Inputs

`df` :	The data frame
`treatment` :	The name of the treatment variable as a character string, or the number of the treatment variable column
`t1` :	The name of the value to be used as group 1, so that delta has the desired sign.
`response` :	The name of the response variable as a character string, or the number of the response variable column
`X` :	A character vector with the names of explanatory variables to use, or a numeric vector with the column numbers of the explanatory variables
`support` :	Observations with estimated propensity scores not between $(1 - \texttt{support})/2$ and $(1 + \texttt{support})/2$ will be removed. Default is 0.99.
`alpha` :	Numeric. Used to create a (1 - `alpha`) confidence interval. Default is 0.05
`r1` :	Optional. If response is binary, `r1` specifies the name of the outcome to be considered 1 in P(Y = 1).

Example:

```
DR.est(df, treatment = ‘Major’, t1 = ‘31FYC’, response = ‘semesterGPA’,
       X = c(‘SATVerb’, ‘SATMath’,‘HighSchoolGPA’ ,‘Gender’), support = 0.98)
```

Outputs

The function returns a list with the following elements

`estimates` :	A vector containing $\hat{\mu}_1, \hat{\mu}_0, \hat{\Delta}$, The lower and upper bounds of a $(1 - \texttt{alpha})$ confidence interval, $\hat{SE}$, p-value, and `alpha`
`removed` :	A matrix showing the number and percent of observations removed for missing values and extreme propensity scores
`model0` :	The `glm` object from the regression of the response on `X` for the untreated
`model1` :	The `glm` object from the regression of the response on `X` for the treated
`modelZ` :	The `glm` object from the regression of the treatment on `X`
`e0` :	Vector of estimated propensity scores for the treated
`e1` :	Vector of estimated propensity scores for the untreated

```r
DR.est <- function(df,  treatment, t1, response, X, support = 0.99, alpha = 0.05, r1 = F){
    logistic <- function(x) 1/(1 + exp(-x))
    miss     <- as.logical(rowSums(is.na(df[,c(treatment, response, X)])))
    orig.n   <- nrow(df)
    if(any(miss)) df <- df[!miss,]

    if(length(table(df[,treatment])) != 2) stop('Treatment variable is not binary.')
    df[, treatment] <- ifelse(df[,treatment] == t1, 1, 0)
    trt <- df[,treatment] == 1

    modelZ <- glm(reformulate(X, treatment), data = df, family = binomial)
    e      <- logistic(predict(modelZ, df))
    trim   <- e < (1 - support)/2 | e > (1 + support)/2
    if(any(trim)){
        df <- df[!trim,]
        e  <- e[!trim]
    }

    n.ylev <- length(table(df[,response]))
    if(n.ylev == 2){
        cat('Response assumed binary.','\n')
        if(r1 == F & !is.numeric(df[, response])){
            r1 <- names(table(df[, response]))[1]
            cat(paste0('Using indicator of response = ', r1,'.'),'\n')
        }
        df[, response] <- ifelse(df[, response] == r1, 1, 0)
        model0 <- glm(reformulate(X, response), data = df[!trt, ], family = binomial)
        model1 <- glm(reformulate(X, response), data = df[trt, ], family = binomial)
    } else if(!is.numeric(df[, treatment])){
        stop('Response is character/factor with !=2 levels. Response must be binary or continuous.
    } else {
        cat('Response assumed continuous.','\n')
        split(df)
        model0 <- glm(reformulate(X, response), data = df[!trt, ])
        model1 <- glm(reformulate(X, response), data = df[trt, ])
    }

    m0 <- predict(model0, df)
    m1 <- predict(model1, df)
    e0 <- e[!trt]
    e1 <- e[trt]
    Z  <- df[, treatment]
    Y  <- df[, response]
    if(n.ylev == 2){
        m0 <- logistic(m0)
        m1 <- logistic(m1)
    }

    d0    <- ((1 - Z)*Y + (Z - e)*m0)/(1 - e)
    d1    <- (Z*Y - (Z - e)*m1)/e
    mu0   <- mean(d0)
    mu1   <- mean(d1)
    delta <- mu1 - mu0
    SE    <- sqrt(sum(((d1 - d0) - delta)^2))/nrow(df)
    p     <- 2*min(pnorm(c(-1,1)*delta/SE))
    conf  <- delta + c(-1,1)*qnorm(1 - alpha/2)*SE
    names(conf) <- c('lower','upper')

    est <- round(c(mu0 = mu0, mu1 = mu1, delta = delta, conf, SE = SE, p = p, alpha = alpha), 4)
    print(est);cat('\n')
    rem <- c(missing = sum(miss), extreme.ps = sum(trim))
    rem <- rbind(n = rem, '%' = round(100*rem/orig.n,1))
    cat('Removed Observations:','\n')
    print(rem)

    return(list(estimates = est, removed = rem,model0 = model0, model1 = model1, modelZ = modelZ,
}
```