Generalized IPW

Xiaoting Chen

2024-05-24

This file is translated from the Generalized_IPW.ipynb file and expanded on the spline function and prediction curve generation.

Prep

packages

```
library(readr)
library(tidyr)
library(ggplot2)
library(dplyr)
library(stats) # handle normal distribution calculations
library(splines) # using splines
library(mgcv)
library(DescTools) # calculate mode
library(tableone)
library(cobalt)
```

data

```
all_combos <- readRDS("all_combos.rds")
# subset of only mesa and jhs with cvd_10y_HF
all_combos <- all_combos[c(1:20, 81:100)]

# read in moderation and mediation results
significant_moderation <- read.csv("../results/moderation_result/moderation_result.csv") %>%
  filter(significant == "Y") %>%
  select(1:4) %>%
  rename(data_name = Data,
         M = moderator)

significant_mediation <- read.csv("../results/mediation_result/significant_mediat.csv")
```

```
ind_feature_map <- c('0' = 1, '1' = 2, '2' = 3)
```

```r
mesa_std <- read.csv('../data_processed/MESA/mesa_std.csv')

analysis_feature <- c('cvd_10y_HF', 'cvd_10y_noHF', 'nSES', 'nFavFood', 'nPhysFac', 'nRS',
                      'FamIncome', 'nutrition', 'PhysAct', 'currentSmoker', 'alc',
                      'age', 'gender', 'Diabetes', 'hdl', 'totchol', 'sbp',
                      'site', 'race')

mesa_std <- mesa_std %>%
  select(analysis_feature) %>%
  mutate(# # treat ind behavirol variables as conitnuous, avoid partial significant
    Diabetes = as.factor(Diabetes),
         site = as.factor(site),
         race = as.factor(race)) %>%
  mutate( # recode to avoid 0
    nutrition = recode(nutrition, !!!ind_feature_map),
    PhysAct = recode(PhysAct, !!!ind_feature_map),
    currentSmoker = recode(currentSmoker, !!!ind_feature_map),
    alc = recode(alc, !!!ind_feature_map)
  )
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(analysis_feature)
##
##   # Now:
##   data %>% select(all_of(analysis_feature))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```r
mesa_bla_std <- mesa_std[mesa_std$race == 1, ] %>%
  select(-race)
```

```r
jhs_std <- read.csv('../data_processed/JHS/jhs_std.csv')

analysis_feature <- c('cvd_10y_HF', 'cvd_10y_noHF', 'nSES', 'nFavFood', 'nPhysFac', 'nRS',
                      'FamIncome', 'nutrition', 'PhysAct', 'currentSmoker', 'alc',
                      'age', 'gender', 'Diabetes', 'hdl', 'totchol', 'sbp')

jhs_std <- jhs_std %>%
  select(analysis_feature) %>%
  mutate(# # treat ind behavirol variables as conitnuous, avoid partial significant
    Diabetes = as.factor(Diabetes)) %>%
  mutate( # recode to avoid 0
    nutrition = recode(nutrition, !!!ind_feature_map),
    PhysAct = recode(PhysAct, !!!ind_feature_map),
    currentSmoker = recode(currentSmoker, !!!ind_feature_map),
    alc = recode(alc, !!!ind_feature_map)
  )
```

```r
named_datasets <- list(mesa = mesa_std, jhs = jhs_std)
names(named_datasets) <- c("mesa", "jhs")
```

function

```r
conditional_densities <- function(data, treatment, formula_ps_no_con, formula_ps_con, use_confounders =

  formula <- if(use_confounders) formula_ps_con else formula_ps_no_con

  # Fit the linear model
  model <- lm(as.formula(formula), data = data)

  # Calculate the fitted values and standard deviation of residuals
  fitted_values <- fitted(model)
  resid_std <- sd(resid(model))

  # Calculate the density of treatment under a normal distribution with parameters from the model
  densities <- dnorm(data[[treatment]], mean = fitted_values, sd = resid_std)

  # Return the densities as a vector indexed similarly to the fitted values
  return(setNames(densities, names(fitted_values)))
}
```

result generation

loop to estimate nb effect with moderator

```r
report_df <- data.frame(
  Data = character(),
  Y = numeric(),
  X = numeric(),
  intercept = numeric(),
  intercept_pvalue = numeric(),
  X_coef = numeric(),
  X_pvalue = numeric(),
  X_ci_lwr = numeric(),
  X_ci_upr = numeric(),
  moderator = numeric(),
  moderator_coef = numeric(),
  moderator_pvalue = numeric(),
  moderator_ci_lwr = numeric(),
  moderator_ci_upr = numeric(),
  interaction_coef = numeric(),
  interaction_pvalue = numeric(),
  interact_ci_lwr = numeric(),
  interact_ci_upr = numeric(),
  stringsAsFactors = FALSE
)
```

```r
for (combo in all_combos) {

  # get values
  data_name = combo$data
  data <- named_datasets[[data_name]]

  Y <- combo$Y
  X <- combo$X
  M <- combo$M
  Z <- combo$covariates_final

  # generate formula; add interaction based on moderation result
  matches <- which(significant_moderation$data_name == data_name &
                     significant_moderation$Y == Y &
                     significant_moderation$X == X &
                     significant_moderation$M == M)

  if(length(matches) > 0) { # exist moderation, add interaction term in outcome model

    formula_ps_no_str <- paste(X, "~ 1")
    formula_ps_no <- as.formula(formula_ps_no_str)

    formula_ps_str <- paste(X, "~ ", M, "+" ,paste(Z, collapse=" + "))
    formula_ps <- as.formula(formula_ps_str)

    formula_outcome_str <- paste(Y, "~",X,"+",M,"+",paste(X, M, sep=":"))
    formula_outcome <- as.formula(formula_outcome_str)

    moderator <- M

  } else { # no moderation

    formula_ps_no_str <- paste(X, "~ 1")
    formula_ps_no <- as.formula(formula_ps_no_str)

    formula_ps_str <- paste(X, "~ ", M, "+" ,paste(Z, collapse=" + "))
    formula_ps <- as.formula(formula_ps_str)

    formula_outcome_str <- paste(Y, "~",X)
    formula_outcome <- as.formula(formula_outcome_str)

    moderator <- NA
  }


  # estimate

  denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
  numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
  propensity_density = numerator / denominator
  threshold <- quantile(propensity_density, 0.99)
  data <- data[propensity_density <= threshold, ]
  propensity_density <- propensity_density[propensity_density <= threshold]
```

```r
outcome_mod <- glm(formula = formula_outcome, data = data, family = binomial(), weights = propensity_
summary_model <- summary(outcome_mod)

# store model result
# there will be duplicate rows of results since we dont consider M unless significant moderation
# remove duplicates later

coefficients <- summary_model$coefficients
ci <- confint(outcome_mod)

intercept <- round(coefficients["(Intercept)", "Estimate"], digits = 5)  ## common parts
intercept_pvalue <- round(coefficients["(Intercept)", "Pr(>|z|)"], digits = 5)

X_coef <- round(coefficients[X, "Estimate"], digits = 5)
X_pvalue <- round(coefficients[X, "Pr(>|z|)"], digits = 5)
X_ci_lwr <- round(ci[X,"2.5 %"],digits = 5)
X_ci_upr <- round(ci[X,"97.5 %"],digits = 5)

if(is.na(moderator)) {

  moderator_coef <- NA
  moderator_pvalue <- NA
  moderator_ci_lwr <- NA
  moderator_ci_upr <- NA

  interaction_coef <- NA
  interaction_pvalue <- NA
  interact_ci_lwr <- NA
  interact_ci_upr <- NA

} else{

  moderator_coef <- round(coefficients[moderator, "Estimate"], digits = 5)
  moderator_pvalue <- round(coefficients[moderator, "Pr(>|z|)"], digits = 5)
  moderator_ci_lwr <- round(ci[moderator,"2.5 %"],digits = 5)
  moderator_ci_upr <- round(ci[moderator,"97.5 %"],digits = 5)

  interact_term_name <- tail(rownames(coefficients), 1)
  interaction_coef <- round(coefficients[interact_term_name, "Estimate"], digits = 5)
  interaction_pvalue <- round(coefficients[interact_term_name, "Pr(>|z|)"], digits = 5)
  interact_ci_lwr <- round(ci[interact_term_name,"2.5 %"],digits = 5)
  interact_ci_upr <- round(ci[interact_term_name,"97.5 %"],digits = 5)
}

new_df <- data.frame(
            Data = data_name,
            Y = Y,
            X = X,
            intercept = intercept,
            intercept_pvalue = intercept_pvalue,
            X_coef = X_coef,
            X_pvalue = X_pvalue,
            X_ci_lwr = X_ci_lwr,
```

```
             X_ci_upr = X_ci_upr,
             moderator = moderator,
             moderator_coef = moderator_coef,
             moderator_pvalue = moderator_pvalue,
             moderator_ci_lwr = moderator_ci_lwr,
             moderator_ci_upr = moderator_ci_upr,
             interaction_coef = interaction_coef,
             interaction_pvalue = interaction_pvalue,
             interact_ci_lwr = interact_ci_lwr,
             interact_ci_upr = interact_ci_upr,
             stringsAsFactors = FALSE )

  report_df <- rbind(report_df, new_df)

}
```

```
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
```

```
## Waiting for profiling to be done...
## Waiting for profiling to be done...
## Waiting for profiling to be done...
```

```
# exclude duplicate rows
ipw_moderate <- report_df %>%
  distinct()

# exclude the rows that should have been moderated
ipw_moderate <- ipw_moderate[-c(2,6,9), ]
```

effect with mediation

```
ipw_mediate <- ipw_moderate
ipw_mediate$mediator <- NA
ipw_mediate$mediator_coef <- NA
ipw_mediate$mediator_pvalue <- NA
ipw_mediate$mediator_ci_lwr <- NA
ipw_mediate$mediator_ci_upr <- NA
```

MESA nSES mediated by nutrition

```
ipw_mediate[(ipw_mediate$Data=="mesa")&(ipw_mediate$X =="nSES"),"mediator"] <- "nutrition"

data <- mesa_std
treatment = "nSES"
formula_ps_no = "nSES ~ 1"

formula_ps = "nSES ~ 1 + nRS + nFavFood + nPhysFac + \
                FamIncome + nutrition + PhysAct + currentSmoker + alc + \
                sbp + Diabetes + hdl + totchol + age + gender +\
                site + race"

formula_outcome_logit = 'cvd_10y_HF ~ 1 + nSES + nutrition'
```

Step 1: estimate propensity density

```
denominator = conditional_densities(data, treatment, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,treatment, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator

# exclude extreme values
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
```

Step 2: Outcome model

```r
modelA <- glm(formula = formula_outcome_logit, data = data, family = binomial(), weights = propensity_d
summary(modelA)
```

```
##
## Call:
## glm(formula = formula_outcome_logit, family = binomial(), data = data,
##     weights = propensity_density)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3884  -0.4199  -0.3769  -0.3089   5.2313
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.6651     0.1639  -10.16  < 2e-16 ***
## nSES          0.1329     0.0645    2.06  0.03940 *
## nutrition    -0.3534     0.1166   -3.03  0.00245 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2386.3  on 4571  degrees of freedom
## Residual deviance: 2371.7  on 4569  degrees of freedom
## AIC: 2792
##
## Number of Fisher Scoring iterations: 5
```

```r
confint(modelA)
```

```
## Waiting for profiling to be done...
```

```
##                    2.5 %     97.5 %
## (Intercept) -1.987490401 -1.3445203
## nSES         0.007634353  0.2605915
## nutrition   -0.585074546 -0.1274607
```

```r
# ipw_mediate[1,"intercept"] <- -1.29558
# ipw_mediate[1,"X_coef"] <- 0.12714
# ipw_mediate[1,"X_pvalue"] <- 0.04996
# ipw_mediate[1,"X_ci_lwr"] <- 0.001158415
# ipw_mediate[1,"X_ci_upr"] <- 0.2555166
# ipw_mediate[1,"mediator_coef"] <- "-0.34408; -0.15921"
# ipw_mediate[1,"mediator_pvalue"] <- "0.00319; 0.01471"
# ipw_mediate[1,"mediator_ci_lwr"] <- "-0.575831300; -0.286160205"
# ipw_mediate[1,"mediator_ci_upr"] <- "-0.1180496; -0.0301479"
#
# ipw_mediate[5,"intercept"] <- -1.7693
# ipw_mediate[5,"X_coef"] <- 0.2386
# ipw_mediate[5,"X_pvalue"] <- 0.000937
# ipw_mediate[5,"X_ci_lwr"] <- 0.09896727
# ipw_mediate[5,"X_ci_upr"] <- 0.3817504
```

```
# ipw_mediate[5,"mediator_coef"] <- -0.4484
# ipw_mediate[5,"mediator_pvalue"] <- 0.000552
# ipw_mediate[5,"mediator_ci_lwr"] <- -0.70724600
# ipw_mediate[5,"mediator_ci_upr"] <- -0.1977654
```

JHS with mediator JHS nSES mediated by FamIncome and moderated by PhysAct

```
ipw_mediate[(ipw_mediate$Data=="jhs")&(ipw_mediate$X =="nSES"),"mediator"] <- "FamIncome"

data <- jhs_std
treatment = "nSES"
formula_ps_no = "nSES ~ 1"

formula_ps = "nSES ~ 1 + nRS + nFavFood + nPhysFac + \
                FamIncome + nutrition + PhysAct + currentSmoker + alc + \
                sbp + Diabetes + hdl + totchol + age + gender"

formula_outcome_logit = 'cvd_10y_HF ~ 1 + nSES + PhysAct + nSES:PhysAct + FamIncome'
```

Step 1: estimate propensity density

```
denominator = conditional_densities(data, treatment, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,treatment, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator

# exclude extreme values
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
```

Step 2: Outcome model

```
modelA <- glm(formula = formula_outcome_logit, data = data, family = binomial(), weights = propensity_d
summary(modelA)
```

```
##
## Call:
## glm(formula = formula_outcome_logit, family = binomial(), data = data,
##     weights = propensity_density)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1154  -0.5009  -0.4186  -0.3239   4.4203
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.80091    0.19682  -4.069 4.72e-05 ***
## nSES          0.46369    0.17569   2.639  0.00831 **
## PhysAct      -0.32686    0.08125  -4.023 5.75e-05 ***
## FamIncome    -0.29667    0.06193  -4.791 1.66e-06 ***
## nSES:PhysAct -0.24787    0.09605  -2.581  0.00986 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2244.9  on 3494  degrees of freedom
## Residual deviance: 2185.9  on 3490  degrees of freedom
## AIC: 2339.9
##
## Number of Fisher Scoring iterations: 5
```

```
confint(modelA)
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %      97.5 %
## (Intercept)  -1.1896005 -0.41765491
## nSES          0.1205294  0.80926042
## PhysAct      -0.4882297 -0.16946085
## FamIncome    -0.4179247 -0.17506205
## nSES:PhysAct -0.4342918 -0.05761731
```

JHS nRS mediated by FamIncome

```
ipw_mediate[(ipw_mediate$Data=="jhs")&(ipw_mediate$X == "nRS"),"mediator"] <- "FamIncome"

data <- jhs_std
treatment = "nRS"
formula_ps_no = "nRS ~ 1"

formula_ps = "nRS ~ 1 + nSES + nFavFood + nPhysFac + \
                FamIncome + nutrition + PhysAct + currentSmoker + alc + \
                sbp + Diabetes + hdl + totchol + age + gender"

formula_outcome_logit = 'cvd_10y_HF ~ 1 + nRS + FamIncome'
```

Step 1: estimate propensity density

```
denominator = conditional_densities(data, treatment, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,treatment, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator

# exclude extreme values
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
```

Step 2: Outcome model

```
modelA <- glm(formula = formula_outcome_logit, data = data, family = binomial(), weights = propensity_d
summary(modelA)
```

```
##
## Call:
## glm(formula = formula_outcome_logit, family = binomial(), data = data,
##     weights = propensity_density)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1173  -0.4983  -0.4144  -0.3253   3.8656
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.11724    0.16086  -6.945 3.77e-12 ***
## nRS          0.17628    0.06880   2.562   0.0104 *
## FamIncome   -0.37719    0.06115  -6.168 6.91e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2178.1  on 3494  degrees of freedom
## Residual deviance: 2129.0  on 3492  degrees of freedom
## AIC: 2440.6
##
## Number of Fisher Scoring iterations: 5
```

```
confint(modelA)
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %     97.5 %
## (Intercept) -1.43663384 -0.8057310
## nRS          0.04341236  0.3132949
## FamIncome   -0.49719172 -0.2573647
```

```
# write.csv(ipw_mediate, "../results/generalized_IPW/ipw_final_0530.csv", row.names = FALSE)
```

```
ipw_significant <- read.csv("/Users/cxt/Documents/research/CVD/Effect-of-nSES-on-CVD/results/generalized
  filter(X_pvalue <= 0.05) %>%
  filter(Data != "mesa_bla")

# write.csv(ipw_significant, "../results/generalized_IPW/ipw_significant_0530.csv", row.names = FALSE)
```

export result

plot prediction curve for significant results

```
# ipw_final <- read.csv("../results/generalized_IPW/ipw_final_0528.csv")
#
# plot_ipw <- ipw_final %>%
#   filter(X_pvalue <= 0.05)
```

```
# for (i in 1:nrow(plot_ipw)) {
#   print(i)
#
#   # extract values
#   data_name <- plot_ipw[i, "Data"]
#   data <- named_datasets[[data_name]]
#   Y <- plot_ipw[i, "Y"]
#   X <- plot_ipw[i, "X"]
#   moderator <- plot_ipw[i, "moderator"]
#   X_coef <- plot_ipw[i, "X_coef"]
#   X_pvalue <- plot_ipw[i, "X_pvalue"]
#   interaction_coef <- plot_ipw[i, 'interaction_coef']
#   interaction_pvalue <- plot_ipw[i, "interaction_pvalue"]
#
#   # generate formula
#
#   ## covariates
#   base_X <- c("nSES", "nPhysFac", "nFavFood", "nRS")
#   base_Z <- c('age', 'gender', 'Diabetes', 'hdl', 'totchol', 'sbp')
#   base_M <- c("FamIncome", "nutrition", "PhysAct", "currentSmoker", "alc")
#
#   if (data_name == "mesa") {
#     final_Z = c(base_Z, 'race', 'site')
#   } else if (data_name == "mesa_bla") {
#     final_Z = c(base_Z, 'site')
#   } else if (data_name == "jhs") {
#     final_Z = base_Z
#   }
#
#   ## formula
#   formula_ps_no <- as.formula(paste(X, "~ 1"))
#
#   formula_ps <- as.formula(paste(X, "~ ", paste(setdiff(base_X,X), collapse = "+"),"+",
#                                  paste(base_M, collapse=" + "), "+" ,paste(final_Z, collapse=" + ")))
#
#   formula_outcome_str <- ifelse(is.na(moderator),
#                                 paste(Y, "~",X),
#                                 paste(Y, "~",X,"+",moderator,"+",paste(X, moderator, sep=":"))
#                                 )
#   formula_outcome <- as.formula(formula_outcome_str)
#
#   # fit model
#   denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
#   numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
#   propensity_density = numerator / denominator
#   threshold <- quantile(propensity_density, 0.99)
#   data <- data[propensity_density <= threshold, ]
#   propensity_density <- propensity_density[propensity_density <= threshold]
```

```
#   outcome_mod <- glm(formula = formula_outcome, data = data, family = binomial(), weights = propensit
#
#   # Predicted probabilities and plotting
#   if (is.na(moderator)) {
#     pred_data <- data.frame(seq(from = min(data[[X]]), to = max(data[[X]]), by = 0.1))
#     names(pred_data) <- X
#     preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
#     se <- preds$se.fit
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     plt <- ggplot(pred_data, aes(x = !!sym(X))) +
#       geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +
#       geom_line(aes(y = fit), color = "blue") +
#       geom_hline(yintercept = 0, color = "black") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#          x = X, y = "Estimated Probability") +
#       theme_minimal()
#
#   } else if (moderator == "alc") {
#     X_seq <- seq(from = min(data[[X]]), to = max(data[[X]]), by = 0.1)
#     X_length <- length(X_seq)
#     pred_data <- data.frame(X = rep(X_seq, 2),
#                             alc = c(rep(1, X_length), rep(2, X_length)))
#     names(pred_data) <- c(X, "alc")
#
#     preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
#     se <- preds$se.fit
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     plt <- ggplot(pred_data, aes(x = !!sym(X), y = fit,
#                                  group = factor(alc, levels = c(1,2), labels = c("No", "Yes")),
#                                  color = factor(alc))) +
#       geom_line() +
#       geom_hline(yintercept = 0, color = "black") +
#       scale_color_manual(values = c("blue", "red"),  labels = c("No", "Yes"),
#                          name = "Alcohol Drinking") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#          x = X, y = "Estimated Probability") +
#       theme_minimal()
#
#   } else if (moderator == "FamIncome") {
#     X_seq <- seq(from = min(data[[X]]), to = max(data[[X]]), by = 0.1)
#     X_length <- length(X_seq)
#     pred_data <- data.frame(X = rep(X_seq, 4),
#                             FamIncome = c(rep(1, X_length), rep(2, X_length), rep(3, X_length), rep(4
#     names(pred_data) <- c(X, "FamIncome")
#
#     preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
#     se <- preds$se.fit
```

```
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     plt <- ggplot(pred_data, aes(x = !!sym(X), y = fit,
#                                  group = factor(FamIncome, levels = c(1,2,3,4)),
#                                  color = factor(FamIncome))) +
#       geom_line() +
#       geom_hline(yintercept = 0, color = "black") +
#       scale_color_manual(values = c("red", "blue", "green", "orange"),
#                          labels = c("$0-11,999", "$12,000-24,999", "$25,000-74,999", "$75,000+"),
#                          name = "Family Income") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#          x = X, y = "Estimated Probability") +
#       theme_minimal()
#
#   } else if (moderator == "PhysAct") {
#     X_seq <- seq(from = min(data[[X]]), to = max(data[[X]]), by = 0.1)
#     X_length <- length(X_seq)
#     pred_data <- data.frame(X = rep(X_seq, 3),
#                             PhysAct = c(rep(1, X_length), rep(2, X_length), rep(3, X_length)))
#     names(pred_data) <- c(X, "PhysAct")
#
#     preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
#     se <- preds$se.fit
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     plt <- ggplot(pred_data, aes(x = !!sym(X), y = fit,
#                                  group = factor(PhysAct, levels = c(1,2,3)),
#                                  color = factor(PhysAct))) +
#       geom_line() +
#       geom_hline(yintercept = 0, color = "black") +
#       scale_color_manual(values = c("blue", "red","green"),  labels = c("Poor", "Intermediate","Ideal
#                          name = "Physical Activity") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#          x = X, y = "Estimated Probability") +
#       theme_minimal()
#   } else {
#     print("error")
#   }
#
#   # export plot
#   ggsave(paste0(data_name,"_",Y,"_",X,".png"), plot = plt,
#             path = "../results/generalized_IPW/significant_prediction_curve",
#             width = 8, height = 6, dpi = 300)
# }
```

```
# for (i in 1:nrow(plot_ipw)) {
#   print(i)
```

```
#
#   # extract values
#   data_name <- plot_ipw[i, "Data"]
#   data <- named_datasets[[data_name]]
#   Y <- plot_ipw[i, "Y"]
#   X <- plot_ipw[i, "X"]
#   moderator <- plot_ipw[i, "moderator"]
#   X_coef <- plot_ipw[i, "X_coef"]
#   X_pvalue <- plot_ipw[i, "X_pvalue"]
#   interaction_coef <- plot_ipw[i, 'interaction_coef']
#   interaction_pvalue <- plot_ipw[i, "interaction_pvalue"]
#
#   # generate formula
#
#   ## covariates
#   base_X <- c("nSES", "nPhysFac", "nFavFood", "nRS")
#   base_Z <- c('age', 'gender', 'Diabetes', 'hdl', 'totchol', 'sbp')
#   base_M <- c("FamIncome", "nutrition", "PhysAct", "currentSmoker", "alc")
#
#   if (data_name == "mesa") {
#     final_Z = c(base_Z, 'race', 'site')
#   } else if (data_name == "mesa_bla") {
#     final_Z = c(base_Z, 'site')
#   } else if (data_name == "jhs") {
#     final_Z = base_Z
#   }
#
#   ## formula
#   formula_ps_no <- as.formula(paste(X, "~ 1"))
#
#   formula_ps <- as.formula(paste(X, "~ ", paste(setdiff(base_X,X), collapse = "+"),"+",
#                                  paste(base_M, collapse=" + "), "+" ,paste(final_Z, collapse=" + ")))
#
#   formula_outcome_str <- ifelse(is.na(moderator),
#                             paste(Y, "~",X),
#                             paste(Y, "~",X,"+",moderator,"+",paste(X, moderator, sep=":"))
#                             )
#   formula_outcome <- as.formula(formula_outcome_str)
#
#   # fit model
#   denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
#   numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
#   propensity_density = numerator / denominator
#   threshold <- quantile(propensity_density, 0.99)
#   data <- data[propensity_density <= threshold, ]
#   propensity_density <- propensity_density[propensity_density <= threshold]
#   outcome_mod <- glm(formula = formula_outcome, data = data, family = binomial(), weights = propensit
#
#   # Predicted probabilities and plotting
#   if (is.na(moderator)) {
#     pred_data <- data.frame(seq(from = mean(data[[X]])-2*sd(data[[X]]),
#                                 to = mean(data[[X]])+2*sd(data[[X]]), by = 0.1))
#     names(pred_data) <- X
```

```
#     preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
#     se <- preds$se.fit
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     plt <- ggplot(pred_data, aes(x = !!sym(X))) +
#       geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +
#       geom_line(aes(y = fit), color = "blue") +
#       geom_hline(yintercept = 0, color = "black") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#          x = X, y = "Estimated Probability") +
#       theme_minimal()
#
#   } else if (moderator == "alc") {
#     stats <- data %>%
#       group_by(alc) %>%
#       summarize(mean_X = mean(!!sym(X), na.rm = TRUE),
#                 sd_X = sd(!!sym(X), na.rm = TRUE))
#
#     pred_data <- do.call(rbind, lapply(1:2, function(level) {
#       mean_X <- stats$mean_X[stats$alc == level]
#       sd_X <- stats$sd_X[stats$alc == level]
#       X_seq <- seq(from = mean_X - 2 * sd_X, to = mean_X + 2 * sd_X, by = 0.1)
#       data.frame(X = X_seq, alc = rep(level, length(X_seq)))
#     }))
#     names(pred_data) <- c(X, "alc")
#
#     preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
#     se <- preds$se.fit
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     plt <- ggplot(pred_data, aes(x = !!sym(X), y = fit,
#                                  group = factor(alc, levels = c(1,2), labels = c("No", "Yes")),
#                                  color = factor(alc))) +
#       geom_line() +
#       geom_ribbon(aes(ymin = lwr, ymax = upr), alpha = 0.2) +
#       geom_hline(yintercept = 0, color = "black") +
#       scale_color_manual(values = c("blue", "red"),  labels = c("No", "Yes"),
#                          name = "Alcohol Drinking") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#          x = X, y = "Estimated Probability") +
#       theme_minimal()
#
#   } else if (moderator == "FamIncome") {
#     stats <- data %>%
#       group_by(FamIncome) %>%
#       summarize(mean_X = mean(!!sym(X), na.rm = TRUE),
#                 sd_X = sd(!!sym(X), na.rm = TRUE))
#
#     pred_data <- do.call(rbind, lapply(1:4, function(level) {
```

```
#       mean_X <- stats$mean_X[stats$FamIncome == level]
#       sd_X <- stats$sd_X[stats$FamIncome == level]
#       X_seq <- seq(from = mean_X - 2 * sd_X, to = mean_X + 2 * sd_X, by = 0.1)
#       data.frame(X = X_seq, FamIncome = rep(level, length(X_seq)))
#     }))
#     names(pred_data) <- c(X, "FamIncome")
#
#     preds <- predict(outcome_mod, pred_data, type = "response", se.fit = TRUE)
#     se <- preds$se.fit
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     plt <- ggplot(pred_data, aes(x = !!sym(X), y = fit,
#                                  group = factor(FamIncome, levels = c(1, 2, 3, 4)),
#                                  color = factor(FamIncome))) +
#       geom_line() +
#       geom_ribbon(aes(ymin = lwr, ymax = upr), alpha = 0.2) +
#       geom_hline(yintercept = 0, color = "black") +
#       scale_color_manual(values = c("red", "blue", "green", "orange"),
#                          labels = c("$0-11,999", "$12,000-24,999", "$25,000-74,999", "$75,000+"),
#                          name = "Family Income") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#            x = X, y = "Estimated Probability") +
#       theme_minimal()
#
#
#   } else if (moderator == "PhysAct") {
#     stats <- data %>%
#       group_by(PhysAct) %>%
#       summarize(mean_X = mean(!!sym(X), na.rm = TRUE),
#                 sd_X = sd(!!sym(X), na.rm = TRUE))
#
#     # Generate prediction data
#     pred_data <- do.call(rbind, lapply(1:3, function(level) {
#       mean_X <- stats$mean_X[stats$PhysAct == level]
#       sd_X <- stats$sd_X[stats$PhysAct == level]
#       X_seq <- seq(from = mean_X - 2 * sd_X, to = mean_X + 2 * sd_X, by = 0.1)
#       data.frame(X = X_seq, PhysAct = rep(level, length(X_seq)))
#     }))
#
#     # Set column names
#     names(pred_data) <- c(X, "PhysAct")
#
#     # Get predictions
#     preds <- predict(outcome_mod, pred_data, type = "response", se.fit = TRUE)
#     se <- preds$se.fit
#     pred_data$fit <- preds$fit
#     pred_data$lwr <- preds$fit - 1.96 * se
#     pred_data$upr <- preds$fit + 1.96 * se
#
#     # Plot
#     plt <- ggplot(pred_data, aes(x = !!sym(X), y = fit,
```

```
#                                   group = factor(PhysAct, levels = c(1, 2, 3)),
#                                   color = factor(PhysAct))) +
#       geom_line() +
#       geom_ribbon(aes(ymin = lwr, ymax = upr), alpha = 0.2) +
#       geom_hline(yintercept = 0, color = "black") +
#       scale_color_manual(values = c("blue", "red", "green"),
#                          labels = c("Poor", "Intermediate", "Ideal"),
#                          name = "Physical Activity") +
#       labs(title = "Predicted Probabilities of Positive Outcome",
#            x = X, y = "Estimated Probability") +
#       theme_minimal()
#   } else {
#     print("error")
#   }
#
#   # export plot
#   ggsave(paste0(data_name,"_",Y,"_",X,".png"), plot = plt,
#            path = "../results/generalized_IPW/significant_prediction_curve_0528",
#            width = 8, height = 6, dpi = 300)
# }
```

```
# test_dat <- data %>%
#   mutate(PhysAct = factor(PhysAct, levels = c(1, 2, 3)))
#
# # Create the density plot
# plt_density <- ggplot(test_dat, aes(x = !!sym("nSES"), fill = PhysAct, color = PhysAct)) +
#   geom_density(alpha = 0.4) +
#   scale_fill_manual(values = c("blue", "red", "green"),
#                     labels = c("Poor", "Intermediate", "Ideal"),
#                     name = "Physical Activity") +
#   scale_color_manual(values = c("blue", "red", "green"),
#                      labels = c("Poor", "Intermediate", "Ideal"),
#                      name = "Physical Activity") +
#   labs(title = "Density Plot of X by Levels of Physical Activity",
#        x = X, y = "Density") +
#   theme_minimal()
#
# print(plt_density)
```

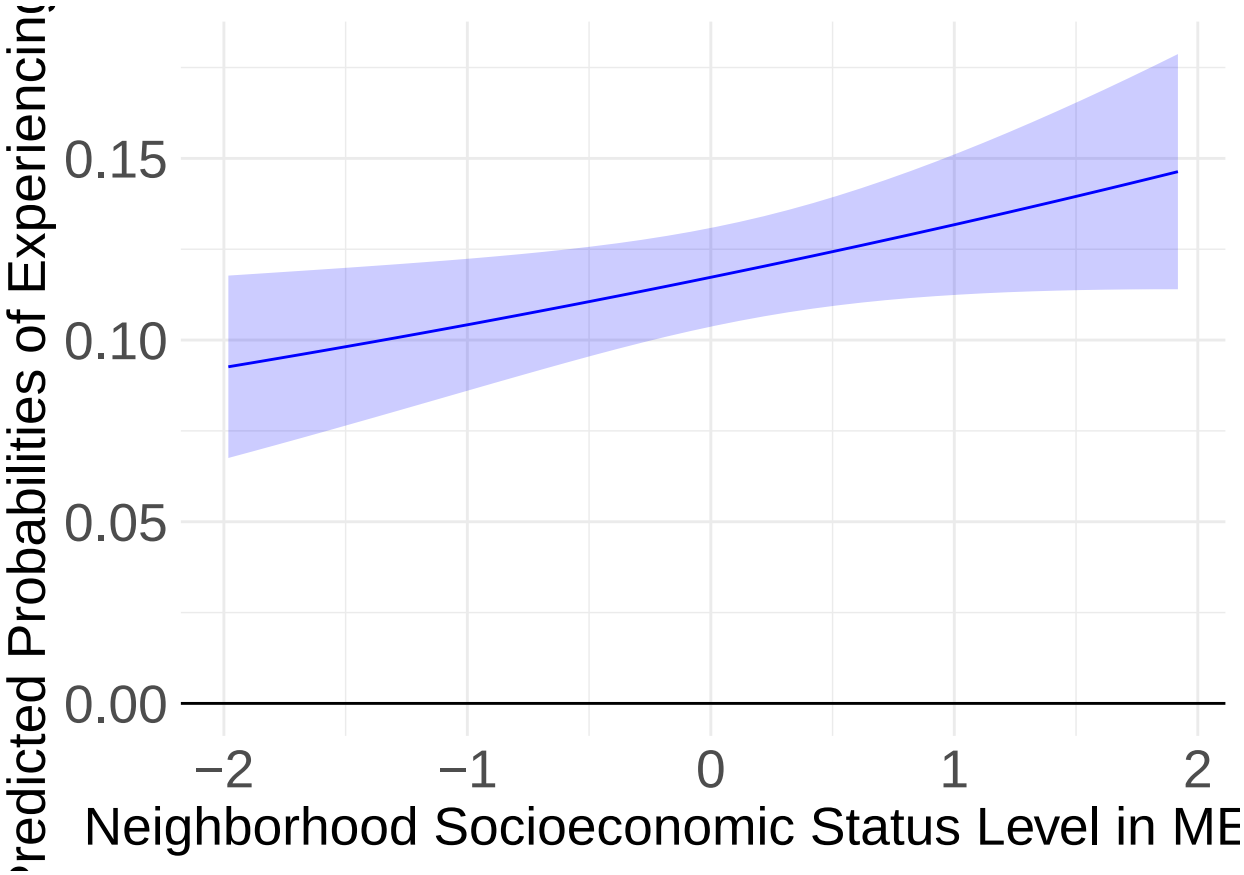
plot: X range +- 2sd

Manually plot (final)

MESA nSES

```
data <- mesa_std
Y <- "cvd_10y_HF"
X <- "nSES"

# generate formula
formula_ps_no <- as.formula("nSES ~ 1")
```

```r
formula_ps <- as.formula("nSES ~ nPhysFac+nFavFood+nRS+\
                         FamIncome+nutrition+PhysAct+currentSmoker+alc+\
                         age+gender+Diabetes+hdl+totchol+sbp+\
                         race+site")
formula_outcome <- as.formula("cvd_10y_HF ~ nSES + nutrition")

# fit model
denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
outcome_mod <- glm(formula = formula_outcome, data = data, family = binomial(), weights = propensity_de

# Predicted probabilities and plotting
## generate pred data
pred_data <- data.frame(seq(from = mean(data[[X]])-2*sd(data[[X]]),
                                to = mean(data[[X]])+2*sd(data[[X]]), by = 0.1))
names(pred_data) <- X
pred_data$nutrition <- Mode(mesa_std$nutrition)

preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
se <- preds$se.fit
pred_data$fit <- preds$fit
pred_data$lwr <- preds$fit - 1.96 * se
pred_data$upr <- preds$fit + 1.96 * se

plt1 <- ggplot(pred_data, aes(x = !!sym(X))) +
  geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +
  geom_line(aes(y = fit), color = "blue") +
  geom_hline(yintercept = 0, color = "black") +
  labs(x = "Neighborhood Socioeconomic Status Level in MESA", y = "Predicted Probabilities of Experienc
  theme_minimal() +
  theme(
        axis.title.x = element_text(size = 20),
    axis.title.y = element_text(size = 20),
    axis.text.x = element_text(size = 20),
    axis.text.y = element_text(size = 20)
      )

plt1
```



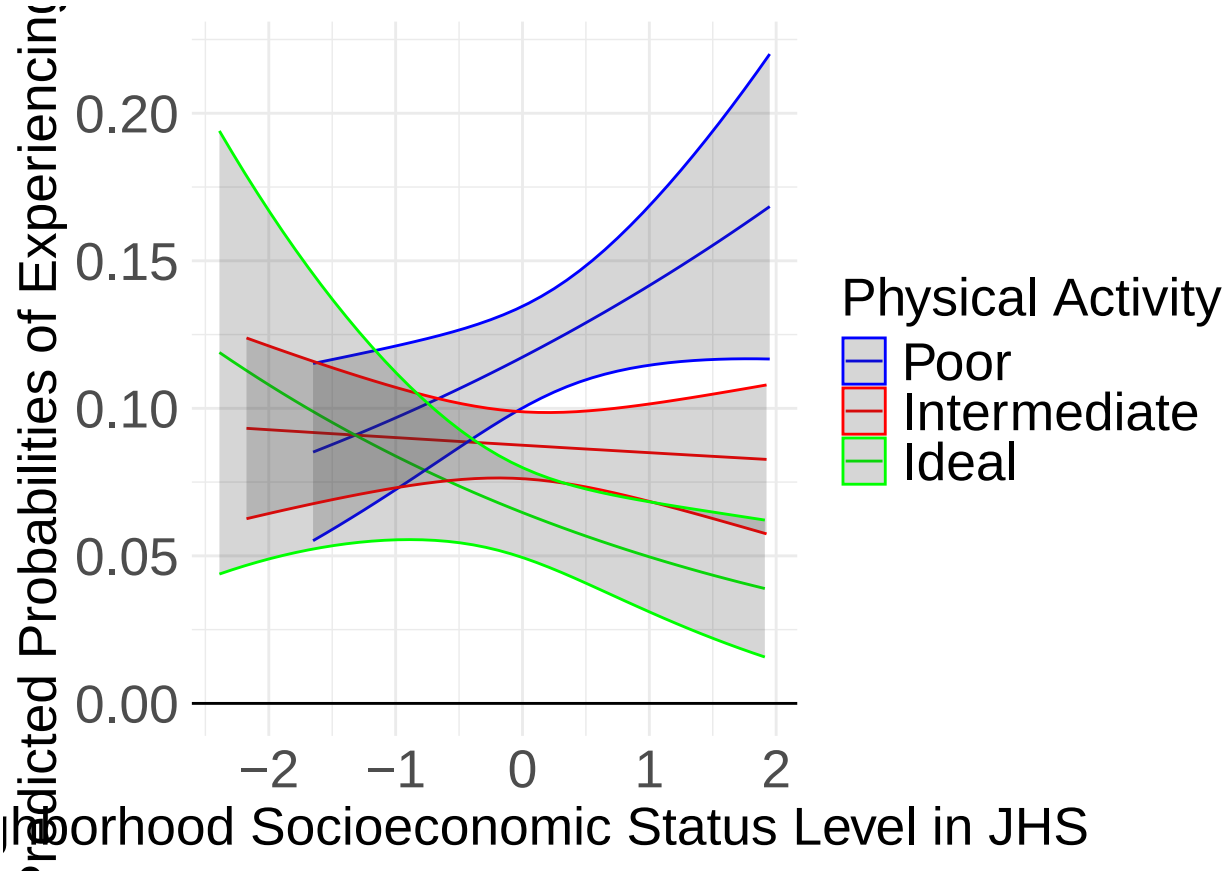

```
# ggsave("mesa_nses.png", plot = plt1,
#            path = "../results/generalized_IPW/prediction_curve_final",
#            width = 10, height = 8, dpi = 300)
```

JHS nSES

```
data <- jhs_std
Y <- "cvd_10y_HF"
X <- "nSES"

# generate formula
formula_ps_no <- as.formula("nSES ~ 1")
formula_ps <- as.formula("nSES ~ nPhysFac+nFavFood+nRS+\
                         FamIncome+nutrition+PhysAct+currentSmoker+alc+\
                         age+gender+Diabetes+hdl+totchol+sbp")
formula_outcome <- as.formula("cvd_10y_HF ~ nSES + PhysAct + nSES:PhysAct + FamIncome")

# fit model
denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
outcome_mod <- glm(formula = formula_outcome, data = data, family = binomial(), weights = propensity_de
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```r
# Predicted probabilities and plotting
## generate pred data
stats <- data %>%
      group_by(PhysAct) %>%
      summarize(mean_X = mean(!!sym(X), na.rm = TRUE),
                sd_X = sd(!!sym(X), na.rm = TRUE))

# Generate prediction data
pred_data <- do.call(rbind, lapply(1:3, function(level) {
  mean_X <- stats$mean_X[stats$PhysAct == level]
  sd_X <- stats$sd_X[stats$PhysAct == level]
  X_seq <- seq(from = mean_X - 2 * sd_X, to = mean_X + 2 * sd_X, by = 0.1)
  data.frame(X = X_seq, PhysAct = rep(level, length(X_seq)))
    }))

# Set column names
names(pred_data) <- c(X, "PhysAct")
pred_data$FamIncome <- Mode(jhs_std$FamIncome)

# Get predictions
preds <- predict(outcome_mod, pred_data, type = "response", se.fit = TRUE)
se <- preds$se.fit
pred_data$fit <- preds$fit
pred_data$lwr <- preds$fit - 1.96 * se
pred_data$upr <- preds$fit + 1.96 * se

# Plot
plt2 <- ggplot(pred_data, aes(x = !!sym(X), y = fit,
                              group = factor(PhysAct, levels = c(1, 2, 3)),
                              color = factor(PhysAct))) +
      geom_line() +
      geom_ribbon(aes(ymin = lwr, ymax = upr), alpha = 0.2) +
      geom_hline(yintercept = 0, color = "black") +
      scale_color_manual(values = c("blue", "red", "green"),
                         labels = c("Poor", "Intermediate", "Ideal"),
                         name = "Physical Activity") +
      labs(x = "Neighborhood Socioeconomic Status Level in JHS", y = "Predicted Probabilities of Experi
  theme_minimal() +
  theme(
        axis.title.x = element_text(size = 20),
    axis.title.y = element_text(size = 20),
    axis.text.x = element_text(size = 20),
    axis.text.y = element_text(size = 20),
        legend.text = element_text(size = 20),
    legend.title = element_text(size = 20)
      )
plt2
```

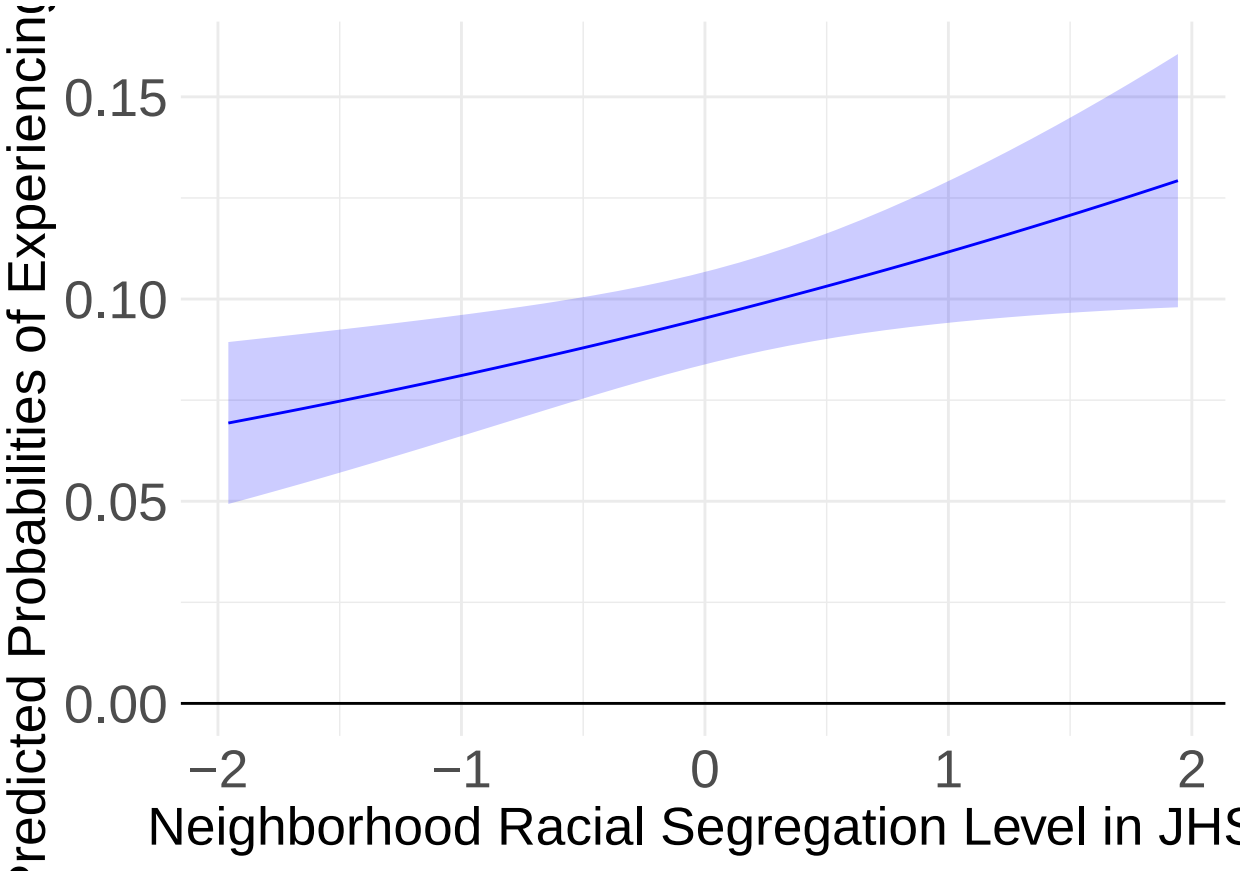



```
# ggsave("jhs_nses.png", plot = plt2,
#            path = "../results/generalized_IPW/prediction_curve_final",
#            width = 12, height = 8, dpi = 300)
```

JHS nRS

```
data <- jhs_std
Y <- "cvd_10y_HF"
X <- "nRS"

# generate formula
formula_ps_no <- as.formula("nRS ~ 1")
formula_ps <- as.formula("nRS ~ nSES+nPhysFac+nFavFood+\
                         FamIncome+nutrition+PhysAct+currentSmoker+alc+\
                         age+Diabetes+hdl+totchol+sbp")
formula_outcome <- as.formula("cvd_10y_HF ~ nRS + FamIncome")

# fit model
denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
outcome_mod <- glm(formula = formula_outcome, data = data, family = binomial(), weights = propensity_de
```

```r
# Predicted probabilities and plotting
## generate pred data
pred_data <- data.frame(seq(from = mean(data[[X]])-2*sd(data[[X]]),
                            to = mean(data[[X]])+2*sd(data[[X]]), by = 0.1))
names(pred_data) <- X
pred_data$FamIncome <- Mode(jhs_std$FamIncome)

preds <- predict(outcome_mod, pred_data,type = "response", se.fit = TRUE)
se <- preds$se.fit
pred_data$fit <- preds$fit
pred_data$lwr <- preds$fit - 1.96 * se
pred_data$upr <- preds$fit + 1.96 * se

plt3 <- ggplot(pred_data, aes(x = !!sym(X))) +
  geom_ribbon(aes(ymin = lwr, ymax = upr), fill = "blue", alpha = 0.2) +
  geom_line(aes(y = fit), color = "blue") +
  geom_hline(yintercept = 0, color = "black") +
  labs(x = "Neighborhood Racial Segregation Level in JHS", y = "Predicted Probabilities of Experiencing
  theme_minimal() +
  theme(
        axis.title.x = element_text(size = 20),
    axis.title.y = element_text(size = 20),
    axis.text.x = element_text(size = 20),
    axis.text.y = element_text(size = 20)
      )

plt3
```




```
# ggsave("jhs_nrs.png", plot = plt3,
#            path = "../results/generalized_IPW/prediction_curve_final",
#            width = 10, height = 8, dpi = 300)
```

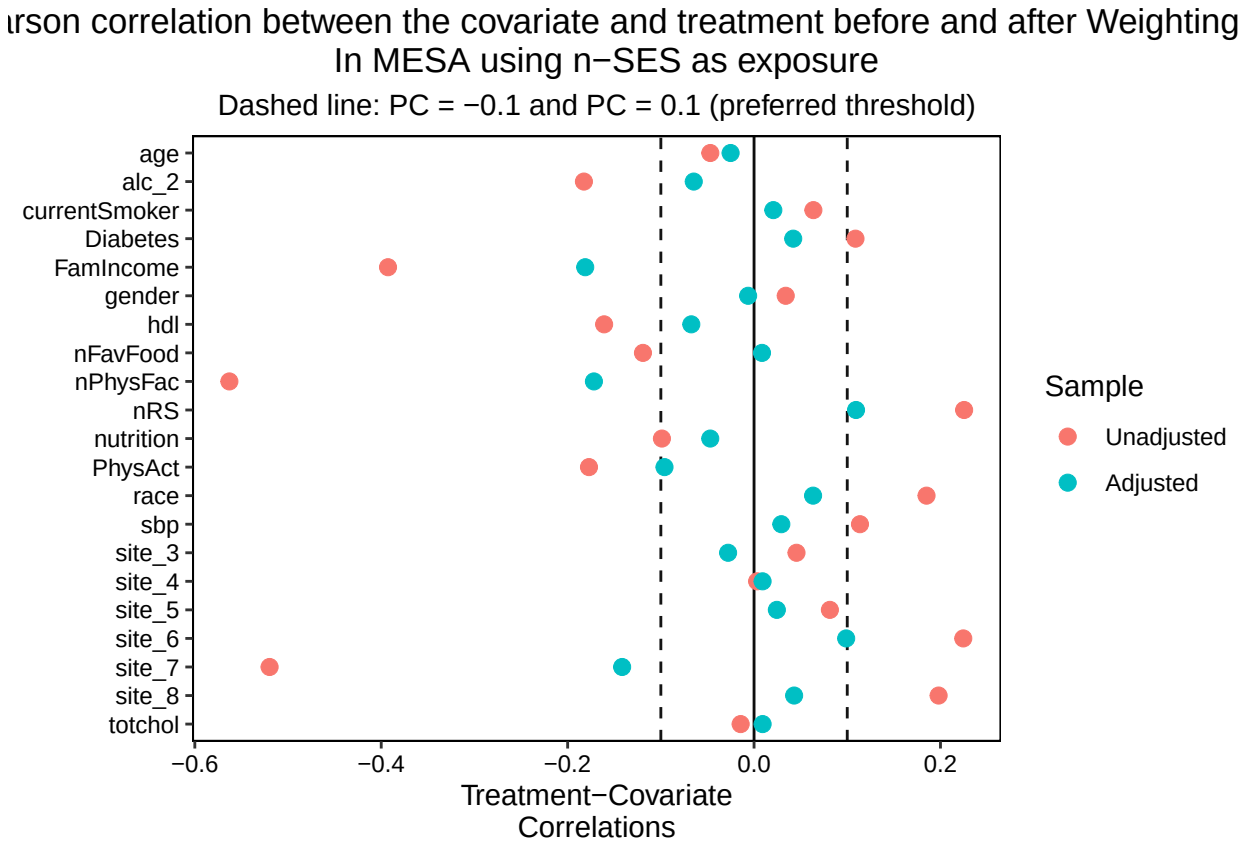
Check assumption

dichotomize the neighborhood exposure variables after weight calculation, to examine overlap and balance\
tutorial from https://ehsanx.github.io/psw/s3.html//

MESA nSES

```
data <- mesa_std
Y <- "cvd_10y_HF"
X <- "nSES"
Z <- c("nPhysFac","nFavFood","nRS",
       "FamIncome","nutrition","PhysAct","currentSmoker","alc",
       "age","Diabetes","hdl","totchol","sbp","gender",
       "race","site")

# generate formula
formula_ps_no <- as.formula("nSES ~ 1")
formula_ps <- as.formula("nSES ~ nPhysFac+nFavFood+nRS+\
                         FamIncome+nutrition+PhysAct+currentSmoker+alc+\
                         age+Diabetes+hdl+totchol+sbp+gender+\
                         race+site")
```

```r
# get weights
denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
data$weights <- propensity_density


smd_after <- bal.tab(
  data[, Z],
  treat = "nSES",
  data =data,
  weights = "weights",
  un = TRUE
)

# Plotting the SMDs before and after weighting
plt <- love.plot(smd_after,
          threshold = 0.1,  # Preferred threshold for good balance
          var.order = "alphabetical") +
  labs(title = "Pearson correlation between the covariate and treatment before and after Weighting\
  In MESA using n-SES as exposure",
       subtitle = "Dashed line: PC = -0.1 and PC = 0.1 (preferred threshold)")

plt
```




```
# ggsave("mesa_nses_balance.png", plot = plt,
#            path = "../results/generalized_IPW/check_balance",
#            width = 8, height = 6, dpi = 300)
```

JHS nSES

```
data <- jhs_std
Y <- "cvd_10y_HF"
X <- "nSES"
Z <- c("nPhysFac","nFavFood","nRS",
       "FamIncome","nutrition","PhysAct","currentSmoker","alc",
       "age","Diabetes","hdl","totchol","sbp","gender")

# generate formula
formula_ps_no <- as.formula("nSES ~ 1")
formula_ps <- as.formula("nSES ~ nPhysFac+nFavFood+nRS+\
                         FamIncome+nutrition+PhysAct+currentSmoker+alc+\
                         age+gender+Diabetes+hdl+totchol+sbp")

# fit model
denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
```

```
propensity_density <- propensity_density[propensity_density <= threshold]
data$weights <- propensity_density

smd_after <- bal.tab(
  data[, Z],
  treat = "nSES",
  data =data,
  weights = "weights",
  un = TRUE
)

# Plotting the SMDs before and after weighting
plt <- love.plot(smd_after,
          threshold = 0.1,  # Preferred threshold for good balance
          var.order = "alphabetical") +
  labs(title = "Pearson correlation between the covariate and treatment before and after Weighting\
  In JHS using n-SES as exposure",
       subtitle = "Dashed line: PC = -0.1 and PC = 0.1 (preferred threshold)")

# ggsave("jhs_nses_balance.png", plot = plt,
#            path = "../results/generalized_IPW/check_balance",
#            width = 8, height = 6, dpi = 300)
```

JHS nRS

```
data <- jhs_std
Y <- "cvd_10y_HF"
X <- "nRS"
Z <- c("nPhysFac","nFavFood","nSES",
       "FamIncome","nutrition","PhysAct","currentSmoker","alc",
       "age","Diabetes","hdl","totchol","sbp","gender")

# generate formula
formula_ps_no <- as.formula("nRS ~ 1")
formula_ps <- as.formula("nRS ~ nSES+nPhysFac+nFavFood+\
                         FamIncome+nutrition+PhysAct+currentSmoker+alc+\
                         age+gender+Diabetes+hdl+totchol+sbp")

# fit model
denominator = conditional_densities(data, X, formula_ps_no,formula_ps, use_confounders=T)
numerator = conditional_densities(data,X, formula_ps_no,formula_ps, use_confounders=F)
propensity_density = numerator / denominator
threshold <- quantile(propensity_density, 0.99)
data <- data[propensity_density <= threshold, ]
propensity_density <- propensity_density[propensity_density <= threshold]
data$weights <- propensity_density

# using continuous nSES
smd_after <- bal.tab(
  data[, Z],
  treat = "nRS",
  data =data,
  weights = "weights",
```

```r
  un = TRUE
)

# Plotting the SMDs before and after weighting
plt <- love.plot(smd_after,
          threshold = 0.1,  # Preferred threshold for good balance
          var.order = "alphabetical") +
  labs(title = "Pearson correlation between the covariate and treatment before and after Weighting\
  In JHS using n-Racial Segregation as exposure",
       subtitle = "Dashed line: PC = -0.1 and PC = 0.1 (preferred threshold)")

# ggsave("jhs_nrs_balance.png", plot = plt,
#            path = "../results/generalized_IPW/check_balance",
#            width = 8, height = 6, dpi = 300)
```