function (exposure, family, link, numerator = NULL, denominator,

data, trunc = NULL, ...)

{

tempcall <- match.call()

if (!("exposure" %in% names(tempcall)))

stop("No exposure variable specified")

if (!("family" %in% names(tempcall)) | ("family" %in% names(tempcall) &

!(tempcall$family %in% c("binomial", "multinomial", "ordinal",

"gaussian"))))

stop("No valid family specified (\"binomial\", \"multinomial\", \"ordinal\", \"gaussian\")")

if (tempcall$family == "binomial") {

if (!(tempcall$link %in% c("logit", "probit", "cauchit",

"log", "cloglog")))

stop("No valid link function specified for family = binomial (\"logit\", \"probit\", \"cauchit\", \"log\", \"cloglog\")")

}

if (tempcall$family == "ordinal") {

if (!(tempcall$link %in% c("logit", "probit", "cauchit",

"cloglog")))

stop("No valid link function specified for family = binomial (\"logit\", \"probit\", \"cauchit\", \"cloglog\")")

}

if (!("denominator" %in% names(tempcall)))

stop("No denominator model specified")

if (!is.null(tempcall$numerator) & !is(eval(tempcall$numerator),

"formula"))

stop("Invalid numerator formula specified")

if (!is.null(tempcall$denominator) & !is(eval(tempcall$denominator),

"formula"))

stop("Invalid denominator formula specified")

if (tempcall$family %in% c("gaussian") & !("numerator" %in%

names(tempcall)))

stop("Numerator necessary for family = \"gaussian\"")

if (!("data" %in% names(tempcall)))

stop("No data specified")

if (!is.null(tempcall$trunc)) {

if (tempcall$trunc < 0 | tempcall$trunc > 0.5)

stop("Invalid truncation percentage specified (0-0.5)")

}

tempdat <- data.frame(exposure = data[, as.character(tempcall$exposure)])

if (tempcall$family == "binomial") {

if (tempcall$link == "logit")

lf <- binomial(link = logit)

if (tempcall$link == "probit")

lf <- binomial(link = probit)

if (tempcall$link == "cauchit")

lf <- binomial(link = cauchit)

if (tempcall$link == "log")

lf <- binomial(link = log)

if (tempcall$link == "cloglog")

lf <- binomial(link = cloglog)

if (is.null(tempcall$numerator))

tempdat$w.numerator <- 1

else {

mod1 <- glm(formula = eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$numerator,

width.cutoff = 500), sep = ""))), family = lf,

data = data, na.action = na.fail, ...)

tempdat$w.numerator <- vector("numeric", nrow(tempdat))

tempdat$w.numerator[tempdat$exposure == 0] <- 1 -

predict.glm(mod1, type = "response")[tempdat$exposure ==

0]

tempdat$w.numerator[tempdat$exposure == 1] <- predict.glm(mod1,

type = "response")[tempdat$exposure == 1]

mod1$call$formula <- eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$numerator,

width.cutoff = 500), sep = "")))

mod1$call$family <- tempcall$link

mod1$call$data <- tempcall$data

}

mod2 <- glm(formula = eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$denominator,

width.cutoff = 500), sep = ""))), family = lf, data = data,

na.action = na.fail, ...)

tempdat$w.denominator <- vector("numeric", nrow(tempdat))

tempdat$w.denominator[tempdat$exposure == 0] <- 1 - predict.glm(mod2,

type = "response")[tempdat$exposure == 0]

tempdat$w.denominator[tempdat$exposure == 1] <- predict.glm(mod2,

type = "response")[tempdat$exposure == 1]

mod2$call$formula <- eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$denominator,

width.cutoff = 500), sep = "")))

mod2$call$family <- tempcall$link

mod2$call$data <- tempcall$data

tempdat$ipw.weights <- tempdat$w.numerator/tempdat$w.denominator

}

if (tempcall$family == "multinomial") {

if (is.null(tempcall$numerator))

tempdat$p.numerator <- 1

else {

mod1 <- multinom(formula = eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$numerator,

width.cutoff = 500), sep = ""))), data = data,

na.action = na.fail, ...)

pred1 <- as.data.frame(predict(mod1, type = "probs"))

tempdat$w.numerator <- vector("numeric", nrow(tempdat))

for (i in 1:length(unique(tempdat$exposure))) tempdat$w.numerator[with(tempdat,

exposure == sort(unique(tempdat$exposure))[i])] <- pred1[tempdat$exposure ==

sort(unique(tempdat$exposure))[i], i]

mod1$call$formula <- eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$numerator,

width.cutoff = 500), sep = "")))

mod1$call$data <- tempcall$data

}

mod2 <- multinom(formula = eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$denominator,

width.cutoff = 500), sep = ""))), data = data, na.action = na.fail,

...)

pred2 <- as.data.frame(predict(mod2, type = "probs"))

tempdat$w.denominator <- vector("numeric", nrow(tempdat))

for (i in 1:length(unique(tempdat$exposure))) tempdat$w.denominator[with(tempdat,

exposure == sort(unique(tempdat$exposure))[i])] <- pred2[tempdat$exposure ==

sort(unique(tempdat$exposure))[i], i]

mod2$call$formula <- eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$denominator,

width.cutoff = 500), sep = "")))

mod2$call$data <- tempcall$data

tempdat$ipw.weights <- tempdat$w.numerator/tempdat$w.denominator

}

if (tempcall$family == "ordinal") {

if (tempcall$link == "logit")

m <- "logistic"

if (tempcall$link == "probit")

m <- "probit"

if (tempcall$link == "cloglog")

m <- "cloglog"

if (tempcall$link == "cauchit")

m <- "cauchit"

if (is.null(tempcall$numerator))

tempdat$p.numerator <- 1

else {

mod1 <- polr(formula = eval(parse(text = paste("as.factor(",

deparse(tempcall$exposure, width.cutoff = 500),

")", deparse(tempcall$numerator, width.cutoff = 500),

sep = ""))), data = data, method = m, na.action = na.fail,

...)

pred1 <- as.data.frame(predict(mod1, type = "probs"))

tempdat$w.numerator <- vector("numeric", nrow(tempdat))

for (i in 1:length(unique(tempdat$exposure))) tempdat$w.numerator[with(tempdat,

exposure == sort(unique(tempdat$exposure))[i])] <- pred1[tempdat$exposure ==

sort(unique(tempdat$exposure))[i], i]

mod1$call$formula <- eval(parse(text = paste("as.factor(",

deparse(tempcall$exposure, width.cutoff = 500),

")", deparse(tempcall$numerator, width.cutoff = 500),

sep = "")))

mod1$call$data <- tempcall$data

mod1$call$method <- m

}

mod2 <- polr(formula = eval(parse(text = paste("as.factor(",

deparse(tempcall$exposure, width.cutoff = 500), ")",

deparse(tempcall$denominator, width.cutoff = 500),

sep = ""))), data = data, method = m, na.action = na.fail,

...)

pred2 <- as.data.frame(predict(mod2, type = "probs"))

tempdat$w.denominator <- vector("numeric", nrow(tempdat))

for (i in 1:length(unique(tempdat$exposure))) tempdat$w.denominator[with(tempdat,

exposure == sort(unique(tempdat$exposure))[i])] <- pred2[tempdat$exposure ==

sort(unique(tempdat$exposure))[i], i]

mod2$call$formula <- eval(parse(text = paste("as.factor(",

deparse(tempcall$exposure, width.cutoff = 500), ")",

deparse(tempcall$denominator, width.cutoff = 500),

sep = "")))

mod2$call$data <- tempcall$data

mod2$call$method <- m

tempdat$ipw.weights <- tempdat$w.numerator/tempdat$w.denominator

}

if (tempcall$family == "gaussian") {

mod1 <- glm(formula = eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$numerator,

width.cutoff = 500), sep = ""))), data = data, na.action = na.fail,

...)

tempdat$w.numerator <- dnorm(tempdat$exposure, predict(mod1),

sd(mod1$residuals))

mod1$call$formula <- eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$numerator,

width.cutoff = 500), sep = "")))

mod1$call$data <- tempcall$data

mod2 <- glm(formula = eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$denominator,

width.cutoff = 500), sep = ""))), data = data, na.action = na.fail,

...)

tempdat$w.denominator <- dnorm(tempdat$exposure, predict(mod2),

sd(mod2$residuals))

mod2$call$formula <- eval(parse(text = paste(deparse(tempcall$exposure,

width.cutoff = 500), deparse(tempcall$denominator,

width.cutoff = 500), sep = "")))

mod2$call$data <- tempcall$data

tempdat$ipw.weights <- tempdat$w.numerator/tempdat$w.denominator

}

if (sum(is.na(tempdat$ipw.weights)) > 0)

stop("NA's in weights!")

if (!(is.null(tempcall$trunc))) {

tempdat$weights.trunc <- tempdat$ipw.weights

tempdat$weights.trunc[tempdat$ipw.weights <= quantile(tempdat$ipw.weights,

0 + trunc)] <- quantile(tempdat$ipw.weights, 0 +

trunc)

tempdat$weights.trunc[tempdat$ipw.weights > quantile(tempdat$ipw.weights,

1 - trunc)] <- quantile(tempdat$ipw.weights, 1 -

trunc)

}

if (is.null(tempcall$trunc)) {

if (is.null(tempcall$numerator))

return(list(ipw.weights = tempdat$ipw.weights, call = tempcall,

den.mod = mod2))

else return(list(ipw.weights = tempdat$ipw.weights, call = tempcall,

num.mod = mod1, den.mod = mod2))

}

else {

if (is.null(tempcall$numerator))

return(list(ipw.weights = tempdat$ipw.weights, weights.trunc = tempdat$weights.trunc,

call = tempcall, den.mod = mod2))

else return(list(ipw.weights = tempdat$ipw.weights, weights.trunc = tempdat$weights.trunc,

call = tempcall, num.mod = mod1, den.mod = mod2))

}

}