw_estimate:\t", ipw_est8, "\nstderr:\t\t", ipw_sd8, "\n=============
```

```
## ================
## regime 8
## ipw_estimate:     133.7209
## stderr:       4.982602
## ================
```

```r
# d1
ipw_estd1 = calc_ipw(data = ldl, regime = regime_d2, K = 4)
ipw_sdd1 = bootstrap_ipw(data = ldl, regime = regime_d2, K = 4,rep = 100)

cat("================\nregime d1\nipw_estimate:\t", ipw_estd1, "\nstderr:\t\t", ipw_sdd1, "\n==========
```

```
## ================
## regime d1
## ipw_estimate:     113.821
## stderr:       5.625715
## ================
```

```r
ipw_estd2 = calc_ipw(data = ldl, regime = regime_d2, K = 4)
ipw_sdd2 = bootstrap_ipw(data = ldl, regime = regime_d2, K = 4, rep = 100)

cat("================\nregime d2\nipw_estimate:\t", ipw_estd2, "\nstderr:\t\t", ipw_sdd2, "\n==========
```

```
## ================
## regime d2
## ipw_estimate:     113.821
## stderr:       5.513688
## ================
```

```r
etas = seq(90, 200, 10)

for(i in 1:length(etas)){
  eta_i = etas[i]

  regime_eta = function(L, S, A, dk){
    return(S[dk] == 0 && L[dk] > eta_i)
  }

  ipw_estd2 = calc_ipw(data = ldl, regime = regime_d2, K = 4)
  ipw_sdd2 = bootstrap_ipw(data = ldl, regime = regime_d2, K = 4, rep = 100)

  cat("================\nregime eta=", eta_i,"\nipw_estimate:\t", ipw_estd2, "\nstderr:\t\t", ipw_sdd2,

}
```

```
## ================
## regime eta= 90
## ipw_estimate:     113.821
## stderr:       6.094825
## ================================
## regime eta= 100
## ipw_estimate:     113.821
## stderr:       5.296156
## ================================
## regime eta= 110
## ipw_estimate:     113.821
## stderr:       6.308763
## ================================
```

```
## regime eta= 120
## ipw_estimate:     113.821
## stderr:       6.315105
## ================================
## regime eta= 130
## ipw_estimate:     113.821
## stderr:       5.388843
## ================================
## regime eta= 140
## ipw_estimate:     113.821
## stderr:       6.394335
## ================================
## regime eta= 150
## ipw_estimate:     113.821
## stderr:       6.585507
## ================================
## regime eta= 160
## ipw_estimate:     113.821
## stderr:       5.997337
## ================================
## regime eta= 170
## ipw_estimate:     113.821
## stderr:       6.414407
## ================================
## regime eta= 180
## ipw_estimate:     113.821
## stderr:       6.82205
## ================================
## regime eta= 190
## ipw_estimate:     113.821
## stderr:       6.300405
## ================================
## regime eta= 200
## ipw_estimate:     113.821
## stderr:       6.708031
## ================
```

c

```
# equation 5.33 on slide 314
calc_ipw_star = function(data, regime, K){
  n = dim(data)[1]

  Y = data$Y
  L = cbind(data$L1, data$L2, data$L3, data$L4)
  # again need 0s becasue there are no side effects at the beginning
  S = cbind(rep(0, n), data$S2, data$S3, data$S4, data$S5)
  A = cbind(data$A1, data$A2, data$A3, data$A4)

  cd = calc_cd(data, regime, K)
  gamma = calc_gamma(data)
```

```r
  sum1 = 0
  sum2 = 0

  for(i in 1:n){
    num = cd[i]

    # only need to calculate if Cd == 1
    if(cd[i]){

      denom = 1
      # calculate denominator
      # the product of the propensities in equation 7
      for(k in 1:K){
        val = gamma[k, 1] + gamma[k, 2] * L[i,k] + gamma[k, 3] * S[i,k]
        p = logistic_func(val)
        dk = regime(L[i,], S[i,], A[i,], k)
        denom = denom*(dk * p + (1-dk)*(1-p))
      }

      sum1 = sum1 + cd[i] / denom
      sum2 = sum2 + cd[i] * Y[i] / denom
    } # end if
  }

  return(sum2 / sum1)
}

bootstrap_ipw_star = function(data, regime, K, rep){
  out = NULL
  nrow = dim(data)[1]

  for(i in 1:rep){
    sample = data[sample(nrow, replace=TRUE),]
    ipw_star = calc_ipw_star(sample, regime, K)
    out = rbind(out, ipw_star)
  }
  return(sd(out))
}


# regime 1
ipw_star_est1 = calc_ipw_star(data = ldl, regime = stat_reg1, K = 4)
ipw_star_sd1 = bootstrap_ipw_star(data = ldl, regime = stat_reg1, K = 4, rep = 100)

cat("================\nregime 1\nipw_star_estimate:\t", ipw_star_est1, "\nstderr:\t\t\t", ipw_star_sd1,
```

```
## ================
## regime 1
## ipw_star_estimate:    143.8031
## stderr:           1.583392
## ================
```

```r
# regime 2
ipw_star_est2 = calc_ipw_star(data = ldl, regime = stat_reg2, K = 4)
ipw_star_sd2 = bootstrap_ipw_star(data = ldl, regime = stat_reg2, K = 4, rep = 100)

cat("================\nregime 2\nipw_star_estimate:\t", ipw_star_est2, "\nstderr:\t\t\t", ipw_star_sd2,
```

```
## ================
## regime 2
## ipw_star_estimate:    135.6302
## stderr:              1.464978
## ================
```

```r
# regime 3
ipw_star_est3 = calc_ipw_star(data = ldl, regime = stat_reg3, K = 4)
ipw_star_sd3 = bootstrap_ipw_star(data = ldl, regime = stat_reg3, K = 4, rep = 100)

cat("================\nregime 3\nipw_star_estimate:\t", ipw_star_est3, "\nstderr:\t\t\t", ipw_star_sd3,
```

```
## ================
## regime 3
## ipw_star_estimate:    124.2947
## stderr:              1.49693
## ================
```

```r
# regime 4
ipw_star_est4 = calc_ipw_star(data = ldl, regime = stat_reg4, K = 4)
ipw_star_sd4 = bootstrap_ipw_star(data = ldl, regime = stat_reg4, K = 4, rep = 100)

cat("================\nregime 4\nipw_star_estimate:\t", ipw_star_est4, "\nstderr:\t\t\t", ipw_star_sd4,
```

```
## ================
## regime 4
## ipw_star_estimate:    115.0639
## stderr:              1.608884
## ================
```

```r
# regime 5
ipw_star_est5 = calc_ipw_star(data = ldl, regime = stat_reg5, K = 4)
ipw_star_sd5 = bootstrap_ipw_star(data = ldl, regime = stat_reg5, K = 4, rep = 100)

cat("================\nregime 5\nipw_star_estimate:\t", ipw_star_est5, "\nstderr:\t\t\t", ipw_star_sd5,
```

```
## ================
## regime 5
## ipw_star_estimate:    106.0979
## stderr:              1.726447
## ================
```

```r
# regime 6
ipw_star_est6 = calc_ipw_star(data = ldl, regime = stat_reg6, K = 4)
ipw_star_sd6 = bootstrap_ipw_star(data = ldl, regime = stat_reg6, K = 4, rep = 100)

cat("================\nregime 6\nipw_star_estimate:\t", ipw_star_est6, "\nstderr:\t\t\t", ipw_star_sd6,
```

```
## ================
## regime 6
## ipw_star_estimate:    113.8119
## stderr:           1.274456
## ================
```

```r
# regime 7
ipw_star_est7 = calc_ipw_star(data = ldl, regime = stat_reg7, K = 4)
ipw_star_sd7 = bootstrap_ipw_star(data = ldl, regime = stat_reg7, K = 4, rep = 100)

cat("================\nregime 7\nipw_star_estimate:\t", ipw_star_est7, "\nstderr:\t\t\t", ipw_star_sd7,
```

```
## ================
## regime 7
## ipw_star_estimate:    123.1004
## stderr:           1.239523
## ================
```

```r
# regime 8
ipw_star_est8 = calc_ipw_star(data = ldl, regime = stat_reg8, K = 4)
ipw_star_sd8 = bootstrap_ipw_star(data = ldl, regime = stat_reg8, K = 4, rep = 100)

cat("================\nregime 8\nipw_star_estimate:\t", ipw_star_est8, "\nstderr:\t\t\t", ipw_star_sd8,
```

```
## ================
## regime 8
## ipw_star_estimate:    133.9061
## stderr:           1.335779
## ================
```

```r
# d1
ipw_star_estd1 = calc_ipw_star(data = ldl, regime = regime_d2, K = 4)
ipw_star_sdd1 = bootstrap_ipw_star(data = ldl, regime = regime_d2, K = 4,rep = 100)

cat("================\nregime d1\nipw_star_estimate:\t", ipw_star_estd1, "\nstderr:\t\t\t", ipw_star_sd
```

```
## ================
## regime d1
## ipw_star_estimate:    115.0639
## stderr:           1.491497
## ================
```

```r
ipw_star_estd2 = calc_ipw_star(data = ldl, regime = regime_d2, K = 4)
ipw_star_sdd2 = bootstrap_ipw_star(data = ldl, regime = regime_d2, K = 4, rep = 100)

cat("================\nregime d2\nipw_star_estimate:\t", ipw_star_estd2, "\nstderr:\t\t\t", ipw_star_sd
```

```
## ================
## regime d2
## ipw_star_estimate:    115.0639
## stderr:           1.627002
## ================
```

```r
etas = seq(90, 200, 10)

for(i in 1:length(etas)){
  eta_i = etas[i]

  regime_eta = function(L, S, A, dk){
    return(S[dk] == 0 && L[dk] > eta_i)
  }

  ipw_star_estd2 = calc_ipw_star(data = ldl, regime = regime_d2, K = 4)
  ipw_star_sdd2 = bootstrap_ipw_star(data = ldl, regime = regime_d2, K = 4, rep = 100)

  cat("================\nregime eta=", eta_i,"\nipw_star_estimate:\t", ipw_star_estd2, "\nstderr:\t\t\t
}
```

```
## ================
## regime eta= 90
## ipw_star_estimate:    115.0639
## stderr:           1.715738
## ================================
## regime eta= 100
## ipw_star_estimate:    115.0639
## stderr:           1.548604
## ================================
## regime eta= 110
## ipw_star_estimate:    115.0639
## stderr:           1.741786
## ================================
## regime eta= 120
## ipw_star_estimate:    115.0639
## stderr:           1.711195
## ================================
## regime eta= 130
## ipw_star_estimate:    115.0639
## stderr:           1.627289
## ================================
## regime eta= 140
## ipw_star_estimate:    115.0639
## stderr:           1.476964
## ================================
## regime eta= 150
## ipw_star_estimate:    115.0639
## stderr:           1.855413
## ================================
## regime eta= 160
## ipw_star_estimate:    115.0639
## stderr:           1.636021
## ================================
## regime eta= 170
## ipw_star_estimate:    115.0639
## stderr:           1.539118
## ================================
## regime eta= 180
```

```
## ipw_star_estimate:    115.0639
## stderr:           1.6557
## ================================
## regime eta= 190
## ipw_star_estimate:    115.0639
## stderr:           1.525308
## ================================
## regime eta= 200
## ipw_star_estimate:    115.0639
## stderr:           1.701649
## ================
```

d

The results for (b) and (c) are very similar to one another in both estimates and standard errors. These similarities held across all kinda of regimes. They show the same value pattern across static regimes as g-computation.

They had less disparity in value when between regimes d1 and d2.

They also had similar (well, exactly the same) values across all eta choices.