pset 6 675

```
#======================#
# ==== PS 6 Metrics ====
#======================#

#=======================================#
# ==== Load packages clear workspace ====
#=======================================#

# clear workspace
rm(list = ls(pos = ".GlobalEnv"), pos = ".GlobalEnv")
options(scipen = 999)
cat("\f")

# laod packages
library(data.table)          # helps do everything faster and better
library(ggplot2)             # for pretty plots
library(xtable)              # for latex tables
library(rdrobust)            # for RD plots and other stuff
library(rddensity)           # for RD density continuity tests
library(rdlocrand)           # for RD randomization inference
library(grid)
library(gridGraphics)
library(ggplotify)           # use this to fix the wierd graphs in rdrobust
library(broom)
library(sandwich)
# output folder
f_out <- "c:/Users/Nmath_000/Documents/Code/courses/econ 675/ps_6_tex/"

# plot attributes
plot_attributes <- theme( plot.background = element_rect(fill = "lightgrey"),
                          panel.grid.major.x = element_line(color = "gray90"),
                          panel.grid.minor  = element_blank(),
                          panel.background = element_rect(fill = "white", colour = "black") ,
                          panel.grid.major.y = element_line(color = "gray90"),
                          text = element_text(size= 20),
                          plot.title = element_text(vjust=0, hjust = 0.5, colour = "#0B6357",face = "bo


#=====================#
# ==== Question 2 ====
#=====================#

# load data
hs <- fread("c:/Users/Nmath_000/Documents/MI_school/Second Year/675 Applied Econometrics/hw/hw6/HeadSta
hs

#================#
# ==== Q 2.1 ====
#================#
```

```
#=================#
# ==== Q2.1.1 ====
#=================#

  # Evenly-spaced bins, IMSE optimal
  rdplot(hs[,mort_related_pre],
         hs[,povrate60],
         c =0,
         p=1,
         binselect = "es",
        x.label="povrate60",
        y.label="mort_related_pre",
        title="Evenly-spaced bins, IMSE optimal")

  # I guess do this because this since I can't make ggplots with this function
  dev.copy(pdf, paste0(f_out, 'plot_211ia.pdf'))
  dev.off()

  # Evenly-spaced bins, mimicking variance
  rdplot(hs[,mort_related_pre],
         hs[,povrate60],
         p=1,
         binselect = "esmv",
         x.label="povrate60",
         y.label="mort_related_pre",
         title="Evenly-spaced bins, mimicking variance ")

  dev.copy(pdf, paste0(f_out, 'plot_211ib.pdf'))
  dev.off()

  # Quantile-spaced bins, IMSE optimal
  rdplot(hs[,mort_related_pre],
         hs[,povrate60],
         p=1,
         binselect = "qs",
         x.label="povrate60",
         y.label="mort_related_pre",
         title="Quantile-spaced bins, IMSE optimal")

  dev.copy(pdf, paste0(f_out, 'plot_211iia.pdf'))
  dev.off()

  # Quantile-spaced bins, mimicking variance
  rdplot(hs[,mort_related_pre],
         hs[,povrate60],
         p=1,
         binselect = "qsmv",
         x.label="povrate60",
         y.label="mort_related_pre",
         title="Quantile-spaced bins, mimicking variance")

  dev.copy(pdf, paste0(f_out, 'plot_211iib.pdf'))
  dev.off()
```

```r
#==================#
# ==== Q 2.1.2 ====
#==================#

##### i Histogram ###
   # add above below zero flag
  hs[povrate60 >= 0, f_cut := "Above Cutoff"]
  hs[povrate60 < 0, f_cut := "Below Cutoff"]

  # make a histogram before and after cutoff
  plot_2.1.2.i <- ggplot(hs, aes(povrate60)) +
                  geom_histogram(aes(fill = f_cut), breaks = seq(-50,25,2)) +
                  xlab("60 - Poverty rate") +
                  ylab("Count") + ggtitle("Histogram of Running Variable") +
                  scale_fill_discrete(name = "RD Group")
  # check it out
  plot_2.1.2.i

  # then add atrribute that make it look good once save
  plot_2.1.2.i <- plot_2.1.2.i + plot_attributes

## ii local binomial test ###
# make a grid of bandwidiths to test
bi_test <- data.table(bandwidth = seq(.4,4,.2))

# get number above and below cutoff for each bandwidth
bi_test[,below_c := nrow(hs[abs(povrate60) <= bandwidth/2 & f_cut == "Below Cutoff"]), bandwidth]
bi_test[,above_c := nrow(hs[abs(povrate60) <= bandwidth/2 & f_cut == "Above Cutoff"]), bandwidth]
bi_test[,total := below_c + above_c]

# do binomial tests
bi_test[, bin_test := binom.test(below_c, total, .5)$p.value, bandwidth]

# make this table for hw
table_2.1.2.ii <- bi_test[, -c("total")]
colnames(table_2.1.2.ii) <- c("Bandwidth",
                              "Number Below Cutoff",
                              "Number Above Cutoff",
                              "binomial P Valu")

## iii continuity in design tests
## Continuity in density tests (defaults are triangular kernel, jackknife SEs)
cdt <- rddensity(hs$povrate60)

# save plot
png(paste0(f_out, "plot_212i.png"),
    height = 400,
    width = 800,
    type = "cairo")
print(plot_2.1.2.i)
dev.off()

# save table
```

```
  print(xtable(table_2.1.2.ii, type = "latex"),
        file = paste0(f_out, "table_212ii.tex"),
        include.rownames = FALSE,
        floating = FALSE)

#================#
# ==== Q 2.2 ====
#================#

  #==================#
  # ==== Q 2.2.1 ====
  #==================#

  # create polynomials
  p_dt <- as.data.table(poly(hs$povrate60, 6))
  colnames(p_dt) <- paste0("poly_pov60_", colnames(p_dt))
  p_dt[,poly_pov60_1 := NULL ]
  hs <- cbind(hs, p_dt)
  hs[, treatment := as.numeric(f_cut == "Above Cutoff")]

  # initialize data
  table_2.2.1 <- data.table(value = c("Estimate", "Standard Error"))

  # write loop  to make everything we need for polynomial of order N
  poly_n <- 3
  for(poly_n in 3:6){

    # get x variables we need, there is a better way to do this out this works fine
    x_vars <- c("povrate60",
                "treatment",
                grep(paste(as.character(c(1:poly_n)), collapse = "|"),
                     colnames(hs),
                     value = TRUE))

    # make formula
    reg_form <- as.formula(paste0("mort_related_post ~", paste(x_vars, collapse = " + ")))

    # run regressin
    reg_o1 <- lm(reg_form, data = hs)
    reg_o <- data.table(tidy(reg_o1))
    tab_col <- reg_o[term == "treatment", c(estimate, std.error)]

    # put stuff in table
    table_2.2.1[, temp := tab_col]
    setnames(table_2.2.1, "temp", paste0("Polynomial ", poly_n))

    # get fitted values and data
    temp.rd = rdplot(hs[,mort_related_post], hs[,povrate60] ,hide=TRUE)

    temp.rd_dt <- data.table(rdplot_mean_x = temp.rd$vars_bins$rdplot_mean_x,
                             rdplot_mean_y =  temp.rd$vars_bins$rdplot_mean_y)

    fitted_dt <- data.table(x = hs$povrate60, y = reg_o1$fitted.values)
```

```
  # make plot
  t_plot <- ggplot() + geom_point(data = temp.rd_dt,
                                  aes(x = rdplot_mean_x, y = rdplot_mean_y, color = "Binned Values"))

  t_plot <- t_plot + geom_point(data = fitted_dt,
                                aes(x = x, y = y, color = "Fitted Values")) +
            geom_vline(xintercept = 0)

  t_plot <- t_plot + xlab("60 - Poverty rate") +
            ylab("HS Related Mortality") +
            scale_color_manual(values = c("black", "blue")) +
            theme(legend.title=element_blank())

  t_plot <- t_plot + ggtitle(paste0("Fitted Values for Polynomial of degree ", poly_n)) +
            plot_attributes
  t_plot

  # save plot
  png(paste0(f_out,
             "plot_221_poly_",
             poly_n,
             ".png"),
      height = 400,
      width = 800,
      type = "cairo")

  print(t_plot)
  dev.off()

}

#==================#
# ==== Q 2.2.2 ====
#==================#

  # Make interaction terms
  int_dt <- as.data.table(poly(hs$povrate60, 6))
  colnames(int_dt) <- paste0("treat_poly_pov60_", colnames(int_dt))
  int_dt[,treat_poly_pov60_1 := NULL ]
  hs <- cbind(hs, int_dt)
  cols <- grep("treat_poly", colnames(hs), value = TRUE)
  hs[, (cols) := lapply(.SD, function(x) x*treatment), .SDcols = cols]

  # initialize data
  table_2.2.2 <- data.table(value = c("Estimate", "Standard Error"))

  # write loop  to make everything we need for polynomial of order N
  poly_n <- 3
  for(poly_n in 3:6){

    # get x variables we need, there is a better way to do this out this works fine
    x_vars <- c("povrate60",
                "treatment",
```

```
            grep(paste(as.character(c(1:poly_n)), collapse = "|"),
                 colnames(hs),
                 value = TRUE))

# make formula
reg_form <- as.formula(paste0("mort_related_post ~", paste(x_vars, collapse = " + ")))

# run regressin
reg_o1 <- lm(reg_form, data = hs)
reg_o <- data.table(tidy(reg_o1))
tab_col <- reg_o[term == "treatment", c(estimate, std.error)]

# put stuff in table
table_2.2.2[, temp := tab_col]
setnames(table_2.2.2, "temp", paste0("Polynomial ", poly_n))

# get fitted values and data
temp.rd = rdplot(hs[,mort_related_post], hs[,povrate60] ,hide=TRUE)
temp.rd_dt <- data.table(rdplot_mean_x = temp.rd$vars_bins$rdplot_mean_x,
                         rdplot_mean_y =  temp.rd$vars_bins$rdplot_mean_y)
fitted_dt <- data.table(x = hs$povrate60, y = reg_o1$fitted.values)

# make plot
t_plot <- ggplot() + geom_point(data = temp.rd_dt,
                                aes(x = rdplot_mean_x,
                                    y = rdplot_mean_y,
                                    color = "Binned Values"))

t_plot <- t_plot + geom_point(data = fitted_dt,
                              aes(x = x,
                                  y = y,
                                  color = "Fitted Values")) + geom_vline(xintercept = 0)

t_plot <- t_plot + xlab("60 - Poverty rate") +
  ylab("HS Related Mortality") + scale_color_manual(values = c("black", "blue")) +
  theme(legend.title=element_blank())

t_plot <- t_plot +
  ggtitle(paste0("Fitted Values for Polynomial of degree ", poly_n, " With Interactions")) +
  plot_attributes

t_plot

# save plot
png(paste0(f_out,
           "plot_222_poly_",
           poly_n,
           ".png"),
    height = 400,
    width = 800,
    type = "cairo")

print(t_plot)
```

```
    dev.off()

  }

#=================#
# ==== q2.2.3 ====
#=================#

  # drop the interaction terms
  hs <- hs[,
           grep( "treat_poly",
                 colnames(hs),
                 invert = TRUE,
                 value = TRUE),
           with = FALSE]


  in_dt <- hs
  F_223 <- function(p, bw, in_dt = hs){

    # get x variables we need, there is a better way to do this out this works fine
    x_vars <- c("povrate60",
                "treatment",
                grep(paste(as.character(c(1:p)), collapse = "|"),
                     colnames(hs), value = TRUE))

    # subset data down to appropriate binwidth
    w_dt <- in_dt[abs(povrate60) <= bw]

    # make formula
    reg_form <- as.formula(paste0("mort_related_post ~", paste(x_vars, collapse = " + ")))

    # run regressin
    reg_o1 <- lm(reg_form, data = w_dt)
    reg_o <- data.table(tidy(reg_o1))
    tab_col <- reg_o[term == "treatment", c(estimate, std.error)]

    # put in table
    out_tab <-  data.table(value = c("Estimate", "Standard Error"), temp = tab_col)
    setnames(out_tab, "temp", paste0("Polynomial ", p))

    # get fitted values and data
    temp.rd = rdplot(w_dt[,mort_related_post], w_dt[,povrate60] ,hide=TRUE)
    temp.rd_dt <- data.table(rdplot_mean_x = temp.rd$vars_bins$rdplot_mean_x,
                             rdplot_mean_y =  temp.rd$vars_bins$rdplot_mean_y)
    fitted_dt <- data.table(x = w_dt$povrate60, y = reg_o1$fitted.values)

    # make plot
    t_plot <- ggplot() + geom_point(data = temp.rd_dt, aes(x = rdplot_mean_x, y = rdplot_mean_y, colo
    t_plot <- t_plot + geom_point(data = fitted_dt, aes(x = x, y = y, color = "Fitted Values"))  + ge
    t_plot <- t_plot + xlab("60 - Poverty rate") + ylab("HS Related Mortality") + scale_color_manual(
    t_plot <- t_plot + ggtitle(paste0("Fitted Values for Polynomial of degree ", p, " Bin Width ", bw
    t_plot
```

```
      # save plot
      png(paste0(f_out, "plot_223_poly_", p, "_bw_", bw, ".png"), height = 400, width = 800, type = "ca
      print(t_plot)
      dev.off()

      # return table
      return(out_tab)

      }

      # lapply over different polynomials for each bw
      bw1 <- Reduce( function(x,y) merge(x, y, by = "value"), lapply(1:3, F_223, bw = 1))
      bw5 <- Reduce( function(x,y) merge(x, y, by = "value"), lapply(1:3, F_223, bw = 5))
      bw9 <- Reduce( function(x,y) merge(x, y, by = "value"), lapply(1:3, F_223, bw = 9))
      bw18 <- Reduce( function(x,y) merge(x, y, by = "value"), lapply(1:3, F_223, bw = 18))

      # save tables
      tabs <- grep("bw", ls(), value = TRUE)
      for(tab_i in tabs){

        print(xtable(get(tab_i), type = "latex"),
              file = paste0(f_out, "table_223_", tab_i, ".tex"),
              include.rownames = FALSE,
              floating = FALSE)

      }

#==============#
# ==== 2.3 ====
#==============#

  #================#
  # ==== 2.3.1 ====
  #================#

    # run this thing for 3 polynomials
    rd_regs <- lapply(0:2, function(i) rdrobust(y = hs[,mort_related_post], x = hs[,povrate60], p = i))

    # grab out results we need
    tables_231 <- lapply(1:3,
                         function(i) data.table(Reduce( function(x,y) cbind(x,y) ,
                                                        list(rd_regs[[i]]$coef,
                                                             rd_regs[[i]]$se,
                                                             rd_regs[[i]]$ci)),
                                                keep.rownames = TRUE))

    # rename rn and add in polynomial
    fun_231 <- function(i, in_dt){
      setnames(in_dt, "rn", "Estimator")
      in_dt[, polynomial := i]
      return(in_dt)
    }
    tables_231 <-  mapply(fun_231, 0:2, tables_231, SIMPLIFY = FALSE)
```

```
  # save them
  for(i in 1:3){

    print(xtable(tables_231[[i]], type = "latex"),
          file = paste0(f_out, "table_231_poly_", paste0(i-1), ".tex"),
          include.rownames = FALSE,
          floating = FALSE)

  }

#==================#
# ==== Q 2.3.2 ====
#==================#

  #=============#
  # ==== a ====
  #=============#

  # run placebo test with other variables
  pl_1 <- rdrobust(hs[, mort_related_pre], hs[,povrate60], p = 1)
  pl_2 <- rdrobust(hs[, mort_injury_post], hs[,povrate60], p = 1)

  table_232ai <- cbind(pl_1$coef, pl_1$se, pl_1$ci)
  table_232aii <- cbind(pl_2$coef, pl_2$se, pl_2$ci)


      print(xtable(table_232ai, type = "latex"),
        file = paste0(f_out, "table_232ai.tex"),
        include.rownames = TRUE,
        floating = FALSE)

      print(xtable(table_232aii, type = "latex"),
            file = paste0(f_out, "table_232aii.tex"),
            include.rownames = TRUE,
            floating = FALSE)


  #=============#
  # ==== b ====
  #=============#

  h_l <- data.table(h= c(1:10), m =1)

  kern_l <- data.table(kern = c("triangular", "uniform", "epanechnikov"), m =1)

  xwalk <- merge(h_l, kern_l, by = "m", allow.cartesian = TRUE)

  # function to run on all these
  f_232b <- function(h_i, kern_i){

    rdrobust(hs[, mort_related_post], hs[,povrate60], p = 1, h = h_i, kernel = kern_i)$coef[[2]]
  }
```

```r
# DO IT. JUST DO IT!!!! DOOOO IT (labeouf 2015)
xwalk[, estimate := mapply(f_232b, xwalk[, h], xwalk[, kern], SIMPLIFY = FALSE)]
xwalk[, m := NULL]

table_2.3.2b <- dcast.data.table(xwalk, kern ~ h, value.var = "estimate")

setnames(table_2.3.2b, "kern", "kernal")

cols_change <- grep("kern", colnames(table_2.3.2b), value = TRUE, invert = TRUE)
setnames(table_2.3.2b, cols_change, paste0("Bw = ", cols_change))

# Save it
print(xtable(table_2.3.2b, type = "latex"),
      file = paste0(f_out, "table_232b.tex"),
      include.rownames = TRUE,
      floating = FALSE)

#=============#
# ==== c ====
#=============#

# sort data
hs_sort <- copy(hs)
hs_sort[, abs_pov60 := abs(povrate60)]
hs_sort <- setorder(hs_sort, abs_pov60)

# donut hole function
hole <- 1
do_hole <- function(hole = NULL){

  # remove hole
  w_dt <- hs_sort[-hole]

  # run the thing
  res <- rdrobust(w_dt[, mort_related_post], w_dt[,povrate60], p = 1)$coef[[2]]
  return(res)

  }

# run it on 1-10
table_2.3.2c <- data.table( l = c(1:10),
                            est = unlist(lapply(1:10, do_hole)),
                            "# obs dropped" = "estimate")
table_2.3.2c <- dcast.data.table(table_2.3.2c,
                                 `# obs dropped`~l,
                                 value.var = "est")

# Save it
print(xtable(table_2.3.2c, type = "latex"),
      file = paste0(f_out, "table_232c.tex"),
      include.rownames = TRUE,
      floating = FALSE)
```

```
    #=============#
    # ==== d ====
    #=============#

    cutoffs = seq(-10,10,2)

    # funciton
    c_fun <-  function(c_i){

    res <-  rdrobust(hs[, mort_related_post], hs[,povrate60], p = 1, c= c_i)
    out_t <- data.table(Statistic = c("Estimate", "p Value"),
                        temp = c(res$coef[[2]], res$pv[[2]]))
    setnames(out_t, "temp", paste0("c = ", c_i))

    }

    rd.cutoffs = lapply(cutoffs, c_fun)

    table_2.3.2d <- Reduce( function(x,y) merge(x, y, by = "Statistic"),rd.cutoffs)

    # Save it
    print(xtable(table_2.3.2d, type = "latex"),
          file = paste0(f_out, "table_232d.tex"),
          include.rownames = TRUE,
          floating = FALSE)


#================#
# ==== Q 2.4 ====
#================#

    # windows
    windows <- seq(.8, 2.6, .2)

    # function to do the stuff I need for a given window
    wind_i <- 1
    f_2.4 <- function(wind_i){

       # subset data to window
        w_dt <- hs[ abs(povrate60) <= wind_i, ]

      # run regression
        reg <- lm( mort_related_post~povrate60, data = w_dt)
        reg_t <- data.table(tidy(reg))

      # put in table
        tab_i <- data.table(Statistic = c("Estimate", "Std Error", "P-Value"),
                            w = as.numeric(reg_t[term == "povrate60",
                                                 c("estimate", "std.error", "p.value")]))
        setnames(tab_i, "w", paste0("w = ", wind_i))

      # return it
        return(tab_i)
```

```
  }

  # run it on all the windows
  table_2.4 <- lapply(windows, f_2.4)
  table_2.4 <- Reduce( function(x,y) merge(x, y, by = "Statistic"),table_2.4)

  #save it
  print(xtable(table_2.4, type = "latex"),
        file = paste0(f_out, "table_24.tex"),
        include.rownames = TRUE,
        floating = FALSE)


#============================#
# ==== save other tables ====
#============================#

print(xtable(table_2.2.1, type = "latex"),
      file = paste0(f_out, "table_221.tex"),
      include.rownames = FALSE,
      floating = FALSE)

print(xtable(table_2.2.2, type = "latex"),
      file = paste0(f_out, "table_222.tex"),
      include.rownames = FALSE,
      floating = FALSE)
```