**AIR POLLUTION ANALYSIS**

**APPLICATION CODE:**

UI.R

#load the library

library(shiny)

library(RSQLite)

library(ggplot2)

library(shinythemes)

#define shiny ui

shinyUI(

fluidPage(

titlePanel(title = h2("AIR POLUTION IN SUMMER AND WINTER!!",align="center")),

sidebarLayout(

################# SideBar Panel ###########################

sidebarPanel(h4("SELECT THE FOLLOWING:"),

helpText("The Following selectInput drop down choices are dynamically populated based on dataset selected by User"), br(),

#select input with list of datasets

selectInput("season","SELECT THE SEASON",choices =

c("summer\_data","winter\_data")),br(),

uiOutput("v"), # v is coming from renderUI in server.r

br(),

radioButtons("plot","SELECT THE TYPE OF GRAPH:",choices = c("Line","Points","Both\_L\_&\_P","Stairs")),

shinythemes::themeSelector()

),

################## Main Panel ##########################

mainPanel(

tabsetPanel(type="tab",

tabPanel("GRAPH",plotOutput("mygraph")),

tabPanel("CALCULATION","SEASON:",verbatimTextOutput("sname") ,br(),

"MIN:",verbatimTextOutput("minm") ,br(),

"MAX:",verbatimTextOutput("maxm"),br(),

"MEAN:",verbatimTextOutput("mean") ,br(),

"VARIANCE:",verbatimTextOutput("variance") ,br(),br(),

h3( "Select the Pollutants :"),br(),

uiOutput("c1"),br(),

uiOutput("c2"),br(),

"correlation coefficient:",br(),

verbatimTextOutput("corr"),br(),br(),

"Significance Level(p<0.001)",br(),

verbatimTextOutput("sig"),br(),br()

),

tabPanel("T-Test Analysis",

uiOutput("t1"),br(),

uiOutput("t2"),br(),

"t value:",br(),

verbatimTextOutput("tvalue"),br(),br(),

"Difference Between Means:",br(),

verbatimTextOutput("dfmean"),br()

),

tabPanel("Query's AND Abstract View",

"Q.WHAT IS CORRELATION COEFFICIENT DOES?",br(),

"ANS:. The positive correlation indicates increase in corresponding concentration of one pollutant with increase in concentration of other pollutant, while negative correlation indicates corresponding decrease in concentration of one pollutant with increase in concentration of the other pollutant. ",br(),br(),

"Q.What is T Test?",br(),

"ANS:t test is used when we wish to compare difference between means or difference between two databases or difference between variance",br(),br(),

h3("Abstract View Of Summer AND Winter Pollutants"),br(),br(),

plotOutput("abview")

)

)))

################ Main Panel End ########################

))

SERVER.R

#define library

library(shiny)

library(ggplot2)

library(dplyr)

library(grid)

library(gridExtra)

shinyServer(

function(input,output)

{

############### Basic Tab #############

dataset= reactive({

switch(input$season,

"summer\_data" = names(summer\_data[,c(2,3,4,5,6)]),

"winter\_data" = names(winter\_data[,c(2,3,4,5,6)])

)

})

output$v = renderUI({

selectInput("type","SELECT THE POLLUTANT",choices = dataset())

})

output$data= renderTable({

attach(get(input$season))

get(input$type)

})

#Graphs

output$mygraph=renderPlot({

attach(get(input$season))

y=get(input$type)

switch(input$plot,

"Line" = switch(input$season,

"summer\_data" = ggplot(database1, aes(x=s\_date,y=y)) + geom\_line(stat='identity', position='dodge')+xlab("Time in Days")+ylab("Pollutants in Ug/m3"),

"winter\_data" = ggplot(database1, aes(x=w\_date,y=y)) + geom\_line(stat='identity', position='dodge')+xlab("Time in Days")+ylab("Pollutants in Ug/m3"),

),

"Points" = switch(input$season,

"summer\_data" = ggplot(database1, aes(x=s\_date,y=y)) + geom\_point()+xlab("Time in Days")+ylab("Pollutants in Ug/m3"),

"winter\_data" = ggplot(database1, aes(x=w\_date,y=y)) + geom\_point()+xlab("Time in Days")+ylab("Pollutants in Ug/m3"),

),

"Both\_L\_&\_P" = switch(input$season,

"summer\_data" = plot(s\_date,get(input$type),type="b",xlab="Time in Days",ylab="Pollutants in Ug/m3") ,

"winter\_data" = plot(w\_date,get(input$type),type="b",xlab="Time in Days",ylab="Pollutants in Ug/m3" ),

),

"Stairs" = switch(input$season,

"summer\_data" = plot(s\_date,get(input$type),type="s",xlab="Time in Days",ylab="Pollutants in Ug/m3" ),

"winter\_data" = plot(w\_date,get(input$type),type="s",xlab="Time in Days",ylab="Pollutants in Ug/m3" ),

) ) })

################### CALCULATE Tab #########################

output$sname=renderText({

switch(input$season,

"summer\_data" = paste("SUMMER"),

"winter\_data" = paste("WINTER")

)})

output$minm =renderText({

y=get(input$type)

min(y) })

output$maxm =renderText({

y=get(input$type)

max(y) })

output$mean =renderText({

y=get(input$type)

mean(y) })

output$variance = renderText({

y=get(input$type)

var(y) })

######## correlation calculation ########

output$c1 = renderUI({

selectInput("cor1","SELECT THE POLLUTANT",choices = dataset())

})

output$c2 = renderUI({

selectInput("cor2","SELECT THE POLLUTANT",choices = dataset())

})

output$corr =renderText({

x=get(input$cor1)

y=get(input$cor2)

cor(x,y)})

######## Significance level((p<0.001) ########

output$sig = renderText({

x=get(input$cor1)

y=get(input$cor2)

cor.test(x,y)$p.value})

################### T Test Analysis Tab #######################

output$t1 = renderUI({

selectInput("test1","SELECT THE POLLUTANT",choices = names(summer\_data[,c(2,3,4,5,6)]))

})

output$t2 = renderUI({

selectInput("test2","SELECT THE POLLUTANT",choices = names(winter\_data[,c(2,3,4,5,6)]))

})

output$tvalue = renderText({

#attach(get(winter\_data))

x=get(input$test1)

y=get(input$test2)

t.test(x,y)$statistic })

output$dfmean= renderText({

x=get(input$test1)

y=get(input$test2)

mean(x)-mean(y) })

# ######## Significance level((p<0.001) ########

# output$sig1 = renderText({

# x=get(input$test1)

# y=get(input$test2)

# p=mean(x)

# q=mean(y)

# cor.test(p,q)$p.value

#

# })

############## Query's And Abstract View Tab #################

#making a multiplot function

multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {

library(grid)

# Make a list from the ... arguments and plotlist

plots <- c(list(...), plotlist)

numPlots = length(plots)

# If layout is NULL, then use 'cols' to determine layout

if (is.null(layout)) {

# Make the panel

# ncol: Number of columns of plots

# nrow: Number of rows needed, calculated from # of cols

layout <- matrix(seq(1, cols \* ceiling(numPlots/cols)),

ncol = cols, nrow = ceiling(numPlots/cols)) }

if (numPlots==1) {

print(plots[[1]])

} else {

# Set up the page

grid.newpage()

pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

# Make each plot, in the correct location

for (i in 1:numPlots) {

# Get the i,j matrix positions of the regions that contain this subplot

matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,

layout.pos.col = matchidx$col))

} } }

p1 = ggplot(smelt, aes(x=s\_date, y=value, fill=variable,colour=variable)) + geom\_line(stat='identity', position='dodge')+xlab("Time in Days")+ylab("Pollutants in Ug/m3")

p2 = ggplot(wmelt, aes(x=w\_date, y=value, fill=variable,colour=variable)) + geom\_line(stat='identity', position='dodge')+xlab("Time in Days")+ylab("Pollutants in Ug/m3")

output$abview = renderPlot({

#par(mfrow = c(2, 1))

#ggplot(smelt, aes(x=s\_date, y=value, fill=variable,colour=variable)) + geom\_line(stat='identity', position='dodge')+xlab("Time in Days")+ylab("Pollutants in Ug/m3")

#ggplot(wmelt, aes(x=w\_date, y=value, fill=variable,colour=variable)) + geom\_line(stat='identity', position='dodge')+xlab("Time in Days")+ylab("Pollutants in Ug/m3")

multiplot(p1,p2,cols = 2)

})

############Summary Tab ##################

# sum = reactive({

# switch(input$season,

# "summer\_data" = summary(summer\_data),

# "winter\_data" = summary(winter\_data)

# )

# })

# output$summary = renderTable(

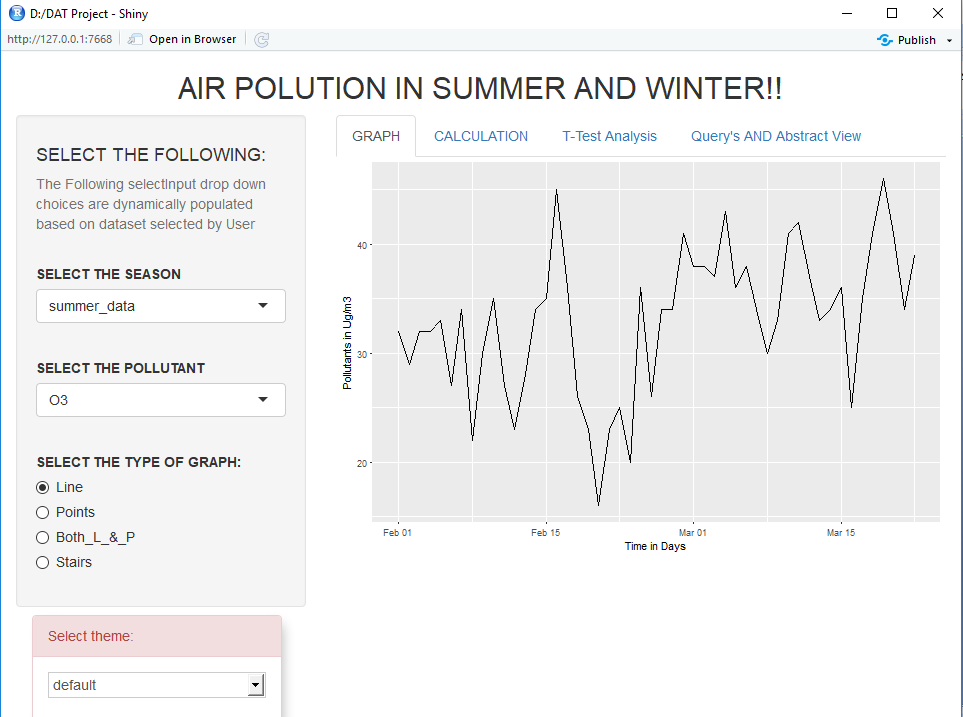
# data.frame(sum)

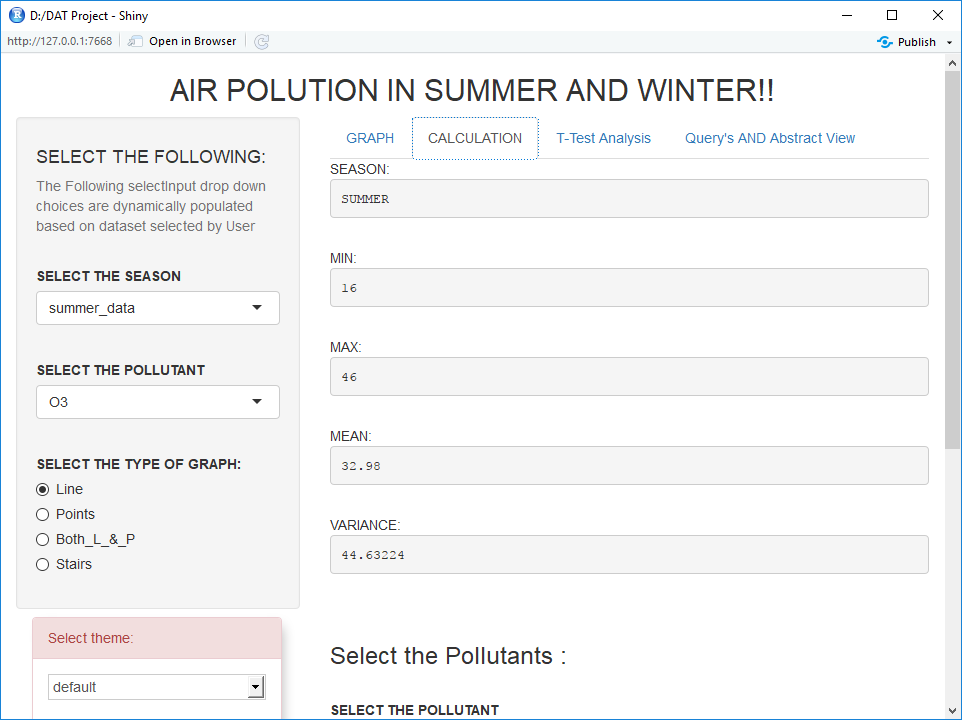
# )

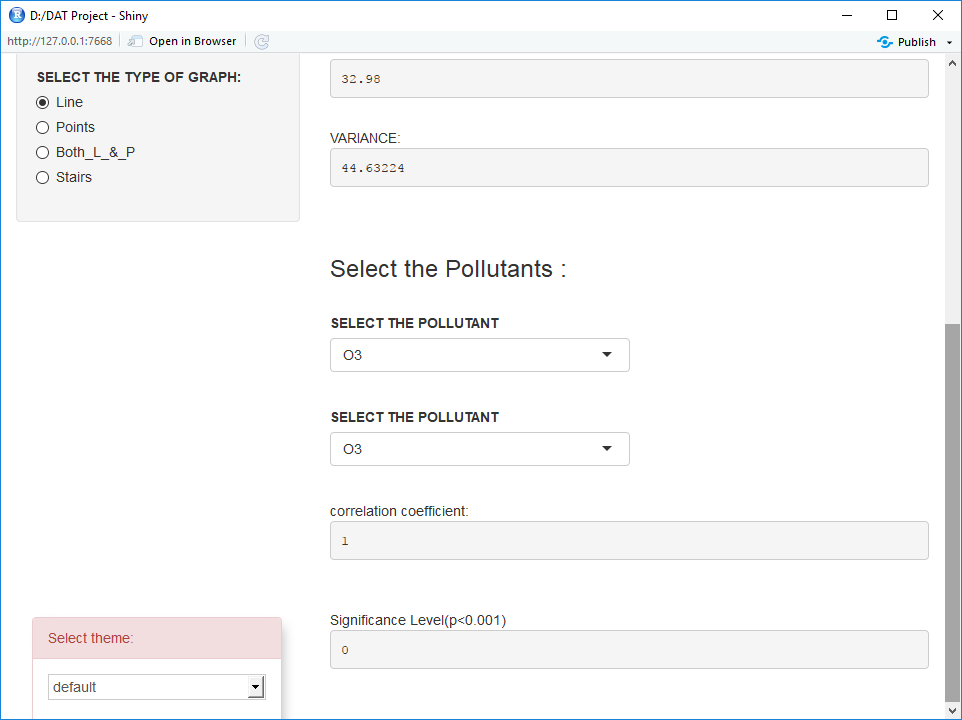
################################################

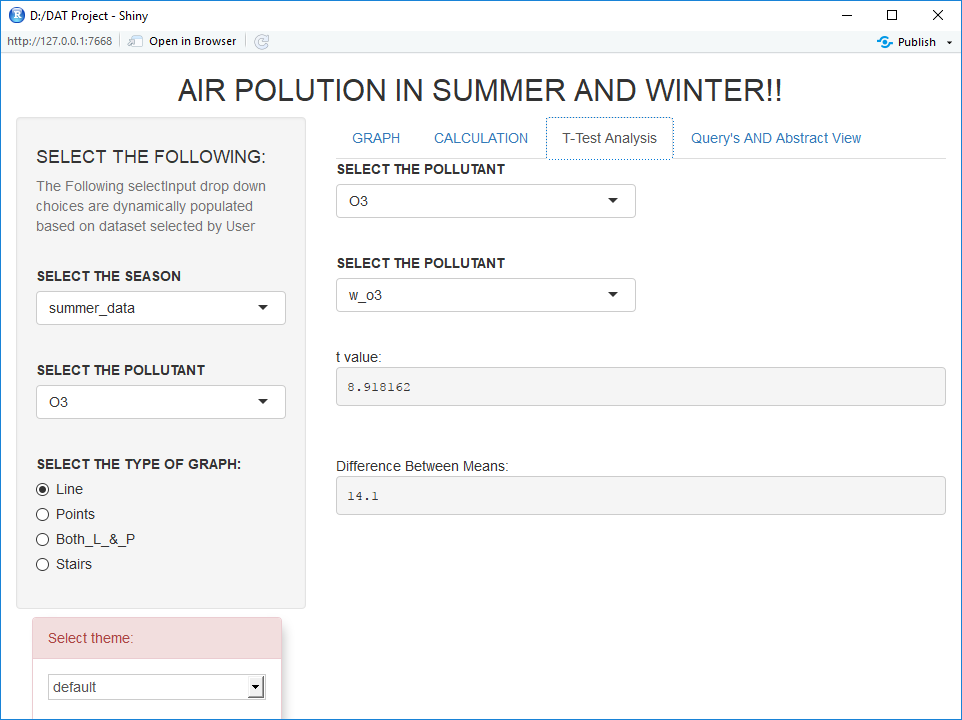
})

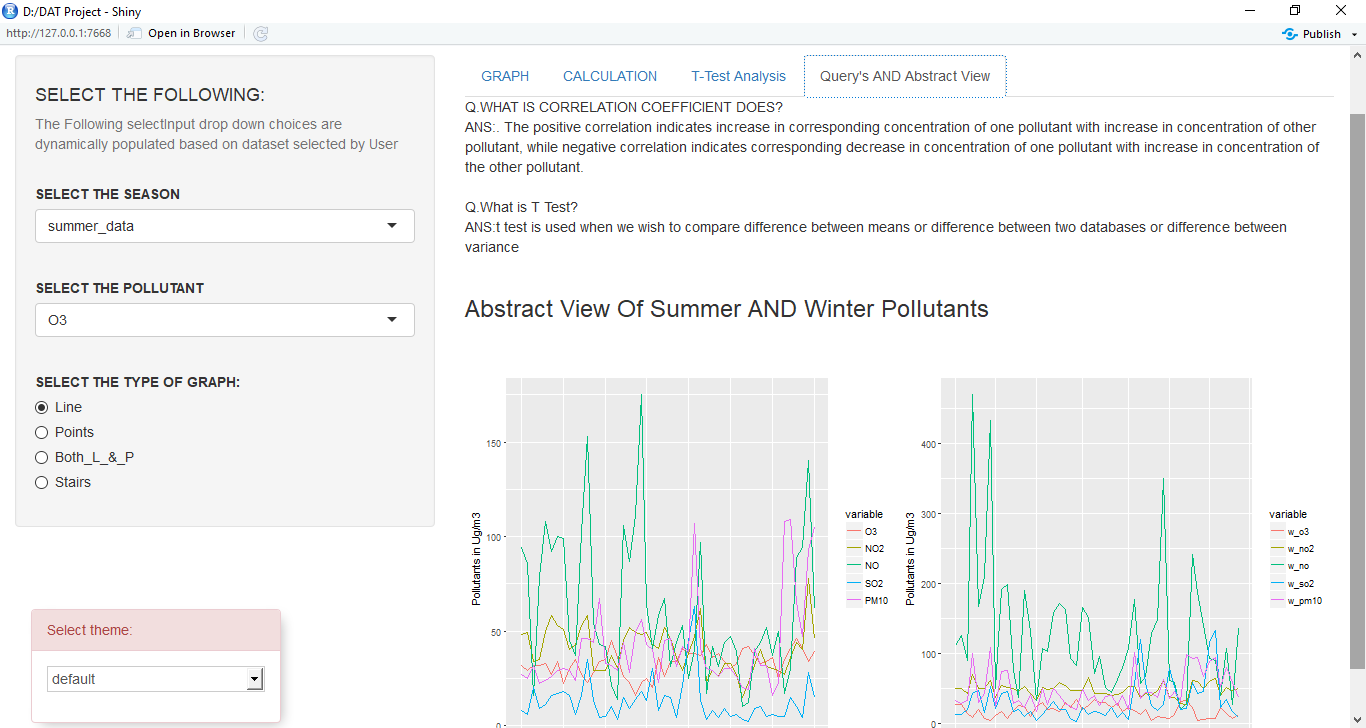
SCREENSHOTS:





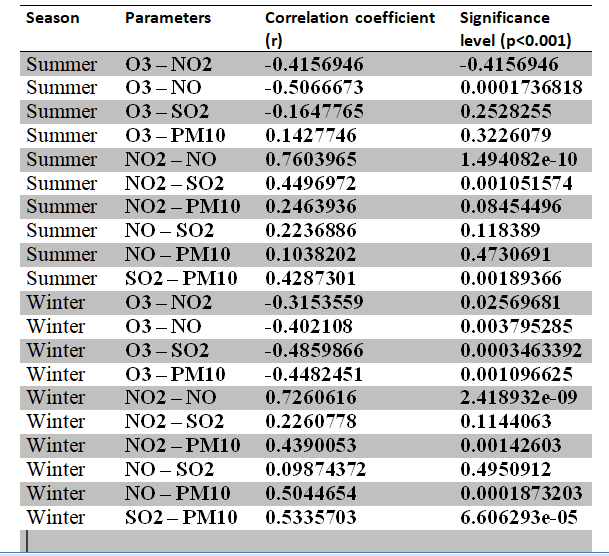




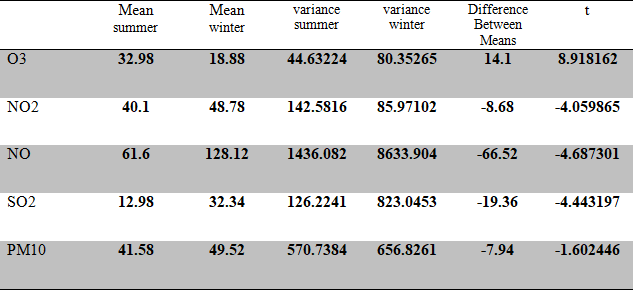


**TABLES:**

**CORRELATION ANALYSIS**

****

**T test analysis**

****