

CA1- statistics for data analytics



Ashok Yalagala (10537771) / Ganji Bharath (10533478) / MANIKANTA YKR (10540014) / Vivek Sanu(10534553)

### Application name : CA1 B9DA101 - Statistics for Data Analytics

### Course : MSc (Data Analytics) - Sep 2019 - Group <<D>>

### Developed by : Ashok Yalagala (10537771) / Ganji Bharath (10533478) / MANIKANTA YKR (10540014) / Vivek Sanu(10534553)

### College : Dublin Business School

### URL : << <https://viveksanu.shinyapps.io/statisticalApplication/>>>

###RUN\_[URL:<<](URL:%3c%3c) <http://viveksanu.com/caStats.zip> >>

## ui.R

**###>>> Begin >>> Vivek Sanu(10534553) >>>**

#install.packages("shinydashboard")

#install.packages("ggplot2")

#install.packages("ggiraph")

#install.packages("ggiraphExtra")

#install.packages("plyr")

library(shiny)

library(shinydashboard)

library(ggplot2)

library(DT)

#library(ggiraph)

#library(ggiraphExtra)

#library(plyr)

dashboardPage(

dashboardHeader(title = "Statistics For Data Analytics",titleWidth =300),

dashboardSidebar(

width = 300,

sidebarMenu(

menuItem("Introduction", tabName = "tabIntroduction", icon = icon("dashboard")),

menuItem("GLM", icon = icon("bar-chart-o"), startExpanded = TRUE,

menuSubItem("Simple LR", tabName = "tabSlr"),

menuSubItem("Multiple LR", tabName = "tabLr"),

menuSubItem("Logistic Regression", tabName = "tabLog")

),

menuItem("Test for Mean", tabName = "tabHypothesis", icon = icon("th")),

menuItem("Discrete Probabilitiy", icon = icon("th"), startExpanded = TRUE,

menuSubItem("Binomial", tabName = "tabBinomial"),

menuSubItem("Poisson", tabName = "tabPoisson")

),

menuItem("Descriptive Statistics", tabName = "tabDescriptive", icon = icon("th"))

)),

dashboardBody(

tabItems(

# Tab Introduction

tabItem(tabName = "tabIntroduction",

fluidRow(

box(

title = "Introduction", status = "primary",width=12,solidHeader = TRUE,

"The project is done as part of CA1 evaluation for the course Statistics For Data Analytics.",

tags$br(),

"Various statistical techniques have been applied on data sets and interesting results have been

observed. Various models of probability model, hypothesis testing and linear regression have been showcased.",tags$br(),

"Dashboard framework has been used to design the UI and it is reponsive."

#. Discrete probability models like Binomial and Poisson model have been used.Hypothesis testing has been done using test of mean using two tailed,lower tailed and upper tailed method.",tags$br(),"Three models of regression model have been done for dynamic datasets.Simple Linear Regression, Multiple Linear Reression and Logistic Linear Regression have been implemented. Various plots have been used to analyze the performance. Descriptve statistcal insights for any dataset can be seen from the 'Descriptive Statistics' Section"

)

),

fluidRow(

box(

title = "Probability Model", status = "warning",width=6,solidHeader = TRUE,

"Discrete probability model like binomial and poisson have been implemented. Various inputs from

users are dynamically accepted and models are made. Graphs are plotted for insights.

"

),

box(

title = "Descriptive Statistics", status = "info",width=6,solidHeader = TRUE,

"Any csv file can be uploaded in the module and various statistical inference can be observed.

"

),

),fluidRow(

box(

title = "Hypothesis Testing", status = "danger",width=6,solidHeader = TRUE,

"Hypothesis testing using test for mean covering lower tailed, upper tailed and two tailed test have been implemented.

Users can provide data and the critical value along with whether the hypothesis is accepted or rejected is shown.

Probability distribution graph is also rendered.

"

),

box(

title = "Regression Model", status = "success",width=6,solidHeader = TRUE,

"Regression model like simple linear regression, multiple linear regression and logistic regression have been implemented. Users can upload files and view the model performance.

Diagnostic plots, plot along LSRL, Confusion Matrix plot and cutoff based on accuracy have been generated.

"

)

)

#tab ends

),

# Tab Probability

tabItem(tabName = "widgets",

h2("Widgets tab content")

),

# Tab GLM

#tab simple lr

tabItem(tabName = "tabSlr",

fluidRow(

box(width=12,

box( solidHeader = TRUE, width = 3,fileInput("fileFile", "Choose CSV File",

multiple = TRUE,

accept = c("text/csv",

"text/comma-separated-values,text/plain",

".csv"))

),

box( solidHeader = TRUE,width=3,

selectInput(inputId="selectPredictorSlr", label = ("Select Predictor"),

choices = list()

)

),

box(solidHeader = TRUE,width=3,

selectInput(inputId="selectResponseSlr", label = ("Select Response"),

choices = list()

)

),

box(solidHeader = TRUE,width=3,tags$br(),

actionButton(inputId = "buttonGo",

label = "Apply Simple LR",style="color: #fff; background-color: #337ab7; border-color: #2e6da4")

)

)

),#end fr

fluidRow(

box(width=12,DT::dataTableOutput("slmData"),title = "View Data Set (Click + to expand)",status = 'primary',solidHeader = TRUE, collapsible = TRUE,collapsed = TRUE

)

),

fluidRow(

box(width=12,verbatimTextOutput("slmSummary"),title = "Summary",status = 'primary',solidHeader = TRUE, collapsible = TRUE,collapsed = TRUE

),

box(width=6,plotOutput("slmGraph"),title = "Plot Along LSRL ",status = 'primary',solidHeader = TRUE, collapsible = TRUE

),

box(width=6,plotOutput("slmGraph2"),title = "Diagnostic Plots",status = 'primary',solidHeader = TRUE, collapsible = TRUE

)

)#end fr

),

#tab multi linear regression

tabItem(tabName = "tabLr",

fluidRow(

box(width=12,

box( solidHeader = TRUE, width = 3,fileInput("fileFileLr", "Choose CSV File",

multiple = TRUE,

accept = c("text/csv",

"text/comma-separated-values,text/plain",

".csv"))

),

box( solidHeader = TRUE,width=3,

selectInput(inputId="selectPredictorLr", label = ("Select Predictors"),multiple = TRUE,

choices = list()

)

),

box(solidHeader = TRUE,width=3,

selectInput(inputId="selectResponseLr", label = ("Select Response"),

choices = list()

)

),

box(solidHeader = TRUE,width=3,tags$br(),

actionButton(inputId = "buttonGoLr",

label = "Apply Multiple LR",style="color: #fff; background-color: #337ab7; border-color: #2e6da4")

)

)

),#end fr

fluidRow(

box(width=12,DT::dataTableOutput("lrData"),title = "View Data Set (Click + to expand)",status = 'primary',solidHeader = TRUE, collapsible = TRUE,collapsed = TRUE

)

),

fluidRow(

box(width=12,verbatimTextOutput("lrSummary"),title = "Summary",status = 'primary',solidHeader = TRUE, collapsible = TRUE,collapsed = TRUE

)

,box(width=12,plotOutput("lrGraph"),title = "Diagnostic Plot ",status = 'primary',solidHeader = TRUE, collapsible = TRUE

)

)#end fr

),#tab logistic regression

tabItem(tabName = "tabLog",

fluidRow(

box(width=12,

box( solidHeader = TRUE, width = 3,fileInput("fileFileLog", "Choose CSV File",

multiple = TRUE,

accept = c("text/csv",

"text/comma-separated-values,text/plain",

".csv"))

),

box( solidHeader = TRUE,width=3,

selectInput(inputId="selectPredictorLog", label = ("Select Predictors"),multiple = TRUE,

choices = list()

)

),

box(solidHeader = TRUE,width=3,

selectInput(inputId="selectResponseLog", label = ("Select Response"),

choices = list()

)

),

box(solidHeader = TRUE,width=3,tags$br(),

actionButton(inputId = "buttonGoLog",

label = "Logistic Regression",style="color: #fff; background-color: #337ab7; border-color: #2e6da4")

)

)

),#end fr

fluidRow(

box(width=12,DT::dataTableOutput("logData"),title = "View Data Set (Click + to expand)",status = 'primary',solidHeader = TRUE, collapsible = TRUE,collapsed = TRUE

)

),

fluidRow(

box(width=12,verbatimTextOutput("logSummary"),title = "Summary",status = 'primary',solidHeader = TRUE, collapsible = TRUE,collapsed = TRUE

),

box(width=12,plotOutput("logGraph"),title = "Performance Plots",status = 'primary',solidHeader = TRUE, collapsible = TRUE

)

)#end fr

),

**###<<< End   <<< Vivek Sanu(10534553) <<<**

**###>>> Begin >>> Ganji Bharath (10533478) >>>**

# tab descriptive

tabItem(tabName = "tabDescriptive",

fluidRow(

box(width=12,

box( solidHeader = TRUE, width = 6,fileInput("fileFileDes", "Choose CSV File",

multiple = TRUE,

accept = c("text/csv",

"text/comma-separated-values,text/plain",

".csv"))

),

box(solidHeader = TRUE,width=6,tags$br(),

actionButton(inputId = "buttonGoDes",

label = "View Basic Descriptive Stats",style="color: #fff; background-color: #337ab7; border-color: #2e6da4")

)

)

),#end fr

fluidRow(

box(width=12,DT::dataTableOutput("desData"),title = "View Data Set (Click + to expand)",status = 'primary',solidHeader = TRUE, collapsible = TRUE,collapsed = TRUE

)

),

fluidRow(

box(width=12,verbatimTextOutput("desSummary"),title = "Summary",status = 'primary',solidHeader = TRUE, collapsible = TRUE)

)#end fr

),

**###<<< End <<< Ganji Bharath (10533478) <<<**

**###>>> Begin >>> Ashok Yalagala (10537771) >>>**

# Tab Hypothesis

tabItem(tabName = "tabHypothesis",

fluidRow(

box(width=12,

box( solidHeader = TRUE,width=3,

numericInput("textSampleMean", "Sample Mean:",value=0)

),

box(solidHeader = TRUE,width=3,

numericInput("textPopulationMean", "Population Mean:",value=0),

),

box(solidHeader = TRUE,width=3,

numericInput("textSD", "Population SD:",value=0),

),

box(solidHeader = TRUE,width=3,

numericInput("textSize", "Sample Size:",value=0),

)

)

),#end fr

fluidRow(

box(width=12,

box(solidHeader = TRUE,width=4,

selectInput(inputId="selectTestType", label = ("Select Test"),

choices = list("Two Tailed","Lower Tailed","Upper Tailed")

)

),

box(solidHeader = TRUE,width=4,

sliderInput("sliderSig", "Significane Level:", 0, 0.25, .05)

),

box(solidHeader = TRUE,width=4,tags$br(),

actionButton(inputId = "buttonGoHypo",

label = "Apply Hypothesis",style="color: #fff; background-color: #337ab7; border-color: #2e6da4")

)

)

),#end fr

fluidRow(

box(width=12,verbatimTextOutput("hypoSummary"),title = "Summary",status = 'primary',solidHeader = TRUE, collapsible = TRUE

),

box(width=12,plotOutput("hypoGraph"),title = "Probability Distribution",status = 'primary',solidHeader = TRUE, collapsible = TRUE

)

)#end fr

),

**###<<< End <<< Ashok Yalagala (10537771) <<<**

**###>>> Begin >>> MANIKANTA YKR (10540014) >>>**

#Tab probability poisson

tabItem(tabName = "tabPoisson",

fluidRow(

box(width=12,

box(solidHeader = TRUE,width=2,

numericInput("textLambdaPos", "Lambda:",value=0),

),

box(solidHeader = TRUE,width=2,

numericInput("textUpperXPos", "Upper Limit X:",value=0),

),

box(solidHeader = TRUE,width=2,

numericInput("textJPos", "J:",value=0),

),

box(solidHeader = TRUE,width=4,

sliderInput("sliderPos", "Count Of Stimulated Data:", 1, 800, 80)

),

box(solidHeader = TRUE,width=2,tags$br(),

actionButton(inputId = "buttonGoPos",

label = "Apply",style="color: #fff; background-color: #337ab7; border-color: #2e6da4")

)

)

),#end fr

fluidRow(

box(width=12,plotOutput("posGraph"),title = "Plots",status = 'primary',solidHeader = TRUE, collapsible = TRUE

)

)#end fr

),

tabItem(tabName = "tabBinomial",

fluidRow(

box(width=12,

box(solidHeader = TRUE,width=2,

numericInput("textPBin", "Parameter P:",value=0),

),

box(solidHeader = TRUE,width=2,

numericInput("textNBin", "Parameter N:",value=0),

),

box(solidHeader = TRUE,width=2,

numericInput("textUpperXBin", "Upper Limit X:",value=0),

),

box(solidHeader = TRUE,width=2,

numericInput("textJBin", "J:",value=0),

),

box(solidHeader = TRUE,width=4,

sliderInput("sliderBin", "Count Of Stimulated Data:", 1, 800, 80)

),

box(solidHeader = TRUE,width=2,tags$br(),

actionButton(inputId = "buttonGoBin",

label = "Apply",style="color: #fff; background-color: #337ab7; border-color: #2e6da4")

)

)

),#end fr

fluidRow(

box(width=12,plotOutput("binGraph"),title = "Plots",status = 'primary',solidHeader = TRUE, collapsible = TRUE

)

)#end fr

)

)

)

)

**###<<< End <<< MANIKANTA YKR (10540014) <<<**

**##SERVER.UI**

**###>>> Begin >>> Vivek Sanu(10534553) >>>**

(function(input, output,session) {

#simple Linear Regression

csvData <- reactive({ #reactive event to store csv file data

csvFile <- input$fileFile

if (is.null(csvFile)) {

return()

}

initialData = read.csv(file=csvFile$datapath)

})

#adding option to select input and clearing graph area

observe({

selectFields=colnames(csvData());#saving column name

updateSelectInput(session, "selectPredictorSlr",

choices = selectFields,selected = selectFields[1])#setting default of predtictor to first column

updateSelectInput(session, "selectResponseSlr",choices = selectFields,selected = selectFields[2])#setting default of predtitor to second column

output$slmGraph = renderPlot({})

output$slmGraph2 = renderPlot({})

})

output$slmData=DT::renderDataTable({

tempData=csvData();

DT::datatable(tempData, options = list(lengthChange = TRUE,scrollX = TRUE))

})

observeEvent(input$buttonGo, {#triggers when go is clicked

validate(

need(input$selectPredictorSlr != "", ""),

need(input$selectResponseSlr != "", "")

)

temp=csvData();

temp=na.omit(temp)

xt = temp[,input$selectPredictorSlr]

yt=temp[,input$selectResponseSlr]

lmFit=lm(yt~xt,data=temp)

xlabel=input$selectPredictorSlr

ylabel=input$selectResponseSlr

summarySlm=summary(lmFit)

output$slmGraph <- renderPlot({

ggplot(temp,aes(y=yt,x=xt))+geom\_point()+geom\_smooth(method="lm")+ labs(x=xlabel, y=ylabel)

#ggPredict(lmFit,se=TRUE,interactive=TRUE)

})

output$slmGraph2 <- renderPlot({

par(mfrow=c(2,2))

plot(lmFit)

#plot(predict(lmFit),residuals(lmFit))

})

output$slmSummary <- renderPrint({

summarySlm

})

})

#Multiple linear regression

csvDataLr <- reactive({ #reactive event to store csv file data

csvFileLr <- input$fileFileLr

if (is.null(csvFileLr)) {

return()

}

initialDataLr = read.csv(file=csvFileLr$datapath)

})

#adding option to select input and clearing graph area

observe({

selectFieldsLr=colnames(csvDataLr());#saving column name

updateSelectInput(session, "selectPredictorLr",choices = selectFieldsLr)#setting default of predtictor to first column

updateSelectInput(session, "selectResponseLr",choices = selectFieldsLr,selected = 0)#setting default of predtitor to second column

output$lrGraph = renderPlot({})

output$lrSummary = renderPrint({})

})

output$lrData=DT::renderDataTable({

tempDataLr=csvDataLr();

DT::datatable(tempDataLr, options = list(lengthChange = TRUE,scrollX = TRUE))

})

observeEvent(input$buttonGoLr, {#triggers when go is clicked

validate(

need(input$selectPredictorLr != "", ""),

need(input$selectResponseLr != "", "")

)

tempLr=csvDataLr();

tempLr=na.omit(tempLr)#data cleaning

#xtLr = tempLr[,input$selectPredictorLr]

#ytLr=tempLr[,input$selectResponseLr]

form=reformulate(termlabels = input$selectPredictorLr,response = input$selectResponseLr)

#print("1");

lmFitLr=lm(form,data=tempLr)

#print("2");

#xlabel=input$selectPredictorSlr

#ylabel=input$selectResponseSlr

summaryLr=summary(lmFitLr)

#summary(lmFitLr)

output$lrGraph <- renderPlot({

#print("eeeeeeeeeeeee")

par(mfrow=c(2,2))

plot(lmFitLr)

#plot(predict(lmFit),residuals(lmFit))

})

output$lrSummary <- renderPrint({

summaryLr

})

})

#Logistic regression

csvDataLog <- reactive({ #reactive event to store csv file data

csvFileLog <- input$fileFileLog

if (is.null(csvFileLog)) {

return()

}

initialDataLog = read.csv(file=csvFileLog$datapath)

})

#adding option to select input and clearing graph area

observe({

selectFieldsLog=colnames(csvDataLog());#saving column name

updateSelectInput(session, "selectPredictorLog",choices = selectFieldsLog)#setting default of predtictor to first column

updateSelectInput(session, "selectResponseLog",choices = selectFieldsLog,selected = 0)#setting default of predtitor to second column

output$logGraph = renderPlot({})

output$logSummary = renderPrint({})

})

output$logData=DT::renderDataTable({

tempDataLog=csvDataLog();

DT::datatable(tempDataLog, options = list(lengthChange = TRUE,scrollX = TRUE))

})

observeEvent(input$buttonGoLog, {#triggers when go is clicked

validate(

need(input$selectPredictorLog != "", ""),

need(input$selectResponseLog != "", "")

)

tempLog=csvDataLog();

tempLog=na.omit(tempLog)

#xtLog = tempLog[,input$selectPredictorLog]

#ytLog=tempLog[,input$selectResponseLog]

form=reformulate(termlabels = input$selectPredictorLog,response = input$selectResponseLog)

#print("1");

lmFitLog=glm(form,data=tempLog,family = "binomial")

summaryLog=summary(lmFitLog)

#summary(lmFitLog)

output$logGraph <- renderPlot({

#print("eeeeeeeeeeeee")

#lmFitLog=lmFitLog;

accry = NULL

thold = seq(0.1,0.9, by=.01)

for (i in 1:length(thold)){

accry[i] = Conf(lmFitLog, cutoff = thold[i])$acc

}

cutoff = thold[which.max(accry)]

conf.mat = Conf(lmFitLog, cutoff = cutoff)

layout(matrix(1:2,ncol = 2))

plot(thold, accry, type = "l", main = "Cutoff Based on Accuracy")

abline(h=max(accry), v = cutoff, col="red")

fourfoldplot(conf.mat$table, main = "Confusion Matrix Plot", color=c("red","green"))

})

output$logSummary <- renderPrint({

summaryLog

})

})

**###<<< End <<< Vivek Sanu(10534553) <<<**

**###>>> Begin >>> Ganji Bharath (10533478) >>>**

#descriptive stats

csvDataDes <- reactive({ #reactive event to store csv file data

csvFileDes <- input$fileFileDes

if (is.null(csvFileDes)) {

return()

}

initialDataDes = read.csv(file=csvFileDes$datapath)

})

#adding option to select input and clearing graph area

observe({

selectFieldsDes=colnames(csvDataDes());#saving column name

output$desSummary = renderPrint({})

})

output$desData=DT::renderDataTable({

tempDataDes=csvDataDes();

DT::datatable(tempDataDes, options = list(lengthChange = TRUE,scrollX = TRUE))

})

observeEvent(input$buttonGoDes, {#triggers when go is clicked

tempDes=csvDataDes();

summaryDes=summary(tempDes)

#summary(lmFitDes)

output$desSummary <- renderPrint({

if (is.na(summaryDes)==FALSE){

summaryDes

# print(summaryDes)

}

})

})

**###<<< End <<< Ganji Bharath (10533478) <<<**

**###>>> Begin >>> Ashok Yalagala (10537771) >>>**

#####hypothesis mean

observeEvent(input$buttonGoHypo, {#triggers when go is clicked

#validate(

#need(input$selectColumn != "", ""),

# need(input$textMean!= "", "")

# )

#xtHypo = tempHypo[,input$selectPredictorHypo]

#ytHypo=tempHypo[,input$selectResponseHypo]

#summaryHypo=summary(lmFitHypo)

#summary(lmFitHypo)

#print(input$selectTestType)

#print("ASdasd");

testType=input$selectTestType

sampleMean=(input$textSampleMean);

sd=(input$textSD)

sampleSize=(input$textSize)

populationMean=(input$textPopulationMean)

sig=(input$sliderSig)

print(sig)

testStatistic=(sampleMean-populationMean)/(sd/sqrt(sampleSize))

output$hypoGraph <- renderPlot({

#print("eeeeeeeeeeeee")

#lmFitHypo=lmFitHypo;

x= seq(-1\*(populationMean+15), populationMean+15, by = 1)

y <- dnorm(x,populationMean, sd )

plot(x,y)

})

output$hypoSummary <- renderPrint({

if(testType=="Upper Tailed"){

halfAlpha=qt(1-sig,df = sampleSize-1)

print(paste("Critical Value =", halfAlpha));

print(paste("Test Statistic = ",testStatistic))

if(testStatistic<halfAlpha){

print("We accept the hypothesis")

}else {

print("We reject the hypothesis")

}

}else if(testType=="Lower Tailed"){

halfAlpha=qt(1-sig,df = sampleSize-1)

nhalfAlpha=-1\*halfAlpha

print(paste("Critical Value =", nhalfAlpha));

print(paste("Test Statistic = ",testStatistic))

if(testStatistic>=nhalfAlpha){

print("We accept the hypothesis")

}else {

print("We reject the hypothesis")

}

} else if(testType=="Two Tailed"){

halfAlpha=qt(1-sig/2,df = sampleSize-1)

nhalfAlpha=-1\*halfAlpha

print(paste("Critical Values = (",halfAlpha,")(",nhalfAlpha,")"));

print(paste("Test Statistic = ",testStatistic))

if((testStatistic>=nhalfAlpha) && (testStatistic<=halfAlpha) ){

print("We accept the hypothesis")

}else {

print("We reject the hypothesis")

}

}

})

})

**###<<< End <<< Ashok Yalagala (10537771) <<<**

**###>>> Begin >>> MANIKANTA YKR (10540014) >>>**

#poisson model

observeEvent(input$buttonGoPos, {#triggers when go is clicked

count=input$sliderPos

lambda=input$textLambdaPos

upperX=input$textUpperXPos

output$posGraph <- renderPlot({

par(mfrow=c(1,2))

distr=rpois(count, lambda)

tabu=table(distr)

xL=0:upperX

yL=dpois(xL,lambda)

barplot(tabu,col='green')

plot(xL,yL,type='b')

})

})

#binomial model

observeEvent(input$buttonGoBin, {#triggers when go is clicked

countBin=input$sliderBin

pBin=input$textPBin

nBin=input$textNBin

upperXG=input$textUpperXBin

output$binGraph <- renderPlot({

par(mfrow=c(1,2))

d <- density(rbinom(1000,nBin,pBin))

plot(d, main="Kernel Density of generated data")

polygon(d, col="green", border="blue")

x=0:nBin

plot(x,dbinom(x,nBin,pBin))

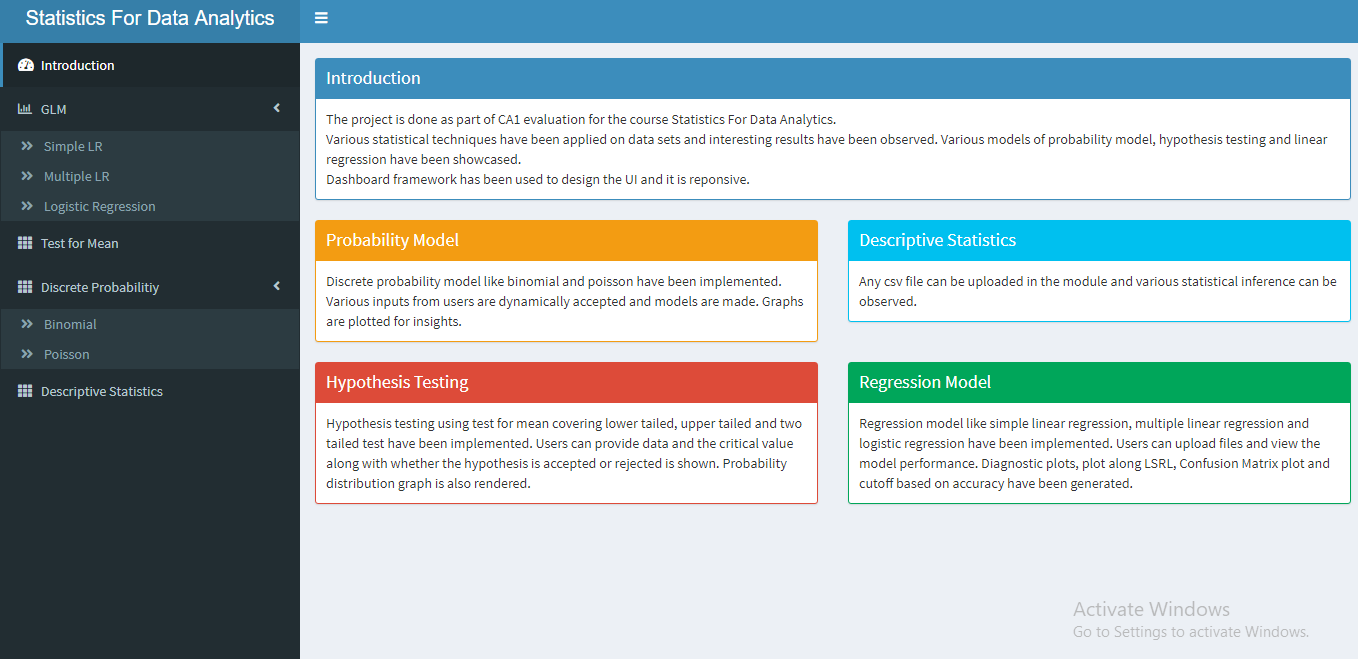
})

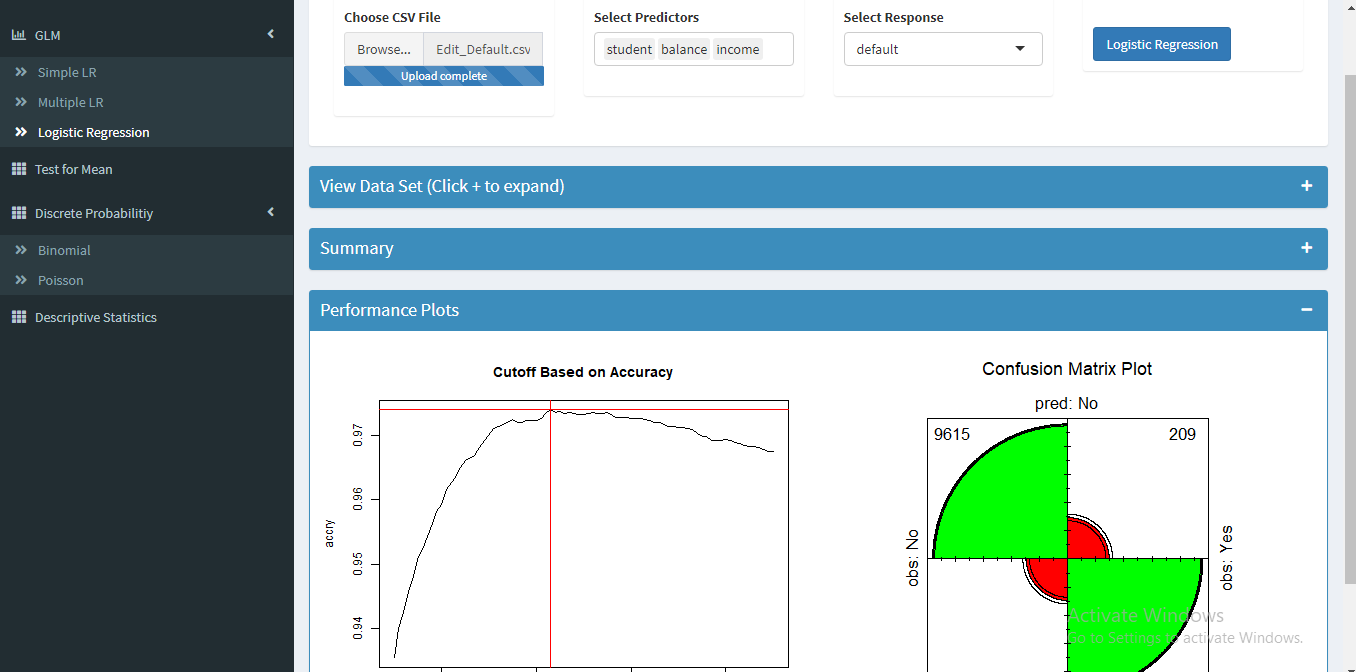
})

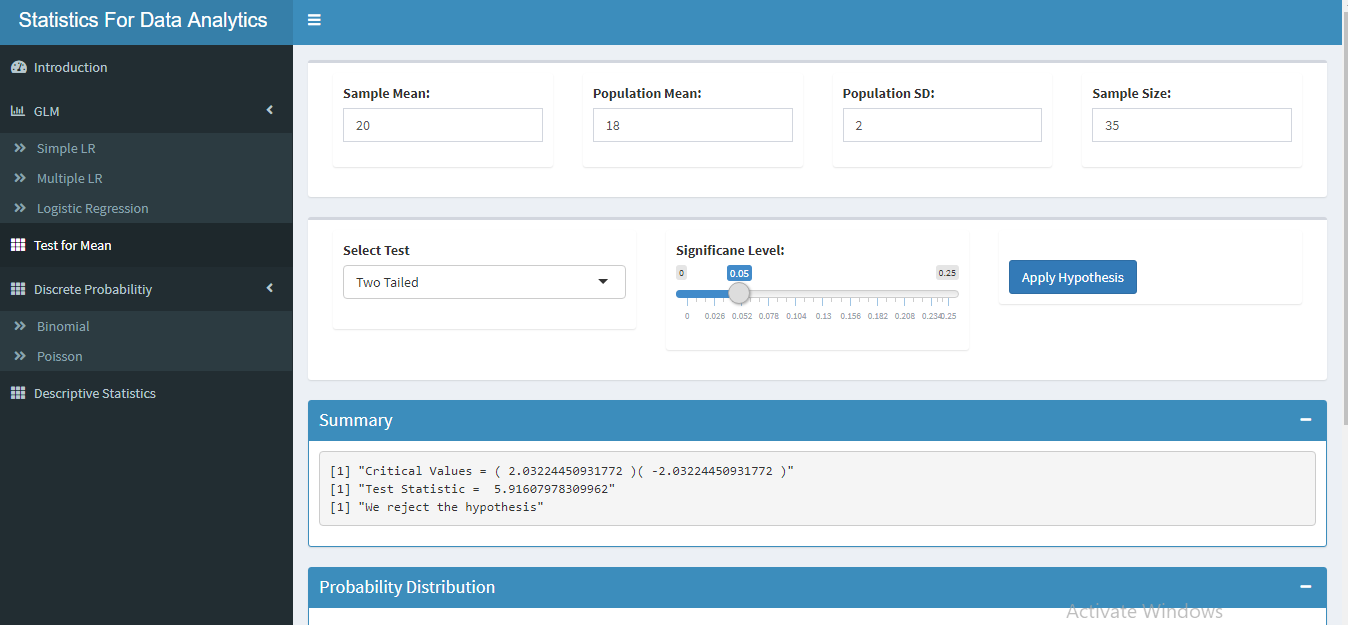
})

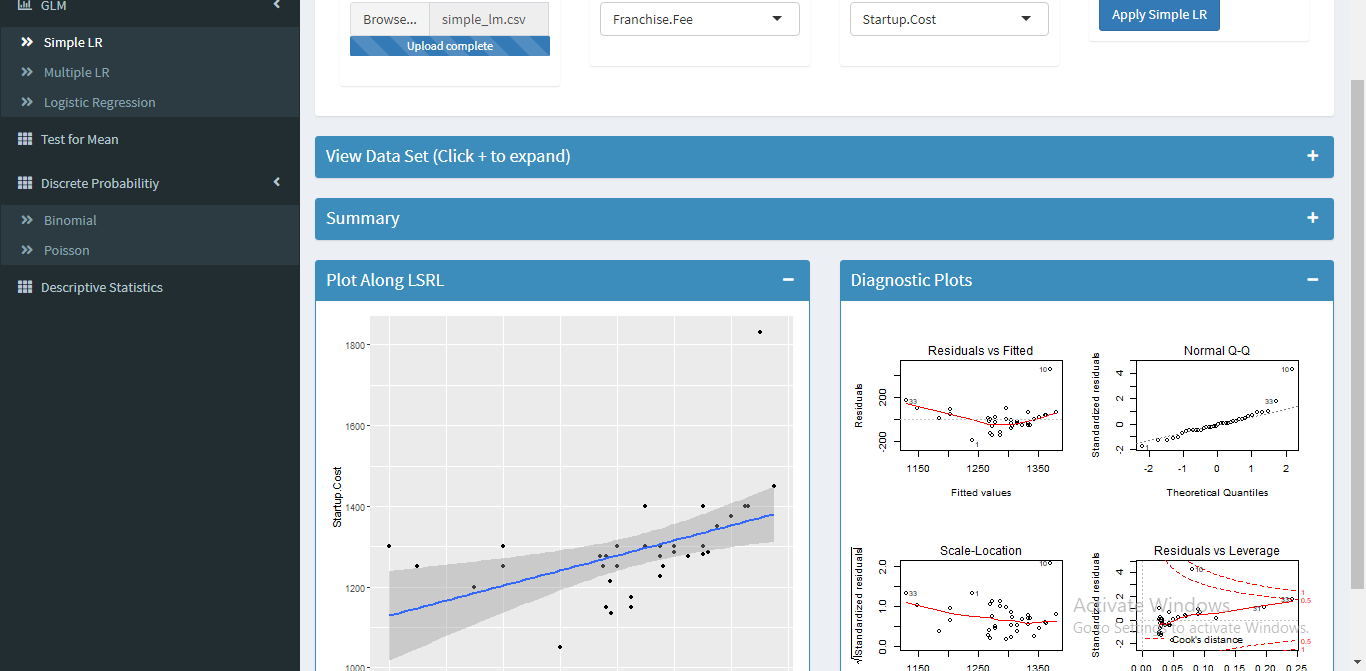
**###<<< End <<< MANIKANTA YKR (10540014) <<<**

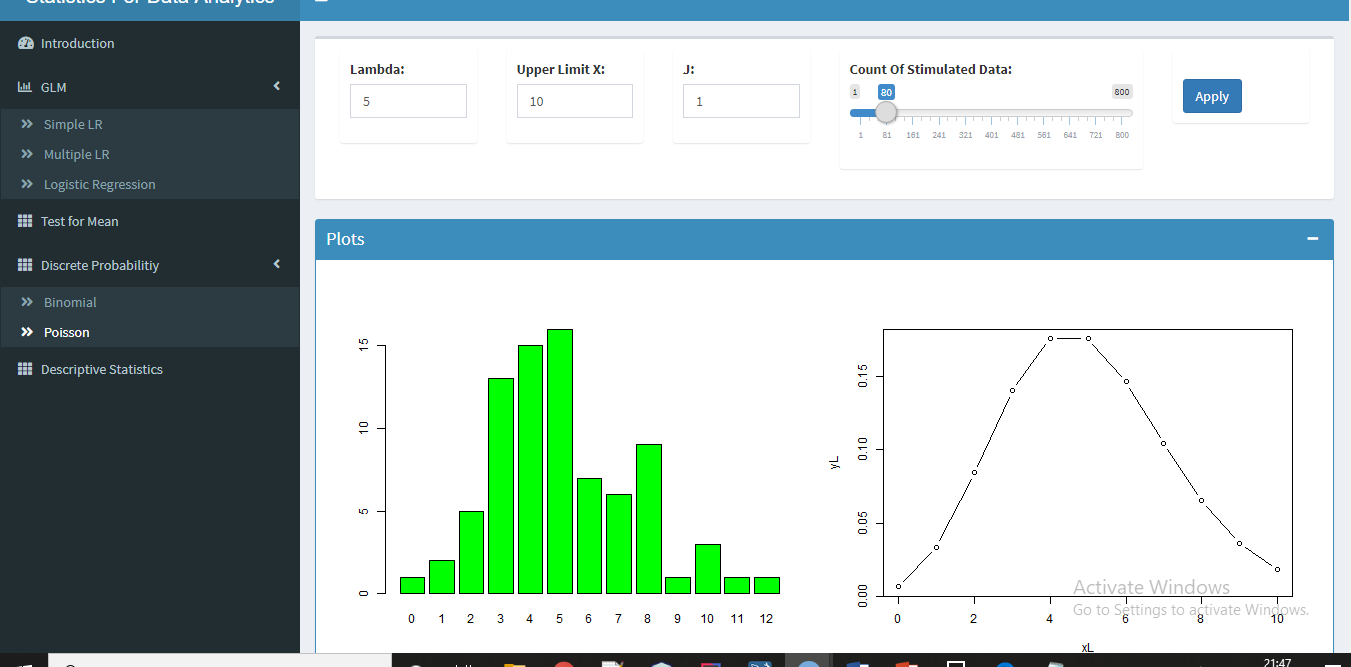
**SCREEN SHOTS**

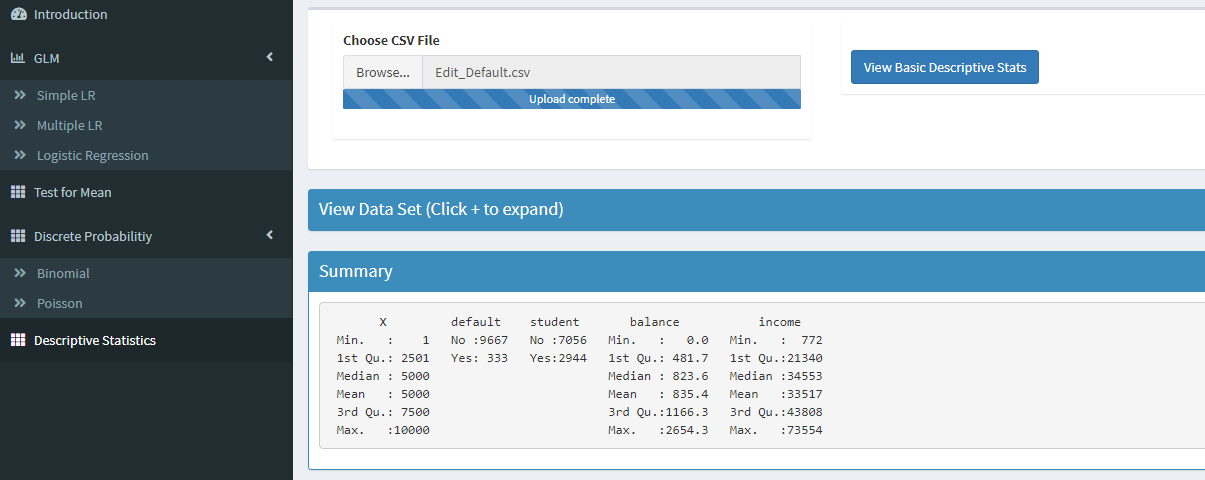
****

****

****

****

****

****