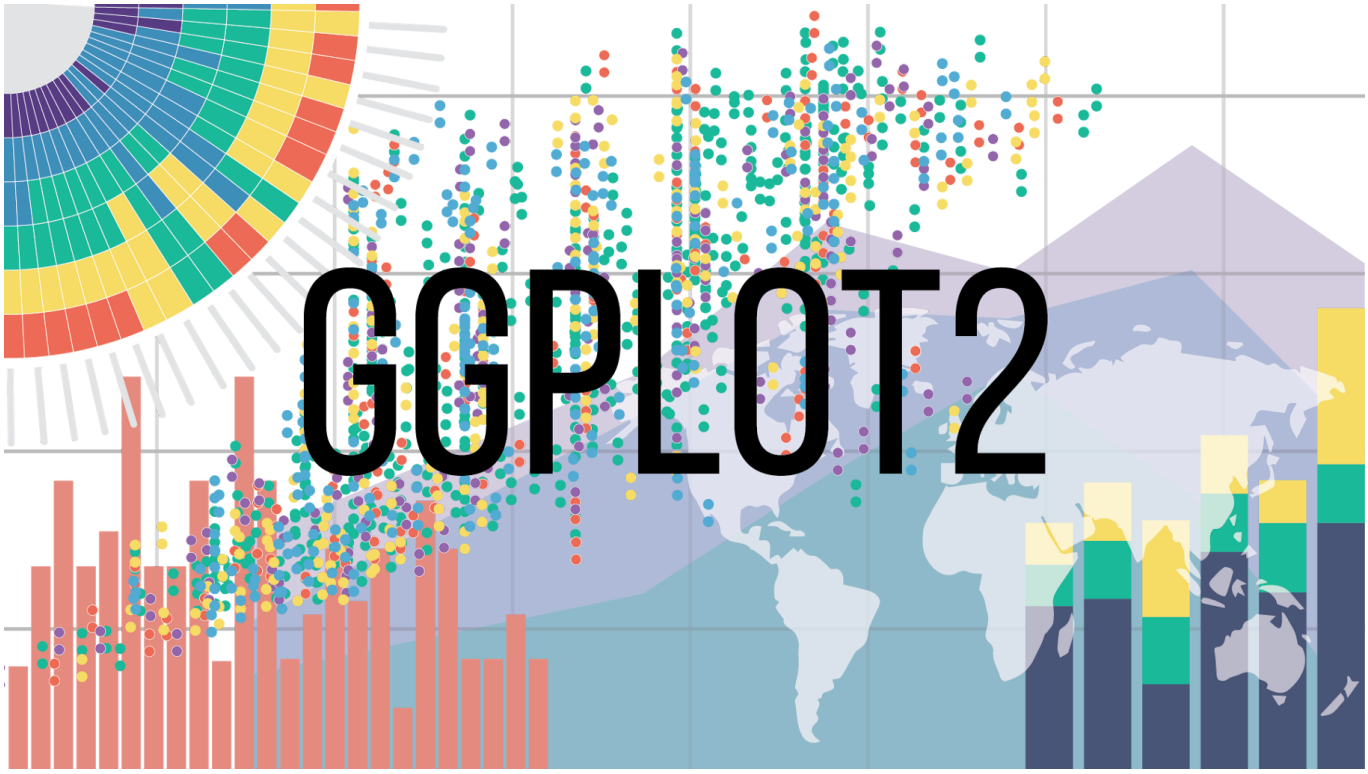
Exploratory Data Analysis in R for beginners (Part 2)

A more advanced method of doing EDA with 'ggplot2' and 'tidyverse'

Joe Tran Follow
Oct 19, 2019 · 9 min read ★



ggplot2

In my previous article, '***Exploratory Data Analysis in R for beginners (Part 1)***', I have introduced a basic step-by-step approach from data importing to cleaning and visualization. Here is a quick summary of Part 1:

- Import data appropriately with *fileEncoding* and *na.strings* arguments. I showed how it is different from the normal way of importing csv file with read.csv().

- Some basic cleaning the data with 'tidyverse' package
- Visualization with Boxplot, Barplot, Correlation plot in 'ggplot2' package

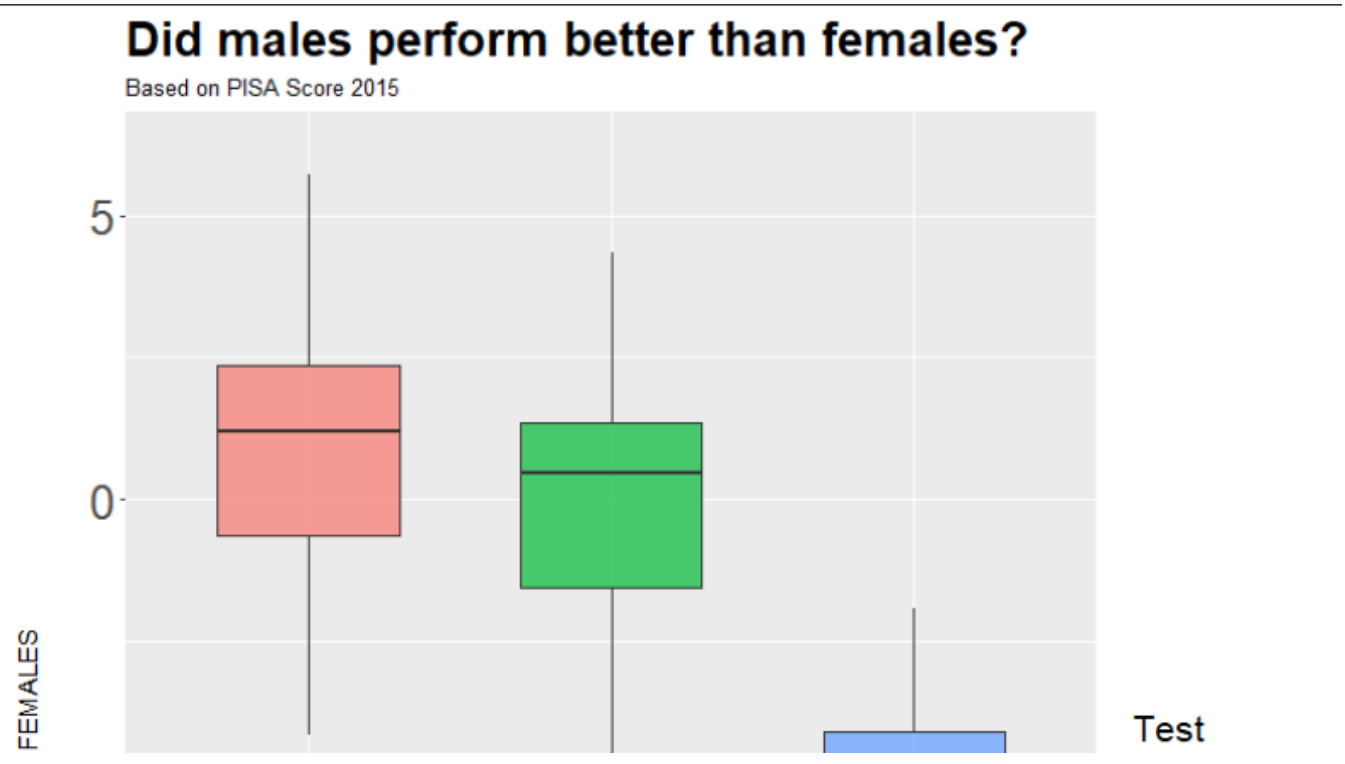
Those are the basic steps in performing simple EDA. However, to make our plots, charts and graphs more informative and of course visually appealing, we need to make one step further. What do I mean by that? Let's find it out!

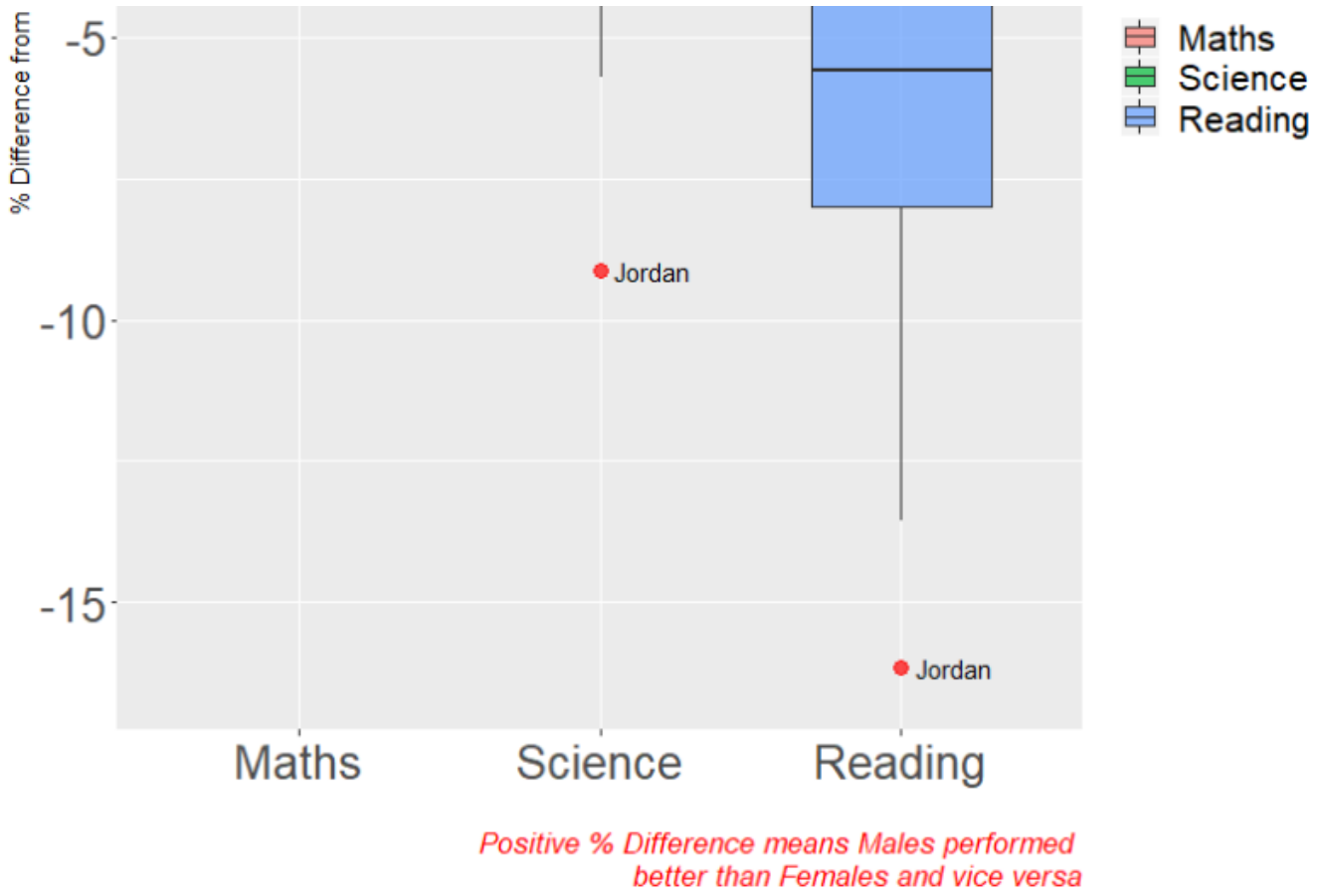
What would you expect to find in this article?

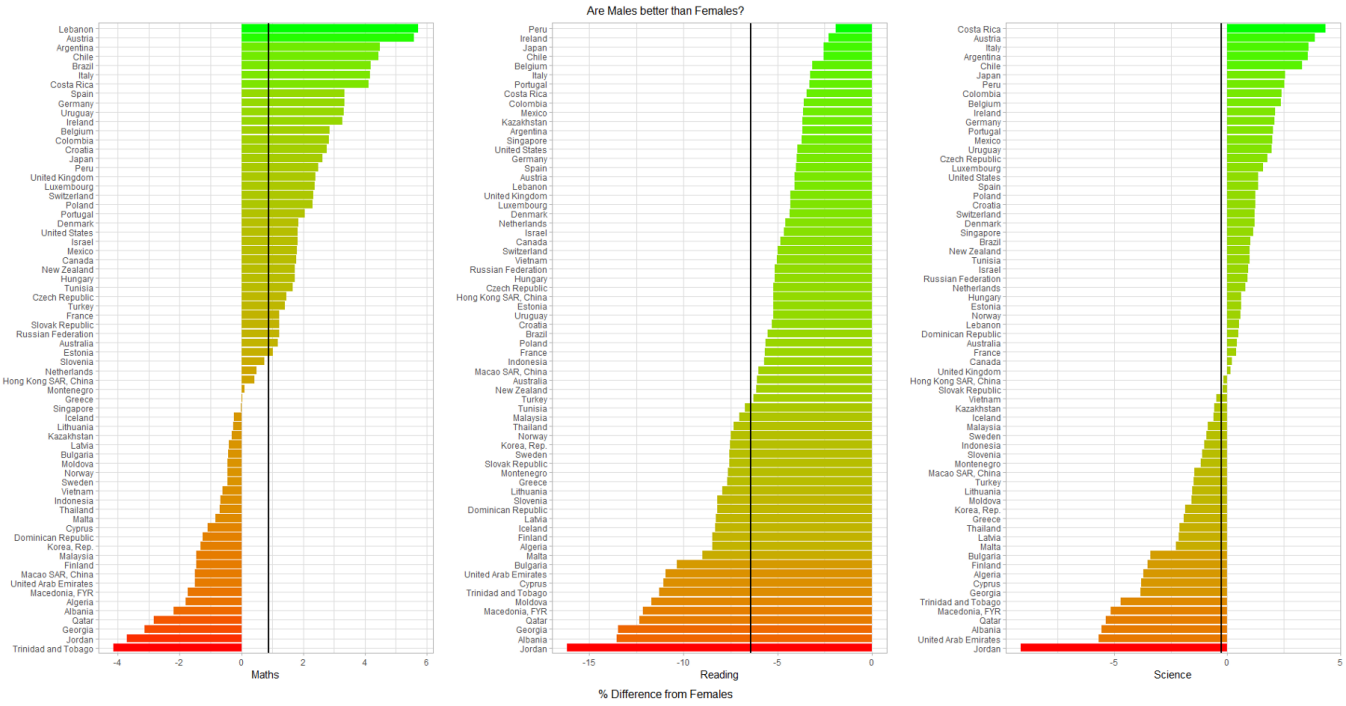
Doing EDA is not merely about plotting graphs. It is about making **informative** graphs. In this article, you would expect to find the following tricks:

1. How to play around with the dataset to get the best version for each type of analysis? There is no one-size-fits-all dataset. Each analysis and visualization have different purposes, hence there comes different data structures.
2. Change the order of the legends in the plot
3. Let R identify the outliers and label them on the plot
4. Combine graphs with the use of '***gridExtra***' package

By the end of this article, you would be able to generate the following plots:







Let's get started, folks!!!!

To refresh your memories

Let's quickly look at the types of datasets that we created in Part 1 of this series.

```
view(df)
```

	Country.Name	Maths	Maths.F	Maths.M	Reading	Reading.F	Reading.M	Science	Science.F	Science.M
1	Albania	413.1570	417.7500	408.5455	405.2588	434.6396	375.7592	427.2250	439.4430	414.9576
2	Algeria	359.6062	363.0725	356.4951	349.8593	366.2082	335.1854	375.7451	383.2209	369.0352
3	Argentina	409.0333	400.4431	418.3884	425.3031	432.9581	416.9666	432.2262	424.9944	440.1020
5	Australia	493.8962	490.9855	496.7613	502.9006	518.8658	487.1855	509.9939	508.9216	511.0493
6	Austria	496.7423	483.1330	510.0982	484.8656	495.0752	474.8460	495.0375	485.5268	504.3712
9	Belgium	506.9844	499.7390	514.0026	498.5242	506.6386	490.6642	501.9997	496.0319	507.7805
15	Brazil	377.0695	369.5493	385.0406	407.3486	418.5617	395.4633	400.6821	398.7000	402.7830
16	Bulgaria	441.1899	442.1631	440.3189	431.7175	456.5986	409.4498	445.7720	453.9011	438.4966
20	Canada	515.6474	511.1417	520.1661	526.6678	539.7624	513.5355	527.7047	527.1562	528.2548
22	Chile	422.6714	413.4490	431.7981	458.5709	464.5616	452.6422	446.9561	439.6174	454.2186
23	Colombia	389.6438	384.4883	395.3911	424.9052	432.2819	416.6816	415.7288	411.0316	420.9651
26	Costa Rica	400.2534	392.3129	408.4516	427.4875	434.8748	419.8605	419.6080	410.8349	428.6660
28	Croatia	464.0401	457.9612	470.5987	486.8632	499.5858	473.1367	475.3912	472.5863	478.4173
30	Cyprus	437.1443	439.5341	434.7064	442.8443	468.6583	416.8271	432.5964	440.9482	424.1478
31	Czech Republic	492.3254	488.6656	495.7942	487.2501	500.6527	474.5475	492.8300	488.3983	497.0304
32	Denmark	511.0876	506.3748	515.7565	499.8146	510.9516	488.7816	501.9369	498.9027	504.9427
33	Dominican Republic	327.7020	329.7459	325.5866	357.7377	372.7806	342.1682	331.6388	330.8290	332.4770
37	Estonia	519.5291	516.8728	522.0804	519.1429	533.3620	505.4863	534.1937	532.5228	535.7986
39	Finland	511.0769	514.9650	507.4528	526.4247	550.5112	503.9746	530.6612	540.5118	521.4797
40	France	492.9204	489.9540	495.9317	499.3061	513.7640	484.6293	494.9776	494.0342	495.9353
42	Georgia	403.8332	410.5960	397.7478	401.2881	431.8820	373.7585	411.1315	419.6164	403.4965
43	Germany	505.9713	497.5311	514.1177	509.1041	519.6741	498.9021	509.1406	503.8121	514.2837
45	Greece	453.6299	453.5732	453.6821	467.0395	486.4600	449.1362	454.8288	459.4177	450.5984
49	Hong Kong SAR, China	547.9310	546.7682	549.0658	526.6753	540.9844	512.7113	523.2774	523.7491	522.8172
50	Hungary	476.8309	472.7395	480.9055	469.5233	481.9596	457.1377	476.7475	475.2484	478.2405
51	Iceland	488.0332	488.5870	487.4457	481.5255	501.7167	460.1036	473.2301	474.6556	471.7177
52	Indonesia	386.1096	387.4450	384.7793	397.2595	408.9994	385.5642	403.0997	405.1289	401.0783
54	Ireland	503.7220	495.4450	511.5797	520.8148	526.9491	514.9914	502.5751	497.1740	507.7026
55	Israel	469.6695	465.5169	473.9902	478.9606	490.1650	467.3026	466.5528	464.4477	468.7432
56	Italy	489.7287	479.8237	499.7621	484.7580	492.7091	476.7038	480.5468	472.1190	489.0838
57	Japan	532.4399	525.4960	539.2673	515.9585	522.6553	509.3740	538.3948	531.5329	545.1415
58	Jordan	380.2590	387.3772	373.0013	408.1022	443.5964	371.9124	408.6691	427.9967	388.9627

```
View(df2)
```

	Country.Name	Score	value

1	Albania	Maths.F	417.7500
2	Albania	Maths.M	408.5455
3	Albania	Reading.F	434.6396
4	Albania	Reading.M	375.7592
5	Albania	Science.F	439.4430
6	Albania	Science.M	414.9576
7	Algeria	Maths.F	363.0725
8	Algeria	Maths.M	356.4951
9	Algeria	Reading.F	366.2082
10	Algeria	Reading.M	335.1854
11	Algeria	Science.F	383.2209
12	Algeria	Science.M	369.0352
13	Argentina	Maths.F	400.4431
14	Argentina	Maths.M	418.3884
15	Argentina	Reading.F	432.9581
16	Argentina	Reading.M	416.9666
17	Argentina	Science.F	424.9944
18	Argentina	Science.M	440.1020
19	Australia	Maths.F	490.9855
20	Australia	Maths.M	496.7613
21	Australia	Reading.F	518.8658
22	Australia	Reading.M	487.1855
23	Australia	Science.F	508.9216
24	Australia	Science.M	511.0493
25	Austria	Maths.F	483.1330
26	Austria	Maths.M	510.0982
27	Austria	Reading.F	495.0752
28	Austria	Reading.M	474.8460
29	Austria	Science.F	485.5268
30	Austria	Science.M	504.3712
31	Belgium	Maths.F	499.7390
32	Belgium	Maths.M	514.0026

```
View(df4) ## df3 combines with df2 to get df4
```

Please refer back to my previous article for a detailed explanation on how to manipulate the original dataset to get these various versions.

Great! Let's start our new visual plots now

Boxplot

Simple boxplot

First of all, we want to achieve this kind of boxplot

In order to get this, we must have a data frame where the **rows** are the **countries** and 4 columns, namely, Country names, % difference in Maths, Reading and Science. Now refer back to all the dataframes we created earlier, we can see that **df** has all of these requirements. Hence we will select relevant columns from **df** and name it as **df5**

```
df5 = df[,c(1,11,12,13)]
boxplot(df5$Maths.Diff, df5$Reading.Diff, df5$Science.Diff,
        main = 'Are Males better than Females?',
        names = c('Maths','Reading','Science'),
        col = 'green'
        )
```

Done! Following this code, you should be able to get the plot as above.

Higher-level boxplot

Now we want to move on to the next level. This is the plot that we want to get

Notice the following differences:

1. The subtitle
2. The titles of the axes
3. The layout and color of outliers.

4. The caption

Thankfully, the '**ggplot2**' package has everything we need. It is just a matter of whether we can find these **arguments** and **functions** in the '**ggplot2**' package. But before we even move on to what arguments and functions to use, we need to determine what kind of data frame/structure the dataset should be. By looking at this plot, we can see that the data frame must have **1 column of 3 categories (Maths, Reading and Science), 1 column of numeric results for the % difference in performance in each subject and of course, 1 column for the country name. Hence 3 columns in total**.

Now we have **df5** that looks like this

How would we transform this into a data frame described above? Remember a trick I introduced in Part 1 of this series, the magical function is **pivot_longer().** Now, lets do it:

```
df6 = df5 %>% pivot_longer(c(2,3,4))
names(df6) = c('Country','Test Diff','Result')
View(df6)
```

```
new = rep(c('Maths', 'Reading', 'Science'),68) #to create a new
column indicating 'Maths', 'Reading' and 'Science'

df6 = cbind(df6, new)
names(df6) = c('Country','Test Diff','Result', 'Test')   #change
column names
View(df6)
```

The column ‘Test Diff’ is now redundant so you can choose to delete that, simply use the code below. (This step is not compulsory)

```
df6$'Test Diff' = NULL
```

Here, I will not delete it.

Great! Now that we have the data set in the right structure ready for visualization, let’s use ggplot now

1. To get the title, subtitle, caption, use **labs(title = ‘ …’ , y = ‘ …’, x = ‘ …’, caption = ‘ …’, subtitle = ‘ …’)**

2. If the title or caption is long, use ‘ **…. \n …**’ to break it down into 2 lines.

3. To indicate outliers, inside geom_boxplot()

```
geom_boxplot(alpha = 0.7,
               outlier.colour='blue',   #color of outlier
               outlier.shape=19,        #shape of outlier
               outlier.size=3,          #size of outlier
               width = 0.6, color = "#1F3552", fill = "#4271AE"
               )
```

Combine everything together:

```
ggplot(data = df6, aes(x=Test,y=Result, fill=Test)) +
  geom_boxplot(alpha = 0.7,
               outlier.colour='blue',
               outlier.shape=19,
               outlier.size=3,
               width = 0.6, color = "#1F3552", fill = "#4271AE"
               )+
  theme_grey() +
  labs(title = 'Did males perform better than females?',
       y='% Difference from FEMALES',x='',
       caption  = 'Positive % Difference means Males performed \n
better than Females and vice versa',
       subtitle = 'Based on PISA Score 2015')
```

Is this good enough? The answer is **NO:** Caption, titles and subtitles are too small, the proportion and size of the plot is not as good as the plot we introduced at the start of this section.

We can do a better job.

To adjust the size and so on, use **theme ()** function

```
theme(axis.text=element_text(size=20),
        plot.title = element_text(size = 20, face = "bold"),
        plot.subtitle = element_text(size = 10),
        plot.caption = element_text(color = "Red", face = "italic",
size = 13)

        )
```

Here, don't ask me how I got those numbers to put. This is trial and error. You can just simply try out any numbers until you get the perfect size and position.

Let's now combine everything together

```
ggplot(data = df6, aes(x=Test,y=Result, fill=Test)) +
  geom_boxplot(alpha = 0.7,
               outlier.colour='blue',
               outlier.shape=19,
               outlier.size=3,
               width = 0.6, color = "#1F3552", fill = "#4271AE"
               )+
  theme_grey() +
  labs(title = 'Did males perform better than females?',
       y='% Difference from FEMALES',x='',
       caption  = 'Positive % Difference means Males performed \n
better than Females and vice versa',
       subtitle = 'Based on PISA Score 2015') +
theme(axis.text=element_text(size=20),
        plot.title = element_text(size = 20, face = "bold"),
        plot.subtitle = element_text(size = 10),
        plot.caption = element_text(color = "Red", face = "italic",
```

```
size = 13)
)
```

The plot looks much better now. You can stop here. However, I would like to introduce a way to rearrange the order of variables to get a decreasing trend. Simply use

```
scale_x_discrete(limits=c("Maths","Science","Reading"))
```

Hence, combining everything

```
ggplot(data = df6, aes(x=Test,y=Result, fill=Test)) +
  geom_boxplot(alpha = 0.7,
               outlier.colour='blue',
               outlier.shape=19,
               outlier.size=3,
               width = 0.6, color = "#1F3552", fill = "#4271AE"
               )+
scale_x_discrete(limits=c("Maths","Science","Reading"))+

  theme_grey() +
  labs(title = 'Did males perform better than females?',
       y='% Difference from FEMALES',x='',
       caption  = 'Positive % Difference means Males performed \n
better than Females and vice versa',
       subtitle = 'Based on PISA Score 2015') +
theme(axis.text=element_text(size=20),
        plot.title = element_text(size = 20, face = "bold"),
        plot.subtitle = element_text(size = 10),
        plot.caption = element_text(color = "Red", face = "italic",
size = 13)
)
```

Awesome!! Now the plot looks really informative. However, better data analysts would produce something even more informative like the following:

How can we let R identify the outliers and label them for us?

```
df6$Test = factor(df6$Test, levels = c("Maths","Science","Reading"))
# To change order of legend
```

Let's define the outlier. This part requires a bit of Statistics knowledge. I would recommend you to read Michael Galarnyk's article here. He explains the concepts pretty

well.

```
is_outlier <- function(x) {
   return(x < quantile(x, 0.25) - 1.5 * IQR(x) | x > quantile(x,
0.75) + 1.5 * IQR(x))
}   # define a function to detect outliers

str(df6)
```

Notice that the columns ‘Country’ and ‘Test’ are factors. First let’s change it to characters.

```
df6$Country = as.character(df6$Country)
df7 <- df6  %>% group_by(as.character(Test)) %>%
  mutate(is_outlier=ifelse(is_outlier(Result), Country,
as.numeric(NA)))

### What we are doing now is that we are creating a new data frame
with the last column 'is_outlier' indicating whether the data point
is an outlier or not

View(df7)
```

As you can see, the last column of the data frame shows that Jordan is an outlier. Now, at the 'Country' column, we want to make sure that only the outliers are labeled, the rest should be put as 'NA'. This will be helpful for us later when we code the visualization plot.

```
df7$Country[which(is.na(df7$is_outlier))] <- as.numeric(NA)
View(df7)
```

Now, let's plot the graph. The same code as above. However we need to add in **geom_text ()** to label the outliers

```
ggplot(data = df7, aes(x=Test,y=Result, fill=Test)) +
  geom_boxplot(alpha = 0.7,
               outlier.colour='red',
               outlier.shape=19,
               outlier.size=3,
               width = 0.6)+
  geom_text(aes(label = Country), na.rm = TRUE, hjust = -0.2)+
  theme_grey() +
  labs(title = 'Did males perform better than females?',
       y='% Difference from FEMALES',x='',
       caption  = 'Positive % Difference means Males performed \n
better than Females and vice versa',
       subtitle = 'Based on PISA Score 2015') +
  theme(axis.text=element_text(size=20),
        legend.text = element_text(size = 16),
        legend.title = element_text(size = 16),

        legend.position = 'right', aspect.ratio = 1.4,
        plot.title = element_text(size = 20, face = "bold"),
        plot.subtitle = element_text(size = 10),
        plot.caption = element_text(color = "Red", face = "italic",
size = 13)
  )
```

Combine multiple plots

For each of the plot in the combined plot above, we have gone through how to create it in Part 1 of this series. Here is the recap:

```
plot1 = ggplot(data=df, aes(x=reorder(Country.Name, Maths.Diff),
y=Maths.Diff)) +
  geom_bar(stat = "identity", aes(fill=Maths.Diff)) +
  coord_flip() +
  theme_light() +
  geom_hline(yintercept = mean(df$Maths.Diff), size=1, color="black")
+
  labs(x="", y="Maths")+
  scale_fill_gradient(name="% Difference Level", low = "red", high =
"green")+
  theme(legend.position = "none")

plot2 = ggplot(data=df, aes(x=reorder(Country.Name, Reading.Diff),
y=Reading.Diff)) +
  geom_bar(stat = "identity", aes(fill=Reading.Diff)) +
  coord_flip() +
  theme_light() +
  geom_hline(yintercept = mean(df$Reading.Diff), size=1,
color="black") +
  labs(x="", y="Reading")+
  scale_fill_gradient(name="% Difference Level", low = "red", high =
"green") +
  theme(legend.position = "none")

plot3 = ggplot(data=df, aes(x=reorder(Country.Name, Science.Diff),
y=Science.Diff)) +
  geom_bar(stat = "identity", aes(fill=Science.Diff)) +
  coord_flip() +
  theme_light() +
  geom_hline(yintercept = mean(df$Science.Diff), size=1,
color="black") +
  labs(x="", y="Science")+
  scale_fill_gradient(name="% Difference", low = "red", high =
"green") +
  theme(legend.position = "none")
```

To combine them all together, use ‘gridExtra’ package.

```
install.packages('gridExtra')
library(gridExtra)

grid.arrange(plot1, plot2,plot3, nrow = 1,
             top = 'Are Males better than Females?',
```

```
                bottom = '% Difference from Females'
                )

  #nrow=1 means all the plots are placed in one row
```

That is it! Again, I hope you guys enjoyed and picked up something from this article. Of course, this guide is not exhaustive, and there are a lot of other techniques we can use to do EDA. However, I believe this guide will more or less give you some ideas of how to do **improve from a simple plot to a more complicated and informative plot with R**.

If you have any questions, feel free to put them down in the comment section below. Once again, thank you for your read. Have a great day and happy programming!!!
