**S.P. Mandali’s**

Ramnarain Ruia Autonomous College

Matunga, Mumbai-400019

Mumbai University

**Project Report**

On

“ANALYSIS ON DOG CARE”

**Under the Guidance of**

Prof. Rasika Mundhe

**Project By**

PRASAD SHANKAR YADAV

MSc COMPUTER SCIENCE PART-II

Department of Computer Science

2019-20

DECLARATION

I the undersigned, student of **M.Sc. COMPUTER SCIENCE PART II** hereby declare that I have attempted to complete the project on “**ANALYSIS OF DOGCARE**”. The information submitted is true and original to the best of Knowledge.

Signature of the student

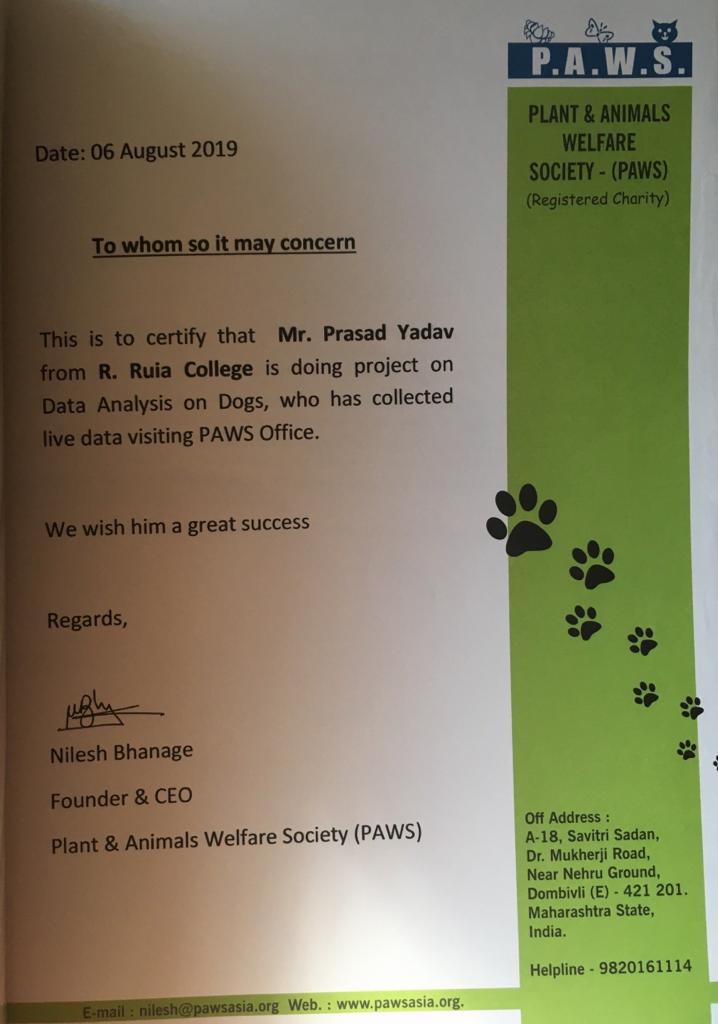
                                                             Name of the Student

Prasad Shankar Yadav

Acknowledgement

* I, Mr. Prasad Shankar Yadav, student of Ramnarain Ruia Autonomous College, acknowledge my sincere gratitude to all those who helped me in the project work of MSc. Computer Science.
* I extend my gratitude towards my project guide, Prof Rasika Mundhe to help me to go ahead with my project and appreciating my work at every stage.

Prasad Shankar Yadav



Contents

[Title and Abstract 6](file:/C:/Users/acer/Downloads/project%20doc%20(1).docx#_Toc19878442)

[Background and literature 7](file:/C:/Users/acer/Downloads/project%20doc%20(1).docx#_Toc19878443)

[Project and Scope. 8](file:/C:/Users/acer/Downloads/project%20doc%20(1).docx#_Toc19878444)

[Results and Interpretation 9](file:/C:/Users/acer/Downloads/project%20doc%20(1).docx#_Toc19878445)

[Conclusions: - 21](file:/C:/Users/acer/Downloads/project%20doc%20(1).docx#_Toc19878446)

[Future Scope and Development 22](file:/C:/Users/acer/Downloads/project%20doc%20(1).docx#_Toc19878447)

[Reference and appendix: - 45](file:/C:/Users/acer/Downloads/project%20doc%20(1).docx#_Toc19878448)

**Title and Abstract**

**ANALYSIS OF DOG CARE**

The main aim of this project is to provide a deep analysis on live data generated by the NGO’s. In dog care, a large number of structured and unstructured data is generated by NGO’s. The big data analytical platform is one of the best ways to analyze the data of all type generated by the NGO”s and perform different statistical operations on them to get desired output.

This project has used the data generated from Plant and Animal welfare society. This project has applied some of the models on data to improve the conditions for dogs in many ways.

The project contains an interesting demographic representation of the data, as well as some features that can improve the medical as well as hygienic conditions of dogs. This project also contains the trends of the diseases, adoptions, sex ratio seen in past years and applying prediction models on this data to predict the future conditions.

**Background and literature**

Data analytics can help the dog care system in many ways. For any product developed in dog care field requires both NGO's and doctor’s approach as they are dependent on each other. The strategy behind doing analysis on the data from hospital in dog care is to improve the conditions in hospital for the benefits of the dogs as well as doctors. Predictive analytics is the process of analyzing the data using automated statistical processes and summarizing results into useful information. The information acquired from the predictive analysis can be very useful to the doctors. This can be helpful in decision making for the NGO’s for the upcoming challenges.

**Project and Scope.**

Respect for animal welfare is often based on the belief that nonhuman animals are sentient and that consideration should be given to their well-being or suffering, especially when they are under the care of humans. These concerns can include how animals are slaughtered for food, how they are used in scientific research, how they are kept (as pets, in zoos, farms, circuses, etc.), and how human activities affect the welfare and survival of wild species. With the R tool for visualization it is easy to understand and interpret the results of dataset produced by applying various techniques on it. R performs wide variety of functions, such as data manipulation, statistical modeling in graphics.

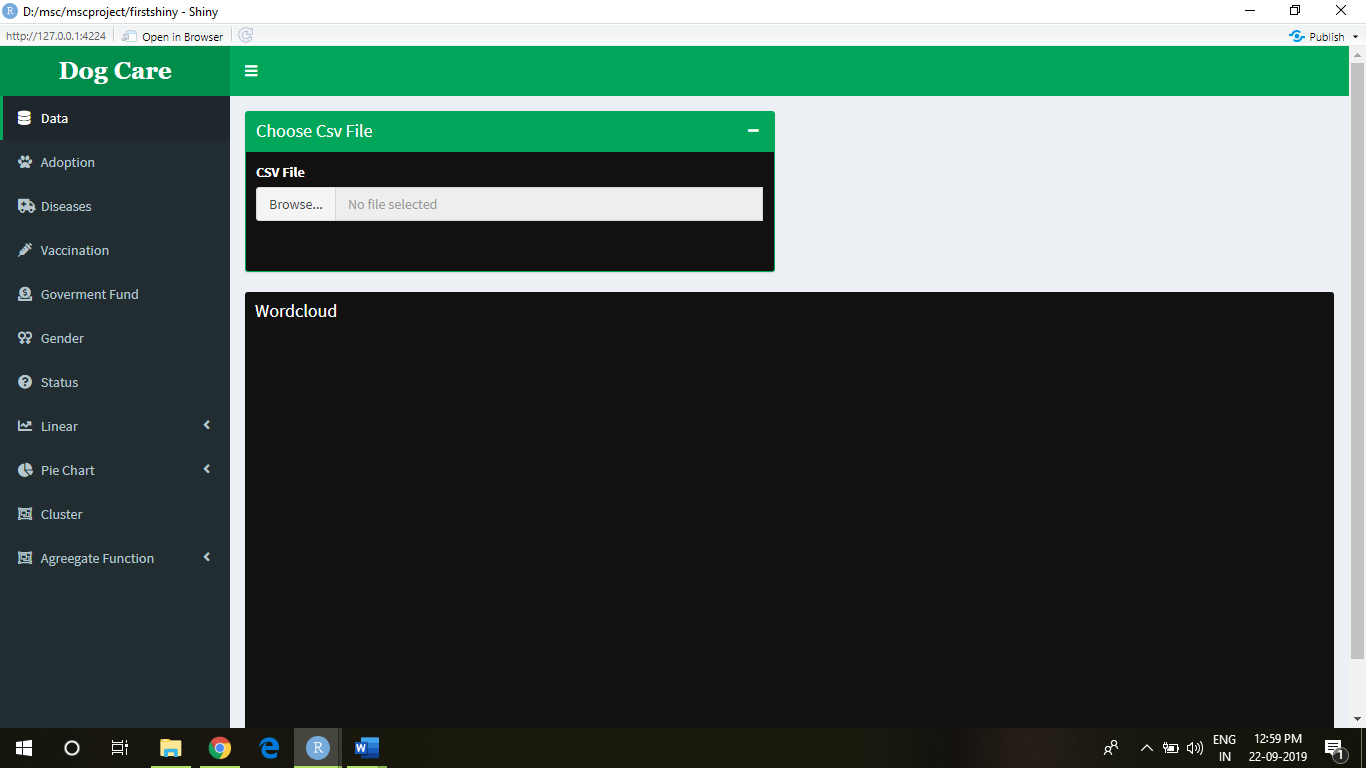
Data gathered for this project is collected from Plant and animal welfare society, Dombivli. It is one of the NGO'S that is in forefront working for the welfare of dogs and other animals. In this institution, the government funding is not available so they arrange various programs in order to get donations. The donation received it used for betterment of dogs i.e vaccination, food, shelter etc. Hence this project can be considered a real environment-based analysis.

With the help of predictive modeling technique like Linear Regression, it is easy and significantly efficient to understand the status of dog and puppies based on the their death rate etc. This will help the government to maintain their records. Clustering helps to relate observations in the same group to be similar and observations in different groups to be dissimilar.

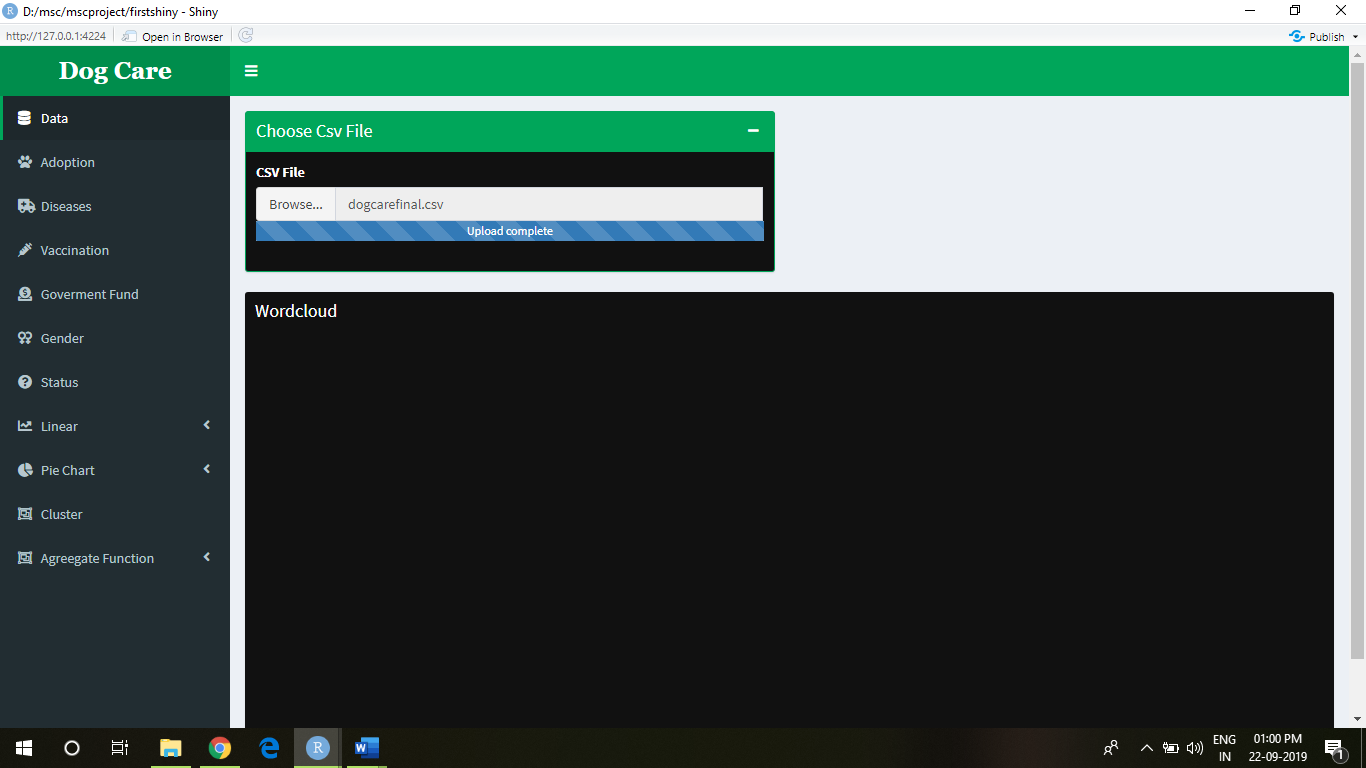
With aggregation methods, it is easy to understand the average number of dogs that suffer from particular types of disease based on their age (whether a dog or a puppy). Using this we can also understand the average deaths that occur according to the disease. Using the method, it is easy to interpret the total number of deaths of dogs and puppies in a year. It also makes understanding the trend of diseases on a monthly basis.

**Results and Interpretation**

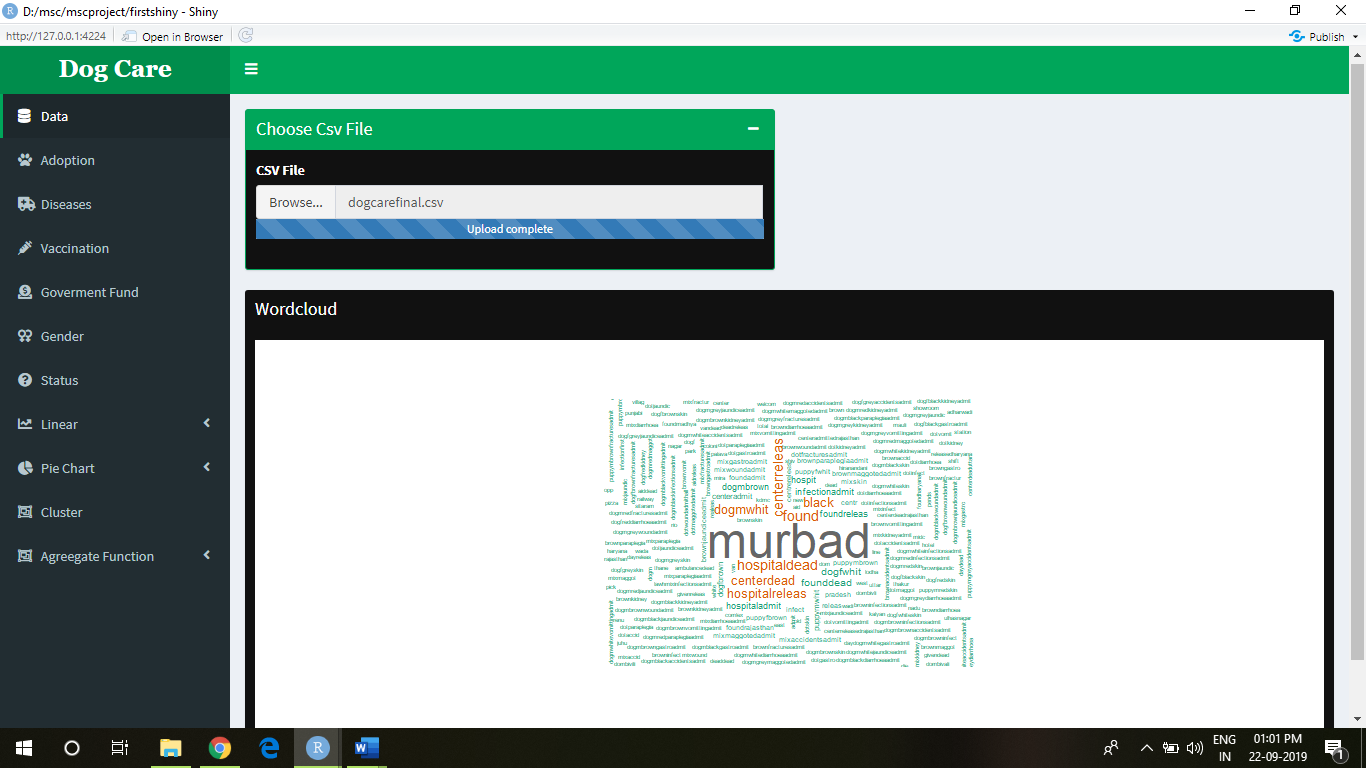
Dashboard



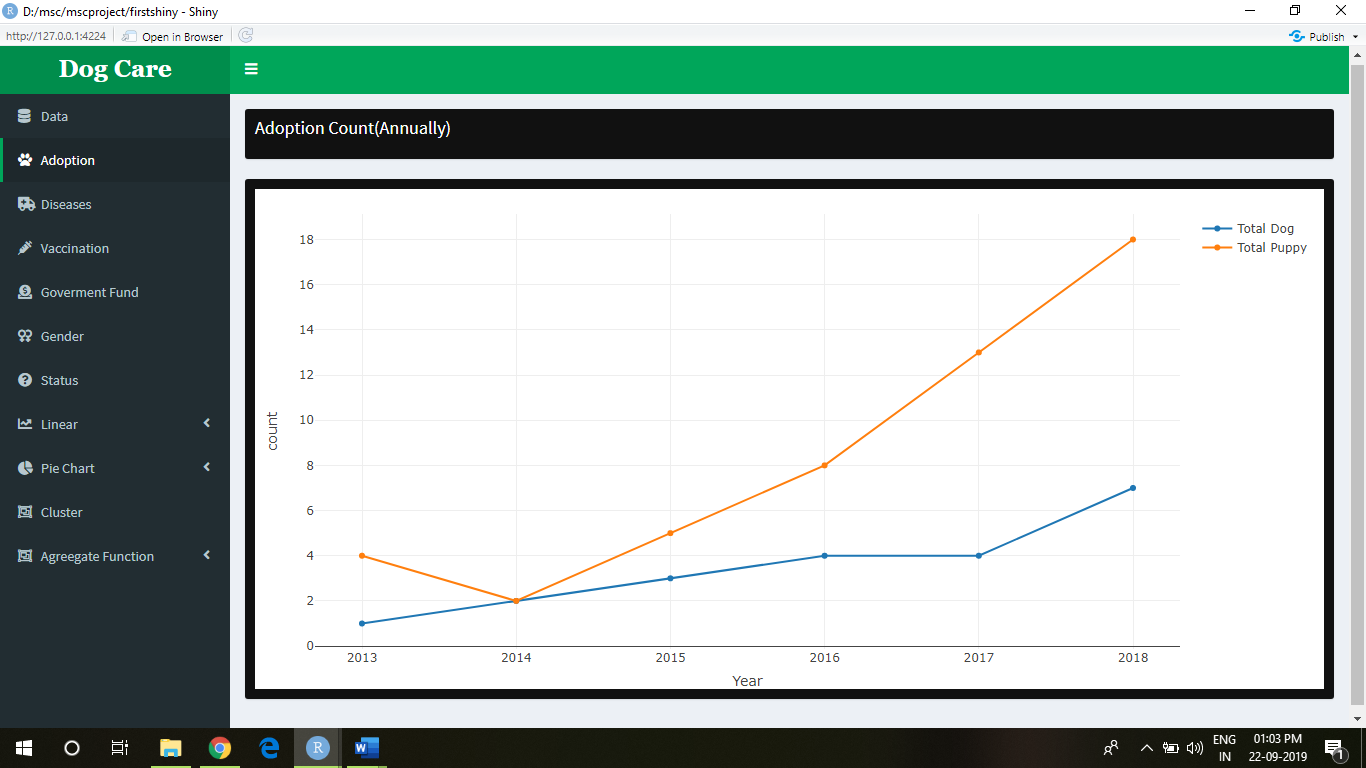
Browse File



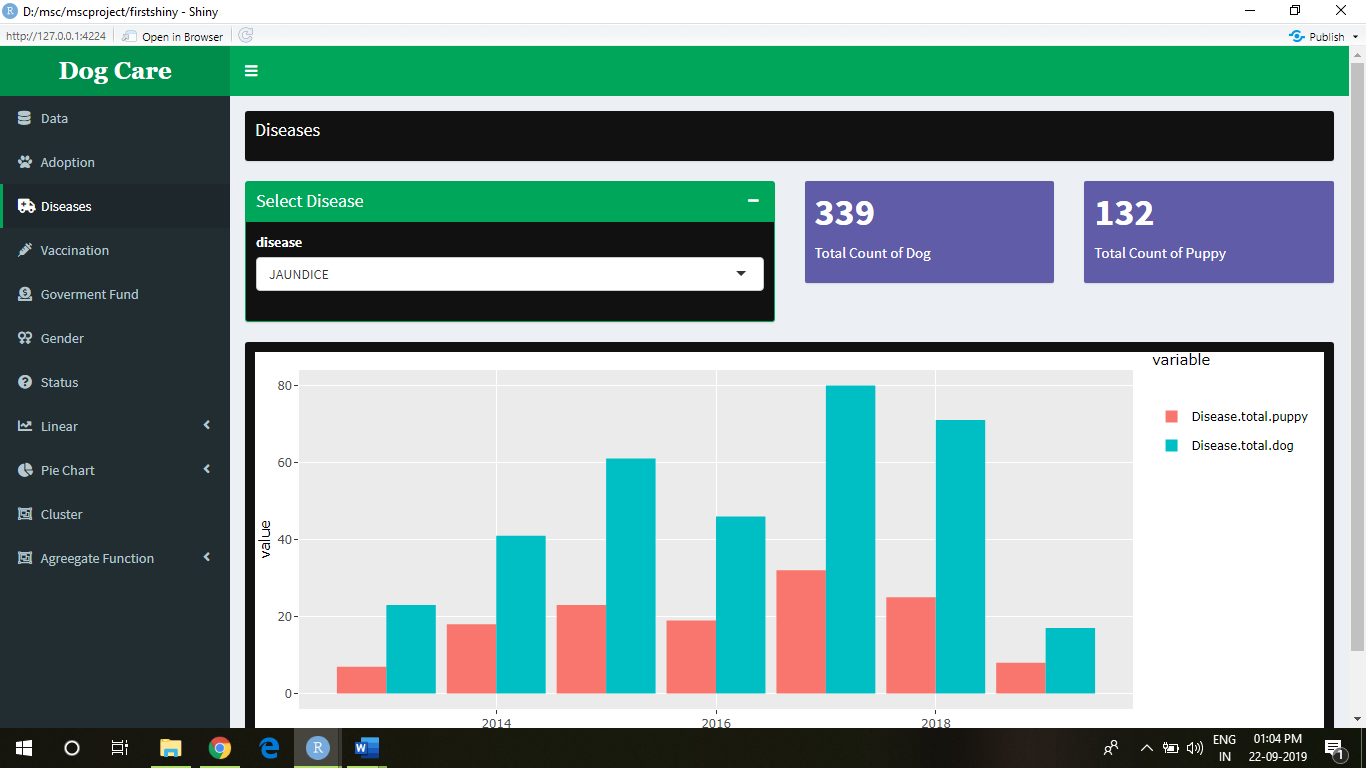
1. How to understand the title analysis of dog care using word cloud?



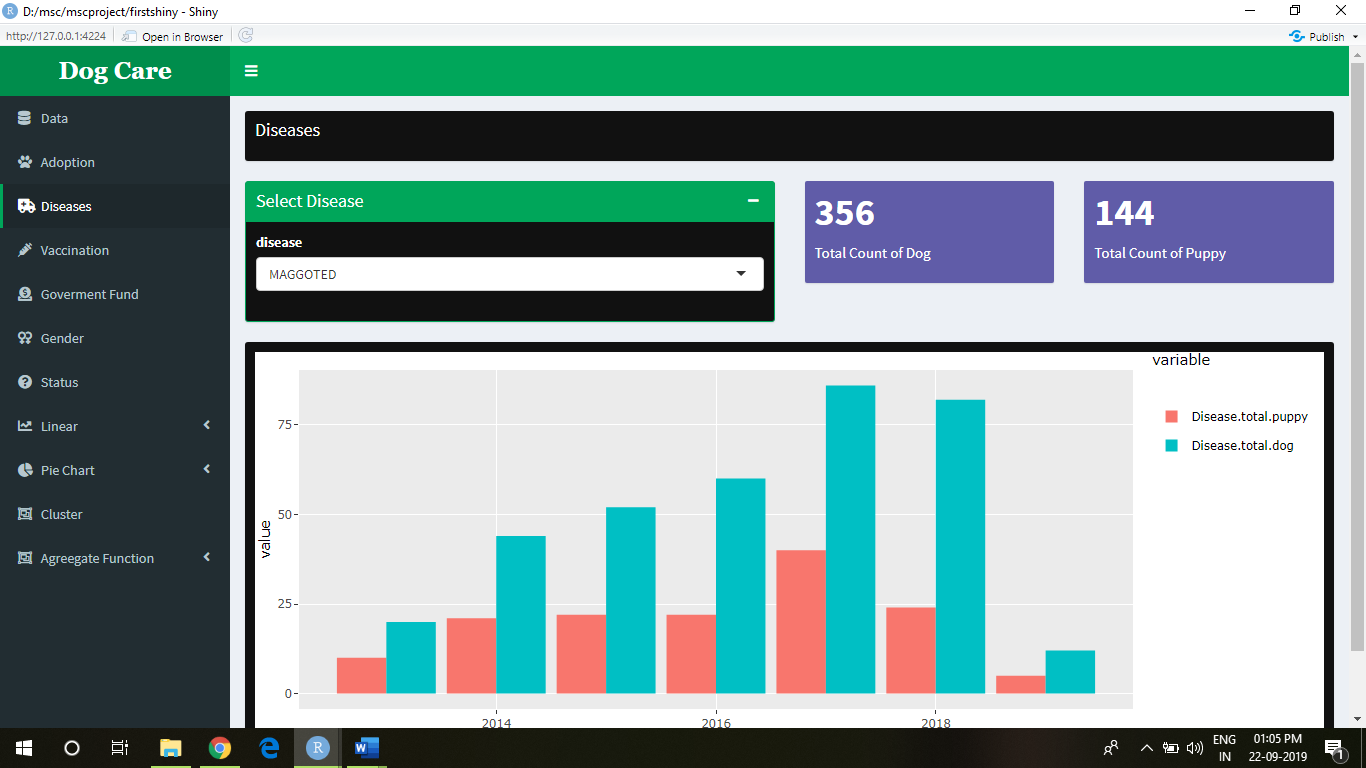
1. How many dogs and puppies are adopted per year?



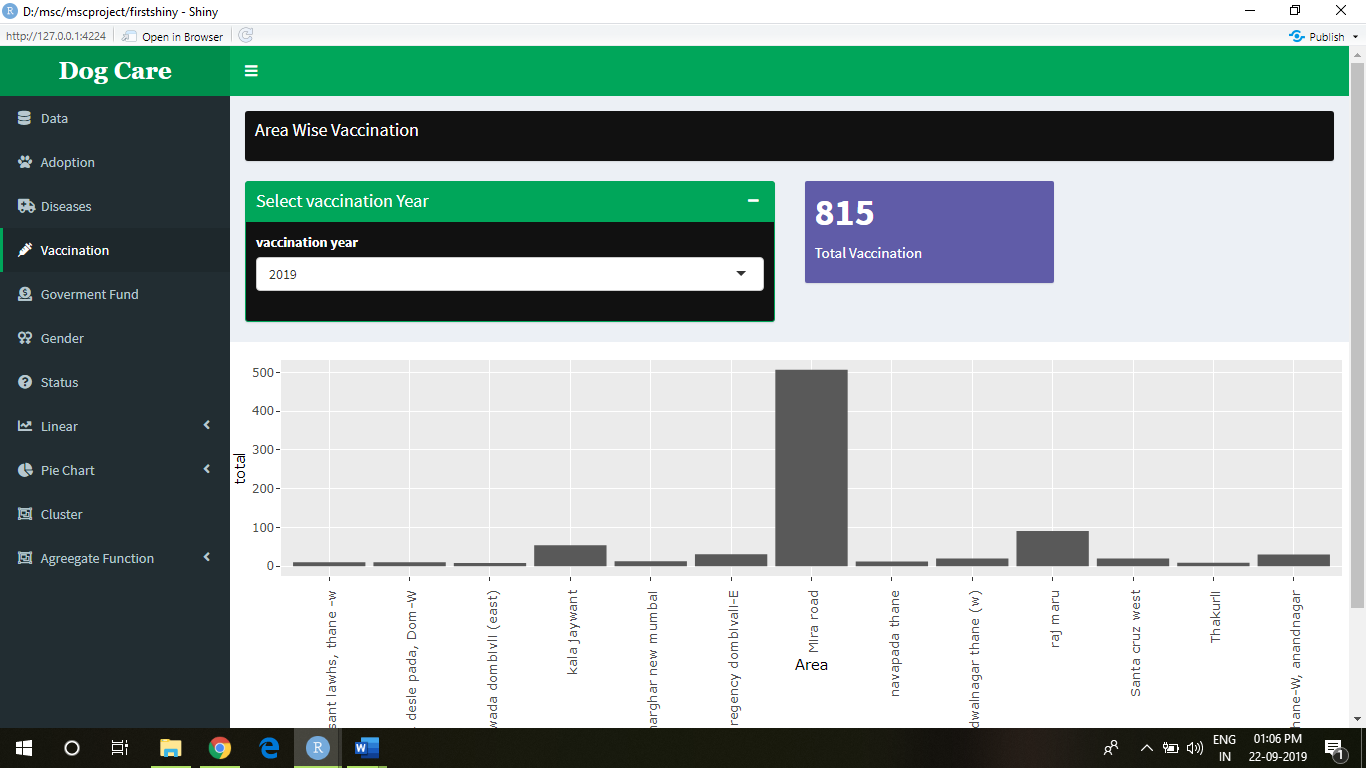
1. How many dogs and puppies are suffered from Jaundice?



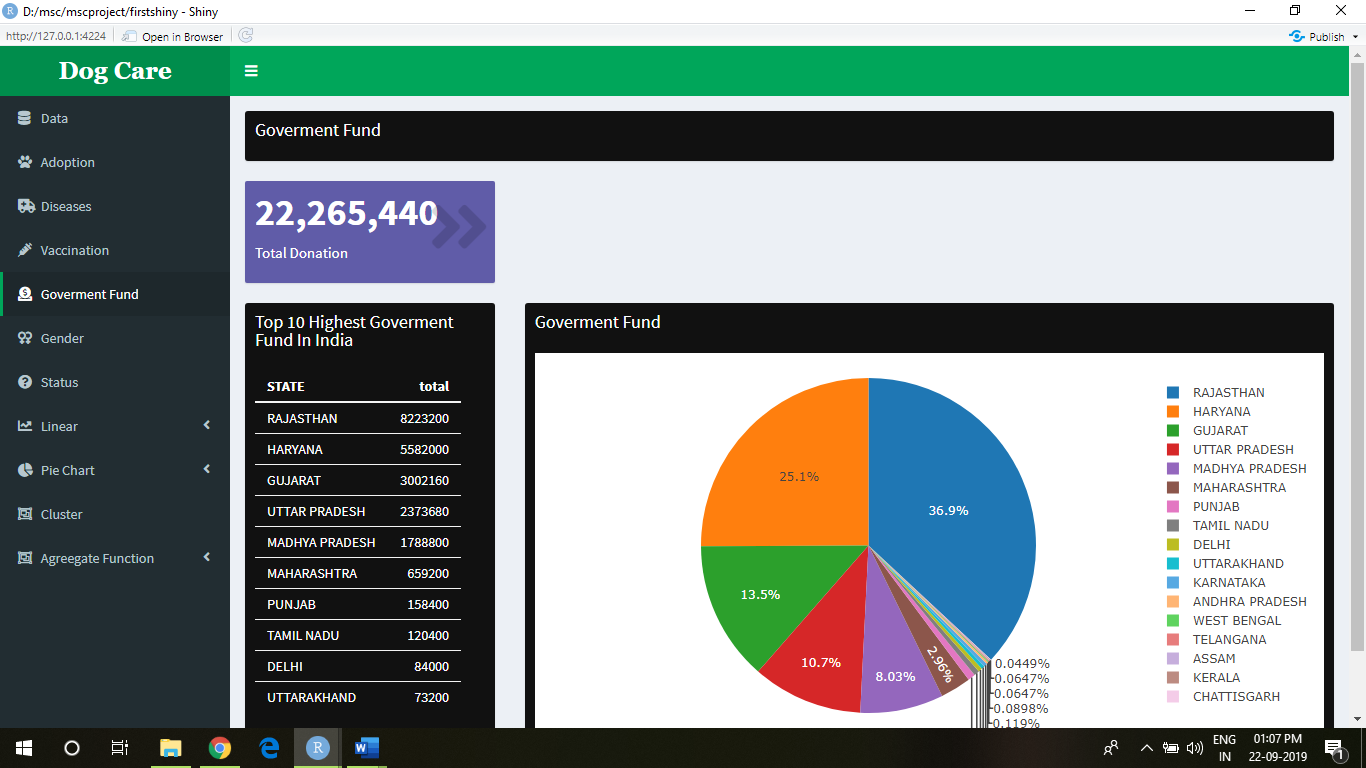
1. How many dogs and puppies were infected with Maggots?



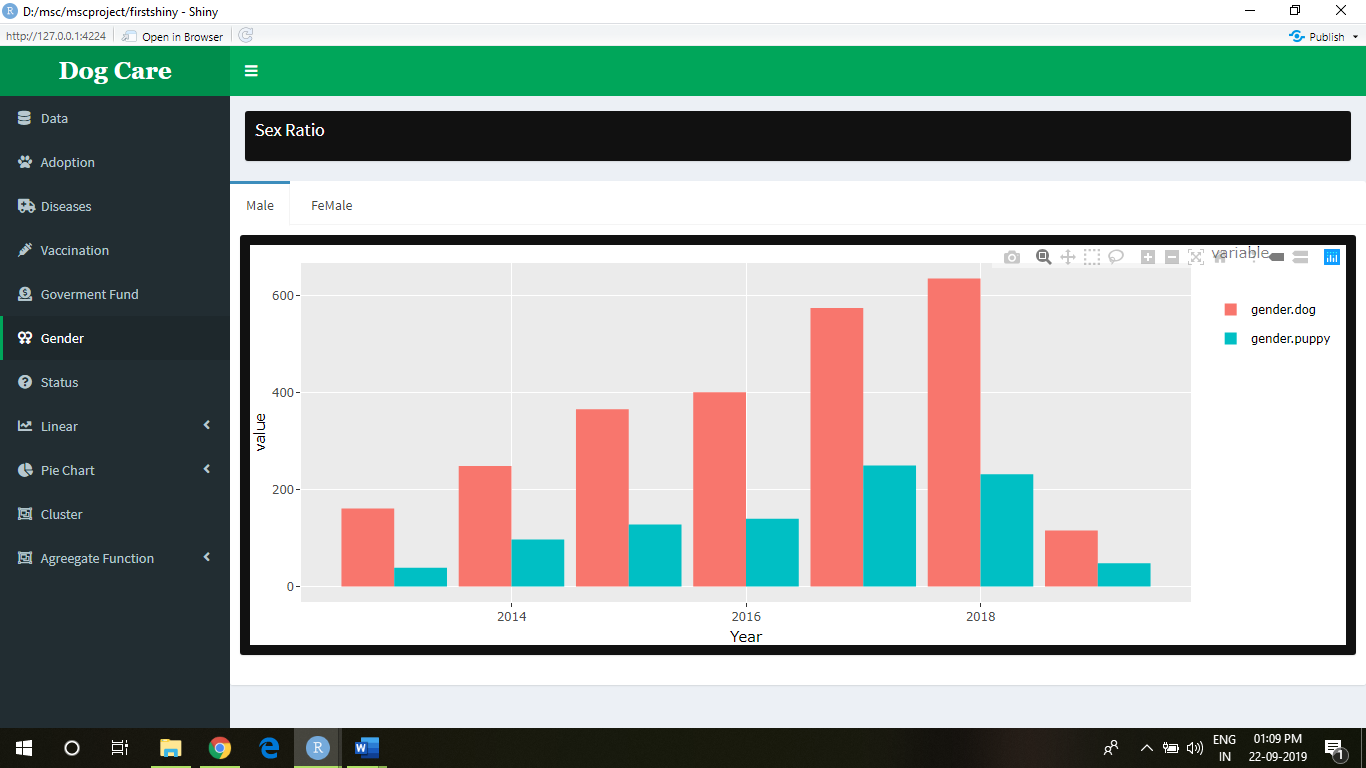
1. In which area the vaccination rate is high?



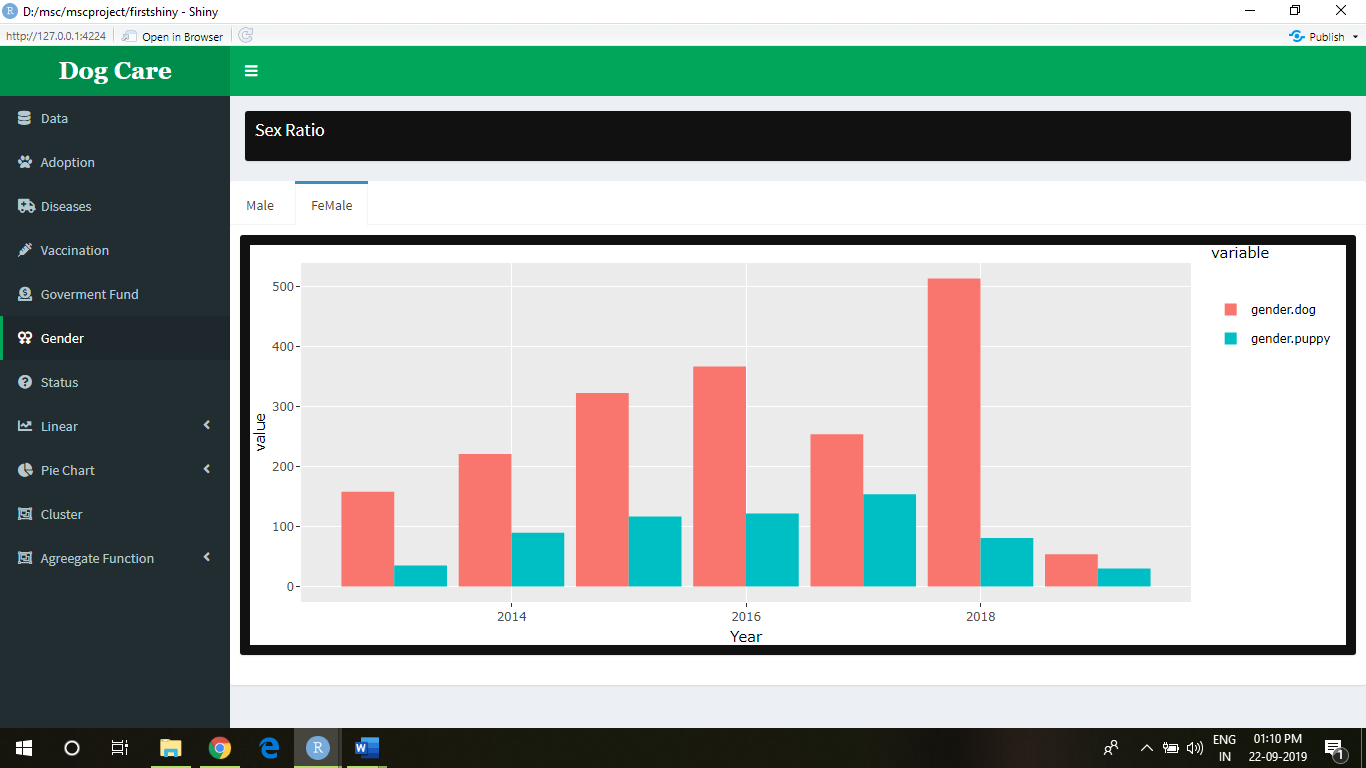
1. How much government funds are used to help animals?



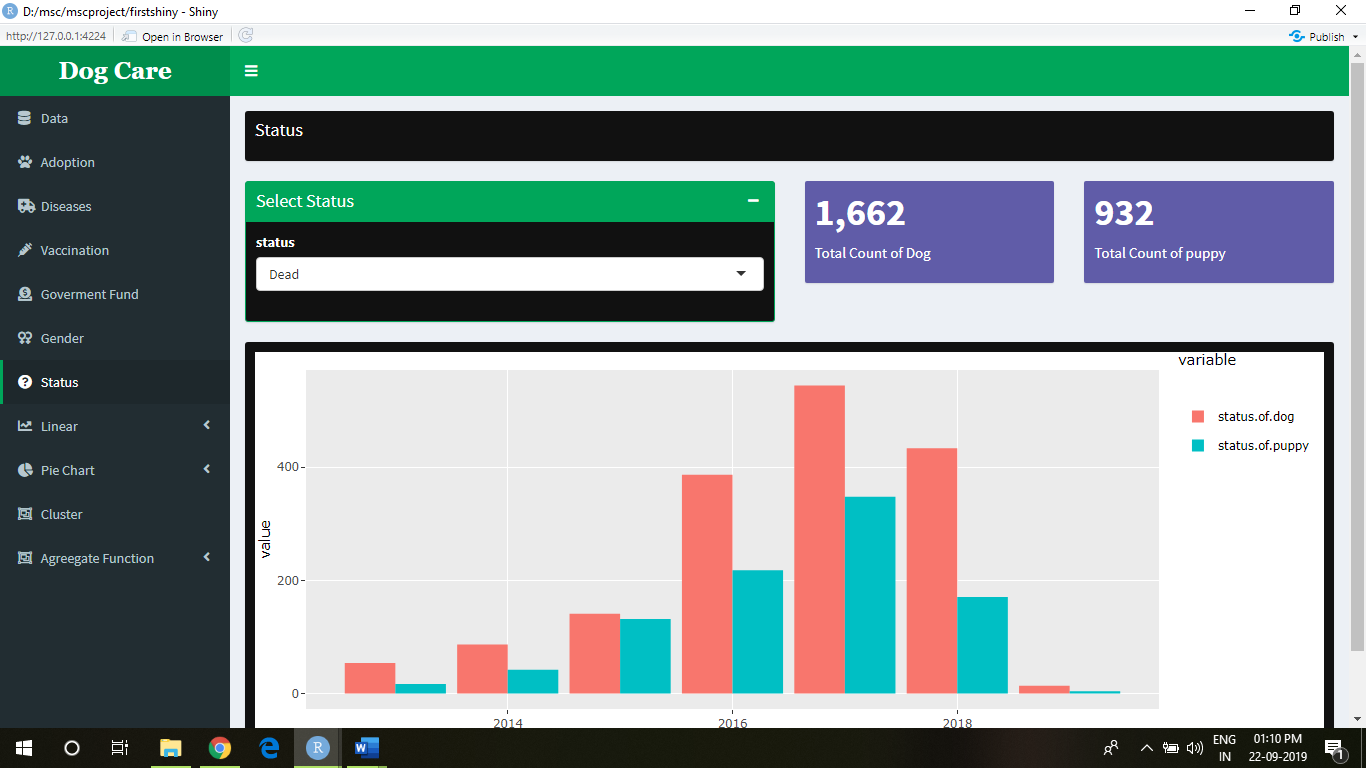
1. What is sex ratio of male?



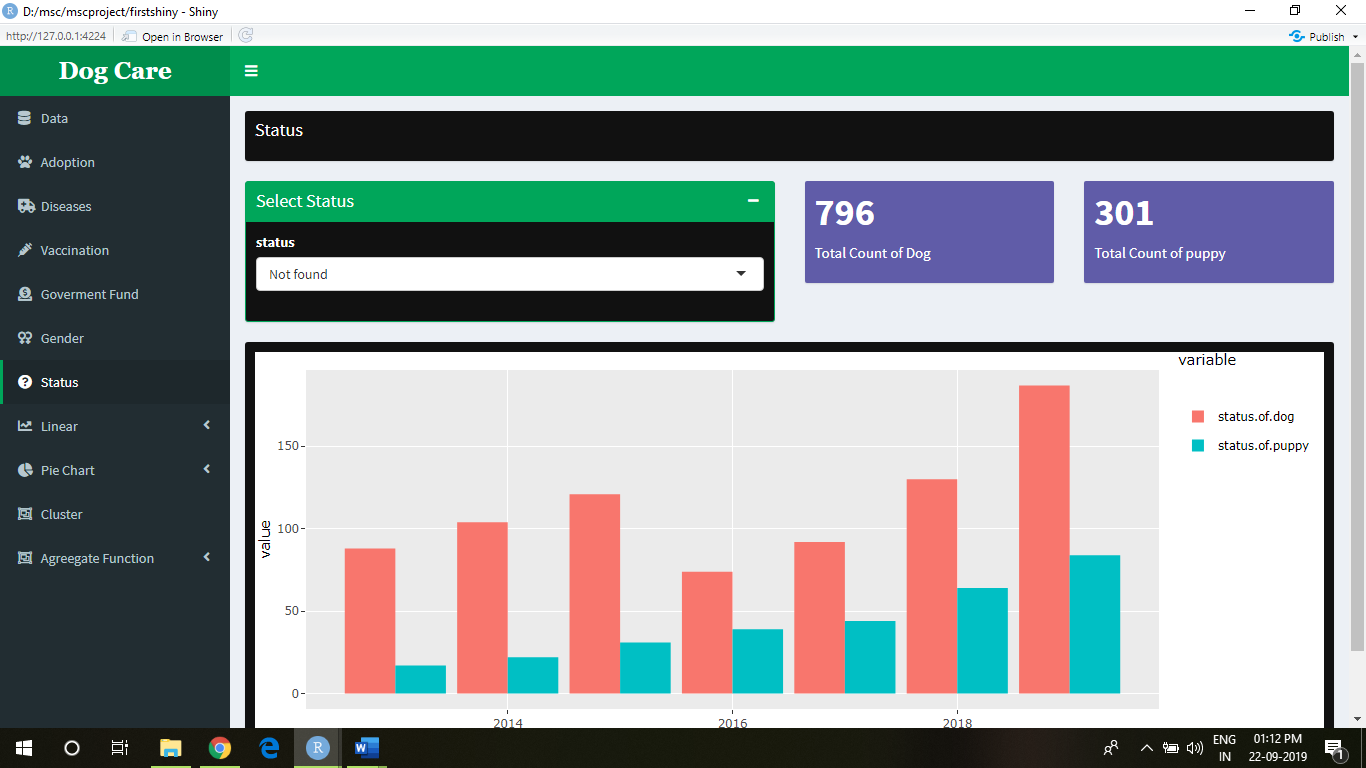
1. What is sex ratio of female?



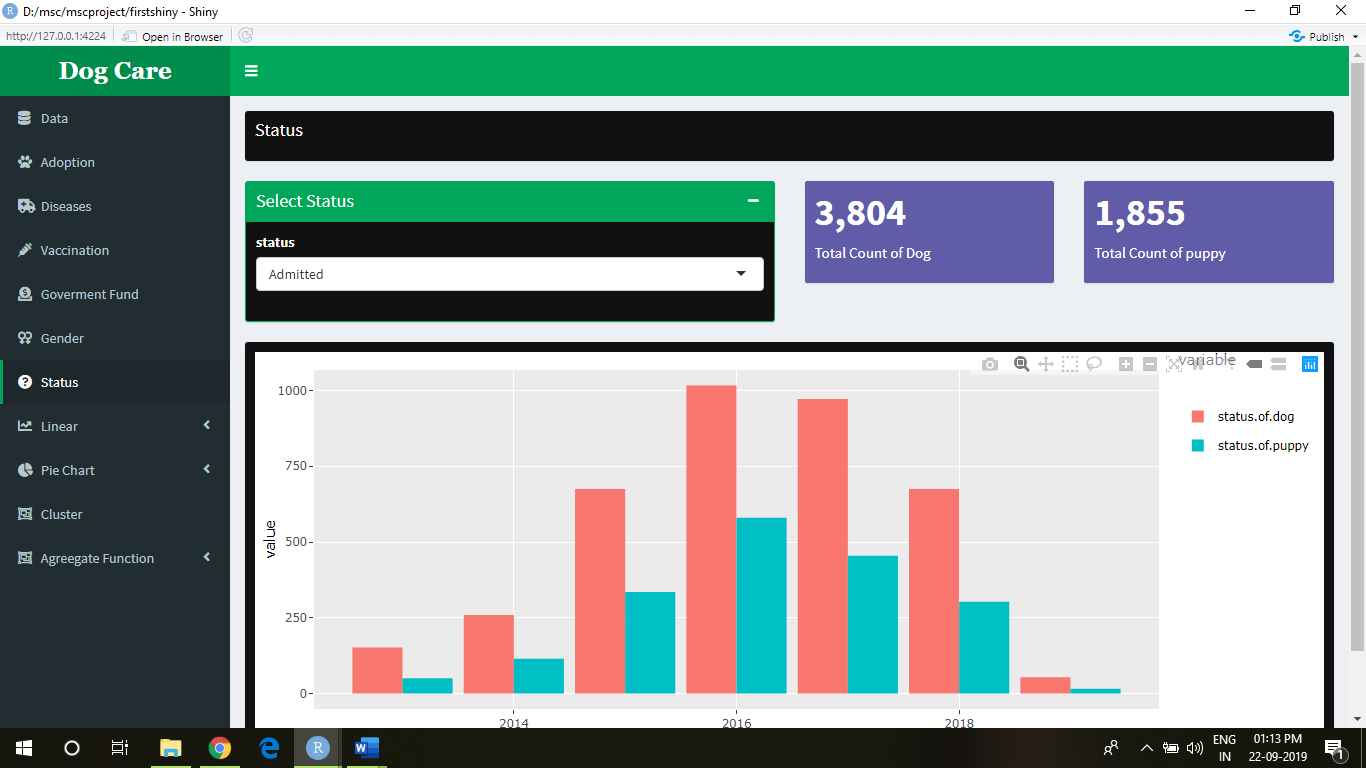
1. How many dogs and puppies have died?



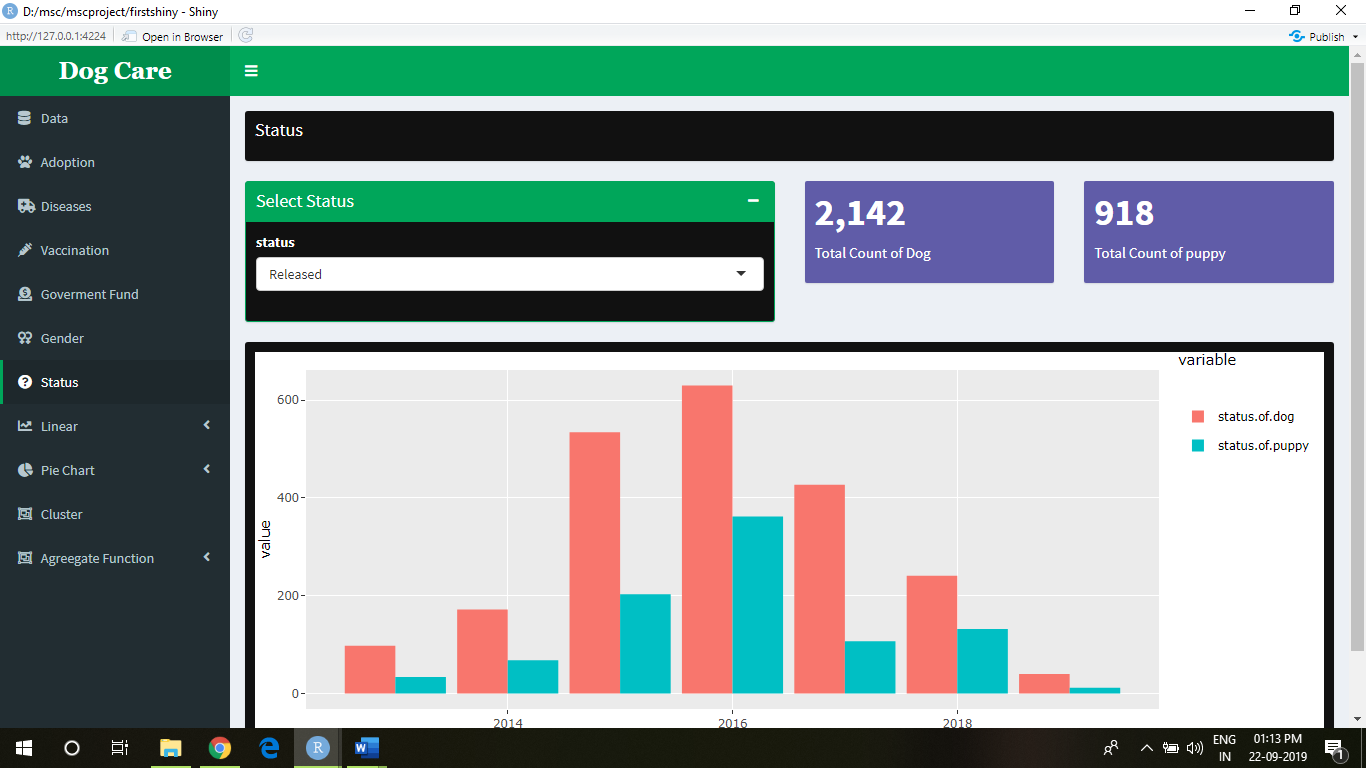
1. How many dogs and puppies are not found?



1. How many dogs and puppies are admitted?



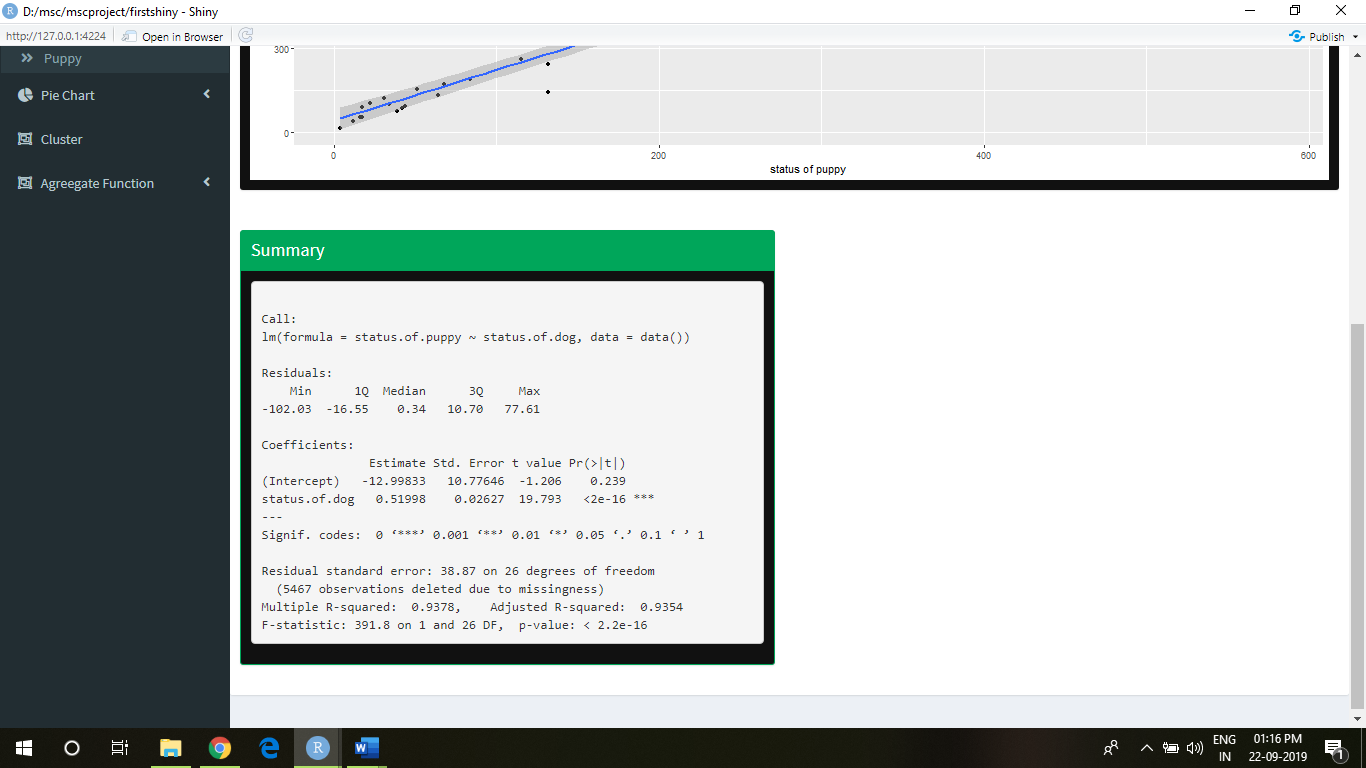
1. How many dogs and puppies are released?



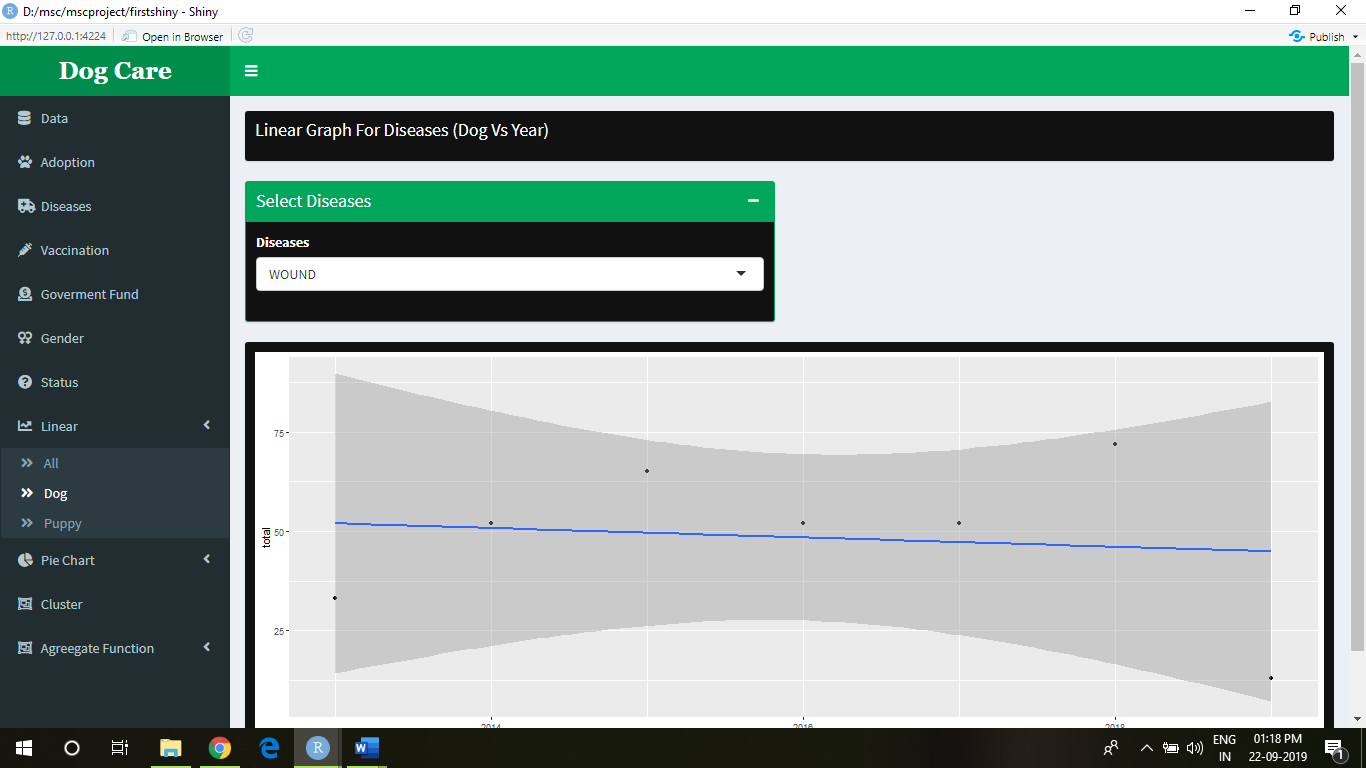
1. Using Regression model, how to summarise the status of dog and puppy in a year.



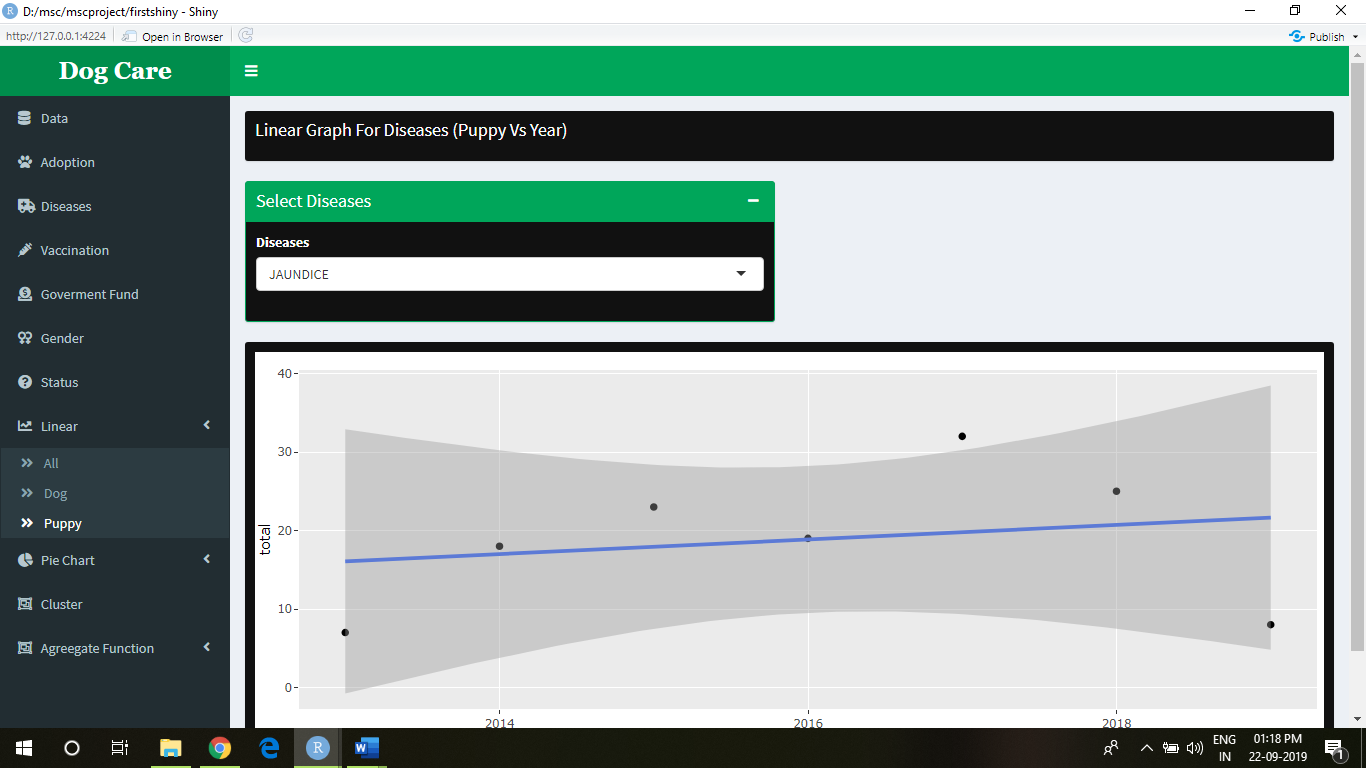
Summary



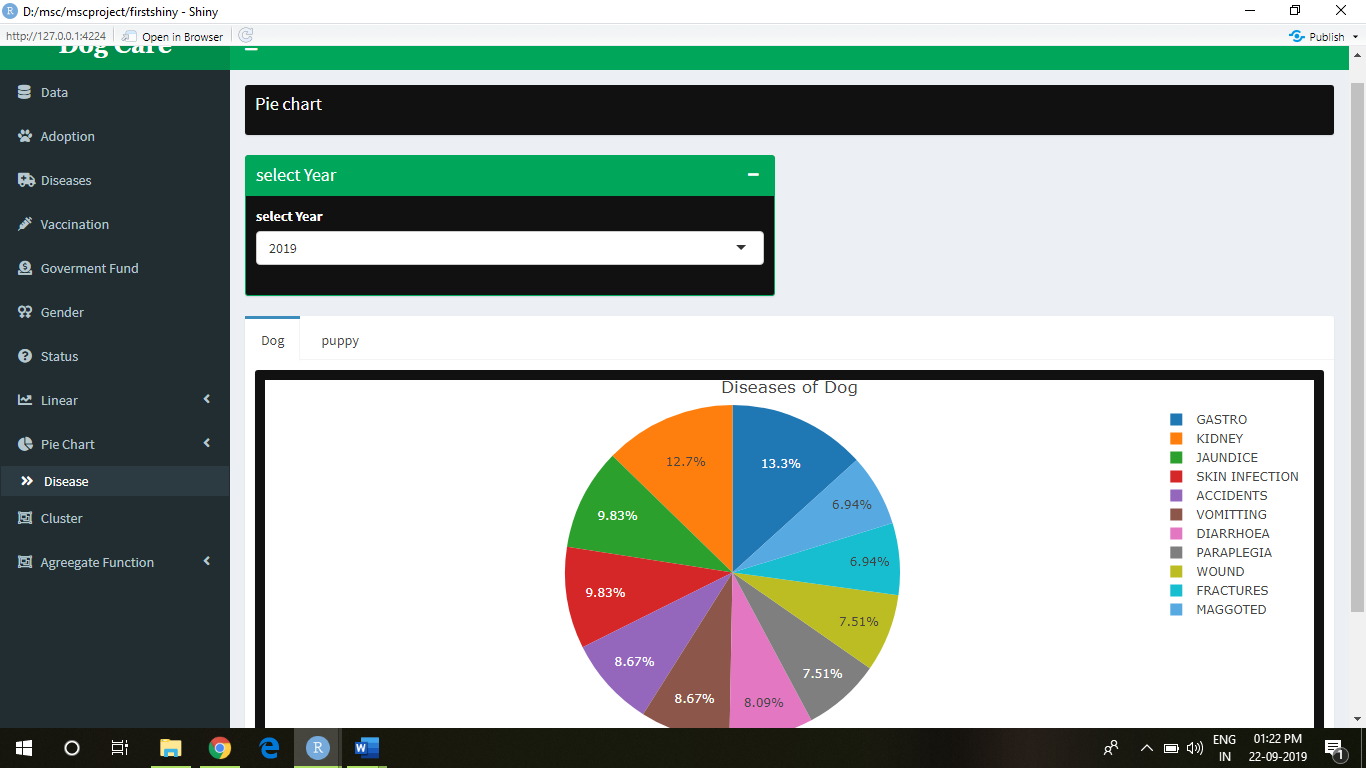
1. The count of dogs that are wounded



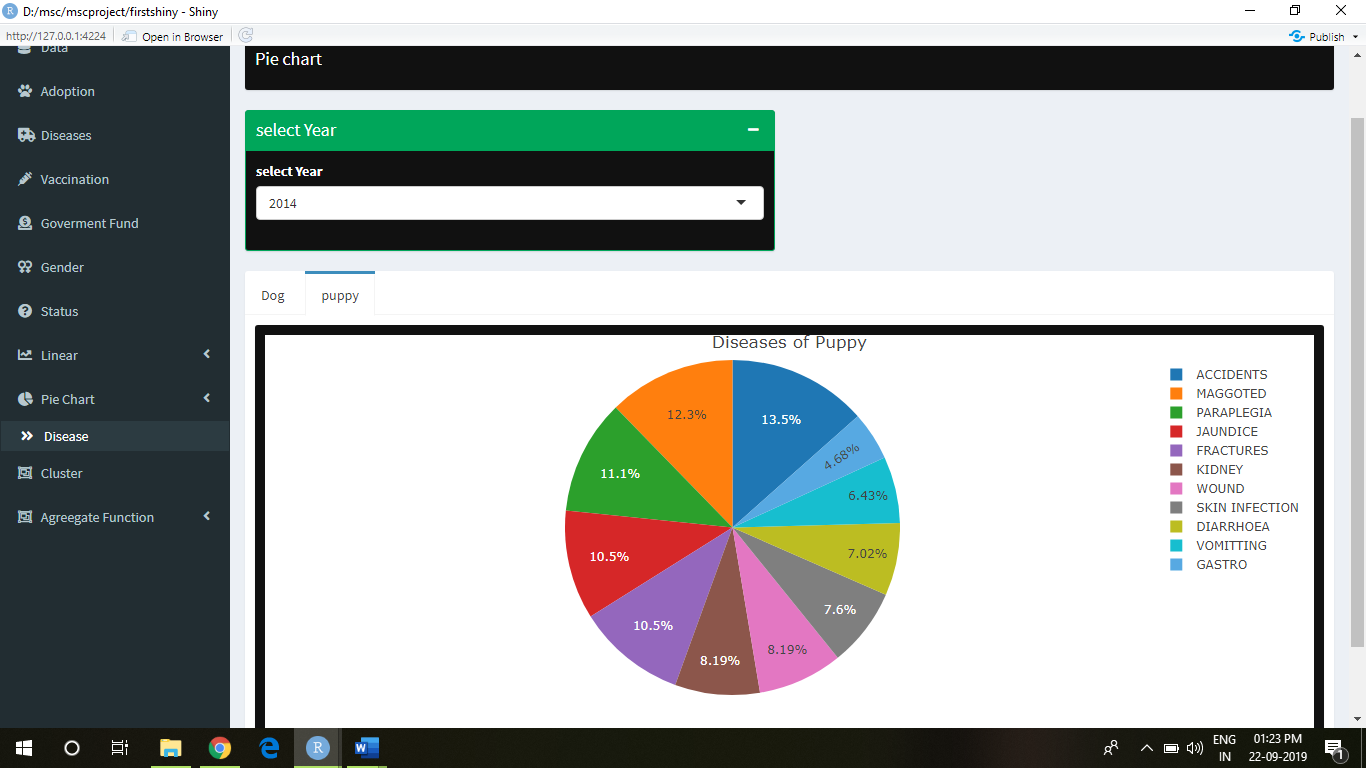
2.The count of puppies having jaundice.



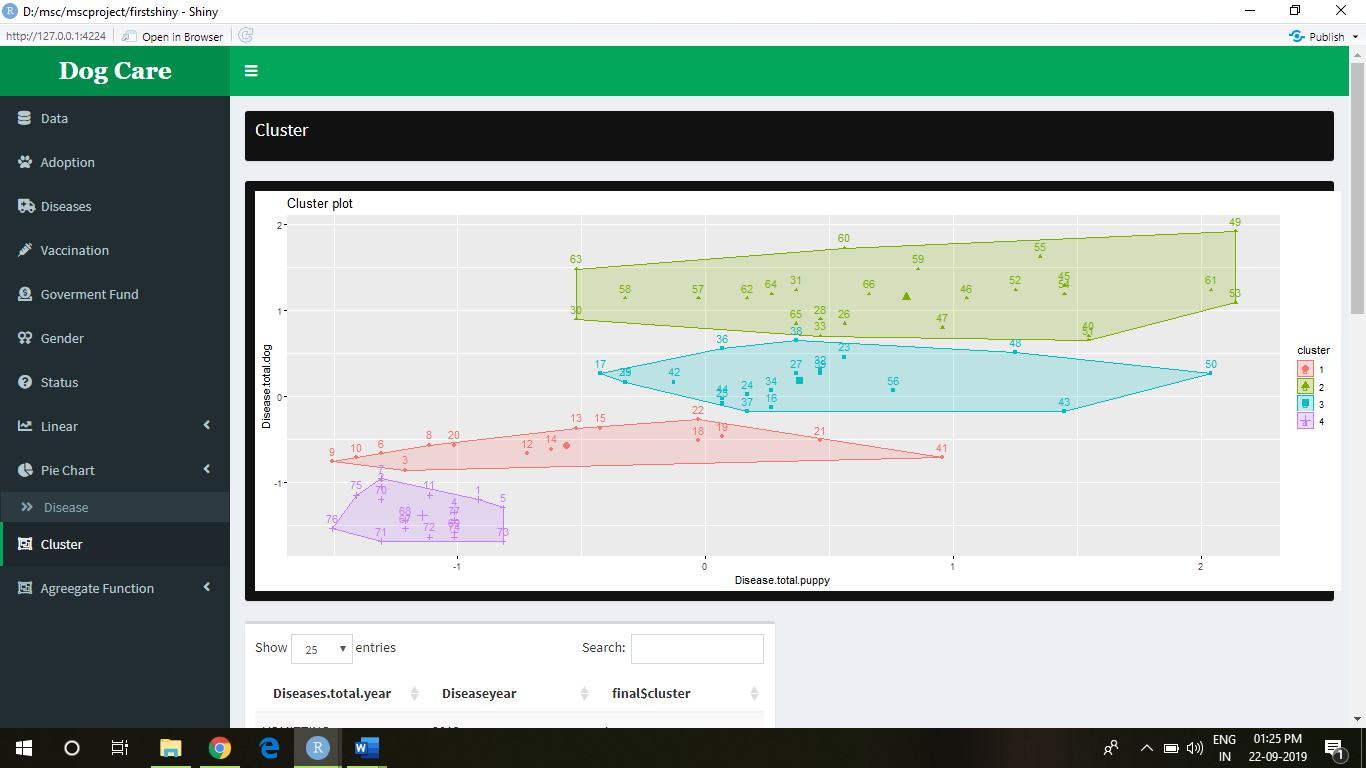
14. What is percentage of gastro in year 2019 for dogs?



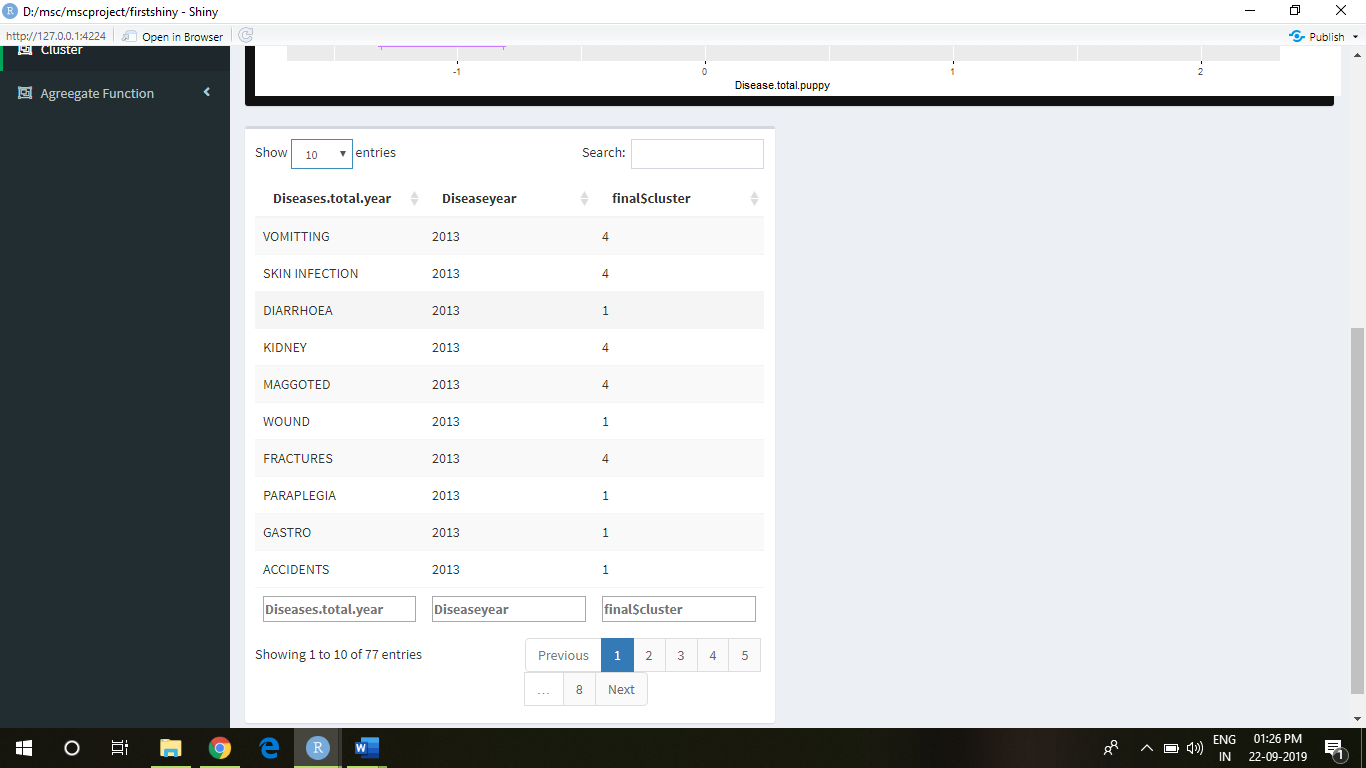
1. What is percentage of Accidents for puppies in year 2014?



1. How K-means clustering can be implemented on dog and puppy count by diseases based on year

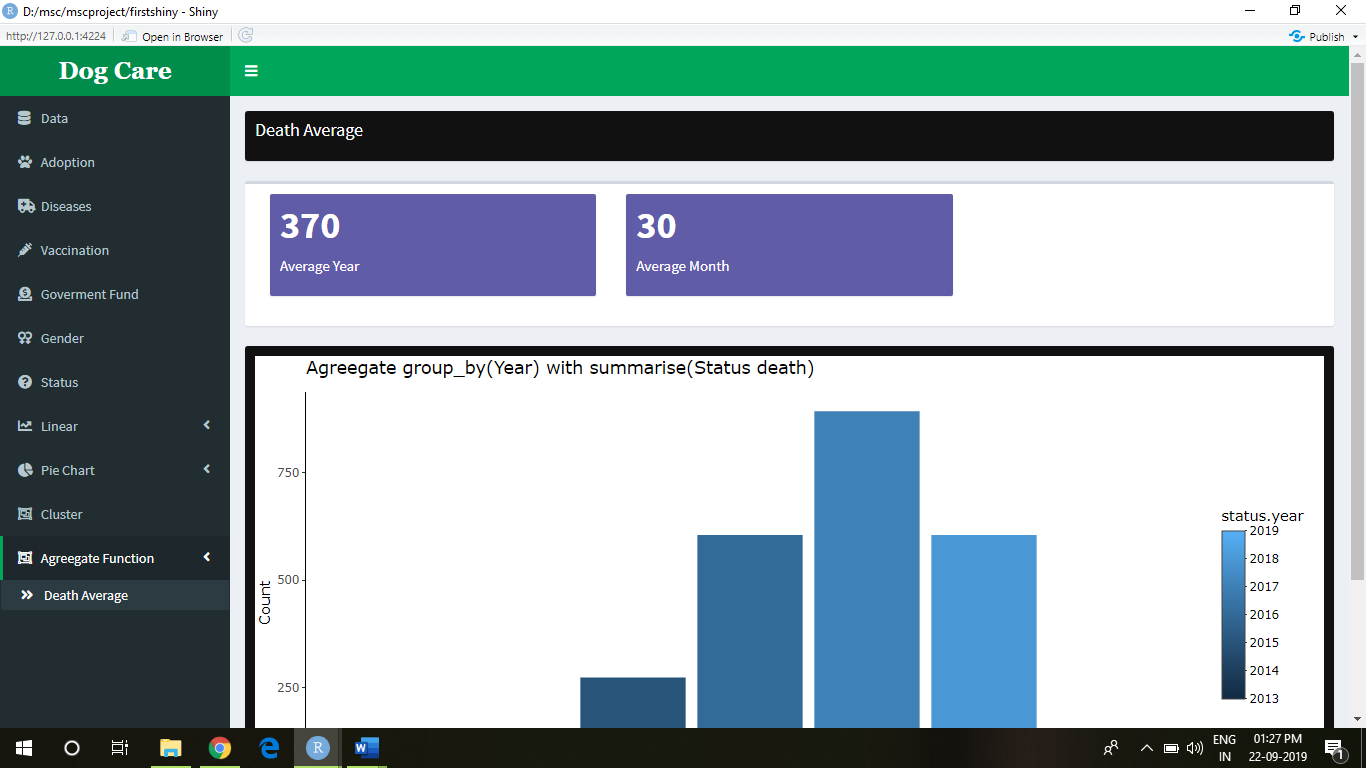


Showing which disease in which cluster?



K-means clustering can be used to classify observations into k groups of diseases of selected year, based on their similarity. Each group is represented by the mean value of points in the group, known as the cluster centroid. K-means algorithm requires users to specify the number of clusters to generate. After, computing k-means clustering, the R function fviz\_cluster () [factoextra package] can be used to visualize the results. The format is fviz\_cluster (km.res, data), where km.res is k-means results and data corresponds to the original data sets.

1. What is annual average death rate of dogs and puppies?



**Conclusions: -**

* There are 370 dogs and puppies which die every year and 30 dogs and puppies die every month based on various diseases.
* Less number of dogs are adopted than puppies.
* NGO’s are trying to increase the numbers to help dogs and puppies every year.
* Due to unprovided government fund Ngo’s cannot help dogs and puppies as much as they want to.
* In addition to that, the prediction of the data is significantly satisfiable and gets the clear solution of the increasing dog and puppies for different diseases for particular year.
* Technology will help to increase accuracy and efficiency of incident response and reporting and, thus resources, like medicines, vaccinations or any other can be allocated accordingly and in a better way.
* This in turn will enhance the Ngo’s as well as government and help to increase the efficient service

**Future Scope and Development:**

* This analysis is major useful for NGO’s and government for better medical service.
* The knowledge gain from analysis helps to predict the diseases with higher rates and precautions to be taken to eradicate the diseases.

**Reference and appendix: -**

* <http://www.awbi.in/>
* Code

library(shiny)

library(shinydashboard)

library(ggplot2)

library(leaflet)

dashboardPage(skin = "green",

dashboardHeader(title = "Dog Care"),

dashboardSidebar(

sidebarMenu(

menuItem("Data", tabName = "data", icon = icon("fas fa-database")),

menuItem("Adoption", tabName = "adoption", icon = icon("paw")),

menuItem("Diseases", tabName = "diseases", icon = icon("fas fa-ambulance")),

menuItem("Vaccination", tabName = "vaccination", icon = icon("syringe")),

menuItem("Goverment Fund", tabName = "Goverment\_Fund", icon = icon("fas fa-donate")),

menuItem("Gender", tabName = "Gender", icon = icon("fas fa-venus-double")),

menuItem("Status", tabName = "status", icon = icon("fas fa-question-circle")),

menuItem("Linear", tabName = "linear", icon = icon("fas fa-chart-line"),

menuSubItem("All",tabName = "all"),

menuSubItem("Dog",tabName = "doglinear"),

menuSubItem("Puppy",tabName = "diseaselinear") ),

menuItem("Pie Chart", tabName = "pie", icon = icon("fas fa-chart-pie"),

menuSubItem("Disease",tabName = "diseasepie")),menuItem("Cluster",tabName = "cluster",icon = icon("fas fa-object-group")),menuItem("Agreegate Function",tabName = "agreegate",icon = icon("fas fa-object-group"),

menuSubItem("Death Average",tabName = "deathagreegate")))

)),

dashboardBody(

tabItems(tabItem(tabName = "diseaselinear",

fluidRow(box(title = "Linear Graph For Diseases (Puppy Vs Year)",height = 50,width = 12,background = "black",solidHeader = TRUE)),

width = 500,

fluidRow(box(background = "black",

solidHeader = TRUE,

collapsible = TRUE,status = "success",title = "Select Diseases",

selectInput("diseaselinear","Diseases",""))),

fluidRow(box(background = "black",width=12,plotlyOutput("diseaselinearplot")))),

tabItem(tabName = "doglinear",

fluidRow(box(title = "Linear Graph For Diseases (Dog Vs Year)",height = 50,width = 12,background = "black",solidHeader = TRUE)),

width = 500,

fluidRow(box(background = "black",

solidHeader = TRUE,

collapsible = TRUE,status = "success",title = "Select Diseases",

selectInput("diseasedoglinear","Diseases",""))),

fluidRow(box(background = "black",width=12,plotOutput("diseasedoglinearplot")))),

tabItem(tabName = "deathagreegate",

fluidRow(box(title = "Death Average",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(box(width = 12,valueBoxOutput("avgyr\_",width = 4),valueBoxOutput("avgmonth",width = 4))),

fluidRow(box(background = "black",width=12,plotlyOutput("agreegate",height = 500)))

#fluidRow(box(title = "Summarise",background = "black",width=3,plotOutput("summariseagreegate",height = 500)))

), tabItem(tabName = "data",

fluidRow(box(title = "Choose Csv File",status = "success",background = "black",

solidHeader = TRUE,

collapsible = TRUE,

fileInput('file1', 'CSV File',

accept=c('text/csv',

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

'.csv'))

#selectInput('animalid', 'Select animal', "")

)),

fluidRow(box(title="Wordcloud"

,background = "black",width=12,plotOutput("wordcloud")))

#fluidRow(tableOutput('test'))

), tabItem(tabName = "vaccination",

fluidRow(box(title = "Area Wise Vaccination",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(box(background = "black",

solidHeader = TRUE,

collapsible = TRUE,status = "success",title = "Select vaccination Year",selectInput('vaccyr', 'vaccination Year', "")),valueBoxOutput("vaccvalue",width = 3)),

fluidRow(plotlyOutput("vaccplot",height = 500))),

#adoption

tabItem(tabName = "adoption",

fluidRow(box(title = "Adoption Count(Annually)",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(box(background = "black",width=12,plotlyOutput("plot1",height = 500)))

),

#donation

tabItem(tabName = "Goverment\_Fund",

fluidRow(box(title = "Goverment Fund",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(valueBoxOutput("Min\_",width = 3)),

fluidRow(box(background = "black",width=3,title='Top 10 Highest Goverment Fund In India',tableOutput("donationtable")),

box(background = "black",width=9,title='Goverment Fund',plotlyOutput("Piefund")))),

#fluidRow(column(width=6,d3Output("top\_donation")))),

#diseases

tabItem(tabName = "diseases",

fluidRow(box(title = "Diseases",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(box(background = "black",

solidHeader = TRUE,

title = "Select Disease",

collapsible = TRUE,status = "success",selectInput('diseaseid',"Diseases","")),valueBoxOutput("diseasevalue",width = 3),valueBoxOutput("diseasevaluepuppy",width = 3)

),

fluidRow(box(background = "black",width=12,plotlyOutput("plot2")))

),

#gender

tabItem(tabName = "Gender",

fluidRow(box(title = "Sex Ratio",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(tabBox(width = 500,#title = tagList(shiny::icon("gear"), "tabBox status"),

tabPanel(title="Male",fluidRow(box(background = "black",width=12,plotlyOutput("view\_male")))),

tabPanel(title="FeMale",fluidRow(box(background = "black",width=12,plotlyOutput("view\_female"))))

))),

#status

tabItem(tabName = "status",

fluidRow(box(title = "Status",height = 50,width = 12,solidHeader = TRUE,background = "black")),

#selectInput('deathyrid',"Select year ",""),

fluidRow(box(background = "black",

solidHeader = TRUE,

title = "Select Status",

collapsible = TRUE,status = "success",selectInput('statusid',"Status ","")),valueBoxOutput("statusvalue",width = 3),valueBoxOutput("statusvaluepuppy",width = 3)

),

fluidRow(box(background = "black",width=12,plotlyOutput("statusplot")))

),

#linear

tabItem(tabName = "all",

fluidRow(box(title = "Linear Graph for Status",height = 50,width = 12,solidHeader = TRUE,background = "black")),

fluidRow(tabBox(width = 500,

tabPanel(title="Status",fluidRow(box(background = "black",width=12,plotOutput("statusid"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black",verbatimTextOutput("summarystatus")))) # tabPanel(title="Dead",fluidRow(plotlyOutput("dead")),verbatimTextOutput("summarydead"))

))

),

tabItem(tabName = "dog",

fluidRow(tabBox(width = 500,

# tabPanel(title="Status",fluidRow(plotOutput("statusid")),verbatimTextOutput("summarystatus")),

tabPanel(title="Not Found",fluidRow(box(background = "black",width=12,plotlyOutput("notdog"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black",verbatimTextOutput("summarynotdog")),box(title = "Prediction",status = "success",background = "black",verbatimTextOutput("predictnotdog")))),

tabPanel(title="Admitted",fluidRow(box(background = "black",width=12,plotOutput("admitdog"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black",verbatimTextOutput("summaryadmitdog")))),

tabPanel(title="Released",fluidRow(box(background = "black",width=12,plotlyOutput("releaseddog"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black",verbatimTextOutput("summaryreleaseddog")))),

tabPanel(title="Dead",fluidRow(box(background = "black",width=12,plotlyOutput("deaddog"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black",verbatimTextOutput("summarydeaddog"))))

))

),

tabItem(tabName = "puppy",

fluidRow(tabBox(width = 500,

#tabPanel(title="Status",fluidRow(plotOutput("statusid")),verbatimTextOutput("summarystatus")),

tabPanel(title="Not Found",fluidRow(box(background = "black",width=12,plotlyOutput("notpuppy"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black",verbatimTextOutput("summarynotpuppy")),box(title = "Prediction",status = "success",background = "black",verbatimTextOutput("predictnotpuppy")))),

tabPanel(title="Admitted",fluidRow(box(background = "black",width=12,plotOutput("admitpuppy"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black", verbatimTextOutput("summaryadmitpuppy")))),

tabPanel(title="Released",fluidRow(box(background = "black",width=12,plotlyOutput("releasedpuppy"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black", verbatimTextOutput("summaryreleasedpuppy")))),

tabPanel(title="Dead",fluidRow(box(background = "black",width=12,plotlyOutput("deadpuppy"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black", verbatimTextOutput("summarydeadpuppy"))))

))

),

tabItem(tabName = "dogdisease",

tabBox(width = 500,

tabPanel(title="Dog",fluidRow(box(background = "black",width=12,plotOutput("dogdiseases"))),

br(),fluidRow(box(title = "Summary",status = "success",background = "black",verbatimTextOutput("summarydiseasedog"))))

#tabPanel(title="puppy",fluidRow(plotlyOutput("pie")))

)),

#pie

tabItem(tabName = "diseasepie",

fluidRow(box(title = "Pie chart",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(box(title = "select Year",solidHeader = TRUE,collapsible = TRUE,background = "black",status = "success",

selectInput('pieid',"Year ",""))),

tabBox(width = 20,

tabPanel(title="Dog",fluidRow(box(background = "black",width=12,plotlyOutput("piedog")))),

tabPanel(title="puppy",fluidRow(box(background = "black",width=12,plotlyOutput("pie"))))

)),

tabItem(tabName = "statuspie",

fluidRow(box(title = "Pie chart",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(box(title = "select Year",solidHeader = TRUE,collapsible = TRUE,background = "black",status = "success"

,selectInput('piestatusyr',"Year ","")),box(title = "select Status",solidHeader = TRUE,collapsible = TRUE,background = "black",status = "success",

selectInput('statuspieid',"Status",""))),

tabBox(width = 20,

tabPanel(title="Dog",fluidRow(plotlyOutput("statuspiedog"))),

tabPanel(title="puppy",fluidRow(plotlyOutput("statuspuppypie")))

)),

#cluster

tabItem(tabName = "cluster",

fluidRow(box(title = "Cluster",height = 50,width = 12,background = "black",solidHeader = TRUE)),

fluidRow(box(background = "black",width=12,plotOutput("cluster"))),

fluidRow(box(dataTableOutput("clustertabel")))

)

),

#header (dog care)

tags$head(tags$style(HTML('

.main-header .logo {

font-family: "Georgia", Times, "Times New Roman", serif;

font-weight: bold;

font-size: 24px;

}

')))

)

)

**Server.r**

library(shiny)

library(shinydashboard)

library(dplyr)

library(SnowballC)

library(tm)

library(RColorBrewer)

library(wordcloud)

library(ggplot2)

library(ggmap)

library(googleway)

library(ggalt)

library(factoextra)

library(cluster)

library(plotly)

library(highcharter)

library(r2d3)

library(config)

options(shiny.maxRequestSize = 30\*1024^2)

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

data <- reactive({

req(input$file1) ## ?req # require that the input is available

inFile <- input$file1

df <- read.csv(inFile$datapath, header = T, sep = ',')

output$wordcloud=renderPlot(

{

text <- readLines(inFile$datapath)

docs <- Corpus(VectorSource(text))

inspect(docs)

toSpace<-content\_transformer(function(x,pattern)gsub(pattern,"",x))

docs <- tm\_map(docs,toSpace,"/")

docs <- tm\_map(docs,toSpace,"@")

docs <- tm\_map(docs,content\_transformer(tolower))

docs <- tm\_map(docs,removeNumbers)

docs <- tm\_map(docs,removeWords,stopwords("english"))

docs <- tm\_map(docs,removeWords,c("blabla1","blabla2"))

docs <- tm\_map(docs,removePunctuation)

docs <- tm\_map(docs,stripWhitespace)

docs <- tm\_map(docs,stemDocument)

dtm <- TermDocumentMatrix(docs)

m <- as.matrix(dtm)

v <- sort(rowSums(m),decreasing = TRUE)

d <- data.frame(word=names(v),freq=v)

head(d,10)

set.seed(1234)

wordcloud(words = d$word,freq = d$freq,min.freq = 1,max.words = 900,random.order = FALSE,rot.per = 0.15,colors = brewer.pal(20,"Dark2"))

}

)

updateSelectInput(session,inputId = 'animalid', label='Year of adoption', choices = df$year.of.adoption, selected = df$year.of.adoption)

updateSelectInput(session,inputId = 'statusid', label='status', choices = df$status.of.animal, selected = df$status.of.animal)

updateSelectInput(session,inputId = 'statuspieid', label='Status', choices = df$status.dead, selected = df$status.dead)

updateSelectInput(session,inputId = 'genderid', label='gender', choices = df$gender, selected = df$gender)

updateSelectInput(session ,inputId = 'col',label = "Intake Type",choices = df$Intake.Type,selected = df$Intake.Type)

updateSelectInput(session, inputId = 'diseaseid',label='disease', choices = df$Diseases.total.year, selected = df$Diseases.total.year)

updateSelectInput(session,inputId = 'vaccyr', label='vaccination year', choices = df$year.of.vaccination, selected = df$year.of.vaccination)

updateSelectInput(session, inputId = 'pieid',label='select Year', choices = df$Diseaseyear, selected = df$Diseaseyear)

updateSelectInput(session, inputId = 'piestatusyr',label='select Year', choices = df$Diseaseyear, selected = df$Diseaseyear)

updateSelectInput(session, inputId = 'diseaselinear',label='Diseases', choices = df$Diseases.total.year, selected = df$Diseases.total.year)

updateSelectInput(session, inputId = 'diseasedoglinear',label='Diseases', choices = df$Diseases.total.year, selected = df$Diseases.total.year)

updateSelectInput(session, inputId = 'yrdeath',label='year', choices = df$Diseases.year, selected = df$Diseases.year)

return(df)

na.omit(df)

})

observe({

output$plot1 = renderPlotly({

p <- plot\_ly(data(), x = ~total.year, y = ~total.dog, name = 'Total Dog', type = 'scatter', mode = 'lines+markers') %>%

add\_trace(y = ~total.puppy, name = 'Total Puppy', mode = 'lines+markers') %>% layout(xaxis=list(title='Year'),yaxis=list(title='count'))

})

#disease

output$plot2 <-renderPlotly({

a<-data() %>%filter(data()$Diseases.total.year== input$diseaseid) %>%select(Diseases.total.year,Diseaseyear,Disease.total.dog,Disease.total.puppy)

b<-data.frame(list(c(a)))

#print(b)

dfm <- melt(b[,c('Diseaseyear','Disease.total.puppy','Disease.total.dog')],id.vars = 1)

g <- ggplot(dfm,aes(x= Diseaseyear,y= value)) +

geom\_bar(aes(fill = variable),stat = "identity",position = "dodge")

g

})

output$diseasevalue <- renderValueBox({

a<-data() %>%filter(data()$Diseases.total.year== input$diseaseid) %>%select(Diseases.total.year,Diseaseyear,Disease.total.dog,Disease.total.puppy)

b<-data.frame(list(c(a)))

sum\_dog <- b%>%group\_by(input$diseaseid)%>%summarise(total=sum(Disease.total.dog))

# print(sum\_dog)

sum\_dog%>%tally(sum\_dog$total)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Total Count of Dog ",color = "purple")

})

output$diseasevaluepuppy <- renderValueBox({

a<-data() %>%filter(data()$Diseases.total.year== input$diseaseid) %>%select(Diseases.total.year,Diseaseyear,Disease.total.dog,Disease.total.puppy)

b<-data.frame(list(c(a)))

sum\_puppy <- b%>%group\_by(input$diseaseid)%>%summarise(total=sum(Disease.total.puppy))

# print(sum\_dog)

sum\_puppy%>%tally(sum\_puppy$total)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Total Count of Puppy ",color = "purple")

})

#vaccination

output$vaccplot = renderPlotly({

a<-data() %>%filter(data()$year.of.vaccination==input$vaccyr) %>%select(Area,Vaccinated)

b<-data.frame(list(c(a)))

count<- b %>%group\_by(Area)%>%summarise(total=sum(Vaccinated))

#print(count)

p <- ggplot(count,aes(Area,total)) +geom\_bar(stat = "identity") # geom\_point()+geom\_line(aes(x=name),color="red",group=1)

p + theme(axis.text.x =

element\_text(size = 10,

angle=90,

hjust = 1,

vjust = 1)) #+ geom\_text(aes(label=total),vjust=-0.5, color="black", size=3,position = position\_nudge(y = -0.1))

})

output$vaccvalue <- renderValueBox({

a<-data() %>%filter(data()$year.of.vaccination==input$vaccyr) %>%select(Area,Vaccinated)

b<-data.frame(list(c(a)))

sum\_vacc <- b%>%group\_by(input$vaccyr)%>%summarise(total=sum(Vaccinated))

# print(sum\_dog)

sum\_vacc%>%tally(sum\_vacc$total)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Total Vaccination ",color = "purple")

})

#status

output$statusplot=renderPlotly({

a<-data() %>%filter(data()$status.of.animal== input$statusid) %>%select(status.year,status.of.animal,status.of.dog,status.of.puppy)

b<-data.frame(list(c(a)))

#print(b)

dfm <- melt(b[,c('status.year','status.of.dog','status.of.puppy')],id.vars = 1)

#print(dfm)

g <- ggplot(dfm,aes(x= status.year,y= value)) +

geom\_bar(aes(fill = variable),stat = "identity",position = "dodge") + xlab("Year")

#+ scale\_y\_log10()# + scale\_x\_continuous(labels=c("2013", "2014", "2015", "2016", "2017", "2018"))# + scale\_y\_continuous(breaks=seq(0,50,100))

g

})

output$statusvalue <- renderValueBox({

a<-data() %>%filter(data()$status.of.animal== input$statusid) %>%select(status.year,status.of.animal,status.of.dog,status.of.puppy)

b<-data.frame(list(c(a)))

sum\_dog <- b%>%group\_by(input$statusid)%>%summarise(total=sum(status.of.dog))

# print(sum\_dog)

sum\_dog%>%tally(sum\_dog$total)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Total Count of Dog ",color = "purple")

})

output$statusvaluepuppy <- renderValueBox({

a<-data() %>%filter(data()$status.of.animal== input$statusid) %>%select(status.year,status.of.animal,status.of.dog,status.of.puppy)

b<-data.frame(list(c(a)))

sum\_puppy <- b%>%group\_by(input$statusid)%>%summarise(total=sum(status.of.puppy))

# print(sum\_dog)

sum\_puppy%>%tally(sum\_puppy$total)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Total Count of puppy ",color = "purple")

})

#gender

output$view\_male=renderPlotly({

a<-data() %>%filter(data()$gender== "M") %>%select(gender.year,gender,gender.dog,gender.puppy)

b<-data.frame(list(c(a)))

#print(b)

dfm <- melt(b[,c('gender.year','gender.dog','gender.puppy')],id.vars = 1)

#print(dfm)

g <- ggplot(dfm,aes(x= gender.year,y= value)) +

geom\_bar(aes(fill = variable),stat = "identity",position = "dodge") +xlab("Year") #+ xlim(2013,2020)

#scale\_y\_log10()# + scale\_x\_continuous(labels=c("2013", "2014", "2015", "2016", "2017", "2018"))# + scale\_y\_continuous(breaks=seq(0,50,100))

g

})

output$view\_female=renderPlotly({

a<-data() %>%filter(data()$gender== "F") %>%select(gender.year,gender,gender.dog,gender.puppy)

b<-data.frame(list(c(a)))

#print(b)

dfm <- melt(b[,c('gender.year','gender.dog','gender.puppy')],id.vars = 1)

#print(dfm)

g <- ggplot(dfm,aes(x= gender.year,y= value)) +

geom\_bar(aes(fill = variable),stat = "identity",position = "dodge")+xlab("Year") #+

# scale\_y\_log10()# + scale\_x\_continuous(labels=c("2013", "2014", "2015", "2016", "2017", "2018"))# + scale\_y\_continuous(breaks=seq(0,50,100))

g

})

#donation

output$Min\_ <- renderValueBox({

data()%>%tally(data()$AMOUNT)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Total Donation",color = "purple",icon = icon("angle-double-right"))

# valueBox(value = min(s),

# subtitle =" Total Donation")

})

output$top\_donation <- renderD3({

data()%>%group\_by(STATE,AMOUNT)%>%

tally()%>%

collect()%>%

arrange(desc(n))%>%

head(10)%>%

arrange(AMOUNT)%>%

mutate(AMOUNT=str\_sub(AMOUNT,1,30))%>%

rename(

x=STATE,

y=n,

label=AMOUNT

)

r2d3("bar\_plot.js")

})

output$donationtable <- renderTable({

count<- data() %>%group\_by(STATE)%>%summarise(total=sum(AMOUNT))

c <- count%>% arrange(desc(total))

} )

#linear

output$notpuppy <- renderPlotly({

ggplot(data(),aes(stat.yr,not.puppy))+geom\_point()+geom\_smooth(method = lm)+xlab("Year")+ylab("count")

})

output$summarynotpuppy <- renderPrint({

linear<-lm(stat.yr~not.puppy,data())

summary(linear)

})

output$predictnotpuppy <- renderPrint({

linear<-lm(stat.yr~not.puppy,data())

# reduce <- lm(status.of.dog~status.of.puppy,data())

# full <- lm(status.of.dog~status.of.puppy,data())

# anova(reduce,full)

predict(linear,data.frame(not.puppy=17),interval='confidence')

})

#statuslinear

output$statusid <- renderPlot({

linear2<-lm(status.of.puppy~status.of.dog,data())

# print(summary(linear2))

ggplot(data(),aes(status.of.puppy,status.of.dog))+geom\_point()+geom\_smooth(method = lm)+xlab("status of puppy")+ylab('status of dog')

})

output$summarystatus <- renderPrint({

model <- lm(status.of.puppy~status.of.dog,data())

summary(model)

})

#lineardisease

output$diseaselinearplot <- renderPlotly({

a<-data() %>%filter(data()$Diseases.total.year== input$diseaselinear) %>%select(Diseases.total.year,Disease.total.puppy,Diseaseyear)

b<-data.frame(list(c(a)))

vacc\_sum <- b%>%group\_by(Diseaseyear)%>%summarise(total=sum(Disease.total.puppy))

# print(vacc\_sum)

ggplot(vacc\_sum,aes(Diseaseyear,total))+geom\_point()+geom\_smooth(method = lm)+xlab("Year")

})

output$diseasedoglinearplot <- renderPlot({

a<-data() %>%filter(data()$Diseases.total.year== input$diseasedoglinear) %>%select(Diseases.total.year,Disease.total.dog,Diseaseyear)

b<-data.frame(list(c(a)))

# print(b)

vacc\_sum <- b%>%group\_by(Diseaseyear)%>%summarise(total=sum(Disease.total.dog))

#print(vacc\_sum)

ggplot(vacc\_sum,aes(Diseaseyear,total))+geom\_point()+geom\_smooth(method = lm)+xlab("Year")

})

#piechart

output$pie <- renderPlotly({

a<-data() %>%filter(data()$Diseaseyear== input$pieid) %>%select(Diseaseyear,Diseases.total.year,Disease.total.puppy)

b<-data.frame(list(c(a)))

p <- plot\_ly(b, labels = ~Diseases.total.year, values = ~Disease.total.puppy, type = 'pie') %>%

layout(title = 'Diseases of Puppy',

xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),

yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))

})

output$piedog <- renderPlotly({

a<-data() %>%filter(data()$Diseaseyear== input$pieid) %>%select(Diseaseyear,Diseases.total.year,Disease.total.dog)

b<-data.frame(list(c(a)))

p <- plot\_ly(b, labels = ~Diseases.total.year, values = ~Disease.total.dog, type = 'pie') %>%

layout(title = 'Diseases of Dog',

xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),

yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))

})

output$statuspiedog <- renderPlotly({

a<-data() %>%filter(data()$Diseaseyear== input$piestatusyr & data()$status.dead== "Dead") %>%select(Diseaseyear,Diseases.total.year,Disease.total.dog,status.dead)

b<-data.frame(list(c(a)))

# print(b)

p <- plot\_ly(b, labels = ~Diseases.total.year, values = ~Disease.total.dog, type = 'pie') %>%

layout(title = 'Status of Dog',

xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),

yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))

})

output$Piefund <- renderPlotly({

p <- plot\_ly(data(), labels = ~STATE, values = ~AMOUNT, type = 'pie') %>%

layout(

xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),

yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))

})

})

observe({

#cluster

a <- data()

b<-a[12:15]

b1<-a[12:13]

d<-na.omit(b)

d1<-na.omit(b1)

#print(d)

sum(is.na(d))

c1<-d[-1:-2]

# print(head(c1))

e<-na.omit(data.frame(list(c(c1))))

#print(e)

sum(is.na(e))

set.seed(123)

final <- kmeans(e, 4, nstart = 25)

# print(final)

#agreegate

a<- data()%>%filter(data()$status.of.animal== "Dead")%>%select(status.of.animal,status.of.dog,status.of.puppy,status.year)

# print(a)

count<- a%>%group\_by(status.of.animal,status.year)%>%summarise(total=sum(status.of.dog,status.of.puppy))

#print(count)

output$cluster <- renderPlot({

fviz\_cluster(final, data = e)

#print(cbind(d,final$cluster))

#stringsAsFactors = FALSE

})

output$clustertabel <- renderDataTable(

cbind(d1,final$cluster)

)

output$agreegate <- renderPlotly({

count%>%

#Step 2

group\_by(status.of.animal,status.year)%>%

#Step 3

summarise(Average = mean(total))%>%

#Step 4

ggplot(aes(x = status.year, y = Average, fill = status.year)) +

geom\_bar(stat = "identity") +

theme\_classic() +

labs(

x = "Year",

y = "Count",

title = paste(

"Agreegate group\_by(Year) with summarise(Status death)"

)

)

})

output$avgyr\_ <- renderValueBox({

a<- summarise(count, Average =mean(total))

a%>%tally(a$Average)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Average Year",color = "purple")

})

output$avgmonth <-renderValueBox({

a<- summarise(count, Average =mean(total))

b <- data.frame(a/12)

b%>%tally(b$Average)%>%pull()%>%as.integer()%>%prettyNum(big.mark = ",")%>%valueBox(subtitle =" Average Month",color = "purple")

})

output$statusagreegate <- renderPlotly({

a<-data()%>%filter(data()$Diseases.year == input$yrdeath)%>%select(Status,Diseases.year)

#print(a)

b<-data.frame(list(c(a)))

p <- plot\_ly(b, labels = ~Status, values = ~count, type = 'pie') %>%

layout(title = 'Status of Dog',

xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),

yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))

})

output$map <- renderPlot({

mapdata <- get\_data\_from\_map(download\_map\_data("https://code.highcharts.com/mapdata/countries/in/in-all.js"))

glimpse(mapdata)

pop = as.data.frame(c(43200,0,14400,0,84000,10000,0,0,

0,0,3002160,0,5582000,0,0,72000,

14400,0,659200,0,0,1788800,0,0,

0,158400,0,8223200,0,120400,20000,73200,

2373680,26400))

state= mapdata%>%

select(`hc-a2`)%>%

arrange(`hc-a2`)

State\_pop = as.data.frame(c(state, pop))

names(State\_pop)= c("State", "Population")

hcmap("https://code.highcharts.com/mapdata/countries/in/in-all.js", data = State\_pop, value = "Population",

joinBy = c("hc-a2", "State"), name = "Donation",

dataLabels = list(enabled = TRUE, format = '{point.name}'),

borderColor = "#FAFAFA", borderWidth = 0.1,

tooltip = list(valueDecimals = 0))

(hcmap)

})

})

}