Get the Materials!

- GitHub Repo:
  - https://github.com/datasciencedojo/meetup/tree/master/r_programming_excel_users

- Kaggle Titanic Competition:
  - https://www.kaggle.com/c/titanic

Intro to R Programming for Excel Users

May 3rd, 2017

datasciencedojo
data science for everyone

Who Am I?

- Dave Langer, VP of Data Science – Data Science Dojo

- 20+ years in technology:
  - Roles in development, architecture, & BI/DW/analytics.
  - Last job – Sr. Director, BI & Analytics @ Microsoft.

- Hooked on Data Science 5 years ago:
  - Extensive background in data and analytics.
  - Learned Machine Learning from 2nd place Netflix Prize winner.
  - More tutorials on my YouTube channel!

- Joined Data Science Dojo to democratize Data Science.

Motivation

The IEEE's 2016 Ranking of most Popular Programming Languages

Language Rank	Types	Spectrum Ranking
1. C		100.0
2. Java		98.1
3. Python		98.0
4. C++		95.9
5. R		87.9
6. C#		86.7
7. PHP		82.8
8. JavaScript		82.2
9. Ruby		74.5
10. Go		71.9

http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages

R has experienced rapid YoY increases in popularity.

This is remarkable as R is a specialized language for data and analytics!

Expectation Setting

- I am assuming the following:
  - You are experienced with Excel – tables, formulas, functions.
  - You are not familiar with R.
  - You are interested in learning R.

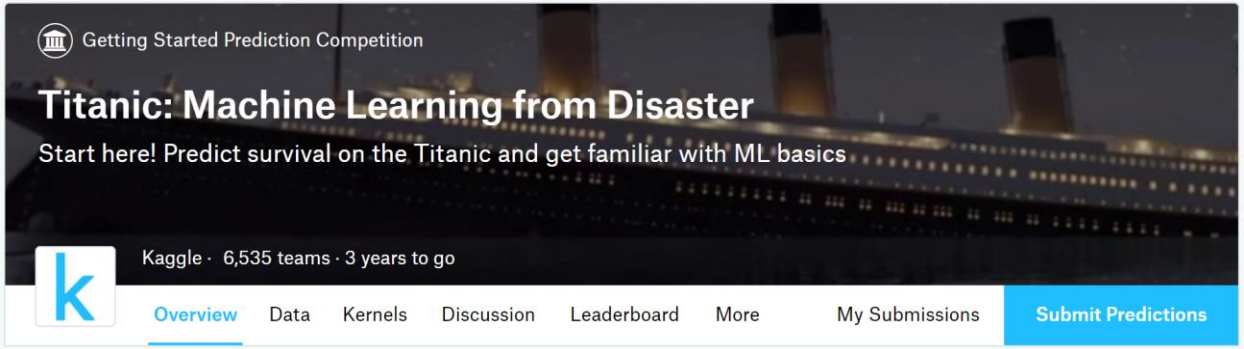
- This is a quick intro to R using Excel as a framework:
  - I will gloss over a lot of things.
  - I will illustrate some “art of the possible”.
  - More in-depth coverage is available on my YouTube channel.

- My goal is to make you confident and excited about learning R!

Prerequisites

- To follow along you will need the following:
  - Excel
  - R
  - RStudio
- The following R packages are required to follow along:
  - Ggplot2 and dplyr
- The GitHub repo has source, data, and slide files.

The Data



Why use this dataset?

1. Everyone is familiar with the problem domain.

2. It is a good proxy for common business data – for example, customer profile data.

The Data

Data Dictionary

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

The Scenario

- We've been asked to analyze the Titanic data in terms of finding patterns of passengers that survived vs. those that did not.
- This analysis has common analogies in business – for example, customer churn analysis.
- We will use common analytical activities in Excel and then illustrate the same in R.

LET’S LEARN SOME R!

QUESTIONS

APPENDIX

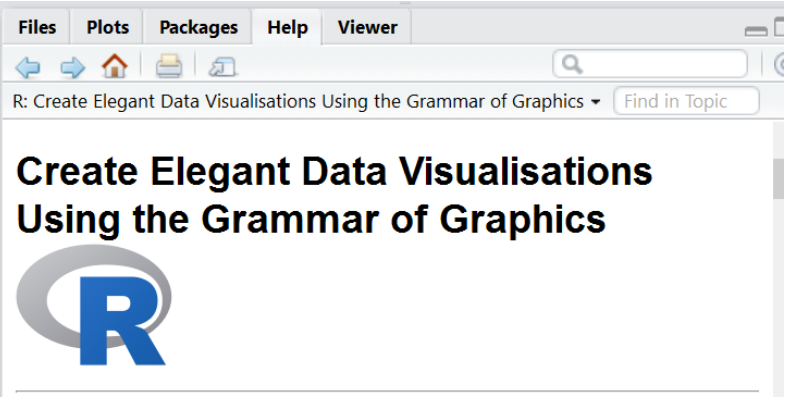
Get the Materials!

- GitHub Repo:
  - https://github.com/datasciencedojo/meetup/tree/master/r_programming_excel_users

- Kaggle Titanic Competition:
  - https://www.kaggle.com/c/titanic

ggplot2

- De facto standard visualization package in R.
- Designed for print-quality graphics.



- Fine-grained control via an API focusing on layering graphical elements to build visualizations.

ggplot2

The collection of data that we're working with.

The aesthetic – how data is mapped to visuals.

Main function – the starting point.

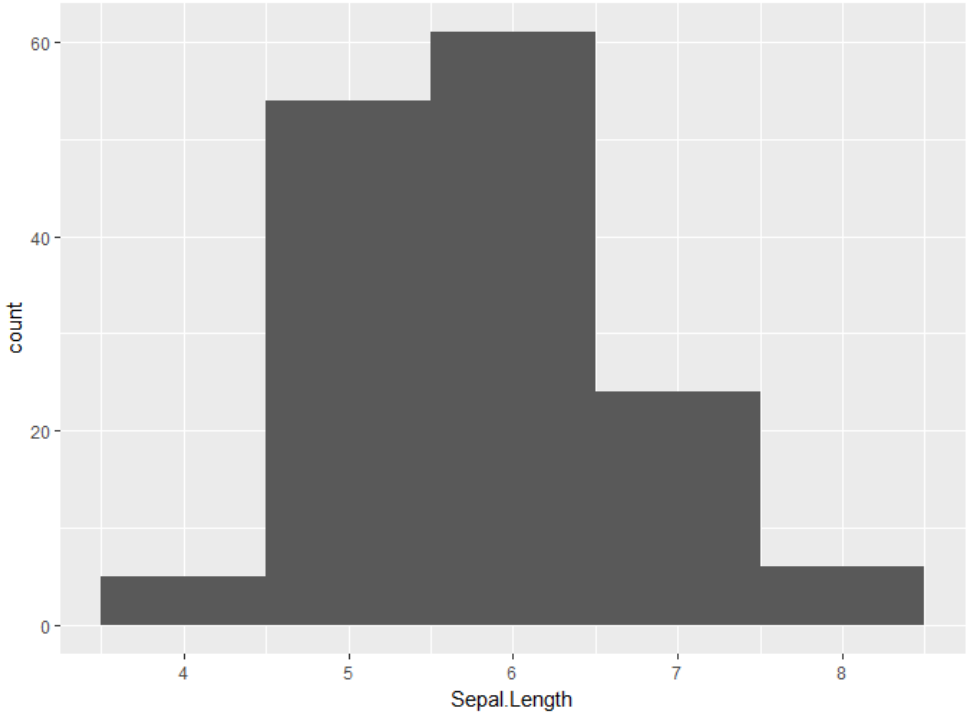
The aesthetic – how data is mapped to visuals.

```
library(ggplot2)

data("iris")

ggplot(iris, aes(x = Sepal.Length)) +
  geom_histogram(binwidth = 1)
```

A visual to layer on to the canvas.

Visual parameters – control the look.

ggplot2

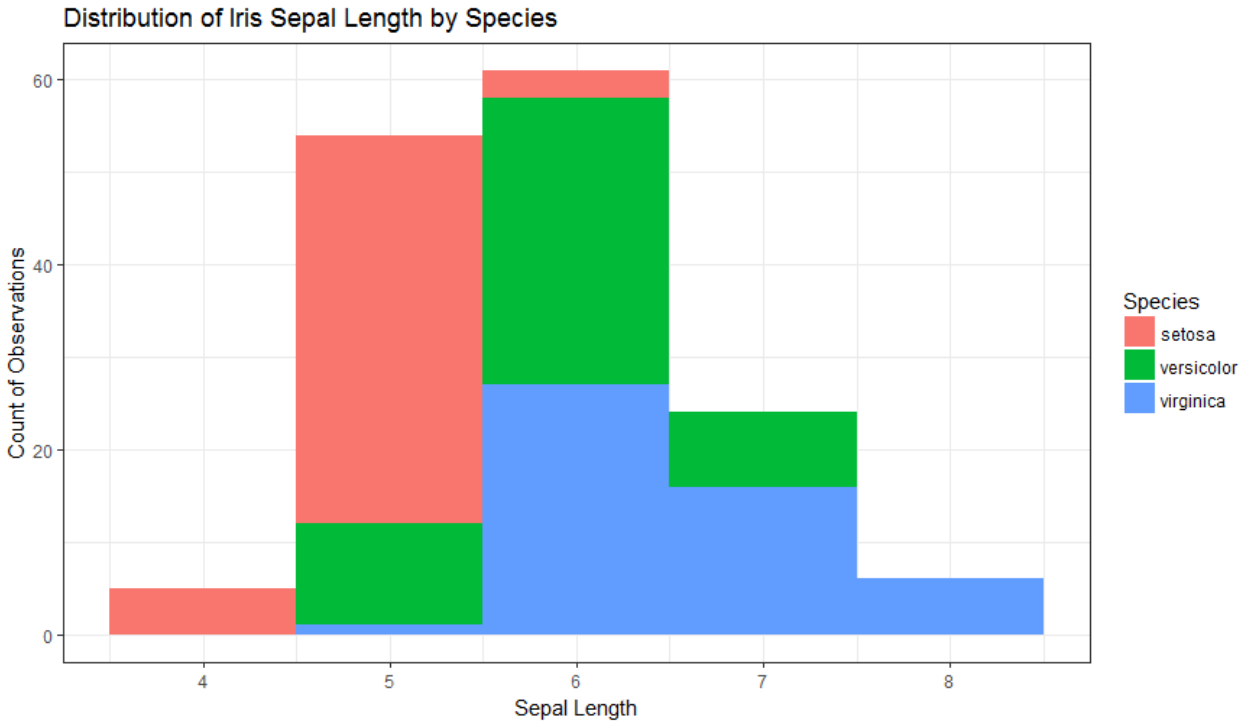


ggplot2

Incrementally build more powerful visualizations

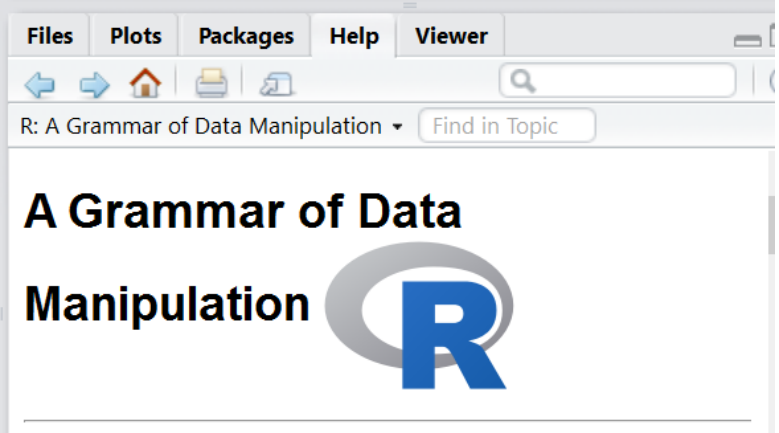
```r
library(ggplot2)

data("iris")

ggplot(iris, aes(x = Sepal.Length, fill = Species)) +
  theme_bw() +
  geom_histogram(binwidth = 1) +
  labs(x = "Sepal Length",
       y = "Count of Observations",
       title = "Distribution of Iris Sepal Length by Species")
```

ggplot2



dplyr

- Popular package for data wrangling.
- Intuitive coding style based on data pieplines.



- Very intuitive to those with familiarity with SQL – just about everything you can do with SQL you can do with dplyr.

dplyr

The data we're working with.

Pipe data to the next operation.

Filter data down.

Group into categories.

Summarize groups.

```
library(dplyr)

iris.stats <- iris %>%
  filter(Species == "setosa" |
         Species == "virginica") %>%
  group_by(Species) %>%
  summarize(Sepal.Length.Min = min(Sepal.Length),
            Sepal.Length.Max = max(Sepal.Length),
            Sepal.Length.Mean = mean(Sepal.Length),
            Sepal.Length.Median = median(Sepal.Length),
            Sepal.Length.SD = sd(Sepal.Length))
View(iris.stats)
```

dplyr

Filter

	Species	Sepal.Length.Min	Sepal.Length.Max	Sepal.Length.Mean	Sepal.Length.Median	Sepal.Length.SD
1	setosa	4.3	5.8	5.006	5.0	0.3524897
2	virginica	4.9	7.9	6.588	6.5	0.6358796