Statistical Analysis for Quantitative and Qualitative Data in Shiny

Jerson Romario Gomez Cahuana

University National of the Altiplano - Puno

Faculty of Statistical and Computer Engineering

Course: Computational Statistics

Teacher: Fred Torrez Cruz

1. Statistical analysis for quantitative and qualitative variables in Shiny

Shiny is a web application framework for R. It helps us create interactive web apps to show data, graphs, and analysis.

In this project, we will use Shiny to do statistical analysis of two types of data:

- Quantitative variables: These are numbers. For example, age, weight, or income.
- Qualitative variables: These are categories. For example, gender, color, or favorite subject.

With Shiny, we can:

- Show tables and charts.
- Calculate mean, median, mode.
- Create bar plots, pie charts, and histograms.
- Compare data using filters and inputs.

2. Code in Shiny

2.1. This code loads the main R packages used in my project. The shiny and shinydashboard packages are used to create interactive web applications. The DT package helps to show data tables in a nice way. The ggplot2 package is used to make beautiful graphs. The nortest package is used for normality tests. Finally, the car package gives extra tools for regression and data analysis. These libraries are important to run my R code correctly. You can find it at the following URL https://cuervo.shinyapps.io/datos_cuantitativos/

```
library(shiny)
library(shinydashboard)
library(DT)
library(ggplot2)
library(nortest)
library(car)
library(psych)
library(moments)

ui <- dashboardPage(
  dashboardHeader(title = "Análisis Datos Cuantitativos"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Cargar Datos", tabName = "datos", icon = icon("database")),
      menuItem("Estadística Descriptiva", tabName = "descriptiva", icon = icon("chart-bar"))
      menuItem("Distribución y Normalidad", tabName = "normalidad", icon = icon("bell")),
      menuItem("Comparación de Grupos", tabName = "comparacion", icon = icon("not-equal")),
      menuItem("Correlación y Regresión", tabName = "regresion", icon = icon("project-diagra
    )
  ),
  dashboardBody(
    tabItems(
      tabItem(tabName = "datos",
              fluidRow(
                box(title = "Cargar Datos", width = 12,
                    fileInput("file1", "Seleccione archivo CSV",
                              accept = c("text/csv",
                                         "text/comma-separated-values,text/plain",
                                         ".csv")),
                    checkboxInput("header", "Encabezado", TRUE),
                    radioButtons("sep", "Separador",
                                 choices = c(Coma = ",",
                                             PuntoComa = ";",
                                             Tabulador = "\t"),
```

```
                                selected = ","),
                    radioButtons("quote", "Comillas",
                                choices = c(Ninguna = "",
                                            "Doble" = '"',
                                            "Simple" = "'"),
                                selected = '"')
                  ),
                  box(title = "Vista previa", width = 12,
                      DTOutput("contents")
                  )
                )
      ),

      tabItem(tabName = "descriptiva",
                fluidRow(
                  box(title = "Opciones Descriptivas", width = 4,
                      selectInput("var_cuanti", "Seleccione variable cuantitativa:", choices =
                      checkboxGroupInput("estadisticos", "Estadísticos a mostrar:",
                                         choices = c("Medidas de tendencia central" = "central"
                                                     "Medidas de dispersión" = "dispersion",
                                                     "Medidas de forma" = "forma",
                                                     "Cuartiles y percentiles" = "cuantiles"),
                                         selected = c("central", "dispersion")),
                      radioButtons("tipo_grafico", "Tipo de gráfico:",
                                   choices = c("Histograma" = "hist",
                                               "Boxplot" = "box",
                                               "Densidad" = "dens",
                                               "Q-Q Plot" = "qq"),
                                   selected = "hist")
                  ),
                  box(title = "Resultados Descriptivos", width = 8,
                      verbatimTextOutput("resumen_descriptivo"),
                      plotOutput("grafico_descriptivo"),
                      DTOutput("tabla_descriptiva")
                  )
                )
      ),

      tabItem(tabName = "normalidad",
                fluidRow(
                  box(title = "Opciones de Normalidad", width = 4,
                      selectInput("var_normal", "Variable para análisis de normalidad:", choic
```

```
                radioButtons("prueba_normal", "Prueba de normalidad:",
                             choices = c("Shapiro-Wilk" = "shapiro",
                                         "Kolmogorov-Smirnov" = "ks",
                                         "Anderson-Darling" = "ad",
                                         "Lilliefors" = "lillie")),
                actionButton("calcular_normal", "Calcular")
            ),
            box(title = "Resultados de Normalidad", width = 8,
                verbatimTextOutput("resultado_normalidad"),
                plotOutput("grafico_normalidad")
            )
        )
    ),

    tabItem(tabName = "comparacion",
        fluidRow(
            box(title = "Opciones de Comparación", width = 4,
                selectInput("var_resp", "Variable respuesta:", choices = NULL),
                selectInput("var_grupo", "Variable de agrupación:", choices = NULL),
                radioButtons("prueba_comp", "Prueba estadística:",
                             choices = c("t-test (2 grupos)" = "t.test",
                                         "ANOVA (múltiples grupos)" = "anova",
                                         "Welch (varianzas desiguales)" = "welch",
                                         "Kruskal-Wallis (no paramétrico)" = "kruskal"))
                conditionalPanel(
                  condition = "input.prueba_comp == 't.test'",
                  radioButtons("tipo_t", "Tipo de t-test:",
                               choices = c("Muestras independientes" = "indep",
                                           "Muestras pareadas" = "pareado"))
                ),
                actionButton("calcular_comp", "Calcular")
            ),
            box(title = "Resultados de Comparación", width = 8,
                verbatimTextOutput("resultado_comparacion"),
                plotOutput("grafico_comparacion")
            )
        )
    ),

    tabItem(tabName = "regresion",
        fluidRow(
            box(title = "Opciones de Análisis", width = 4,
```

```
                    selectInput("var_x", "Variable X (predictora):", choices = NULL),
                    selectInput("var_y", "Variable Y (respuesta):", choices = NULL),
                    radioButtons("tipo_analisis", "Tipo de análisis:",
                                 choices = c("Correlación" = "cor",
                                             "Regresión lineal" = "reg",
                                             "Gráfico de dispersión" = "disp")),
                    conditionalPanel(
                      condition = "input.tipo_analisis == 'cor'",
                      radioButtons("metodo_cor", "Método de correlación:",
                                   choices = c("Pearson" = "pearson",
                                               "Spearman" = "spearman",
                                               "Kendall" = "kendall"))
                    ),
                    actionButton("calcular_reg", "Calcular")
                ),
                box(title = "Resultados", width = 8,
                    verbatimTextOutput("resultado_regresion"),
                    plotOutput("grafico_regresion")
                )
              )
      )
    )
  )
)

server <- function(input, output, session) {

  datos <- reactive({
    req(input$file1)
    df <- read.csv(input$file1$datapath,
                   header = input$header,
                   sep = input$sep,
                   quote = input$quote)
    updateSelectInput(session, "var_cuanti", choices = names(df)[sapply(df, is.numeric)])
    updateSelectInput(session, "var_normal", choices = names(df)[sapply(df, is.numeric)])
    updateSelectInput(session, "var_resp", choices = names(df)[sapply(df, is.numeric)])
    updateSelectInput(session, "var_grupo", choices = names(df)[!sapply(df, is.numeric)])
    updateSelectInput(session, "var_x", choices = names(df)[sapply(df, is.numeric)])
    updateSelectInput(session, "var_y", choices = names(df)[sapply(df, is.numeric)])
    return(df)
  })
```

```
output$contents <- renderDT({
  datatable(datos(), options = list(scrollX = TRUE))
})

output$resumen_descriptivo <- renderPrint({
  req(input$var_cuanti)
  var <- datos()[[input$var_cuanti]]

  cat("=== RESUMEN ESTADÍSTICO ===\n\n")

  if("central" %in% input$estadisticos) {
    cat("-- Medidas de tendencia central --\n")
    cat("Media:", mean(var, na.rm = TRUE), "\n")
    cat("Mediana:", median(var, na.rm = TRUE), "\n")
    cat("Moda:", names(sort(table(var), decreasing = TRUE))[1], "\n\n")
  }

  if("dispersion" %in% input$estadisticos) {
    cat("-- Medidas de dispersión --\n")
    cat("Desviación estándar:", sd(var, na.rm = TRUE), "\n")
    cat("Varianza:", var(var, na.rm = TRUE), "\n")
    cat("Coeficiente de variación:", sd(var, na.rm = TRUE)/mean(var, na.rm = TRUE)*100, "%
    cat("Rango:", range(var, na.rm = TRUE), "\n")
    cat("Rango intercuartílico (IQR):", IQR(var, na.rm = TRUE), "\n\n")
  }

  if("forma" %in% input$estadisticos) {
    cat("-- Medidas de forma --\n")
    cat("Asimetría (skewness):", skewness(var, na.rm = TRUE), "\n")
    cat("Curtosis:", kurtosis(var, na.rm = TRUE), "\n\n")
  }

  if("cuantiles" %in% input$estadisticos) {
    cat("-- Cuantiles --\n")
    print(quantile(var, na.rm = TRUE))
    cat("\nPercentiles (5%, 25%, 50%, 75%, 95%):\n")
    print(quantile(var, probs = c(0.05, 0.25, 0.5, 0.75, 0.95), na.rm = TRUE))
  }
})

output$grafico_descriptivo <- renderPlot({
  req(input$var_cuanti)
```

```
  var <- datos()[[input$var_cuanti]]
  df <- data.frame(valor = var)

  if(input$tipo_grafico == "hist") {
    ggplot(df, aes(x = valor)) +
      geom_histogram(aes(y = ..density..), bins = 30, fill = "steelblue", color = "white") +
      geom_density(alpha = 0.2, fill = "red") +
      labs(title = paste("Histograma de", input$var_cuanti), x = input$var_cuanti) +
      theme_minimal()
  } else if(input$tipo_grafico == "box") {
    ggplot(df, aes(y = valor)) +
      geom_boxplot(fill = "steelblue") +
      labs(title = paste("Boxplot de", input$var_cuanti), y = input$var_cuanti) +
      theme_minimal()
  } else if(input$tipo_grafico == "dens") {
    ggplot(df, aes(x = valor)) +
      geom_density(fill = "steelblue", alpha = 0.5) +
      labs(title = paste("Densidad de", input$var_cuanti), x = input$var_cuanti) +
      theme_minimal()
  } else if(input$tipo_grafico == "qq") {
    ggplot(df, aes(sample = valor)) +
      stat_qq() + stat_qq_line() +
      labs(title = paste("Q-Q Plot de", input$var_cuanti)) +
      theme_minimal()
  }
})

output$tabla_descriptiva <- renderDT({
  req(input$var_cuanti)
  var <- datos()[[input$var_cuanti]]
  df <- data.frame(
    Estadístico = c("Media", "Mediana", "Desviación", "Varianza", "Mínimo", "Máximo", "Asi
    Valor = c(mean(var, na.rm = TRUE),
              median(var, na.rm = TRUE),
              sd(var, na.rm = TRUE),
              var(var, na.rm = TRUE),
              min(var, na.rm = TRUE),
              max(var, na.rm = TRUE),
              skewness(var, na.rm = TRUE),
              kurtosis(var, na.rm = TRUE))
  )
  datatable(df, options = list(dom = 't'))
```

```
})

observeEvent(input$calcular_normal, {
  output$resultado_normalidad <- renderPrint({
    req(input$var_normal)
    var <- datos()[[input$var_normal]]

    if(input$prueba_normal == "shapiro") {
      cat("=== PRUEBA DE SHAPIRO-WILK ===\n")
      if(length(var) > 5000) {
        cat("Advertencia: Shapiro-Wilk es recomendable para muestras < 5000\n")
        cat("Considera usar Kolmogorov-Smirnov o Anderson-Darling\n\n")
      }
      prueba <- shapiro.test(var)
      print(prueba)
    } else if(input$prueba_normal == "ks") {
      cat("=== PRUEBA DE KOLMOGOROV-SMIRNOV ===\n")
      prueba <- ks.test(var, "pnorm", mean = mean(var), sd = sd(var))
      print(prueba)
    } else if(input$prueba_normal == "ad") {
      cat("=== PRUEBA DE ANDERSON-DARLING ===\n")
      prueba <- ad.test(var)
      print(prueba)
    } else if(input$prueba_normal == "lillie") {
      cat("=== PRUEBA DE LILLIEFORS ===\n")
      prueba <- lillie.test(var)
      print(prueba)
    }

    cat("\nINTERPRETACIÓN:\n")
    cat("p-value > 0.05: No se rechaza normalidad\n")
    cat("p-value <= 0.05: Se rechaza normalidad\n")
  })

  output$grafico_normalidad <- renderPlot({
    req(input$var_normal)
    var <- datos()[[input$var_normal]]
    df <- data.frame(valor = var)

    p1 <- ggplot(df, aes(x = valor)) +
      geom_histogram(aes(y = ..density..), bins = 30, fill = "steelblue", color = "white") +
      geom_density(color = "red") +
```

```
      labs(title = "Histograma con curva normal", x = input$var_normal) +
      theme_minimal()

    p2 <- ggplot(df, aes(sample = valor)) +
      stat_qq() + stat_qq_line() +
      labs(title = "Q-Q Plot") +
      theme_minimal()

    gridExtra::grid.arrange(p1, p2, ncol = 2)
  })
})

observeEvent(input$calcular_comp, {
  output$resultado_comparacion <- renderPrint({
    req(input$var_resp, input$var_grupo)
    resp <- datos()[[input$var_resp]]
    grupo <- datos()[[input$var_grupo]]

    if(input$prueba_comp == "t.test") {
      cat("=== PRUEBA T DE STUDENT ===\n")
      if(input$tipo_t == "indep") {
        prueba <- t.test(resp ~ grupo)
      } else {
        prueba <- t.test(resp ~ grupo, paired = TRUE)
      }
      print(prueba)
    } else if(input$prueba_comp == "anova") {
      cat("=== ANÁLISIS DE VARIANZA (ANOVA) ===\n")
      modelo <- aov(resp ~ grupo)
      print(summary(modelo))

      if(summary(modelo)[[1]]$`Pr(>F)`[1] < 0.05) {
        cat("\n=== PRUEBA POST-HOC (Tukey HSD) ===\n")
        print(TukeyHSD(modelo))
      }
    } else if(input$prueba_comp == "welch") {
      cat("=== PRUEBA T DE WELCH (VARIANZAS DESIGUALES) ===\n")
      prueba <- t.test(resp ~ grupo, var.equal = FALSE)
      print(prueba)
    } else if(input$prueba_comp == "kruskal") {
      cat("=== PRUEBA DE KRUSKAL-WALLIS (NO PARAMÉTRICA) ===\n")
      prueba <- kruskal.test(resp ~ grupo)
```

```
      print(prueba)
    }
  })

  output$grafico_comparacion <- renderPlot({
    req(input$var_resp, input$var_grupo)
    resp <- datos()[[input$var_resp]]
    grupo <- datos()[[input$var_grupo]]
    df <- data.frame(resp = resp, grupo = grupo)

    ggplot(df, aes(x = grupo, y = resp, fill = grupo)) +
      geom_boxplot() +
      stat_summary(fun = mean, geom = "point", shape = 20, size = 5, color = "red") +
      labs(title = paste("Comparación de", input$var_resp, "por", input$var_grupo),
           x = input$var_grupo, y = input$var_resp) +
      theme_minimal() +
      theme(legend.position = "none")
  })
})

observeEvent(input$calcular_reg, {
  output$resultado_regresion <- renderPrint({
    req(input$var_x, input$var_y)
    x <- datos()[[input$var_x]]
    y <- datos()[[input$var_y]]

    if(input$tipo_analisis == "cor") {
      cat("=== ANÁLISIS DE CORRELACIÓN ===\n")
      cor_test <- cor.test(x, y, method = input$metodo_cor)
      print(cor_test)

      cat("\nCoeficiente de correlación:", cor_test$estimate, "\n")
      cat("Interpretación:\n")
      valor <- abs(cor_test$estimate)
      if(valor < 0.3) cat("Correlación muy débil\n")
      else if(valor < 0.5) cat("Correlación débil\n")
      else if(valor < 0.7) cat("Correlación moderada\n")
      else if(valor < 0.9) cat("Correlación fuerte\n")
      else cat("Correlación muy fuerte\n")
    } else if(input$tipo_analisis == "reg") {
      cat("=== REGRESIÓN LINEAL ===\n")
      modelo <- lm(y ~ x)
```

```
        print(summary(modelo))

        cat("\nEcuación de regresión:\n")
        cat("y =", coef(modelo)[1], "+", coef(modelo)[2], "* x\n")
      }
    })

    output$grafico_regresion <- renderPlot({
      req(input$var_x, input$var_y)
      x <- datos()[[input$var_x]]
      y <- datos()[[input$var_y]]
      df <- data.frame(x = x, y = y)

      if(input$tipo_analisis == "disp") {
        ggplot(df, aes(x = x, y = y)) +
          geom_point() +
          geom_smooth(method = "lm", se = TRUE, color = "red") +
          labs(title = paste("Diagrama de dispersión:", input$var_y, "vs", input$var_x),
               x = input$var_x, y = input$var_y) +
          theme_minimal()
      } else if(input$tipo_analisis == "reg") {
        modelo <- lm(y ~ x)
        df$predicted <- predict(modelo)
        df$residuals <- residuals(modelo)

        p1 <- ggplot(df, aes(x = x, y = y)) +
          geom_point() +
          geom_line(aes(y = predicted), color = "red") +
          labs(title = "Regresión lineal", x = input$var_x, y = input$var_y) +
          theme_minimal()

        p2 <- ggplot(df, aes(x = predicted, y = residuals)) +
          geom_point() +
          geom_hline(yintercept = 0, linetype = "dashed") +
          labs(title = "Residuos vs Ajustados", x = "Valores ajustados", y = "Residuos") +
          theme_minimal()

        gridExtra::grid.arrange(p1, p2, ncol = 2)
      }
    })
  })
}
```

```
shinyApp(ui, server)
```

2.2. This R Shiny code creates an interactive web application for statistical analysis of qualitative data. It uses libraries like shiny, shinydashboard, DT, and ggplot2 to design a user-friendly interface with tabs for loading data, descriptive statistics, inferential analysis, and non-parametric tests. Users can upload a CSV file and choose variables for different analyses. The app shows tables, graphs, and statistical test results like Chi-square, Fisher's test, Cramér's V, and more. It helps users understand and compare qualitative data easily using visual tools and summaries. You can find it at the following URL https://cuervo.shinyapps.io/datos_cualitativos/

```
library(shiny)
library(shinydashboard)
library(DT)
library(ggplot2)
library(vcd)
library(coin)
library(DescTools)
library(MASS)

ui <- dashboardPage(
  dashboardHeader(title = "Analisis Estadistico para datos cualitativos"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Cargar Datos", tabName = "datos", icon = icon("database")),
      menuItem("Estadística Descriptiva", tabName = "descriptiva", icon = icon("chart-bar")),
      menuItem("Estadística Inferencial", tabName = "inferencial", icon = icon("project-diagr
      menuItem("Pruebas No Paramétricas", tabName = "no_param", icon = icon("chart-line"))
    )
  ),
  dashboardBody(
    tabItems(
      tabItem(tabName = "datos",
              fluidRow(
                box(title = "Cargar Datos", width = 12,
                    fileInput("file1", "Seleccione archivo CSV",
                              accept = c("text/csv",
                                         "text/comma-separated-values,text/plain",
                                         ".csv")),
                    checkboxInput("header", "Encabezado", TRUE),
                    radioButtons("sep", "Separador",
```

```
                                choices = c(Coma = ",",
                                            PuntoComa = ";",
                                            Tabulador = "\t"),
                                selected = ","),
                    radioButtons("quote", "Comillas",
                                choices = c(Ninguna = "",
                                            "Doble" = '"',
                                            "Simple" = "'"),
                                selected = '"')
                  ),
                  box(title = "Vista previa", width = 12,
                      DTOutput("contents")
                  )
                )
        ),

        tabItem(tabName = "descriptiva",
                fluidRow(
                  box(title = "Opciones Descriptivas", width = 4,
                      selectInput("var_cuali", "Seleccione variable cualitativa:", choices = NU
                      selectInput("var_cuali2", "Seleccione segunda variable (opcional):", choi
                      radioButtons("tipo_grafico", "Tipo de gráfico:",
                                choices = c("Barras" = "barras",
                                            "Sectores" = "pastel",
                                            "Tabla" = "tabla"),
                                selected = "barras")
                  ),
                  box(title = "Resultados Descriptivos", width = 8,
                      verbatimTextOutput("resumen_descriptivo"),
                      plotOutput("grafico_descriptivo"),
                      DTOutput("tabla_frecuencias")
                  )
                )
        ),

        tabItem(tabName = "inferencial",
                fluidRow(
                  box(title = "Opciones Inferenciales", width = 4,
                      selectInput("var_inf1", "Variable 1:", choices = NULL),
                      selectInput("var_inf2", "Variable 2:", choices = NULL),
                      radioButtons("prueba_inf", "Prueba estadística:",
                                choices = c("Chi-cuadrado" = "chisq",
```

```
                                            "Fisher" = "fisher",
                                            "V de Cramer" = "cramer",
                                            "Coef. Phi" = "phi",
                                            "Lambda" = "lambda",
                                            "Coef. Contingencia" = "cont")),
                        actionButton("calcular_inf", "Calcular")
                    ),
                    box(title = "Resultados Inferenciales", width = 8,
                        verbatimTextOutput("resultado_inferencial")
                    )
                )
        ),

        tabItem(tabName = "no_param",
                fluidRow(
                    box(title = "Opciones No Paramétricas", width = 4,
                        selectInput("var_resp", "Variable respuesta:", choices = NULL),
                        selectInput("var_grupo", "Variable de agrupación:", choices = NULL),
                        radioButtons("prueba_np", "Prueba no paramétrica:",
                                     choices = c("Mann-Whitney" = "mw",
                                            "Kruskal-Wallis" = "kw",
                                            "Wilcoxon" = "wx",
                                            "Friedman" = "fried",
                                            "Spearman" = "spear",
                                            "Kolmogorov-Smirnov" = "ks")),
                        actionButton("calcular_np", "Calcular")
                    ),
                    box(title = "Resultados No Paramétricos", width = 8,
                        verbatimTextOutput("resultado_noparam")
                    )
                )
        )
    )
  )
)

server <- function(input, output, session) {

  datos <- reactive({
    req(input$file1)
    df <- read.csv(input$file1$datapath,
                   header = input$header,
```

```
                  sep = input$sep,
                  quote = input$quote)
  updateSelectInput(session, "var_cuali", choices = names(df))
  updateSelectInput(session, "var_cuali2", choices = c("Ninguna", names(df)))
  updateSelectInput(session, "var_inf1", choices = names(df))
  updateSelectInput(session, "var_inf2", choices = names(df))
  updateSelectInput(session, "var_resp", choices = names(df))
  updateSelectInput(session, "var_grupo", choices = names(df))
  return(df)
})

output$contents <- renderDT({
  datatable(datos(), options = list(scrollX = TRUE))
})

output$resumen_descriptivo <- renderPrint({
  req(input$var_cuali)
  var <- datos()[[input$var_cuali]]

  cat("=== FRECUENCIAS ABSOLUTAS ===\n")
  print(table(var))

  cat("\n=== FRECUENCIAS RELATIVAS ===\n")
  print(prop.table(table(var)))

  cat("\n=== MEDIDAS RESUMEN ===\n")
  moda <- names(sort(table(var), decreasing = TRUE))[1]
  cat("Moda:", moda, "\n")

  if(!is.null(input$var_cuali2) && input$var_cuali2 != "Ninguna") {
    var2 <- datos()[[input$var_cuali2]]
    cat("\n=== TABLA DE CONTINGENCIA ===\n")
    print(table(var, var2))

    cat("\n=== RAZÓN ENTRE CATEGORÍAS ===\n")
    tab <- table(var, var2)
    if(nrow(tab) >= 2 && ncol(tab) >= 2) {
      razon <- tab[1,1] / tab[2,2]
      cat("Razón entre [1,1] y [2,2]:", razon, "\n")
    }
  }
})
```

```
output$grafico_descriptivo <- renderPlot({
  req(input$var_cuali)
  var <- datos()[[input$var_cuali]]
  df <- as.data.frame(table(var))
  names(df) <- c("Categoria", "Frecuencia")

  if(input$tipo_grafico == "barras") {
    ggplot(df, aes(x = Categoria, y = Frecuencia, fill = Categoria)) +
      geom_bar(stat = "identity") +
      labs(title = "Gráfico de Barras", x = input$var_cuali, y = "Frecuencia") +
      theme_minimal()
  } else if(input$tipo_grafico == "pastel") {
    ggplot(df, aes(x = "", y = Frecuencia, fill = Categoria)) +
      geom_bar(stat = "identity", width = 1) +
      coord_polar("y", start = 0) +
      labs(title = "Gráfico de Sectores", fill = input$var_cuali) +
      theme_void()
  }
})

output$tabla_frecuencias <- renderDT({
  req(input$var_cuali)
  var <- datos()[[input$var_cuali]]
  df <- as.data.frame(table(var))
  df$Relativa <- prop.table(df$Freq)
  names(df) <- c("Categoría", "Frecuencia Absoluta", "Frecuencia Relativa")
  datatable(df)
})

observeEvent(input$calcular_inf, {
  output$resultado_inferencial <- renderPrint({
    req(input$var_inf1, input$var_inf2)
    var1 <- datos()[[input$var_inf1]]
    var2 <- datos()[[input$var_inf2]]
    tab <- table(var1, var2)

    cat("=== TABLA DE CONTINGENCIA ===\n")
    print(tab)

    if(input$prueba_inf == "chisq") {
      cat("\n=== PRUEBA CHI-CUADRADO ===\n")
      if(length(dim(tab)) == 1) {
```

```
        prueba <- chisq.test(tab)
        print(prueba)
      } else {
        prueba <- chisq.test(tab)
        print(prueba)
      }
    } else if(input$prueba_inf == "fisher") {
      cat("\n=== PRUEBA EXACTA DE FISHER ===\n")
      prueba <- fisher.test(tab)
      print(prueba)
    } else if(input$prueba_inf == "cramer") {
      cat("\n=== V DE CRAMER ===\n")
      cramer <- sqrt(chisq.test(tab)$statistic / (sum(tab) * (min(dim(tab)) - 1)))
      names(cramer) <- "V de Cramer"
      print(cramer)
    } else if(input$prueba_inf == "phi") {
      cat("\n=== COEFICIENTE PHI ===\n")
      phi <- sqrt(chisq.test(tab)$statistic / sum(tab))
      names(phi) <- "Coeficiente Phi"
      print(phi)
    } else if(input$prueba_inf == "lambda") {
      cat("\n=== LAMBDA DE GOODMAN Y KRUSKAL ===\n")
      lambda <- Lambda(tab)
      print(lambda)
    } else if(input$prueba_inf == "cont") {
      cat("\n=== COEFICIENTE DE CONTINGENCIA ===\n")
      cont <- sqrt(chisq.test(tab)$statistic / (chisq.test(tab)$statistic + sum(tab)))
      names(cont) <- "Coeficiente de Contingencia"
      print(cont)
    }
  })
})

observeEvent(input$calcular_np, {
  output$resultado_noparam <- renderPrint({
    req(input$var_resp, input$var_grupo)
    resp <- datos()[[input$var_resp]]
    grupo <- datos()[[input$var_grupo]]

    if(input$prueba_np == "mw") {
      cat("=== PRUEBA DE MANN-WHITNEY ===\n")
      prueba <- wilcox.test(resp ~ grupo, exact = FALSE)
```

```
        print(prueba)
      } else if(input$prueba_np == "kw") {
        cat("=== PRUEBA DE KRUSKAL-WALLIS ===\n")
        prueba <- kruskal.test(resp ~ grupo)
        print(prueba)
      } else if(input$prueba_np == "wx") {
        cat("=== PRUEBA DE WILCOXON (PARA DATOS APAREADOS) ===\n")
        prueba <- wilcox.test(resp, grupo, paired = TRUE, exact = FALSE)
        print(prueba)
      } else if(input$prueba_np == "fried") {
        cat("=== PRUEBA DE FRIEDMAN ===\n")
        prueba <- friedman.test(resp ~ grupo | rep(1:length(resp)))
        print(prueba)
      } else if(input$prueba_np == "spear") {
        cat("=== COEFICIENTE DE CORRELACIÓN DE SPEARMAN ===\n")
        prueba <- cor.test(resp, as.numeric(grupo), method = "spearman")
        print(prueba)
      } else if(input$prueba_np == "ks") {
        cat("=== PRUEBA DE KOLMOGOROV-SMIRNOV ===\n")
        grupo1 <- resp[grupo == levels(grupo)[1]]
        grupo2 <- resp[grupo == levels(grupo)[2]]
        prueba <- ks.test(grupo1, grupo2)
        print(prueba)
      }
    })
  })
}

shinyApp(ui, server)
```

You can find it in the following git repository: https://github.com/ravencuervo/estadistica_computacional.git