Teacher Awareness of ASD: Reproducible Report

Contents

1 Settings

1.1 Clear working environment

```
rm(list = ls())
```

1.2 Set a random seed

```
set.seed(123)
```

1.3 Required packages

rcompanion : Compute effect sizes following nonparametric tests

ggplot2: Create jitter and median plots

FSA : Perform Dunn's post-hoc pairwise analysis

boot : Perform bootstrapping

```
packages <- c("rcompanion","ggplot2","FSA","boot")

for (package in packages) {

  if (!requireNamespace(package, quietly = TRUE))
    install.packages(package)

  suppressPackageStartupMessages(library(package, character.only = TRUE))

}
```

```
## Warning: package 'rcompanion' was built under R version 4.4.3

## Warning: package 'ggplot2' was built under R version 4.4.3

## Warning: package 'FSA' was built under R version 4.4.3

## Warning: package 'boot' was built under R version 4.4.3
```

1.4 Import and inspect data attributes

```r
#load data and set seed
df <- read.csv("teacher_awareness_dataset.csv")

attributes(df)[names(attributes(df)) != "row.names"]
```

```
## $names
##  [1] "ID"                    "gender_male"           "bachelor_or_above"
##  [4] "experience"            "heard"                 "contact"
##  [7] "awareness"             "social_media"          "colleagues"
## [10] "scientific_resources" "workshops"             "personal_experience"
##
## $class
## [1] "data.frame"
```

2 Teacher awareness across grouping variables

2.1 Define Variables, result objects, and helper functions

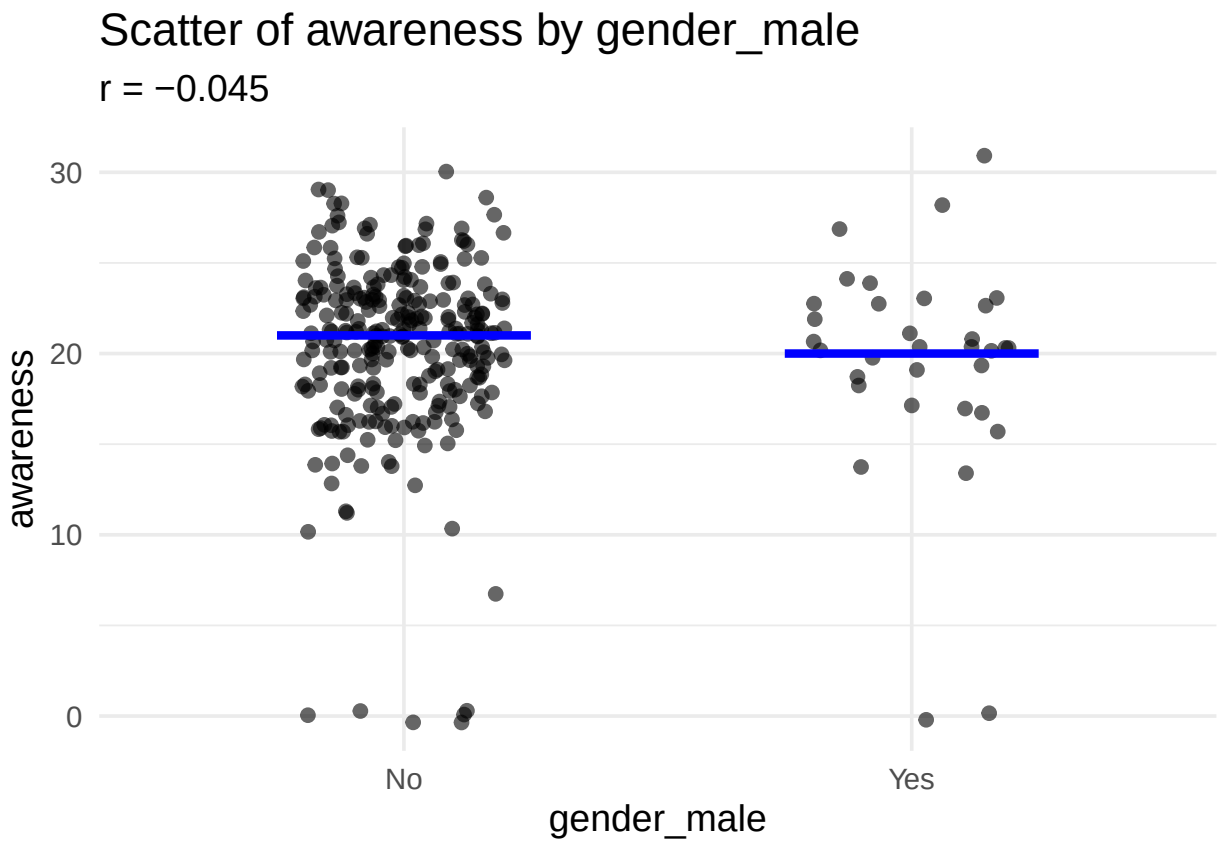
```
# Store the outcome and grouping variables
y_var          <- "awareness"
binary_groups  <- c("gender_male","bachelor_or_above","social_media", "colleagues",
                    "scientific_resources", "workshops", "personal_experience")
multi_groups   <- c("contact")
```

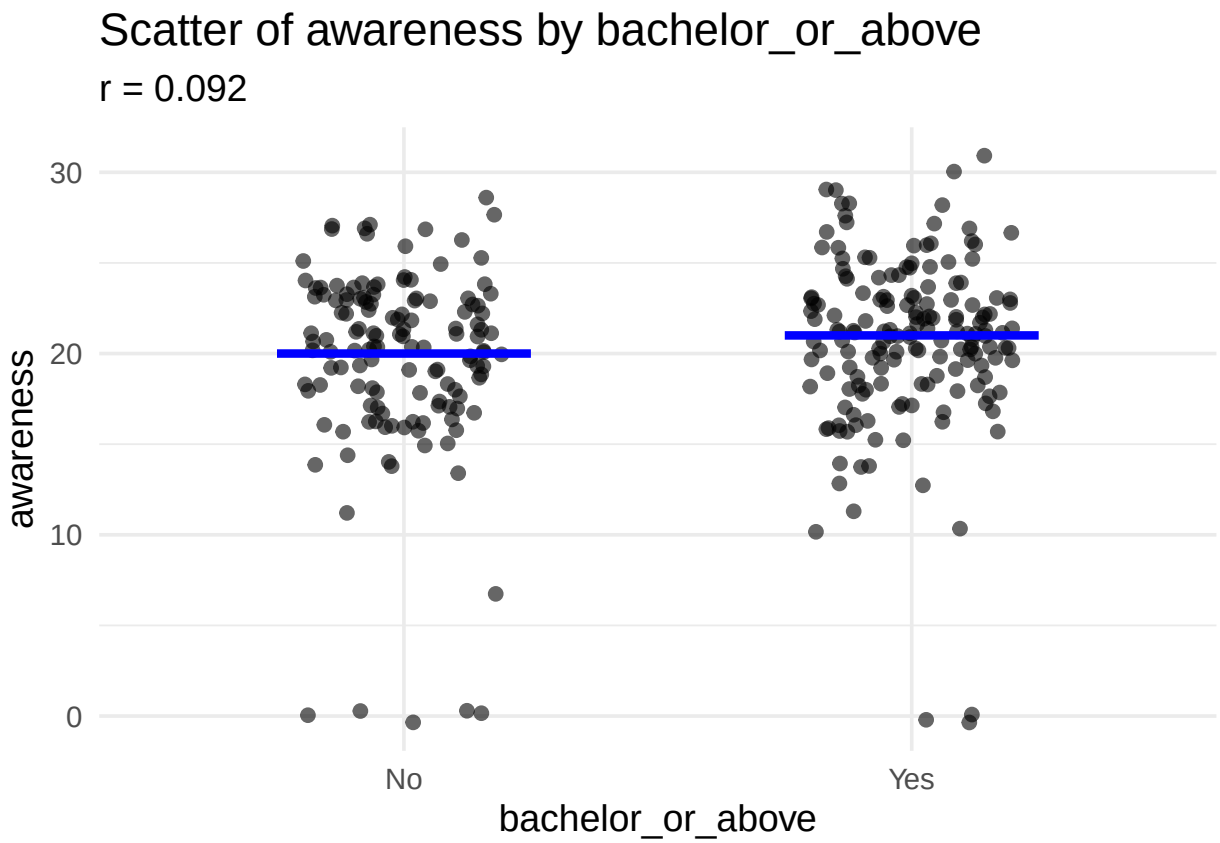
```
# Prepare an empty frame for Man-Whitney U test results
results_binary <- data.frame(group       = character(),
                             U_stat      = numeric(),
                             p_value     = numeric(),
                             r_value  =    numeric(),
                             CI_lower_95 = numeric(),
                             CI_upper_95 = numeric(),
                             stringsAsFactors = FALSE
                             )

# Prepare an empty data frame for Krustkal-Wallis test results
results_multi <- data.frame(group        = character(),
                            H_value      = numeric(),
                            df           = integer(),
                            p_value      = numeric(),
                            eps_sq       = numeric(),
                            CI_lower_95  = numeric(),
                            CI_upper_95  = numeric(),
                            stringsAsFactors = FALSE
                            )

# Prepare an empty data frame for Dunn's test pairwise analysis
results_pair <- data.frame(group_1     = character(),
                           group_2     = character(),
                           p_value     = numeric(),
                           p_adjusted  = numeric(),
                           r_value     = numeric(),
                           CI_lower_95 = numeric(),
                           CI_upper_95 = numeric(),
                           stringsAsFactors = FALSE
                           )
```

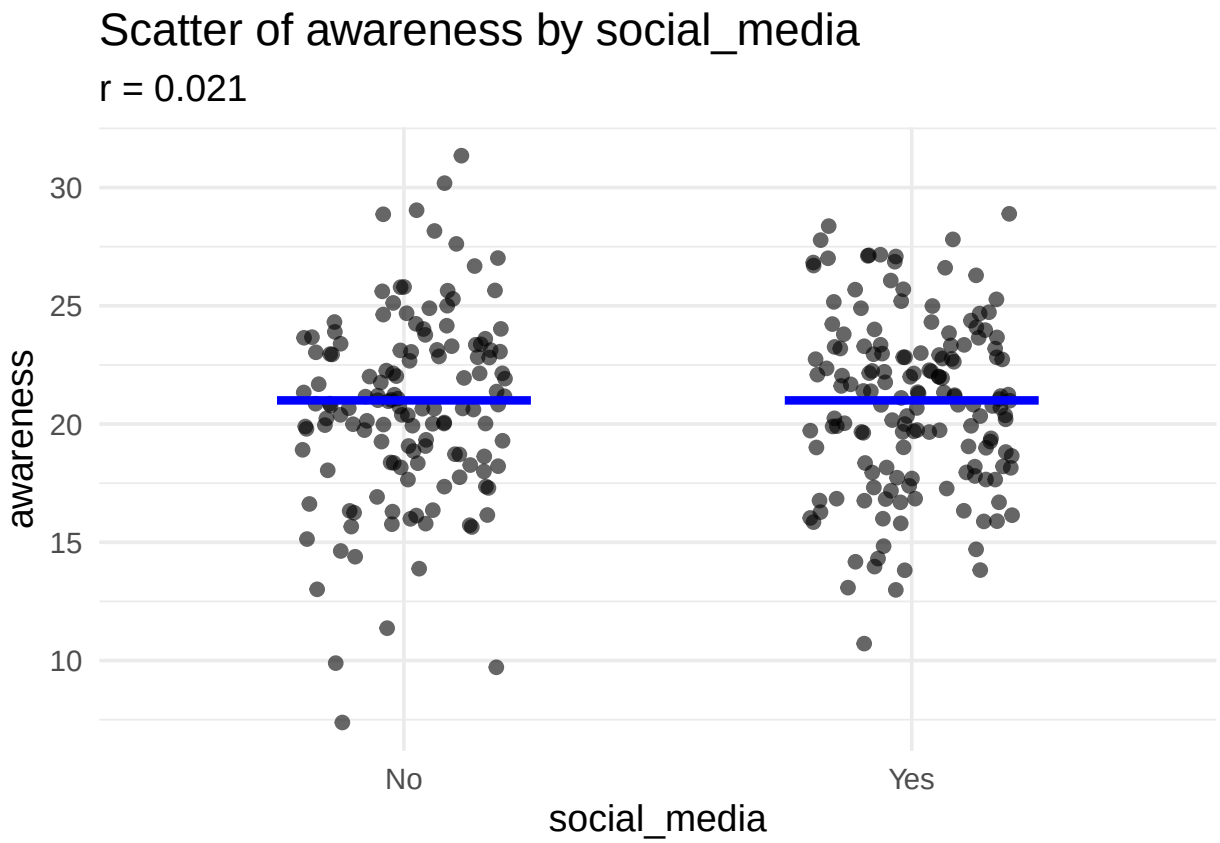
```r
# Define a helper function to perform bootstrapping
boot_fn <- function(data, indices) {
  dBoot <- data[indices, ]

  # Recompute ranks of y within the bootstrap sample
  dBoot$rank_y <- rank(dBoot$y)

  dt <- dunnTest(y ~ g, data = dBoot, method = "none")$res
  row_i <- dt[dt$Comparison == comparison, ]
  if (nrow(row_i) == 0) {
    return(NA_real_)
  }
  Z_boot <- row_i$Z

  # Compute mean-rank difference in this bootstrap replicate
  mean_rank_1_boot <- mean(dBoot$rank_y[dBoot$g == group_1], na.rm = TRUE)
  mean_rank_2_boot <- mean(dBoot$rank_y[dBoot$g == group_2], na.rm = TRUE)
  mean_rank_diff_boot <- mean_rank_2_boot - mean_rank_1_boot

  r_boot <- (abs(Z_boot) / sqrt(nrow(dBoot))) * sign(mean_rank_diff_boot)
  return(r_boot)
}
```

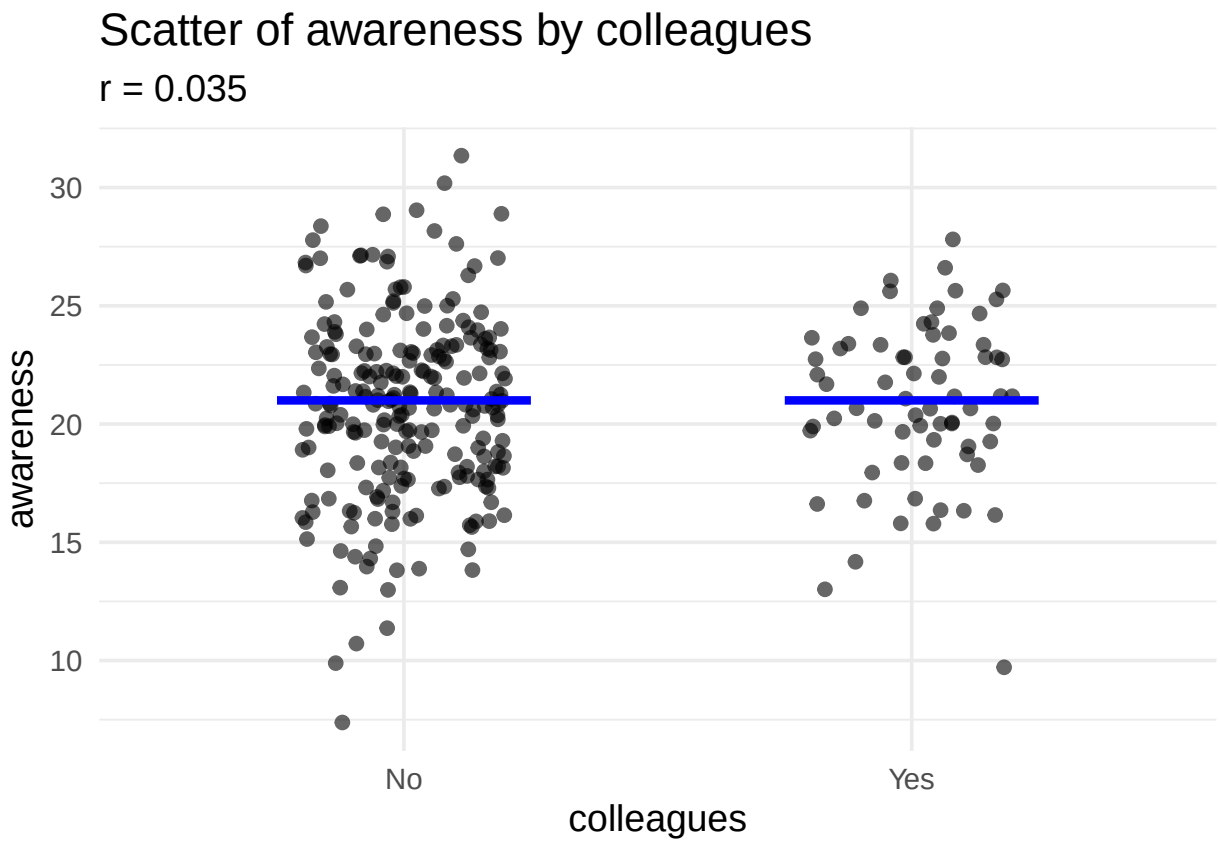
2.2 Wilixicon R effect size for binary grouping variables

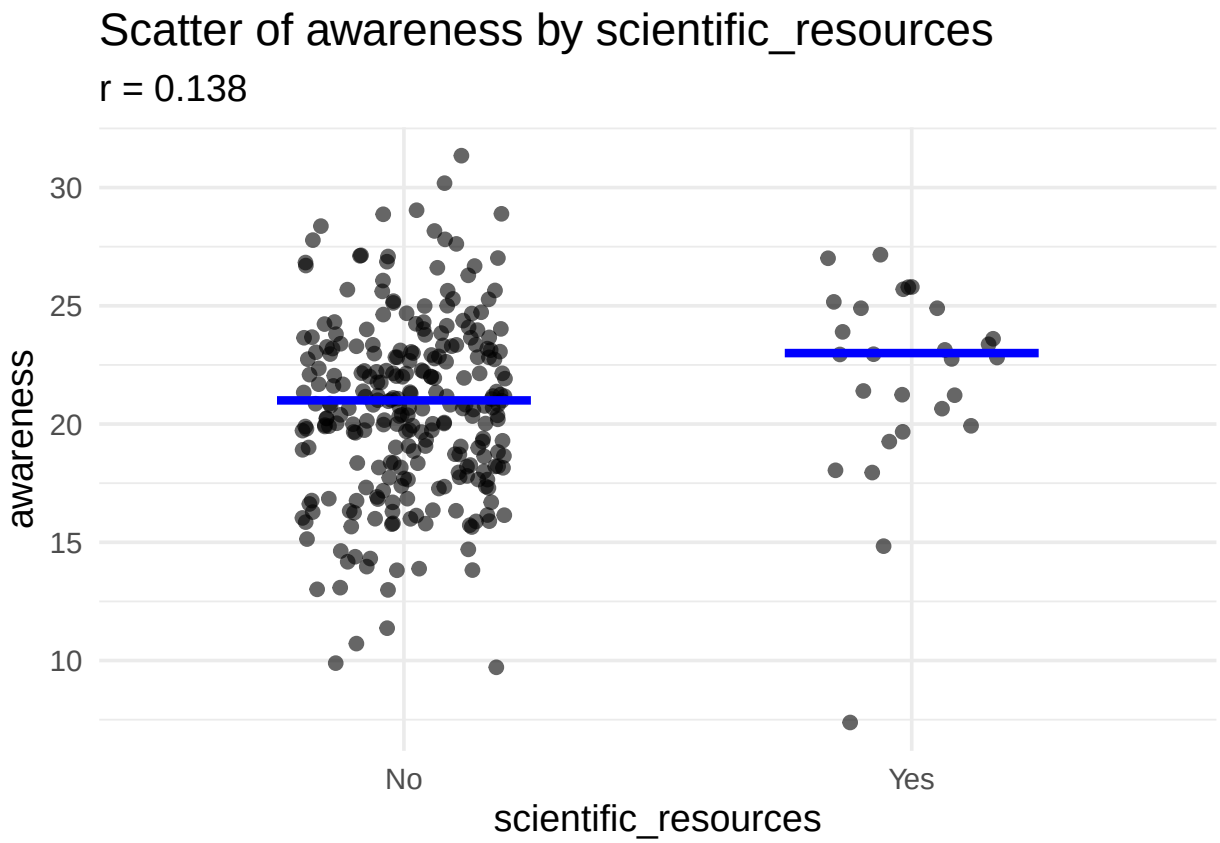
```r
# Loop over binary grouping variables
for (g in binary_groups) {

  # Subset and clean the data
  sub <- df[, c(y_var, g)]
  names(sub) <- c("y","g")
  sub <- na.omit(sub)
  sub$g <- factor(sub$g, levels = c("No","Yes"))

  # Mann-Whitney test
  wt <- wilcox.test(y ~ g, data = sub, exact = FALSE)

  # R-values effect size
  set.seed(123)
  r <- -1*wilcoxonR(x          = sub$y,
                    g          = sub$g,
                    ci         = TRUE,
                    conf.level = 0.95,
                    type       = "bca",
                    R          = 10000
                    )
  r[c(2, 3)] <- r[c(3, 2)]

  # Jitter and median crossbar plots
  p <- ggplot(sub, aes(x = g, y = y)) +
        geom_jitter(width = 0.2, alpha = 0.6) +
        stat_summary(fun = median,
                     geom = "crossbar",
                     width = 0.5,
                     color = "blue",
                     linewidth = 0.6
                     ) +
        labs(title    = paste0("Scatter of ", y_var, " by ", g),
             subtitle = paste0("r = ", round(r, 3)),
             x        = g,
             y        = y_var
             ) +
        theme_minimal(base_size = 14)

  print(p)

  # Append result
  results_binary <- rbind(results_binary,
                          data.frame(group       = g,
                                     U_stat      = as.numeric(wt$statistic),
                                     p_value     = wt$p.value,
                                     r_value     = r,
                                     stringsAsFactors = FALSE
                                     )
                          )
}
```

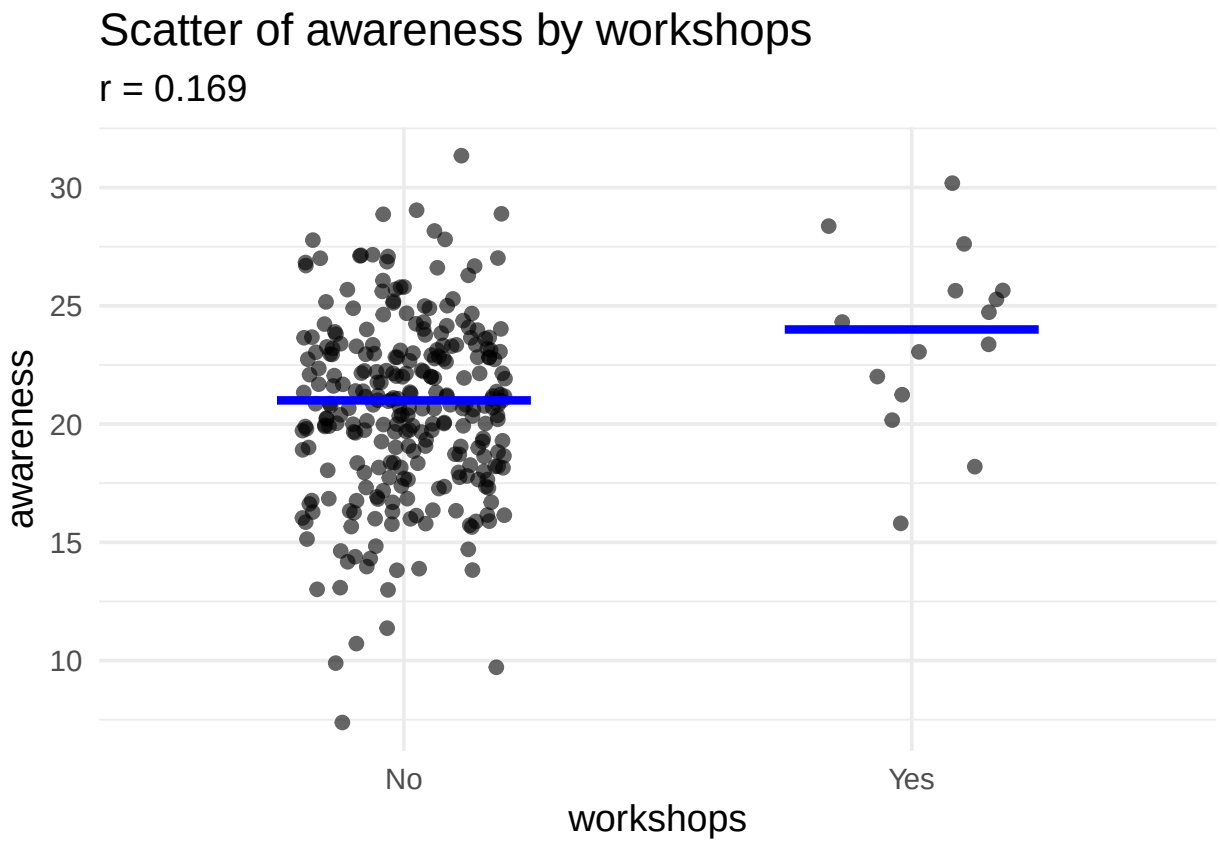

Scatter of awareness by gender_male
r = −0.045
30
20
10
0
awareness
No
Yes
gender_male


Scatter of awareness by bachelor_or_above
r = 0.092
30
20
10
0
awareness
No
Yes
bachelor_or_above


Scatter of awareness by social_media
r = 0.021
awareness
30
25
20
15
10
No
Yes
social_media


Scatter of awareness by colleagues
r = 0.035
awareness
30
25
20
15
10
No
Yes
colleagues


Scatter of awareness by scientific_resources
r = 0.138
awareness
30
25
20
15
10
No
Yes
scientific_resources

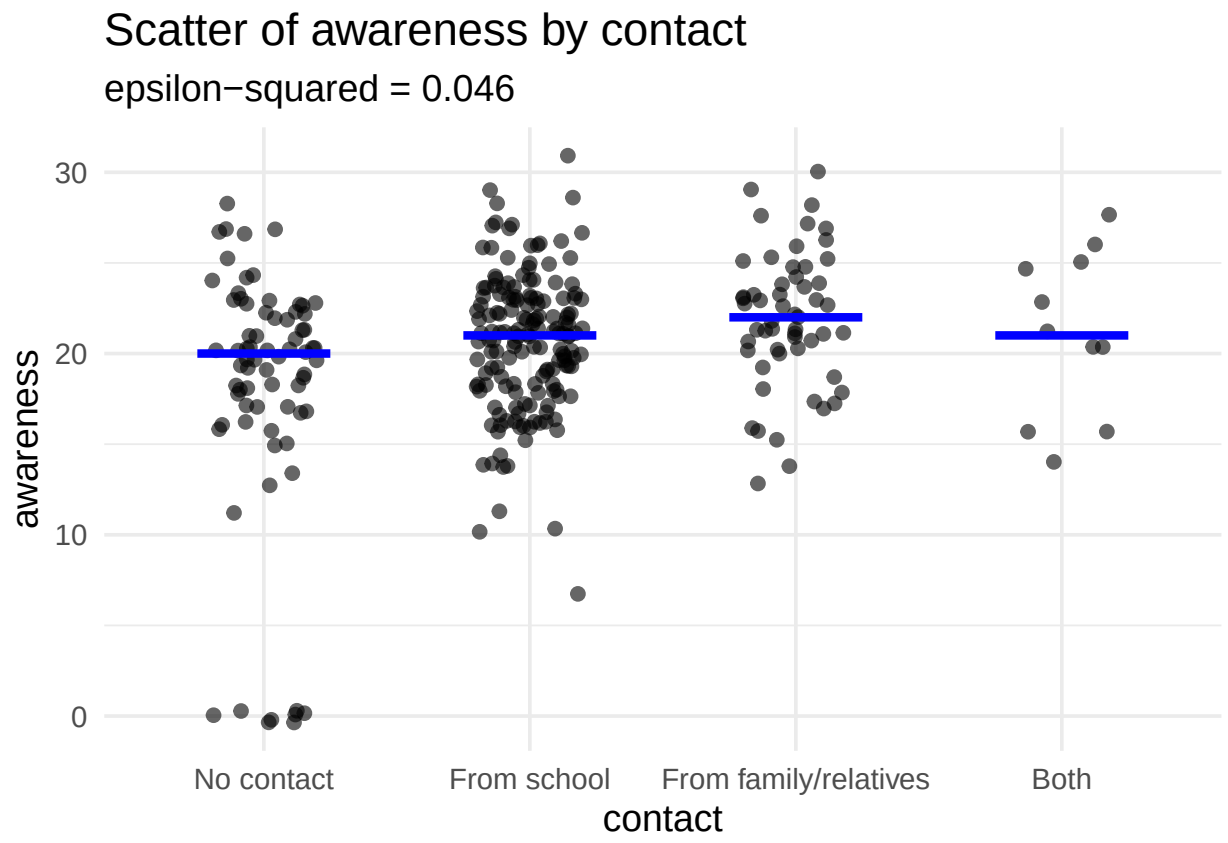

Scatter of awareness by workshops
r = 0.169
awareness
30
25
20
15
10
No
Yes
workshops


```
# Print results
print(results_binary)
```

```
##                  group  U_stat    p_value r_value.r r_value.lower.ci
## 1          gender_male  4788.5 0.43617654   -0.0450          -0.1560
## 2    bachelor_or_above 10012.5 0.11113222    0.0916          -0.0211
## 3         social_media 10423.0 0.71569459    0.0213          -0.0946
## 4           colleagues  7279.5 0.54401506    0.0355          -0.0771
## 5 scientific_resources  2599.0 0.01763696    0.1380           0.0191
## 6            workshops  1165.0 0.00387354    0.1690           0.0475
## 7  personal_experience 10494.0 0.77253241    0.0169          -0.0946
##   r_value.upper.ci
## 1           0.0669
## 2           0.2030
## 3           0.1360
## 4           0.1480
## 5           0.2450
## 6           0.2750
## 7           0.1310
```

2.3 Epsilon-squared effect size for multi-level grouping variables

```r
# Subset and clean the data
sub        <- df[, c(y_var, "contact")]
names(sub) <- c("y", "g")
sub        <- na.omit(sub)
sub$y      <- as.numeric(sub$y)
sub$g      <- factor(sub$g, levels = c("No contact", "From school",
                                       "From family/relatives", "Both"))

# Kruskal-Wallis test
kw <- kruskal.test(y ~ g, data = sub)

# Epsilon-squared effect size
set.seed(123)
eps <- epsilonSquared(x           = sub$y,
                      g           = sub$g,
                      ci          = TRUE,
                      conf.level  = 0.95,
                      type        = "bca",
                      R           = 10000
                      )

# Jitter and median crossbar plots
p <- ggplot(sub, aes(x = g, y = y)) +
  geom_jitter(width = 0.2, alpha = 0.6) +
  stat_summary(fun = median,
               geom = "crossbar",
               width = 0.5,
               color = "blue",
               linewidth = 0.6
               ) +
  labs(title    = paste0("Scatter of ", y_var, " by contact"),
       subtitle = paste0("epsilon-squared = ", round(eps, 3)),
       x        = "contact",
       y        = y_var
       ) +
  theme_minimal(base_size = 14)

print(p)
```




```
# Store results
results_multi <- rbind(results_multi,
                       data.frame(group   = "contact",
                                  H_value = as.numeric(kw$statistic),
                                  f       = kw$parameter,
                                  p_value = kw$p.value,
                                  eps_sq  = eps,
                                  stringsAsFactors = FALSE
                                  )
                       )

# Print results
print(results_multi)
```

```
##      group  H_value f     p_value eps_sq.epsilon.squared eps_sq.lower.ci
## df contact 13.72453 3 0.003305138                 0.0457         0.00857
##    eps_sq.upper.ci
## df          0.0922
```

2.4 Pairwise Effect Sizes (r) Following Dunn's Test

```
# Subset and clean the data
sub <- na.omit(df[, c(y_var, multi_groups)])
names(sub) <- c("y", "g")
sub$g <- factor(sub$g)
N <- nrow(sub)

# Precompute overall ranks of y for the observed data
sub$rank_y <- rank(sub$y)

# Dunn's test
dunn_result <- dunnTest(y ~ g, data = sub, method = "bonferroni")
dunn_table  <- dunn_result$res

# Loop over each pairwise comparison
for (i in seq_len(nrow(dunn_table))) {

  # Extract the two group names, raw p-value, adjusted p, and Z
  comparison <- dunn_table$Comparison[i]
  groups     <- strsplit(comparison, " - ")[[1]]
  group_1    <- groups[1]
  group_2    <- groups[2]

  p_raw <- dunn_table$P.unadj[i]
  p_adj <- dunn_table$P.adj[i]
  Z_obs <- dunn_table$Z[i]

  # Mean rank difference
  mean_rank_1 <- mean(sub$rank_y[sub$g == group_1], na.rm = TRUE)
  mean_rank_2 <- mean(sub$rank_y[sub$g == group_2], na.rm = TRUE)
  mean_rank_diff <- mean_rank_2 - mean_rank_1

  # R-values effect size
  r_obs <- (abs(Z_obs) / sqrt(N)) * sign(mean_rank_diff)

  # Bootstrapping with 10,000 samples
  set.seed(123)
  boot_out <- boot(data = sub, statistic = boot_fn, R = 10000)

  # 95% BCa interval for R-values
  ci_obj <- tryCatch(boot.ci(boot_out, type = "bca"),
                     error = function(e) NULL
                     )
  if (!is.null(ci_obj) && !is.null(ci_obj$bca)) {
    ci_lower <- ci_obj$bca[4]  # 2.5% endpoint
    ci_upper <- ci_obj$bca[5]  # 97.5% endpoint

  } else {
    ci_lower <- NA_real_
    ci_upper <- NA_real_
  }
```

```r
  # Append results
  results_pair <- rbind(results_pair,
                        data.frame(group_1     = group_1,
                                   group_2     = group_2,
                                   p_value     = p_raw,
                                   p_adjusted  = p_adj,
                                   r_value     = r_obs,
                                   CI_lower_95 = ci_lower,
                                   CI_upper_95 = ci_upper,
                                   stringsAsFactors = FALSE
                                   )
  )
}

# Print results
print(results_pair)
```

```
##                 group_1               group_2      p_value  p_adjusted
## 1                  Both From family/relatives 0.6057755097 1.000000000
## 2                  Both           From school 0.6285768967 1.000000000
## 3 From family/relatives           From school 0.0395801775 0.237481065
## 4                  Both            No contact 0.1385118147 0.831070888
## 5 From family/relatives            No contact 0.0002746757 0.001648054
## 6           From school            No contact 0.0198269216 0.118961529
##       r_value CI_lower_95  CI_upper_95
## 1  0.02974826 -0.09137443  0.160411104
## 2 -0.02788178 -0.15044011  0.103695143
## 3 -0.11862713 -0.22419182 -0.006309852
## 4 -0.08538392 -0.21515309  0.043302626
## 5 -0.20969557 -0.31555788 -0.100009633
## 6 -0.13427633 -0.24058563 -0.019695093
```