COVID STEROID 2: HTE Analysis

Analysis

First the required R packages are loaded, as well as the data set. We also source a couple of functions to allow for running BART models with multiple MCMC chains (without needing multiple cores).

```
rm(list = ls())
library(BART)
library(caret)
library(rpart)
library(rpart.plot)
source("clusterfunctions.R")

# Load data from appropriate directory (edit as needed)
dat <- read.csv2("~/Downloads/synth_covid.csv")
```

Next we do a small amount of data cleaning/preparation.

```
# Clean up data variables types and remove the small amount of missing data
dat$resp_sup <- as.factor(dat$resp_sup)
dat$dead90 <- ifelse(dat$dead90 == TRUE, 1, 0)
dat <- dat[complete.cases(dat), ]

# Standardize continuous covariates
dat$age <- (dat$age - mean(dat$age)) / sd(dat$age)
dat$BL9_Weight <- (dat$BL9_Weight - mean(dat$BL9_Weight)) / sd(dat$BL9_Weight)

# Make datasets under each counterfactual
dat1 <- dat0 <- dat
dat1$allocation <- TRUE
dat0$allocation <- FALSE
```

Then we run a BART analysis focused on the binary mortality outcome.

```
# Create 10 folds of the data set for cross-validation
set.seed(60622)
folds <- createFolds(dat$dead90, k = 10, list = TRUE, returnTrain = FALSE)

# Initialize output matrices for prediction error from each model
cvoutput <- expand.grid(1:3, c(0.25, 0.5, 0.95), c(50, 200, 400), NA)
colnames(cvoutput) <- c("Power", "Base", "Ntrees", "CVMSE")
mse <- array(NA, dim = c(27, 10))

# Perform cross validation (may take >2 hours)
for (hp in 7:8) {
  for (i in 1:10) {

    # BART model
    bartmod <- lbart.cluster(x.train = dat[-folds[[i]], c(1, 4:13)],
                             y.train = dat$dead90[-folds[[i]]],
                             x.test = dat[folds[[i]], c(1, 4:13)],
                             power = cvoutput$Power[hp],
                             base = cvoutput$Base[hp],
                             ntree = cvoutput$Ntrees[hp], nchains = 4)
    bartmod$yhat.test.collapse <- apply(bartmod$yhat.test, 2, rbind)
    pred <- exp(colMeans(bartmod$yhat.test.collapse)) /
            (1 + exp(colMeans(bartmod$yhat.test.collapse)))
    mse[hp, i] <- mean((dat$dead90[folds[[i]]] - pred)^2)

  }

}

# Calculate 10-fold CV error for each hyperparameter combination
cvoutput$CVMSE <- rowMeans(mse)

# Fit final model under hyperparameters with minimum CV error
set.seed(60622)
bartmod1 <-
  lbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dead90,
                x.test = dat1[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE)])
bartmod0 <-
  lbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dead90,
                x.test = dat0[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE)])

# Collapse predictions across chains for certain calculations
bartmod1$yhat.train.collapse <- apply(bartmod1$yhat.train, 2, rbind)
bartmod1$yhat.test.collapse <- apply(bartmod1$yhat.test, 2, rbind)
bartmod0$yhat.train.collapse <- apply(bartmod0$yhat.train, 2, rbind)
bartmod0$yhat.test.collapse <- apply(bartmod0$yhat.test, 2, rbind)
```

Then conditional average treatment effects are estimated using the predictions under each counterfactual.

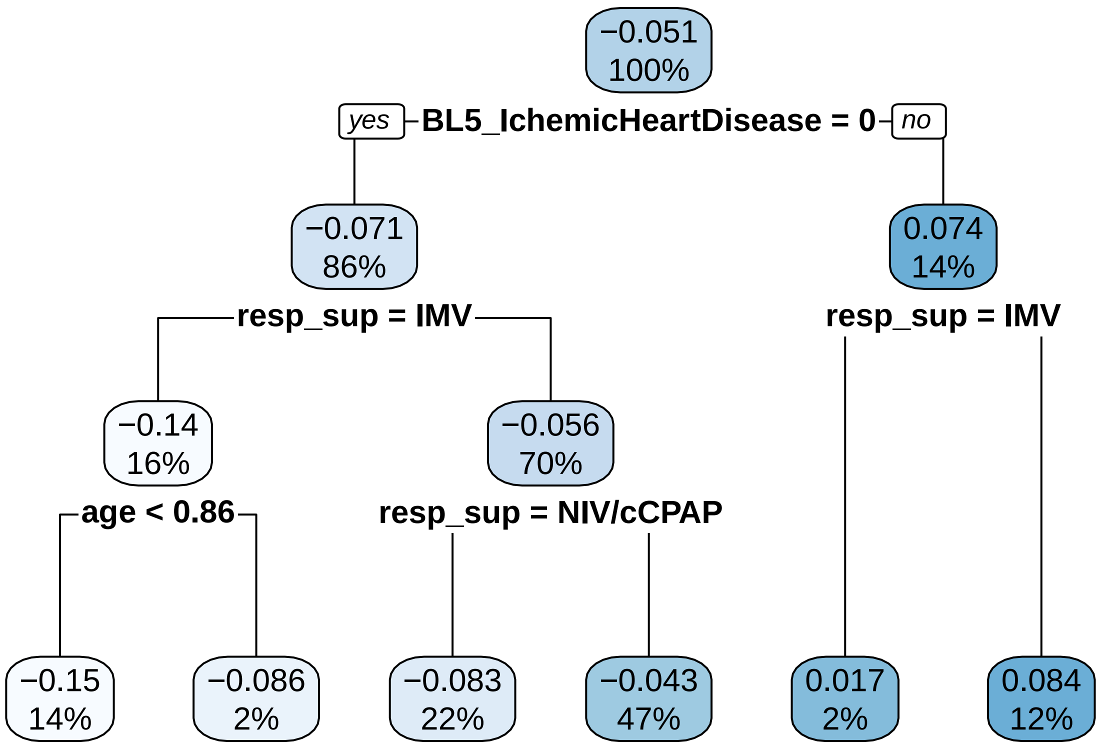
```
dat$cate <-
  exp(colMeans(bartmod1$yhat.test.collapse)) /
    (1 + exp(colMeans(bartmod1$yhat.test.collapse))) -
  exp(colMeans(bartmod0$yhat.test.collapse)) /
    (1 + exp(colMeans(bartmod0$yhat.test.collapse)))
```

This full process is then repeated for the continuous outcome (days alive without life support by day 90).

```r
# Initialize output for prediction error from each model
cvoutput$CVMSE_c <- NA
mse_c <- array(NA, dim = c(27, 10))

# Perform cross validation (should take much less time than the binary outcome)
set.seed(60622)
for (hp in 7:8) {

  for (i in 1:10) {

    # BART model
    bartmod_c <- wbart.cluster(x.train = dat[-folds[[i]], c(1, 4:13)],
                               y.train = dat$dawols90[-folds[[i]]],
                               x.test = dat[folds[[i]], c(1, 4:13)],
                               power = cvoutput$Power[hp],
                               base = cvoutput$Base[hp],
                               ntree = cvoutput$Ntrees[hp], nchains = 4)
    bartmod_c$yhat.test.collapse <- apply(bartmod_c$yhat.test, 2, rbind)
    pred_c <- colMeans(bartmod_c$yhat.test.collapse)
    mse_c[hp, i] <- mean((dat$dawols90[folds[[i]]] - pred_c)^2)

  }

}

# Calculate 10-fold CV error for each hyperparameter combination
cvoutput$CVMSE_c <- rowMeans(mse_c)

# Fit final models under hyperparameters with minimum CV error
set.seed(60622)
bartmod1_c <-
  wbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dawols90,
                x.test = dat1[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE_c)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE_c)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE_c)])
bartmod0_c <-
  wbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dawols90,
                x.test = dat0[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE_c)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE_c)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE_c)])

# Collapse predictions across chains for certain calculations
bartmod1_c$yhat.train.collapse <- apply(bartmod1_c$yhat.train, 2, rbind)
bartmod1_c$yhat.test.collapse <- apply(bartmod1_c$yhat.test, 2, rbind)
bartmod0_c$yhat.train.collapse <- apply(bartmod0_c$yhat.train, 2, rbind)
bartmod0_c$yhat.test.collapse <- apply(bartmod0_c$yhat.test, 2, rbind)

# Estimate CATEs
dat$cate_c <- colMeans(bartmod1_c$yhat.test.collapse) -
              colMeans(bartmod0_c$yhat.test.collapse)
```

Finally, the "fit-the-fit" approach is used to find subgroups exhibiting heterogeneity of treatment effect, starting with the binary outcome. In particular, a CART model is fit with the CATE for 90-day mortality as the outcome and the covariates as possible predictors. The model is first fit under default CART hyperparameter settings.

```
# CART model for 90 day mortality with default CART hyperparameter and
# all covariates considered
cartmod <- rpart(cate ~ ., data = dat[, c(4:14)], method = "anova")
rpart.plot(cartmod)
```



Now we prune the tree for interpretability using the stepwise approach of Hu et al. In particular, covariates are added to the CART model sequentially according to greatest increase in model R^2 until an increase of less than 1% occurs.

```
# Pruned tree using stepwise approach of Hu et al.
r2 <- c()
covar_include <- c()
covar_cols <- 4:13
currentr2 <- 0
maxcovars <- 4

for (numcovars in 1:maxcovars) {

  for (covar in covar_cols) {
    cartmod <- rpart(cate ~ ., data = dat[, c(covar, covar_include, 14)],
                     method = "anova")
    r2[which(covar_cols == covar)] <-
      1 - printcp(cartmod)[dim(printcp(cartmod))[1], 3]
  }

  if ((max(r2) - currentr2) / currentr2 > 0.01) {
    covar_include <- c(covar_include, covar_cols[which.max(r2)])
    covar_cols <- covar_cols[covar_cols != covar_cols[which.max(r2)]]
    currentr2 <- max(r2)
    r2 <- c()
  } else {
    break
  }

}
```

Print out the number of covariates selected and the resulting CART model output.

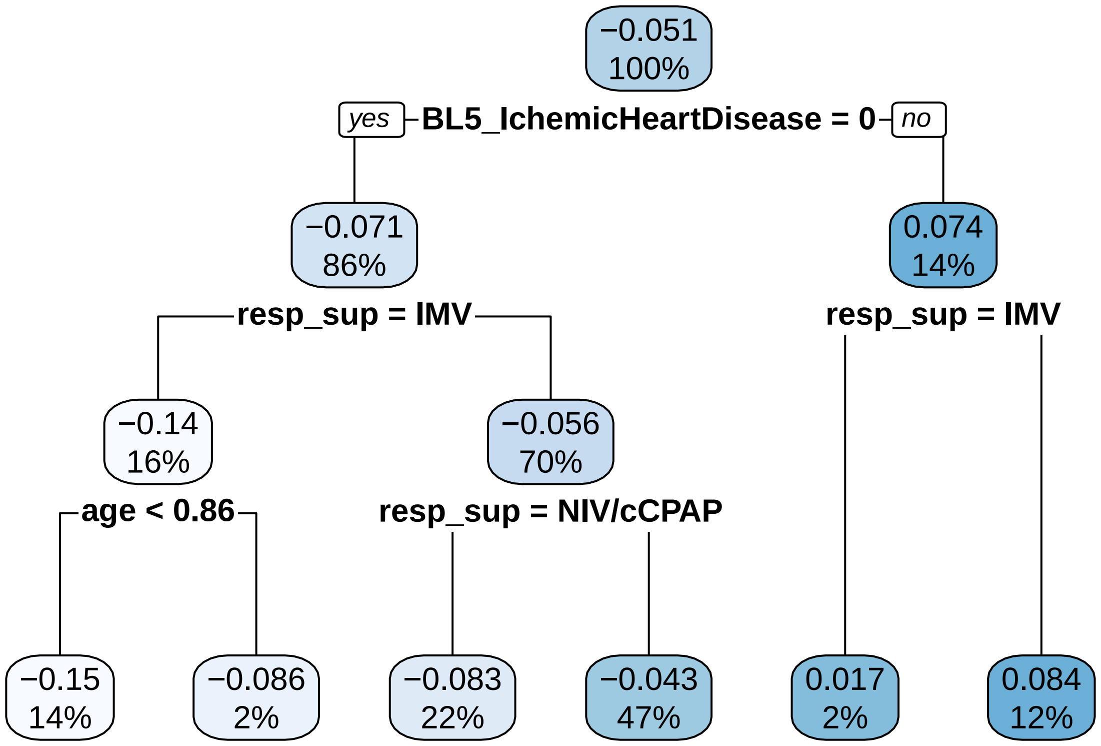
```
print("Number of covariates:")
```

```
## [1] "Number of covariates:"
```

```
print(numcovars - 1)
```

```
## [1] 3
```

```
cartmod <- rpart(cate ~ ., data = dat[, c(covar_include, 14)],
                     method = "anova")
rpart.plot(cartmod)
```

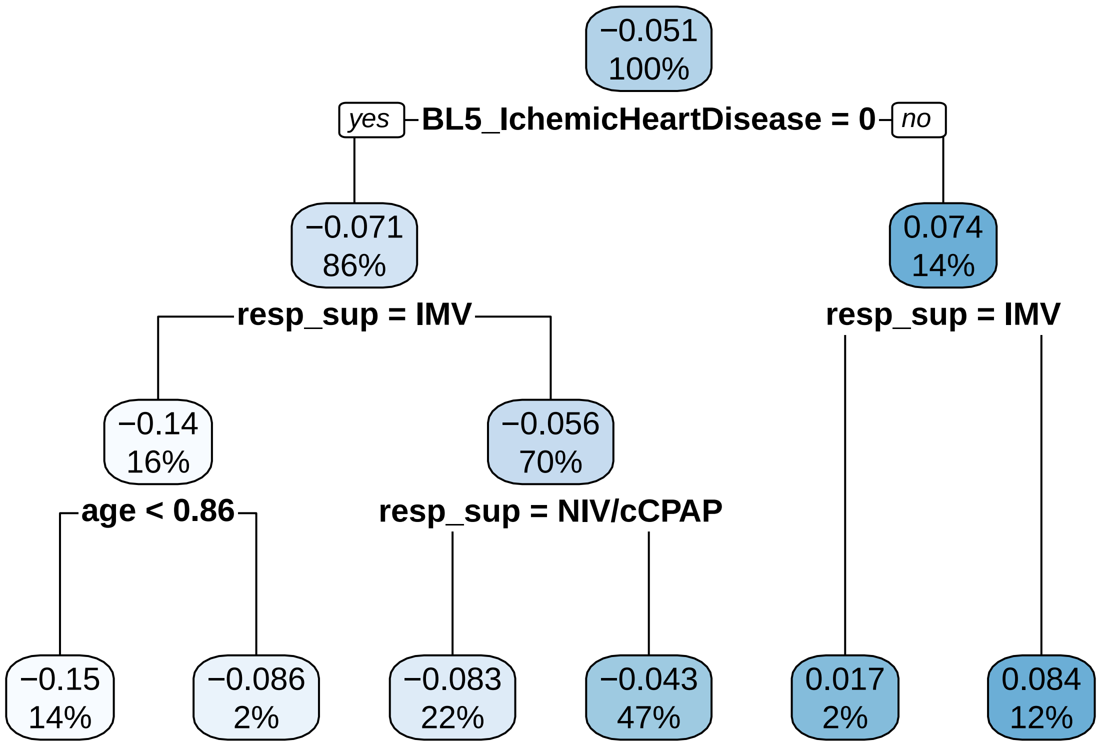


Finally we explore further pruning:

```
printcp(cartmod)
```

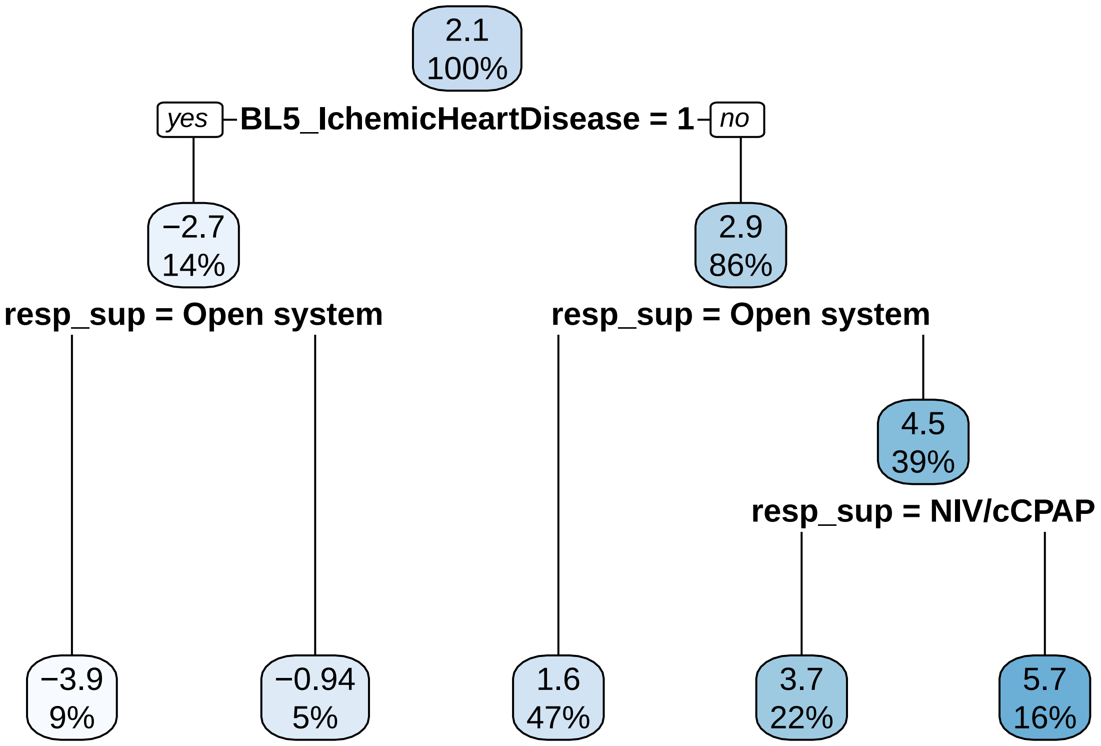
```
##
## Regression tree:
## rpart(formula = cate ~ ., data = dat[, c(covar_include, 14)],
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] age                     BL5_IchemicHeartDisease resp_sup
##
## Root node error: 4.4107/968 = 0.0045565
##
## n= 968
##
##         CP nsplit rel error  xerror      xstd
## 1 0.566291      0   1.00000 1.00112 0.0506063
## 2 0.189915      1   0.43371 0.43539 0.0194449
## 3 0.052782      2   0.24379 0.25408 0.0107530
## 4 0.018894      3   0.19101 0.19349 0.0088925
## 5 0.014260      4   0.17212 0.17425 0.0081266
## 6 0.010000      5   0.15786 0.16441 0.0076384
```

```
prunedtree <-
  prune(cartmod,
        cp = cartmod$cptable[which.min(cartmod$cptable[, "xerror"]), "CP"])
rpart.plot(prunedtree)
```



Next the same fit-the-fit approach is used to summarize the results for the continuous outcome, starting with a CART model using all covariates and default hyperparameter.

```
# CART model for 90 day mortality with default CART hyperparameter and
# all covariates considered
cartmod <- rpart(cate_c ~ ., data = dat[, c(4:13, 15)], method = "anova")
rpart.plot(cartmod)
```



Now we prune the tree for interpretability using the stepwise approach of Hu et al.

```
# Pruned tree using stepwise approach of Hu et al.
r2 <- c()
covar_include <- c()
covar_cols <- 4:13
currentr2 <- 0
maxcovars <- 4

for (numcovars in 1:maxcovars) {

  for (covar in covar_cols) {
    cartmod <- rpart(cate_c ~ ., data = dat[, c(covar, covar_include, 15)],
                     method = "anova")
    r2[which(covar_cols == covar)] <-
      1 - printcp(cartmod)[dim(printcp(cartmod))[1], 3]
  }

  if ((max(r2) - currentr2) / currentr2 > 0.01) {
    covar_include <- c(covar_include, covar_cols[which.max(r2)])
    covar_cols <- covar_cols[covar_cols != covar_cols[which.max(r2)]]
    currentr2 <- max(r2)
    r2 <- c()
  } else {
    break
  }
}
```

Print out the number of covariates selected and the resulting CART model output.

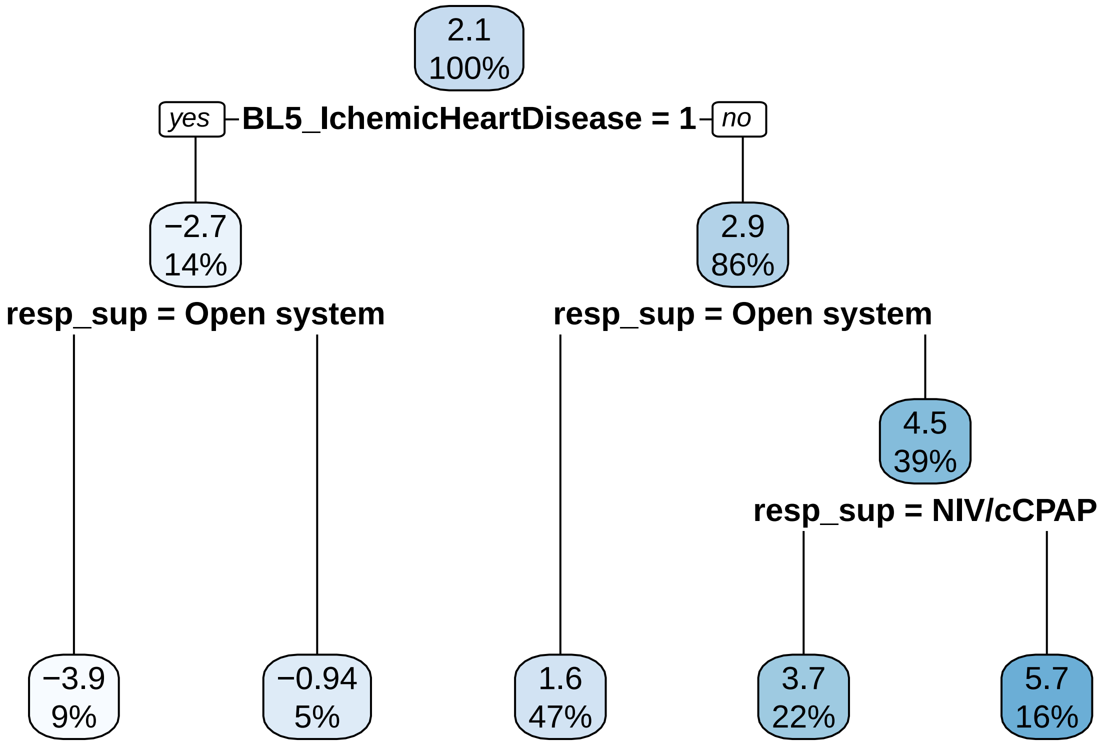
```
print("Number of covariates:")
```

```
## [1] "Number of covariates:"
```

```
print(numcovars - 1)
```

```
## [1] 2
```

```
cartmod <- rpart(cate_c ~ ., data = dat[, c(covar_include, 15)],
                      method = "anova")
rpart.plot(cartmod)
```



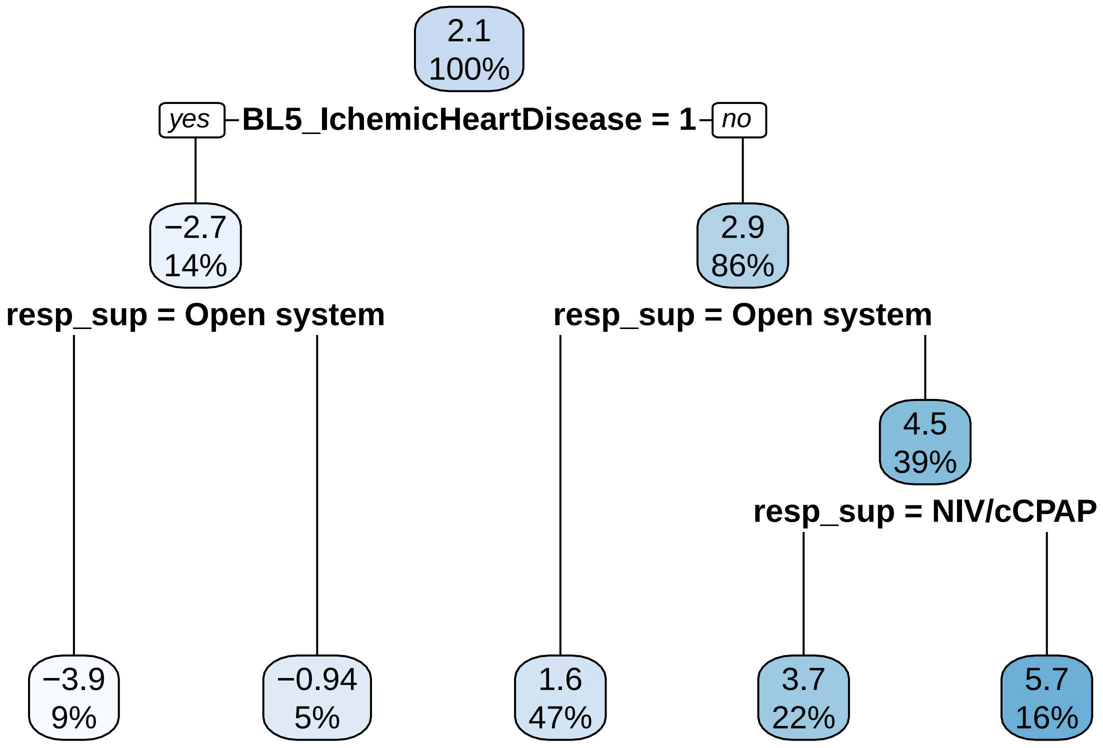
Finally we explore further pruning:

```
printcp(cartmod)
```

```
##
## Regression tree:
```

```
## rpart(formula = cate_c ~ ., data = dat[, c(covar_include, 15)],
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] BL5_IchemicHeartDisease resp_sup
##
## Root node error: 6748.3/968 = 6.9714
##
## n= 968
##
##         CP nsplit rel error   xerror      xstd
## 1 0.567383      0  1.000000 1.000905 0.0500998
## 2 0.264690      1  0.432617 0.433516 0.0163141
## 3 0.050725      2  0.167928 0.169038 0.0091177
## 4 0.041675      3  0.117203 0.118065 0.0080459
## 5 0.010000      4  0.075528 0.076331 0.0041452
```

```
prunedtree <-
  prune(cartmod,
        cp = cartmod$cptable[which.min(cartmod$cptable[, "xerror"]), "CP"])
rpart.plot(prunedtree)
```

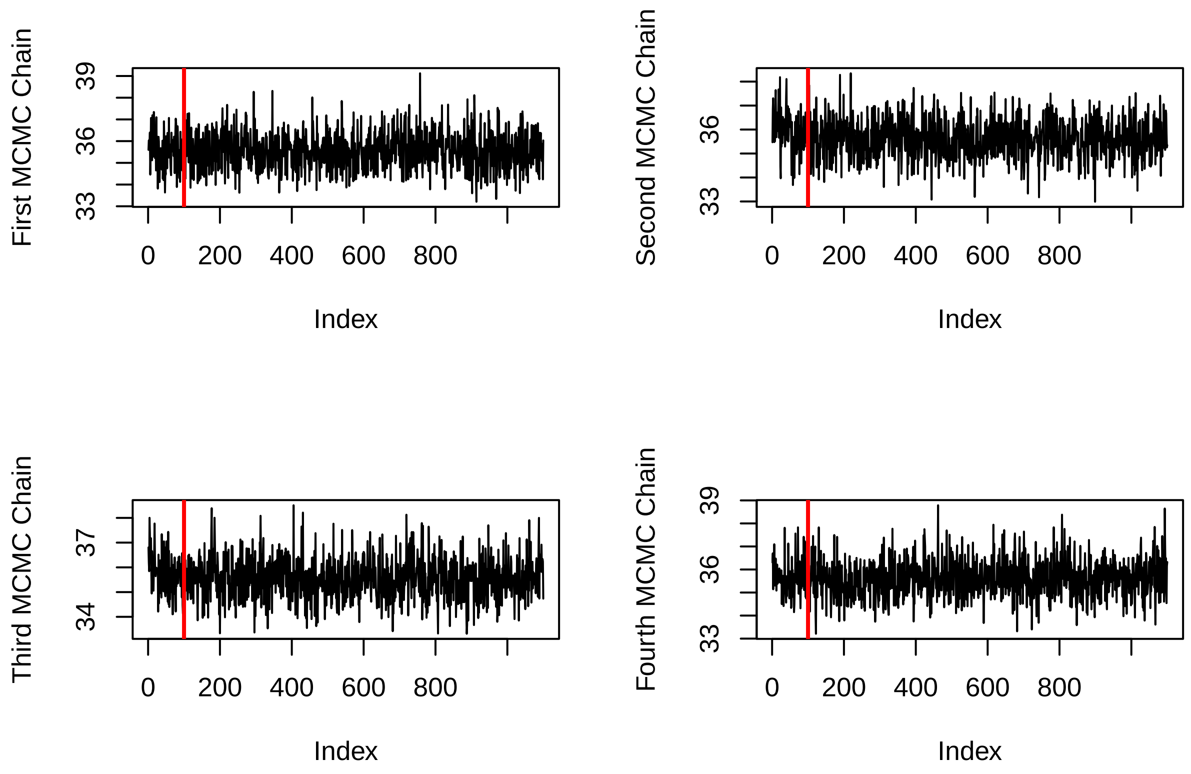


Further (non-automated) pruning may need to be considered here for trees that remain too large to easily interpret.

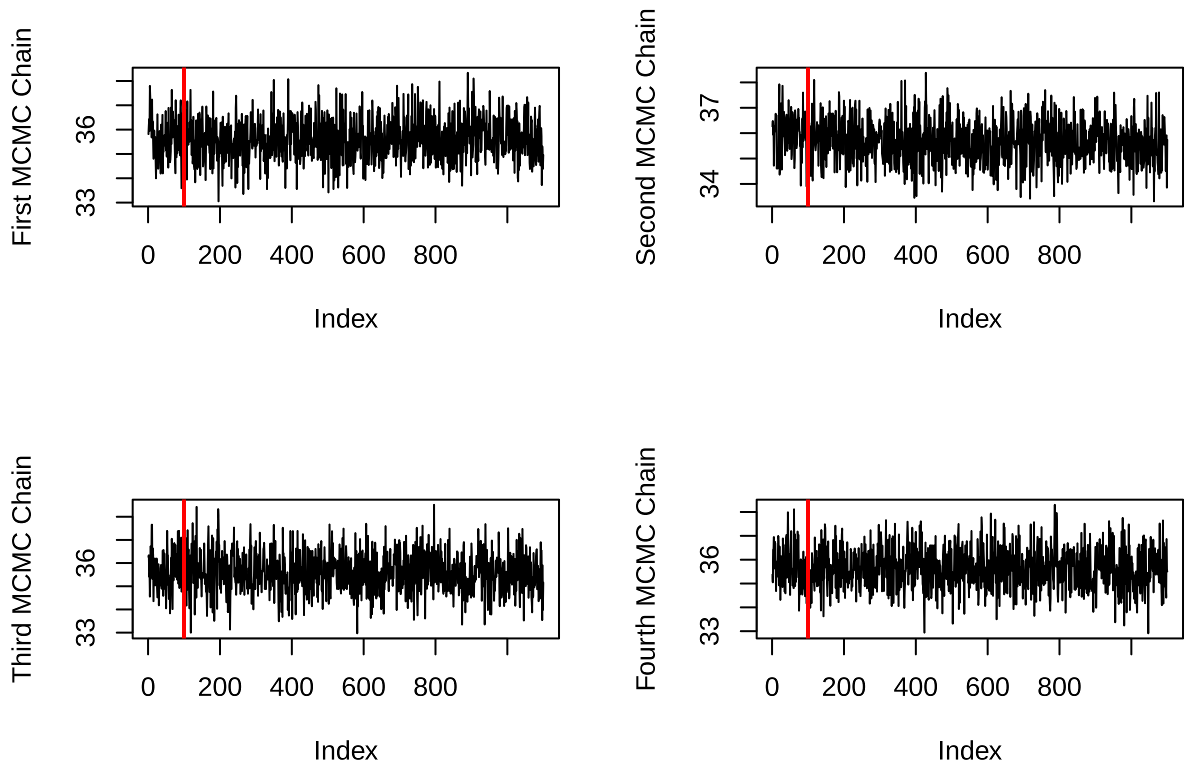
Diagnostics

Next we run standard diagnostics for each of the models. We begin with diagnostics for the continuous outcome models, which are simpler.

```r
# MCMC chains for parameter in models of continuous outcome
par(mfrow = c(2, 2))
plot(bartmod0_c$sigma[, 1], type = "l", ylab = "First MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
plot(bartmod0_c$sigma[, 2], type = "l", ylab = "Second MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
plot(bartmod0_c$sigma[, 3], type = "l", ylab = "Third MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
plot(bartmod0_c$sigma[, 4], type = "l", ylab = "Fourth MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
```



```r
plot(bartmod1_c$sigma[, 1], type = "l", ylab = "First MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
plot(bartmod1_c$sigma[, 2], type = "l", ylab = "Second MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
plot(bartmod1_c$sigma[, 3], type = "l", ylab = "Third MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
plot(bartmod1_c$sigma[, 4], type = "l", ylab = "Fourth MCMC Chain")
abline(v = 100, lwd = 2, col = "red")
```

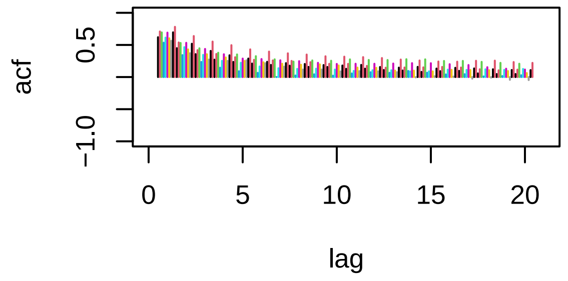


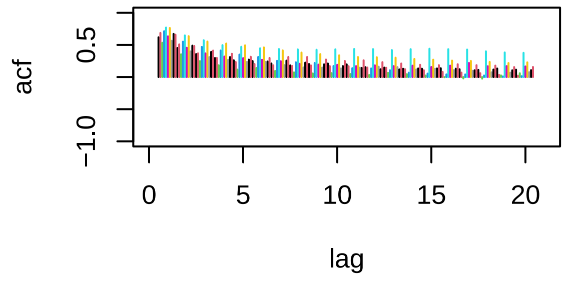
One should check that each chain has converged after burn-in (designated by the vertical red lines). In general, the chains should be converging to approximately the same value.

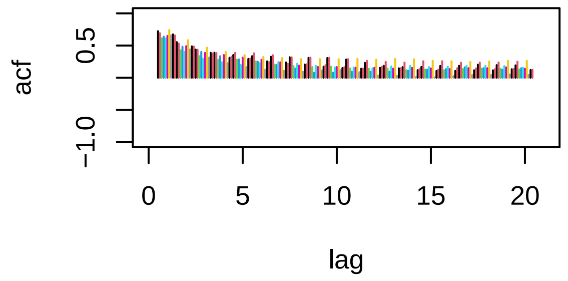
Next consider diagnostics for the models with the binary mortality outcome as described in Sparapani (2021). First consider the autocorrelation of the estimated response surface from BART from 10 randomly selected subjects. This may start somewhat correlated for nearby observations, but should reduce to 0 correlation for observations further apart.

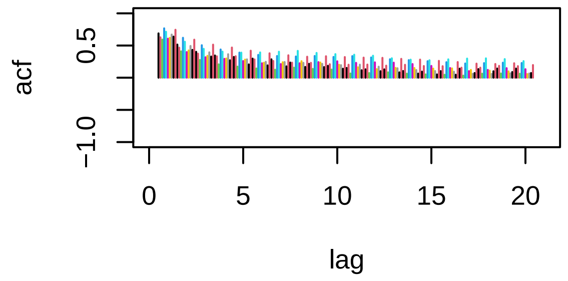
```
# First for bartmod0, one panel for each chain
par(mfrow = c(2, 2))

auto.corr <- acf(bartmod0$yhat.train[ , sample(1:dim(dat)[1], 10), 1],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}
```

```r
auto.corr <- acf(bartmod0$yhat.train[ , sample(1:dim(dat)[1], 10), 2],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}

auto.corr <- acf(bartmod0$yhat.train[ , sample(1:dim(dat)[1], 10), 3],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}

auto.corr <- acf(bartmod0$yhat.train[ , sample(1:dim(dat)[1], 10), 4],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}
```



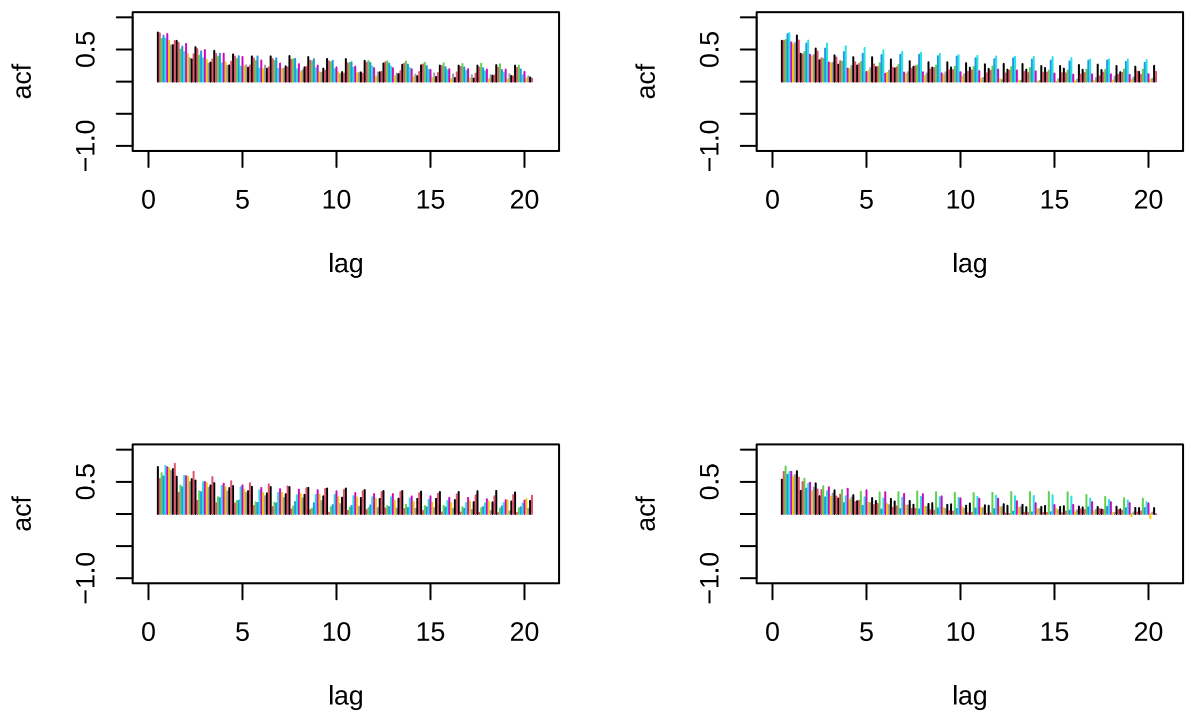






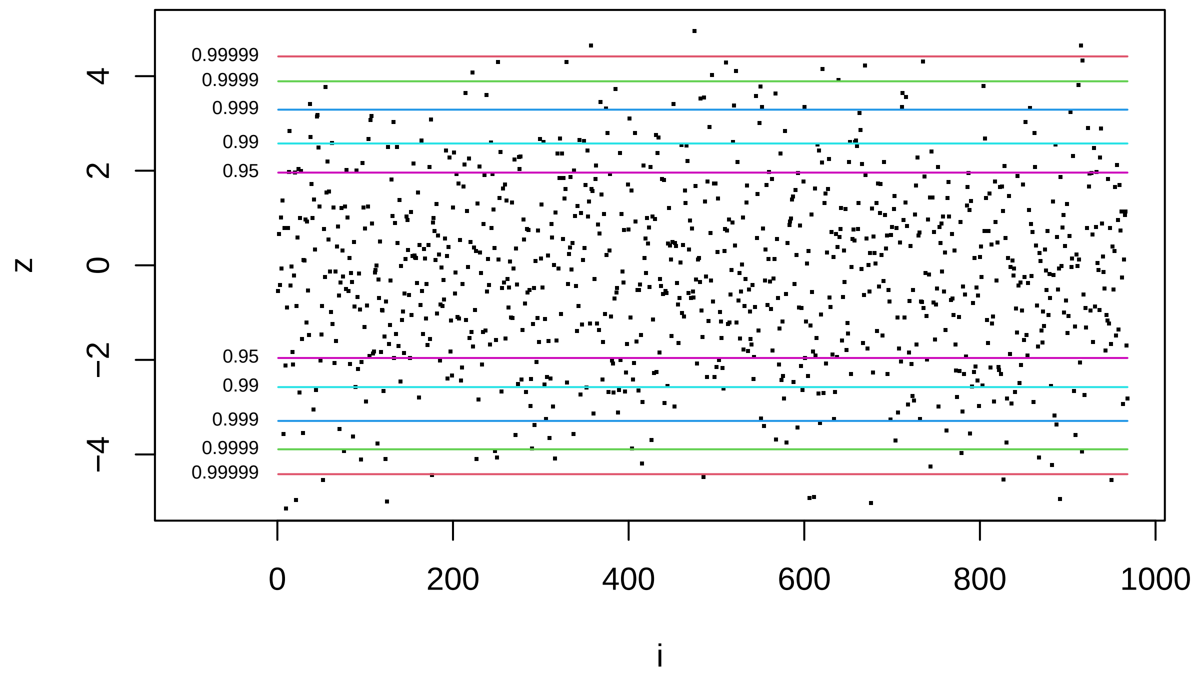
```
# Then for bartmod1
auto.corr <- acf(bartmod1$yhat.train[ , sample(1:dim(dat)[1], 10), 1],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}

auto.corr <- acf(bartmod1$yhat.train[ , sample(1:dim(dat)[1], 10), 2],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
```

```r
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}

auto.corr <- acf(bartmod1$yhat.train[ , sample(1:dim(dat)[1], 10), 3],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}

auto.corr <- acf(bartmod1$yhat.train[ , sample(1:dim(dat)[1], 10), 4],
                 plot = FALSE)
max.lag <- max(auto.corr$lag[ , 1, 1])

j <- seq(-0.5, 0.4, length.out = 10)
for (h in 1:10) {
  if (h == 1) {
    plot(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
         type = 'h', xlim = c(0, max.lag + 1), ylim = c(-1, 1),
         ylab = 'acf', xlab = 'lag')
  } else {
    lines(1:max.lag + j[h], auto.corr$acf[1 + (1:max.lag), h, h],
          type = 'h', col = h)
  }
}
```

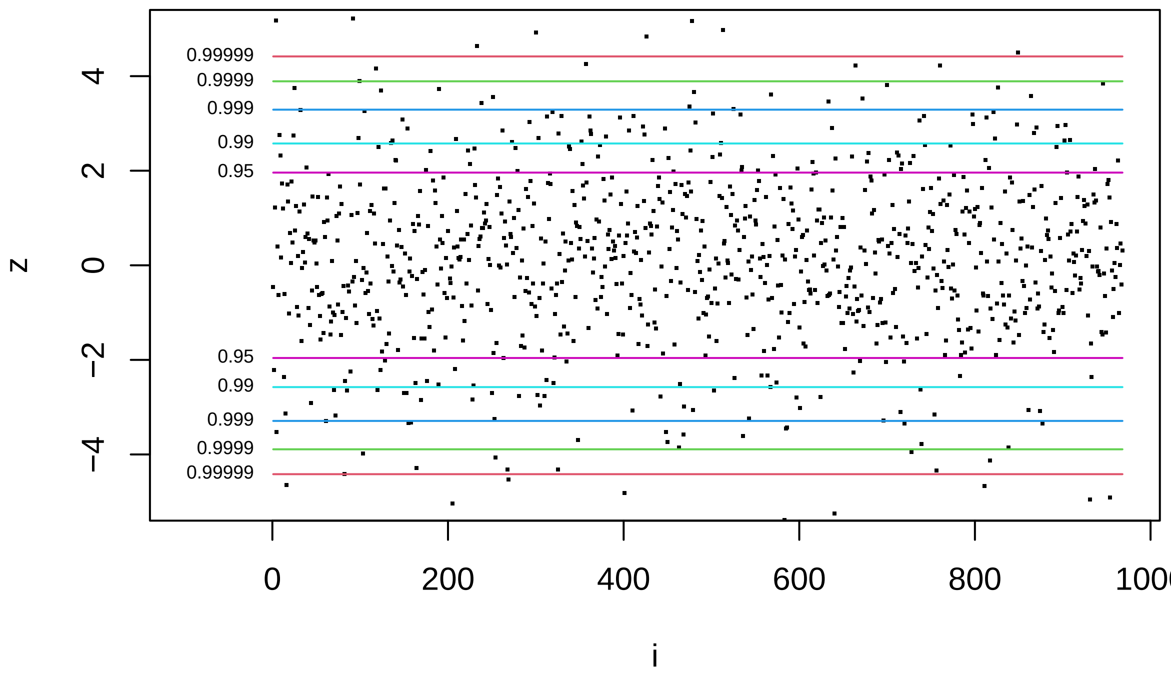


Next, we plot the Geweke Z statistics for each individual, which should be approximately distributed as a standard Normal.

```
# First for bartmod0
geweke <- gewekediag(bartmod0$yhat.train.collapse)
n <- dim(dat)[1]
j <- -10^(log10(n) - 1)
plot(geweke$z, pch = '.', cex = 2, ylab = 'z', xlab = 'i',
     xlim=c(j, n), ylim=c(-5, 5))
lines(1:n, rep(-1.96, n), type='l', col=6)
lines(1:n, rep(+1.96, n), type='l', col=6)
lines(1:n, rep(-2.576, n), type='l', col=5)
lines(1:n, rep(+2.576, n), type='l', col=5)
lines(1:n, rep(-3.291, n), type='l', col=4)
lines(1:n, rep(+3.291, n), type='l', col=4)
lines(1:n, rep(-3.891, n), type='l', col=3)
lines(1:n, rep(+3.891, n), type='l', col=3)
lines(1:n, rep(-4.417, n), type='l', col=2)
lines(1:n, rep(+4.417, n), type='l', col=2)
text(c(1, 1), c(-1.96, 1.96), pos=2, cex=0.6, labels='0.95')
text(c(1, 1), c(-2.576, 2.576), pos=2, cex=0.6, labels='0.99')
text(c(1, 1), c(-3.291, 3.291), pos=2, cex=0.6, labels='0.999')
text(c(1, 1), c(-3.891, 3.891), pos=2, cex=0.6, labels='0.9999')
text(c(1, 1), c(-4.417, 4.417), pos=2, cex=0.6, labels='0.99999')
```



```
# Then for bartmod1
geweke <- gewekediag(bartmod1$yhat.train.collapse)
plot(geweke$z, pch = '.', cex = 2, ylab = 'z', xlab = 'i',
     xlim=c(j, n), ylim=c(-5, 5))
lines(1:n, rep(-1.96, n), type='l', col=6)
lines(1:n, rep(+1.96, n), type='l', col=6)
lines(1:n, rep(-2.576, n), type='l', col=5)
lines(1:n, rep(+2.576, n), type='l', col=5)
lines(1:n, rep(-3.291, n), type='l', col=4)
lines(1:n, rep(+3.291, n), type='l', col=4)
lines(1:n, rep(-3.891, n), type='l', col=3)
lines(1:n, rep(+3.891, n), type='l', col=3)
lines(1:n, rep(-4.417, n), type='l', col=2)
lines(1:n, rep(+4.417, n), type='l', col=2)
text(c(1, 1), c(-1.96, 1.96), pos=2, cex=0.6, labels='0.95')
text(c(1, 1), c(-2.576, 2.576), pos=2, cex=0.6, labels='0.99')
text(c(1, 1), c(-3.291, 3.291), pos=2, cex=0.6, labels='0.999')
text(c(1, 1), c(-3.891, 3.891), pos=2, cex=0.6, labels='0.9999')
text(c(1, 1), c(-4.417, 4.417), pos=2, cex=0.6, labels='0.99999')
```



If several points lie beyond the dark blue line or further, consider using more thinning when fitting the models.

Sensitivity analysis

We conduct two simple sensitivity analyses, one assuming that all missing mortality data correspond to an alive status and one assuming that all missing mortality data correspond to a deceased status. In the former scenario, missing days without life support will be set to a random sample with replacement from the observed days without life support among those known to be alive at day 90; in the latter scenario it will be set to a random sample with replacement from the observed days without life support among those not alive at day 90. Default hyperparameters are used in these analyses.

Note that these analyses are unlikely to output the exact same trees as the originl analysis, but they should hopefully output trees which largely tell the same HTE story as the original analysis.

Sensitivity analysis 1: Impute missing outcomes as alive

First the memory is cleared and the data set is reloaded (the cross validation output is saved for future use).

```
rm(list = ls()[!(ls() %in% list("cvoutput"))])
library(BART)
library(caret)
library(rpart)
library(rpart.plot)
source("clusterfunctions.R")
```

```r
# Load data from appropriate directory (edit as needed)
dat <- read.csv2("~/Downloads/synth_covid.csv")
```

Next we do a small amount of data cleaning/preparation.

```r
# Clean up data variables types and impute the small amount of missing data
dat$resp_sup <- as.factor(dat$resp_sup)
dat$dead90 <- ifelse(dat$dead90 == TRUE, 1, 0)
dat$dawols90[is.na(dat$dawols90)] <-
  sample(dat$dawols90[!is.na(dat$dawols90) & dat$dead90 == 0],
         sum(is.na(dat$dawols90)), replace = TRUE)
dat$dead90[is.na(dat$dead90)] <- 0

# Standardize continuous covariates
dat$age <- (dat$age - mean(dat$age)) / sd(dat$age)
dat$BL9_Weight <- (dat$BL9_Weight - mean(dat$BL9_Weight)) / sd(dat$BL9_Weight)

# Make datasets under each counterfactual
dat1 <- dat0 <- dat
dat1$allocation <- TRUE
dat0$allocation <- FALSE
```

Then we run a BART analysis focused on the binary mortality outcome. Note that we use the hyperparameters selected during the cross-validation procedures in the main analysis.

```r
# Fit BART models, get predictions under each trt
set.seed(60622)
bartmod1 <-
  lbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dead90,
                x.test = dat1[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE)])
bartmod0 <-
  lbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dead90,
                x.test = dat0[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE)])

# Collapse predictions across chains for certain calculations
bartmod1$yhat.train.collapse <- apply(bartmod1$yhat.train, 2, rbind)
bartmod1$yhat.test.collapse <- apply(bartmod1$yhat.test, 2, rbind)
bartmod0$yhat.train.collapse <- apply(bartmod0$yhat.train, 2, rbind)
bartmod0$yhat.test.collapse <- apply(bartmod0$yhat.test, 2, rbind)
```

Then conditional average treatment effects are estimated using the predictions under each counterfactual.

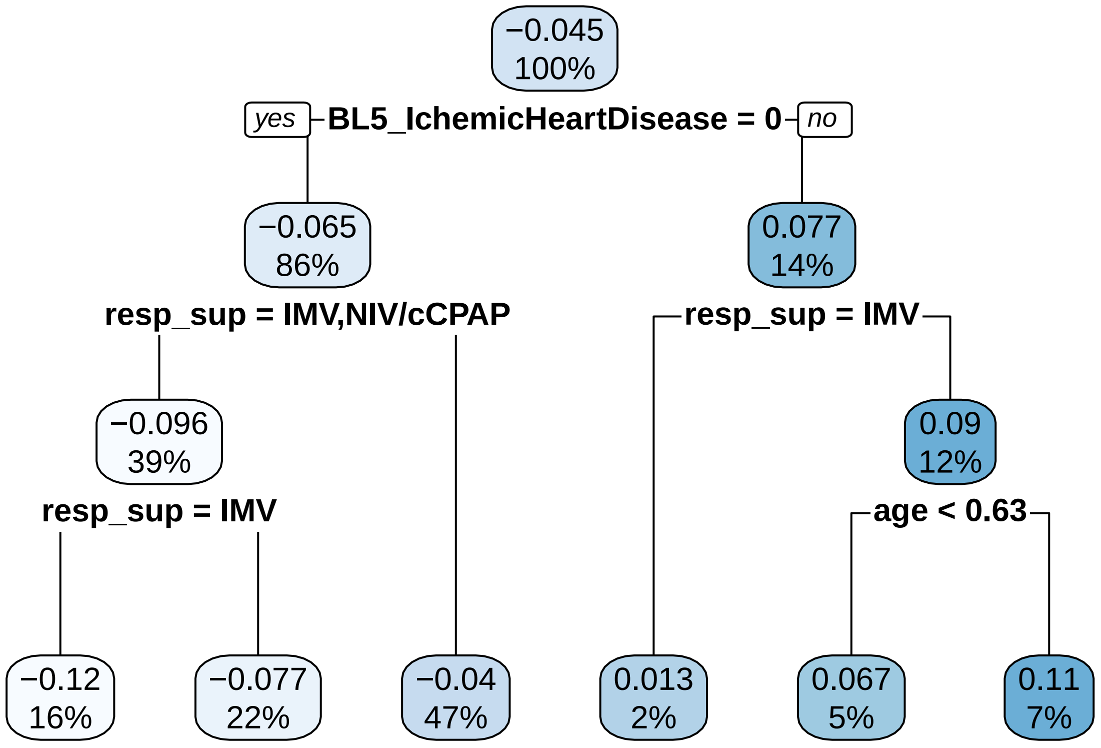
```r
dat$cate <-
  exp(colMeans(bartmod1$yhat.test.collapse)) /
    (1 + exp(colMeans(bartmod1$yhat.test.collapse))) -
  exp(colMeans(bartmod0$yhat.test.collapse)) /
    (1 + exp(colMeans(bartmod0$yhat.test.collapse)))
```

This full process is then repeated for the continuous outcome (days alive without life support by day 90).

```r
# Fit BART models, get predictions under each trt
set.seed(60622)
bartmod1_c <-
  wbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dawols90,
                x.test = dat1[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE_c)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE_c)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE_c)])
bartmod0_c <-
  wbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dawols90,
                x.test = dat0[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE_c)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE_c)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE_c)])

# Collapse predictions across chains for certain calculations
bartmod1_c$yhat.train.collapse <- apply(bartmod1_c$yhat.train, 2, rbind)
bartmod1_c$yhat.test.collapse <- apply(bartmod1_c$yhat.test, 2, rbind)
bartmod0_c$yhat.train.collapse <- apply(bartmod0_c$yhat.train, 2, rbind)
bartmod0_c$yhat.test.collapse <- apply(bartmod0_c$yhat.test, 2, rbind)

# Estimate CATEs
dat$cate_c <- colMeans(bartmod1_c$yhat.test.collapse) -
              colMeans(bartmod0_c$yhat.test.collapse)
```

Finally, the "fit-the-fit" approach is used to find subgroups exhibiting heterogeneity of treatment effect, starting with the binary outcome. In particular, a CART model is fit with the CATE for 90-day mortality as the outcome and the covariates as possible predictors. The model is first fit under default CART hyperparameter settings.

```r
# CART model for 90 day mortality with default CART hyperparameter and
# all covariates considered
cartmod <- rpart(cate ~ ., data = dat[, c(4:14)], method = "anova")
rpart.plot(cartmod)
```



Now we prune the tree for interpretability using the stepwise approach of Hu et al. In particular, covariates are added to the CART model sequentially according to greatest increase in model R^2 until an increase of less than 1% occurs.

```
# Pruned tree using stepwise approach of Hu et al.
r2 <- c()
covar_include <- c()
covar_cols <- 4:13
currentr2 <- 0
maxcovars <- 4

for (numcovars in 1:maxcovars) {

  for (covar in covar_cols) {
    cartmod <- rpart(cate ~ ., data = dat[, c(covar, covar_include, 14)],
                     method = "anova")
    r2[which(covar_cols == covar)] <-
      1 - printcp(cartmod)[dim(printcp(cartmod))[1], 3]
  }

  if ((max(r2) - currentr2) / currentr2 > 0.01) {
    covar_include <- c(covar_include, covar_cols[which.max(r2)])
    covar_cols <- covar_cols[covar_cols != covar_cols[which.max(r2)]]
    currentr2 <- max(r2)
    r2 <- c()
  } else {
    break
  }

}
```

Print out the number of covariates selected and the resulting CART model output.

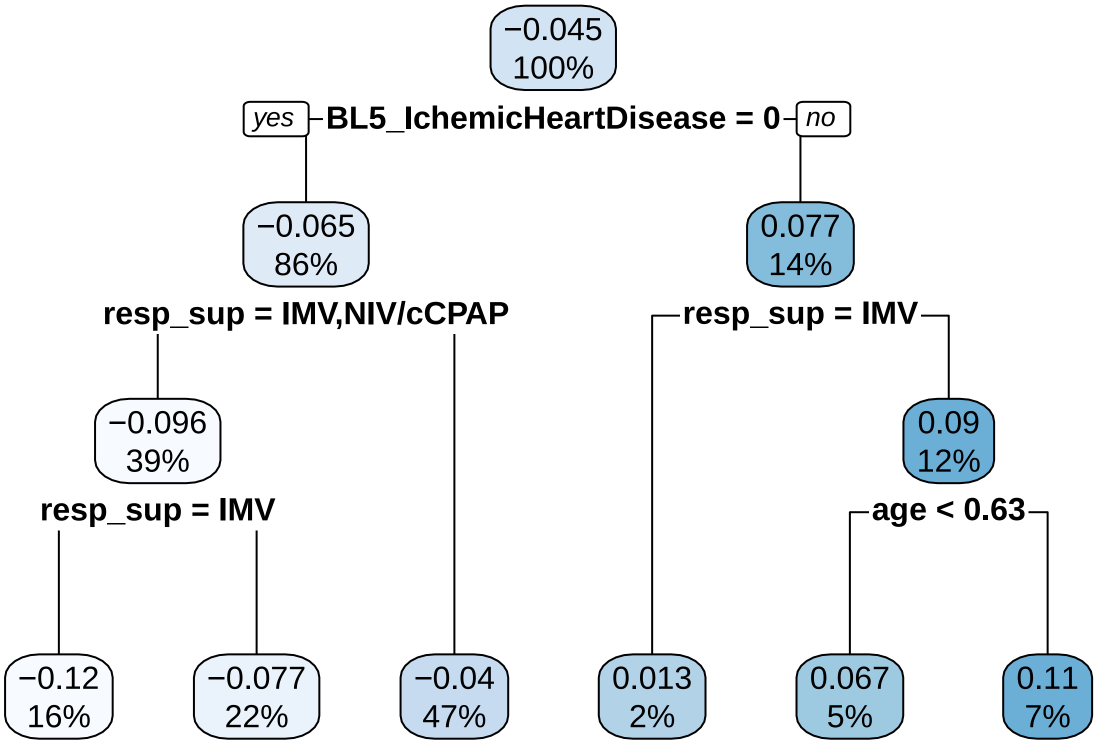
```
print("Number of covariates:")
```

```
## [1] "Number of covariates:"
```

```
print(numcovars - 1)
```

```
## [1] 3
```

```
cartmod <- rpart(cate ~ ., data = dat[, c(covar_include, 14)],
                     method = "anova")
rpart.plot(cartmod)
```

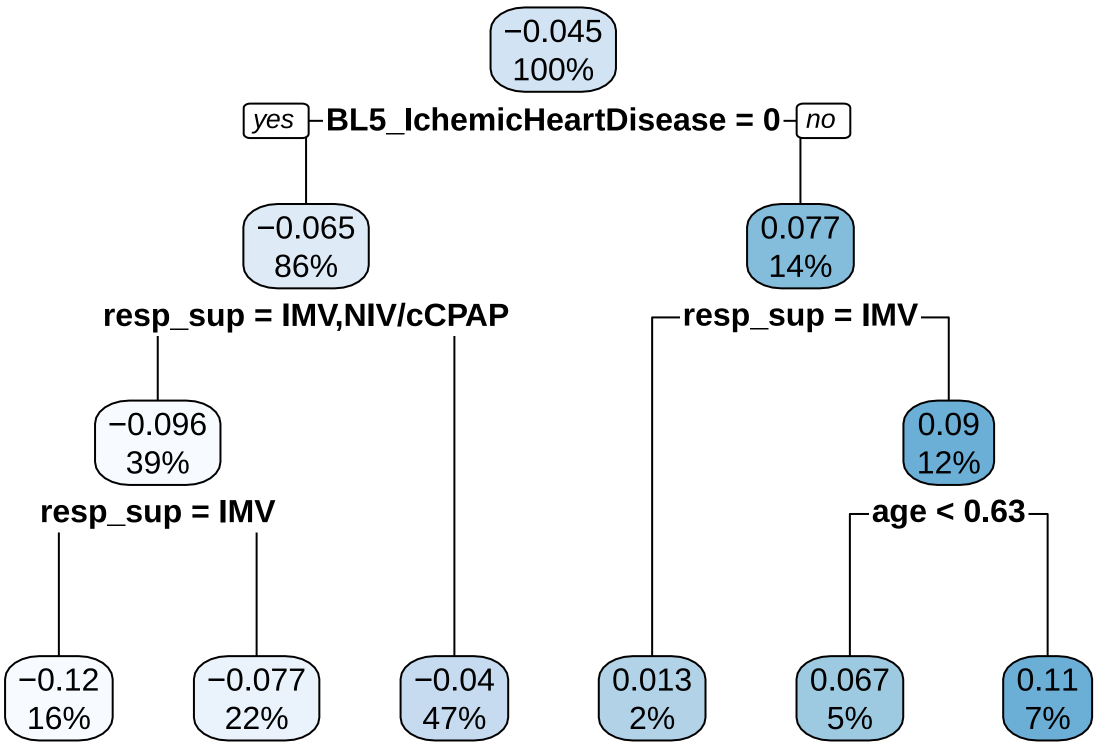


Finally we explore further pruning:

```
printcp(cartmod)
```

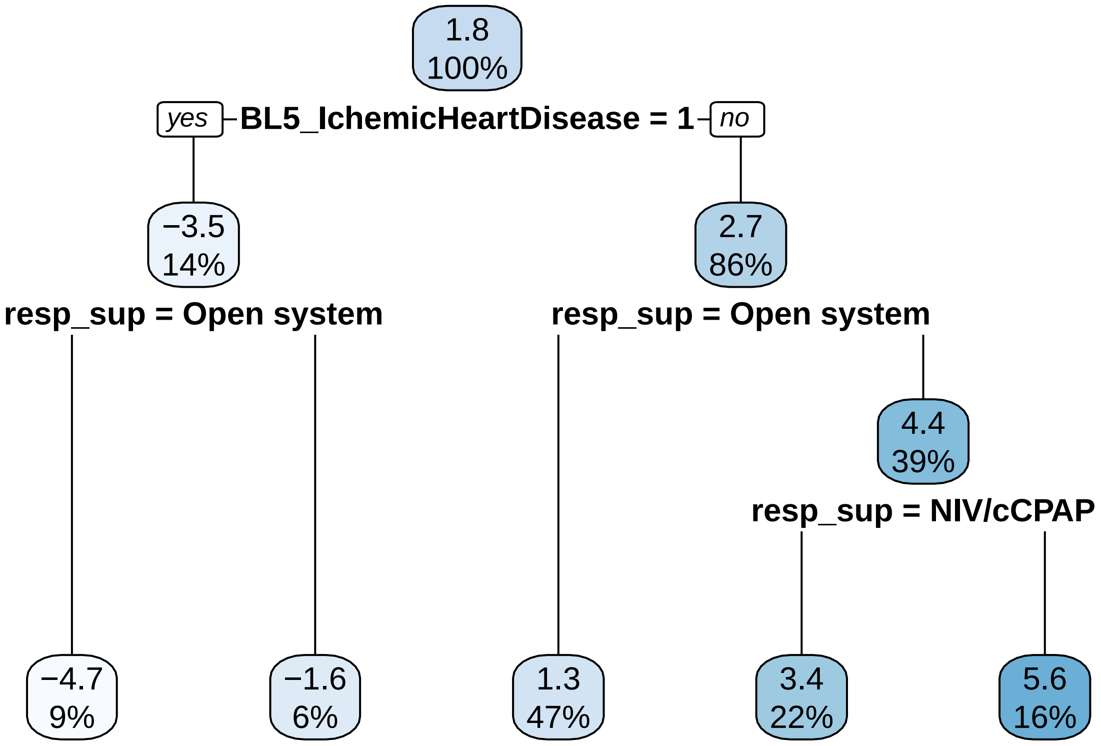
```
##
## Regression tree:
## rpart(formula = cate ~ ., data = dat[, c(covar_include, 14)],
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] age                      BL5_IchemicHeartDisease resp_sup
##
## Root node error: 4.1096/982 = 0.004185
##
## n= 982
##
##         CP nsplit rel error  xerror      xstd
## 1 0.601207      0   1.00000 1.00162 0.0534795
## 2 0.159823      1   0.39879 0.40099 0.0173349
## 3 0.046114      2   0.23897 0.24721 0.0115911
## 4 0.028728      3   0.19286 0.19550 0.0098252
## 5 0.011450      4   0.16413 0.16633 0.0079972
## 6 0.010000      5   0.15268 0.15603 0.0076890
```

```r
prunedtree <-
  prune(cartmod,
        cp = cartmod$cptable[which.min(cartmod$cptable[, "xerror"]), "CP"])
rpart.plot(prunedtree)
```



Next the same fit-the-fit approach is used to summarize the results for the continuous outcome, starting with a CART model using all covariates and default hyperparameter.

```r
# CART model for 90 day mortality with default CART hyperparameter and
# all covariates considered
cartmod <- rpart(cate_c ~ ., data = dat[, c(4:13, 15)], method = "anova")
rpart.plot(cartmod)
```



Now we prune the tree for interpretability using the stepwise approach of Hu et al.

```r
# Pruned tree using stepwise approach of Hu et al.
r2 <- c()
covar_include <- c()
covar_cols <- 4:13
currentr2 <- 0
maxcovars <- 4

for (numcovars in 1:maxcovars) {

  for (covar in covar_cols) {
    cartmod <- rpart(cate_c ~ ., data = dat[, c(covar, covar_include, 15)],
                     method = "anova")
    r2[which(covar_cols == covar)] <-
      1 - printcp(cartmod)[dim(printcp(cartmod))[1], 3]
  }

  if ((max(r2) - currentr2) / currentr2 > 0.01) {
    covar_include <- c(covar_include, covar_cols[which.max(r2)])
    covar_cols <- covar_cols[covar_cols != covar_cols[which.max(r2)]]
    currentr2 <- max(r2)
    r2 <- c()
  } else {
    break
  }

}
```

Print out the number of covariates selected and the resulting CART model output.

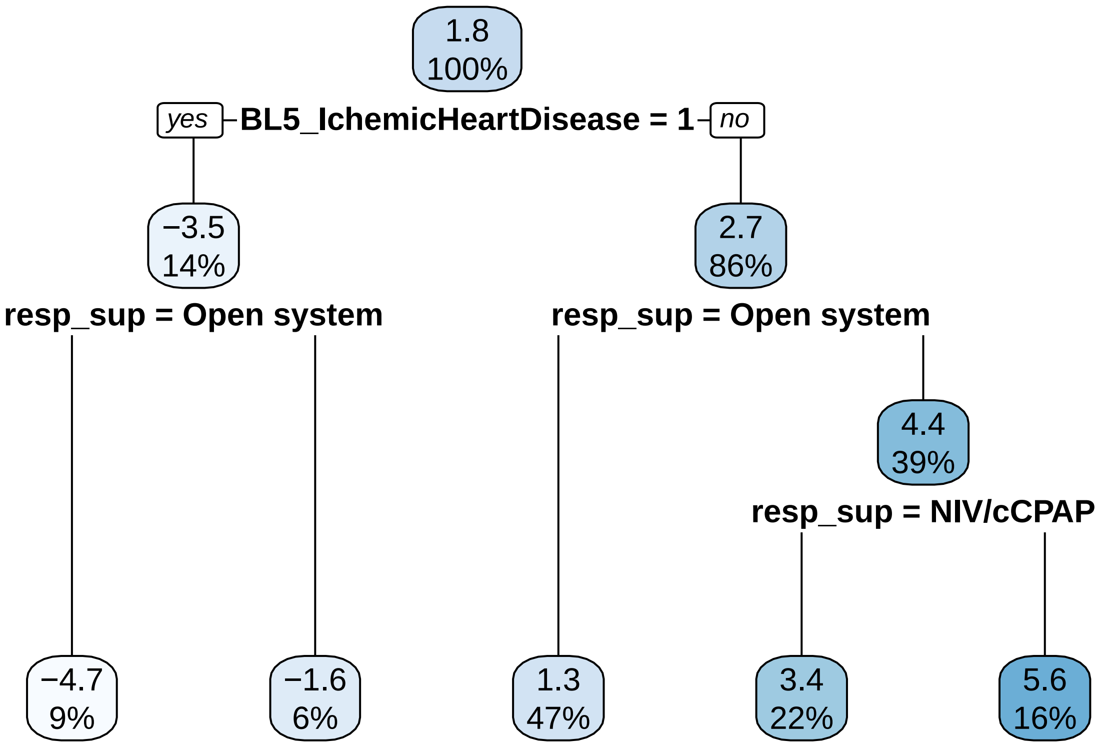
```
print("Number of covariates:")
```

```
## [1] "Number of covariates:"
```

```
print(numcovars - 1)
```

```
## [1] 2
```

```
cartmod <- rpart(cate_c ~ ., data = dat[, c(covar_include, 15)],
                     method = "anova")
rpart.plot(cartmod)
```

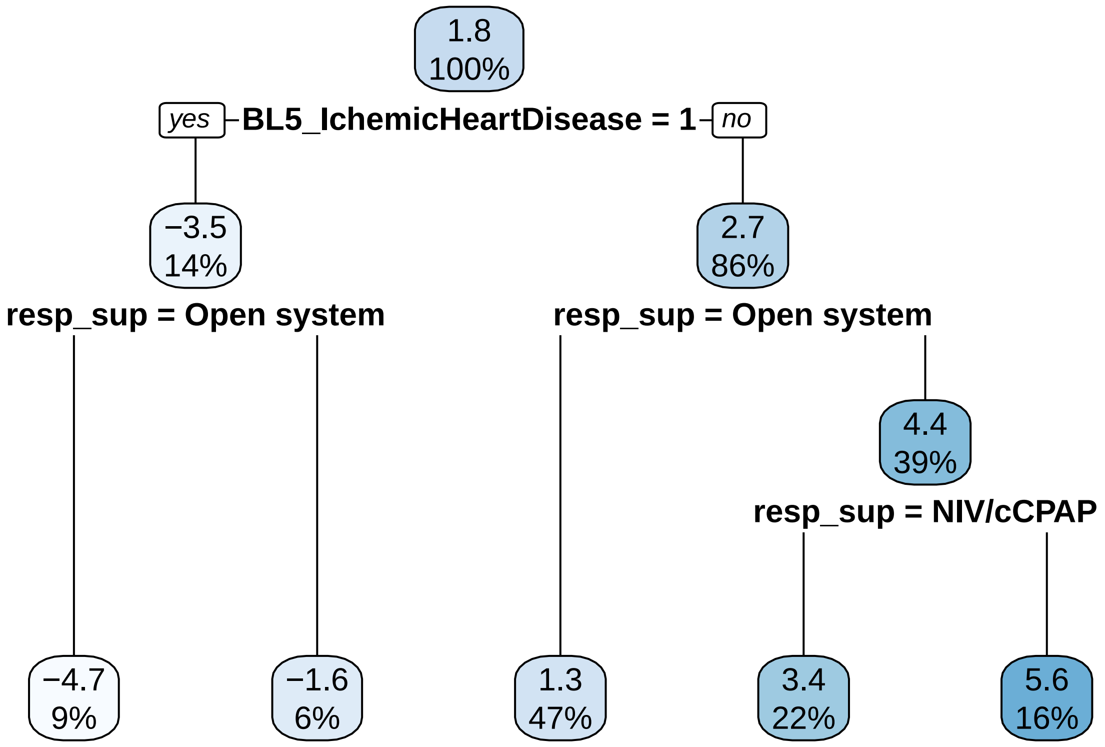


Finally we explore further pruning:

```r
printcp(cartmod)
```

```
##
## Regression tree:
## rpart(formula = cate_c ~ ., data = dat[, c(covar_include, 15)],
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] BL5_IchemicHeartDisease resp_sup
##
## Root node error: 7845.8/982 = 7.9896
##
## n= 982
##
##          CP nsplit rel error   xerror      xstd
## 1 0.603514      0  1.000000 1.000955 0.0509835
## 2 0.246272      1  0.396486 0.397326 0.0143023
## 3 0.056579      2  0.150214 0.150915 0.0081209
## 4 0.041664      3  0.093635 0.094377 0.0072942
## 5 0.010000      4  0.051971 0.052652 0.0029402
```

```r
prunedtree <-
  prune(cartmod,
        cp = cartmod$cptable[which.min(cartmod$cptable[, "xerror"]), "CP"])
rpart.plot(prunedtree)
```



Sensitivity analysis 2: Impute missing outcomes as deceased

First the memory is cleared and the data set is reloaded.

```r
rm(list = ls()[!(ls() %in% list("cvoutput"))])
library(BART)
library(caret)
library(rpart)
library(rpart.plot)
source("clusterfunctions.R")

# Load data from appropriate directory (edit as needed)
dat <- read.csv2("~/Downloads/synth_covid.csv")
```

Next we do a small amount of data cleaning/preparation.

```r
# Clean up data variables types and impute the small amount of missing data
dat$resp_sup <- as.factor(dat$resp_sup)
dat$dead90 <- ifelse(dat$dead90 == TRUE, 1, 0)
dat$dawols90[is.na(dat$dawols90)] <-
  sample(dat$dawols90[!is.na(dat$dawols90) & dat$dead90 == 1],
         sum(is.na(dat$dawols90)), replace = TRUE)
dat$dead90[is.na(dat$dead90)] <- 1

# Standardize continuous covariates
dat$age <- (dat$age - mean(dat$age)) / sd(dat$age)
dat$BL9_Weight <- (dat$BL9_Weight - mean(dat$BL9_Weight)) / sd(dat$BL9_Weight)

# Make datasets under each counterfactual
dat1 <- dat0 <- dat
dat1$allocation <- TRUE
dat0$allocation <- FALSE
```

Then we run a BART analysis focused on the binary mortality outcome.

```r
# Fit BART models under default hyperparameters, get predictions under each trt
set.seed(60622)
bartmod1 <-
  lbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dead90,
                x.test = dat1[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE)])
bartmod0 <-
  lbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dead90,
                x.test = dat0[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE)])

# Collapse predictions across chains for certain calculations
bartmod1$yhat.train.collapse <- apply(bartmod1$yhat.train, 2, rbind)
bartmod1$yhat.test.collapse <- apply(bartmod1$yhat.test, 2, rbind)
```

```
bartmod0$yhat.train.collapse <- apply(bartmod0$yhat.train, 2, rbind)
bartmod0$yhat.test.collapse <- apply(bartmod0$yhat.test, 2, rbind)
```

Then conditional average treatment effects are estimated using the predictions under each counterfactual.

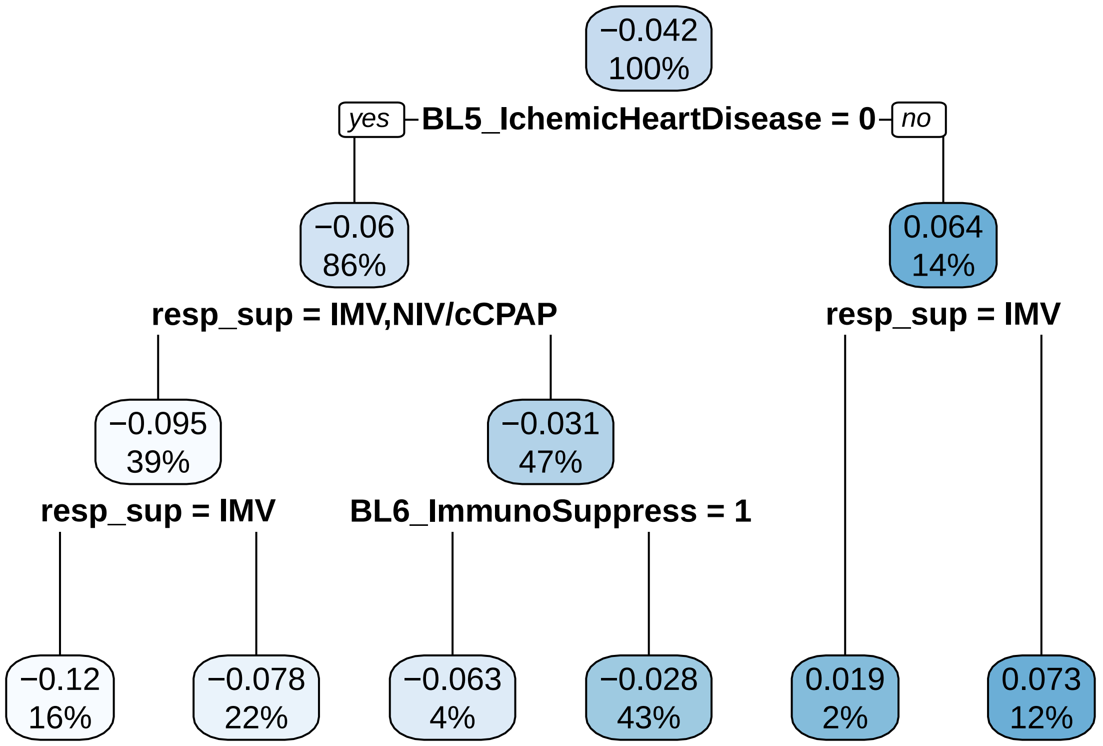
```
dat$cate <-
  exp(colMeans(bartmod1$yhat.test.collapse)) /
    (1 + exp(colMeans(bartmod1$yhat.test.collapse))) -
  exp(colMeans(bartmod0$yhat.test.collapse)) /
    (1 + exp(colMeans(bartmod0$yhat.test.collapse)))
```

This full process is then repeated for the continuous outcome (days alive without life support by day 90).

```
# Fit BART models under default hyperparameters, get predictions under each trt
set.seed(60622)
bartmod1_c <-
  wbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dawols90,
                x.test = dat1[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE_c)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE_c)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE_c)])
bartmod0_c <-
  wbart.cluster(x.train = dat[, c(1, 4:13)], y.train = dat$dawols90,
                x.test = dat0[, c(1, 4:13)], nchains = 4,
                power = cvoutput$Power[which.min(cvoutput$CVMSE_c)],
                base = cvoutput$Base[which.min(cvoutput$CVMSE_c)],
                ntree = cvoutput$Ntrees[which.min(cvoutput$CVMSE_c)])

# Collapse predictions across chains for certain calculations
bartmod1_c$yhat.train.collapse <- apply(bartmod1_c$yhat.train, 2, rbind)
bartmod1_c$yhat.test.collapse <- apply(bartmod1_c$yhat.test, 2, rbind)
bartmod0_c$yhat.train.collapse <- apply(bartmod0_c$yhat.train, 2, rbind)
bartmod0_c$yhat.test.collapse <- apply(bartmod0_c$yhat.test, 2, rbind)

# Estimate CATEs
dat$cate_c <- colMeans(bartmod1_c$yhat.test.collapse) -
              colMeans(bartmod0_c$yhat.test.collapse)
```

Finally, the "fit-the-fit" approach is used to find subgroups exhibiting heterogeneity of treatment effect, starting with the binary outcome. In particular, a CART model is fit with the CATE for 90-day mortality as the outcome and the covariates as possible predictors. The model is first fit under default CART hyperparameter settings.

```
# CART model for 90 day mortality with default CART hyperparameter and
# all covariates considered
cartmod <- rpart(cate ~ ., data = dat[, c(4:14)], method = "anova")
rpart.plot(cartmod)
```



Now we prune the tree for interpretability using the stepwise approach of Hu et al. In particular, covariates are added to the CART model sequentially according to greatest increase in model R^2 until an increase of less than 1% occurs.

```
# Pruned tree using stepwise approach of Hu et al.
r2 <- c()
covar_include <- c()
covar_cols <- 4:13
currentr2 <- 0
maxcovars <- 4

for (numcovars in 1:maxcovars) {

  for (covar in covar_cols) {
    cartmod <- rpart(cate ~ ., data = dat[, c(covar, covar_include, 14)],
                     method = "anova")
    r2[which(covar_cols == covar)] <-
      1 - printcp(cartmod)[dim(printcp(cartmod))[1], 3]
  }

  if ((max(r2) - currentr2) / currentr2 > 0.01) {
    covar_include <- c(covar_include, covar_cols[which.max(r2)])
    covar_cols <- covar_cols[covar_cols != covar_cols[which.max(r2)]]
    currentr2 <- max(r2)
    r2 <- c()
  } else {
    break
  }

}
```

Print out the number of covariates selected and the resulting CART model output.

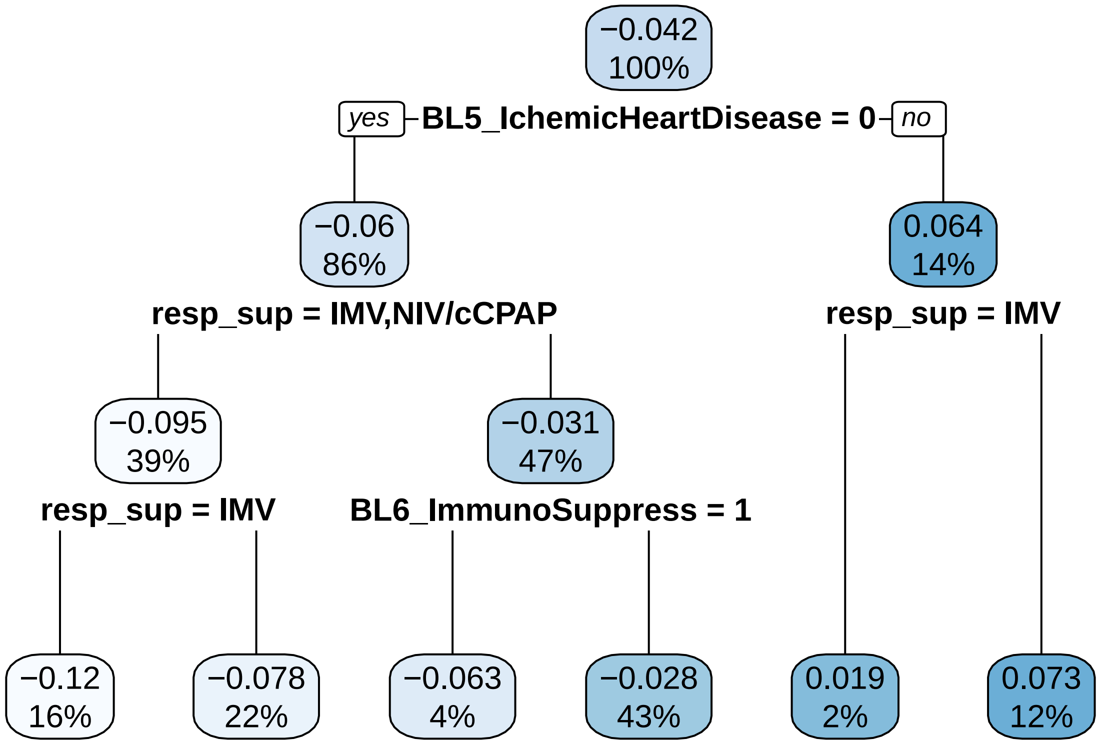
```
print("Number of covariates:")
```

```
## [1] "Number of covariates:"
```

```
print(numcovars - 1)
```

```
## [1] 3
```

```
cartmod <- rpart(cate ~ ., data = dat[, c(covar_include, 14)],
                     method = "anova")
rpart.plot(cartmod)
```

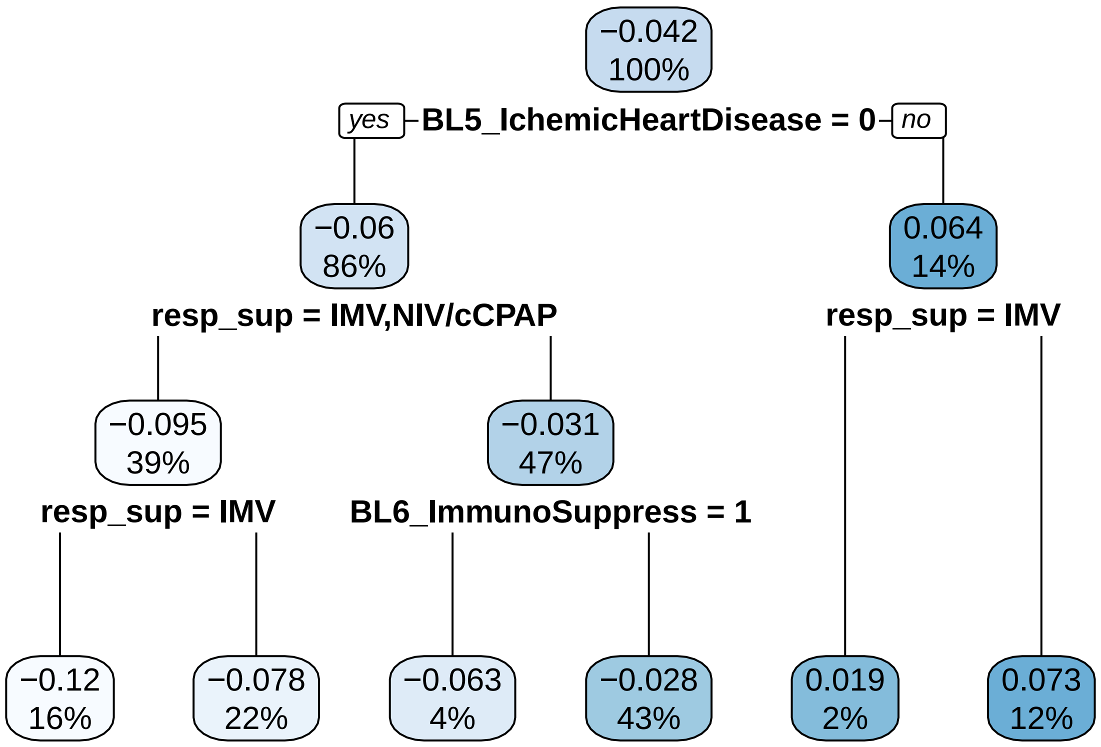


Finally we explore further pruning:

```
printcp(cartmod)
```

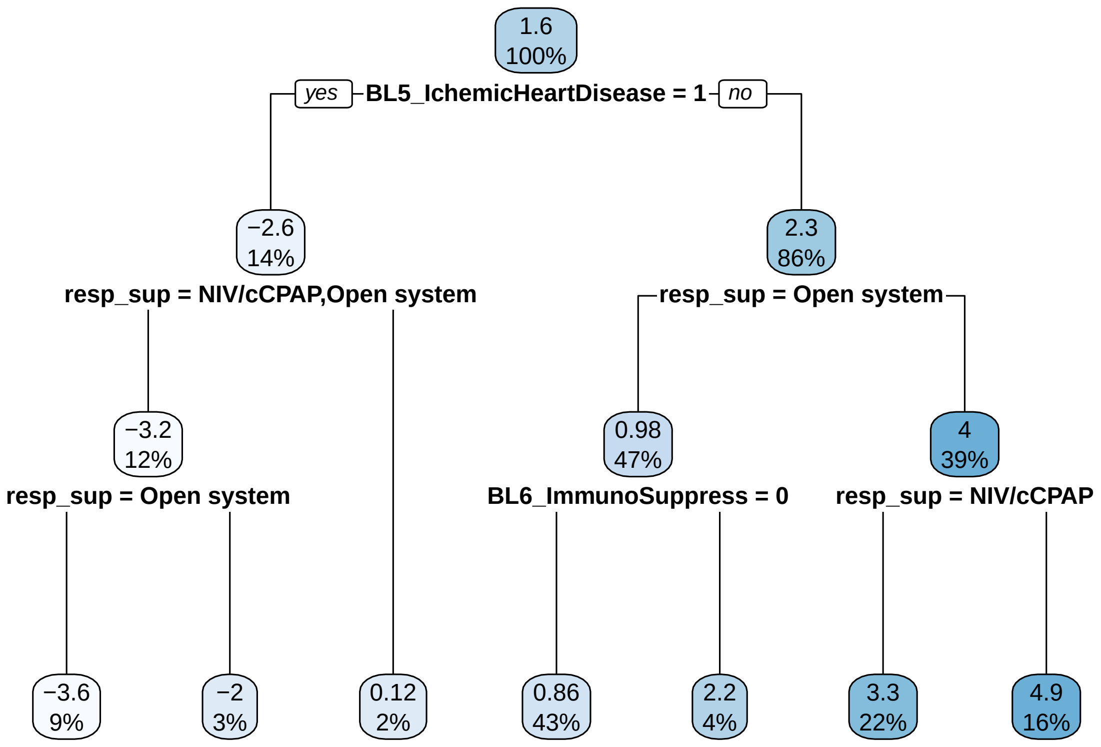
```
##
## Regression tree:
## rpart(formula = cate ~ ., data = dat[, c(covar_include, 14)],
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] BL5_IchemicHeartDisease BL6_ImmunoSuppress      resp_sup
##
## Root node error: 3.5063/982 = 0.0035706
##
## n= 982
##
##         CP nsplit rel error  xerror      xstd
## 1 0.530893      0   1.00000 1.00188 0.0480532
## 2 0.242800      1   0.46911 0.47127 0.0173316
## 3 0.039093      2   0.22631 0.22862 0.0109906
## 4 0.016678      3   0.18721 0.18970 0.0097420
## 5 0.013178      4   0.17053 0.17572 0.0088853
## 6 0.010000      5   0.15736 0.16699 0.0084884
```

```
prunedtree <-
  prune(cartmod,
        cp = cartmod$cptable[which.min(cartmod$cptable[, "xerror"]), "CP"])
rpart.plot(prunedtree)
```



Next the same fit-the-fit approach is used to summarize the results for the continuous outcome, starting with a CART model using all covariates and default hyperparameter.

```r
# CART model for 90 day mortality with default CART hyperparameter and
# all covariates considered
cartmod <- rpart(cate_c ~ ., data = dat[, c(4:13, 15)], method = "anova")
rpart.plot(cartmod)
```



Now we prune the tree for interpretability using the stepwise approach of Hu et al.

```r
# Pruned tree using stepwise approach of Hu et al.
r2 <- c()
covar_include <- c()
covar_cols <- 4:13
currentr2 <- 0
maxcovars <- 4

for (numcovars in 1:maxcovars) {

  for (covar in covar_cols) {
    cartmod <- rpart(cate_c ~ ., data = dat[, c(covar, covar_include, 15)],
                     method = "anova")
    r2[which(covar_cols == covar)] <-
      1 - printcp(cartmod)[dim(printcp(cartmod))[1], 3]
  }

  if ((max(r2) - currentr2) / currentr2 > 0.01) {
    covar_include <- c(covar_include, covar_cols[which.max(r2)])
    covar_cols <- covar_cols[covar_cols != covar_cols[which.max(r2)]]
    currentr2 <- max(r2)
    r2 <- c()
  } else {
    break
  }

}
```

Print out the number of covariates selected and the resulting CART model output.

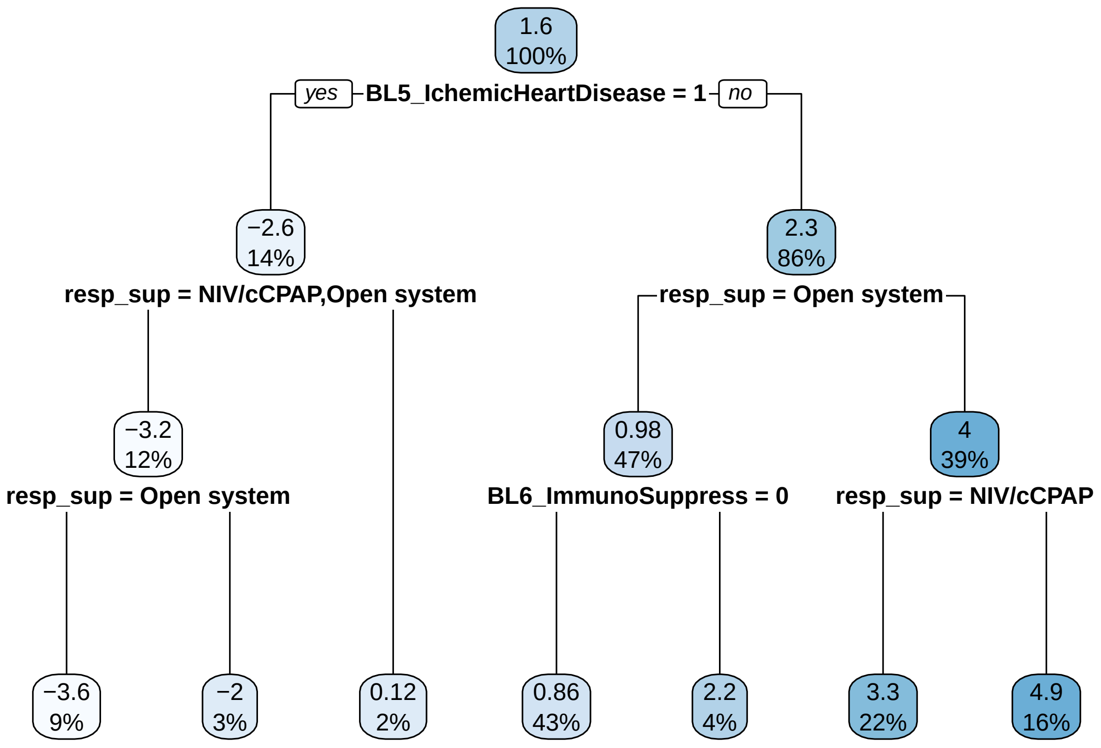
```
print("Number of covariates:")
```

```
## [1] "Number of covariates:"
```

```
print(numcovars - 1)
```

```
## [1] 3
```

```
cartmod <- rpart(cate_c ~ ., data = dat[, c(covar_include, 15)],
                     method = "anova")
rpart.plot(cartmod)
```

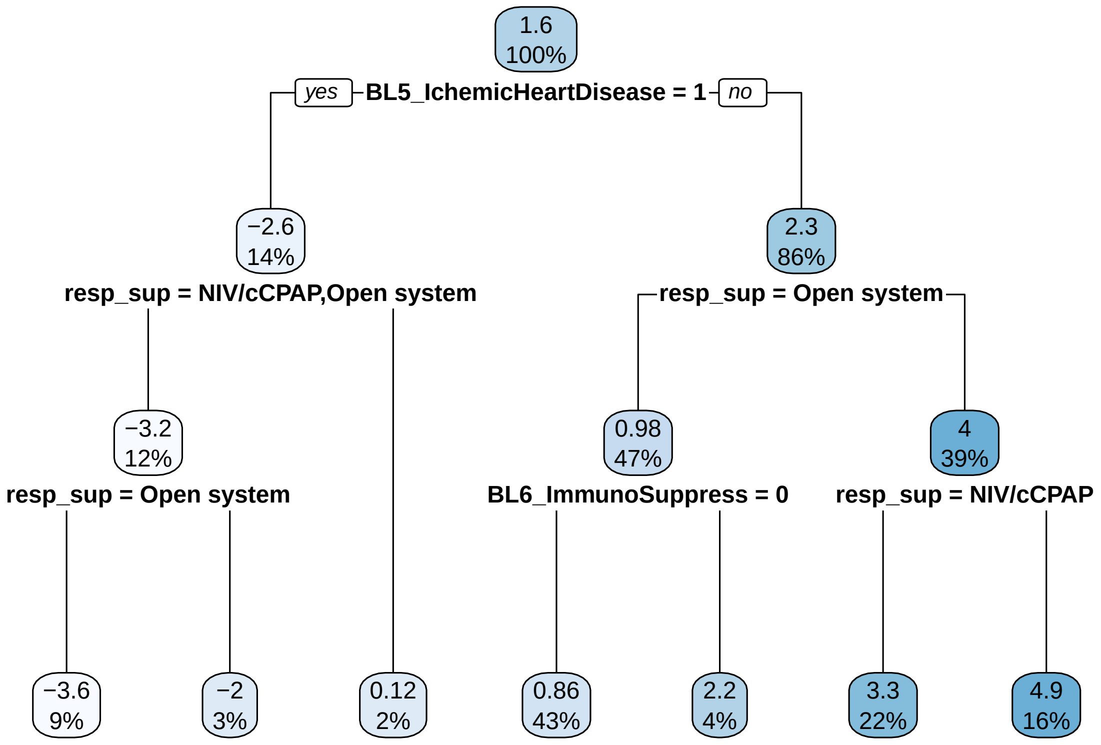


Finally we explore further pruning:

```r
printcp(cartmod)
```

```
##
## Regression tree:
## rpart(formula = cate_c ~ ., data = dat[, c(covar_include, 15)],
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] BL5_IchemicHeartDisease BL6_ImmunoSuppress      resp_sup
##
## Root node error: 5704.3/982 = 5.8089
##
## n= 982
##
##         CP nsplit rel error   xerror      xstd
## 1 0.522422      0  1.000000 1.002207 0.0468492
## 2 0.322677      1  0.477578 0.479924 0.0162823
## 3 0.040190      2  0.154902 0.156761 0.0091151
## 4 0.038512      3  0.114712 0.125554 0.0084035
## 5 0.011526      4  0.076200 0.078842 0.0042233
## 6 0.010087      5  0.064674 0.071597 0.0038845
## 7 0.010000      6  0.054586 0.065145 0.0036343
```

```r
prunedtree <-
  prune(cartmod,
        cp = cartmod$cptable[which.min(cartmod$cptable[, "xerror"]), "CP"])
rpart.plot(prunedtree)
```


1.6
100%
yes
BL5_IchemicHeartDisease = 1
no
−2.6
14%
resp_sup = NIV/cCPAP,Open system
2.3
86%
resp_sup = Open system
−3.2
12%
resp_sup = Open system
0.98
47%
BL6_ImmunoSuppress = 0
4
39%
resp_sup = NIV/cCPAP
−3.6
9%
−2
3%
0.12
2%
0.86
43%
2.2
4%
3.3
22%
4.9
16%