function (formula = formula(data), data, n.trees = 10000, interaction.depth = 3,

shrinkage = 0.01, bag.fraction = 1, perm.test.iters = 0,

print.level = 2, iterlim = 1000, verbose = TRUE, estimand = "ATE",

stop.method = c("ks.mean", "es.mean"), sampw = NULL, multinom = FALSE,

...)

{

if (is.null(sampw))

sampW <- rep(1, nrow(data))

else sampW <- sampw

type <- alert <- NULL

dots <- list(...)

if (!is.null(dots$plots))

warning("From version 1.2, the plots argument has been removed from ps(). \nPlease use the plot() function instead.")

stop.method[stop.method == "ks.stat.mean"] <- "ks.mean"

stop.method[stop.method == "es.stat.mean"] <- "es.mean"

stop.method[stop.method == "ks.stat.max"] <- "ks.max"

stop.method[stop.method == "es.stat.max"] <- "es.max"

stop.method[stop.method == "ks.stat.max.direct"] <- "ks.max.direct"

stop.method[stop.method == "es.stat.max.direct"] <- "es.max.direct"

if (!(estimand %in% c("ATT", "ATE")))

stop("estimand must be either \"ATT\" or \"ATE\".")

allowableStopMethods <- c("ks.mean", "es.mean", "ks.max",

"es.max", "ks.max.direct", "es.max.direct")

nMethod <- length(stop.method)

methodList <- vector(mode = "list", length = nMethod)

for (i in 1:nMethod) {

if (is.character(stop.method[i])) {

if (!(stop.method[i] %in% allowableStopMethods)) {

print(allowableStopMethods)

stop("Each element of stop.method must either be one of \nthe above character strings, or an object of the stop.method class.")

}

methodList[[i]] <- get(stop.method[i])

methodName <- paste(stop.method[i], ".", estimand,

sep = "")

methodList[[i]]$name <- methodName

}

else {

if (class(stop.method[i]) != "stop.method") {

print(allowableStopMethods)

stop("Each element of stop.method must either be one of \nthe above character strings, or an object of the stop.method class.")

}

methodList[[i]] <- stop.method[i]

}

}

stop.method <- methodList

terms <- match.call()

mf <- match.call(expand.dots = FALSE)

m <- match(c("formula", "data"), names(mf), 0)

mf <- mf[c(1, m)]

mf[[1]] <- as.name("model.frame")

mf$na.action <- na.pass

mf$subset <- rep(FALSE, nrow(data))

mf <- eval(mf, parent.frame())

Terms <- attr(mf, "terms")

var.names <- attributes(Terms)$term.labels

if (length(var.names) < 2)

stop("At least two variables are needed in the right-hand side of the formula.\n")

treat.var <- as.character(formula[[2]])

stop.method.names <- sapply(stop.method, function(x) {

x$name

})

i <- which(sapply(stop.method, function(x) {

x$direct

}))

if (length(i) > 0)

cat(paste("\*\*\* WARNING: Stop method", stop.method.names[i],

"involves direct\noptimization, which is very time consuming, especially\nfor datasets with more than a few hundred cases. Consider\nusing another stop.method or be prepared for a very long wait. \*\*\*\n\n"))

desc <- vector("list", 1 + length(stop.method))

names(desc) <- c("unw", stop.method.names)

p.s <- data.frame(matrix(NA, nrow = nrow(data), ncol = length(stop.method)))

names(p.s) <- stop.method.names

w <- data.frame(matrix(NA, nrow = nrow(data), ncol = length(stop.method)))

names(w) <- stop.method.names

alerts.stack <- textConnection("alert", "w")

if (verbose)

cat("Fitting gbm model\n")

form <- paste(deparse(formula, 500), collapse = "")

gbm1 <- gbm(formula(form), data = data, weights = sampW,

distribution = "bernoulli", n.trees = n.trees, interaction.depth = interaction.depth,

n.minobsinnode = 10, shrinkage = shrinkage, bag.fraction = bag.fraction,

train.fraction = 1, verbose = verbose, keep.data = FALSE)

if (verbose)

cat("Diagnosis of unweighted analysis\n")

if (is.factor(data[, treat.var]))

stop("Treatment indicator must be numeric, not a factor")

desc$unw <- desc.wts(data = data[, c(treat.var, var.names)],

treat.var = treat.var, w = sampW, sampw = rep(1, nrow(data)),

tp = "unw", na.action = "level", perm.test.iters = perm.test.iters,

verbose = verbose, alerts.stack = alerts.stack, estimand = estimand,

multinom = multinom)

desc$unw$n.trees <- NA

balance <- matrix(NA, ncol = nMethod, nrow = 25)

for (i.tp in 1:nMethod) {

tp <- stop.method.names[i.tp]

if (verbose)

cat("Optimizing with", tp, "stopping rule\n")

iters <- round(seq(1, gbm1$n.trees, length = 25))

bal <- rep(0, length(iters))

for (j in 1:length(iters)) {

bal[j] <- MetricI(iters[j], fun = match.fun(stop.method[[i.tp]]$metric),

vars = var.names, treat.var = treat.var, data = data,

sampw = sampW, rule.summary = match.fun(stop.method[[i.tp]]$rule.summary),

na.action = stop.method[[i.tp]]$na.action, gbm1 = gbm1,

estimand = estimand, multinom = multinom)

balance[, i.tp] <- bal

interval <- which.min(bal) + c(-1, 1)

interval[1] <- max(1, interval[1])

interval[2] <- min(length(iters), interval[2])

}

opt <- optimize(MetricI, interval = iters[interval],

maximum = FALSE, tol = 1, fun = match.fun(stop.method[[i.tp]]$metric),

vars = var.names, treat.var = treat.var, data = data,

sampw = sampW, rule.summary = match.fun(stop.method[[i.tp]]$rule.summary),

na.action = stop.method[[i.tp]]$na.action, gbm1 = gbm1,

estimand = estimand, multinom = multinom)

if (verbose)

cat(" Optimized at", round(opt$minimum), "\n")

if (gbm1$n.trees - opt$minimum < 100)

warning("Optimal number of iterations is close to the specified n.trees. n.trees is likely set too small and better balance might be obtainable by setting n.trees to be larger.")

p.s[, i.tp] <- predict(gbm1, newdata = data, n.trees = round(opt$minimum),

type = "response")

if (estimand == "ATT") {

w[, i.tp] <- p.s[, i.tp]/(1 - p.s[, i.tp])

w[data[, treat.var] == 1, i.tp] <- 1

}

if (estimand == "ATE") {

w[data[, treat.var] == 1, i.tp] <- 1/p.s[data[, treat.var] ==

1, i.tp]

w[data[, treat.var] == 0, i.tp] <- 1/(1 - p.s[data[,

treat.var] == 0, i.tp])

}

w[, i.tp] <- w[, i.tp] \* sampW

if (stop.method[[i.tp]]$direct) {

if (verbose)

cat(" Proceeding to direct optimization\n")

obj <- nlm(match.fun(stop.method[[i.tp]]$metric),

p = log(w[data[, treat.var] == 0, i.tp]), fscale = 0.1,

data = data, vars = var.names, treat.var = treat.var,

ndigit = 3, rule.summary = match.fun(stop.method[[i.tp]]$rule.summary),

na.action = stop.method[[i.tp]]$na.action, print.level = print.level,

iterlim = iterlim, estimand = estimand)

if (obj$code %in% c(4, 5))

warning("Failed to completely optimize metric directly.")

w[data[, treat.var] == 0, i.tp] <- exp(obj$estimate)

}

if (verbose)

cat("Diagnosis of", tp, "weights\n")

desc[[tp]] <- desc.wts(data[, c(treat.var, var.names)],

treat.var = treat.var, w = w[, i.tp], sampw = sampW,

tp = type, na.action = stop.method[[i.tp]]$na.action,

perm.test.iters = perm.test.iters, verbose = verbose,

alerts.stack = alerts.stack, estimand = estimand,

multinom = multinom)

desc[[tp]]$n.trees <- ifelse(stop.method[[i.tp]]$direct,

NA, round(opt$minimum))

}

close(alerts.stack)

if (verbose)

cat(alert, sep = "\n")

result <- list(gbm.obj = gbm1, treat = data[, treat.var],

treat.var = treat.var, desc = desc, ps = p.s, w = w,

sampw = sampW, estimand = estimand, datestamp = date(),

parameters = terms, alerts = alert, iters = iters, balance = balance,

n.trees = n.trees, data = data)

class(result) <- "ps"

return(result)

}