# Post 1- Statistics 133 (Fall 2017)

*Shivani Prabala*

*10/31/2017*

# More Plotting Functions in R: Dygraph and HighCharter

## Introduction:

This post will explore the intersection of plotting and data frames and how to create different kinds of plots depending on the type of data we are looking at. In class so far we have learned about several graphing functions including plot, ggplot, barplot, hist, etc. The goal of this post is to reiterate the usefulness of those functions we have already learned about and then to explore two new graphing functions in R which include **dygraph** and **highchart**. These functions have many different features which are useful in visually representing other forms of data. My hope is that the information in this post can equip my fellow students with more tools to represent data in different ways.

## Motivation:

The motivation behind this post comes from a desire to learn new plotting functions in R. The importance of representing data visually in other data science classes and technical positions gives this topic its relevance. I want to learn more ways in which I can present my findings not only in this class, but also in any professional role I may have as well as to share this knowledge with my classmates in Statistics 133.

## Background:

We have learned quite a bit about data frames in class so far including how to manipulate, access, and create tables of data. We have even looked at visual representations of various forms of data. As part of the Statistics 133 cirriculum we have already looked at several plotting functions such as:

- **plot()**
- **barplot()**
- **ggplot()**
- **lines()**
- **hist()**
- **density()**

## Examples:

### Familiar plotting functions:

#### (Example 1) Automobile Land Speed Records from automobile races:

Let's begin by creating a sample data frame and then using familiar functions to plot this data:

```
#load all necessary packages into the R session
library(ggplot2)
library(scatterplot3d)
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
```

```
library(dygraphs)
library(magrittr)
library(highcharter)
```

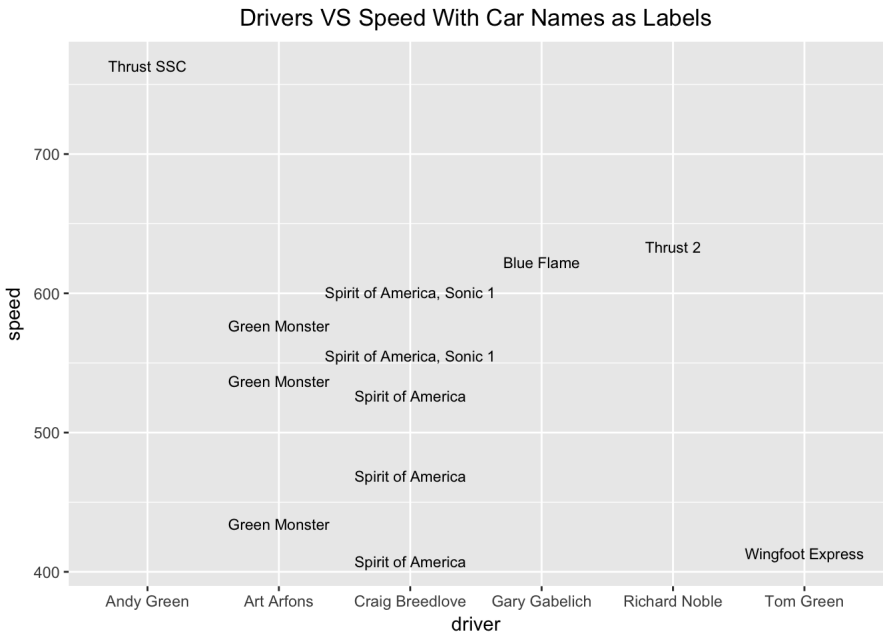
```
## Highcharts (www.highcharts.com) is a Highsoft software product which is
```

```
## not free for commercial and Governmental use
```

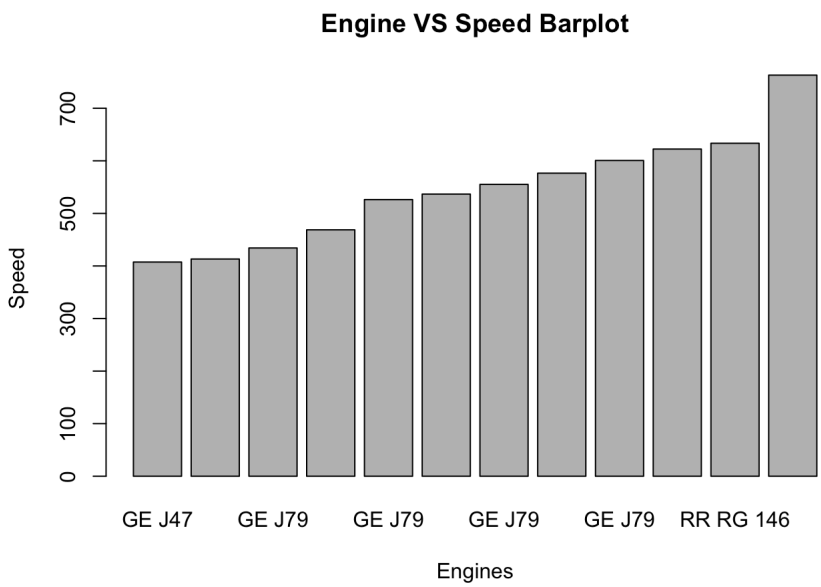
```r
library("forecast")

#creating vectors with each set of data
speed <- c(407.447, 413.199, 434.22, 468.719, 526.277, 536.712, 555.127, 576.553, 600.601, 622.407, 633.468, 763.035)
driver <- c('Craig Breedlove', 'Tom Green', 'Art Arfons', 'Craig Breedlove', 'Craig Breedlove', 'Art Arfons', 'Craig Breedlove','Art Arfons','Craig Breedlove', 'Gary Gabelich', 'Richard Noble', 'Andy Green')
car <- c('Spirit of America', 'Wingfoot Express', 'Green Monster', 'Spirit of America', 'Spirit of America', 'Green Monster', 'Spirit of America, Sonic 1', 'Green Monster', 'Spirit of America, Sonic 1', 'Blue Flame', 'Thrust 2', 'Thrust SSC')
engine <- c('GE J47', 'WE J46', 'GE J79', 'GE J79', 'GE J79', 'GE J79', 'GE J79', 'GE J79', 'GE J79', 'Rocket', 'RR RG 146', 'RR Spey')

date <- c(1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, 1971,1972,1973, 1974)
num_races<- c(190,200,160,190,190,160,190,160,190,110, 150, 130) # number of races driven by a driver
#compiling the vectors into a data frame
auto_dat <- data.frame(speed, driver, car, engine, date, num_races)
#displaying the data frame
auto_dat
```

```
##      speed          driver                         car    engine date
## 1  407.447 Craig Breedlove          Spirit of America    GE J47 1963
## 2  413.199       Tom Green           Wingfoot Express    WE J46 1964
## 3  434.220      Art Arfons              Green Monster    GE J79 1965
## 4  468.719 Craig Breedlove          Spirit of America    GE J79 1966
## 5  526.277 Craig Breedlove          Spirit of America    GE J79 1967
## 6  536.712      Art Arfons              Green Monster    GE J79 1968
## 7  555.127 Craig Breedlove Spirit of America, Sonic 1    GE J79 1969
## 8  576.553      Art Arfons              Green Monster    GE J79 1970
## 9  600.601 Craig Breedlove Spirit of America, Sonic 1    GE J79 1971
## 10 622.407   Gary Gabelich                 Blue Flame    Rocket 1972
## 11 633.468   Richard Noble                   Thrust 2 RR RG 146 1973
## 12 763.035      Andy Green                 Thrust SSC   RR Spey 1974
##    num_races
## 1        190
## 2        200
## 3        160
## 4        190
## 5        190
## 6        160
## 7        190
## 8        160
## 9        190
## 10       110
## 11       150
## 12       130
```

```r
#Driver vs. Speed on a Scatterplot
ggplot(auto_dat, aes(x=driver, y=speed)) + geom_text(aes(label=car), size=3) + ggtitle("Drivers VS Speed With Car Names as Labels") + theme(plot.title = element_text(hjust = 0.5))
```



```
#Engine vs. Speed on a Barplot
#select the data frame and the relevant columns we would like explore the relationship between
eng_dat <- select(auto_dat, speed, engine)
#arranging the data by speed in increasing order
eng_dat <- arrange(eng_dat, speed)
#plotting the engine type vs speed in ascending order of speed on barplot
barplot(eng_dat$speed, main= "Engine VS Speed Barplot", names.arg = eng_dat$engine, xlab= "Engines", ylab="Speed")
```




Here we have created two simple graphs. The first graph is a scatterplot of driver names plotted against speed with the points labeled as the name of the car the driver was using. The second graph is a barplot of engines plotted against speed with the x-axis labeled with engine names. We can use many of the graphing methods we have already learned in class to represent the same data in different ways, depending on which relationships we would like to furthur explore. Next, let's look at a new plotting function that I discovered in doing some furthur research on R. This function is called dygraph and it is a plotting function which allows one to handle large sets of data easily, as well as provides the means to chart time-series data in R.Let's look at an example of how it works:

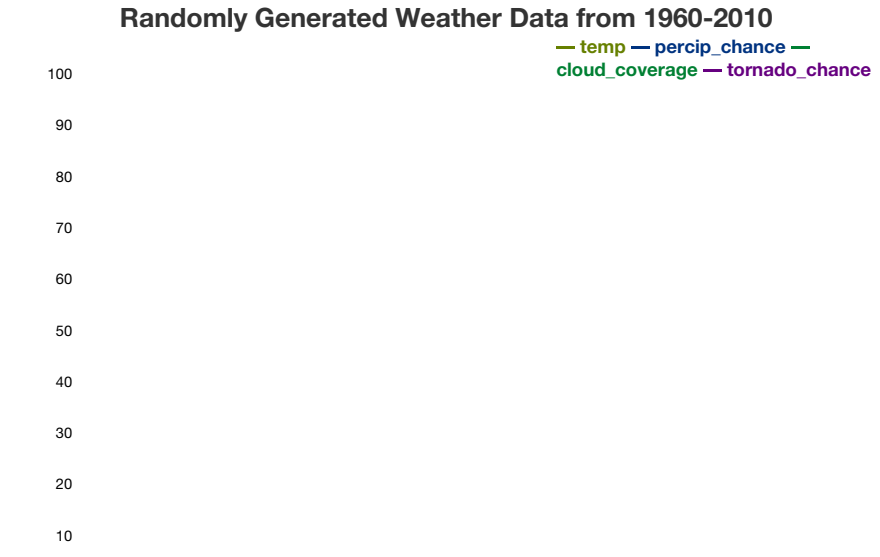
# Dygraph:

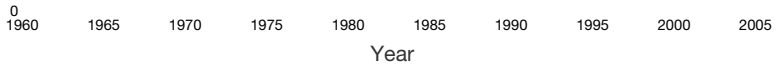
## (Example 2) Randomly Generated Weather Data from 1960-2010:

### Why dygraphs are useful:

Dygraph not only allows us to graph multiple numeric sets against one another (which previous functions have also allowed), but it also includes a new feature which is interaction. As you have probably noticed, if you hover your arrow over the graph and move it around, a vertical set of dots will move along the graph allowing you to see all corresponding y values for a particular x value. This adds depth to analysis because it allows one to precisely see both values and furthurmore to track (continuously) how they change over the x values. From this randomly generated weather data we see that tracking for example the temperature and chance of tornadoes over time becomes a lot easier because the tracker allows one to easily see how the values move.

```
#creating the variables of the data frame weather_dat: date, temperature, chance of percipitation, cloud coverage,
and chance of tornadoes
date <- c(1960:2010)
temp <- sample(30:98, 51, replace=TRUE)
percip_chance<- sample(30:60, 51, replace=TRUE)
cloud_coverage <- sample(1:100, 51, replace=TRUE)
tornado_chance <- sample(10:20, 51, replace=TRUE)
#creating the data frame
weather_dat <- data.frame(date, temp, percip_chance,cloud_coverage, tornado_chance)
#graphing the data using dygraph
dygraph(weather_dat,main= "Randomly Generated Weather Data from 1960-2010", xlab = "Year" )
```

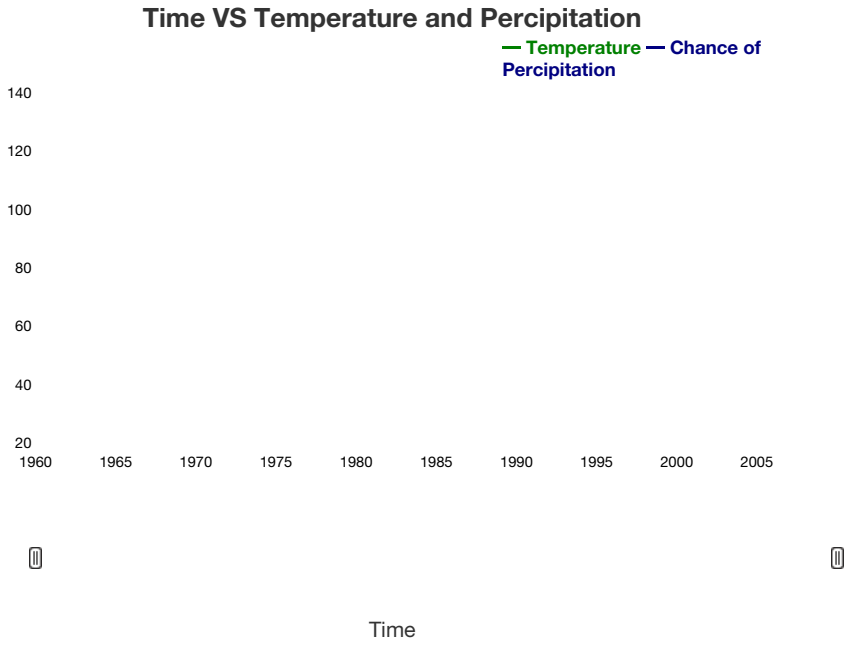





## Stacked Graphs:

Stacked graphs are a feature of dygraph that allows one to directly compare two variables. As you can see below, one is able to view how temperatures and chance of percipitation change over time. This representation can also be useful in determining what percentage of one variable another variable makes up. For example, if the chance of percipitation was a combination of the amount of cloud coverage and level of humidity then one could graph the two together and see which factor contributes more to the chance of percipitation.

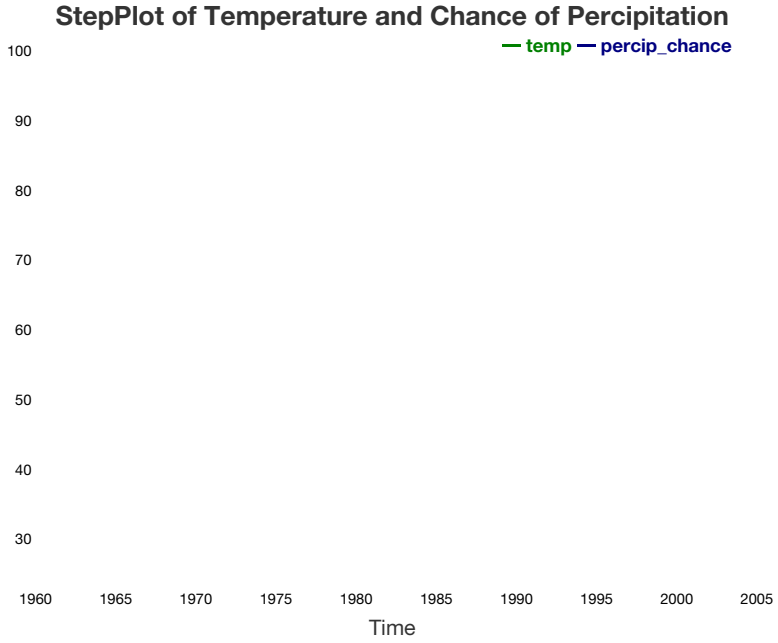
```
#creating a data table that includes date, temperature, and chance of percipitation columns from weather_dat
stacked_dat <- select(weather_dat, date, temp,percip_chance)

#using dygraph to graph stacked_dat in a stacked graph format
dygraph(stacked_dat, main = "Time VS Temperature and Percipitation", xlab="Time") %>%
  dySeries("temp", label="Temperature") %>%
  dySeries("percip_chance", label="Chance of Percipitation") %>%
  dyOptions(stackedGraph= TRUE) %>%
  dyRangeSelector(height=100)
```




## StepPlots:

StepPlots are another feature of dygraph and are especially useful for discrete data that has sudden jumps in it. If one would like to observe such jumps or even to view otherwise continuous data in a discrete format, stepplot can be employed. In the following graph we will view Temperature and Chance of Percipitation against one another in a stepplot format:

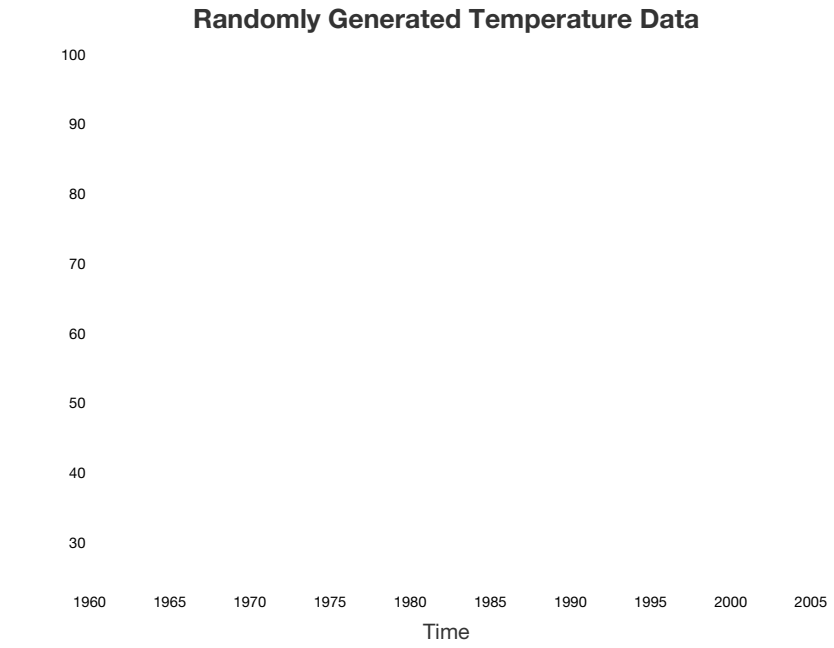
```
#referencing stacked_dat under a new variable name
step_dat <- stacked_dat
#graphing temperature and chance of percipitation against one another using stepplot function
dygraph(step_dat, main="StepPlot of Temperature and Chance of Percipitation", xlab = "Time") %>%
  dyOptions(stepPlot= TRUE)
```




## Fill Graphs:

Another feature of dygraph includes fill graphs which can be used to view continuous or discrete variables and are especially useful if one would like to observe area under the curve. In the following graph, we can view temperature as a continuous variables and see how it varies over time through the fill graph option in dygraph:

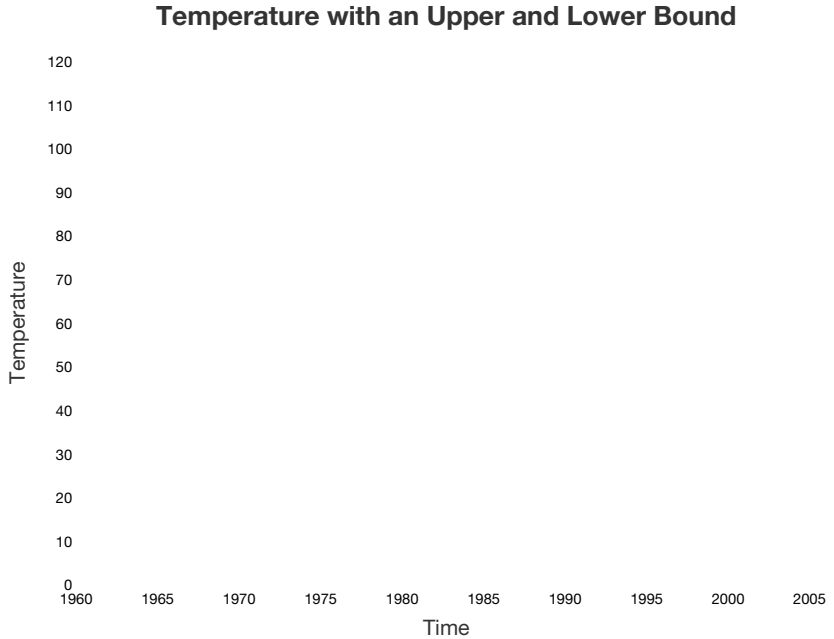
```
#selected the date and temperature columns of weather_dat to create a new data frame
single_var <- select(weather_dat, date, temp)
#plotting the new data frame using dygraph and the fill graph option
dygraph(single_var, main="Randomly Generated Temperature Data", xlab = "Time") %>%
  dyOptions(fillGraph = TRUE, fillAlpha= 0.4)
```




## Upper/Lower Fill Graphs:

Upper/Lower fill graphs allow us to look at how certain data will vary over time with added degrees of freedom. Suppose that we have temperature data which was included in **weather_dat** and a meterologist determines that in this time period the actual temperatures may vary from the data by 20 degrees. These upper/lower graphs allow for us to account for this variance visually. If we were to graph the aforementioned scenario, it would look something like this:

```
#create a new data frame with the temperature variable and two new variables containing upper and lower bounds of
temperature
new <- data.frame(date=c(1960:2010), temp=weather_dat$temp, temp.upper=weather_dat$temp+20, temp.lower=weather_dat
$temp-20)
hw <- HoltWinters(new, gamma = FALSE) #creating the variance above and below the temp data
p <- predict(hw, n.ahead=3, prediction.interval=FALSE)

#graphing the temp data using dygraph with the upper/lower graph option
dygraph(new, "Temperature with an Upper and Lower Bound", xlab = "Time", ylab="Temperature") %>%
   dySeries("temp") %>%
  dySeries(c("temp.lower", "temp", "temp.upper"), label = "Temperature")
```




Another plotting function from R that we haven't seen is highcharter. This function provides some unique features that allow us to get really creative with our graphs:

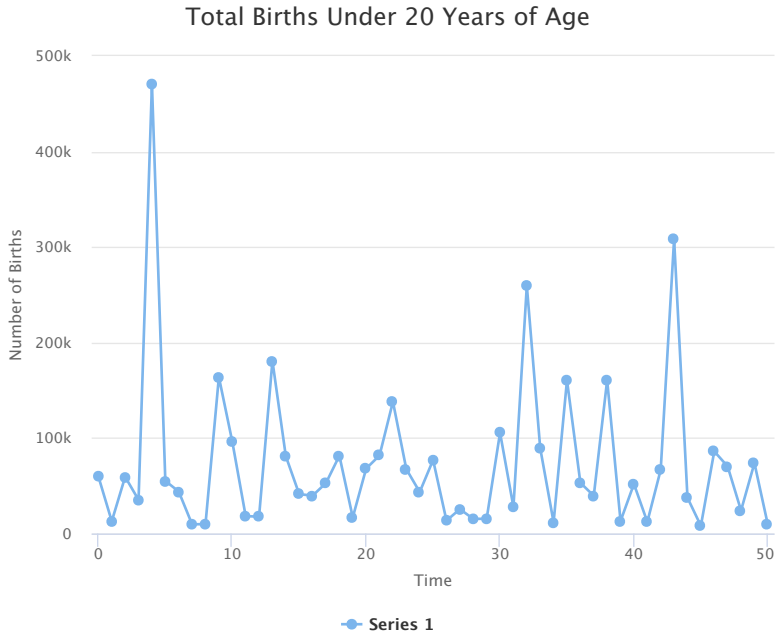
# HighCharter:

## (Example 3) Randomly Generated Weather Data from 1960-2010:

## Why High Charts are useful:

From this randomly generated weather data we see that tracking for example the temperature and chance of tornadoes over time becomes a lot easier because the tracker allows one to easily see how the values move similar to dygraph. Highchart is another useful graphing function that allows one to display time-series data in an easy-to-understand, visually pleasing way. Let's look at a simple example of a scatterplot using highchart:

```
#creating the variables of the new data set: teen_stats
per_hs_grads1990 <- c(65.9, 70.6, 68.5, 76.4, 63.1, 73, 78.5, 68.5, 56.6, 56.2, 60.8, 74.3, 76.7, 72.9, 72.2, 82.7
, 78.4, 68.5, 58.7, 76.1, 70.5, 76.6, 62, 89.4, 63.8, 70.3, 86.4, 84.2, 76.5, 71.7, 77.2, 57.3, 60.4, 66.7, 86.9,
76.4, 77.2, 71.1, 71.7, 64.9, 58.2, 79.9, 68.7, 65.4, 79.5, 80.4, 69.9, 74.7, 78, 82.7, 85.1)

standard_score <- per_hs_grads1990/100

total_births_under20 <- c(59736, 12844, 59348, 35221, 470951, 55123, 43988, 9618, 9870, 163816, 96340, 18307, 1756
7, 180737, 80946, 41221, 39679, 52885, 81458, 16903, 68015, 81780, 138021, 67412, 43449, 76978, 13494, 25551, 1532
5, 15453, 105566, 27757, 259465, 89397, 11721, 160474, 53137, 39486, 160528, 13033, 51900, 12130, 66757, 308164, 3
7451, 8036, 86052, 70230, 24132, 73743, 9366)

#compiling the variables into a data frame
teen_stats <- data.frame(per_hs_grads1990, standard_score, total_births_under20)

#using the highchart function of R to create a simple scatterplot of the total number of births under twenty years
of age
hc <- highchart() %>%
  hc_title(text = "Total Births Under 20 Years of Age") %>%
  hc_add_series(data = teen_stats$total_births_under20) %>%
  hc_yAxis(title= list(text="Number of Births")) %>%
  hc_xAxis(title=list(text="Time"))
hc
```

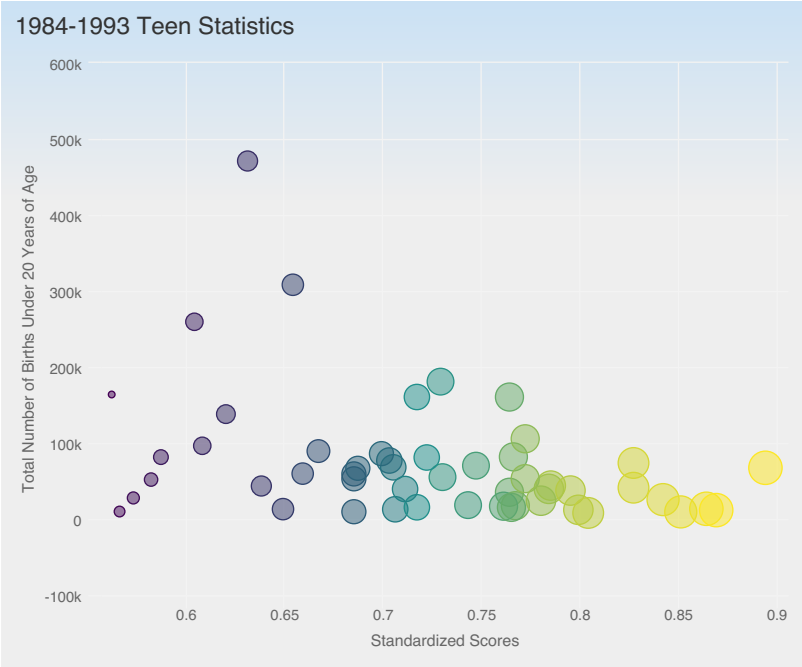



## Bubble Charts:

Bubble charts allow one to view multiple pieces of data and creates size and color bubbles which correspond to those values. In the following chart, we are graphing three pieces of data from **teen_stats** which include standard scores of teens, total births under 20 years of age, and number of graduates. Each bubble has and x and y coordinate of the first two variables, and then a size based on the third variable. In this graph we see that there seems to be no distinguishable association between standard scores of teens and number of births under the age of 20 years, but we do observe the bubble getting smaller as we move downwards along the xaxis. What does this mean? The size of the bubble represents the third variable that we are graphing which is **percentage of highschool grads**. So this pattern tells us that as standardized scores increase the number of highschool graduates also increases which logically seems correct. In this way the bubble chart allows us to analyze mutiple associations at once in a visually aethetic way.

```
#using highchart to create a bubble chart of standard scores vs. total births under 20 years of age with percentag
e of high school grads representing bubble size
highchart() %>%
  hc_title(text="1984-1993 Teen Statistics") %>%
  hc_yAxis(title= list(text="Total Number of Births Under 20 Years of Age")) %>%
  hc_xAxis(title=list(text="Standardized Scores"))%>%
  hc_add_series_scatter(teen_stats$standard_score, teen_stats$total_births_under20, teen_stats$per_hs_grads1990, c
olor=teen_stats$per_hs_grads1990)%>%
  hc_add_theme(hc_theme_ffx()) #theme create backgroun gradient
```

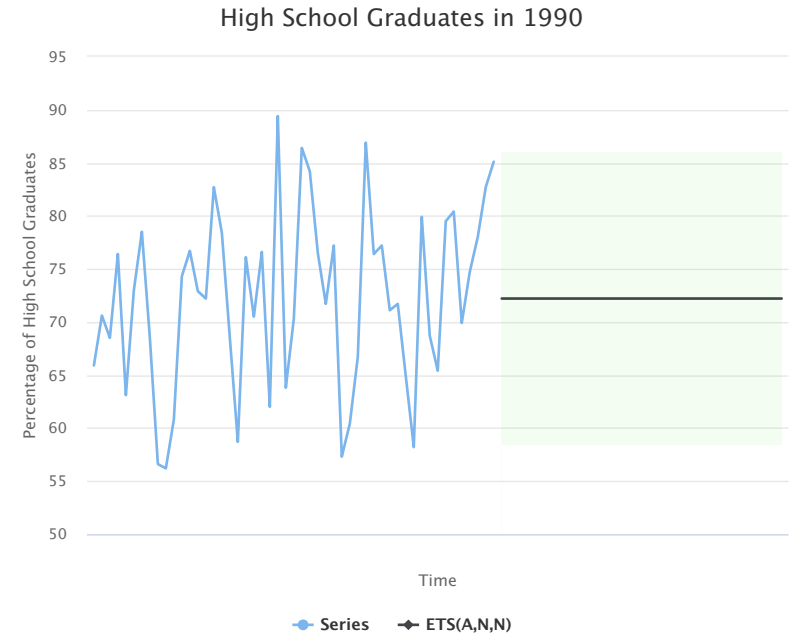
```
## Warning: 'hc_add_series_scatter' is deprecated.
## Use 'hc_add_series' instead.
## See help("Deprecated")
```




## Forecast Charts:

Forecast Charts are especially useful when analyzing financial data. This feature can be used to project how a stock's value will move. In this scenario our data includes teens standardized scores and I have created the chart so as to display an averaged value of predicted values for the next three years. Depending on what kind of forecast you would like to analyze, you can use forecast charts to estimate how fluctuate in the future.

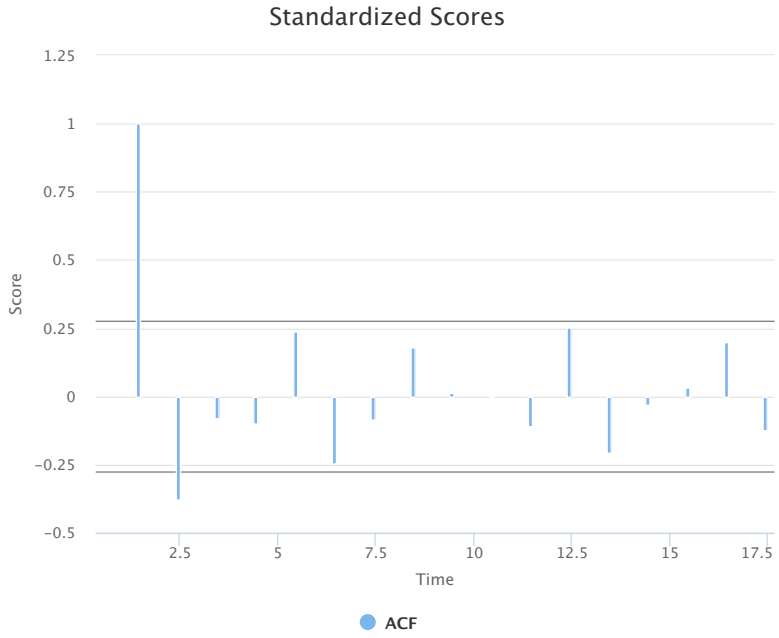
```
#using the forecast function to predict the averaged value of future score predictions
x <- forecast(teen_stats$per_hs_grads1990, h = 36, level = 90)

#using highchart to graph the average prediction on a scatterplot with the real data
hchart(x)%>%
  hc_title(text="High School Graduates in 1990") %>%
  hc_yAxis(title= list(text="Percentage of High School Graduates")) %>%
  hc_xAxis(title=list(text="Time"))
```




## Errorbar Charts:

Errorbar charts are likely most useful in analyzing scientific data. This feature of highchart will not only allow one to compare magnitudes across a variable in a visually simple way, but it will also provide an estimated error level represented by the black horizontal line on the graph. If one would like to present complex data this feature can be useful not only in understanding the data but also in ensuring that necessary degrees of freedom are accounted for.

```
#creating a vector with the appropriate error levels
x <- acf(diff(teen_stats$standard_score), plot = FALSE)

#using highchart to graph the data in an errorbar format
hchart(x)%>%
  hc_title(text="Standardized Scores") %>%
  hc_yAxis(title= list(text="Score")) %>%
  hc_xAxis(title=list(text="Time"))
```




## Conclusion:

We see from these functions that graphing data is a matter of understanding what the analyst would like to associate and describe. In class, we have covered several plotting functions such as plot, ggplot, barplot, hist, etc. These have given us a ultilitarian basis for how to plot data and understand its rudimentary make-up. The functions dygraph and highcharter discussed in this post allow us to analyze data on a deeper level. The features of dygraph allow us to track multiple y values as x values change, as well as to represent data with added variance. The features of highchart allow us to forecast and to view multiple associations at once. We can use these new features to improve our analyses in this class and beyond. The take away message from this post is that there is more than one way to represent data, and the most successful data analysis is highly dependent on understanding what you or your client is interested in observing about the data and then using the appropriate graphing devices to create useful summaries and graphs.

I hope this post was an interesting read for you and thank you for grading it!

## References:


- http://www.sedl.org/afterschool/toolkits/science/pdf/ast_sci_data_tables_sample.pdf
- https://stackoverflow.com/questions/40675778/center-plot-title-in-ggplot2
- http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf
- https://blog.rstudio.com/2015/04/14/interactive-time-series-with-dygraphs/
- https://rstudio.github.io/dygraphs/
- https://rstudio.github.io/dygraphs/gallery-series-options.html
- http://www.htmlwidgets.org/showcase_highcharts.html
- http://mathforum.org/workshops/sum96/data.collections/datalibrary/data.set6.html
- http://jkunst.com/highcharter/highcharts.html
- https://github.com/jbkunst/highcharter/issues/254
- https://www.highcharts.com/