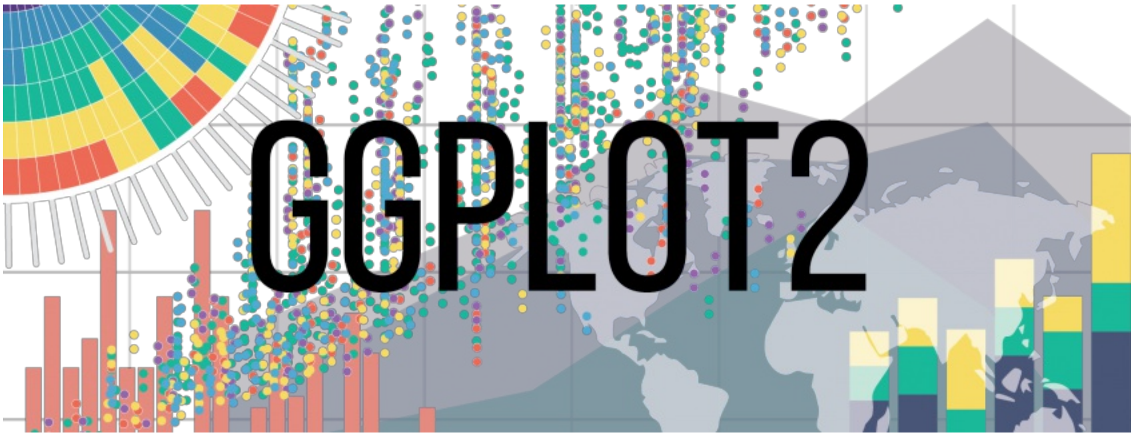
post02 - mastering ggplot2

Yongtae Lee

12/03/2017

Introduction

- This post serves to learn deeply about data visualization through the most widely used R library: ggplot2. When confronted with enormous data with thousands of rows, it is necessary to import the data in an appropriate way and analyze this massive raw text file by using effective graphic tools. Throughout this blog post, we will employ multiple advanced data visualizing methods from ggplot2 to make the confusing data to be interpretable. So let's begin our journey!



Getting started

Download required packages

At first, it is important to download required packages that assist with analyzing data. There are four packages needed to analyze the data: readr, dplyr, ggplot2, ggExtra.

- The readr package helps with importing data files and putting adjustments on them. To learn more details, please check the following link: https://cran.r-project.org/web/packages/readr/index.html
- The dplyr package allows us to do some data wrangling. To learn more details, please check the following link: https://cran.r-project.org/web/packages/dplyr/index.html
- The gglot2 package, the main package we will be exploring with, is essential to present data visualization. It has various features to produce graphics. To learn more details, please check the following link: https://cran.r-project.org/web/packages/ggplot2/index.html
- The ggExtra package is the essential tool for putting extra features onto ggplot2. To learn more details, please check the following link: https://cran.r-project.org/web/packages/ggExtra/index.html

Also, it is very crucial to load them by using library:

```
# loading packages by using library
library(readr)    # importing data
library(dplyr)    # data wrangling
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)  # graphics
library(ggExtra)  # essential for putting extra features onto ggplot2
```

Importing the data

The dataset that we will be working with can be found in Stat 133 github repository: https://github.com/ucb-stat133/stat133-fall-2017/blob/master/data/nba2017-roster.csv

Our dataset 'nba2017-roster.csv' has information about the NBA player with the details of team name, position, height, weight, age, experience, salary. Download the data with the following lines of code, and save it as 'nba-roster.csv':

```
github <- "https://raw.githubusercontent.com/ucb-stat133/stat133-fall-2017/master/"
data <- "data/nba2017-roster.csv"
download.file(url = paste0(github,data), destfile = 'nba2017-roster.csv')
```

Here, we'll be reading the data by using read_csv from readr package.

```
# importing the data nba2017-roster.csv
roster <- read_csv('nba2017-roster.csv',
                  # assigning column types for each column
                  col_types = cols(
                   team = col_character(),
                   position = col_character(),
                   height = col_integer(),
                   weight = col_integer(),
                   age =  col_integer(),
                   experience = col_integer(),
                   salary = col_integer()
                  ))
roster
```

```
## # A tibble: 395 x 8
##               player  team position height weight   age experience   salary
##                <chr> <chr>    <chr>  <int>  <int> <int>      <int>    <int>
##  1    A.J. Hammons   DAL        C     84    260    24          0   650000
##  2    Aaron Brooks   IND       PG     72    161    32          8  2700000
##  3    Aaron Gordon   ORL       SF     81    220    21          2  4351320
##  4   Adreian Payne   MIN       PF     82    237    25          2  2022240
##  5      Al Horford   BOS        C     82    245    30          9 26540100
##  6    Al Jefferson   IND        C     82    289    32         12 10230179
##  7 Al-Farouq Aminu   POR       SF     81    220    26          6  7680965
##  8   Alan Anderson   LAC       SF     78    220    34          7  1315448
##  9   Alan Williams   PHO        C     80    260    24          1   874636
## 10      Alec Burks   UTA       SG     78    214    25          5 10154495
## # ... with 385 more rows
```

Exploring the data

Yes! We have now successfully loaded our data. In order to find usefulness of ggplot2, we will be creating various visuals with the package.

Today, we will be producing four creative graphic features onto this data source:

1. Two variables - continuous variables
2. Two variables - discrete variable vs. continuous variable
3. Two variables - marginal graphs
4. Three variables - heat map

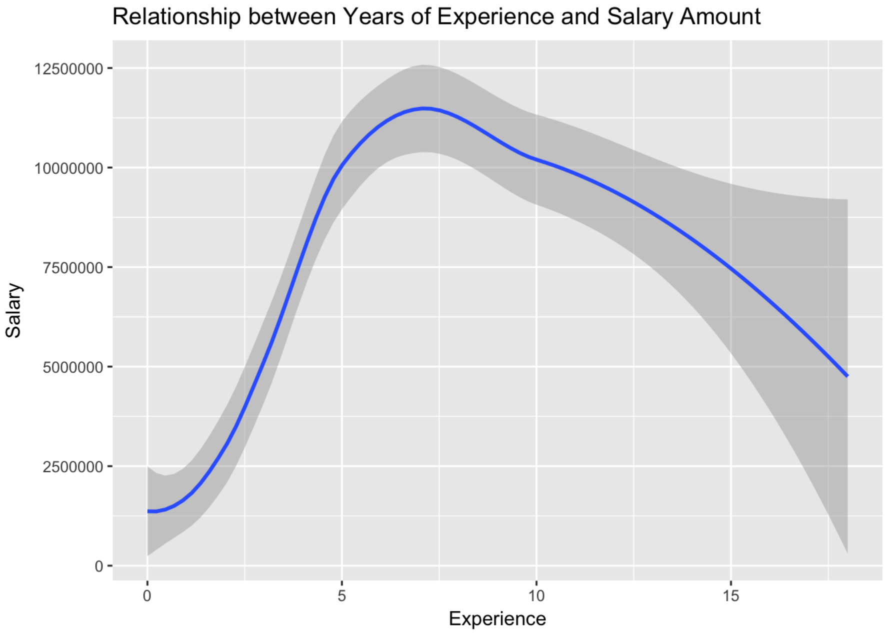
Two variables - continuous variables

One of the most efficient way of creating a graphic visual between two continuous variables is using a smoothing plot Here, let's try to find a relationship between years of experience and salary amount.

```
# creating a smoothing plot
# relationship between years of experience and salary amount
g1 <- ggplot(roster, aes(x=experience, y = salary)) + # putting experience as x-variable and salary as y-variable
  geom_smooth() + # applying smoothing plot
  labs(title="Relationship between Years of Experience and Salary Amount",
       x="Experience",
       y="Salary")  # modifying axis, legend, and plot labels

g1
```

```
## `geom_smooth()` using method = 'loess'
```



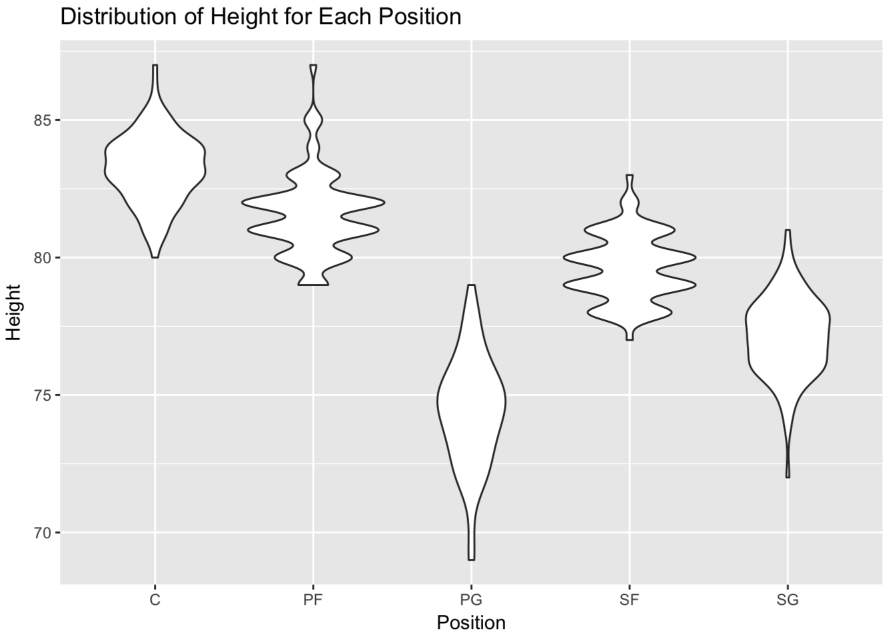
From the smoothing graph, we can observe that experience has a positive correlation with salary until about 7-8 years of experience, and consequently shows a negative correlation afterward.

Two variables - discrete variable vs. continuous variable

Now, how do we create a graph of a discrete variable and a continuous variable then? Here, we can use a violin plot from ggplot2 - it presents the distribution of the dependent variable most easily. Let us try an experiment with position as x-variable and height as y-variable.

```
# creating a violin plot
# distribution of height for each position
g2 <- ggplot(roster, aes(x=position, y = height)) + # putting position as x-variable and height as y-variable
  geom_violin() + # applying violin plot
  labs(title="Distribution of Height for Each Position",
       x="Position",
       y="Height")  # modifying axis, legend, and plot labels

g2
```



Here, we can easily observe that players in center and power forward positions are relatively taller while point guards are relatively shorter.

More information about violin plot can be found from the following link: http://www.sthda.com/english/wiki/ggplot2-violin-plot-quick-start-guide-r-software-and-data-visualization

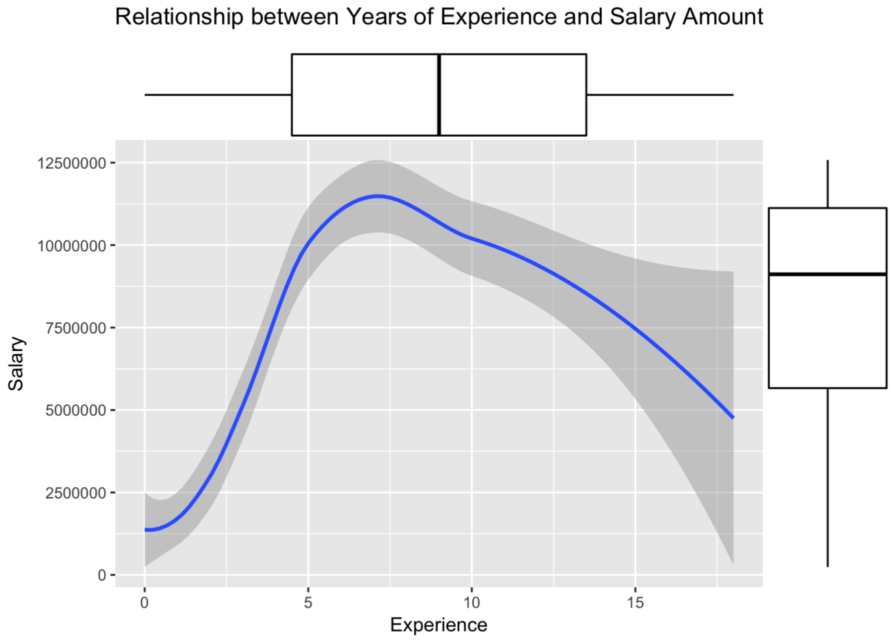
Two variables - marginal graphs

Furthermore, if you want to present marginal graphs at x-axis and y-axis of the main graph, you can easily do that in the following way. Let's try to add them from the first example:

```
# adding boxplots on x-axis and y-axis after creating a violin plot
# relationship between years of experience and salary amount
g3 <- ggplot(roster, aes(x=experience, y = salary)) + # putting experience as x-variable and salary as y-variable
  geom_smooth() + # applying smoothing plot
  labs(title="Relationship between Years of Experience and Salary Amount",
       x="Experience",
       y="Salary")  # modifying axis, legend, and plot labels

ggMarginal(g3, type = "boxplot", fill="transparent") # creating boxplots on x-axis and y-axis
```

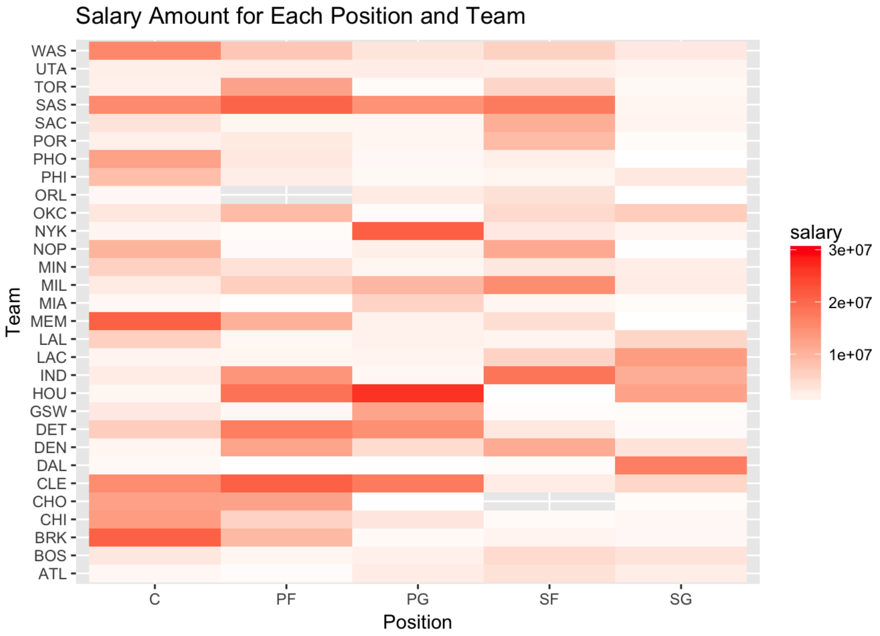
```
## `geom_smooth()` using method = 'loess'
## `geom_smooth()` using method = 'loess'
## `geom_smooth()` using method = 'loess'
```



Three variables

Now, we are going to create a plot of three variables using heat map. We will be using a ‘geom_raster’ graphic feature from ggplot2 to observe the result: position as x-axis, team as y-axis, and salary as gradient of color for each box.

```
# heat map of salary amount for each position and team
g4 <- ggplot(roster, aes(x = position, y = team)) + # assigning position as x-axis and team as y-axis
  geom_raster(aes(fill = salary)) + # filling each box by salary
  scale_fill_gradient(low = "white", high = "red") + # assigning colors to fill each box
  labs(title="Salary Amount for Each Position and Team",
       x = "Position",
       y = "Team") # modifying axis, legend, and plot labels
g4
```



From this heat map, we can easily find that Houston’s point guards have the highest salary among the all positions and teams.

More information about heat map can be found from the following link: https://learnr.wordpress.com/2010/01/26/ggplot2-quick-heatmap-plotting/

Putting all together

From these four graphics, we have learned that some meaningful observations can be successfully made by using various features of ggplot2. In fact, there are more tools available in the package. Many researchers and students produce advanced graphics with ggplot2 and support their arguments. The following link is a cheat sheet for ggplot2 provided by RStudio: https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

I hope this post helped you better understand graphic features available in R. You can also apply your own data and produce many other graphics! Take advantage of this amazing toolkits provided by R and conduct your own experiments. Data visualization with ggplot2 is not only efficient, but also very fun!

References

- ggplot2 cheatsheet: https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf
- readr package: https://cran.r-project.org/web/packages/readr/index.html
- dplyr package: https://cran.r-project.org/web/packages/dplyr/index.html
- gglot2 package: https://cran.r-project.org/web/packages/ggplot2/index.html
- ggExtra package: https://cran.r-project.org/web/packages/ggExtra/index.html
- ggplot2 violin plot: http://www.sthda.com/english/wiki/ggplot2-violin-plot-quick-start-guide-r-software-and-data-visualization
- ggplot2 heatmap: https://learnr.wordpress.com/2010/01/26/ggplot2-quick-heatmap-plotting/
- image: https://datasciencedojo.com/wp-content/uploads/12_R-and-GGPLOT2-01-845x321.png