PROGRAM CODE

SalesForecast.R

library(plyr)

library(ggplot2)

library(lubridate)

library(dplyr)

library(forecast)

library(tidyr)

nsalesdf<-read.csv("F:/Msc Part 2 Sem 2/Project/Global\_supermarket.csv",TRUE)

View(nsalesdf)

names(nsalesdf)<-c("Invoice ID","Date","Gender","City","Branch","Customer Type","Time","Productline","Quantity","Payment","Sales","Gross Income","Rating")

View(nsalesdf)

#Data preprocessing

nsalesd<- nsalesdf %>%

mutate(DATE=mdy(Date))%>%

select(everything())

View(nsales)

nsalesd <- nsalesd%>%

mutate(

YEAR=year(DATE)

)

nsalesd<-nsalesd %>%

separate(Time, sep=":", into = c("Time"))

nsales <- data.frame(nsalesdf$`Invoice ID`,nsalesd$DATE,nsalesd$YEAR,nsalesd$Gender,nsalesd$City,nsalesd$Branch,nsalesd$`Customer Type`,nsalesd$Time,nsalesd$Productline,

nsalesd$Quantity,nsalesd$Payment,nsalesd$Sales,nsalesd$`Gross Income`,nsalesd$Rating)

names(nsales)<-c('Invoice ID','DATE','YEAR','Gender','City','Branch',

'CustomerType','Time','Productline','Quantity','Payment','Sales','gross\_income',

'Rating')

View(nsales)

nBranchA<- filter(nsales, Branch=='A')

nBranchB<- filter(nsales, Branch=='B')

nBranchC<- filter(nsales, Branch=='C')

View(nBranchA)

nSalesSum<-aggregate(nsales$Sales, by=list(YEAR=nsales$YEAR,Branch=nsales$Branch), FUN=sum)

View(nSalesSum)

#Daily sales based on the type of product

#graph shows the number of transactions per store over the period.

count(nsales,"Branch")

# graph shows quantity of sales per category in each branch. It appears that each store has a certain category that they sell more of than the other stores

View(nsales)

#total sales per Year per product line over the time period for Branch A

#Total sales in branches

tsd1<-aggregate(nsales$Sales, by=list(Branch=nsales$Branch), FUN=sum)

#Total sales in years

tsd2<-aggregate(nsales$Sales, by=list(Year=nsales$YEAR), FUN=sum)

#Total sales in branch per product category

tsd3<-aggregate(nsales$Sales, by=list(Branch=nsales$Branch,Product=nsales$Productline), FUN=sum)

View(tsd3)

#Total sales in year per product cateogory

tsd4<-aggregate(nsales$Sales, by=list(Year=nsales$YEAR,Product=nsales$Productline), FUN=sum)

View(tsd4)

#pie chart per product category in each branch

brAag<-aggregate(nBranchA$Quantity, by=list(Products=nBranchA$Productline), FUN=sum)

View(brAag)

brBag<-aggregate(nBranchB$Quantity, by=list(Products=nBranchB$Productline), FUN=sum)

View(brBag)

brCag<-aggregate(nBranchC$Quantity, by=list(Products=nBranchC$Productline), FUN=sum)

View(brCag)

#Bar plot for gross income per branch

grossdf<-aggregate(nsales$gross\_income, by=list(Branch=nsales$Branch,Year=nsales$YEAR), FUN=sum)

View(grossdf)

names(grossdf)=c('Branch','Year','Total\_gross\_income')

grossdf2<-aggregate(nsales$gross\_income, by=list(Product=nsales$Productline,Year=nsales$YEAR), FUN=sum)

View(grossdf2)

names(grossdf2)=c('Product','Year','Total\_gross\_income')

#Sales by hour in supermarket

#Quantity of Products sold per hour

timedf3<-aggregate(nsales$Quantity,by=list(Time=nsales$Time),FUN=sum)

View(timedf3)

x=c(timedf3$x)

labels=c(timedf3$Time)

#All branches

timedf2<-aggregate(nsales$Quantity,by=list(Time=nsales$Time,Branch=nsales$Branch),FUN=sum)

View(timedf2)

names(timedf2)=c('Time','Branch','Quantity')

#Rating

#Overall ratings

ratedf<-aggregate(nsales$Rating,by=list(Branch=nsales$Branch,Productline=nsales$Productline),FUN=mean)

View(ratedf)

names(ratedf)=c('Branch','Product','Rating')

#Payment mode

pay1<-count(nsales,vars="Payment")

View(pay1)

paydf2<-aggregate(nsales$Sales, by=list(Payment=nsales$Payment,Branch=nsales$Branch), FUN=sum)

View(paydf2)

names(paydf2)=c('Payment','Branch','Sales')

#Gender

gen1<-aggregate(nsales$Quantity, by=list(Product=nsales$Productline,Gender=nsales$Gender), FUN=sum)

View(gen1)

names(gen1)=c('Product','Gender','Total\_Quantity')

#Customer type

type1<-aggregate(nsales$Quantity, by=list(Product=nsales$Productline,Customer\_type=nsales$CustomerType), FUN=sum)

View(type1)

names(type1)=c('Product','Customer\_type','Total\_Quantity')

type2<-aggregate(nsales$Quantity, by=list(Branch=nsales$Branch,Customer\_type=nsales$CustomerType), FUN=sum)

View(type2)

custA<- filter(type2, Branch=='A')

custB<- filter(type2, Branch=='B')

custC<- filter(type2, Branch=='C')

View(custA)

#Time series analysis and forecasting

Data\_frame\_1=aggregate(nsales$Sales, by=list(Month=month(nsales$DATE),Year=year(nsales$DATE)), FUN=sum)

View(sn3)

sn4=aggregate(nsales$Sales, by=list(Date=nsales$DATE), FUN=sum)

View(sn4)

Training\_dataset<-Data\_frame\_1[1:38,]

Testing\_dataset<-Data\_frame\_1[39:48,]

colnames(Training\_dataset)<-c('Month','Year','Total Sales')

colnames(Testing\_dataset)<-c('Month','Year','Total Sales')

View(Training\_dataset)

View(Testing\_dataset)

Time\_Series\_1=ts(Training\_dataset$`Total Sales`

,

start =2011,

frequency =12 )

Time\_Series\_1

plot(Time\_Series\_1)

ETS\_Model=ets(Time\_Series\_1)

summary(ETS\_Model)

ETS\_Forecast=forecast(ETS\_Model,h=10)

ETS\_Forecast

etsmean=ETS\_Forecast$mean

monthc=c('March','Apr','May','June','July','Aug','Sept','Oct','Nov','Dec')

etsd=cbind(monthc,round(Testing\_dataset$`Total Sales`,2),round(etsmean,2),round(ETS\_Forecast$lower,2),round(ETS\_Forecast$upper,2))

colnames(etsd)<-c('Month','Actuals','Forecasted','Hi 80','Hi 95','Lo 80','Lo 95')

View(etsd)

accuracy(ETS\_Forecast,Testing\_dataset$`Total Sales`)

Final\_time\_series=ts(Data\_frame\_1$x,

start =2011,

frequency =12 )

Final\_time\_series

ts.plot(Final\_time\_series)

Final\_ETS\_model<-ets(Final\_time\_series)

summary(Final\_ETS\_model)

Final\_Forecast=forecast(Final\_ETS\_model,h=12) #finaletsf

Final\_Forecast

fidf=data.frame(Final\_Forecast)

View(fidf)

ARIMA\_Model\_1=auto.arima(Time\_Series\_1)

ARIMA\_Model\_1

ARIMA\_Forecast=forecast(ARIMA\_Model\_1,h=10)

ARIMA\_Forecast

plot(ARIMA\_Forecast)

arimamean=ARIMA\_Forecast$mean

arimamean

arimad=cbind(monthc,round(Testing\_dataset$`Total Sales`,2),round(arimamean,2),round(ARIMA\_Forecast$upper,2),round(ARIMA\_Forecast$lower,2))

colnames(arimad)<-c('Month','Actuals','Forecasted','Hi 80','Hi 95','Lo 80','Lo 95')

View(arimad)

summary(ARIMA\_Model\_1)

accuracy(ARIMA\_Forecast,Testing\_dataset$`Total Sales`)

finalarima<-auto.arima(Final\_time\_series)

summary(finalarima)

finalarimaf=forecast(finalarima,h=12)

finalarimaf

#Time series analysis of Branch A

Data\_frame\_A=aggregate(nBranchA$Sales, by=list(Month=month(nBranchA$DATE),Year=year(nBranchA$DATE)), FUN=sum)

View(Data\_frame\_A)

BranchA\_time\_series=ts(Data\_frame\_A$x,

start =2011,

frequency =12 )

BranchA\_time\_series

ts.plot(BranchA\_time\_series)

BranchA\_ETS\_model<-ets(BranchA\_time\_series)

summary(BranchA\_ETS\_model)

BranchA\_Forecast=forecast(BranchA\_ETS\_model,h=12) #finaletsf

plot(BranchA\_Forecast)

#Time series analysis of Branch B

Data\_frame\_B=aggregate(nBranchB$Sales, by=list(Month=month(nBranchB$DATE),Year=year(nBranchB$DATE)), FUN=sum)

View(Data\_frame\_B)

BranchB\_time\_series=ts(Data\_frame\_B$x,

start =2011,

frequency =12 )

BranchB\_time\_series

ts.plot(BranchB\_time\_series)

BranchB\_ETS\_model<-ets(BranchB\_time\_series)

summary(BranchB\_ETS\_model)

BranchB\_Forecast=forecast(BranchB\_ETS\_model,h=12) #finaletsf

plot(BranchB\_Forecast)

#Time series analysis of Branch C

Data\_frame\_C=aggregate(nBranchC$Sales, by=list(Month=month(nBranchC$DATE),Year=year(nBranchC$DATE)), FUN=sum)

View(Data\_frame\_C)

BranchC\_time\_series=ts(Data\_frame\_C$x,

start =2011,

frequency =12 )

BranchC\_time\_series

ts.plot(BranchC\_time\_series)

BranchC\_ETS\_model<-ets(BranchC\_time\_series)

summary(BranchC\_ETS\_model)

BranchC\_Forecast=forecast(BranchC\_ETS\_model,h=12) #finaletsf

plot(BranchC\_Forecast)

ui.R

library(shiny)

library(shinydashboard)

library(shinythemes)

library(dygraphs)

library(shinyWidgets)

header <- dashboardHeader(

title = "Analysis and Prediction of Supermarket Sales",

titleWidth = 600

)

sidebar <- dashboardSidebar(

width = 300,

sidebarMenu(

menuItem("Data", tabName = "DataMenu",icon = icon("table")),

menuItem("Analysis of Number of sales",

tabName = "nos2",icon = icon("bar-chart")

),

menuItem("Analysis of Total sales",

tabName = "daMenu",icon = icon("dollar"),

menuSubItem("Sales per category in each branch", tabName = "nos4",icon = icon("line-chart")),

menuSubItem("Sales per category in each year", tabName = "nos5",icon = icon("line-chart"))

),

menuItem("Analysis of Total sales across all branches",

tabName = "nos7",icon = icon("line-chart"))

,

menuItem("Analysis of Quantity of Products sold",

tabName = "nos8",icon = icon("pie-chart"))

,

menuItem("Analysis of gross income",

tabName = "nos9",icon = icon("bar-chart-o")

),

menuItem("Analysis of product quantity sold per hour",

tabName = "nos10",icon = icon("bar-chart")

),

menuItem("Analysis of Customer Ratings",

tabName = "nos11",icon = icon("bar-chart")

),

menuItem("Analysis based on Payment mode",

tabName = "nos12",icon = icon("pie-chart")

),

menuItem("Analysis based on Customer Gender",

tabName = "nos13",icon = icon("bar-chart")

),

menuItem("Analysis based on Customer type",

tabName = "ct1",icon = icon("pie-chart")

),

menuItem("Time series analysis",

tabName = "sp",icon = icon("line-chart")

),

menuItem("Training and Testing datasets ", tabName = "traintest"),

menuItem("Forecasting models",

tabName = "sp",icon = icon("th"),

menuSubItem("Exponential Smoothing model", tabName = "ets",icon = icon("th")),

menuSubItem("ARIMA model ", tabName = "arima",icon = icon("th")),

menuSubItem("Exponential Smoothing vs ARIMA model", tabName = "eac",icon = icon("th"))

),

menuItem("Sales prediction",

tabName = "finalf")))

body <- dashboardBody(

tags$head(tags$style(HTML('

/\* main sidebar \*/

.main-sidebar {

background-color: #f4b943;

font-family:Source Sans Pro;

font-size:14.5px;

font-weigth:bold;

}

/\* active selected tab in the sidebarmenu \*/

.main-sidebar .sidebar .sidebar-menu .active a{

background-color:#c299ff;

color:black;

}

/\* other links in the sidebarmenu \*/

.main-sidebar .sidebar .sidebar-menu a{

background-color: black;

color: white;

}

/\* other links in the sidebarmenu when hovered \*/

.main-sidebar .sidebar .sidebar-menu a:hover{

background-color: #944dff ;

color:black;

}

/\* body \*/

.content-wrapper, .right-side {

background-color: #ccccb3;

}

.div{

font-family:Georgia;

}

.skin-purple .main-header .logo {

background-color: #003399;

font-family:Constantia;

font-size:28px;

}

'))

),

tabItems(

tabItem(tabName = "DataMenu",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Dataset"), sep = ""))

),tags$br())

),

fluidRow(

column(2,

h3("Build Dataset"),

),

column(10,align="left",

# Creates the input for uploading files

tags$br(),

fileInput("i\_file",

"Upload your CSV file",

# A character vector of MIME types; gives the browser a hint of what kind of files the server is expecting.

accept = c("text/csv",

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

".csv")),

)

),

fluidRow(

column(width =12,

DT::dataTableOutput("maindf")

))

),

tabItem(tabName = "nos2",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Number of Sales"), sep = ""))

),tags$br())

),

fluidRow(

column(12, align="center",

plotOutput("p2",height = 600,width=800))

)

),

tabItem(tabName = "nos4",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Total Sales"), sep = ""))

),tags$br())

),

fluidRow(

column(5, align="center",

plotOutput("p4",height = 600,width="100%")),

column(7,align="center",

plotOutput("p41",height=600,width="100%")

)

)

),

tabItem(tabName = "nos5",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Total Sales"), sep = ""))

),tags$br())

),

fluidRow(

column(5, align="center",

plotOutput("p5",height = 600,width="100%")),

column(7,align="center",

plotOutput("p51",height=600,width="100%")

)

)

),

tabItem(tabName = "nos7",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Total Sales"), sep = ""))

),tags$br())

),

fluidRow(

column(12, align="center",

plotOutput("p7",height = 600,width=800))

)

),

tabItem(tabName = "nos8",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Quantity of Products Sold"), sep = ""))

),tags$br())

),

fluidRow(

column(4,

align="center",

plotOutput("p8",height = 600)),

column(4,

align="center",

plotOutput("p9",height = 600)),

column(4,

align="center",

plotOutput("p10",height = 600))

)

),

tabItem(tabName = "nos9",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Gross Income"), sep = ""))

),tags$br())

),

fluidRow(

column(12, align="center",

plotOutput("p11",height = 600,width=800))

)

),

tabItem(tabName = "nos10",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Product Quantity Sold per hour"), sep = ""))

),tags$br())

),

fluidRow(

column(5,

align="center",

plotOutput("p12",height = 600)),

column(7,

align="center",

plotOutput("p13",height = 600)),

)

),

tabItem(tabName = "nos11",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis of Customer Ratings"), sep = ""))

),tags$br())

),

fluidRow(

column(4,

align="center",

plotOutput("p15",height = 600)),

column(8,

align="center",

plotOutput("p14",height = 600)))

),

tabItem(tabName = "nos12",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis Based on Payment Mode"), sep = ""))

),tags$br())

),

fluidRow(

column(4,

align="center",

plotOutput("p16",height = 600)),

column(8,

align="center",

plotOutput("p17",height = 600)))

),

tabItem(tabName = "nos13",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis Based on Customer Gender"), sep = ""))

),tags$br())

),

fluidRow(

column(12,

align="center",

plotOutput("p18",height = 600,width = 1100))

)

),

tabItem(tabName = "ct1",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Analysis Based on Customer Type"), sep = ""))

),tags$br())

),

fluidRow(

column(4,

align="center",

plotOutput("ctp2",height = 600))

,

column(4,

align="center",

plotOutput("ctp3",height = 600))

,

column(4,

align="center",

plotOutput("ctp4",height = 600))

)

),

tabItem(tabName ="sp",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Time Series Analysis"), sep = ""))

),tags$br(),tags$br(),tags$br())

),

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#006666;font-size:22px;font-family:Verdana;", "Time Series Plot"), sep = ""))

),tags$br())

),

fluidRow(

column(12,

dygraphOutput("tsplot"),tags$br(),tags$br(),tags$br())

),

fluidRow(

column(width = 12,align="center",uiOutput("o.print.decomp"),tags$br()) ),

fluidRow(

column(width = 12,align="center", plotOutput("o.decomp.plot.additive",height=500,width=800),tags$br(),tags$br(),tags$br())

),

fluidRow(

column(width=12,align="center",uiOutput("o.print.seasonality"),tags$br())

),

fluidRow(

column(width = 12,align="center",

plotOutput("o.seasonplot",height=500,width=800))

)

),

tabItem(tabName = "traintest",

fluidRow(

column(width = 12,align="center",

valueBoxOutput("o.traintest.box",

width = 15))),

fluidRow(

column(width = 6,align="center",

valueBoxOutput("o.train.box.view",

width = 15)),

column(width = 6,align="center",

valueBoxOutput("o.test.box.view",

width = 15))

),

fluidRow(

column(width =6,

DT::dataTableOutput("o.training.data.view")

),

column(width =6,

DT::dataTableOutput("o.testing.data.view")

)

) ),

tabItem(tabName = "ets",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:28px;font-family:Verdana;", "Exponential Smoothing Model"), sep = ""))

),tags$br())

),

fluidRow(

column(12,

align="center",

uiOutput("o.print.etsparameters"),

),

column(width = 12,

verbatimTextOutput("o.print.model"))

),

fluidRow(

align="center",

uiOutput("printactuals"),

column(width = 12,

DT::dataTableOutput("etsdf"))

)

),

tabItem(tabName = "arima",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:28px;font-family:Verdana;", "Autoregressive Integrated Moving Average Model"), sep = ""))

),tags$br())

),

fluidRow(

column(12,

align="center",

uiOutput("o.print.arimaparameters"),

),

column(width = 12,

verbatimTextOutput("o.print.arimamodel"))

),

fluidRow(

column(12,

align="center",

uiOutput("printaractuals"),

),

column(width = 12,

DT::dataTableOutput("arimadf"))

)

),

tabItem(tabName = "eac",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:25px;font-family:Verdana;", "Exponential Smoothing vs ARIMA Model"), sep = ""))

),tags$br(),tags$br())

),

fluidRow(

column(12,align="center",uiOutput("printetsaccuracytext")),

column(width = 12,verbatimTextOutput("printetsmodel"),tags$br(),tags$br())

),

fluidRow(

column(12,align="center",uiOutput("printarimaaccuracytext")),

column(width = 12,verbatimTextOutput("printarimamodel"),tags$br(),tags$br(),tags$br())

),

fluidRow(

column(12,align="center",uiOutput("printetsplot"),tags$br()),

column(width = 12,dygraphOutput("o.dy.pred.on.test"),tags$br(),tags$br())

),

fluidRow(

column(12,align="center",uiOutput("printarimaplot"),tags$br()),

column(width = 12,dygraphOutput("o.dy.pred.on.arimatest"))

)

),

tabItem(tabName = "finalf",

fluidRow(

column(12,align="center",

tags$div(

HTML(paste(tags$span(style="color:#4d0066;font-size:28px;font-family:Verdana;", "Forecasting using Exponential Smoothing Model"), sep = ""))

),tags$br()

)),

fluidRow(

column(width =4 ,

sliderInput(inputId = "no.days",

label = "No. of months to forecast ahead",

min = 1,

max = 12,

value = 6)),

column(4,

selectInput("select", label ="Forecast for",

choices = list("Total Sales at Supermarket" = 1, "Branch A" = 2, "Branch B" = 3,"Branch C"=4),

selected = 1)

),

column(4,

br(),

actionButton("forecast.click",

"Start Forecasting",

icon = icon("edit", "fa-2x"),

width = '70%',

style="color: #ffffff;height=400px;

background-color: #e52240;

border-color: #e52240"),

br(),

)

),

fluidRow(

uiOutput("printfinalsummary"),

column(width = 12,

verbatimTextOutput("printfinalsum")

)

),

fluidRow(

uiOutput("plotftextt"),

column(width = 12,

plotOutput("of"))

),

fluidRow(

uiOutput("plotdftextt"),

column(width = 12,

DT::dataTableOutput("ofdf")

)

)

)

))

shinyUI(

dashboardPage(skin='purple',header, sidebar, body)

)

server.R

library(forecast)

library(ggplot2)

library(lubridate)

library(dplyr)

library(tidyr)

library(dygraphs)

library(shiny)

library(shinydashboard)

library(shinythemes)

shinyServer(function(input,output){

my\_theme<-function(){

theme\_light()+theme(

legend.text = element\_text(color="black"),

text = element\_text(size = 20),

axis.text = element\_text(size = 15),

axis.title = element\_text(size = 18),

plot.title = element\_text(size = 21,hjust = 0.5),

panel.background = element\_rect(fill = "white"),

plot.background = element\_rect(fill="white"),

legend.background = element\_rect(fill="white",size=0.5, linetype="solid",colour ="black")

)

}

dataframe<-reactive({

if (is.null(input$i\_file))

return(NULL)

data<-read.csv(input$i\_file$datapath)

data

})

output$maindf<-DT::renderDataTable({

DT:: datatable(dataframe(),

options = list(pageLength = 15,searching=FALSE))

})

output$p2<-renderPlot({

ggplot (data=nsales)+

geom\_bar(mapping=aes(x=YEAR, fill=Branch), position="dodge")+

my\_theme()+xlab("Year")+ylab("Count")+

scale\_color\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

scale\_fill\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

ggtitle("Number of sales per category in each branch across four years")

})

output$p4<-renderPlot({

ggplot(tsd1, aes(x=Branch, y=x)) +

geom\_bar(position="dodge", stat="identity", width=0.5,fill = "#005c99")+my\_theme()+

xlab("Branch")+ylab("Total sales (in Rs.)")+

ggtitle("Comparison between total sales per branch")

})

output$p41<-renderPlot({

ggplot(tsd3, aes(fill=Product, y=x, x=Branch)) +

geom\_bar(position="dodge", stat="identity") +ggtitle("Comparison between Total Sales per product per branch")+

my\_theme()+

scale\_color\_manual(values = c("#ff4d4d", "#558000", "#33cc00","#0099cc","#24248f","#008080")) +

scale\_fill\_manual(values = c("#ff4d4d", "#558000", "#33cc00","#0099cc","#24248f","#008080"))+

xlab("Branch")+ylab("Total sales (in Rs.)")+

ggtitle("Comparison between total sales per branch")

})

output$p5<-renderPlot({

ggplot(tsd2, aes(x=Year, y=x)) +

geom\_bar(position="dodge", stat="identity", width=0.5,fill = "#005c99")+my\_theme()+

xlab("Year")+ylab("Total sales (in Rs.)")+

ggtitle("Comparison between total sales per Year")

})

output$p51<-renderPlot({

ggplot(tsd4, aes(fill=Product, y=x, x=Year)) +

geom\_bar(position="dodge", stat="identity") +ggtitle("Comparison between Total Sales per product per year")+

ylab("Total Sales (in Rs.)")+xlab("Year")+

my\_theme()+

scale\_color\_manual(values = c("#ff4d4d", "#558000", "#33cc00","#0099cc","#24248f","#008080")) +

scale\_fill\_manual(values = c("#ff4d4d", "#558000", "#33cc00","#0099cc","#24248f","#008080"))

})

output$p7<-renderPlot({

ggplot(nSalesSum, aes(x=YEAR, y=x, color=Branch)) +xlab("Year")+ylab("Total Sales (in Rs.)")+

geom\_line(size=1.5)+geom\_point(size=3.5)+my\_theme()

})

output$p8<-renderPlot({

piepercent<-round(100\*(brAag$x)/sum(brAag$x),1)

pie(brAag$x,labels=piepercent,main='Product line At Branch A pie chart',col=rainbow(length(brAag$x)))

legend("topright",c("Electronic Accessories","Fashion Accessories","Food and beverages","Health and beauty","Home and lifestyle","Sports and Travel"),cex=0.8,fill=rainbow(length(brAag$x)))

})

output$p9<-renderPlot({

piepercent<-round(100\*(brBag$x)/sum(brBag$x),1)

pie(brBag$x,labels=piepercent,main='Product line At Branch B pie chart',col=rainbow(length(brBag$x)))

legend("topright",c("Electronic Accessories","Fashion Accessories","Food and beverages","Health and beauty","Home and lifestyle","Sports and Travel"),cex=0.8,fill=rainbow(length(brBag$x)))

})

output$p10<-renderPlot({

piepercent<-round(100\*(brCag$x)/sum(brCag$x),1)

pie(brCag$x,labels=piepercent,main='Product line At Branch C pie chart',col=rainbow(length(brCag$x)))

legend("topright",c("Electronic Accessories","Fashion Accessories","Food and beverages","Health and beauty","Home and lifestyle","Sports and Travel"),cex=0.8,fill=rainbow(length(brCag$x)))

})

output$p11<-renderPlot({

ggplot(grossdf, aes(fill=Branch, y=Total\_gross\_income, x=grossdf$Year)) +

my\_theme()+

scale\_color\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

scale\_fill\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

geom\_bar(position="dodge", stat="identity") +

xlab("Year")+ylab("Total Gross Income (in Rs.)")+

ggtitle("Comparison between total gross income per branch")

})

output$p12<-renderPlot({

ggplot(timedf3, aes(x=Time, y=x)) +

geom\_bar(stat = "identity",fill = "#00cc99")+

my\_theme()+

xlab("Time in Hour")+ ylab("Total Quantity")+

ggtitle("Quantity of products sold per hour")

})

output$p13<-renderPlot({

ggplot(timedf2, aes(fill=Branch, y=Quantity, x=Time)) + my\_theme()+

scale\_color\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

scale\_fill\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

geom\_bar(position="dodge", stat="identity") +

xlab("Time in Hour")+ylab("Total Quantity")+

geom\_bar(position="dodge", stat="identity") +ggtitle("Comparison between sales per hour in branches")

})

output$p14<-renderPlot({

ggplot(ratedf, aes(fill=Product, y=Rating, x=Branch)) +my\_theme()+xlab("Branch")+ylab("Ratings")+

scale\_color\_manual(values = c("#ff4d4d", "#558000", "#33cc00","#0099cc","#24248f","#008080")) +

scale\_fill\_manual(values = c("#ff4d4d", "#558000", "#33cc00","#0099cc","#24248f","#008080")) +

geom\_bar(position="dodge", stat="identity") +ggtitle("Comparison between ratings per branch per product")

})

output$p15<-renderPlot({

ggplot(nsales, aes(x=Rating)) +xlab("Ratings")+ylab("Count")+

geom\_histogram( binwidth=1, fill="#00cc99", color="white", alpha=0.9) +my\_theme()+

ggtitle("Histogram of Overall Ratings")

})

output$p16<-renderPlot({

piepercent<-round(100\*(pay1$freq)/sum(pay1$freq),1)

pie(pay1$freq,labels=piepercent,main='Payment',col=rainbow(length(pay1$freq)))

legend("topright",c("Cash","Credit card","Ewallet"),cex=0.8,fill=rainbow(length(paydf1$Np)))

})

output$p17<-renderPlot({

ggplot(paydf2, aes(fill=Payment, y=Sales, x=Branch)) +

geom\_bar(position="dodge", stat="identity") +my\_theme()+

scale\_color\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

scale\_fill\_manual(values = c("#739900", "#e6e600","#1f7a1f")) +

geom\_bar(position="dodge", stat="identity") +

xlab("Branch")+ylab("Count")+

ggtitle("Comparison between Payment options used per branch")

})

output$p18<-renderPlot({

ggplot(gen1, aes(fill=Gender, y=Total\_Quantity, x=Product)) + geom\_bar(position="dodge", stat="identity") +

my\_theme()+

scale\_color\_manual(values = c("#739900", "#e6e600")) +

scale\_fill\_manual(values = c("#739900", "#e6e600")) +

geom\_bar(position="dodge", stat="identity") +

xlab("Gender")+ylab("Total Quantity")+

ggtitle("Comparison between products based on gender")

})

output$ctp2<-renderPlot({

piepercent<-round(100\*(custA$x)/sum(custA$x),1)

pie(custA$x,labels=piepercent,main='Branch ',col=rainbow(length(custA$x))) legend("topright",c("Member","Normal"),cex=0.9,fill=rainbow(length(custA$x)))

})

output$ctp3<-renderPlot({

piepercent<-round(100\*(custB$x)/sum(custB$x),1)

pie(custB$x,labels=piepercent,main='Branch ',col=rainbow(length(custB$x))) legend("topright",c("Member","Normal"),cex=0.9,fill=rainbow(length(custB$x)))

})

output$ctp4<-renderPlot({

piepercent<-round(100\*(custC$x)/sum(custC$x),1) pie(custC$x,labels=piepercent,main='Branch C',col=rainbow(length(custC$x))) legend("topright",c("Member","Normal"),cex=0.9,fill=rainbow(length(custC$x)))

})

output$p22<-renderValueBox({

valueBox(value = "",

subtitle = "Time series interactive plot of Total sales",

color = "purple",

width = 12)

})

output$tsplot<-renderDygraph({

dygraph(Final\_time\_series)%>% dyRangeSelector()%>%dySeries(label = "Total sales",color = "#cc0099") %>%

dyOptions(stackedGraph = TRUE)%>%dyAxis("x",drawGrid = FALSE)%>%dyAxis("y",drawGrid = FALSE)

})

output$o.traintest.box <- renderValueBox({

valueBox(value = "Spliting the dataset",

subtitle = "",

color = "navy",

width = 3)

})

output$o.train.box.view <- renderValueBox({

valueBox(value = "Training dataset",

subtitle = "",

color = "yellow",

width = 3)

})

output$o.test.box.view <- renderValueBox({

valueBox(value = "Testing dataset",

subtitle = "",

color = "green",

width = 3)

})

output$o.training.data.view <- DT::renderDataTable({

DT:: datatable(round(Training\_dataset,2),

options = list(pageLength = 10))

})

output$o.testing.data.view <-DT::renderDataTable({

DT::datatable(round(Testing\_dataset,2),options = list(pageLength = 10))

})

output$o.print.decomp <- renderUI({

helpText(h3("Decomposition of time series",

style = "color: #006666"))

})

output$o.add.formula <- renderUI({

withMathJax(

helpText(h3('Additive model: $$y\_t = Seasonality\_t + Trend\_t + Random\_t$$',

style = "color: #4d4d00"))

)

})

output$o.mul.formula <- renderUI({

withMathJax(

helpText(h3('Multiplicative model: $$y\_t = Seasonality\_t \\times Trend\_t \\times Random\_t$$',

style = "color: #000080"))

)

})

output$o.decomp.plot.additive <- renderPlot({

plot(decompose(Final\_time\_series,

type = c("additive")),

col = "#ff6600")

})

output$o.print.seasonality <- renderUI({

helpText(h3("Visualizing seasonality plot",style = "color:#006666"))

})

output$o.seasonplot <- renderPlot({

seasonplot(Final\_time\_series,

s = 12,

main = "Seasonality plot",

ylab = "Total sales",

xlab = "Months",

col = rainbow((dim(sn4)[1])/12))

})

output$o.monthplot <- renderPlot({

monthplot(ntsf,

main = "Monthly plot",

ylab = "Total sales",

xlab = "Months")

})

output$o.print.etsparameters <- renderUI({

helpText(h2("Summary of Exponential Smoothing model",

style = "color: #006666"))

})

output$o.print.model <- renderPrint({

summary(ETS\_Model) })

output$o.dy.pred.on.test <- renderDygraph({

df <- cbind(etsmean,

Testing\_dataset$`Total Sales`)

dygraph(df,

main = "Interactive plot for comparison of Predicted with Actual values")%>%

dyRangeSelector()%>%

dySeries("etsmean",

label = "Predicted values",

color = "red") %>%

dySeries("Testing\_dataset$`Total Sales`",

label = "Actual values",

color = "black")%>%

dyAxis("x", drawGrid = FALSE, label = "Time")%>%

dyAxis("y", drawGrid = FALSE, label = "Total sales") %>%

dyOptions(drawPoints = TRUE,

pointSize = 3)

})

output$etsdf<-DT::renderDataTable(etsd,options=list(paging=FALSE,searching=FALSE))

output$printactuals <- renderUI({

helpText(h2("Actual and Forecasted sales",

style = "color: #006666"))

})

output$o.print.arimaparameters <- renderUI({

helpText(h2("Summary of ARIMA model",

style = "color: #006666"))

})

output$o.print.arimamodel <- renderPrint({

summary(ARIMA\_Model\_1)

})

output$o.dy.pred.on.arimatest <- renderDygraph({

df2 <- cbind(arimamean,Testing\_dataset$`Total Sales`)

dygraph(df2,

main = "Interactive plot for comparison of Predicted with Actual values")%>%

dyRangeSelector()%>%

dySeries("arimamean",

label = "Predicted values",

color = "red") %>%

dySeries("Testing\_dataset$`Total Sales`",

label = "Actual values",

color = "black")%>%

dyAxis("x", drawGrid = FALSE, label = "Time")%>%

dyAxis("y", drawGrid = FALSE, label = "Total sales") %>%

dyOptions(drawPoints = TRUE,

pointSize = 3)

})

output$arimadf<-DT::renderDataTable(arimad,options=list(paging=FALSE,searching=FALSE))

output$printaractuals <- renderUI({

helpText(h2("Actual and Forecsted sales",

style = "color: #006666"))

})

output$printetsaccuracytext<-renderUI({

helpText(h3("Accuracy of Exponential Smoothing model",

style = "color: #006666"))

})

output$printetsmodel<-renderPrint({

accuracy(ETS\_Forecast,Testing\_dataset$`Total Sales`)

})

output$printarimaaccuracytext<-renderUI({

helpText(h3("Accuracy of ARIMA model",

style = "color: #006666"))

})

output$printarimamodel<-renderPrint({

accuracy(ARIMA\_Forecast,Testing\_dataset$`Total Sales`)

})

output$printetsplot<-renderUI({

helpText(h3("ETS plot",

style = "color: #006666"))

})

output$printarimaplot<-renderUI({

helpText(h3("ARIMA plot",

style = "color: #006666"))

})

output$printahead<-renderUI({

helpText(h3("Select the number of months to forecast ahead",

style = "color: #8B008B"))

})

output$plotftitle<-renderUI({

helpText(h2("Forecasting using ETS model",

style = "color: #8B008B;font-size:35px;font-style:bold;"))

})

output$printfinalsummary<-renderUI({

runif(etsmodelbuild())

helpText(h3("Model summary",

style = "color: #006666;"))

})

randomVals <- eventReactive(input$forecast.click, {

runif(input$no.days) && runif(input$select)

input$no.days

})

SalesAt<- eventReactive(input$forecast.click,{

sat<-input$select

sat

})

etsmodelbuild<-reactive({

if(SalesAt()==1){

model\_build <- ets(Final\_time\_series)

}

if(SalesAt()==2){

model\_build <- ets(BranchA\_time\_series)

}

if(SalesAt()==3){

model\_build <- ets(BranchB\_time\_series)

}

if(SalesAt()==4){

model\_build <- ets(BranchC\_time\_series)

}

model\_build

})

output$printfinalsum<-renderPrint({

runif(etsmodelbuild())

summary(etsmodelbuild())

})

etstrail<- reactive({

if(SalesAt()==1){

fore <- forecast(etsmodelbuild(),h = as.numeric(randomVals()))

}

if(SalesAt()==2){

fore <- forecast(etsmodelbuild(),h = as.numeric(randomVals()))

}

if(SalesAt()==3){

fore <- forecast(etsmodelbuild(),h = as.numeric(randomVals()))

}

if(SalesAt()==4){

fore <- forecast(etsmodelbuild(),h = as.numeric(randomVals()))

}

fore

})

edf<-reactive({

odf<-data.frame(etstrail())

odf<-round(odf,2)

colnames(odf)<-c("Forecasted sales","Lo 80%","Hi 80%","Lo 95%","Hi 95%")

odf

})

output$plotftextt<-renderUI({

runif(etstrail())

helpText(h3("Forecast plot",

style = "color: #006666"))

})

output$plotdftextt<-renderUI({

runif(edf())

helpText(h3("Predicted sales for year 2015",

style = "color: #006666"))

})

output$of<-renderPlot(

plot(etstrail())

)

output$ofdf<-DT::renderDataTable({

DT:: datatable(edf(),

options = list(pageLength = 12,searching=FALSE))

})

})