Data Analysis Using Detailed and Comprehensive Plots via the Ggplot2 Package

Samantha Nguyen
11/27/2017

Introduction

R is a programming language used for statistical computing and data analysis. Simple plots can be made with R's built-in plotting features, and a quick and concise look at how to make those kinds of visuals can be found at https://www.statmethods.net/graphs/index.html. However, these plots are very simple and cannot be adjusted by much. R has many different packages that can be utilized to display much more detailed graphs and plots. An easy to use and comprehensive package is the **ggplot2** package. It can be used to make simple and straightforward diagrams, such as barcharts, histograms, and scatterplots. However, there are many other functions in **ggplot2** that can allow users to create beautiful and detailed visuals that help with analyzing data thoroughly.

This post's main purpose is to guide readers toward a deeper understanding of the methods in **ggplot2** to teach them how to use the many layer functions this package has that can transform simple visuals to more comprensive ones. It includes a step-by-step tutorial that goes through how to install the package into RStudio and then how to actually utilize the package with your data. Examples will be used to illustrate a visual approach on how to apply the package on actual data sets. The specific graphs this tutorial will focus on will be lollipop graphs and proportional stacked area graphs.

A concise description on how to use the **ggplot2** package is summed up in a Ggplot2 cheatsheet found here: https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf.

This post will follow the order of:

1. A tutorial on how to install the package into users' RStudio and load the data that will be used in the following guide to learning ggplot2.
2. A step-by-step guide on how to use **ggplot2** with the package's selection of built-in datasets in building a lollipop graph.
3. A step-by-step guide on how to use **ggplot2** with a loaded data set to create a proportional stacked area graph.
4. A summary of what can be learned from this post and an analysis of how this information can be used.
5. A reference list of what sources were used in this post.

Follow the tutorial below for a more comprehensive step-by-step tutorial with a separate example for increased ease in learning how to use the Ggmap package.

Tutorial: Setting up the package and data in R

Step 1: How to download the packages necessary to use ggmap

Downloading **ggplot2** consists of first installing the packages **ggplot2** into your console. You can do this with the following command:

```
# the basic package that you will use to create your map/data visualizations
install.packages("ggplot2", dependencies = TRUE)
install.packages("dplyr")
instlal.packages("gcookbook")
```

You then need to load the actual packages into your R Markdown file, using the following commands in your Rmd code chunk. These packages will help in parsing through the data and also include one of the data sets we will be using.

```
library(ggplot2)
library(plyr)
library(dplyr)
```

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

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

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

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

```
library(gcookbook)
```

Step 2: How to load your data

For this guide, we will use the built-in package *mpg* which gives the Fuel economy data from 1999 and 2008 for 38 popular models of car. More built-in packages that come with **ggplot2** can be found at http://ggplot2.tidyverse.org/reference/ in the Data section.

Let's examine the data a bit to see what we will be working with and analyzing:

```
# show the first six rows of the data table mpg
head(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl      trans   drv   cty   hwy    fl
##          <chr> <chr> <dbl> <int> <int>      <chr> <chr> <int> <int> <chr>
## 1         audi    a4   1.8  1999     4   auto(l5)     f    18    29     p
## 2         audi    a4   1.8  1999     4 manual(m5)     f    21    29     p
## 3         audi    a4   2.0  2008     4 manual(m6)     f    20    31     p
## 4         audi    a4   2.0  2008     4   auto(av)     f    21    30     p
## 5         audi    a4   2.8  1999     6   auto(l5)     f    16    26     p
## 6         audi    a4   2.8  1999     6 manual(m5)     f    18    26     p
## # ... with 1 more variables: class <chr>
```

Step-by-Step Guide: How to display comprehensive visuals using Ggplot2

Step 1: How to put the data into a simple ggplot2 graph

For the purpose of the graph we want to make, we need to retrieve the top 15 cars with the best highway mpg out of all the cars we have the data for. We can do this with the following command:

```
# select the top 15 cars with the most hwy mpg
top35Cars <- mpg %>%
        select(manufacturer, model, year, hwy) %>%
        arrange(desc(hwy)) %>%
        top_n(35)
```

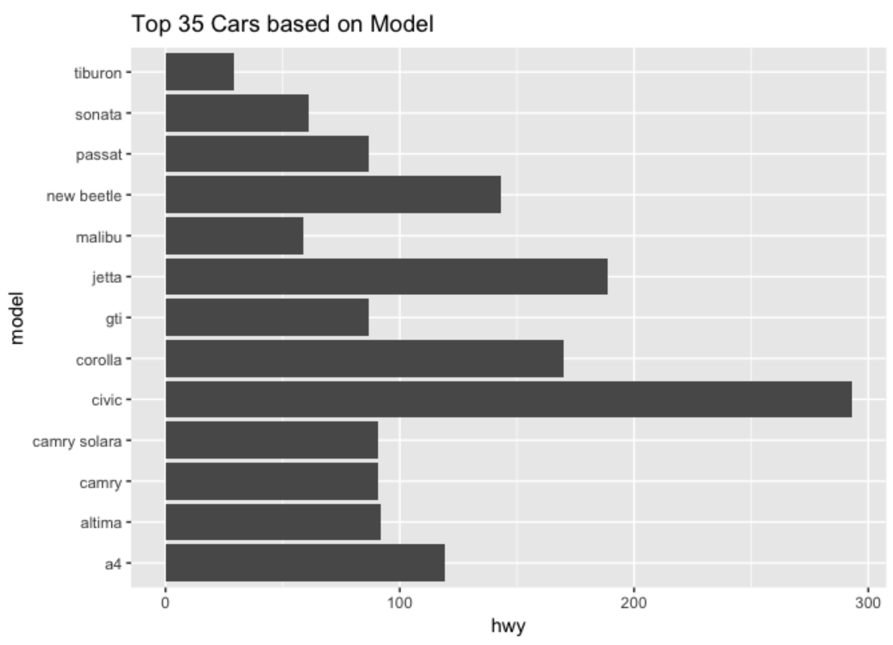
```
## Selecting by hwy
```

```
top35Cars
```

```
## # A tibble: 48 x 4
##    manufacturer      model  year   hwy
##           <chr>      <chr> <int> <int>
##  1   volkswagen      jetta  1999    44
##  2   volkswagen new beetle  1999    44
##  3   volkswagen new beetle  1999    41
##  4       toyota    corolla  2008    37
##  5        honda      civic  2008    36
##  6        honda      civic  2008    36
##  7       toyota    corolla  1999    35
##  8       toyota    corolla  2008    35
##  9        honda      civic  2008    34
## 10        honda      civic  1999    33
## # ... with 38 more rows
```

This data could easily be put into a horizontal bar chart of models of cars vs highway mpg using a simple ggplot2 method:

```
# top15cars bar chart
ggplot(top35Cars, aes(model, hwy)) + geom_bar(stat = "identity") + coord_flip() + ggtitle("Top 35 Cars base
```

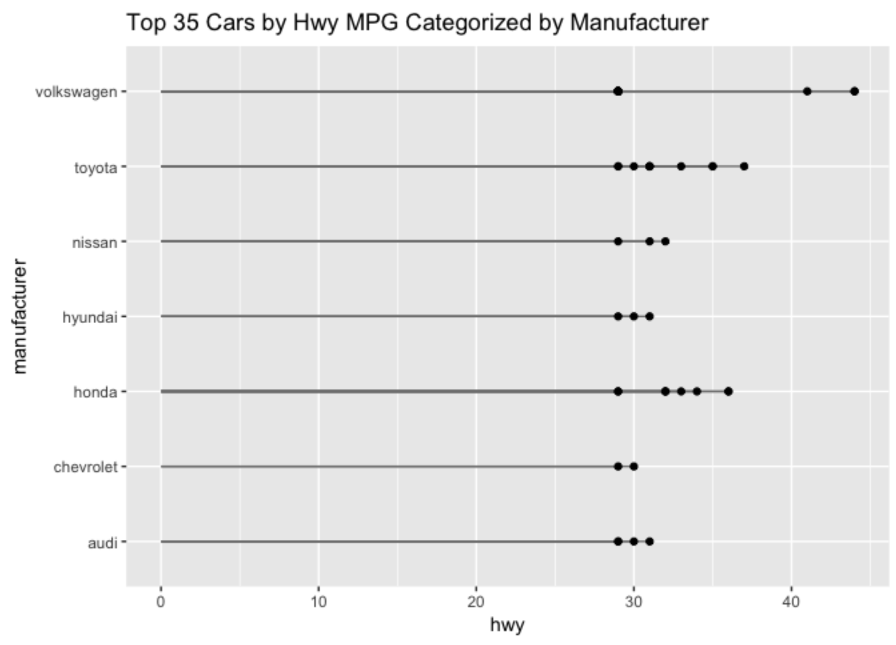


This is a simple horizontal bar chart that shows a very general comparison between the many different cars from the *mpg* data set. Though it can be useful in some analysis, much more detailed analysis can be made with other charts. We will see this in Step 2.

Step 2: Making a lollipop chart to easily compare data and see the differences within points of data

A lollipop graph is a great way to view data in a clearer way than a bar chart when you are working with a lot of data as there are clear lines that go from the y-axis labels to the number correlated with each y-axis point. For example, when looking at the top 35 cars, we can separate them by manufacturer and see who has the best highway milelage comparatively:

```
# simple lollipop graph
ggplot(top35Cars, aes(hwy, manufacturer)) +
        geom_segment(aes(x = 0, y = manufacturer, xend = hwy, yend = manufacturer), color = "grey50") +
        geom_point() + ggtitle("Top 35 Cars by Hwy MPG Categorized by Manufacturer")
```



Step 3: Making a more complex lollipop chart

When comparing data, sometimes you may find that you want to find the averages within data if you are working with large sets of data as a form of comparison with the individual units of data. Using the lollipop graph, you can do just that!

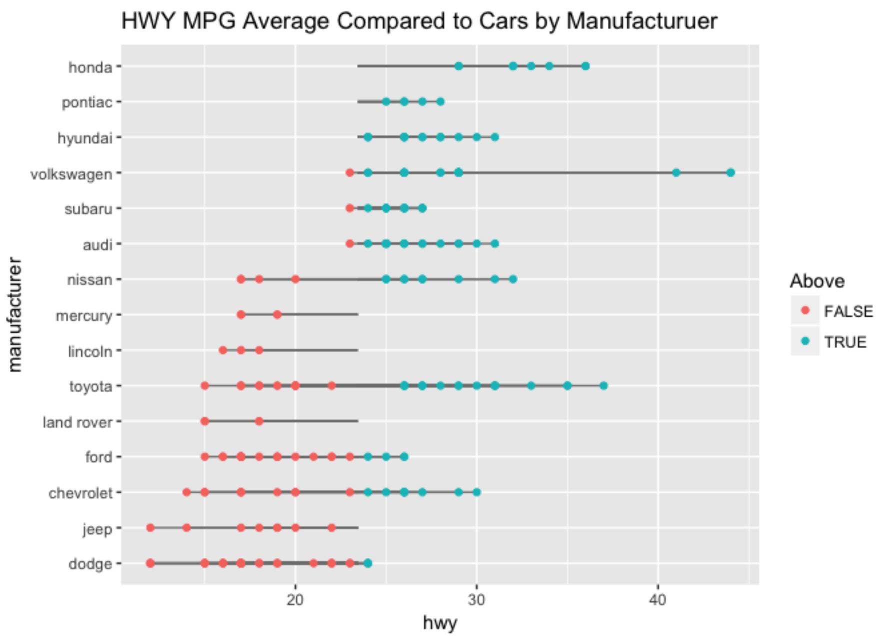
Using our *mpg* data, we can find the hwy average and see how the car manufacturers compare with this average.

```
# create data table with the average of Highway MPGs included
carCompanies <- mpg %>%
        select(manufacturer, hwy) %>%
        arrange(hwy) %>%
        mutate(Avg = mean(hwy, na.rm = TRUE),
               Above = ifelse(hwy - Avg > 0, TRUE, FALSE),
               manufacturer = factor(manufacturer, levels = .$manufacturer))
```

```
## Warning in `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels)
## else paste0(labels, : duplicated levels in factors are deprecated
```

```
# create lollipop chart using geom_segment
ggplot(carCompanies, aes(hwy, manufacturer, color = Above)) +
        geom_segment(aes(x = Avg, y = manufacturer, xend = hwy, yend = manufacturer), color = "grey50") +
        geom_point() + ggtitle("HWY MPG Average Compared to Cars by Manufacturuer")
```

```
## Warning in `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels)
## else paste0(labels, : duplicated levels in factors are deprecated

## Warning in `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels)
## else paste0(labels, : duplicated levels in factors are deprecated

## Warning in `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels)
## else paste0(labels, : duplicated levels in factors are deprecated

## Warning in `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels)
## else paste0(labels, : duplicated levels in factors are deprecated

## Warning in `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels)
## else paste0(labels, : duplicated levels in factors are deprecated

## Warning in `levels<-`(`*tmp*`, value = if (nl == nL) as.character(labels)
## else paste0(labels, : duplicated levels in factors are deprecated
```



This graph shows the car manufacturers that are either above or under the highway average by using clear colors that separate the difference. More information on the specific variables *geom_segment* takes in can be found at http://ggplot2.tidyverse.org/reference/geom_segment.html. This will explain why *xend* and *yend* were used to show how the data points were plotted.

For more practice building lollipop charts, UC Business Analytics R Programming Guide at http://uc-r.github.io/lollipop provides an excellent example as well using the built-in data *midwest* that you can use to practice creating charts with.

The main page, http://uc-r.github.io/ggplot_intro, provides many other examples of using **ggplot2** and includes great samples of types of graphs that can be created.

Tutorial 2: Making a Proportional Stacked Area Graph

Tufts University provides a great and concise tutorial of how to create a proportional stacked area graph at https://ase.tufts.edu/bugs/guide/assets/R%20Graphics%20Cookbook.pdf.

In our tutorial, we will be making a similar graph but will also experiment with the other kinds of features **ggplot2** includes. The data used by Tufts is from the **gcookbook** package. This package contains mainy different data sets that are come in excellent use when practicing analyzing data in R. The other data sets that are found in this package can be looked at here https://github.com/wch/gcookbook/tree/master/data.

Step 1:

First, we should always know what kind of data we are working with. In this tutorial, we will be looking at the population of the United States from 1900 to 2000 and categorized by age group. This can be easily found with using the *head* function that displays the first 6 rows of data.

```
# examine what kind of data we will be working with
head(uspopage)
```

```
##   Year AgeGroup Thousands
## 1 1900       <5      9181
## 2 1900     5-14     16966
## 3 1900    15-24     14951
## 4 1900    25-34     12161
## 5 1900    35-44      9273
## 6 1900    45-54      6437
```

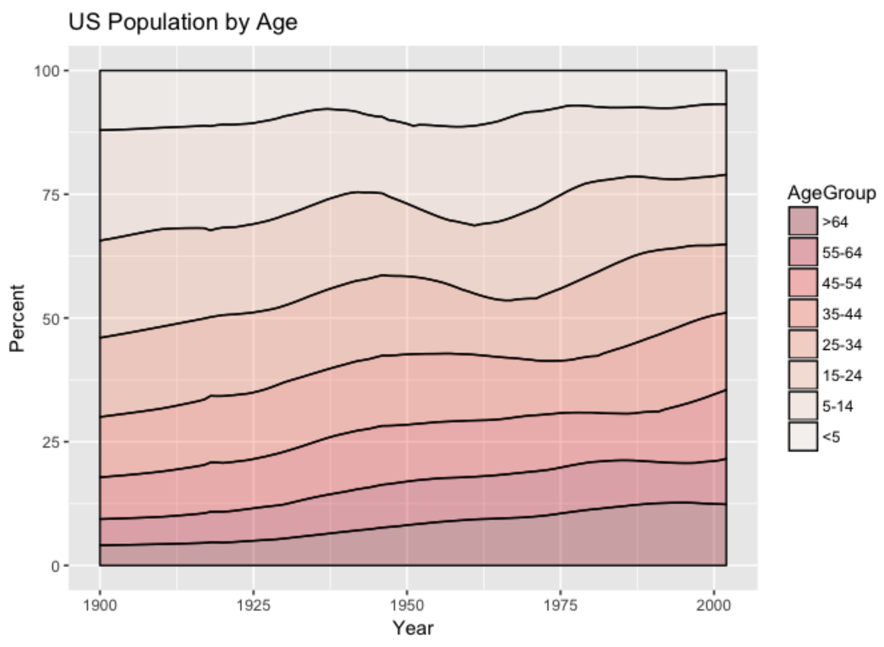
Step 2: Convert the data into usable numbers

The numbers given in this data set were in thousands, but it is much easier to analyze the data when you can see it in the percentage of people as opposed to just bulk numbers. We can do this with the following command:

```
# convert the population number in thousands to percent of population
populationPercent <- ddply(uspopage, "Year", transform,
 Percent = Thousands / sum(Thousands) * 100)
```

Step 3: Plot the proportional stacked area graph

```
# plot the US Population by Age
ggplot(populationPercent, aes(x=Year, y=Percent, fill=AgeGroup)) +
 geom_area(colour="black", size=.5, alpha=.3) +
 scale_fill_brewer(palette="Reds", breaks=rev(levels(uspopage$AgeGroup))) + ggtitle("US Population by Age")
```



This graph can be easily and elegantly viewed because we set the color palette to be "Reds" which provides a nice distinct Red color for each Age Group that allows viewers to see the differences between each group. By adjusting the size of the line in *geom_area* we can make the lines between each group thicker or thinner, which provides a much clearer view of each category.

Summary

The overarching message that should be taken from this post is that a much more thorough analysis can be made at times through the use of packages such as **ggplot2**. When looking at multiple points of data, using just regular barcharts or histograms can make your visual look cluttered, so utilizing **ggplot2** plots such as the lollipop plot or Proportional Stacked Area Graph can make your data look much more clear when laid out.

The examples we used demonstrated visualizations of things such as best car highway MPG. This can be especially useful for consumers looking to buy a new car and are trying to get the best bang for their buck in terms of highway milelage. The lollipop chart would be the perfect way for a consumer to quickly see which company is the best in that specific category. Our second example that analyzed age groups would be especially useful for companies looking to see the population numbers and if there are specific age groups they should target. The trends of the ages fluctuating could impact whether certain products were made in the future or not.

All in all, more detailed diagrams can significantly impact the effectiveness of an analysis, and the **ggplot2** package can do just that.

Reference List

https://www.statmethods.net/graphs/index.html

https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

http://ggplot2.tidyverse.org/reference/

http://sape.inf.usi.ch/quick-reference/ggplot2/geom_segment

http://uc-r.github.io/lollipop

http://uc-r.github.io/ggplot_intro

https://ase.tufts.edu/bugs/guide/assets/R%20Graphics%20Cookbook.pdf

https://github.com/wch/gcookbook/tree/master/data