Data Analysis Homework 4

Jimmy Hickey

11/10/2020

1

```
knitr::opts_chunk$set(echo = TRUE)
library(DynTxRegime)
```

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

```
library(rgenoud)
```

```
## ##  rgenoud (Version 5.8-3.0, Build Date: 2019-01-22)
## ##  See http://sekhon.berkeley.edu/rgenoud for additional documentation.
## ##  Please cite software as:
## ##   Walter Mebane, Jr. and Jasjeet S. Sekhon. 2011.
## ##   ''Genetic Optimization Using Derivatives: The rgenoud package for R.''
## ##   Journal of Statistical Software, 42(11): 1-26.
## ##
```

```
library(modelObj)
library(rpart)
library(xtable)
```

```
## Warning: package 'xtable' was built under R version 4.0.3
```

a

```
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")


fSet4 = function(data){
           subsets = list(list("S41",0L),
                          list("S42",c(0L,1L)))
```

```
          txOpts = rep(x = 'S42', times = nrow(x = data))
          txOpts[data$S4 == 1L] = "S41"

          return(list("subsets" = subsets, "txOpts" = txOpts))
}

moPropen_S41 = buildModelObjSubset(model = ~ 1,
                                   solver.method = 'glm',
                                   solver.args = list("family"='binomial'),
                                   subset = 'S41',
                                   dp = 4L,
                                   predict.method = 'predict.glm',
                                   predict.args = list("type"='response'))
moPropen_S42 = buildModelObjSubset(model = ~ L4,
                                   solver.method = 'glm',
                                   solver.args = list("family"='binomial'),
                                   subset = 'S42',
                                   dp = 4L,
                                   predict.method = 'predict.glm',
                                   predict.args = list("type"='response'))


bowl4 = bowl(moPropen = list(moPropen_S41, moPropen_S42),
                             data = ldl,
                             txName = 'A4',
                             regime = ~ L4,
                             response = -ldl$Y,
                             BOWLObj = NULL,
                             lambdas = 0.01,
                             kernel = 'linear',
                             kparam = NULL,
                             fSet = fSet4,
                             surrogate = 'sqhinge',
                             verbose = 0L)
```

```
## NOTE: subset(s) S41 received tx not in accordance with specified feasible tx sets
```

```
####################
# Decision 3
##################


fSet3 = function(data){
          subsets = list(list("S31",0L),
                         list("S32",c(0L,1L)))

          txOpts = rep(x = 'S32', times = nrow(x = data))
          txOpts[data$S3 == 1L] = "S31"

          return(list("subsets" = subsets, "txOpts" = txOpts))
}

moPropen_S31 = buildModelObjSubset(model = ~ 1,
```

```
                                solver.method = 'glm',
                                solver.args = list("family"='binomial'),
                                subset = 'S31',
                                dp = 3L,
                                predict.method = 'predict.glm',
                                predict.args = list("type"='response'))
moPropen_S32 = buildModelObjSubset(model = ~ L3,
                                solver.method = 'glm',
                                solver.args = list("family"='binomial'),
                                subset = 'S32',
                                dp = 3L,
                                predict.method = 'predict.glm',
                                predict.args = list("type"='response'))


bowl3 = bowl(moPropen = list(moPropen_S31, moPropen_S32),
                            data = ldl,
                            txName = 'A3',
                            regime = ~ L3,
                            response = rep(0,nrow(ldl)),
                            BOWLObj = bowl4,
                            lambdas = 0.01,
                            kernel = 'linear',
                            kparam = NULL,
                            fSet = fSet3,
                            surrogate = 'sqhinge',
                            verbose = FALSE)
```

```
## NOTE: subset(s) S31 received tx not in accordance with specified feasible tx sets
```

```
#################
# decision 2
#################


fSet2 = function(data){
           subsets = list(list("S21",0L),
                          list("S22",c(0L,1L)))

           txOpts = rep(x = 'S22', times = nrow(x = data))
           txOpts[data$S2 == 1L] = "S21"

           return(list("subsets" = subsets, "txOpts" = txOpts))
}

moPropen_S21 = buildModelObjSubset(model = ~ 1,
                                solver.method = 'glm',
                                solver.args = list("family"='binomial'),
                                subset = 'S21',
                                dp = 2L,
                                predict.method = 'predict.glm',
                                predict.args = list(type='response'))
```

```
moPropen_S22 = buildModelObjSubset(model = ~ L2,
                                   solver.method = 'glm',
                                   solver.args = list("family"='binomial'),
                                   subset = 'S22',
                                   dp = 2L,
                                   predict.method = 'predict.glm',
                                   predict.args = list(type='response'))


bowl2 = bowl(moPropen = list(moPropen_S21, moPropen_S22),
                               data = ldl,
                               txName = 'A2',
                               regime = ~ L2,
                               response = rep(0,nrow(ldl)),
                               BOWLObj = bowl3,
                               lambdas = 0.01,
                               kernel = 'linear',
                               kparam = NULL,
                               fSet = fSet2,
                               surrogate = 'sqhinge',
                               verbose = FALSE)
```

```
## NOTE: subset(s) S21 received tx not in accordance with specified feasible tx sets
```

```
###############
# decision 1
###############

fSet1 = function(data){
           subsets = list(list("S1",c(0L,1L)))
           txOpts = rep(x = 'S1', times = nrow(x = data))
           return(list("subsets" = subsets, "txOpts" = txOpts))
}

moPropen_S1 = buildModelObjSubset(model = ~  L1,
                                   solver.method = 'glm',
                                   solver.args = list("family"='binomial'),
                                   subset = 'S1',
                                   dp = 1L,
                                   predict.method = 'predict.glm',
                                   predict.args = list(type='response'))


bowl1 = bowl(moPropen = moPropen_S1,
                               data = ldl,
                               txName = 'A1',
                               regime = ~ L1,
                               response = rep(0,nrow(ldl)),
                               BOWLObj = bowl2,
                               lambdas = 0.01,
                               kernel = 'linear',
                               kparam = NULL,
                               fSet = fSet1,
```

```
                          surrogate = 'sqhinge',
                          verbose = FALSE)
```

b

```
-regimeCoef(bowl1)[1] / regimeCoef(bowl1)[2]
```

```
## [1] 177.5283
```

```
-regimeCoef(bowl2)[1] / regimeCoef(bowl2)[2]
```

```
## [1] 206.6368
```

```
-regimeCoef(bowl3)[1] / regimeCoef(bowl3)[2]
```

```
## [1] 96.11033
```

```
-regimeCoef(bowl4)[1] / regimeCoef(bowl4)[2]
```

```
## [1] 296.163
```

```
estimator(bowl1)
```

```
## [1] -96.17764
```

This gives the following treatment rules.

$$
\begin{aligned}
&\widehat{d}_1^{opt} I(L1M177.5022) \\
&\widehat{d}_2^{opt} = I(L2 < 206.0815) I(S2 = 0) \\
&\widehat{d}_3^{opt} = I(L3 > 96.11033) I(S3 = 0) \\
&\widehat{d}_4^{opt} = I(L4 < 296.163) I(S4 = 0)
\end{aligned}
$$

Here we get a value $\widehat{\mathcal{V}}(d^{opt}) = 96.17531$. This is smaller than 103.6736 that I got from question 2 of homework 3.

2

```
smart = read.table("SMART.dat.txt", header=FALSE)
names(smart) = c("X11", "X12", "X13", "A1",
                 "X21", "X22", "R2", "A2", "Y")

n = nrow(smart)

# e1, trt 0 -> trt 0
cd1 = (smart$A1 == 0 ) * (smart$A2 == 0)
# e2 trt 0 -> trt 0 or trt 1
cd2 = (smart$A1 == 0 ) * (smart$A2 != smart$R2)
# e3 trt1 -> trt 1
cd3 = (smart$A1 == 1 ) * (smart$A2 == 1)
# e4 trt1 -> trt 0 or trt 1
cd4 = (smart$A1 == 1 ) * (smart$A2 == smart$R2)
```

a

i

```
aipw = function(data, cd){
  W = 4 / (data$R2 + 1)
  # from part b of analytical

  est = sum( cd * W * data$Y) / sum(cd * W)

  # from part c of analytical

  var = mean( cd * W^2 * (data$Y - est)^2 )

  se = sqrt(1/nrow(data) * var)
  return( list(est = est, se = se))
}



aipw_r1 = aipw(smart, cd1)
aipw_r2 = aipw(smart, cd2)
aipw_r3 = aipw(smart, cd3)
aipw_r4 = aipw(smart, cd4)
```

ii

```
library(geepack)
```

```
## Warning: package 'geepack' was built under R version 4.0.3
```

```r
# weighted least squares fit
WLS = function(data, cd){
  W = 4 / (data$R2 + 1)

  # from problem
  wlsfit = geese(Y ~ cd, family=gaussian, data=data, weights=W,
                 corstr="independence",id=1:nrow(data))

  # sum for alpha_0 + alpha_1
  est = sum(wlsfit$beta)

  # lambda^T * Var(beta) * lambda
  se = sqrt(c(1,1) %*% wlsfit$vbeta %*% c(1,1))

  return(list(est = est, se = se))
}

wls_r1 = WLS(smart, cd1)
wls_r2 = WLS(smart, cd2)
wls_r3 = WLS(smart, cd3)
wls_r4 = WLS(smart, cd4)
```

iii

```r
# aipw with estimated propensities
aipw_est = function(data, cd){

    prop1 = predict(glm(A1~1,
                        data = data,
                        family = "binomial"),
                        type = "response")[1]

  # 0 -> 0
  prop20 =  predict(glm(A2~1,
                    data=data[(data$A1==0)&(data$R2==0),],
                    family='binomial'),
                    type='response')[1]
  # 1 -> 0
  prop21 =  predict(glm(A2~1,
                    data=data[(data$A1==1)&(data$R2==0),],
                    family='binomial'),
                    type='response')[1]


  omega1 = prop1 * data$A1 + (1-prop1)*(1-data$A1)
  omega2 = data$A1 *
             (data$R2* data$A2 +
             (1 - data$R2)*(prop21*data$A2 + (1-prop21)*(1-data$A2))) +
           (1 - data$A1) *
             (data$R2 * (1-data$A2) +
                 (1 - data$R2) * (prop20 * data$A2 + (1-prop20)*(1-data$A2)) )
  W = 1/(omega1 * omega2)
```

```
  est = sum( cd * W * data$Y) / sum(cd * W)

  # from part c of analytical

  var = mean( cd * W^2 * (data$Y - est)^2 )

  se = sqrt( 1 / nrow(data) * var)
  return( list(est = est, se = se))
}


aipw_est_r1 = aipw_est(smart, cd1)
aipw_est_r2 = aipw_est(smart, cd2)
aipw_est_r3 = aipw_est(smart, cd3)
aipw_est_r4 = aipw_est(smart, cd4)


a_df = data.frame( rbind(c(1, aipw_r1, wls_r1, aipw_est_r1),
                   c(2, aipw_r2, wls_r2, aipw_est_r2),
                   c(3, aipw_r3, wls_r3, aipw_est_r3),
                   c(4, aipw_r4, wls_r4, aipw_est_r4)))


names(a_df) = c("regime",
                  "aipw_est",
                  "aipw_se",
                  "wls_est",
                  "wls_se",
                  "aipw2_est",
                  "aip2_se")
```

regime	est 1	sd 1	est 2	sd 2	est 3	sd 3
1	83.86	1.23	83.86	1.18	84.03	1.189125
2	79.77	1.27	79.77	1.13	79.72	1.114411
3	79.44	1.08	79.44	1.16	79.38	1.155281
4	80.34	1.34	80.34	1.48	80.41	1.488151

Notice that the first estimates are the same as those from the second method. This matches the theory proven in the analytical section. Also, as expected, they have different standard errors.

b

```
calc_p = function(est_se1, est_se2){
  variance = est_se1$se^2+est_se2$se^2

  test_stat = (est_se1$est-est_se2$est)/sqrt(variance)

  pval = 2 * (1 - pnorm(abs(test_stat)))

  return(list(test_stat, pval))
```

```
}
```

```
calc_p(aipw_r1, aipw_r3)
```

```
## [[1]]
## [1] 2.698345
##
## [[2]]
## [1] 0.006968518
```

```
calc_p(wls_r1, wls_r3)
```

```
## [[1]]
##          [,1]
## [1,] 2.669333
##
## [[2]]
##              [,1]
## [1,] 0.007600214
```

```
calc_p(aipw_est_r1, aipw_est_r3)
```

```
## [[1]]
## [1] 2.803768
##
## [[2]]
## [1] 0.005050927
```

method	test stat	pval
1	2.698345	0.00697
2	2.669333	0.00760
3	2.803768	0.00505

In all cases we reject the null hypothesis at the level of 0.05. We are 95% confident that the value of the last burdensome regime is different from the most burdensome.

(Notice that the last test statistic is very large. This is again due to the very small standard errors.)

c

```
calc_p = function(est_se1, est_se2, data, cd1, cd2){
  W = 4 / (data$R2 + 1)

  var = est_se1$se^2 + est_se2$se^2 -
    2* mean(cd1 * W * cd2 * W *
          (data$Y - est_se1$est) * (data$Y - est_se2$est)) / nrow(data)
```

```
  test_stat = (est_se1$est - est_se2$est)/sqrt(var)

  pval = 2 * (1 - pnorm(abs(test_stat)))

  return(c(test_stat, pval))

}

calc_p(aipw_r1, aipw_r3, smart, cd1, cd2)
```

```
## [1] 2.936431776 0.003320118
```

```
calc_p(wls_r1, wls_r3, smart, cd1, cd2)
```

```
## [1] 2.899152581 0.003741728
```

```
calc_p(aipw_est_r1, aipw_est_r3, smart, cd1, cd2)
```

```
## [1] 3.04372964 0.00233665
```

method	test stat	pval
1	2.93643	0.00332
2	2.89915	0.00374
3	3.04372	0.00233

In all cases we reject the null hypothesis at the level of 0.05. We are 95% confident that the value of the regime starting with trt 0 that switches nonresponders to trt 1 is different than keeping nonresponders on trt 0.

d

i

```
###################
# decision 2
###################



fSet2 <- function(data){
  subsets <- list(list("S20",c(0L)),
                  list("S21",c(1L)),
                  list("S22",c(0L,1L)))

  txOpts <- rep(x = 'S22', times = nrow(x = data))
  txOpts[data$A1 == 0L & data$R2 == 1L] <- "S20"
  txOpts[data$A1 == 1L & data$R2 == 1L] <- "S21"
```

```
  return(list("subsets" = subsets, "txOpts" = txOpts))
}

moCont_S22 = buildModelObjSubset(model = ~ A1,
                                 solver.method = "lm",
                                 subset= "S22",
                                 dp = 2L)


moMain_S22 = buildModelObjSubset(model = ~ A1,
                                 solver.method = "lm",
                                 subset= "S22",
                                 dp = 2L)

# remember negative response to minimize
q2 = qLearn(moMain = moMain_S22,
            moCont = moCont_S22,
            data = smart,
            response = -smart$Y,
            txName = 'A2',
            fSet = fSet2)
```

```
## First step of the Q-Learning Algorithm.
##
## Subsets of treatment identified as:
## $S20
## [1] 0
##
## $S21
## [1] 1
##
## $S22
## [1] 0 1
##
## Number of patients in data for each subset:
## S20 S21 S22
##  32  28  90
##
## Outcome regression.

## subset(s) S20, S21 excluded from outcome regression

## Fitting models for  S22 using 90 patient records.
## Regression analysis for Combined:
##
## Call:
## lm(formula = YinternalY ~ A1 + A2 + A1:A2, data = data)
##
## Coefficients:
## (Intercept)           A1           A2        A1:A2
##     -88.478        3.078        7.132       -5.399
##
```

```
##
## Recommended Treatments:
##   0   1
##  32 118
##
## Estimated value: -79.56197
```

```
##################
# decision 1
#################

moMain_1 = buildModelObj(model = ~1,
                         solver.method = 'lm')


moCont_1 = buildModelObj(model = ~1,
                         solver.method = 'lm')


q1 = qLearn(moMain = moMain_1,
            moCont = moCont_1,
            data = smart,
            response = q2,
            txName = "A1",
            fSet = NULL)
```

```
## Step 2 of the Q-Learning Algorithm.
##
## Outcome regression.
## Combined outcome regression model: ~  + A1 .
## Regression analysis for Combined:
##
## Call:
## lm(formula = YinternalY ~ A1, data = data)
##
## Coefficients:
## (Intercept)           A1
##    -79.7156       0.3339
##
##
## Recommended Treatments:
##   1
## 150
##
## Estimated value: -79.38164
```

We can now use our Q functions and estimates to estimate our optimal regime.

$$
\begin{aligned}
d_1^{opt} &= I(\beta_{12} > 0) \\
\widehat{d}_1^{opt} &= I(0.3339) \\
&= 1
\end{aligned}
$$

$$
\begin{aligned}
d_2^{opt} &= I(\beta_{23} + \beta_2 4 a_1 > 0) \\
&= I(7.123 - 5.399 a_1 > 0) \\
&= I(a_1 < 1.32) \\
&= 1
\end{aligned}
$$

Thus our optimal regime is (1,1) with value 79.38164.

ii

```
moPropen_2 = buildModelObjSubset(model = ~ 1,
                solver.method = 'glm',
                solver.args = list("family"='binomial'),
                subset = 'S22',
                dp = 2L,
                predict.method = 'predict.glm',
                predict.args = list(type='response'))



bowl2 = bowl(moPropen = moPropen_2,
               data = smart,
               txName = 'A2',
               regime = ~ A1,
               response = as.vector(x = -smart$Y),
               BOWLObj = NULL,
               lambdas = 10.0^{seq(from = -4, to = -1, by = 1)},
               cvFolds = 10L,
               kernel = 'linear',
               kparam = NULL,
               fSet = fSet2,
               surrogate = 'sqhinge',
               verbose = 0L)
```

```
## subset(s) S20, S21 excluded from propensity regression
```

```
fSet1 = function(data){
  subsets = list(list("S1", c(0, 1)))

  txOpts = rep("S1", nrow(data))

  return(list("subsets" = subsets, "txOpts" = txOpts))
}
```

```
moPropen_1 = buildModelObjSubset(model = ~ 1,
               solver.method = 'glm',
               solver.args = list("family"='binomial'),
               subset = 'S1',
               dp = 1L,
               predict.method = 'predict.glm',
               predict.args = list(type='response'))



bowl1 = bowl(moPropen = moPropen_1,
             data = smart,
            txName = 'A1',
            regime = ~ 1,
            response = rep(0,nrow(smart)),
            BOWLObj = bowl2,
            lambdas = 0.01,
            kernel = 'linear',
            kparam = NULL,
            fSet = fSet1,
            surrogate = 'sqhinge',
            verbose = FALSE)


regimeCoef(bowl2)
```

```
## [1] 0.025188916 0.005133172
```

```
regimeCoef(bowl1)
```

```
## [1] 0.01308159
```

```
estimator(bowl1)
```

```
## [1] -78.42707
```

We can use our given decision functions with our estimates to estimate an optmial regime.

$$
\begin{aligned}
d_1^{opt} &= I(\eta_{11} > 0) \\
\widehat{d}_1^{opt} &= I(0.01301 > 0) \\
&= 1
\end{aligned}
$$

$$
\begin{aligned}
d_2^{opt} &= I(\eta_{21} + \eta_{22} a_1 > 0) \\
&= I(0.02519 + 0.0051 a_1 > 0) \\
&= I(a_1 > -4.9392) \\
&= 1
\end{aligned}
$$

Again, we get an optimal treatment of (1,1). but now with a smaller value of 78.42707.

Notice that the values from part (d) are similar to part (a).

e

i

```
#################
# decision 2
#################
moMain_S22 = buildModelObjSubset(model = ~ X11 + X12 + X13 + A1 + X21 + X22,
                                 solver.method = 'lm',
                                 subset = "S22",
                                 dp = 2L,
                                 predict.method = 'predict.lm')

moCont_S22 = buildModelObjSubset(model = ~ X11 + X12 + X13 + A1 + X21 + X22,
                                 solver.method = 'lm',
                                 subset = "S22",
                                 dp = 2L,
                                 predict.method = 'predict.lm')


q2 = qLearn(moMain = moMain_S22,
            moCont = moCont_S22,
            data = smart,
            response = - smart$Y,
            txName = 'A2',
            fSet = fSet2,
            verbose = FALSE)
```

```
## subset(s) S20, S21 excluded from outcome regression
```

```
##################
# decision 1
##################

moMain_1 = buildModelObj(model = ~ X11 + X12 + X13,
                             solver.method = 'lm',
                             predict.method = 'predict.lm')

moCont_1 = buildModelObj(model = ~ X11 + X12 + X13,
                             solver.method = 'lm',
                             predict.method = 'predict.lm')

q1 = qLearn(moMain = moMain_1,
            moCont = moCont_1,
            data = smart,
            response = q2,
            txName = 'A2',
            fSet = NULL,
            verbose = FALSE)



coef(q2)
```

```
## $outcome
## $outcome$`Subset=S22`
## $outcome$`Subset=S22`$Combined
##   (Intercept)           X11           X12           X13            A1
##  2.1092973152  4.3219712692  0.0520324431 -0.3136207596  3.2461373783
##           X21           X22            A2        X11:A2        X12:A2
## -0.7487432169  3.1101780055 21.7351043351  0.1341992574 -0.1794813675
##        X13:A2         A1:A2        X21:A2        X22:A2
## -0.0948252853 -4.3163313361  0.0008018799  3.1070691010
```

```
coef(q1)
```

```
## $outcome
## $outcome$Combined
##  (Intercept)          X11          X12          X13           A2       X11:A2
##  16.41323891   7.20638389  -0.06780442  -1.01998123 -46.36131396   3.16962287
##       X12:A2       X13:A2
##   0.09539764   0.44085960
```

```
estimator(q1)
```

```
## [1] -78.37626
```

We can now construct our optimal regime.

$$d_1^{opt} = I(\beta_{12}\widetilde{h}_1 > 0)$$
$$\widehat{d}_1^{opt} = I(3.1696X11 - 0.0678X12 + 0.09539X13 > 0)$$

$$d_2^{opt} = I(\beta_{22}\widetilde{h}_2 > 0)$$
$$\widehat{d}_2^{opt} = I(0.13420X11 - 0.17948X12 - 0.094825X13 - 4.31633A1 + 0.0008X21 + 3.10707X22 > 0)$$

With an estiamted value of 78.37626.

ii

```
bowl2 = bowl(moPropen = moPropen_2,
             data = smart,
             txName = 'A2',
             regime = ~ X11 + X12 + X13 + A1 + X21 + X22,
             response = -smart$Y,
             BOWLObj = NULL,
             lambdas = 0.01,
             kernel = 'linear',
             kparam = NULL,
             fSet = fSet2,
             surrogate = 'sqhinge',
             verbose = FALSE)
```

```
## subset(s) S20, S21 excluded from propensity regression
```

```
bowl1 = bowl(moPropen = moPropen_1,
             data = smart,
             txName = 'A1',
             regime = ~ X11 + X12 + X13,
             response = rep(0,n),
             BOWLObj = bowl2,
             lambdas = 0.01,
             kernel = 'linear',
             kparam = NULL,
             fSet = fSet1,
             surrogate = 'sqhinge',
             verbose = FALSE)

regimeCoef(bowl1)
```

```
## [1]  1.897454763 -0.094306449 -0.049030409  0.006687755
```

```
regimeCoef(bowl2)
```

```
## [1] -2.113922408  0.338604482  0.009737196  0.030971812 -0.019804679
## [6] -0.012951952 -0.239988660
```

```
estimator(bowl2)
```

```
## [1] -72.86428
```

```
estimator(bowl1)
```

```
## [1] -54.06177
```

We can now construct our optimal regime.

$$d_1^{opt} = I(eta_{11}^T \widetilde{h}_1 > 0)$$

$$\widehat{d}_1^{opt} = I(1.89745 - 0.09431X11 - 0.04903X12 + 0.00669X13 > 0)$$

$$d_2^{opt} = I(\eta_{21}^T \widetilde{h}_2 > 0)$$

$$\widehat{d}_2^{opt} = I(-2.1139 + 0.3386X11 + 0.00974X12 + 0.309718X13 - 0.019804A1 - 0.012952X21 - 0.23999X22 > 0)$$

With a value of 54.06177. This may be because of the sensitivity to propensity models. Also there are few individuals.

Recommended Treatments		
0	1	NA
36	62	52

f

i

```
regime1 = function(eta1, data){
  return( as.integer(data$X13 > eta1))
}


regime2 = function(eta2, data){
  return( (data$X21 > eta2) *
          (data$R2 == 0) +
            data$A1 * (data$R2 == 1))
}


domains = rbind(c(min(smart$X13)-1, max(smart$X13)+1),
                  c(min(smart$X21)-1, max(smart$X21)+1))
val0 = c(mean(smart$X13), mean(smart$X21))


IPW = optimalSeq(moPropen = list(moPropen_1, moPropen_2),
                 data = smart,
                 response = -smart$Y,
                 txName = c("A1", 'A2'),
                 fSet = list(fSet1, fSet2),
                 regimes = list(regime1, regime2),
                 Domains = domains,
                 pop.size = 1000L,
                 starting.values = val0,
                 verbose = FALSE)
```

```
## IPW estimator will be used

## subset(s) S20, S21 excluded from propensity regression
```

```
regimeCoef(IPW)
```

```
## $`dp=1`
##     eta1
## 94.68847
##
## $`dp=2`
##     eta2
## 97.80412
```

```
estimator(IPW)
```

```
## [1] -58.3431
```

With these η's, we can formulate our estimated decision rules.

$$d_1(h_1;\eta_1) = I(X13 > \eta_1)$$
$$\widehat{d}_1(h_1;\widehat{\eta}_1) = I(X13 > 94.55671)$$

$$d_2(d_2;\eta_2) = I(X13 > \eta_2)I(R2 = 0) + A1I(R2 = 1)$$
$$\widehat{d}_2(d_2;\widehat{\eta}_2) = I(X13 > 97.38361)I(R2 = 0) + A1I(R2 = 1)$$

With a value of 58.3431. This value may be small for the same reasons as before.

ii

```
moMain_2 = buildModelObj(model = ~ A1,
              solver.method = 'lm',
              predict.method = 'predict.lm')

moCont_2 = buildModelObj(model = ~ A1,
              solver.method = 'lm',
              predict.method = 'predict.lm')


AIPW = optimalSeq(moMain = list(moMain_1, moMain_2),
                  moCont = list(moCont_1, moCont_2),
                  moPropen = list(moPropen_1, moPropen_2),
                  data = smart,
                  response = -smart$Y,
                  txName = c("A1", 'A2'),
                  fSet = list(fSet1, fSet2),
                  regimes = list(regime1, regime2),
                  Domains = domains,
                  pop.size = 1000L,
                  starting.values = val0,
                  verbose = FALSE)
```

```
## subset(s) S20, S21 excluded from propensity regression
```

```
## NOTE: subset(s) S20, S21 excluded from outcome regression
```

```
regimeCoef(AIPW)
```

```
## $`dp=1`
##     eta1
## 97.43172
```

```
##
## $`dp=2`
##     eta2
## 82.70732
```

```
estimator(AIPW)
```

```
## [1] -77.89197
```

With these η's, we can formulate our estimated decision rules.

$$d_1(h_1;\eta_1) = I(X13 > \eta_1)$$
$$\widehat{d}_1(h_1;\widehat{\eta}_1) = I(X13 > 97.86032)$$

$$d_2(d_2;\eta_2) = I(X13 > \eta_2)I(R2 = 0) + A1I(R2 = 1)$$
$$\widehat{d}_2(d_2;\widehat{\eta}_2) = I(X13 > 82.38203)I(R2 = 0) + A1I(R2 = 1)$$

With a value of 77.89197.

The AIPW estimate is similar to that of qLearning while the IPW estimate is closer to the BOWL estimate.