Brief introduction to R Plotly package

The plotly package in R is an interface to open source Javascript charting library plotly.js

---

Plots can be created online in the plotly web GUI or offline using the plotly package

---

Interactive plots that can be easily shared online using the plotly API/account

---

Plotly plots can embed into R Markdown, R Shiny

Tool tip feature and other controls

---

ggplot2 graphs can be easily converted to interactive plotly objects using ggplotly() function

---

Package documentation

---

https://cran.r-project.org/web/packages/plotly/plotly.pdf

Install the 'plotly' package

#

install.packages("plotly")

---

install.packages("ggplot2")

---

load the 'plotly' package

library(plotly)

General Syntax plot_ly()

Plot_ly functions creates a plotly object and produces interactive visualization based on data and arguments supplied

---

plot_ly (data, x, y , type, mode, color, size,....)

?plot_ly()

data = a dataframe

---

x and y axis values

---

type = specifies the type of plot or trace such as 'histogram', 'scatter',

'bar', 'box', 'heatmap', 'histogram', 'histogram2d', 'histogram2dcontour', 'pie', 'contour', 'scatter3d',

'surface', 'mesh3d', 'scattergeo', 'choropleth', 'scattergl', 'scatterternary', 'scattermapbox', 'area'

mode = specifies the mode, such as 'line', 'points', 'markers'

color = specifies the color of data points usually a function of I() to avoid scaling

colors = colourbrewer pallete, vector of color in hexa format

size = name or expression that defines the size of the data points

We can use other functions such as layout() for axis formating

and other functions that comes with plotly package

"mtcars A data frame with 32 observations on 11 (numeric) variables. [, 1] mpg Miles/(US) gallon [, 2] cyl Number of cylinders [, 3] disp Displacement (cu.in.) [, 4] hp Gross horsepower [, 5] drat Rear axle ratio [, 6] wt Weight (1000 lbs) [, 7] qsec 1/4 mile time [, 8] vs Engine (0 = V-shaped, 1 = straight) [, 9] am Transmission (0 = automatic, 1 = manual) [,10] gear Number of forward gears "

str(mtcars)

Pipe operator %>% will forward a value, or the

result of an expression,

into the next function call/expression.

mtcars %>% filter(carb > 1) %>% group_by(cyl) %>% summarise(Avg_mpg = mean(mpg)) %>% arrange(desc(Avg_mpg))

Tilde operator is used to define the relationship between

dependent variable and independent variables in a statistical model

formula. The variable on the left-hand side of tilde operator

is the dependent variable and the variable(s)

on the right-hand side of

tilde operator is/are called the independent variable(s).

Simple Scatter Plot

Type of plot is not exclusively defined.

Plotly will guess the best type based on the data type.

x~y

x~.

z<-x~.

help("~")

y=x*2

y~x*2

p = plot_ly(data=mtcars, x=wt, y=mpg) p

Define the plot type exclusively as "scatter"

& mode as "markers"

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers") p

Change color of scatter data points

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", color = I("black")) p

Change color of data points using marker argument

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", marker = list(color="green", size=10)) p

Styled Marker

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", marker = list(size = 7, color = 'rgba(255, 182, 193, .9)', line = list(color = 'rgba(152, 0, 0, .8)', width = 3))) p

str(mtcars)

Set the color scale based on a factor variable

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", color = ~as.factor(cyl))

p

Use color Brewer palletes for data point colors

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", color = ~as.factor(cyl), colors = "Set1")

p

Use customized pallete for data point colors

pal <- c("red", "blue", "orange") p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", color = ~as.factor(cyl), colors = pal)

p

Change data point shape based on factor variable

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", symbol = ~as.factor(cyl), symbols = c('circle','x','o'), marker = list(size = 5))

p

Set the color scale based on a continous variable

p= plot_ly(data=mtcars, x=~wt, y=~mpg, mode="markers", color = ~gear) p

str(mtcars) p= plot_ly(data=mtcars, x=~wt, y=~mpg, mode="markers", color = ~as.factor(disp)) p

str(mtcars)

Scatter plot with diffent data point size based on a continous variable

p = plot_ly(data=mtcars, x=~wt, y=~mpg, type="scatter", mode="markers", color = ~as.factor(cyl), size = ~hp)

p

Hide the legend

p <- plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", symbol = ~as.factor(cyl), symbols = c('circle','x','o'), marker = list(size = 5)) %>% layout(showlegend = TRUE)

p

Change the orientation of the legend (below the plot)

p <- plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", symbol = ~as.factor(cyl), symbols = c('circle','x','o'), marker = list(size = 5)) %>% layout(legend = list(orientation = 'h'))

p

Change the orientation of the legend (inside the plot)

p <- plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", symbol = ~as.factor(cyl), symbols = c('circle','x','o'), marker = list(size = 5)) %>% layout(legend = list(x = 0.8, y = 0.9))

p

Change the orientation of the legend (outside the plot)

p <- plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", symbol = ~as.factor(cyl), symbols = c('circle','x','o'), marker = list(size = 5)) %>% layout(legend = list(x = 1, y = 0.5))

p

Add title to the legen

p <- plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", symbol = ~as.factor(cyl), symbols = c('circle','x','o'), marker = list(size = 5)) %>% layout(legend = list(x = 1, y = 0.5))

p

Adding chart title and axis labels

p <- plot_ly(data=mtcars, x=~wt, y=~mpg, type = "scatter", mode = "markers", symbol = ~as.factor(cyl), symbols = c('circle','x','o'), marker = list(size = 5)) %>%

layout(title="Scatter plot using R Plotly", xaxis=list(title="Weight", showgrid = F), yaxis=list(title="MPG", showgrid = F))

p

Customizing mouse hover text

p = plot_ly(data = mtcars, x=~mpg, y=~wt, type = "scatter", mode="markers", hoverinfo = "text", text = paste("Miles per gallon: ", mtcars$mpg, "
", "Weight: ", mtcars$wt)) p

Add annotations to the plot - add_annotations() function syntax

add_annotations() function is used to add annotations to the plot along with the following arguments

x = set annotations x coordinate position (x axis value based on the dataset)

y = set annotations y coordinate position (y axis value based on the dataset)

text = text associated with annotation

showarrow = 0 or 1 depending whether to show arrow or not

xref = "paper" telling plotly to set x reference to the plot (in which case x=0 means the left most, x=1 means right most)

yref = "paper" telling plotly to set y reference to the plot (in which case y=0 means the bottom, y=1 means top)

xref = "x" (non paper x coordinates)

yref = "y (non paper y coordinates)

arrowhead, arrowsize, font, ax, ay

Example of annotations on a single data points

display annotations for good mileage

plot_ly(data = mtcars, x=~mpg, y=~wt, type = "scatter", mode="markers") %>% add_annotations(x=mtcars$mpg[which.max(mtcars$mpg)], y=mtcars$wt[which.max(mtcars$mpg)], text="Good mileage", showarrow=T)

Example of annotations - placing text at a desired location on the plot

Display Data Source

Demo of x/y ref as "paper"

plot_ly(data = mtcars, x=~mpg, y=~wt, type = "scatter", mode="markers") %>% add_annotations(xref="paper", yref="paper", x=1, y=-0.2, text="Data Source : mtcars",

)

Example #1 of annotations on multiple data points

display annotation for low and high mileage

plot_ly(data = mtcars, x=~mpg, y=~wt, type = "scatter", mode="markers") %>% add_annotations(x=mtcars$mpg[which.max(mtcars$mpg)], y=mtcars$wt[which.max(mtcars$mpg)], text="high mileage") %>% add_annotations(x=mtcars$mpg[which.min(mtcars$mpg)], y=mtcars$wt[which.min(mtcars$mpg)], text="low mileage")

Example #2 of annotations on multiple data points

display annotations for automatic transission cars

plot_ly(data = mtcars, x=~mpg, y=~wt, type = "scatter", mode="markers") %>% add_annotations(x = mtcars$mpg[mtcars$am==0], y= mtcars$wt[mtcars$am==0], text="auto",)

Styling annotations

using font argument

plot_ly(data = mtcars, x=~mpg, y=~wt, type = "scatter", mode="markers") %>% add_annotations(x=mtcars$mpg[which.max(mtcars$mpg)], y=mtcars$wt[which.max(mtcars$mpg)], text="Good mileage", font = list(color = "green", family="sans serif", size = 20))

mtcars