Homework 1. R and Interactive Visualization (60 points)

Nikhil Ambati

2023-09-15

```r
knitr::opts_chunk$set(echo = TRUE)
library(data.table)
library(tidyverse)
library(tidyverse)
library(plotly)
library(ggplot2)
```

In this homework you should use plotly unless said otherwise.

To create pdf version of your homework, knit it first to html and then print it to pdf. Interactive plotly plots can be difficult sometimes to convert to static images suitable for insertion to LaTex documents (that is knitting to PDF).

Look for questions in R-chunks as comments and plain text (they are prefixed as Q.).

Part 1. Iris Dataset. (26 points)

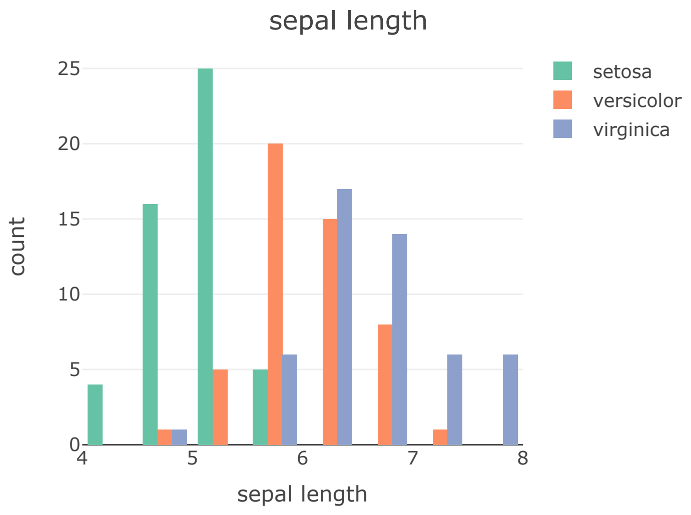
> "The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by the British statistician and biologist Ronald Fisher in his 1936 paper The use of multiple measurements in taxonomic problems as an example of linear discriminant analysis" https://en.wikipedia.org/wiki/Iris_flower_data_set

```r
# Q1.1. Read the iris.csv file  (2 points)
# hint: use fread from data.table, it is significantly faster than default methods
#       be sure to have strings as factors (see stringsAsFactors argument)
df_iristable1<-fread("C:\\Users\\Nikhil\\Sem 2\\SDM 2\\iris.csv",stringsAsFactors=TRUE)
```

```r
# Q1.2. Show some values from data frame (2 points)
head(df_iristable1)
```

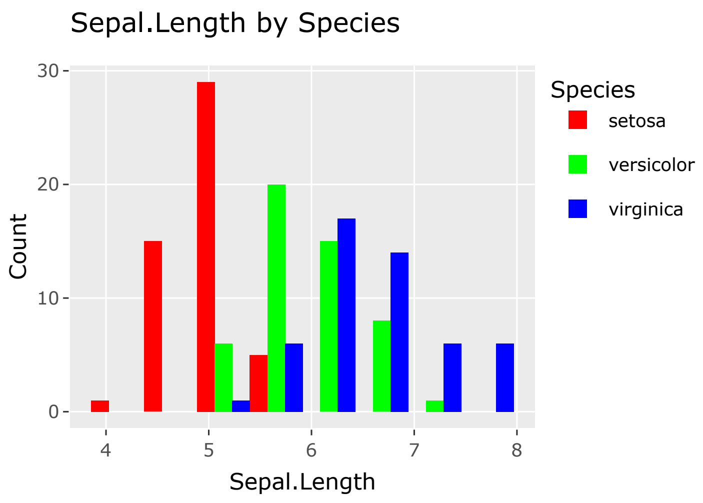
```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1:          5.1         3.5          1.4         0.2  setosa
## 2:          4.9         3.0          1.4         0.2  setosa
## 3:          4.7         3.2          1.3         0.2  setosa
## 4:          4.6         3.1          1.5         0.2  setosa
## 5:          5.0         3.6          1.4         0.2  setosa
## 6:          5.4         3.9          1.7         0.4  setosa
```

```r
# Q1.3. Build histogram plot for Sepal.Length variable for each species using plot_ly
# (use color argument for grouping) (2 points)
# should be one plot
plot<-plot_ly(data=df_iristable1,x = ~Sepal.Length, color = ~Species, type = "histogram")%>%
  layout(
    title="sepal length",
    xaxis=list(title="sepal length",range=c(4, NA)),
    yaxis=list(title="count")
)
plot
```



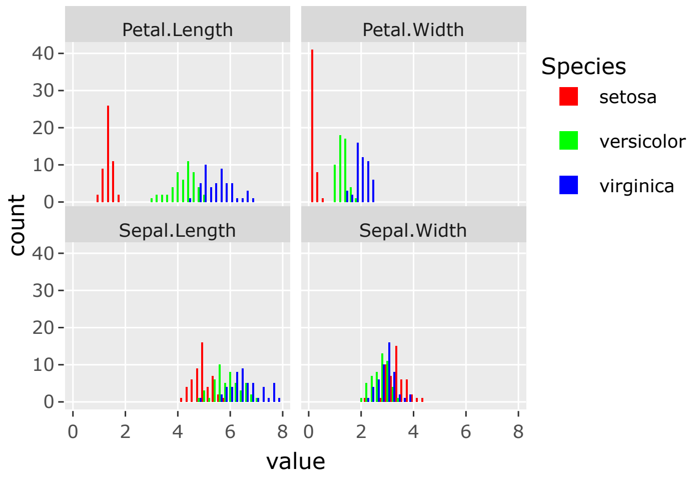
```
# Q1.4. Repeat previous plot with ggplot2 and convert it to plotly with ggplotly (3 points)
```

```r
my_colors <- c("red", "green", "blue")
gg <- ggplot(data=df_iristable1, aes(x=Sepal.Length, fill = Species)) +
  geom_histogram(alpha=1,position="dodge", bins = 8) +
  scale_fill_manual(values = my_colors) +
  labs(title = "Sepal.Length by Species",
       x = "Sepal.Length",
       y = "Count")

ggplotly(gg)
```



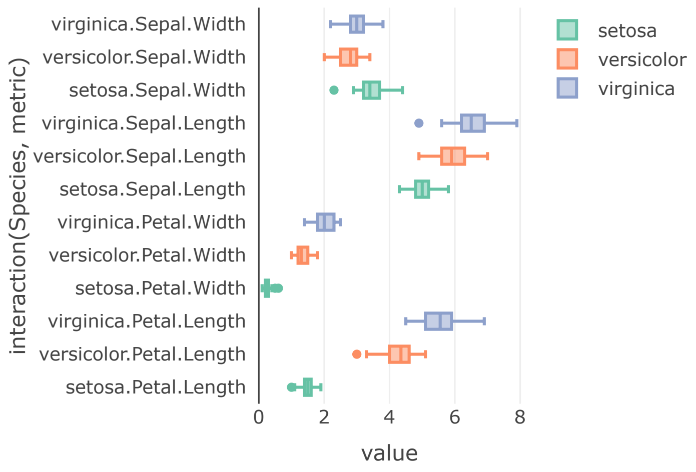
```
# Q1.5. Create facet 2 by 2 plot with histograms similar to previous but for each metric
# (3 points)
```

```r
# hint:
#   following conversion to long format can be useful:
#   iris %>% pivot_longer(...)
#
data_temp <- iris %>%
  gather(key="metric",value="value",-Species)

ggplo2<-ggplot(data_temp,aes(x = value,fill=Species)) +
  geom_histogram(binwidth=0.2,position="dodge") +
  scale_fill_manual(values = my_colors) +
  facet_wrap(~metric,nrow = 2)

ggplotly(ggplo2)
```



Q1.6. Which metrics has best species separations? (3 points) Petal.Length has the best species separations.

```
# Q1.7. Repeat above plot but using box plot (3 points)

data_temp<-df_iristable1 %>%
  pivot_longer(cols=-Species,names_to ="metric",values_to ="value")

plot_7<-plot_ly(data_temp,x=~value,y=~interaction(Species, metric),color=~Species,type="box",marker = li
plot_7
```

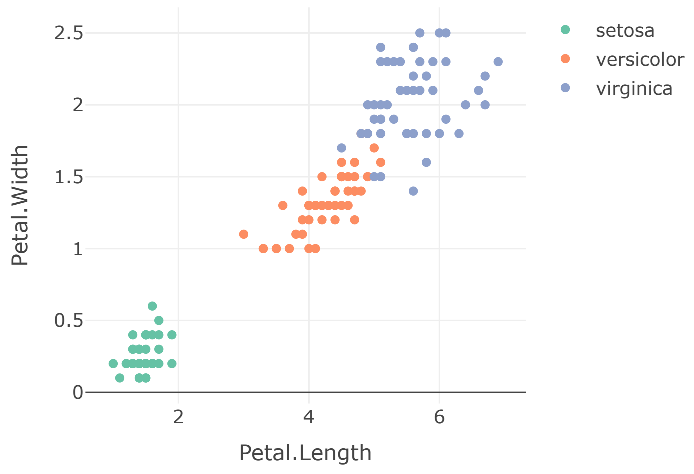


```
# Q1.8. Choose two metrics which separates species the most and use it to make scatter plot
# color points by species (3 points)
```

```r
plot_8<-plot_ly(df_iristable1,x=~Petal.Length,y=~Petal.Width,color=~Species)
plot_8
```

```
## No trace type specified:
##   Based on info supplied, a 'scatter' trace seems appropriate.
##   Read more about this trace type -> https://plotly.com/r/reference/#scatter
```

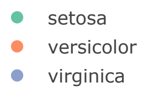
```
## No scatter mode specifed:
##   Setting the mode to markers
##   Read more about this attribute -> https://plotly.com/r/reference/#scatter-mode
```



```
# Q1.9. Choose three metrics which separates species the most and use it to make 3d plot
# color points by species (3 points)
```

```r
plot_9<-plot_ly(df_iristable1,x=~Sepal.Length,y=~Sepal.Width,z=~Petal.Width,color=~Species)


plot_9
```

```
## No trace type specified:
##   Based on info supplied, a 'scatter3d' trace seems appropriate.
##   Read more about this trace type -> https://plotly.com/r/reference/#scatter3d

## No scatter3d mode specifed:
##   Setting the mode to markers
##   Read more about this attribute -> https://plotly.com/r/reference/#scatter-mode
```



Q1.10. Comment on species separation (2 points): Upon analyzing the plot, a clear linear separation among the different species becomes evident. Notably, the setosa species exhibits a distinct characteristic: it possesses a relatively low petal width compared to both versicolor and virginica. Conversely, setosa demonstrates a higher sepal length in comparison to these other two species, versicolor and virginica.

Part 2. Covid-19 Dataset. (34 points)

Download us-states.csv (there is also a copy in homework assignment) from https://github.com/nytimes/covid-19-data/. README.md for details on file content.

```
# Q2.1. Read us-states.csv (3 points)
df_covid<-fread("C:\\Users\\Nikhil\\Sem 2\\SDM 2\\us-states.csv",stringsAsFactors=TRUE)
df_covid
```

```
##               date         state fips   cases deaths
##     1: 2020-01-21    Washington   53       1      0
##     2: 2020-01-22    Washington   53       1      0
##     3: 2020-01-23    Washington   53       1      0
##     4: 2020-01-24      Illinois   17       1      0
##     5: 2020-01-24    Washington   53       1      0
##    ---
## 50682: 2022-09-03      Virginia   51 2045387  21463
## 50683: 2022-09-03    Washington   53 1787487  14109
## 50684: 2022-09-03 West Virginia   54  584771   7294
## 50685: 2022-09-03     Wisconsin   55 1835422  15091
## 50686: 2022-09-03       Wyoming   56  174828   1881
```

```
# Q2.2. Show some values from dataframe (3 points)
head(df_covid)
```

```
##          date      state fips cases deaths
## 1: 2020-01-21 Washington   53     1      0
## 2: 2020-01-22 Washington   53     1      0
## 3: 2020-01-23 Washington   53     1      0
## 4: 2020-01-24   Illinois   17     1      0
## 5: 2020-01-24 Washington   53     1      0
## 6: 2020-01-25 California    6     1      0
```

```
# Q2.3. Create new dataframe with new cases per month for each state (5 points)
# hint:
#    is cases column cumulative or not cumulative?
new_cases_summary <- df_covid %>%
  group_by(state, year = year(date), month = month(date)) %>%
  summarise(new_cases = max(cases)-min(cases)) %>%
  ungroup()
```

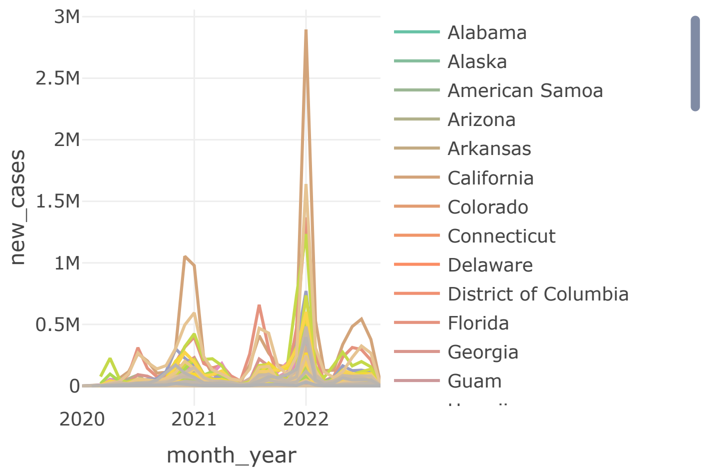
```
## `summarise()` has grouped output by 'state', 'year'. You can override using the
## `.groups` argument.
```

```
new_cases_summary$month_year <- paste(new_cases_summary$year, "-", new_cases_summary$month, "-1")
new_cases_summary$month_year <- as.Date(new_cases_summary$month_year, format = "%Y - %m -%d")
new_cases_summary
```

```
## # A tibble: 1,732 x 5
##    state    year month new_cases month_year
##    <fct>   <dbl> <dbl>     <int> <date>    
##  1 Alabama  2020     3       993 2020-03-01
##  2 Alabama  2020     4      5960 2020-04-01
##  3 Alabama  2020     5     10658 2020-05-01
##  4 Alabama  2020     6     19511 2020-06-01
##  5 Alabama  2020     7     48761 2020-07-01
##  6 Alabama  2020     8     36709 2020-08-01
##  7 Alabama  2020     9     27085 2020-09-01
##  8 Alabama  2020    10     36541 2020-10-01
##  9 Alabama  2020    11     55539 2020-11-01
## 10 Alabama  2020    12    108326 2020-12-01
## # i 1,722 more rows
```

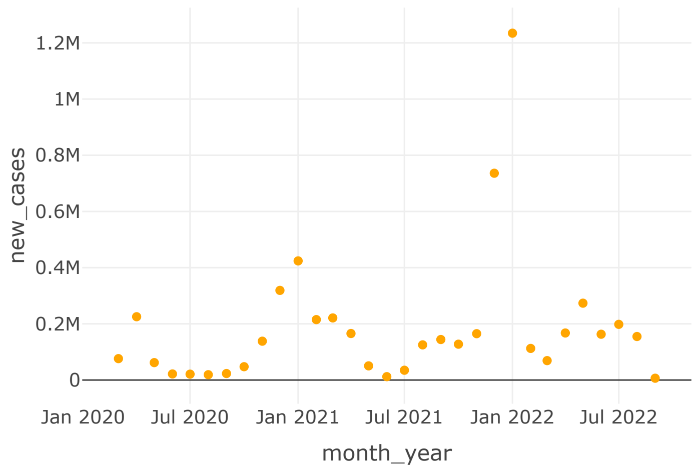
```
# Q2.4.Using previous dataframe plot new monthly cases in states, group by states
# The resulting plot is busy, use interactive plotly capabilities to limit number
# of displayed states
# (4 points)
plot_4<-plot_ly(new_cases_summary,
                x=~month_year,
                y=~new_cases,
                color=~state,
                type="scatter",
                text = ~paste("Month Years: ", month_year, "<br>New Casesss: ", new_cases),
                mode="lines")
plot_4
```

```
## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2 is 8
## Returning the palette you asked for with that many colors

## Warning in RColorBrewer::brewer.pal(N, "Set2"): n too large, allowed maximum for palette Set2 is 8
## Returning the palette you asked for with that many colors
```



```
# Q2.5.Plot new monthly cases only in NY state
# (3 points)
```

```
plot_ly(
  filter(new_cases_summary, state == "New York"),
  x = ~month_year,
  y = ~new_cases,
  type = "scatter",
  mode = "markers",
  marker = list(color = "orange"),
  text = ~paste("Month Year: ", month_year, "<br>New Cases: ", new_cases),
  showlegend = FALSE
)
```

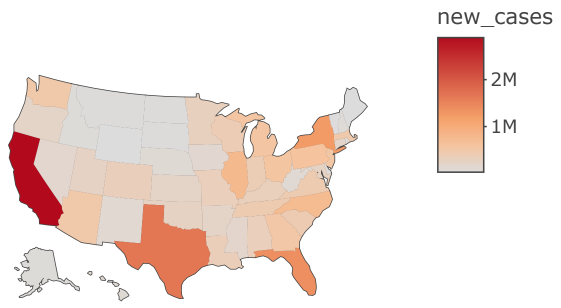


```
# Q2.6. Found the year-month with highest cases in NY state
# (3 points)
```

```r
solution_6<-filter(filter(new_cases_summary,state=="New York"),new_cases == max(new_cases))
solution_6
```

```
## # A tibble: 1 x 5
##   state     year month new_cases month_year
##   <fct>    <dbl> <dbl>     <int> <date>
## 1 New York  2022     1   1234485 2022-01-01
```

```r
# Q2.7. Plot new cases in determined above year-month
# using USA state map, color each state by number of cases  (5 points)
# hint:
#   there two build in constants in R: state.abb and state.name
#   to convert full name to abbreviation
selected_month <- filter(new_cases_summary, year == '2022', month=='1')
selected_month$state_abbr <- state.abb[match(selected_month$state, state.name)]

colors <- c(
  c(0, "blue"),
  c(0.5, "white"),
  c(1, "red")
)
selected_month <- na.omit(selected_month)
g <- list(
  scope = 'usa',
  projection = list(type = 'albers usa'),
  lakecolor = toRGB('red')
)
selected_month <- na.omit(selected_month)
plot_geo(selected_month) %>%
  add_trace(
    z = ~new_cases, text=state.name,span = I(0),
    colorscale = colors,
    locations = state.abb,locationmode='USA-states'
  ) %>%












  layout(geo = g)
```

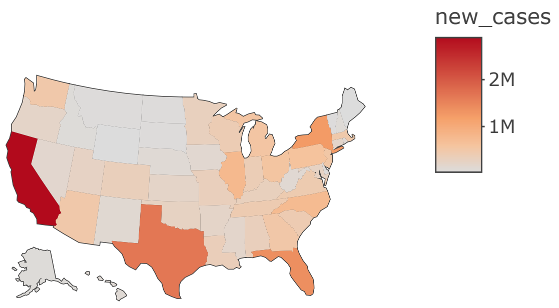


```
# Q2.8. Add animation capability (5 points)
# hint:
```

```r
#     for variable frame you need either integer or character/factorial so
#     convert date to character or factorial
library(plotly)
library(plotly)
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
selected_month$month_year <- as.yearmon(paste(selected_month$year, selected_month$month, "1"), "%Y %m %
selected_month$month_year <- as.character(selected_month$month_year)

# Assuming you have a month_year column in your data
# You may need to create one if it doesn't exist
selected_month <- filter(new_cases_summary, year == '2022', month=='1')
selected_month$state_abbr <- state.abb[match(selected_month$state, state.name)]

colors <- c(
  c(0, "blue"),
  c(0.5, "white"),
  c(1, "red")
)
selected_month <- na.omit(selected_month)
g <- list(
  scope = 'usa',
  projection = list(type = 'albers usa'),
  lakecolor = toRGB('red')
)
selected_month <- na.omit(selected_month)
plot_geo(selected_month) %>%
  add_trace(
    z = ~new_cases, text=state.name,span = I(0),
    colorscale = colors,
    locations = state.abb,locationmode='USA-states'
  ) %>%
  layout(geo = g)
```



Q2.9. Compare animated plot from Q2.8 to plots from Q2.4/Q2.5 (When you would prefer one or another?) (3 points) I feel that Animated plot gives much clarity than the plot which we had in 2.4 and 2.5. These are more interactive than the other which we have.