BCF-IV function

Falco J. Bargagli-Stoffi

02/09/2019

Details on the BCF-IV function

In the following chunck of code is depicted the `bcf-iv` function.

The function takes as inputs:

- `y`: the outcome variable;
- `w`: the reception of the treatment variable (binary);
- `z`: the assignment to the treatment variable (binary);
- `max_depth`: the maximal depth of the tree generated by the function;
- `n_burn`: the number of iterations discarded by the `BCF` algorithm for the *burn-in*;
- `n_sim`: the number of iterations used by the `BCF` algorithm to get the posterior distribution of the estimands;
- `binary`: this option should be set to `TRUE` when the outcome variable is binary (i.e., $y \in \{0, 1\}$) and to `FALSE` if the outcome variable is either discrete or continuous[1].

Probit BCF-IV

The default BCF algorithm by Hahn et al. (2017) is implemented just for continuous outcomes. However, Chipman et al. (2010) and Starling et al. (2019) propose probit versions of the BART and BCF algorithms, respectively, to handle binary outcomes. The new algorithm is described in detail in the following.

The original BCF-IV objective function that we propose is the following:

$$y_i = \mu(x_i, \hat{\pi}(x_i)) + ITT_Y(x_i) \cdot z_i. \tag{1}$$

If we let y be a Gaussian latent variable, then, following the original BART paper by Chipman et al. (2010) and the recent paper by Starling et al. (2019):

$$P(y_i = 1 | z_i, x_i) = \phi(\mu(x_i, \hat{\pi}(x_i)) + ITT_Y(x_i) \cdot z_i), \tag{2}$$

where ϕ is the standard normal CDF. Hence, the counterfactual probabilities are:

$$\omega_i(0) = \phi(\mu(x_i, \hat{\pi}(x_i))) \tag{3}$$

$$\omega_i(1) = \phi(\mu(x_i, \hat{\pi}(x_i)) + ITT_Y(x_i) \cdot z_i), \tag{4}$$

and $y_i \in 0, 1$ is:

$$y_i = \begin{cases} 1 \text{ if } P(y_i = 1 | z_i, x_i) \geq 0.5 \\ 0 \text{ if } P(y_i = 1 | z_i, x_i) < 0.5. \end{cases} \tag{5}$$

[1] The default ofption is `binary = FALSE`.

Method-of-Moments BCF-IV

In the original `bcf-iv` algorithm we implement an IV estimator that runs a 2SLS that controls for the covariates in any single node. In order to reproduce the IV method-of-moments estimator described in the paper (and in order to get the estimation of ITT and the complier subpopulations) we implemented the `mm_bcf_iv` algorithm that is the 2SLS equivalent of the method-of-moment estimator for CACE (namely, we do not control for the covariates in the nodes). We suggest to use this algorithm whenver we can suppose that the unconfoundedness assumption holds within any sub-populations and it is possible to rule out to control for the confounders in the sub-population.

```
bcf_iv <- function(y, w, z, x, max_depth, n_burn, n_sim, binary = FALSE) {

  # Upload the Packages

  require(bcf)
  require(rpart)
  require(lattice)
  require(rattle)
  require(AER)

  ###########################################################
  ####         Step 0: Initialize the Data           ####
  ###########################################################

  iv.data <- as.data.frame(cbind(y, w, z, x))
  names(iv.data)[1:3] <- c("y", "w", "z")

  ###########################################################
  ####  Step 1: Compute the Bayesian Causal Forest  ####
  ###########################################################

  # Compute the Propensity Score though a Logistic Regression

  p.score <- glm(z ~ x,
                 family = binomial,
                 data = iv.data)
  pihat <- predict(p.score, as.data.frame(x))

  # Perform the Bayesian Causal Forest for the ITT

  bcf_fit = invisible(bcf(y, z, x, x, pihat, nburn=n_burn, nsim=n_sim))

  ###########################################################
  ####     Continuous and Discrete Outcomes          ####
  ###########################################################

  if (binary == FALSE){


    # Get posterior of treatment effects

    tau_post = bcf_fit$tau
    tauhat = colMeans(tau_post)
    exp <- as.data.frame(cbind(tauhat, y, x, z))
```

```r
#########################################################
####  Step 2: Build a CART on the Unit Level CITT ####
#########################################################

fit.tree <- rpart(tauhat ~ x,
                  data = exp,
                  maxdepth = max_depth)

#########################################################
####  Step 3: Extract the Causal Rules (Nodes)    ####
#########################################################

rules <- as.numeric(row.names(fit.tree$frame[fit.tree$numresp]))

# Exclude the Root

rules <- rules[-1]

#########################################################
####  Step 4: Run an IV Regression on each Node   ####
#########################################################

# Run an IV Regression on the Root

iv.root <- ivreg(y ~ w + x | z + x,
                 data = iv.data)
summary <- summary(iv.root, diagnostics = TRUE)
iv.effect.root <-  summary$coef[2,1]
p.value.root <- summary$coef[2,4]
p.value.weak.iv.root <- summary$diagnostics[1,4]

# Print Root Results

cat(paste("The effect on the overall sample is", round(iv.effect.root, 4)),"\n")
cat(paste("P-value", p.value.root),"\n")
cat(paste("P-value Weak-Instrument Test", p.value.weak.iv.root),"\n")
cat(paste("Proportion of observations in the node: ", "1.00"),"\n")

# Initialize New Data

names(iv.data) <- paste("x", names(iv.data), sep="")

# Run a loop to get the rules (sub-populations)

for (i in rules){

  # Initialize Data

  attach(iv.data)
  xx <- x

  # Create a Vector to Store all the Dimensions of a Rule
```

```
    sub <- as.data.frame(matrix(NA, nrow = 1,
                                ncol = nrow(as.data.frame(path.rpart(fit.tree,node=i)))-1))

    invisible(capture.output(for (j in 1:ncol(sub)){

      # Store each Rule as a Sub-population

      sub[,j] <- as.character(print(as.data.frame(path.rpart(fit.tree,node=i))[j+1,1]))
      sub_pop <- noquote(paste(sub , collapse = " & "))
    }))


    subset <- iv.data[which(eval(parse(text=sub_pop))),]
    xx <- xx[which(eval(parse(text=sub_pop))),]

    # Run the IV Regression

    if (length(unique(subset$xw))!= 1 | length(unique(subset$xz))!= 1){

    iv.reg <- ivreg(xy ~ xw + xx | xz + xx,
                    data = subset)
    summary <- summary(iv.reg, diagnostics = TRUE)
    iv.effect <-  summary$coef[2,1]
    p.value <- summary$coef[2,4]
    p.value.weak.iv <- summary$diagnostics[1,4]

    # Proportion of observations in the node

    proportion.node <- nrow(subset)/nrow(iv.data)

    ##########################################################
    ####   Step 5: Output the Values of each CCACE   ####
    ##########################################################

    cat(paste("The conditional effect on the subpopulation is", round(iv.effect, 4)),"\n")
    cat(paste("P-value", p.value),"\n")
    cat(paste("P-value Weak-Instrument Test", p.value.weak.iv),"\n")
    cat(paste("Proportion of observations in the node: ", proportion.node),"\n")

    }

    # Delete data

    rm(subset)
    rm(xx)
    detach(iv.data)

  }
}

#########################################################
####           Binary Outcomes (Probit)           ####
#########################################################
```

```r
if (binary == TRUE){


  # Get posterior of treatment effects

  tau_post = bcf_fit$tau
  tauhat = colMeans(tau_post)

  # Probit Transformation of the Outcome (as in Starling et al. 2019)

  tauhat <- pnorm(tauhat)
  tauhat <- ifelse(tauhat>=0.5, 1, 0)
  exp <- as.data.frame(cbind(tauhat, y, x, z))

  ##########################################################
  ####  Step 2: Build a CART on the Unit Level CITT ####
  ##########################################################

  fit.tree <- rpart(tauhat ~ x,
                    data = exp,
                    maxdepth = max_depth)

  ##########################################################
  ####  Step 3: Extract the Causal Rules (Nodes)    ####
  ##########################################################

  rules <- as.numeric(row.names(fit.tree$frame[fit.tree$numresp]))

  # Exclude the Root

  rules <- rules[-1]


  ##########################################################
  ####  Step 4: Run an IV Regression on each Node   ####
  ##########################################################

  # Run an IV Regression on the Root

  iv.root <- ivreg(y ~ w + x | z + x,
                   data = iv.data)
  summary <- summary(iv.root, diagnostics = TRUE)
  iv.effect.root <-  summary$coef[2,1]
  p.value.root <- summary$coef[2,4]
  p.value.weak.iv.root <- summary$diagnostics[1,4]

  # Print Root Results

  cat(paste("The effect on the overall sample is", round(iv.effect.root, 4)),"\n")
  cat(paste("P-value", p.value.root),"\n")
  cat(paste("P-value Weak-Instrument Test", p.value.weak.iv.root),"\n")
  cat(paste("Proportion of observations in the node: ", "1.00"),"\n")
```

```
# Initialize New Data

names(iv.data) <- paste("x", names(iv.data), sep="")

# Run a loop to get the rules (sub-populations)

for (i in rules){

  # Initialize Data

  attach(iv.data)
  xx <- x

  # Create a Vector to Store all the Dimensions of a Rule

  sub <- as.data.frame(matrix(NA, nrow = 1,
                              ncol = nrow(as.data.frame(path.rpart(fit.tree,node=i)))-1))

  invisible(capture.output(for (j in 1:ncol(sub)){

    # Store each Rule as a Sub-population

    sub[,j] <- as.character(print(as.data.frame(path.rpart(fit.tree,node=i))[j+1,1]))
    sub_pop <- noquote(paste(sub , collapse = " & "))
  }))

  subset <- iv.data[which(eval(parse(text=sub_pop))),]
  xx <- xx[which(eval(parse(text=sub_pop))),]

  # Run the IV Regression

  if (length(unique(subset$xw))!= 1 | length(unique(subset$xz))!= 1){

  iv.reg <- ivreg(xy ~ xw + xx | xz + xx,
                  data = subset)
  summary <- summary(iv.reg, diagnostics = TRUE)
  iv.effect <-  summary$coef[2,1]
  p.value <- summary$coef[2,4]
  p.value.weak.iv <- summary$diagnostics[1,4]

  # Proportion of observations in the node

  proportion.node <- nrow(subset)/nrow(iv.data)

  ##########################################################
  ####   Step 5: Output the Values of each CCACE   ####
  ##########################################################

  cat(paste("The conditional effect on the subpopulation is", round(iv.effect, 4)),"\n")
  cat(paste("P-value", p.value),"\n")
  cat(paste("P-value Weak-Instrument Test", p.value.weak.iv),"\n")
  cat(paste("Proportion of observations in the node: ", proportion.node),"\n")
```

```
      }

      # Delete data

      rm(subset)
      rm(xx)
      detach(iv.data)
    }
  }

}

mm_bcf_iv <- function(y, w, z, x, max_depth, n_burn, n_sim, binary = FALSE) {

  # Upload the Packages

  require(bcf)
  require(rpart)
  require(lattice)
  require(rattle)
  require(AER)

  ############################################################
  ####         Step 0: Initialize the Data          ####
  ############################################################

  iv.data <- as.data.frame(cbind(y, w, z, x))
  names(iv.data)[1:3] <- c("y", "w", "z")

  ############################################################
  ####  Step 1: Compute the Bayesian Causal Forest  ####
  ############################################################

  # Compute the Propensity Score though a Logistic Regression

  p.score <- glm(z ~ x,
                 family = binomial,
                 data = iv.data)
  pihat <- predict(p.score, as.data.frame(x))

  # Perform the Bayesian Causal Forest for the ITT

  bcf_fit = invisible(bcf(y, z, x, x, pihat, nburn=n_burn, nsim=n_sim))

  ############################################################
  ####     Continuous and Discrete Outcomes         ####
  ############################################################

  if (binary == FALSE){


    # Get posterior of treatment effects
```

```
tau_post = bcf_fit$tau
tauhat = colMeans(tau_post)
exp <- as.data.frame(cbind(tauhat, y, x, z))

#########################################################
####  Step 2: Build a CART on the Unit Level CITT ####
#########################################################

fit.tree <- rpart(tauhat ~ x,
                  data = exp,
                  maxdepth = max_depth)

#########################################################
####  Step 3: Extract the Causal Rules (Nodes)    ####
#########################################################

rules <- as.numeric(row.names(fit.tree$frame[fit.tree$numresp]))

# Exclude the Root

rules <- rules[-1]

#########################################################
####  Step 4: Run an IV Regression on each Node   ####
#########################################################

# Run an IV Regression on the Root

iv.root <- ivreg(y ~ w | z,
                 data = iv.data)
summary <- summary(iv.root, diagnostics = TRUE)
iv.effect.root <-  summary$coef[2,1]
p.value.root <- summary$coef[2,4]
p.value.weak.iv.root <- summary$diagnostics[1,4]
compliers.root <- length(which(z==w))/nrow(iv.data)
itt.root <- iv.effect.root*compliers.root

# Print Root Results

cat(paste("The effect on the overall sample is", round(iv.effect.root, 4)),"\n")
cat(paste("P-value", p.value.root),"\n")
cat(paste("P-value Weak-Instrument Test", p.value.weak.iv.root),"\n")
cat(paste("Proportion of observations in the node: ", "1.00"),"\n")
cat(paste("Intention-to-treat effect: ", itt.root),"\n")
cat(paste("Proportion of compliers in the node: ", compliers.root),"\n")

# Initialize New Data

names(iv.data) <- paste("x", names(iv.data), sep="")

# Run a loop to get the rules (sub-populations)

for (i in rules){
```

```
# Initialize Data

attach(iv.data)
xx <- x

# Create a Vector to Store all the Dimensions of a Rule

sub <- as.data.frame(matrix(NA, nrow = 1,
                            ncol = nrow(as.data.frame(path.rpart(fit.tree,node=i)))-1))

invisible(capture.output(for (j in 1:ncol(sub)){

  # Store each Rule as a Sub-population

  sub[,j] <- as.character(print(as.data.frame(path.rpart(fit.tree,node=i))[j+1,1]))
  sub_pop <- noquote(paste(sub , collapse = " & "))
}))

subset <- iv.data[which(eval(parse(text=sub_pop))),]
xx <- xx[which(eval(parse(text=sub_pop))),]


# Run the IV Regression

if (length(unique(subset$xw))!= 1 | length(unique(subset$xz))!= 1){

iv.reg <- ivreg(xy ~ xw | xz,
                data = subset)
summary <- summary(iv.reg, diagnostics = TRUE)
iv.effect <-  summary$coef[2,1]
p.value <- summary$coef[2,4]
p.value.weak.iv <- summary$diagnostics[1,4]
compliers <- length(which(subset$xz==subset$xw))/nrow(subset)
itt <- iv.effect*compliers

# Proportion of observations in the node

proportion.node <- nrow(subset)/nrow(iv.data)

##########################################################
####   Step 5: Output the Values of each CCACE   ####
##########################################################

cat(paste("The conditional effect on the subpopulation is", round(iv.effect, 4)),"\n")
cat(paste("P-value", p.value),"\n")
cat(paste("P-value Weak-Instrument Test", p.value.weak.iv),"\n")
cat(paste("Proportion of observations in the node: ", proportion.node),"\n")
cat(paste("Intention-to-treat effect: ", itt),"\n")
cat(paste("Proportion of compliers in the node: ", compliers),"\n")

}

# Delete data
```

```
    rm(subset)
    rm(xx)
    detach(iv.data)

  }
}

#########################################################
####           Binary Outcomes (Probit)           ####
#########################################################

if (binary == TRUE){


  # Get posterior of treatment effects

  tau_post = bcf_fit$tau
  tauhat = colMeans(tau_post)

  # Probit Transformation of the Outcome (as in Starling et al. 2019)

  tauhat <- pnorm(tauhat)
  tauhat <- ifelse(tauhat>=0.5, 1, 0)
  exp <- as.data.frame(cbind(tauhat, y, x, z))

  ########################################################
  ####  Step 2: Build a CART on the Unit Level CITT ####
  ########################################################

  fit.tree <- rpart(tauhat ~ x,
                    data = exp,
                    maxdepth = max_depth)

  ########################################################
  ####  Step 3: Extract the Causal Rules (Nodes)    ####
  ########################################################

  rules <- as.numeric(row.names(fit.tree$frame[fit.tree$numresp]))

  # Exclude the Root

  rules <- rules[-1]


  ########################################################
  ####  Step 4: Run an IV Regression on each Node   ####
  ########################################################

  # Run an IV Regression on the Root

  iv.root <- ivreg(y ~ w | z,
                   data = iv.data)
  summary <- summary(iv.root, diagnostics = TRUE)
```

```
  iv.effect.root <-  summary$coef[2,1]
  p.value.root <- summary$coef[2,4]
  p.value.weak.iv.root <- summary$diagnostics[1,4]
  compliers.root <- length(which(z==w))/nrow(iv.data)
  itt.root <- iv.effect.root*compliers.root

  # Print Root Results

  cat(paste("The effect on the overall sample is", round(iv.effect.root, 4)),"\n")
  cat(paste("P-value", p.value.root),"\n")
  cat(paste("P-value Weak-Instrument Test", p.value.weak.iv.root),"\n")
  cat(paste("Proportion of observations in the node: ", "1.00"),"\n")
  cat(paste("Intention-to-treat effect: ", itt.root),"\n")
  cat(paste("Proportion of compliers in the node: ", compliers.root),"\n")

  # Initialize New Data

  names(iv.data) <- paste("x", names(iv.data), sep="")

  # Run a loop to get the rules (sub-populations)

  for (i in rules){

    # Initialize Data

    attach(iv.data)
    xx <- x

    # Create a Vector to Store all the Dimensions of a Rule

    sub <- as.data.frame(matrix(NA, nrow = 1,
                                ncol = nrow(as.data.frame(path.rpart(fit.tree,node=i)))-1))

    invisible(capture.output(for (j in 1:ncol(sub)){

      # Store each Rule as a Sub-population

      sub[,j] <- as.character(print(as.data.frame(path.rpart(fit.tree,node=i))[j+1,1]))
      sub_pop <- noquote(paste(sub , collapse = " & "))
    }))


    subset <- iv.data[which(eval(parse(text=sub_pop))),]
    xx <- xx[which(eval(parse(text=sub_pop))),]

    # Run IV Regression

    if (length(unique(subset$xw))!= 1 | length(unique(subset$xz))!= 1){

    iv.reg <- ivreg(xy ~ xw | xz,
                    data = subset)
    summary <- summary(iv.reg, diagnostics = TRUE)
    iv.effect <-  summary$coef[2,1]
```

```
      p.value <- summary$coef[2,4]
      p.value.weak.iv <- summary$diagnostics[1,4]
      compliers <- length(which(subset$xz==subset$xw))/nrow(subset)
      itt <- iv.effect*compliers

      # Proportion of observations in the node

      proportion.node <- nrow(subset)/nrow(iv.data)

      ############################################################
      ####   Step 5: Output the Values of each CCACE   ####
      ############################################################

      cat(paste("The conditional effect on the subpopulation is", round(iv.effect, 4)),"\n")
      cat(paste("P-value", p.value),"\n")
      cat(paste("P-value Weak-Instrument Test", p.value.weak.iv),"\n")
      cat(paste("Proportion of observations in the node: ", proportion.node),"\n")
      cat(paste("Intention-to-treat effect: ", itt),"\n")
      cat(paste("Proportion of compliers in the node: ", compliers),"\n")

      }

      # Delete data

      rm(subset)
      rm(xx)
      detach(iv.data)
    }
  }

}
```