library(shiny);library(ggplot2);library(readxl);

library(dplyr);library(labeling)

library(DT);library(rpivotTable);library(readr)

library(stringr);library(tools)

ClustereData <- read\_excel("ClustereData.xlsx")

#ClustedMean <- read\_excel("ClustedMean.xlsx")

clustedmean1row <-read\_excel("df.xlsx")

na.omit(ClustereData)

attach(ClustereData)

str(ClustereData)

#ClustereData$c\_I1I2Indice <- as.factor(c\_I1I2Indice)

dfpivot <- ClustereData[,c("DES\_REGIONE\_ABI\_GEST","c\_I1I2Indice","DES\_PROVINCIA\_ABI\_GEST","REGIONE" )]

ui <- fluidPage(

#theme=shinytheme("slate"), ##

# themeSelector(),

titlePanel(" Benvenuto",windowTitle = "unicredit.app" ),

tags$h3("Welcome to the Interactive User Platform"),

tags$br(),

tags$a("Copy Rights belong to Unicredit.it", href="https://www.unicredit.it/it/privati.html"),

#tags$a("Contact the App Builder", href="http://www00.unibg.it/struttura/strutturasmst.asp?id\_notizia=81930"),

#tags$a(" Retrive Your Customize DataTable", href="https://seymakalay87.shinyapps.io/app5/"),

# navbarPage( title = "",

# id="nav",

sidebarLayout(

# Inputs

sidebarPanel( width = 3,

# Select variable for y-axis

selectInput(inputId = "y",

label = "Y-axis:",

choices = c("index.oninvestment", "index.on.prodection", "INDICE"),

selected = "INDICE"),

# Select variable for x-axis

selectInput(inputId = "x",

label = "X-axis:",

choices = names(ClustereData[,-c(1:6,18:21,44:46)]),

selected = "RISERVA\_MATEMATICA\_201711"),

# Select variable for color

selectInput(inputId = "z",

label = "Color by:",

choices = c("DES\_REGIONE\_ABI\_GEST" = "DES\_REGIONE\_ABI\_GEST",

"DES\_PROVINCIA\_ABI\_GEST" = "DES\_PROVINCIA\_ABI\_GEST",

#"FLAG\_RECESSO\_2017" = "FLAG\_RECESSO\_2017",

"c\_I1I2Indice" = "c\_I1I2Indice"),

selected = "c\_I1I2Indice"),

# Set alpha level

sliderInput(inputId = "alpha",

label = "Alpha:",

min = 0, max = 1,

value = 0.3),

# Select filetype

radioButtons(inputId = "filetype",

label = "Select filetype:",

choices = c("csv", "tsv"),

selected = "csv"),

# Select variables to download

checkboxGroupInput(inputId = "selected\_var",

label = "Select variables:",

choices = names(ClustereData),

selected = c(2:6,18,43:46)),

# Built with Shiny by RStudio

br(), br(),

tags$a("Contact with the App Builder", href="http://www00.unibg.it/struttura/strutturasmst.asp?id\_notizia=81930"),

h6("Built with ",

#img(src = "https://www.rstudio.com/wp-content/uploads/2014/04/shiny.png", height = "30px"),

#"by",

img(src = "https://www.rstudio.com/wp-content/uploads/2014/07/RStudio-Logo-Blue-Gray.png", height = "30px"),

".")

),

# Outputs

mainPanel( width = 9,

tabsetPanel(type = "tabs",

# Tab 1: Plot

tabPanel(title = "Plot",

br(),h4("Visualize Custom Plot"),br(),

plotOutput(outputId = "scatterplot", brush = "plot\_brush"),

textOutput(outputId = "correlation"),

br(),

h4("Visualize the Selected Points "), br(),

DT::dataTableOutput(outputId = "moviestable"),

# HTML("Select Points on the Graph, then hit 'Download data'."),

# br(), br(), # line break and some visual separation

# downloadButton("download\_data1 ", "Download data"),

#h2("Densityplot Based on Custom Plot Above"),

#plotOutput(outputId = "densityplot", height = 200),

h4("Histogram of Xs"),

plotOutput(outputId = "histplot.x", height = 150),

h4("Histogram of Ys"),

plotOutput(outputId = "histplot.y", height = 150)

),

# Tab 2: Data

tabPanel(title = "Summary",

br(),

h4("Mean Values Based on color\_by option"),br(),

h5("Higher the Deviation From the Mean of the Values Dangerous the Cluster is"),

h5("In This Case Cluster 1 and 4 Seems the Most Dangerous Clusters -> INSPECT"),

#DT::dataTableOutput("mytable"),

tableOutput(outputId="mytable1" ),

#bunu ekledim

tableOutput(outputId="df.print")

),

# Tab 2: Data

tabPanel(title = "Table",

br(),

h4("Interactive Table"), br(),

h5("Recomendend: Table with Sub Total Col Heat map (or Bar Chart) and Count"),

h5("Recomendend: Changing the Place of DES\_Regione, DES\_Proviancia, Regione and c\_I1I2Indice"),

rpivotTableOutput("pivot")),

# Tab 3: Data

tabPanel(title = "Data",

br(),br(),

#DT::dataTableOutput( "alldataset"),

DT::dataTableOutput( "alldataset"),

HTML("Select filetype and variables, then hit 'Download data'."),

br(), br(), # line break and some visual separation

#remove Download button # downloadButton("download\_data", "Download data"),

br(),

tags$a(" Visualize Your Data Table", href="https://seymakalay87.shinyapps.io/app5/")

# DT::dataTableOutput(data= ClustereData %>% select(c(2:6),19:21,43:46),

# options = list(pageLength = 50)

#DT::dataTableOutput(outputId = "moviestable")

)

)

)

)

)

#) this is for shinytheme

server <-

function(input, output, session) {

# Create scatterplot object the plotOutput function is expecting

output$scatterplot <- renderPlot({

ggplot(data = ClustereData, aes\_string(x = input$x, y = input$y,color=input$z,size=input$z)) +

geom\_point(alpha = input$alpha)

})

# Create densityplot

# output$densityplot <- renderPlot({

# ggplot(data = ClustereData, aes\_string(x = input$xx)) +

# geom\_density()

#})

# fun\_args.x <- list(mean = mean(ClustereData$x), sd = sd(ClustereData$x))

output$histplot.x <- renderPlot({

ggplot(data = ClustereData, aes\_string(x = input$x)) +

geom\_histogram(aes(y = ..density..),bins = 100,col="darkgreen",fill="darkgreen")+

geom\_density(alpha=.2, fill="#FF6666",col = "red")

# stat\_function(fun = dnorm ,args = fun\_args.x, col = "blue")

})

# fun\_args.y <- list(mean = mean(ClustereData$y), sd = sd(ClustereData$y))

output$histplot.y <- renderPlot({

ggplot(data = ClustereData, aes\_string(x= input$y)) +

geom\_histogram(aes(y = ..density..),bins = 100,col="darkgreen",fill="darkgreen")+

geom\_density(col = "red",alpha=.2, fill="#FF6666")

# stat\_function(fun = dnorm,args = fun\_args.y, col = "blue")

})

output$df.print <- renderTable ({

ClustereData %>%

#group\_by("c\_I1I2Indice")%>%

summarise(mean.indice =round( mean(INDICE), digits = 5),

mean.investment =round( mean(index.oninvestment),digits = 5),

mean.protection = round( mean(index.on.prodection),digits = 5))

})

output$mytable1 <- renderTable ({

ClustereData %>%

group\_by\_(input$z) %>%

summarize(mean.Indice = round( mean(INDICE), digits = 5),

mean.Index.on.Investment=round( mean(index.oninvestment), digits = 5),

mean.Index.on.Protection = round( mean(index.on.prodection), digits = 5)) %>%

arrange(desc(mean.Indice))

})

output$pivot <- renderRpivotTable({

rpivotTable(data = dfpivot ,

rows = c("DES\_REGIONE\_ABI\_GEST") , #,"DES\_PROVINCIA\_ABI\_GEST" for this update dfpivot

cols="c\_I1I2Indice",vals = "Freq", aggregatorName = "Count",

rendererName = "Table", subtotals = TRUE)#width="200%", height="600px")

})

output$mytable = DT::renderDataTable({

DT::datatable(data= ClustedMean, options = list(pageLength = 11))

})

# Create data table

output$alldataset <- DT::renderDataTable({

DT::datatable(data = ClustereData %>% select(1:46),

options = list(pageLength = 20),

rownames = FALSE)

})

# Create text output stating the correlation between the two ploted

output$correlation <- renderText({

r <- round(cor(ClustereData[, input$x], ClustereData[, input$y], use = "pairwise"), 3)

paste0("Correlation = ", r, ". Note: If the relationship between the two variables is not linear,

the correlation coefficient will not be meaningful.")

})

# Print data table

output$moviestable <- DT::renderDataTable({

brushedPoints(ClustereData,input$plot\_brush)%>%

# nearPoints(ClustereData, coordinfo = input$plot\_hover) %>%

select(COD\_CAB\_GEST, REGIONE ,DES\_REGIONE\_ABI\_GEST,DES\_PROVINCIA\_ABI\_GEST, c\_I1I2Indice)

})

# Download file

output$download\_data <- downloadHandler(

filename = function() {

paste0("ClustereData.", input$filetype)

},

content = function(file) {

if(input$filetype == "csv"){

write\_csv(ClustereData %>% select(input$selected\_var), path = file)

}

if(input$filetype == "tsv"){

write\_tsv(ClustereData %>% select(input$selected\_var), path = file)

}

}

)

# Download file Selected Graf

output$download\_data1 <- downloadHandler(

filename = function() {

paste0("selecteddata.", input$filetype)

},

content = function(file) {

if(input$filetype == "csv"){

write\_csv(moviestable %>% select(input$plot\_brush) , path = file)

}

if(input$filetype == "tsv"){

write\_tsv(moviestable %>% select(input$plot\_brush), path = file)

}

}

)

}

shinyApp(ui, server)