Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática
Docente: Fred Torres Cruz
Autor : edilberto wilson mamani emanuel

Trabajo Encargado -

Mi tareaa

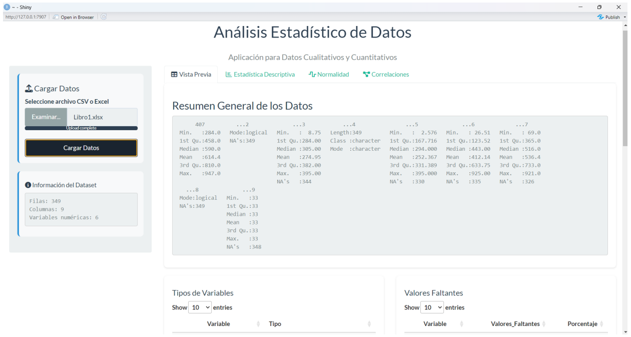
```
Explicación de la Aplicación Shiny:
"Análisis Estadístico de Datos"
Este código es una aplicación interactiva web creada con el framework Shiny de R.

options(shiny.maxRequestSize = 100 * 1024^2) # 100MB max file size

library(shiny)
library(shinythemes)
library(DT)
library(readr)
library(readxl)
library(dplyr)
library(ggplot2)
library(plotly)
library(e1071)
library(nortest)
library(corrplot)
library(shinydashboard)
library(tidyr)

# Interfaz de Usuario
ui <- fluidPage(
  theme = shinytheme("flatly"),

  # Título principal
  titlePanel(
    div(
      h1("Análisis Estadístico de Datos",
         style = "color: #2c3e50; text-align: center; margin-bottom: 30px;"),
      h4("Aplicación para Datos Cualitativos y Cuantitativos",
         style = "color: #7f8c8d; text-align: center; font-weight: 300;")
    )
  ),

  # Layout principal
  sidebarLayout(
    # Panel lateral
    sidebarPanel(
      width = 3,

      # Sección de carga de datos
      div(class = "sidebar-section",
          tags$h4(icon("upload"), "Cargar Datos"),

          fileInput("file", "Seleccione archivo CSV o Excel",
                    accept = c(".csv", ".xlsx", ".xls"),
                    buttonLabel = "Examinar...",
                    placeholder = "Ningún archivo seleccionado"),

          # Opciones para CSV
          conditionalPanel(
            condition = "input.file && input.file.name.endsWith(’.csv’)",
            checkboxInput("header", "¿Tiene encabezado?", TRUE),
            radioButtons("sep", "Separador:",
                         choices = c("Coma" = ",",
                                     "Punto y coma" = ";",
                                     "Tabulador" = "\t"),
                         selected = ","),
            radioButtons("quote", "Comillas:",
                         choices = c("Ninguna" = "",
                                     "Simple" = "’",
                                     "Doble" = ’"’),
                         selected = ’"’)
          ),

          actionButton("load", "Cargar Datos",
                       class = "btn-primary btn-block",
                       style = "margin-top: 15px;")
      ),

      # Información del dataset
      conditionalPanel(
        condition = "output.data_loaded",
        div(class = "sidebar-section",
            tags$h5(icon("info-circle"), "Información del Dataset"),
            verbatimTextOutput("data_info", placeholder = TRUE)
        )
      )
    ),
    # Panel principal
    mainPanel(
      width = 9,

      tabsetPanel(
        id = "main_tabs",

        # Tab 1: Vista Previa de Datos
        tabPanel("Vista Previa",
                 value = "preview",
                 icon = icon("table"),

                 fluidRow(
                   column(12,
                          div(class = "content-section",
                              h3("Resumen General de los Datos"),
                              verbatimTextOutput("summary")
                          )
                   )
                 ),

                 fluidRow(
                   column(6,
                          div(class = "content-section",
                              h4("Tipos de Variables"),
                              DT::dataTableOutput("data_types")
                          )
                   ),
                   column(6,
                          div(class = "content-section",
                              h4("Valores Faltantes"),
                              DT::dataTableOutput("missing_values")
                          )
                   )
                 ),

                 fluidRow(
                   column(12,
                          div(class = "content-section",
                              h4("Vista de Datos (Primeras 50 filas)"),
                              DT::dataTableOutput("preview")
                          )
                   )
                 )
        ),
        # Tab 2: Estadística Descriptiva
        tabPanel("Estadística Descriptiva",
                 value = "descriptive",
                 icon = icon("chart-bar"),

                 sidebarLayout(
                   sidebarPanel(
                     width = 4,

                     selectInput("var_desc", "Seleccione variable:",
                                 choices = NULL),

                     checkboxGroupInput("stats", "Estadísticos a mostrar:",
                                        choices = c("Media" = "mean",
                                                    "Mediana" = "median",
                                                    "Moda" = "mode",
                                                    "Desviación estándar" = "sd",
                                                    "Varianza" = "var",
                                                    "Rango" = "range",
                                                    "Cuartiles" = "quartiles",
                                                    "Asimetría" = "skewness",
                                                    "Curtosis" = "kurtosis"),
                                        selected = c("mean", "median", "sd")),

                     radioButtons("plot_type", "Tipo de gráfico:",
                                  choices = c("Histograma" = "hist",
                                              "Boxplot" = "box",
                                              "Densidad" = "density",
                                              "Q-Q Plot" = "qq"),
                                  selected = "hist"),

                     # Opciones adicionales para gráficos
                     conditionalPanel(
                       condition = "input.plot_type == 'hist'",
                       sliderInput("bins", "Número de bins:",
                                   min = 5, max = 50, value = 20)
                     ),

                     actionButton("calc_desc", "Calcular",
                                  class = "btn-primary btn-block")
                   ),

                   mainPanel(
                     width = 8,
                     fluidRow(
                       column(12,
                              div(class = "content-section",
                                  h4("Resultados Estadísticos"),
                                  verbatimTextOutput("desc_results")
                              )
                       )
                     ),

                     fluidRow(
                       column(12,
                              div(class = "content-section",
                                  h4("Visualización"),
                                  plotlyOutput("desc_plot", height = "400px")
                              )
                       )
                     )
                   )
                 )
        ),

        # Tab 3: Análisis de Normalidad
        tabPanel("Normalidad",
                 value = "normality",
                 icon = icon("wave-square"),

                 sidebarLayout(
                   sidebarPanel(
                     width = 4,

                     selectInput("var_norm", "Variable para análisis:",
                                 choices = NULL),

                     checkboxGroupInput("norm_tests", "Pruebas de normalidad:",
                                        choices = c("Shapiro-Wilk" = "shapiro",
                                                    "Kolmogorov-Smirnov" = "ks",
                                                    "Anderson-Darling" = "ad",
                                                    "Lilliefors" = "lillie"),
                                        selected = c("shapiro", "ks")),

                     numericInput("alpha", "Nivel de significancia:",
                                  value = 0.05, min = 0.01, max = 0.1, step = 0.01),

                     actionButton("calc_norm", "Realizar Pruebas",
                                  class = "btn-primary btn-block")
                   ),

                   mainPanel(
                     width = 8,

                     fluidRow(
                       column(12,
                              div(class = "content-section",
                                  h4("Resultados de Pruebas de Normalidad"),
                                  verbatimTextOutput("norm_results")
                              )
                       )
                     ),

                     fluidRow(
                       column(6,
                              div(class = "content-section",
                                  h5("Q-Q Plot"),
                                  plotlyOutput("qq_plot", height = "300px")
                              )
                       ),
                       column(6,
                              div(class = "content-section",
                                  h5("Histograma con Curva Normal"),
                                  plotlyOutput("norm_hist", height = "300px")
                              )
                       )
                     )
                   )
                 )
        ),

        # Tab 4: Correlaciones (nueva funcionalidad)
        tabPanel("Correlaciones",
                 value = "correlation",
                 icon = icon("project-diagram"),

                 fluidRow(
                   column(4,
                          div(class = "content-section",
                              h4("Configuración"),

                              selectInput("corr_vars", "Seleccionar variables numéricas:
                                          choices = NULL,
                                          multiple = TRUE),

                              radioButtons("corr_method", "Método de correlación:",
                                           choices = c("Pearson" = "pearson",
                                                       "Spearman" = "spearman",
                                                       "Kendall" = "kendall"),
                                           selected = "pearson"),

                              actionButton("calc_corr", "Calcular Correlaciones",
                                           class = "btn-primary btn-block")
                          )
                   ),

                   column(8,
                          div(class = "content-section",
                              h4("Matriz de Correlación"),
                              plotOutput("corr_plot", height = "400px"),

                              h5("Tabla de Correlaciones"),
                              DT::dataTableOutput("corr_table")
                          )
                   )
                 )
        )
      )
    )
  ),

  # CSS personalizado
  tags$head(
    tags$style(HTML("
      .content-section {
        background-color: #ffffff;
        border-radius: 8px;
        padding: 20px;
        margin-bottom: 20px;
        box-shadow: 0 2px 4px rgba(0,0,0,0.1);
      }

      .sidebar-section {
        background-color: #f8f9fa;
        border-radius: 8px;
        padding: 15px;
        margin-bottom: 20px;
        border-left: 4px solid #3498db;
      }

      .btn-primary {
        background-color: #3498db;
        border-color: #3498db;
      }

      .btn-primary:hover {
        background-color: #2980b9;
        border-color: #2980b9;
      }
    "))
  )
)

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

  # Valores reactivos
  rv <- reactiveValues(
    data = NULL,
    data_types = NULL,
    numeric_vars = NULL
  )

  # Indicador de datos cargados
  output$data_loaded <- reactive({
    return(!is.null(rv$data))
  })
  outputOptions(output, "data_loaded", suspendWhenHidden = FALSE)

  # Cargar datos
  observeEvent(input$load, {
    req(input$file)

    tryCatch({
      ext <- tools::file_ext(input$file$name)

      if (ext == "csv") {
        rv$data <- read.csv(input$file$datapath,
                            header = input$header,
                            sep = input$sep,
                            quote = input$quote)
      } else if (ext %in% c("xlsx", "xls")) {
        rv$data <- read_excel(input$file$datapath)
      }

      # Clasificar tipos de variables
      rv$data_types <- sapply(rv$data, function(x) {
        if (is.numeric(x)) {
          unique_vals <- length(unique(x[!is.na(x)]))
          if (unique_vals <= 10) {
            return("Cuantitativa Discreta")
          } else {
            return("Cuantitativa Continua")
          }
        } else {
          unique_vals <- length(unique(x[!is.na(x)]))
          if (unique_vals <= 10) {
            return("Cualitativa Nominal")
          } else {
            return("Cualitativa Ordinal")
          }
        }
      })

      # Variables numéricas
      rv$numeric_vars <- names(rv$data)[sapply(rv$data, is.numeric)]

      # Actualizar opciones de selección
      updateSelectInput(session, "var_desc", choices = names(rv$data))
      updateSelectInput(session, "var_norm", choices = rv$numeric_vars)
      updateSelectInput(session, "corr_vars", choices = rv$numeric_vars)

      showNotification("Datos cargados correctamente", type = "success")

    }, error = function(e) {
      showNotification(paste("Error al cargar datos:", e$message), type = "error")
    })
  })

  # Información del dataset
  output$data_info <- renderText({
    req(rv$data)
    paste(
      paste("Filas:", nrow(rv$data)),
      paste("Columnas:", ncol(rv$data)),
      paste("Variables numéricas:", length(rv$numeric_vars)),
      sep = "\n"
    )
```

```
})

# Resumen de datos
output$summary <- renderPrint({
  req(rv$data)
  summary(rv$data)
})

# Tipos de variables
output$data_types <- renderDataTable({
  req(rv$data_types)
  data.frame(
    Variable = names(rv$data_types),
    Tipo = rv$data_types,
    stringsAsFactors = FALSE
  )
}, options = list(pageLength = 10, searching = FALSE))

# Valores faltantes
output$missing_values <- renderDataTable({
  req(rv$data)
  missing_data <- rv$data %>%
    summarise_all(~sum(is.na(.))) %>%
    gather(key = "Variable", value = "Valores_Faltantes") %>%
    mutate(Porcentaje = round(Valores_Faltantes / nrow(rv$data) * 100, 2)) %>%
    arrange(desc(Valores_Faltantes))

  missing_data
}, options = list(pageLength = 10, searching = FALSE))

# Vista previa de datos
output$preview <- renderDataTable({
  req(rv$data)
  head(rv$data, 50)
}, options = list(scrollX = TRUE, pageLength = 15))

# Estadística descriptiva
output$desc_results <- renderPrint({
  req(rv$data, input$var_desc, input$calc_desc)

  var_data <- rv$data[[input$var_desc]]
  var_data <- var_data[!is.na(var_data)]

  results <- list()
```

```
    if ("mean" %in% input$stats && is.numeric(var_data)) {
      results$Media <- round(mean(var_data), 4)
    }
    if ("median" %in% input$stats && is.numeric(var_data)) {
      results$Mediana <- round(median(var_data), 4)
    }
    if ("mode" %in% input$stats) {
      mode_val <- names(sort(table(var_data), decreasing = TRUE))[1]
      results$Moda <- mode_val
    }
    if ("sd" %in% input$stats && is.numeric(var_data)) {
      results$‘Desviación Estándar‘ <- round(sd(var_data), 4)
    }
    if ("var" %in% input$stats && is.numeric(var_data)) {
      results$Varianza <- round(var(var_data), 4)
    }
    if ("range" %in% input$stats && is.numeric(var_data)) {
      results$Rango <- paste(round(range(var_data), 4), collapse = " - ")
    }
    if ("quartiles" %in% input$stats && is.numeric(var_data)) {
      q <- quantile(var_data, probs = c(0.25, 0.5, 0.75))
      results$‘Q1, Q2, Q3‘ <- paste(round(q, 4), collapse = ", ")
    }
    if ("skewness" %in% input$stats && is.numeric(var_data)) {
      results$Asimetría <- round(e1071::skewness(var_data), 4)
    }
    if ("kurtosis" %in% input$stats && is.numeric(var_data)) {
      results$Curtosis <- round(e1071::kurtosis(var_data), 4)
    }

    results
  })

  # Gráfico descriptivo
  output$desc_plot <- renderPlotly({
    req(rv$data, input$var_desc, input$calc_desc)

    var_data <- rv$data[[input$var_desc]]
    var_name <- input$var_desc

    p <- switch(input$plot_type,
                "hist" = {
                  if (is.numeric(var_data)) {
                    ggplot(data.frame(x = var_data), aes(x = x)) +
                      geom_histogram(bins = input$bins, fill = "#3498db", alpha = 0.7, c
                      labs(title = paste("Histograma de", var_name), x = var_name, y = "
                      theme_minimal()
                  } else {
                    ggplot(data.frame(x = var_data), aes(x = x)) +
                      geom_bar(fill = "#3498db", alpha = 0.7) +
                      labs(title = paste("Gráfico de barras de", var_name), x = var_name
                      theme_minimal() +
                      theme(axis.text.x = element_text(angle = 45, hjust = 1))
                  }
                },
                "box" = {
                  if (is.numeric(var_data)) {
                    ggplot(data.frame(x = var_data), aes(y = x)) +
                      geom_boxplot(fill = "#e74c3c", alpha = 0.7) +
                      labs(title = paste("Boxplot de", var_name), y = var_name) +
                      theme_minimal()
                  } else {
                    ggplot(data.frame(x = "Variable", y = 1), aes(x = x, y = y)) +
                      geom_text(aes(label = "No aplicable para variables categóricas"),
                      theme_void()
                  }
                },
                "density" = {
                  if (is.numeric(var_data)) {
                    ggplot(data.frame(x = var_data), aes(x = x)) +
                      geom_density(fill = "#2ecc71", alpha = 0.7) +
                      labs(title = paste("Densidad de", var_name), x = var_name, y = "De
                      theme_minimal()
                  } else {
                    ggplot(data.frame(x = "Variable", y = 1), aes(x = x, y = y)) +
                      geom_text(aes(label = "No aplicable para variables categóricas"),
                      theme_void()
                  }
                },
                "qq" = {
                  if (is.numeric(var_data)) {
                    ggplot(data.frame(sample = var_data), aes(sample = sample)) +
                      stat_qq() + stat_qq_line(color = "red") +
                      labs(title = paste("Q-Q Plot de", var_name)) +
                      theme_minimal()
                  } else {
                    ggplot(data.frame(x = "Variable", y = 1), aes(x = x, y = y)) +
                      geom_text(aes(label = "No aplicable para variables categóricas"),
                      theme_void()
                  }
```

```
              }
  )

  ggplotly(p)
})

# Pruebas de normalidad
output$norm_results <- renderPrint({
  req(rv$data, input$var_norm, input$calc_norm)

  var_data <- na.omit(rv$data[[input$var_norm]])

  results <- list()

  for (test in input$norm_tests) {
    test_result <- switch(test,
                          "shapiro" = {
                            if (length(var_data) <= 5000) {
                              shapiro.test(var_data)
                            } else {
                              list(method = "Shapiro-Wilk",
                                   p.value = NA,
                                   statistic = NA,
                                   note = "Muestra muy grande (>5000). Use otras prueb
                            }
                          },
                          "ks" = ks.test(var_data, "pnorm", mean(var_data), sd(var_dat
                          "ad" = nortest::ad.test(var_data),
                          "lillie" = nortest::lillie.test(var_data)
    )

    results[[test]] <- test_result
  }

  # Interpretación
  cat("INTERPRETACIÓN DE RESULTADOS:\n")
  cat("H0: Los datos siguen una distribución normal\n")
  cat("H1: Los datos NO siguen una distribución normal\n")
  cat(paste("Nivel de significancia:", input$alpha, "\n\n"))

  for (i in seq_along(results)) {
    test_name <- names(results)[i]
    test_result <- results[[i]]

    cat(paste("===", toupper(test_name), "===\n"))
```

```
      if (!is.na(test_result$p.value)) {
        cat(paste("Estadístico:", round(test_result$statistic, 6), "\n"))
        cat(paste("p-valor:", format(test_result$p.value, scientific = TRUE), "\n"))

        if (test_result$p.value < input$alpha) {
          cat("CONCLUSIÓN: Se RECHAZA H0. Los datos NO siguen distribución normal.\n")
        } else {
          cat("CONCLUSIÓN: NO se rechaza H0. Los datos podrían seguir distribución norma
        }
      } else if (!is.null(test_result$note)) {
        cat(test_result$note, "\n")
      }
      cat("\n")
    }

    results
  })

  # Q-Q Plot para normalidad
  output$qq_plot <- renderPlotly({
    req(rv$data, input$var_norm, input$calc_norm)

    var_data <- na.omit(rv$data[[input$var_norm]])

    p <- ggplot(data.frame(sample = var_data), aes(sample = sample)) +
      stat_qq(color = "#3498db") +
      stat_qq_line(color = "red", linewidth = 1) +
      labs(title = "Q-Q Plot Normal",
           x = "Cuantiles teóricos",
           y = "Cuantiles observados") +
      theme_minimal()

    ggplotly(p)
  })

  # Histograma con curva normal
  output$norm_hist <- renderPlotly({
    req(rv$data, input$var_norm, input$calc_norm)

    var_data <- na.omit(rv$data[[input$var_norm]])

    p <- ggplot(data.frame(x = var_data), aes(x = x)) +
      geom_histogram(aes(y = after_stat(density)), bins = 30,
                     fill = "#3498db", alpha = 0.7, color = "white") +
      stat_function(fun = dnorm,
                    args = list(mean = mean(var_data), sd = sd(var_data)),
                    color = "red", linewidth = 1) +
      labs(title = "Histograma vs Distribución Normal",
           x = input$var_norm, y = "Densidad") +
      theme_minimal()

    ggplotly(p)
  })

  # Análisis de correlación
  output$corr_plot <- renderPlot({
    req(rv$data, input$corr_vars, input$calc_corr)
    req(length(input$corr_vars) >= 2)

    corr_data <- rv$data[input$corr_vars]
    corr_matrix <- cor(corr_data, use = "complete.obs", method = input$corr_method)

    corrplot(corr_matrix, method = "color", type = "upper",
             order = "hclust", tl.cex = 0.8, tl.col = "black",
             addCoef.col = "black", number.cex = 0.7)
  })

  # Tabla de correlaciones
  output$corr_table <- renderDataTable({
    req(rv$data, input$corr_vars, input$calc_corr)
    req(length(input$corr_vars) >= 2)

    corr_data <- rv$data[input$corr_vars]
    corr_matrix <- cor(corr_data, use = "complete.obs", method = input$corr_method)

    # Convertir matriz a formato largo
    corr_df <- expand.grid(Var1 = rownames(corr_matrix), Var2 = colnames(corr_matrix))
    corr_df$Correlacion <- as.vector(corr_matrix)
    corr_df <- corr_df[corr_df$Var1 != corr_df$Var2, ]
    corr_df$Correlacion <- round(corr_df$Correlacion, 4)

    corr_df
  }, options = list(pageLength = 10))
}

# Ejecutar aplicación
shinyApp(ui = ui, server = server)
```



Análisis Estadístico de Datos

Aplicación para Datos Cualitativos y Cuantitativos

Cargar Datos

Seleccione archivo CSV o Excel: Libro1.xlsx

Información del Dataset

```
Filas: 349
Columnas: 9
Variables numéricas: 6
```

Resumen General de los Datos

```
     407          ...2             ...3              ...4                 ...5              ...6              ...7
 Min.   :284.0   Mode:logical   Min.   :  8.75   Length:349         Min.   :  2.576   Min.   : 26.51   Min.   : 69.0
 1st Qu.:458.0   NA's:349       1st Qu.:284.00   Class :character   1st Qu.:167.716   1st Qu.:123.52   1st Qu.:365.0
 Median :590.0                  Median :305.00   Mode  :character   Median :294.000   Median :443.00   Median :516.0
 Mean   :614.4                  Mean   :274.95                      Mean   :252.367   Mean   :412.14   Mean   :536.4
 3rd Qu.:810.0                  3rd Qu.:382.00                      3rd Qu.:331.389   3rd Qu.:633.75   3rd Qu.:733.0
 Max.   :947.0                  Max.   :395.00                      Max.   :395.000   Max.   :925.00   Max.   :921.0
                                NA's   :344                         NA's   :330       NA's   :335      NA's   :326
   ...8              ...9
 Mode:logical   Min.   :33
 NA's:349       1st Qu.:33
                Median :33
                Mean   :33
                3rd Qu.:33
                Max.   :33
                NA's   :348
```

Tipos de Variables

Variable	Tipo

Valores Faltantes

Variable	Valores_Faltantes	Porcentaje

Figura 1: Enter Caption