QTM 151

Week 10 – plotly (cont'd)

Umberto Mignozzetti

Recap

We've learned:

- Programming methods
- `dplyr *_join` methods: joining data
- `tidyr` methods: reshape datasets
- `forcats` methods: working with categorical variables
- `lubridate` methods: to work with dates
- `qplot` methods: to create quick plots
- `ggplotly -- plotly` methods: create quick interactive plots.

Do you have any questions about `ggplotly`?

The quiz is posted. Good luck!

Our GitHub page is: https://github.com/umbertomig/qtm151

Today's Agenda

- `plot_ly` today:
  - Scatter-plots
  - Line-plots
  - Bar-plots

Getting Started

Getting Started: loading packages

```r
# Loading tidyverse
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyv

## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.6     ✓ dplyr   1.0.8
## ✓ tidyr   1.2.0     ✓ stringr 1.4.0
## ✓ readr   2.1.2     ✓ forcats 0.5.1

## Warning: package 'tidyr' was built under R version 4.1.2

## Warning: package 'readr' was built under R version 4.1.2

## Warning: package 'dplyr' was built under R version 4.1.2

## ── Conflicts ────────────────────────────────────────── tidyverse_c
## x dplyr::filter() masks stats::filter()
```

Loading datasets

```r
# Loading tips dataset
tips ← read.csv('https://raw.githubusercontent.com/umbertomig/qtm
head(tips, 2)
```

```
##   obs totbill  tip sex smoker day  time size
## 1   1   16.99 1.01   F     No Sun Night    2
## 2   2   10.34 1.66   M     No Sun Night    3
```

```r
# Loading PErisk dataset
PErisk ← read.csv('https://raw.githubusercontent.com/umbertomig/c
head(PErisk, 2)
```

```
##     country courts       barb2 prsexp2 prscorr2    gdpw2
## 1 Argentina      0 -0.7207754       1         3  9.69017
## 2 Australia      1 -6.9077550       5         4 10.30484
```

Loading datasets

```
# Gapminder
head(gapminder, 2)
```

```
## # A tibble: 2 × 6
##   country     continent  year lifeExp     pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>   <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8 8425333      779.
## 2 Afghanistan Asia       1957    30.3 9240934      821.
```

plot_ly: scatter-plots

`plot_ly` Scatter-plots

- It is very simple to build a plotly scatter-plot.
- The most basic syntax is the following:

```
plot_ly(data = data_set_here,
x = ~x_axis_here,
y = ~y_axis_here,
type = 'scatter')
```

- Let's try?!

plotly

```
head(PErisk)
plot_ly(data = PErisk,
  x = ~barb2,
  y = ~gdpw2)
```

plot_ly Scatter-plots styling

- We can change the style of our plot_ly. For example: we can change the dots and add text to markers:

```
plot_ly(data = PErisk, x = ~barb2, y = ~gdpw2,
        text = ~paste("Country: ", country), # Hover text
        type = 'scatter',
        marker = list(size = 10, # Markers characteristics
                      color = 'red',
                      line = list(color = 'black',
                                  width = 2))) %>%
  layout(title = 'Informal Markets and GDP', # Layout
         yaxis = list(zeroline = FALSE),
         xaxis = list(zeroline = FALSE))
```

plot_ly Scatter-plots qualitative

- We can also color our scatter-plot by a qualitative variable:

```
PErisk$courts2 ← factor(PErisk$courts, labels = c('Not Independent','Independent'))
plot_ly(data = PErisk, x = ~barb2, y = ~gdpw2,
        text = ~paste("Country: ", country), # Hover text
        color = ~courts2, colors = c('blue', 'red'),
        type = 'scatter', mode = 'markers',
        marker = list(size = 10)) %>%
  layout(title = 'Informal Markets and GDP', # Layout
         yaxis = list(zeroline = FALSE),
         xaxis = list(zeroline = FALSE))
```

`plot_ly` 3D-Scatter-plots

- It is also easy to build 3D scatterplots.
- Here we plot infant mortality and GDP in the US by year.
- Warning: the plot is very awkward.

```
gmd ← filter(gapminder, country == 'United States')
plot_ly(data = gmd, x = ~year, y = ~gdpPercap, z = ~lifeExp,
        color = ~year,
        type = 'scatter3d', mode = 'markers',
        marker = list(size = 10)) %>%
  layout(title = 'U.S. GDP and Life Expectancy by Year', # Layout
         yaxis = list(zeroline = FALSE),
         xaxis = list(zeroline = FALSE))
```

`plot_ly` Scatter-plots

- **Your turn:** Create a `plot_ly` with `totbill` and `tip`. Then, change the title, remove the x and y zerolines, and display the weekday (`day`) in the text. Color by `time`.

plot_ly: line-plots

`plot_ly` Line Plots

- Line plots are very useful for time series, such as stock prices.
- The most basic syntax is the following:

```
plot_ly(data = data_set_here,
x = ~x_axis_here,
y = ~y_axis_here,
type = 'scatter',
mode = 'lines')
```

- Let's try?!

plot_ly: line-plots

```
gmd ← filter(gapminder, country == 'United States')
plot_ly(data = gmd, x = ~year, y = ~gdpPercap,
    type = 'scatter',
    mode = 'lines') %>%
  layout(title = 'U.S. GDP per capita over time', # Layout
         yaxis = list(title = 'GDP Per Capita'),
         xaxis = list(title = 'Year'))
```

plot_ly: line-plots

- We can add markers to the lines:

```
plot_ly(data = gmd, x = ~year, y = ~gdpPercap,
    type = 'scatter',
    mode = 'lines+markers') %>%
  layout(title = 'U.S. GDP per capita over time', # Layout
         yaxis = list(title = 'GDP Per Capita'),
         xaxis = list(title = 'Year'))
```

plot_ly: line-plots with multiple lines

- We can also add different countries:

```
gmd ← filter(gapminder, country %in% c('United States', 'Canada'))
plot_ly(data = gmd, x = ~year, y = ~gdpPercap,
  type = 'scatter', linetype = ~country,
  mode = 'lines+markers') %>%
  layout(title = 'U.S. and Canada GDP per capita over time',
         yaxis = list(title = 'GDP Per Capita'),
         xaxis = list(title = 'Year'))
```

plot_ly: line-plots with multiple lines

- We can also plot all countries in the same chart.

```
plot_ly(data = gapminder, x = ~year, y = ~log(gdpPercap)) %>%
  add_lines(color =~country) %>%
  hide_legend()
```

plot_ly: line-plots with multiple lines

- And change the color of a desired country.

```
gapminder %>%
  group_by(country) %>%
  plot_ly(x = ~year, y = ~log(gdpPercap)) %>%
  add_lines(text = 'All Countries', alpha=I(0.2)) %>%
  filter(country=="United States") %>%
  add_lines(name="United States", color=I("blue")) %>%
  add_lines(data = filter(gapminder, country=="Canada"),
    name="Canada", color=I("red"))
```

plot_ly: line-plots

- **Your turn:** In the `gapminder` dataset, select three countries of your choice and plot them.

plot_ly: bar-plots

`plot_ly` Bar-plots

- Bar plots are great to visualize qualitative data.
- The most basic syntax is the following:

```
plot_ly(data = data_set_here,
x = ~x_axis_here,
y = ~y_axis_here,
type = 'bar')
```

- Let's try?!

plot_ly: bar-plots

```
tbl ← table(PErisk$prsexp2); tbl ← as.data.frame(tbl)
plot_ly(data = tbl, x = ~Var1, y = ~Freq,
  type = 'bar') %>%
  layout(title = 'Expropriation Risk in 1992', # Layout
         xaxis = list(title = 'Expropriation Risk (high to low)'))
```

plot_ly: bar-plots

- We can add multiple bars to the barplot:

```
tbl ← table(PErisk$prsexp2); tbl ← as.data.frame(tbl)
tbl2 ← table(PErisk$prscorr2); tbl2 ← as.data.frame(tbl2)
dat ← data.frame(Risk = tbl$Var1, exprop = tbl$Freq, corrup = tbl2$Freq)
plot_ly(data = dat, x = ~Risk, y = ~exprop,
  type = 'bar', name = 'Expropriation Risk') %>%
  add_trace(y = ~corrup, name = 'Corruption Risk') %>%
  layout(title = 'Investment Risks in 1992', # Layout
         yaxis = list(title = 'Frequencies'),
         xaxis = list(title = 'Risks'),
         barmode = 'group')
```

plot_ly: bar-plots

- **Your turn:** In the `tips` dataset, create a barplot with gender and days that the person goes to a pub.

Questions?

Have a great weekend!