post02

Joanne Chen

11/24/2017

The Data Preparation Mini Cycle:

A Closer Look At Exploratory Data Analysis and Cleaning

Introduction

Most of us have heard of the data analysis cycle—the buzz phrase that has gotten our modern society so excited. A broad overview of this cycle was given in my previous post; it outlined the major steps taken by data analysts and included some quick coding examples. This blog post will now delve a bit deeper into one of the phases of the data analysis cycle and discuss it in more detail. Data preparation, one of the most important phases of data analysis, occurs before the analysis itself is even executed. This includes evaluating the quality of the data at hand, and subsequently, tidying and reshaping the data to give us a clean data set that we can use reproducibly. For this to be accomplished, a certain amount of exploration is required for us to get to know the data we are working with in the first place. This process is called EDA: exploratory data analysis. EDA is then followed by the restructuring and reshaping of the data into a usable format; the process of data cleaning. This mini cycle of cleaning, EDA, and cleaning again occurs several times before the larger cycle of data analysis is recommenced. For the purposes of this blog post, I will assume that our data has already been collected with proper sampling procedures and that the information lives inside our directories, ready to be worked with.

So what, specifically, does EDA entail? The EDA process is not so much a defined set of statistical techniques as it is an attitude with which one approaches a data set. The philosophy of EDA is to allow the raw data to speak for itself—to reveal its underlying characteristics and structure. Our hope is that these underlying characteristics will inform us of the validity of our preliminary assumptions, the relationships between explanatory variables, and the surface level “dirt” that we should clean up. While this is typically accomplished using commonly seen statistical techniques and graphics, it is important to understand, before we delve into specific techniques, that these techniques are not a defined rule book to flip through and follow.

Example: Bike Riders

I will now dive into specific examples of what is typically done by data scientists to investigate a piece of data. Below I have loaded in a data set about bike sharing in Washington D.C. during different seasons in the years 2011 and 2012. Each record in the table below is a snapshot in time containing information about the hour and day, weather, temperature, humidity, wind speed, and the number of casual or registered bike riders sighted in that hour.

```
bike = read.csv("data/bikeshare.txt")
head(bike)
```

```
##   instant     dteday season yr mnth hr holiday weekday workingday
## 1       1 2011-01-01      1  0    1  0       0       6          0
## 2       2 2011-01-01      1  0    1  1       0       6          0
## 3       3 2011-01-01      1  0    1  2       0       6          0
## 4       4 2011-01-01      1  0    1  3       0       6          0
## 5       5 2011-01-01      1  0    1  4       0       6          0
## 6       6 2011-01-01      1  0    1  5       0       6          0
##   weathersit temp  atemp  hum windspeed casual registered cnt
## 1          1 0.24 0.2879 0.81    0.0000      3         13  16
## 2          1 0.22 0.2727 0.80    0.0000      8         32  40
## 3          1 0.22 0.2727 0.80    0.0000      5         27  32
## 4          1 0.24 0.2879 0.75    0.0000      3         10  13
## 5          1 0.24 0.2879 0.75    0.0000      0          1   1
## 6          2 0.24 0.2576 0.75    0.0896      0          1   1
```

Some Cleaning

As you can see, many of the variables that are recorded as integer/numeric should be replaced with words that are informative. These include weekday, holiday, and other variables. We will go ahead and mutate the table so that its information is more understandable.

```
# Replace holiday column
no = replace(bike$holiday, bike$holiday==0, 'no')
bike["holiday"] = replace(no, no==1, "yes")

# Replace working day column
no = replace(bike$workingday, bike$workingday==0, 'no')
bike["workingday"] = replace(no, no==1, "yes")

# Replace weekday column
new = replace(bike$weekday, bike$weekday==0, "Sun")
new = replace(new, new==1, "Mon")
new = replace(new, new==2, "Tue")
new = replace(new, new==3, "Wed")
new = replace(new, new==4, "Thu")
new = replace(new, new==5, "Fri")
new = replace(new, new==6, "Sat")
bike["weekday"] = new

# Replace weathersit column
new = replace(bike$weathersit, bike$weathersit==1,"Clear")
new = replace(new, new==2, "Mist")
new = replace(new, new==3, "Light")
new = replace(new, new==4, "Heavy")
bike["weathersit"] = new

# View new columns
head(bike[,7:10], 10)
```

```
##    holiday weekday workingday weathersit
## 1       no     Sat         no      Clear
## 2       no     Sat         no      Clear
## 3       no     Sat         no      Clear
## 4       no     Sat         no      Clear
## 5       no     Sat         no      Clear
## 6       no     Sat         no       Mist
## 7       no     Sat         no      Clear
## 8       no     Sat         no      Clear
## 9       no     Sat         no      Clear
## 10      no     Sat         no      Clear
```

Now that our records are more reader friendly, let's start exploring and modelling the data. Because humans are not efficient at identifying patterns and important characteristics from looking at rectangular data (columns of numbers in a table or spreadsheet), we should use non-tabular techniques to elucidate certain aspects of the data while hiding other less important parts. EDA is typically classified as graphical or non-graphical. Let's start by exploring some non-graphical methods.

Non Graphical EDA

Non-graphical methods typically include summary statistic calculations, the characteristics of interest being value frequencies for qualitative variables, and center / spread / shape for quantitative variables. For our quantitative variables (e.g. temperature, wind speed, casual, registered, etc.), we can use summary statistics functions to find their means, medians, and ranges:

```
# Stats for Casual Bike Riders
summary(bike$casual)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    4.00   17.00   35.68   48.00  367.00
```

```
sd(bike$casual)
```

```
## [1] 49.30503
```

```
# Stats for Registered Bike Riders
summary(bike$registered)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0    34.0   115.0   153.8   220.0   886.0
```

```
sd(bike$registered)
```

```
## [1] 151.3573
```

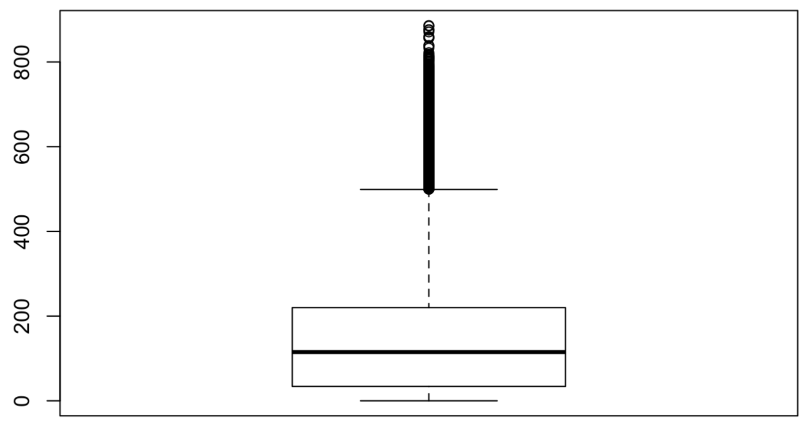
We know that a distribution is symmetric when its mean and median are equal. Therefore, we can see from just these simple summary statistics that both distributions for casual and registered riders are asymmetric and right skewed. How do we know they are right skewed? Both means are greater than the respective medians, indicating that there is a tail to the right that pulls the mean out.

Graphical EDA

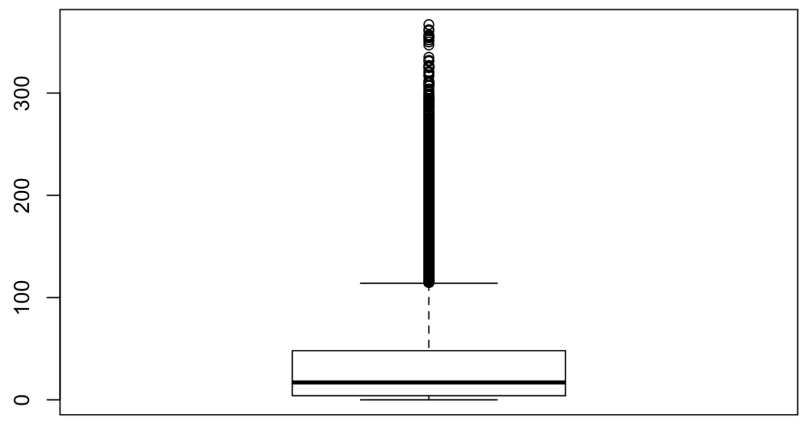
`Boxplots`

Now let's turn to graphical methods. The distributions above could easily be further investigated through box plots and histograms. A boxplot is an extremely useful univariate (single variable) modelling tool that allows users to visualize outliers and skewness in a distribution. Our speculations about skewness are corroborated in the boxplots below, which clearly show the presence of large outliers and extreme right skewness.

```
boxplot(bike$registered)
```



```
boxplot(bike$casual)
```



`Line Plots`

We can also use EDA to visualize multivariate relationships (in this case, bivariate). The model below plots the average number of riders recorded per hour of day, for both casual (blue line) and registered (red line) riders. One can see that regardless of the time of day, the number of registered riders in D.C. is consistently greater than the number of casual riders. What's even more interesting: the number of registered riders dips when that of casual riders peaks. Why might this be? Registered rider counts seem to peak around 8-9 am and 5-6 pm—the times when people typically start and end their workdays. We could hypothesize that registered riders bike for the purpose of commuting to and from work, while casual riders bike more for non-work related purposes.

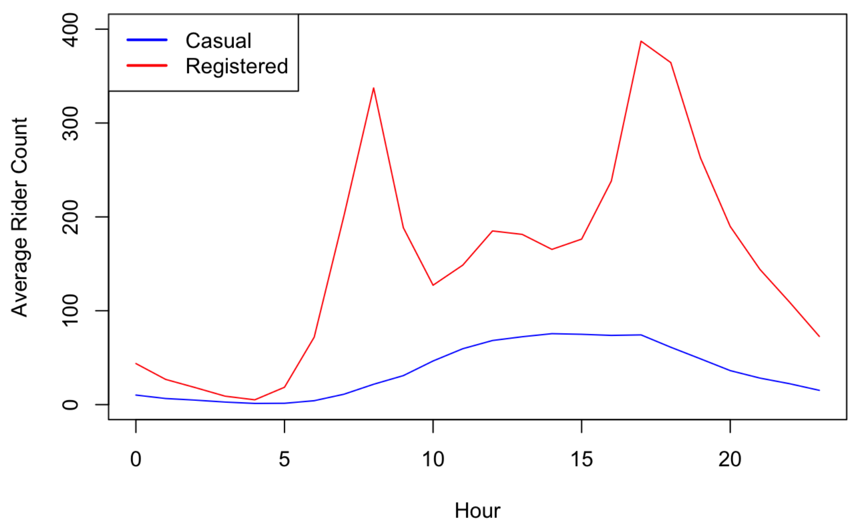
```
library(dplyr)
```

```
##
## 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
```

```r
# Group bike data by hour
casual = summarise(group_by(bike, hr), mean=mean(casual))
registered = summarise(group_by(bike, hr), mean=mean(registered))

# First plot the distribution of casual riders per hour.
plot(casual$hr, casual$mean, type="l", ylim = c(0,400), xlab = "Hour", ylab = "Average Rider Count", col="blue")
# Overlay this plot with the registered riders line plot.
lines(casual$hr, registered$mean, col="red")
# Include color legend
legend("topleft", legend=c("Casual","Registered"), lwd=c(2,2), col=c("blue","red"))
```



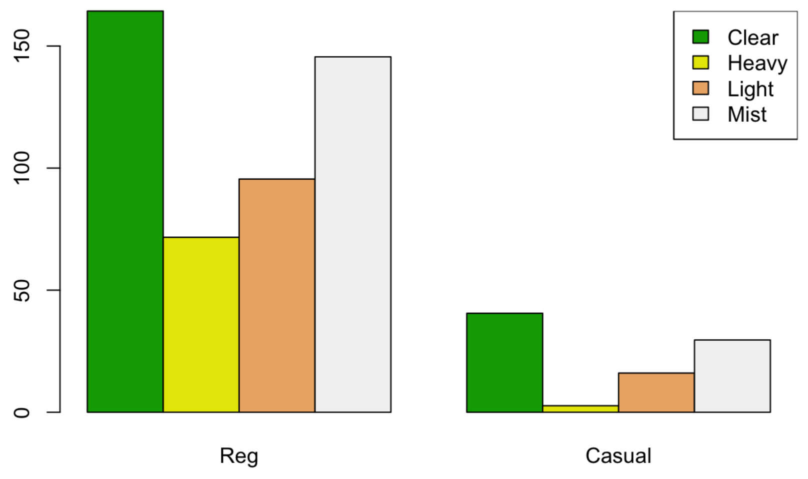
`Bar Plots`

Below is another plot that compares casual and registered rider distributions. This time, we are examining how the two distributions vary based on the weather situation variable, “weathersit”. As expected, the greatest number of riders for both registered and casual come out when the weather is clear (green bar) and the least come out when there is heavy rain. An interesting observation is that the ratio between heavy and clear appears much greater for registered riders (the yellow bar in the left bar plot reaches nearly half of the height of the green bar)! Meanwhile, the yellow bar in the casual rider bar plot is barely a tenth of the green bar’s height. This implies that many registered riders bike regardless of how terrible the weather looks, perhaps because they paid for the bikes or because biking is their only means of transportation to their workplace.

```r
registered2 = summarise(group_by(bike, weathersit), mean=mean(registered))
casual2 = summarise(group_by(bike, weathersit), mean=mean(casual))
merged = merge(registered2,casual2, by="weathersit")

# Create overlaid barplots for casual and registered riders
barplot(as.matrix(merged[,2:3]), width=2,beside = TRUE, names.arg = c("Reg","Casual"), col = terrain.colors(4))

# Include color legend
legend("topright", fill=terrain.colors(4), legend=c("Clear","Heavy","Light","Mist"))
```



Conclusion

This is just the beginning of some very intriguing speculations that would be further investigated when we move on from the EDA-cleaning mini cycle to the actual analysis. The point is, EDA is a philosophy that can come in a variety of forms—whether it is through graphical visualizations, non-graphical observations, or simply playing around which data tables. The main takeaway message for this blog post is not to detect every minute pattern there is in the dataset I provided you, but rather, to begin to see the power of computational tools when combined with an open mindset of exploration.

References

- Seltman, Howard. “Exploratory Data Analysis.” 2016.
- “What is EDA?” Engineering Statistics Handbook, NIST/SEMATECH, 1 June 2003.
- “Creating Stacked Barplot and Grouped Barplot in R using Base Graphics (No ggplot2).” The Coatless Professor, The Coatless Professor, 13 Apr. 2014.
- Gougeon, D. J. (2007). Making sense of data: A practical approach to exploratory data analysis and data mining. Choice, 44(9), 1567.
- Yeager, Ronnie L., et al. “Graphical methods for exploratory analysis of complex data sets.” BioScience, vol. 57, no. 8, 2007, p. 673+. Science in Context, link.galegroup.com/apps/doc/A169410676/SCIC?u=acalaneshs&xid=a05d82d5. Accessed 25 Nov. 2017.
- Wrigley, Neil. “Exploratory Data Analysis.” Area, vol. 11, no. 1, 1979, pp. 18–19. JSTOR, JSTOR.
- Data Science 100 Course Material (STATS/CS c100)