Short Course: Day 1

Data analysis, modeling and reporting using R, RStudio and RMarkdown

International symposium on current trends in modeling and software development in data science and Statistics
Cape Town, South Africa
20.02.2024

Rudradev Sengupta
Ziv Shkedy

Overview

Steps in Data Analysis

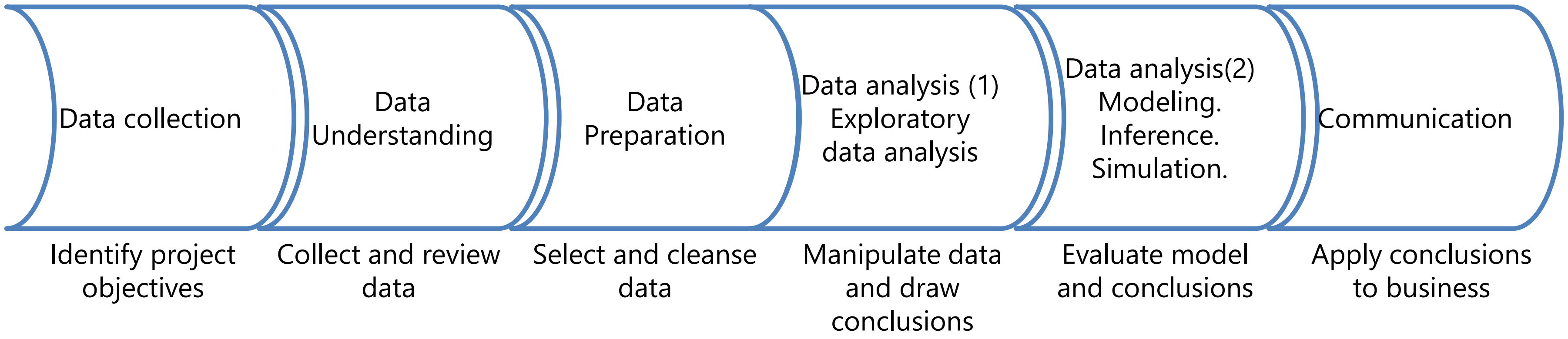
- Data analysis approach in the course:

A Problem → A data analysis Problem → Data driven decision

Scientific problem of interest.

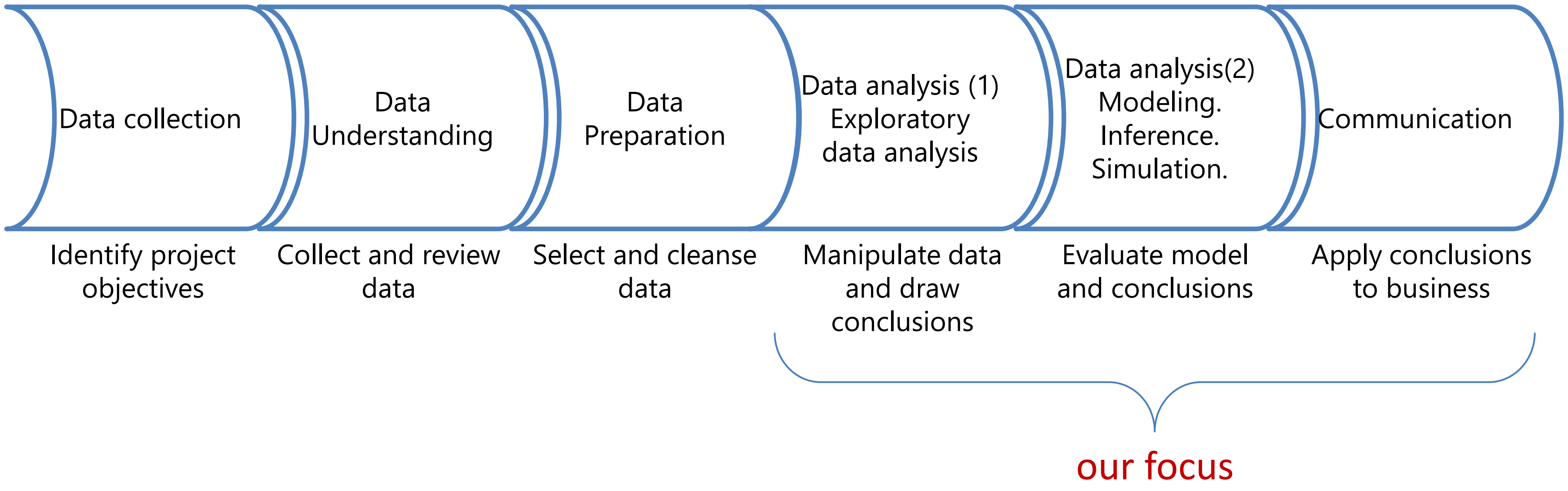
Steps in Data Analysis

- Steps related to data analysis:



Steps in Data Analysis

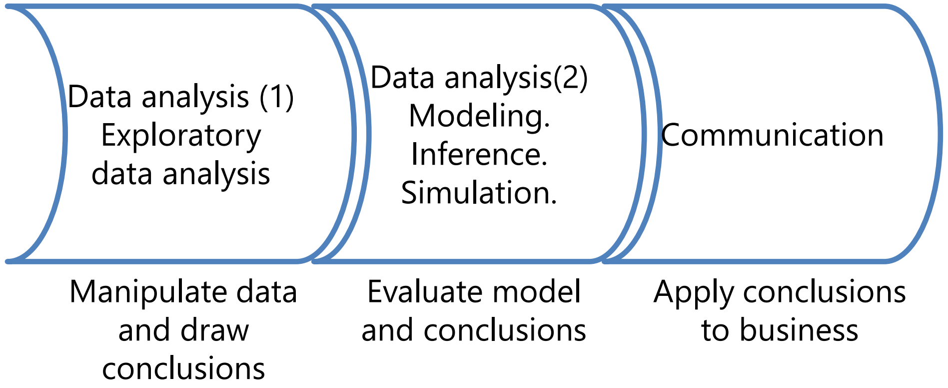
- Steps related to data analysis:



Steps in Data Analysis

motivation behind clinical trials:

A pharma company would like to know if a new drug A (produced by the company) is better than drug B (the standard)



Steps in Data Analysis

A pharma company would like to know if a new drug A (produced by the company) is better than drug B (the standard)

Scientific problem of interest:
how to compare between the two drugs ?

Data analysis (1) Exploratory data analysis	Data analysis(2) Modeling. Inference. Simulation.	Communication
Manipulate data and draw conclusions	Evaluate model and conclusions	Apply conclusions to business

Steps in Data Analysis

A pharma company would like to know if a new drug A (produce by the company) is better than drug B (the standard)

$$H_0: \mu_A = \mu_B$$
$$H_1: \mu_A \neq \mu_B$$

Methodology: two-samples t-test.

Data analysis (1) Exploratory data analysis.

Manipulate data and draw conclusions

Data analysis(2) Modeling. Inference. Simulation.

Evaluate model and conclusions

Communication

Apply conclusions to business

Boxplot by treatment group

A two-sample t-test

A report

The solution

Johnson&Johnson

Steps in Data Analysis

A pharma company would like to know if a new drug A (produce by the company) is better than drug B (the standard)

$$H_0: \mu_A = \mu_B$$
$$H_1: \mu_A \neq \mu_B$$

Methodology: two-sample t-test.

We "translate" the methodology to software usage

Data analysis (1) Exploratory data analysisType equation here.	Data analysis(2) Modeling. Inference. Simulation.	Communication
Manipulate data and draw conclusions	Evaluate model and conclusions	Apply conclusions to business
Boxplot by treatment group	A two-sample t-test	A report
ggplot2()	t.test ()	R markdown to produce a PDF file.

We develop software to produce the solution and to communicate the solution

Johnson&Johnson

Possible Analysis Approaches

- What kind of variables are available in the data?
  - Numbers:
    - Continuous data: Numbers that can take on any value.
    - Binary data: Data that are 0/1 (e.g., death/survival, below/above a threshold, etc.)
    - Count data: Integers (e.g., number of plants, number of hits, etc.)
    - Proportion data: These can take a value between 0 and 1.
  - Factors:
    - categorical data
- Main focus: which research questions to be answered?
- Inference and graphical displays.

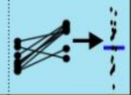
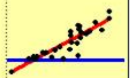
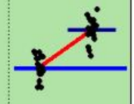
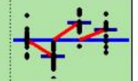
Possible Research Question(s)

- Often is to determine the effect of the predictor variable(s) on a response variable.

Predictor Variable Type	Response Variable Type	Possible Analysis
Continuous	Continuous	linear regression or GLM with Gaussian distribution
Continuous	Binary	GLM with “binomial” family (logistic regression)
Continuous	Counts	GLM with “Poisson” family (Poisson regression)
Continuous	Proportion	GLM with “binomial”/”Quasi-binomial” family
Categorical	Continuous	ANOVA (or t-test to compare means) => can be run as GLM
…	…	…

DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson saMRC advancinglife

Common Statistical Tests are Linear Models

	Common name	Built-in function in R	Equivalent linear model in R	Exact?	The linear model in words	Icon
Simple regression: lm(y ~ 1 + x)	**y is independent of x** P: One-sample t-test N: Wilcoxon signed-rank	t.test(y) wilcox.test(y)	lm(y ~ 1) lm(signed_rank(y) ~ 1)	✓ for N >14	One number (intercept, i.e., the mean) predicts **y**. - (Same, but it predicts the *signed rank* of **y**.)	
	P: Paired-sample t-test N: Wilcoxon matched pairs	t.test(y_1, y_2, paired=TRUE) wilcox.test(y_1, y_2, paired=TRUE)	lm(y_2 - y_1 ~ 1) lm(signed_rank(y_2 - y_1) ~ 1)	✓ for N >14	One intercept predicts the pairwise **y_2-y_1** differences. - (Same, but it predicts the *signed rank* of **y_2-y_1**.)	
	y ~ continuous x P: Pearson correlation N: Spearman correlation	cor.test(x, y, method='Pearson') cor.test(x, y, method='Spearman')	lm(y ~ 1 + x) lm(rank(y) ~ 1 + rank(x))	✓ for N >10	One intercept plus **x** multiplied by a number (slope) predicts **y**. - (Same, but with *ranked* **x** and **y**)	
	y ~ discrete x P: Two-sample t-test P: Welch's t-test N: Mann-Whitney U	t.test(y_1, y_2, var.equal=TRUE) t.test(y_1, y_2, var.equal=FALSE) wilcox.test(y_1, y_2)	lm(y ~ 1 + G_2)^A gls(y ~ 1 + G_2, weights=...^{B})^A lm(signed_rank(y) ~ 1 + G_2)^A	✓ ✓ for N >11	An intercept for **group 1** (plus a difference if **group 2**) predicts **y**. - (Same, but with one variance *per group* instead of one common.) - (Same, but it predicts the *signed rank* of **y**.)	
Multiple regression: lm(y ~ 1 + x_1 + x_2 +...)	P: One-way ANOVA N: Kruskal-Wallis	aov(y ~ group) kruskal.test(y ~ group)	lm(y ~ 1 + G_2 + G_3 +...+ G_N)^A lm(rank(y) ~ 1 + G_2 + G_3 +...+ G_N)^A	✓ for N >11	An intercept for **group 1** (plus a difference if group ≠ 1) predicts **y**. - (Same, but it predicts the *rank* of **y**.)	
	P: One-way ANCOVA	aov(y ~ group + x)	lm(y ~ 1 + G_2 + G_3 +...+ G_N + x)^A	✓	- (Same, but plus a slope on **x**.) *Note: this is discrete AND continuous. ANCOVAs are ANOVAs with a continuous x.*	
	P: Two-way ANOVA	aov(y ~ group * sex)	lm(y ~ 1 + G_2 + G_3 + ... + G_N + S_2 + S_3+ ... + S_K + G_2*S_2+G_3*S_3+...+G_N*S_K)	✓	Interaction term: changing **sex** changes the **y ~ group** parameters. *Note: $G_{2\ to\ N}$ is an indicator (0 or 1) for each non-intercept levels of the **group** variable. Similarly for $S_{2\ to\ K}$ for sex. The first line (with G_i) is main effect of group, the second (with S_i) for sex and the third is the **group × sex** interaction. For two levels (e.g. male/female), line 2 would just be "S_2" and line 3 would be S_2 multiplied with each G_i.*	*[Coming]*
	Counts ~ discrete x N: Chi-square test	chisq.test(groupXsex_table)	**Equivalent log-linear model** glm(y ~ 1 + G_2 + G_3 + ... + G_N + S_2 + S_3+ ... + S_K + G_2*S_2+G_3*S_3+...+G_N*S_K, family=...)^A	✓	Interaction term: (Same as Two-way ANOVA.) *Note: Run glm using the following arguments:* `glm(model, family=poisson())` *As linear-model, the Chi-square test is $\log(y_i) = \log(N) + \log(\alpha_i) + \log(\beta_j) + \log(\alpha_i\beta_j)$ where α_i and β_j are proportions. See more info in the accompanying notebook.*	*Same as Two-way ANOVA*
	N: Goodness of fit	chisq.test(y)	glm(y ~ 1 + G_2 + G_3 +...+ G_N, family=...)^A	✓	(Same as One-way ANOVA and see Chi-Square note.)	*1W-ANOVA*

DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson saMRC advancinglife

Overview

R Software

- R is a free software that can be used to perform statistical analyses
- It is used worldwide, with constant sharing of new information (forums, online new packages, …)
- It is used extensively in the academic environment
- A good guide to learn the very basics of R is provided at http://cran.r-project.org/doc/manuals/R-intro.pdf

R Software

- R is a great tool for data manipulation, calculations, modelling and graphical display:
  - the data handling is easy and effective
  - there are many built in functions that allow data manipulation (from simple operations to complex modelling)
  - graphical facilities for data analysis are very well developed
  - the programming language (called “S") is simple and effective at the same time

- Documentation on how to exploit the many R functionalities can be found on the web (guides, packages user manuals,...)

R Software

- There are many good reasons to use R:
  - Open source (free download at http://cran.r-project.org/)
  - Available for different platforms: MS Windows, Mac OS X, Linux
  - Expandable with more than 3000 libraries (also with free download)
  - Expandable to any kind of new method we might want to implement (possibility of building our own functions)
  - There is a huge sharing of information and experience on R in the web
  - ……

How Is R Programming Used?

- Data Analysis
- Statistical Modelling
- Data Visualization
- Machine Learning
- Bioinformatics
- Academic Research
- Data Reporting and Visualization: R Markdown and Quarto
- Data Mining and Text Analysis
- Quality Control and Manufacturing

Why Use R for Clinical Trial Analytics in Pharma?

- Statistical Tools
- Improved Data Presentation
- Reproducible Research
- Customization and Extensibility
- Integration With Databases
- Predictive Modelling
- Cost Effective Solution
- Community Support and Collaboration
- Regulatory Compliance
- Time Effeciency
- Adaptation to Emerging Technologies

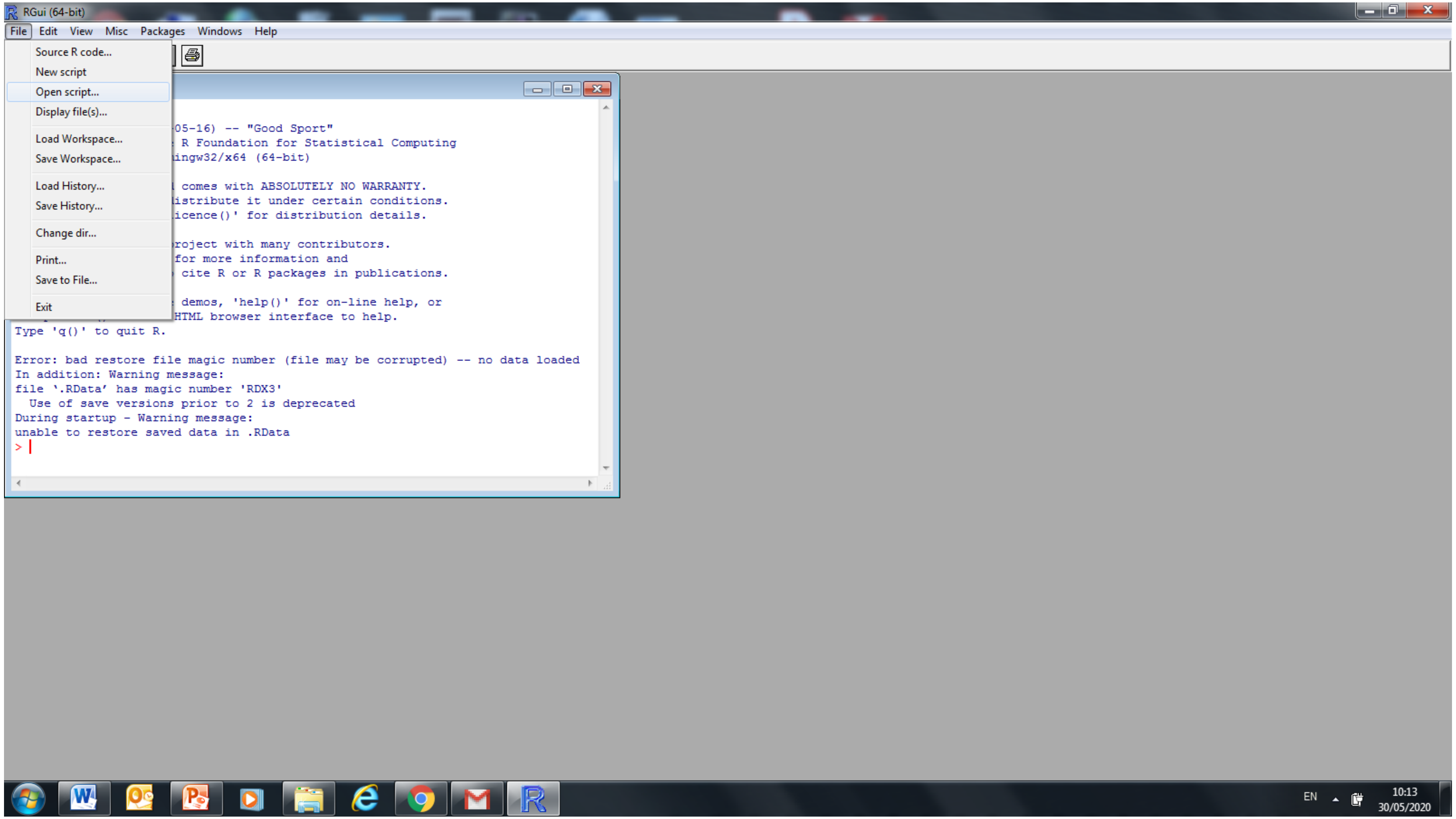
Key Benefits of using R in Pharma

- Advanced Data Analysis
- Quality Control and Assurance
- Decision Support
- Drug Safety and Pharmacovigilance

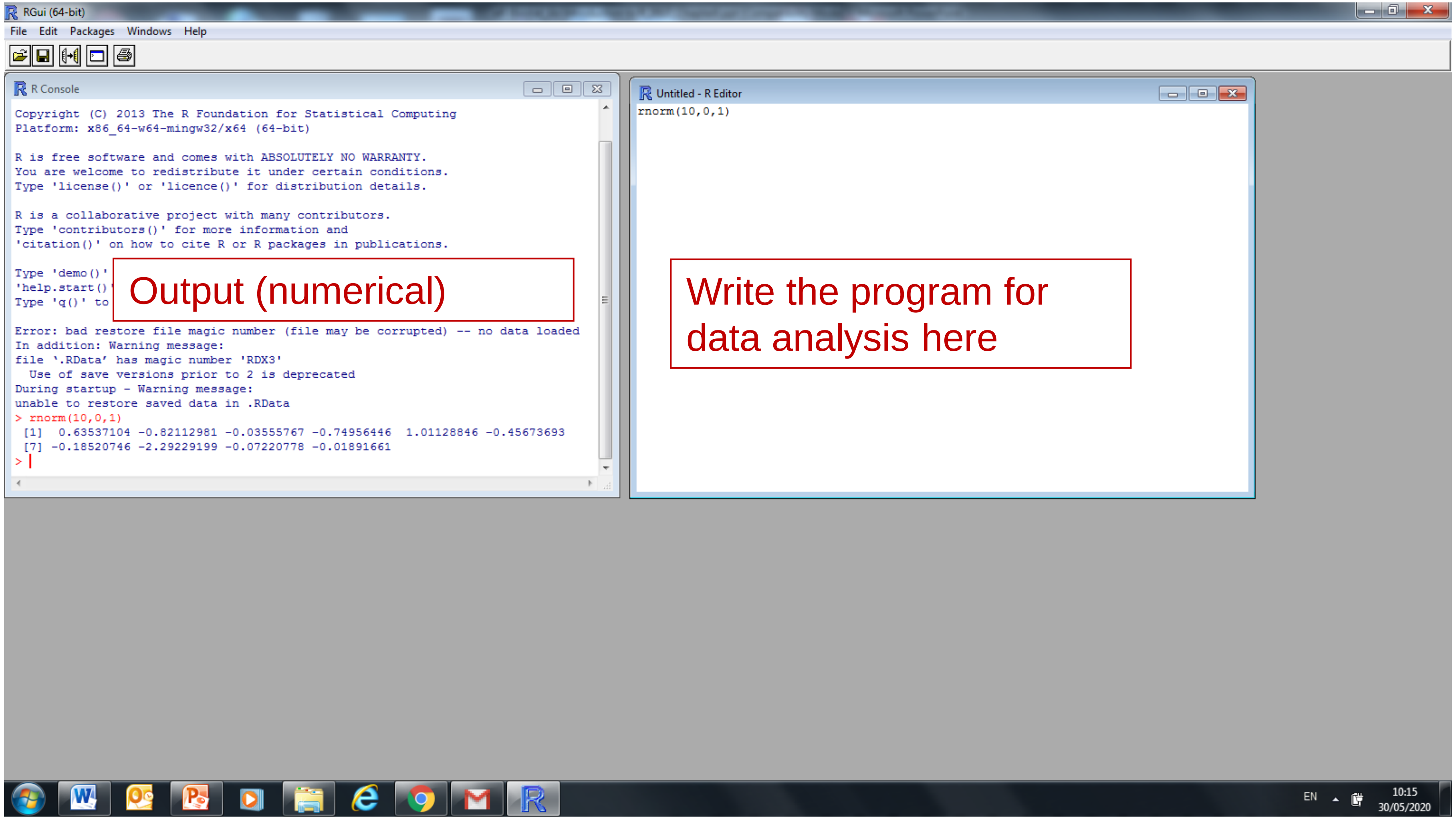
The R environment

- Open R.
- Open a new script window.

Open a script window



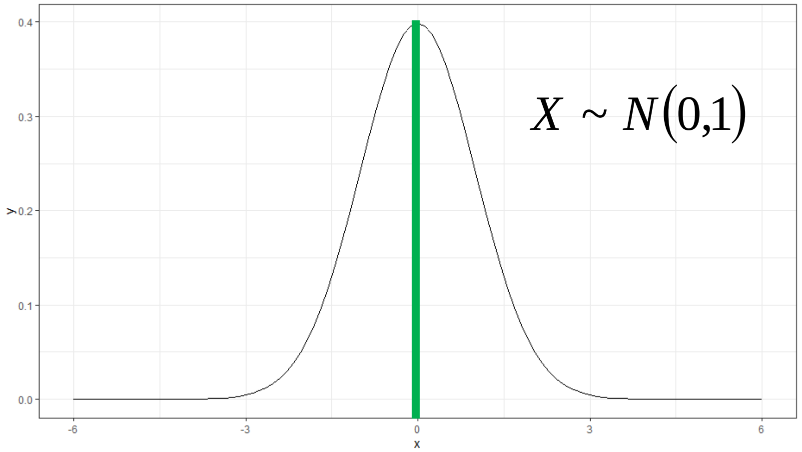
The script & the output

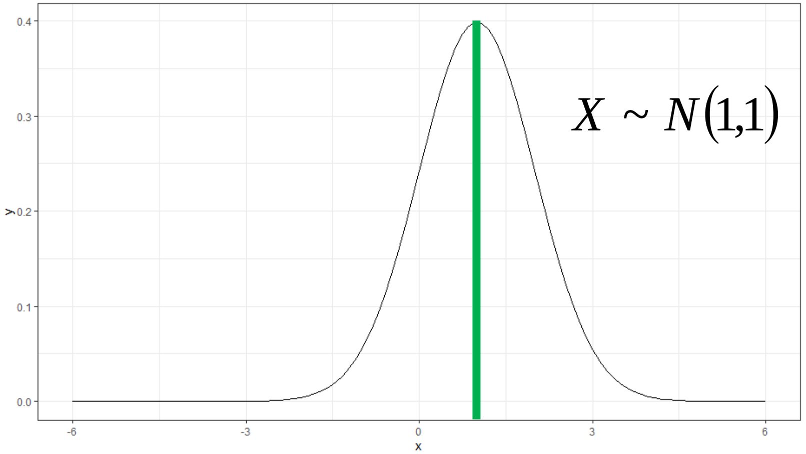


Example: a normal distribution

The Normal Distribution: Location

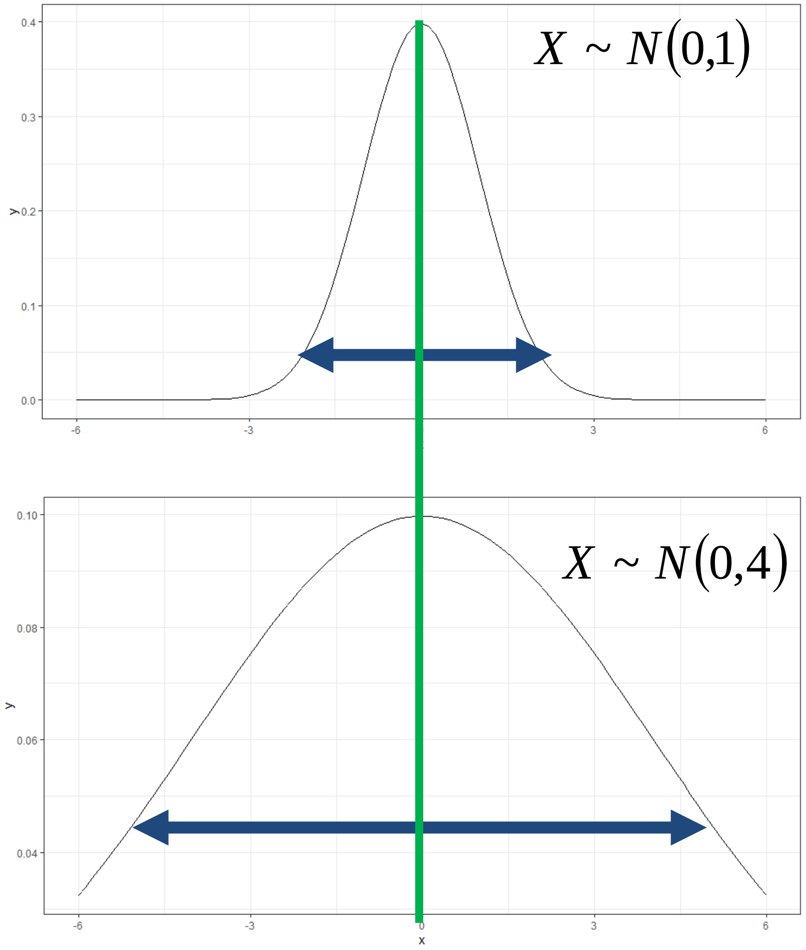
Density function of a normal distribution $X \sim N(\mu, \sigma^2)$





The Normal Distribution: Location

Density function of a normal distribution $X \sim N(\mu, \sigma^2)$



R Functions

function(data)

A procedure that was programed in R that uses data to produce output.

```
> var(x)
```

The r function

data

Calculate the sample variance.

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2$$

Random Sample from a Normal Distribution in R

Draw a random sample of size 100 from a normal distribution with mean –
and variance 1

$$X \sim N\left(\mu, \sigma^2\right)$$

In R

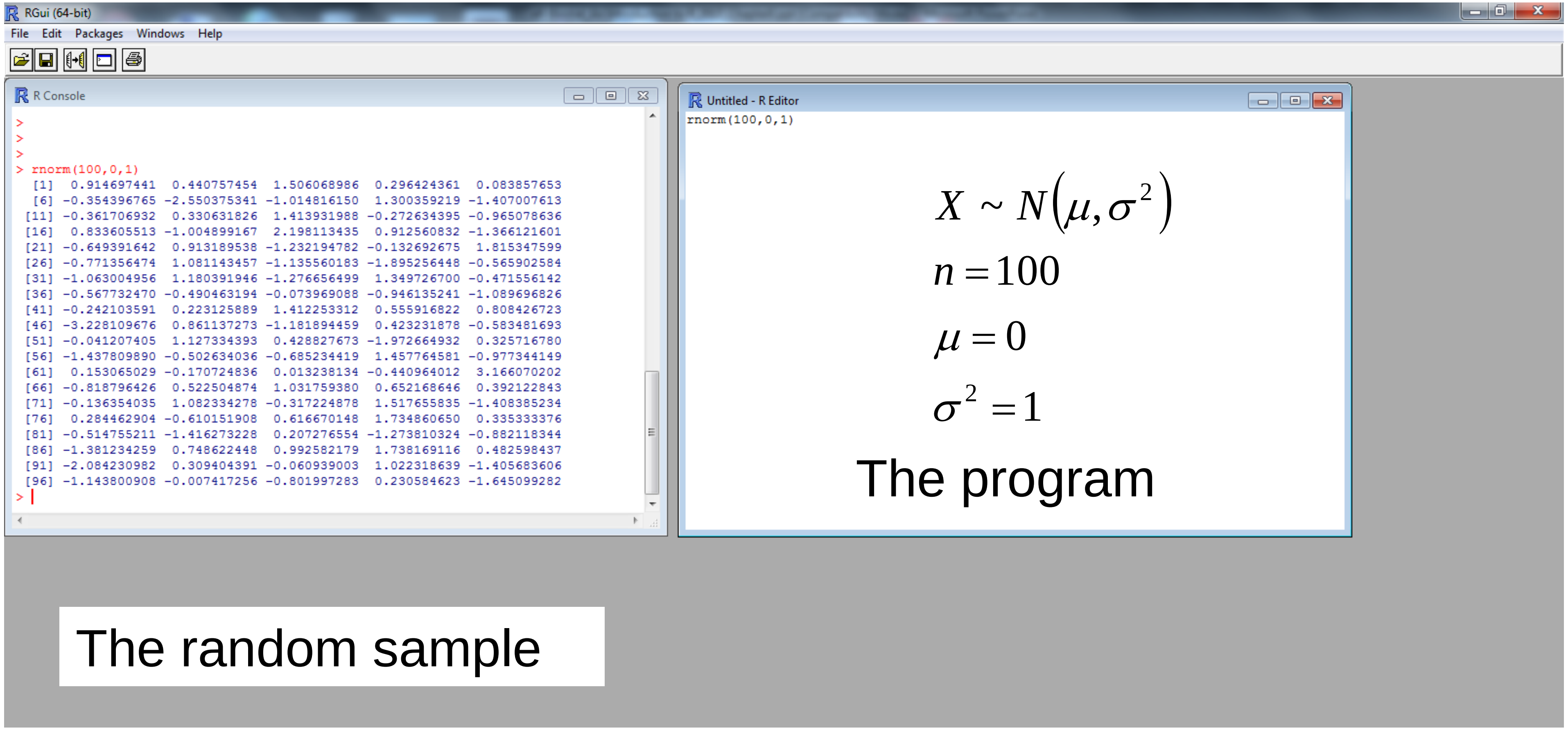
rnorm(sample size, mean,standard deviation)

$$X \sim N\left(0,1\right)$$

rnorm(100, 0, 1)

A function in R that draw a sample from $N\left(\mu, \sigma^2\right)$

The Script & the Output



Random Sample from a Normal Distribution in R

Draw a random sample of size **100** from a normal distribution with mean 0 and variance 1

$$X \sim N\left(\mu, \sigma^2\right) \Rightarrow X \sim N(0,1)$$

> rnorm(100,0,1)

```
  [1] -0.173911348 -0.463196096 -1.084838332  2.373958677 -1.685884982
  [6] -1.952672126 -0.055601310 -0.241913096 -0.999586206  0.308335895
 [11]  0.556993818  2.337451275  0.778734465 -0.501354458  0.004525392
 [16] -1.468709822  0.109901143  0.109103689  0.662434110 -0.177097648
 [21] -1.442033566  0.615239368  0.254080126  1.152977602 -0.089559002
 [26]  0.065022482  0.300405204 -0.190196930 -0.244365328  0.886735849
 [31] -0.667671228 -1.009209277  0.388362272 -0.041883373  0.750480061
 [36] -2.103109677 -1.515839684 -0.477250540 -0.344581482  0.072570862
 [41] -0.364485234 -0.920898769  1.148778190  1.092225688 -0.832389361
 [46] -1.914844153 -0.384265110  0.528078353  1.319149374  0.226817654
 [51] -0.605867376 -0.658048328  0.086126314  0.711404951  1.190303122
 [56]  2.499314086  2.201924724  0.591527333 -0.733622099 -0.656031690
 [61] -0.194759316  0.864421699  0.813854743 -0.628803589  0.362077258
 [66]  0.312250497  1.451227963  1.107136623  0.680487861  1.585879056
 [71] -0.249983835 -1.436293634 -0.470710524 -2.330088808  0.265551343
 [76] -0.847238216 -1.199413581 -1.866542460  0.826973063 -0.592073631
 [81] -1.751735134  0.077115620 -0.306869702  0.120083596 -0.303521155
 [86] -0.644268518  0.295067198  2.004409939  0.310290927  0.221898330
 [91] -1.450606907 -1.264043444 -0.257282348  0.078120141 -0.902925645
 [96]  0.499980835 -0.596173525 -1.085097601 -0.773094391  0.693319162
```

100
observations

Creating an R Object

```
> x <- rnorm(100,0,1)
```

An R object contains the results

```
> x
 [1] -1.91083203  1.04955497 -2.40884482  0.33493954  1.45434660 -2.42198672
 [7]  0.44232862 -0.73804911 -0.36354587  0.39064194 -0.31993512 -1.30809569
[13]  0.11409195  0.43549125 -0.29501115  0.29197212  0.50983934 -0.80452037
[19] -0.61008244  1.80780477  1.31535974 -1.33155401  0.29044725 -0.63380504

[85]  1.03861350  0.89381884  0.86323215 -0.24199953  1.64380126  0.45445204
[91]  1.90708641  0.34088349 -0.25727644 -0.26498359  0.80095645  1.42711451
[97]  1.27998167 -0.54106317 -1.29443674  0.36046722
```

Print the R object

<- means =

Summary Statistics

```
> mean(x)
[1] 0.02149641
```

Sample mean

```
> var(x)
[1] 1.061159
```

Sample variance

A **function in R** that calculate the mean:
mean(my sample)

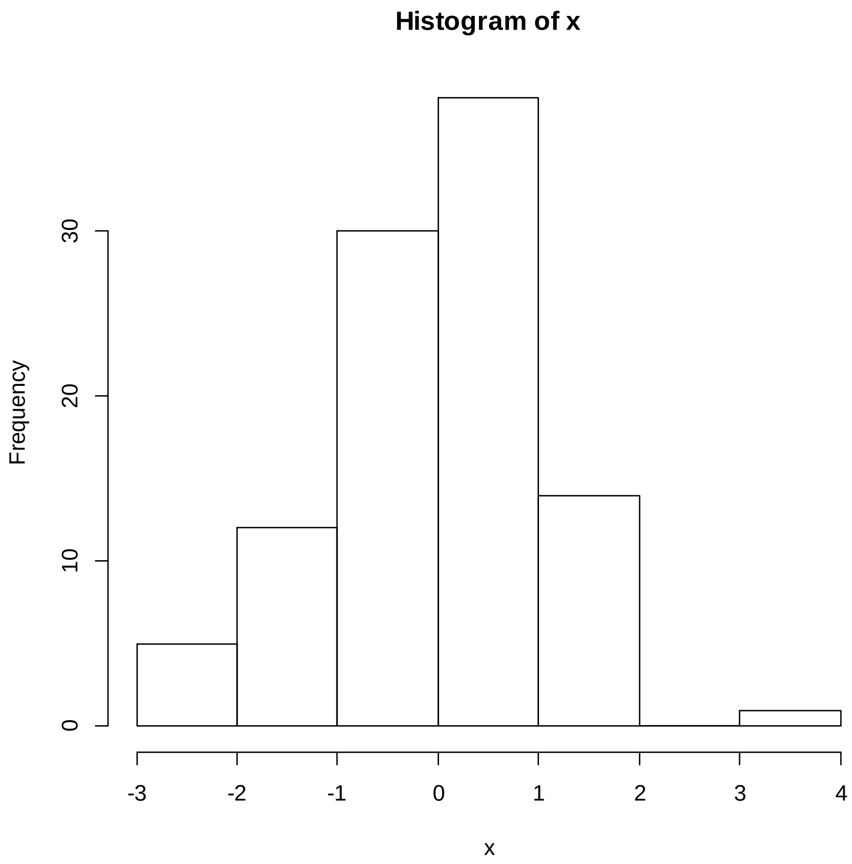
A **function in R** that calculate the variance:
var(my sample)

Histogram of the Sample

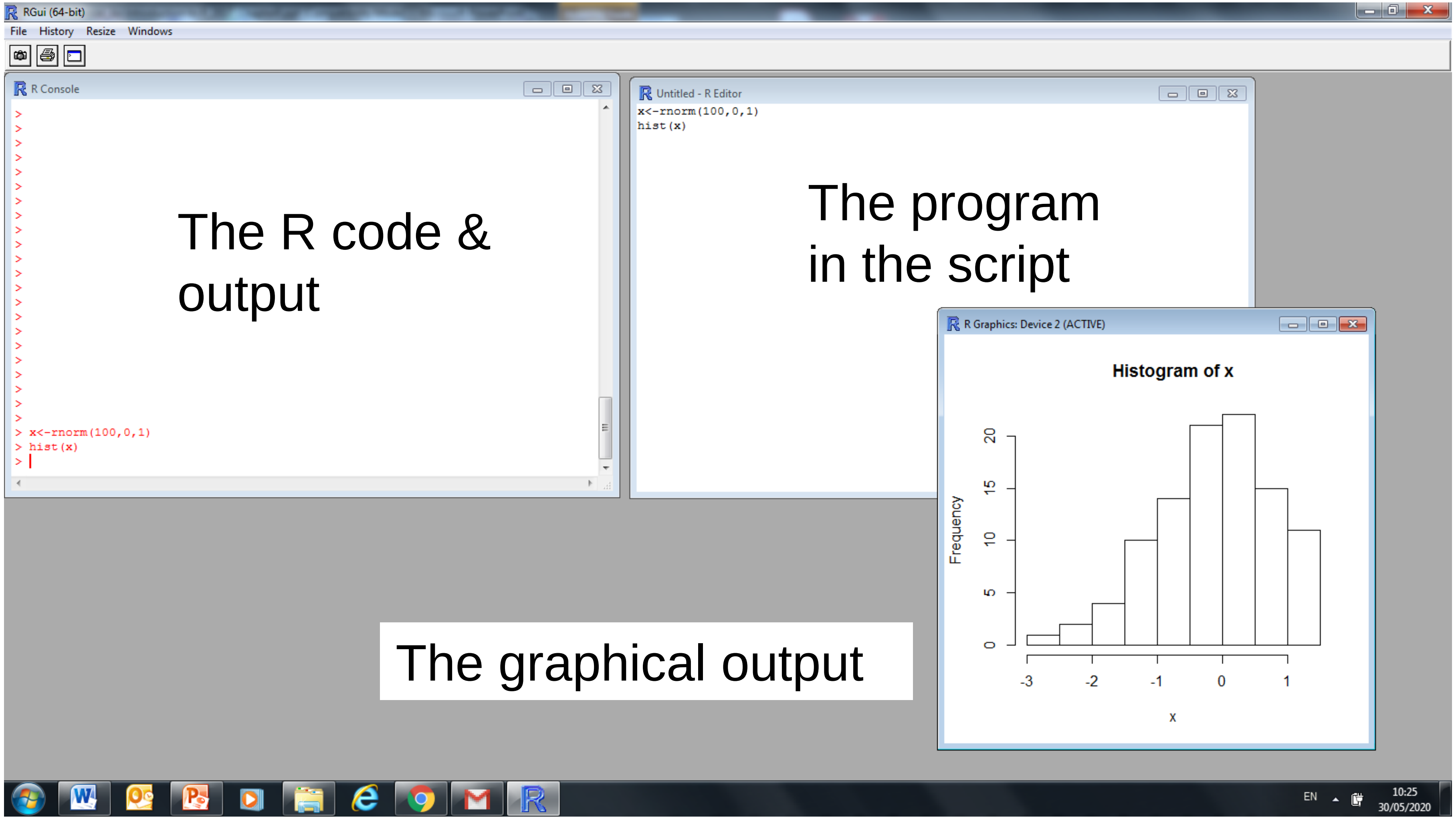
> **hist(x)**

A function in R that produces a histogram

x: R object that contains the data



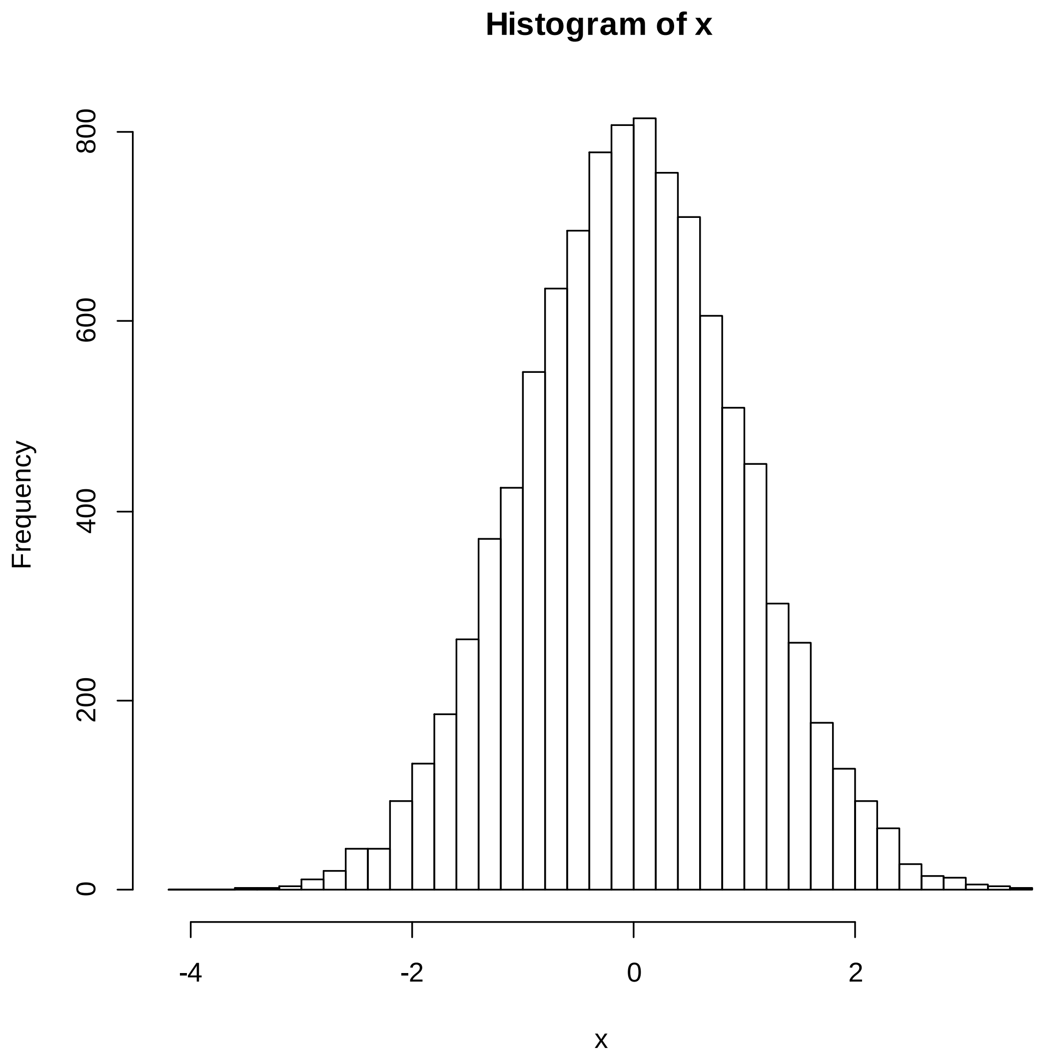
Histogram of the Sample



Histogram of the Sample

```
> x<-rnorm(10000,0,1)
> mean(x)
[1] -0.01259969
> var(x)
[1] 0.9871957
> hist(x,nclass=50)
```

A function in R that produces a histogram



Controlling the Graphical Output

```
> par(mfrow=c(2,2))
```

Split the graphical window to a panel with 2 rows and 2 columns.

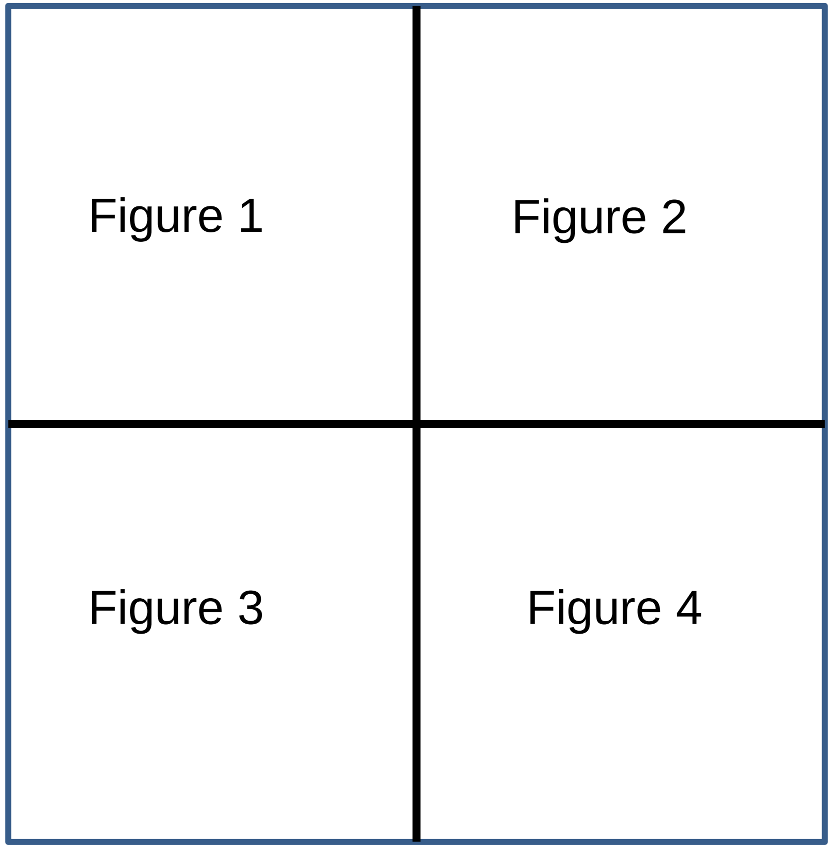
In general:

```
> par(mfrow=c(n,m))
```

Number of rows

Number of columns

mfrow=multi-figures by row



Example: working with data

The cars Dataset in R

1. Write **cars** in the script window.
2. Submit

> cars

	speed	dist
1	4	2
2	4	10
3	7	4
4	7	22
5	8	16
6	9	10
.	.	.
.	.	.
48	24	93
49	24	120
50	25	85

Data frame in R

> help(cars)

Speed and Stopping Distances of Cars

Description

The data give the speed of cars and the distances taken to stop. Note that the data were recorded in the 1920s.

[,1]	speed	numeric	Speed (mph)
[,2]	dist	numeric	Stopping distance (ft)

DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson SAMRC advancinglife

The cars Dataset in R: the $ sign

```
> speed
Error: object 'speed' not found
> cars$speed
 [1]  4  4  7  7  8  9 10 10 10 11 11 12 12 12 12 13 13 13 13 14 14 14 14 15 15
[26] 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 24 24 24 24 25
```

cars$speed: the variable speed **in** the object cars

The cars Dataset in R: Creating a New R Object

```
> cars[,1]
 [1]  4  4  7  7  8  9 10 10 10 11 11 12 12 12 12 13 13 13 13 14 14 14 14 15 15
[26] 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 24 24 24 24 25

> x=cars[,1]

> print(x)
 [1]  4  4  7  7  8  9 10 10 10 11 11 12 12 12 12 13 13 13 13 14 14 14 14 15 15
[26] 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 24 24 24 24 25
```

Basic Plot and Descriptive Statistics

- Analysis:
  - What is the average speed of the cars ?
  - What is the variance of the cars' speed ?
  - What is the min. (max.) speed ?

  - What is the association between speed and stopping distance ?

Descriptive Statistics

```
> mean(cars$speed)
[1] 15.4

> max(cars$speed)
[1] 25

> min(cars$speed)
[1] 4
```

The variable speed **in** the dataset cars

```
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

attach(data)

```
> attach(cars)
```

Tells R to work with the dataset cars.

```
> mean(speed)
[1] 15.4
> max(speed)
[1] 25
> min(speed)
[1] 4
```

We can work with the variables by using thier names.

```
> detach(cars)
```

Stop using the dataset cars.

R Functions

function(data)

A procedure that was programed in R that uses data to produce output.

```
> var(cars$speed)
[1] 27.95918
```

Calculate the variance.

> var(cars$speed)

function

data

R functions

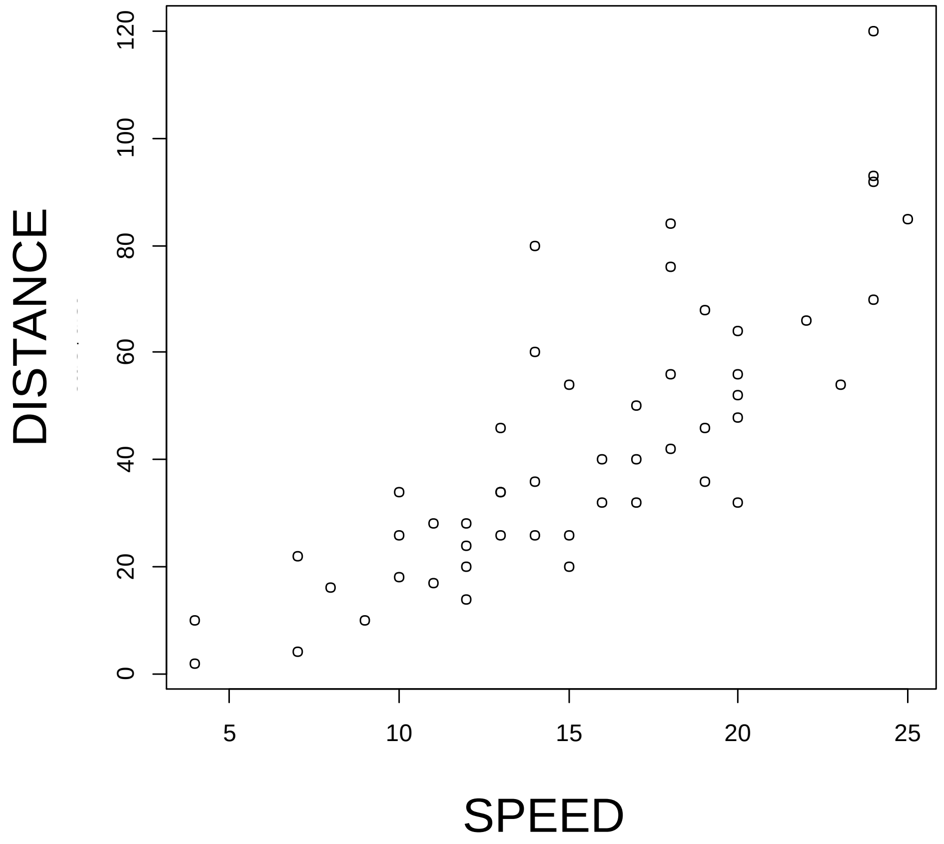
```
> print(cars)
   speed dist
1      4    2
2      4   10
3      7    4
4      7   22
45    23   54
46    24   70
47    24   92
48    24   93
49    24  120
50    25   85
```

```
> cor(cars)
          speed      dist
speed 1.0000000 0.8068949
dist  0.8068949 1.0000000
```

Descriptive Statistics: Basic Plot

Correlation between the variables speed and stopping distance.

```
> cor(cars)
          speed      dist
speed 1.0000000 0.8068949
dist  0.8068949 1.0000000
```

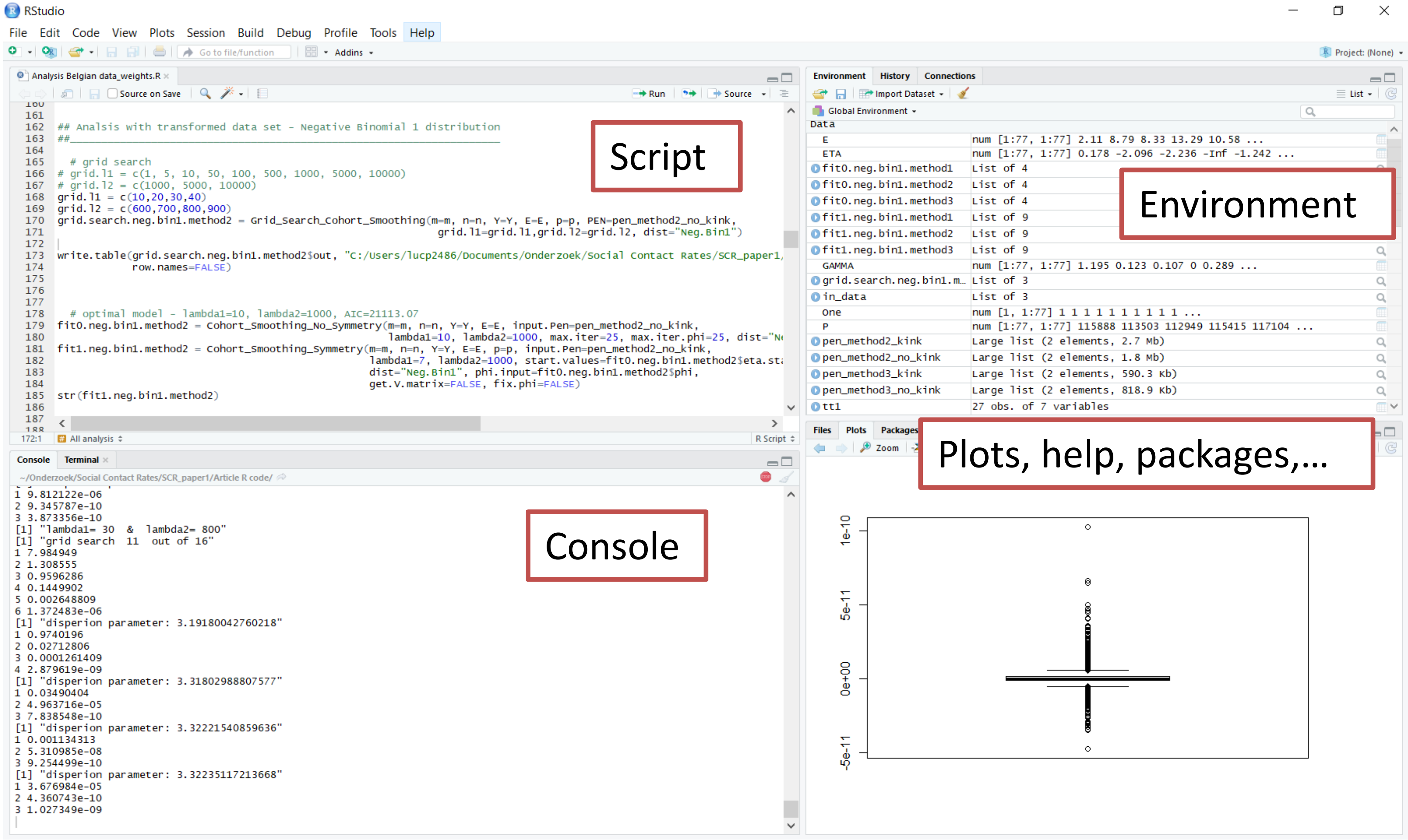


R functions

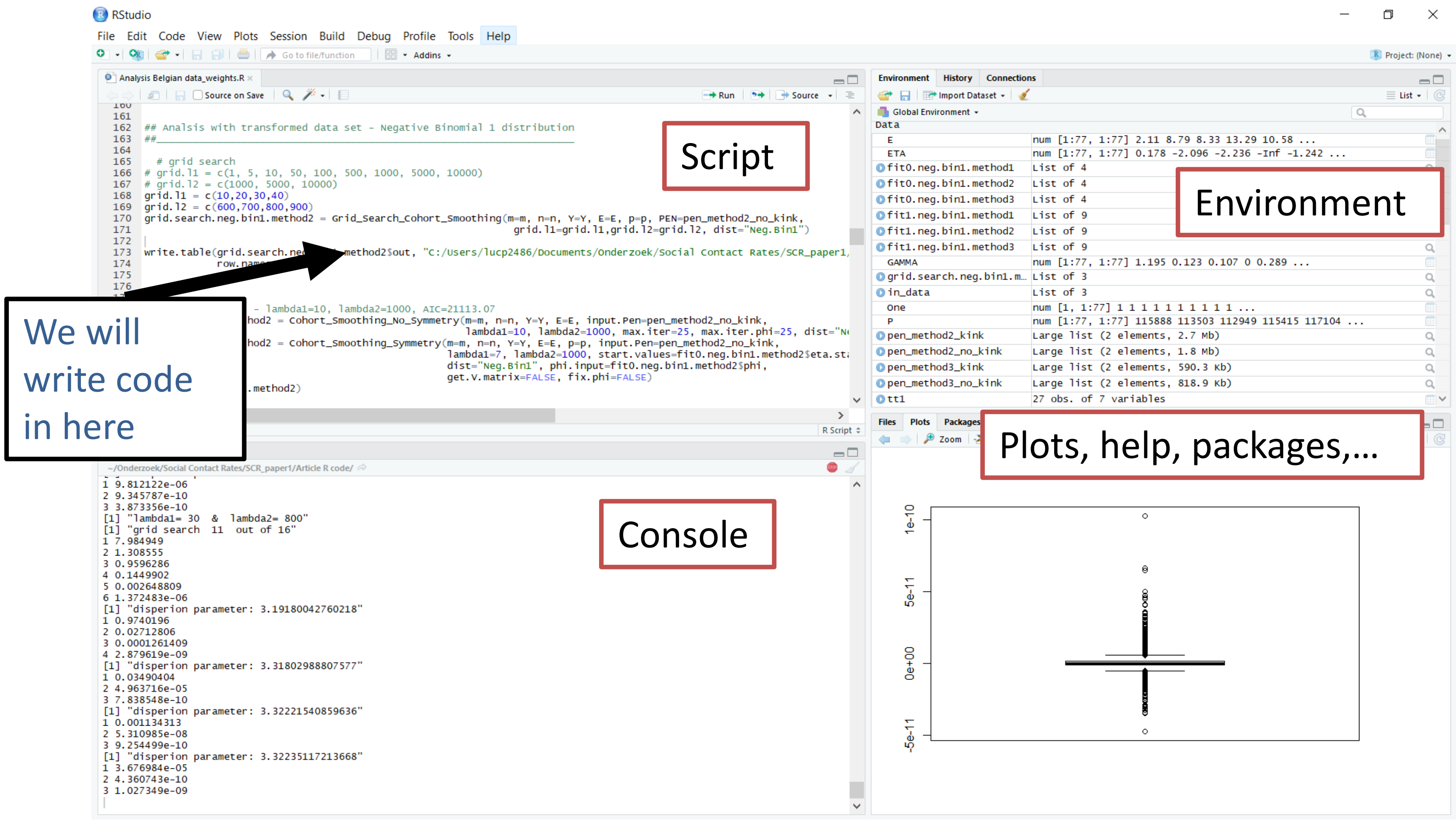
- Can be used for
  - Data analysis: descriptive statistics, testing, modeling, etc.
  - Data manipulation: selection of cases, variables…
  - Data management: reading and writing datasets into/from R.
  - Visualization: plots for the data.
  - …..

Overview

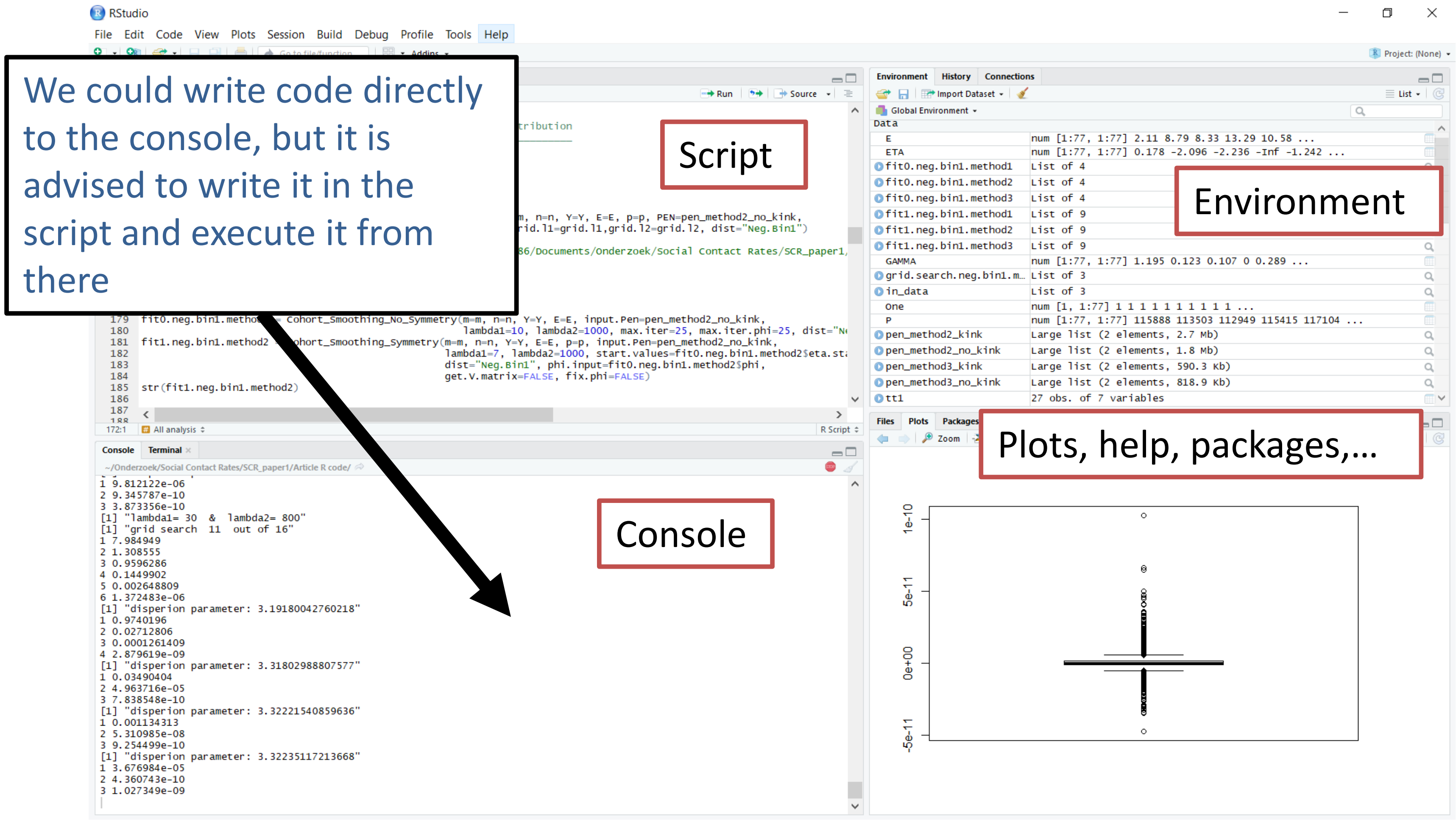
RStudio Interface



RStudio Interface

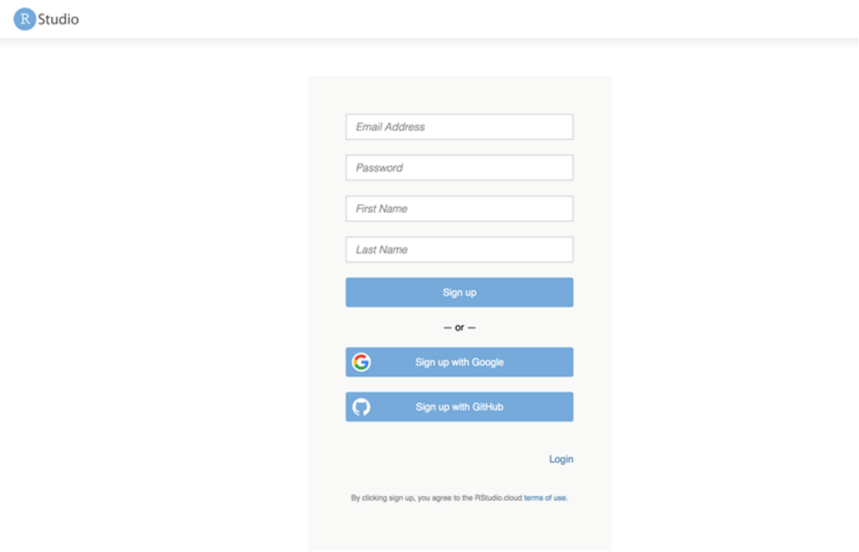


RStudio Interface



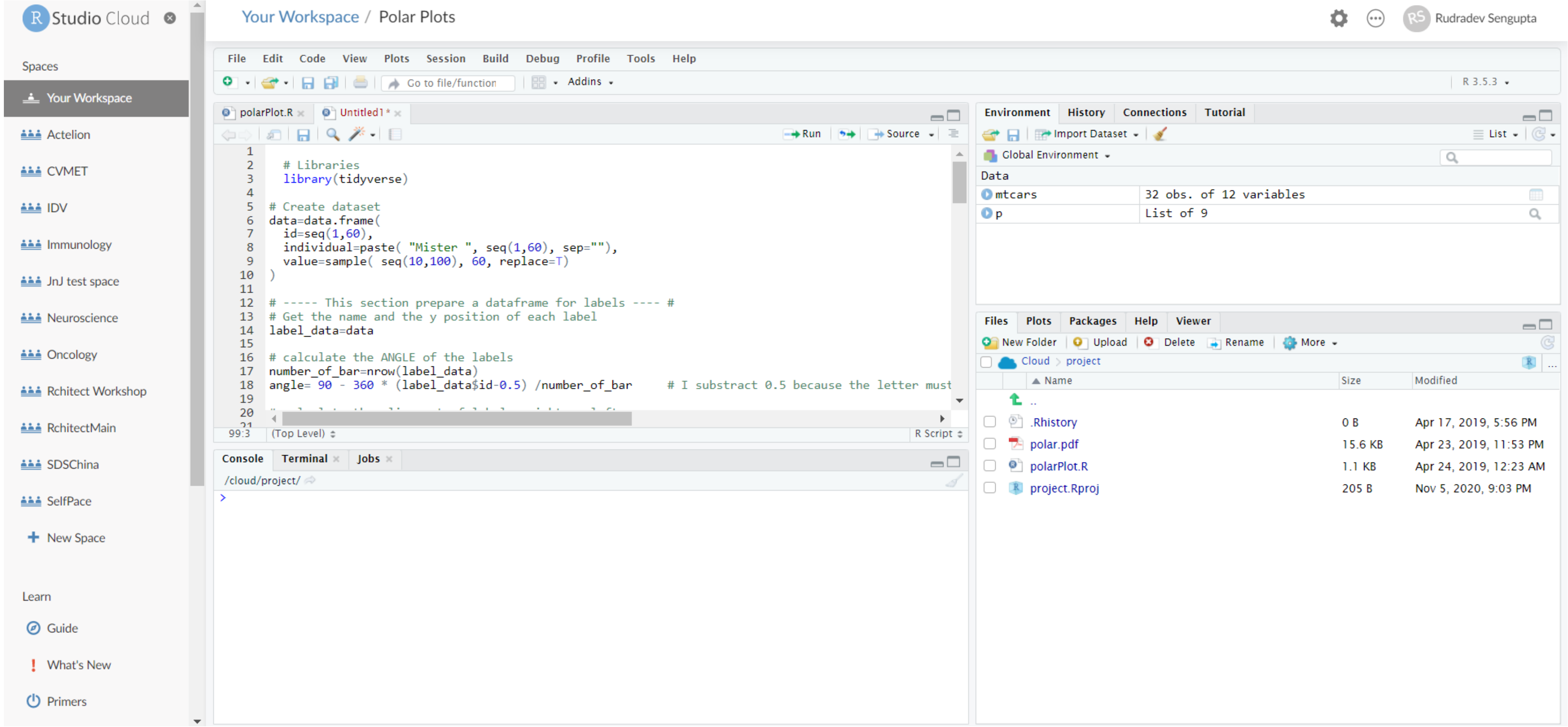
RStudio Cloud

- Why RStudio Cloud?
  - No need to have R installed in your laptop
  - Easy management of R projects
- Easy account creation
- https://rstudio.cloud/



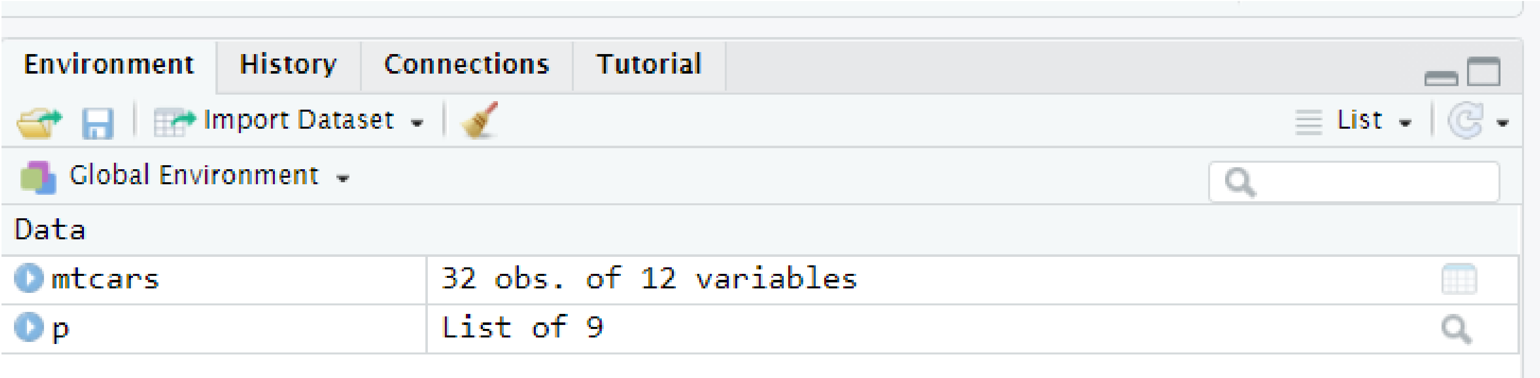
Johnson&Johnson

RStudio Cloud



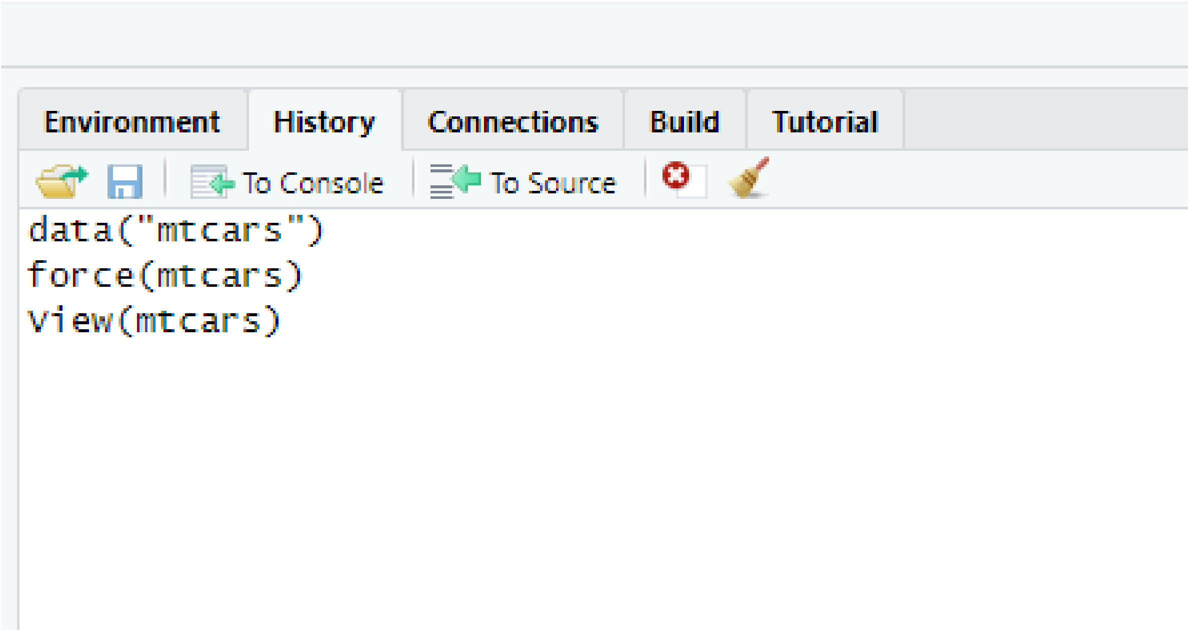
RStudio Workspace

- The workspace tab stores any object, value, function or anything you create during your R session.
- The workspace tab is called “Environment” in recent versions of R.
- Possible to view objects by clicking on them.



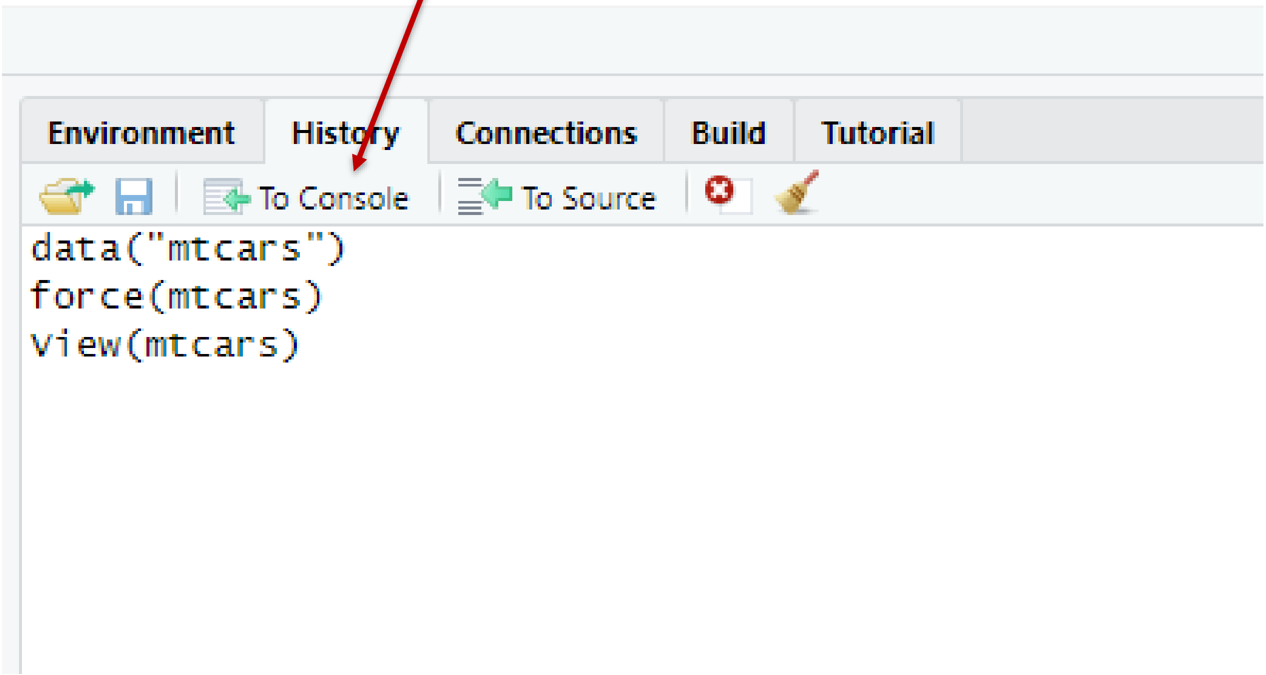
RStudio History

- Captures the commands you run during your R session.
- Easy to send them to console or to script.
- Possibility to save this list of commands.



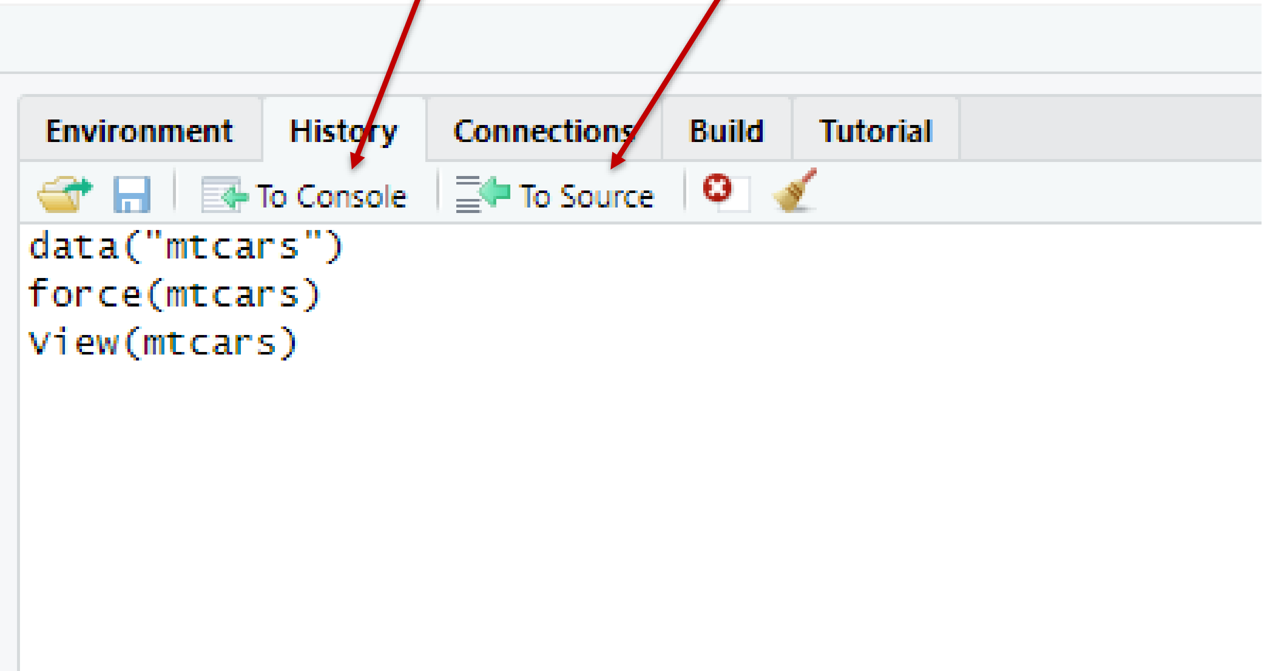
RStudio History

- Captures the commands you run during your R session.
- Easy to send them to console or to script.
- Possibility to save this list of commands.



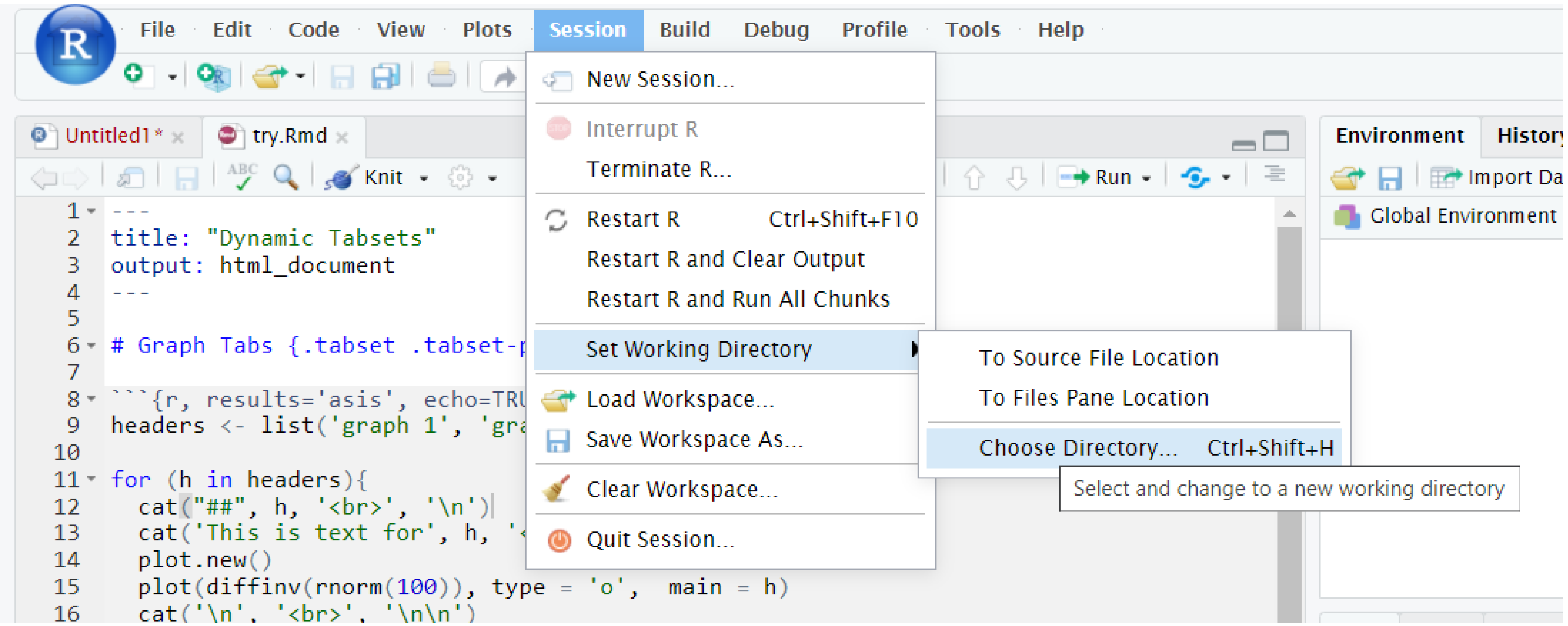
RStudio History

- Captures the commands you run during your R session.
- Easy to send them to console or to script.
- Possibility to save this list of commands.



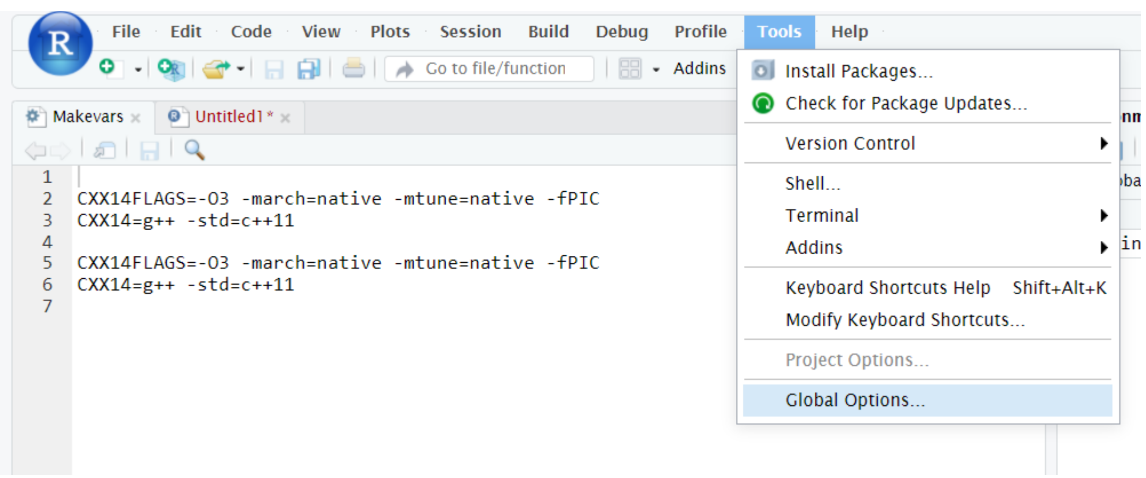
Johnson&Johnson

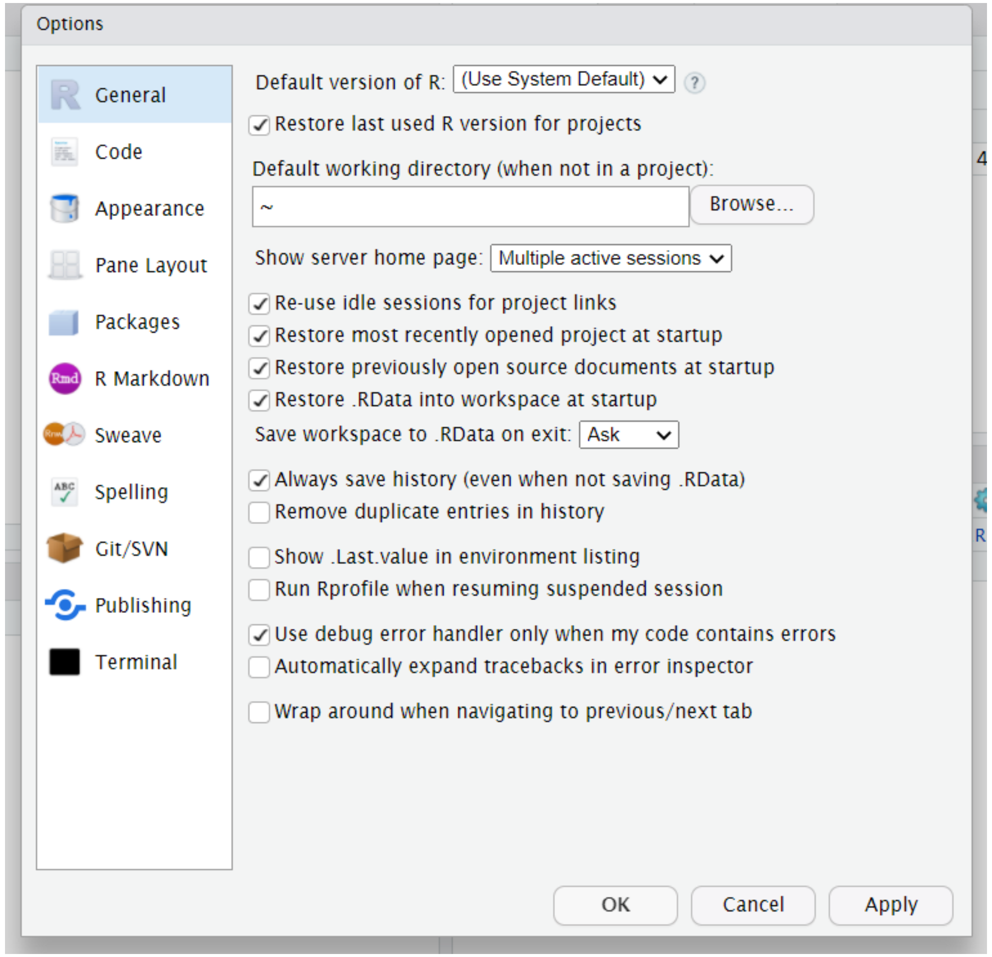
Working Directory



- Shows the working directory (wd): getwd()
- Changes the wd: setwd("C:/myfolder/data")

Global Options





- Specifying different options
- Control R version, appearance, ….

Overview

R Functions

- Two types:
  - Built-in: Already available in R from different packages
    - function: displays the structure of that particular function and what exactly it is doing
    - ?function: shows how to use that particular function
  - Custom:

```
hello <- function() {
  print("Hello, world!")
}


sqrt2 <- function(x, y) {
  temp1 = x*y
  temp1 = abs(temp1)
  temp2 = sqrt(temp1)
  z = sqrt(temp2)
  return(z)
}
```

Parameters of the function

R Functions

- Advantages:
  - No need to change every line of your code
    - Call the function with new parameters
    - Parameters can have default values
  - Possible to call one function within another one or in another script

- More details:

https://www.datacamp.com/community/tutorials/functions-in-r-a-tutorial

R Packages

- R has built-in functions that can be used for data manipulation, graphical exploration, statistical analysis,...
- Many functions are available as packages from different repositories
- Installing and loading packages
- Functions and datasets from specific packages can be used only after the related package has been loaded in the workspace.
- Packages get updated often: Easy way to update via Rstudio
- Keep track of package versions via sessionInfo() to reproduce your work in future.

Repositories

- A global place where packages are located so one can install them from it.
- Possible to have a local repository specific to your organization.
- Option to specify the repository while installing packages:
  - install.packages(pkgs, lib, repos, ….)
- Mostly used:
  - CRAN
  - Bioconductor
  - GitHub

Installing and Loading Packages

> @ijlyttle a package is a like a book, a library is like a library; you use library() to check a package out of the library #rsats
>
> – Hadley Wickham (@hadleywickham) December 8, 2014

Useful Packages

haven - Enables R to read and write data from SAS, SPSS, and Stata

dplyr - Essential shortcuts for subsetting, summarizing, rearranging, and joining together data sets. dplyr is our go to package for fast data manipulation.

tidyr - Tools for changing the layout of your data sets. Use the gather and spread functions to convert your data into the tidy format, the layout R likes best.

ggplot2 - R's famous package for making beautiful graphics. ggplot2 lets you use the grammar of graphics to build layered, customizable plots.

car - car's Anova function is popular for making type II and type III Anova tables.

mgcv - Generalized Additive Models

lme4/nlme - Linear and Non-linear mixed effects models

survival - Tools for survival analysis

caret - Tools for training regression and classification models

shiny - Easily make interactive, web apps with R. A perfect way to explore data and share findings with non-programmers.

R Markdown - The perfect workflow for reproducible reporting. Write R code in your markdown reports. When you run render, R Markdown will replace the code with its results and then export your report as an HTML, pdf, or MS Word document, or a HTML or pdf slideshow. The result? Automated reporting. R Markdown is integrated straight into RStudio.

Useful Packages

haven - Enables R to read and write data from SAS, SPSS, and Stata

dplyr - Essential shortcuts for subsetting, summarizing, rearranging, and joining together data sets. dplyr is our go to package for fast data manipulation.

tidyr - Tools for changing the layout of your data sets. Use the gather and spread functions to convert your data into the tidy format, the layout R likes best.

ggplot2 - R's famous package for making beautiful graphics. ggplot2 lets you use the grammar of graphics to build layered, customizable plots.

car - car's Anova function is popular for making type II and type III Anova tables.

mgcv - Generalized Additive Models

lme4/nlme - Linear and Non-linear mixed effects models

survival - Tools for survival analysis

caret - Tools for training regression and classification models

shiny - Easily make interactive, web apps with R. A perfect way to explore data and share findings with non-programmers.

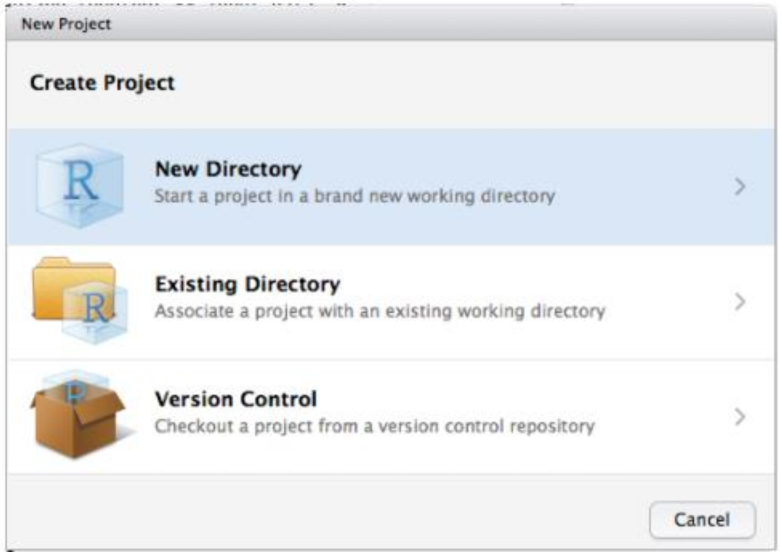
R Markdown - The perfect workflow for reproducible reporting. Write R code in your markdown reports. When you run render, R Markdown will replace the code with its results and then export your report as an HTML, pdf, or MS Word document, or a HTML or pdf slideshow. The result? Automated reporting. R Markdown is integrated straight into RStudio.

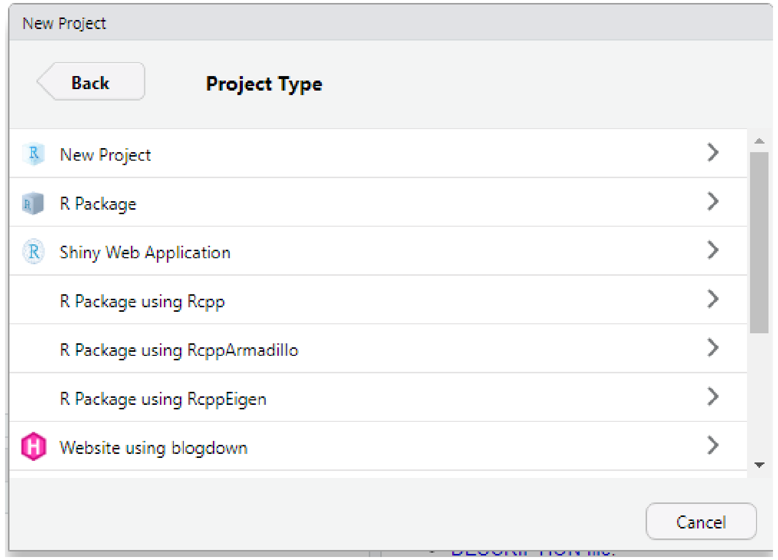
Overview

1. Introduction & Motivation
2. R Software: Basic Introduction
3. R Studio Interface
4. Installing and Loading R Packages
5. **Projects in R and developing R packages**
6. Case Study and analysis with R
7. RMarkdown
8. Advanced RMarkdown
9. Dashboard for Covid-19 using RMarkdown

Working with Projects in R

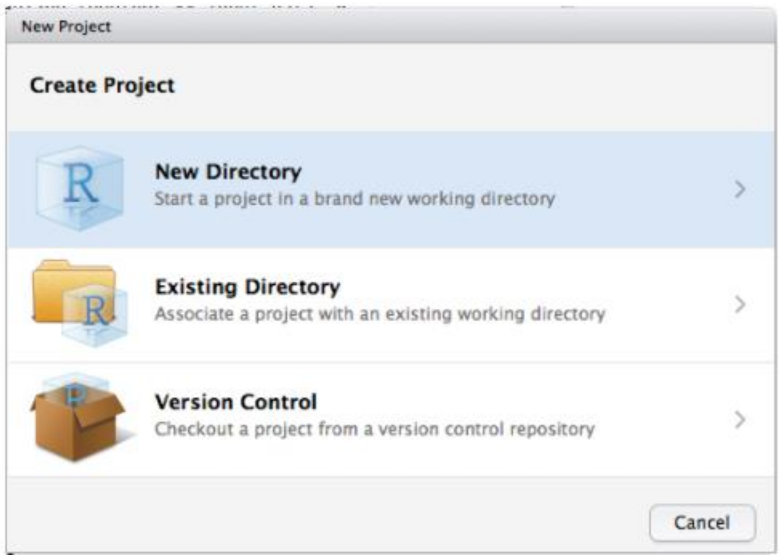
- **Why?**
  - keeps all files associated with a project together - input data, R scripts, analytical results, figures.
- **How?**
  - File -> New Project

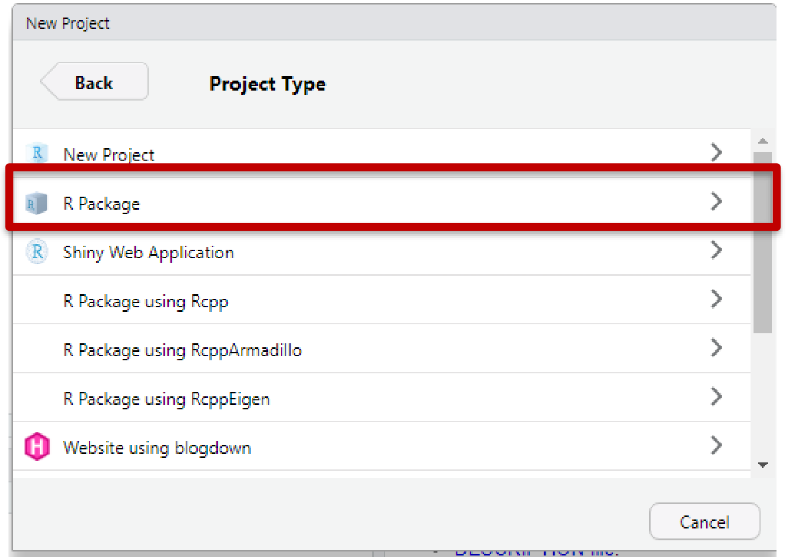




Working with Projects in R

- **Why?**
  - — keeps all files associated with a project together - input data, R scripts, analytical results, figures.
- **How?**
  - — File -> New Project



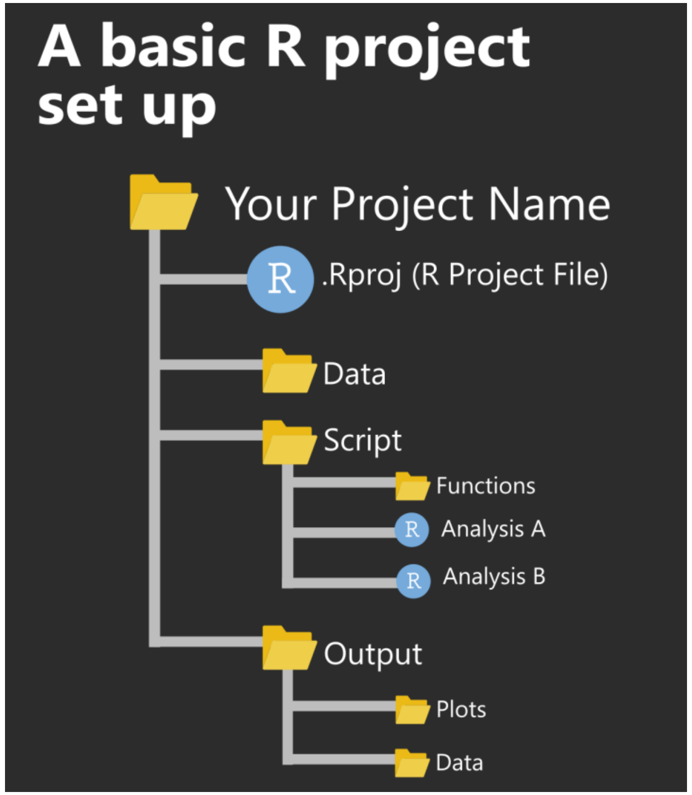


Working with Projects in R

- You can start where you left off when you re-open a project
  - Same working directory and command history (easy relative references to file paths)
  - However, completely fresh environment

- Reproducible work
  - Easy to trace back what was done in the past

- RStudio Reference: https://support.rstudio.com/hc/en-us/articles/200526207-Using-Projects



Developing R Packages

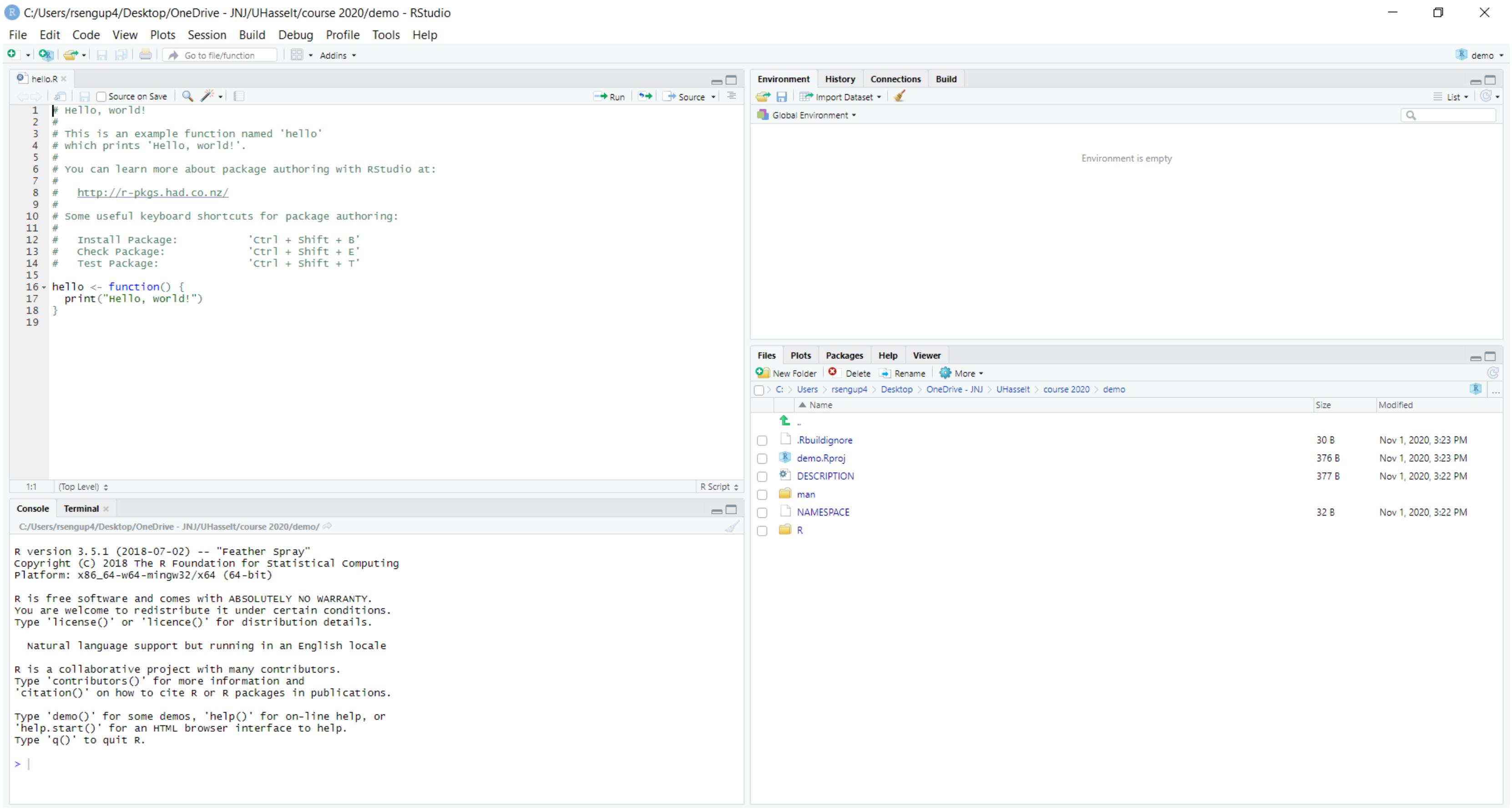
- First step:
  - Call usethis::create_package() or
  - In RStudio, do File -> New Project -> New Directory -> R Package. (This also calls internally usethis::create_package()).

- This produces the smallest possible working package, with three components:
  - An R/ directory and a man/ directory
  - A basic DESCRIPTION file
  - A basic NAMESPACE file
  - Some additional files based on other options selected.

- Keyboard Shortcuts:
  - Install Package: 'Ctrl + Shift + B'
  - Check Package: 'Ctrl + Shift + E'
  - Test Package: 'Ctrl + Shift + T'

Developing R Packages

- Select “R package” project type.

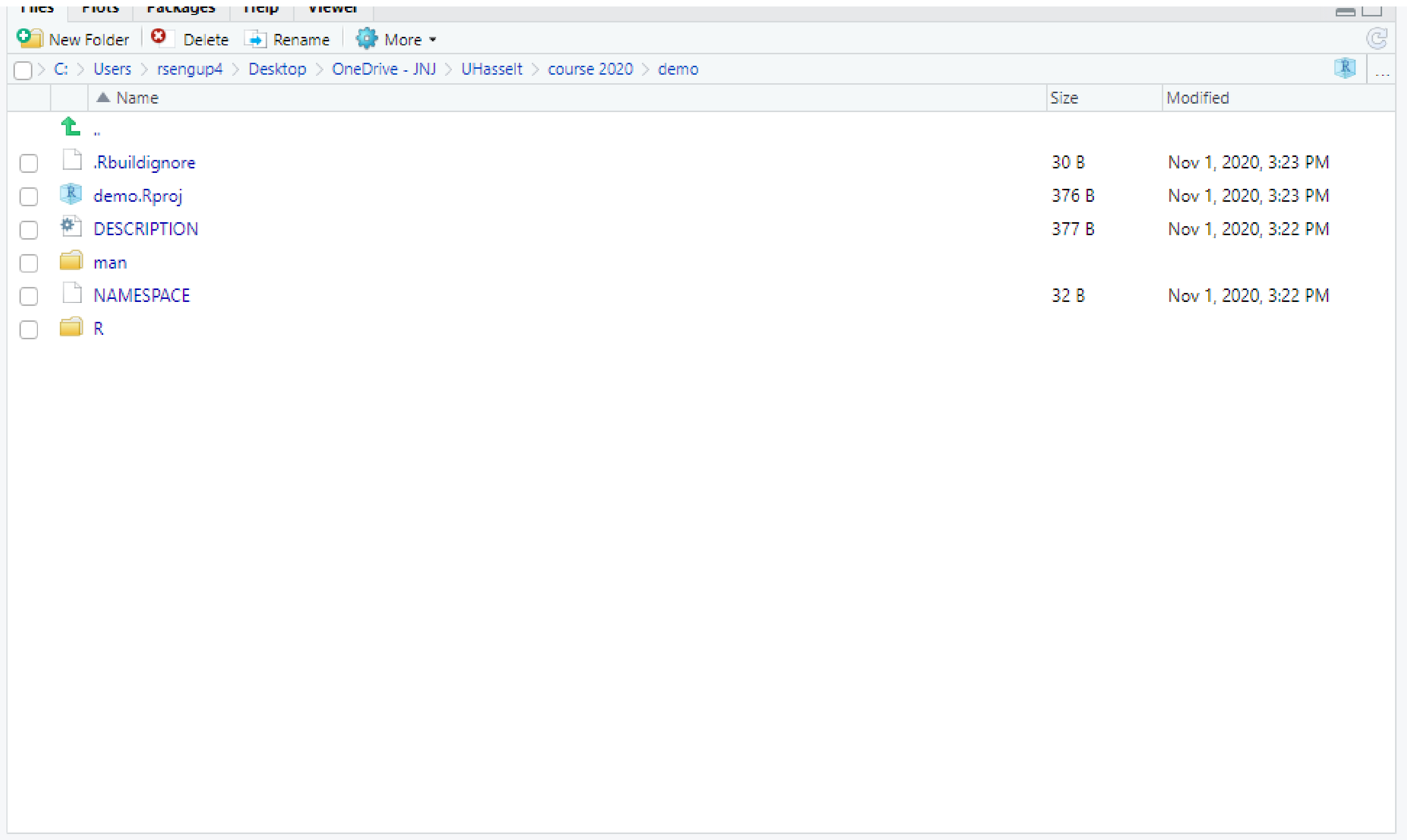


Developing R Packages

- The 'R' folder contains the code for your functions.

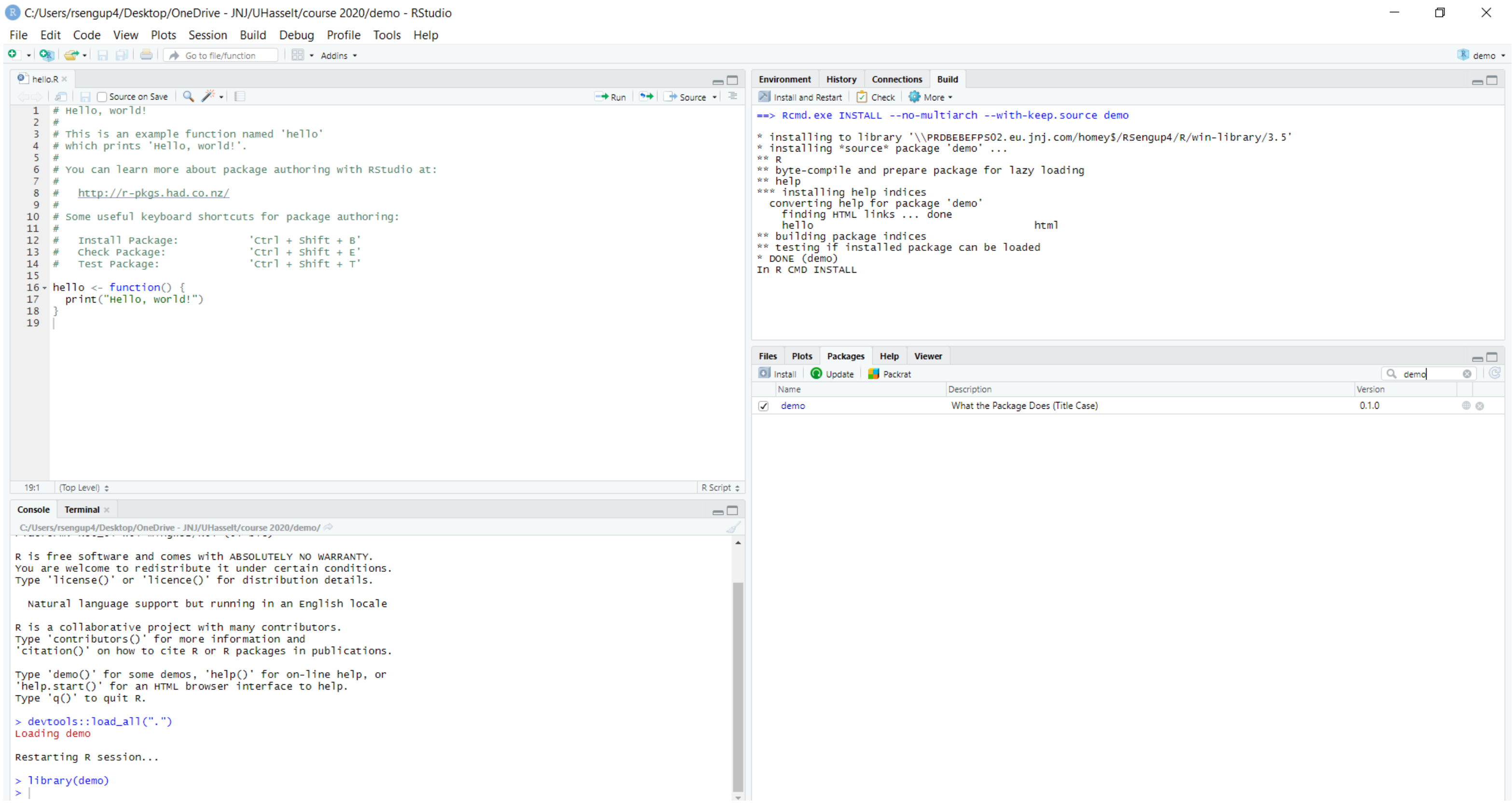
- The 'man' folder contains the help files for each function in your package.

- Need to add a title to each .Rd file in order to compile your package.
  - Roxygen2 package useful for automating this process



Developing R Packages

- Build, Install and load the package



Developing R Packages

Resources:

- https://hilaryparker.com/2014/04/29/writing-an-r-package-from-scratch/ (first 5 steps – great starting point)
- https://support.rstudio.com/hc/en-us/articles/200486488-Developing-Packages-with-RStudio
- http://web.mit.edu/insong/www/pdf/rpackage_instructions.pdf
- https://r-pkgs.org/intro.html (well-documented details)

Overview

1. Introduction & Motivation
2. R Software: Basic Introduction
3. R Studio Interface
4. Installing and Loading R Packages
5. Projects in R and developing R packages
6. **Case Study and analysis with R**
7. RMarkdown
8. Advanced RMarkdown
9. Dashboard for Covid-19 using RMarkdown

Case Studies (I)

- Datasets from R packages:
  - mtcars: on the performance of car models from 1974 Motor Trend US magazine
  - diamonds: shows the prices of over 50,000 diamonds with associated attributes

- First look:

```
> tibble(mtcars)
# A tibble: 32 x 11
     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1  21       6  160    110  3.9   2.62  16.5     0     1     4     4
 2  21       6  160    110  3.9   2.88  17.0     0     1     4     4
 3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
 4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1
 5  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2
 6  18.1     6  225    105  2.76  3.46  20.2     1     0     3     1
 7  14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
 8  24.4     4  147.    62  3.69  3.19  20       1     0     4     2
 9  22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
10  19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4
# i 22 more rows
# i Use `print(n = ...)` to see more rows
```

```
> tibble(diamonds)
# A tibble: 53,940 x 10
   carat cut       color clarity depth table price     x     y     z
   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
 1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
 2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
 3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
 4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
 5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
 6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
 7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
 8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
 9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
# i 53,930 more rows
# i Use `print(n = ...)` to see more rows
```

Case Studies (II)

mpg	Miles/(US) gallon
cyl	Number of cylinders
disp	Displacement (cu.in.)
hp	Gross horsepower
drat	Rear axle ratio
wt	Weight (1000 lbs)
qsec	1/4 mile time
vs	Engine (0 = V-shaped, 1 = straight)
am	Transmission (0 = automatic, 1 = manual)
gear	Number of forward gears

price	price in US dollars
carat	weight of the diamond
cut	quality of the cut
color	diamond color
clarity	measurement of how clear the diamond is
x	length in mm
y	width in mm
z	depth in mm
depth	total depth percentage
table	width of top of diamond relative to widest point

Why use “tibble” (not “head”)?

```
> tibble(mtcars)
# A tibble: 32 x 11
     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1  21       6  160    110  3.9   2.62  16.5     0     1     4     4
 2  21       6  160    110  3.9   2.88  17.0     0     1     4     4
 3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
 4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1
 5  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2
 6  18.1     6  225    105  2.76  3.46  20.2     1     0     3     1
 7  14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
 8  24.4     4  147.    62  3.69  3.19  20       1     0     4     2
 9  22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
10  19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4
# i 22 more rows
# i Use `print(n = ...)` to see more rows
```

Filtering Data (on rows)

- Subset cars with 5 forward gears:

```
> mtcars %>% filter(gear == 5)
                mpg cyl  disp  hp drat    wt qsec vs am gear carb
Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.7  0  1    5    2
Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.9  1  1    5    2
Ford Pantera L 15.8   8 351.0 264 4.22 3.170 14.5  0  1    5    4
Ferrari Dino   19.7   6 145.0 175 3.62 2.770 15.5  0  1    5    6
Maserati Bora  15.0   8 301.0 335 3.54 3.570 14.6  0  1    5    8
```

Filtering Data (on columns)

- Subset cars data with columns mpg, cyl, wt and gear

```
> tibble(mtcars %>% select(mpg, cyl, wt, gear))
# A tibble: 32 x 4
     mpg   cyl    wt  gear
   <dbl> <dbl> <dbl> <dbl>
 1  21       6  2.62     4
 2  21       6  2.88     4
 3  22.8     4  2.32     4
 4  21.4     6  3.22     3
 5  18.7     8  3.44     3
 6  18.1     6  3.46     3
 7  14.3     8  3.57     3
 8  24.4     4  3.19     4
 9  22.8     4  3.15     4
10  19.2     6  3.44     4
# i 22 more rows
# i Use `print(n = ...)` to see more rows
```

Johnson&Johnson

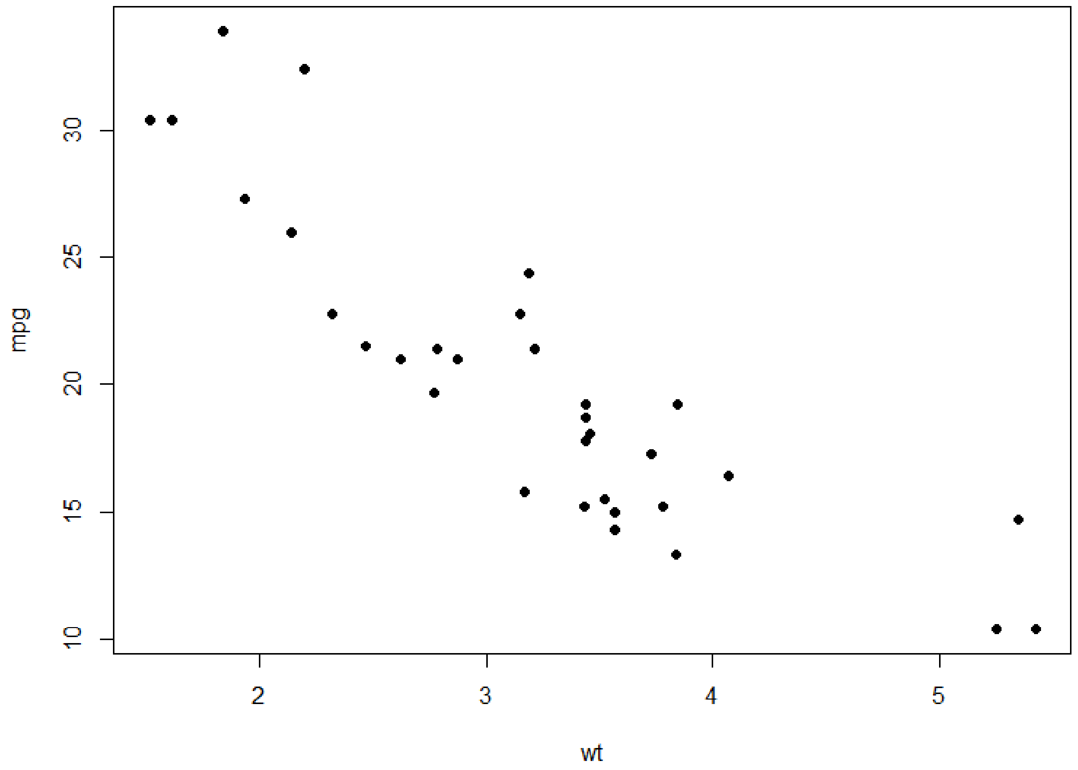
Sorting Data

- Sort cars data from lowest to highest number of gears

```
> mtcars %>% arrange(gear)
                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
```

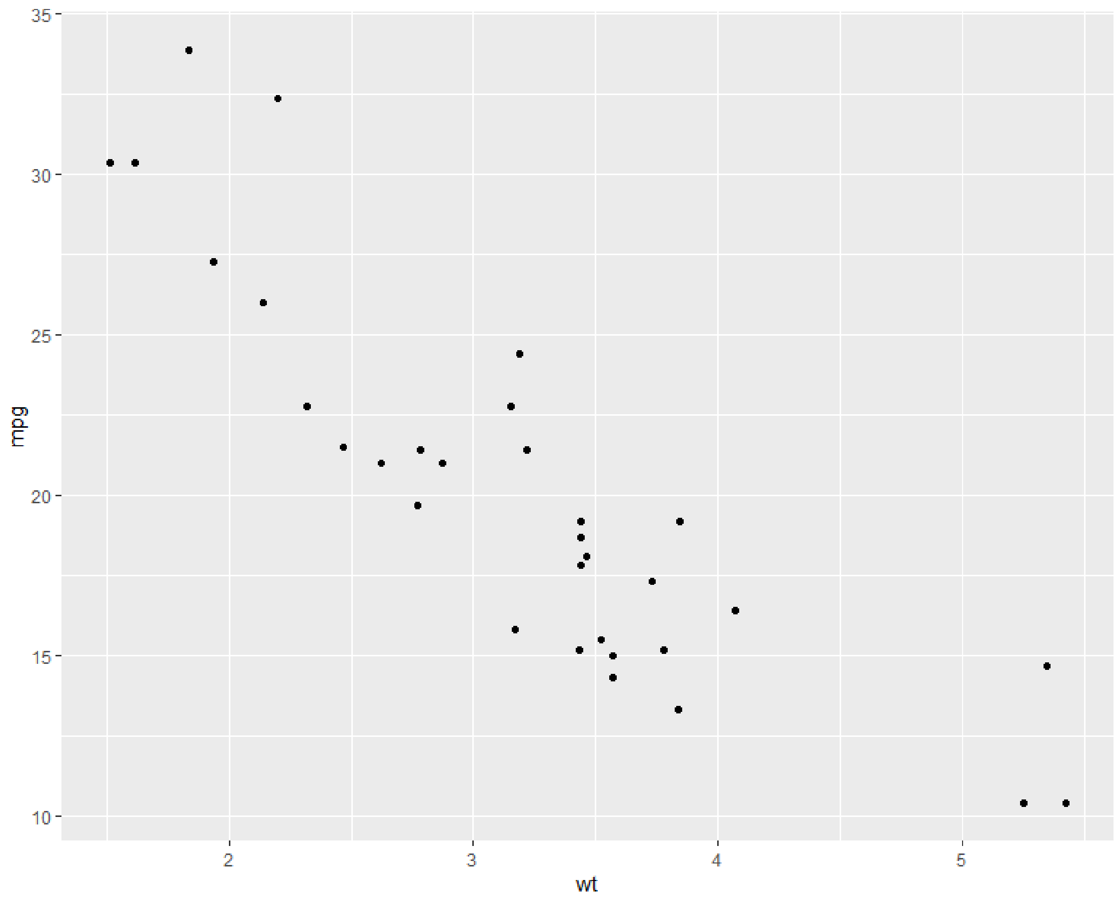
Analysis 1: Continuous vs Continuous

- Linear regression:
- Can we plot the predictor and response variables as a scatterplot ?



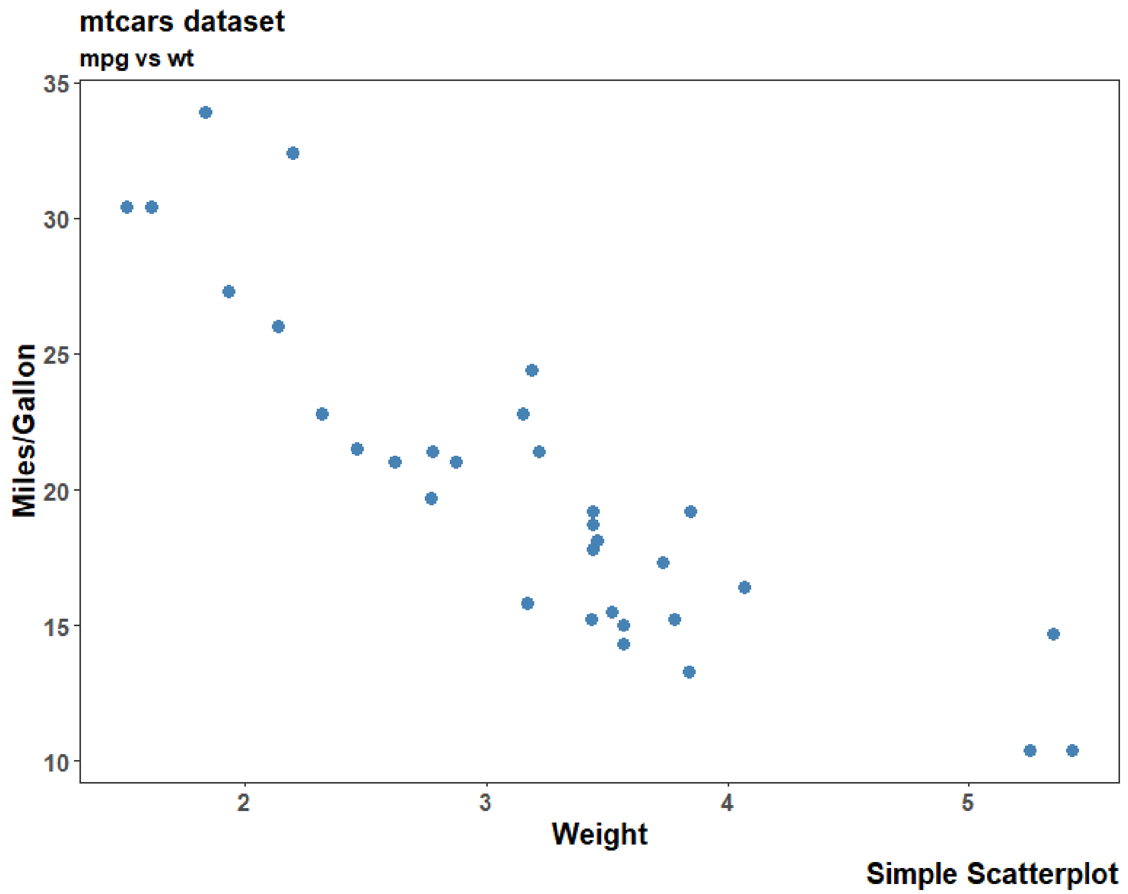
Analysis 1: Continuous vs Continuous

- Linear regression:
- Can we plot the predictor and response variables as a scatterplot ?
- Use ggplot2



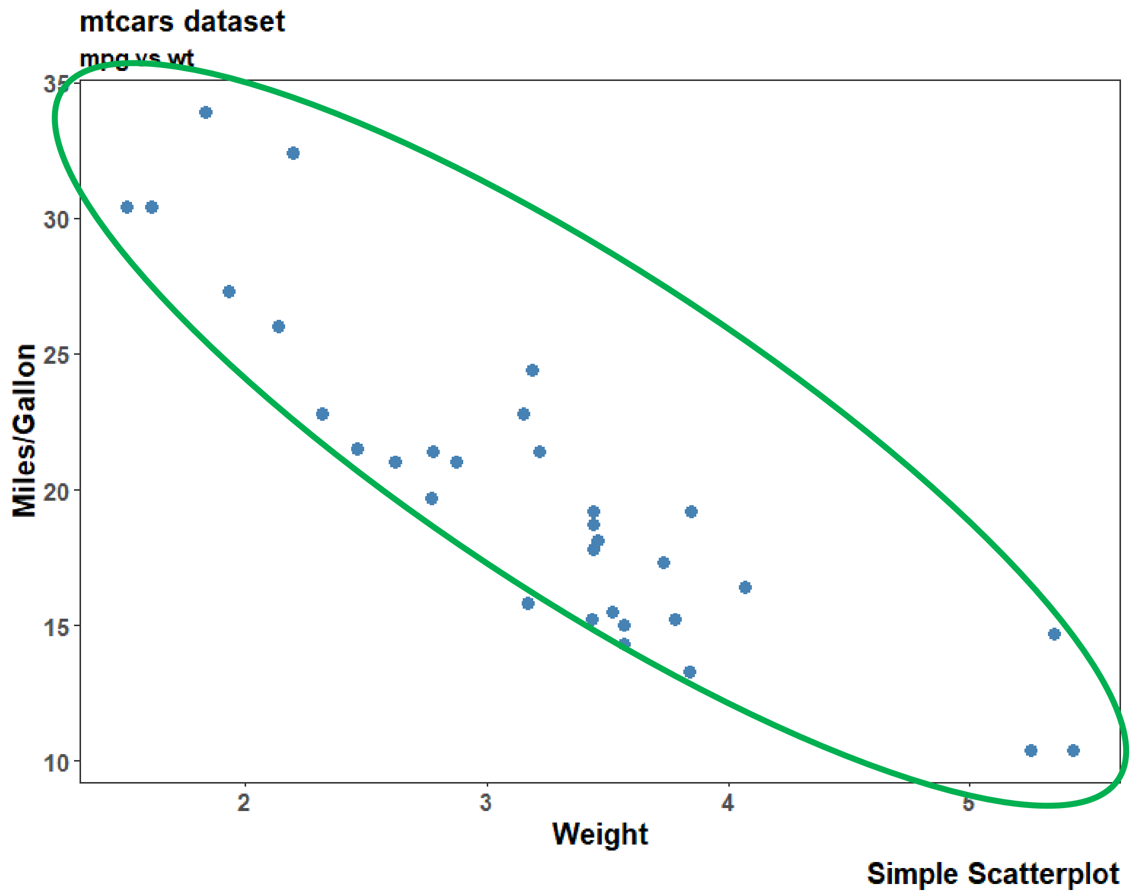
Analysis 1: Continuous vs Continuous

- Linear regression:
- Can we plot the predictor and response variables as a scatterplot ?
- Use ggplot2
- Can we make it better?



Analysis 1: Continuous vs Continuous

- Linear regression:
- Can we plot the predictor and response variables as a scatterplot ?
- Use ggplot2
- Can we make it better?



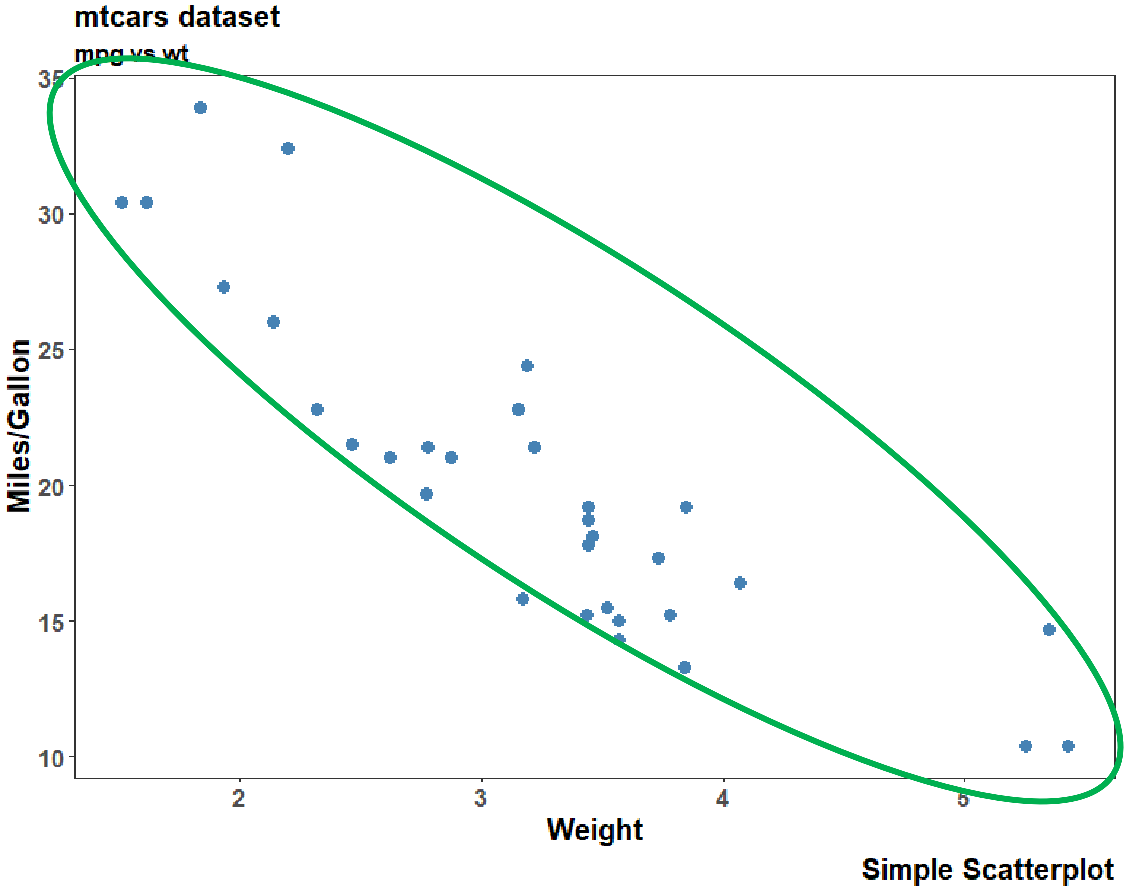
Analysis 1: Linear Regression - lm()

- Linear regression:
  - lm(mpg~wt, data=mtcars)

```
> lm(mpg~wt, data=mtcars)

Call:
lm(formula = mpg ~ wt, data = mtcars)

Coefficients:
(Intercept)           wt
     37.285       -5.344
```



Analysis 1: Linear Regression - lm()

- Linear regression:
  - lm.mod=lm(mpg~wt, data=mtcars)
  - summary(lm.mod)

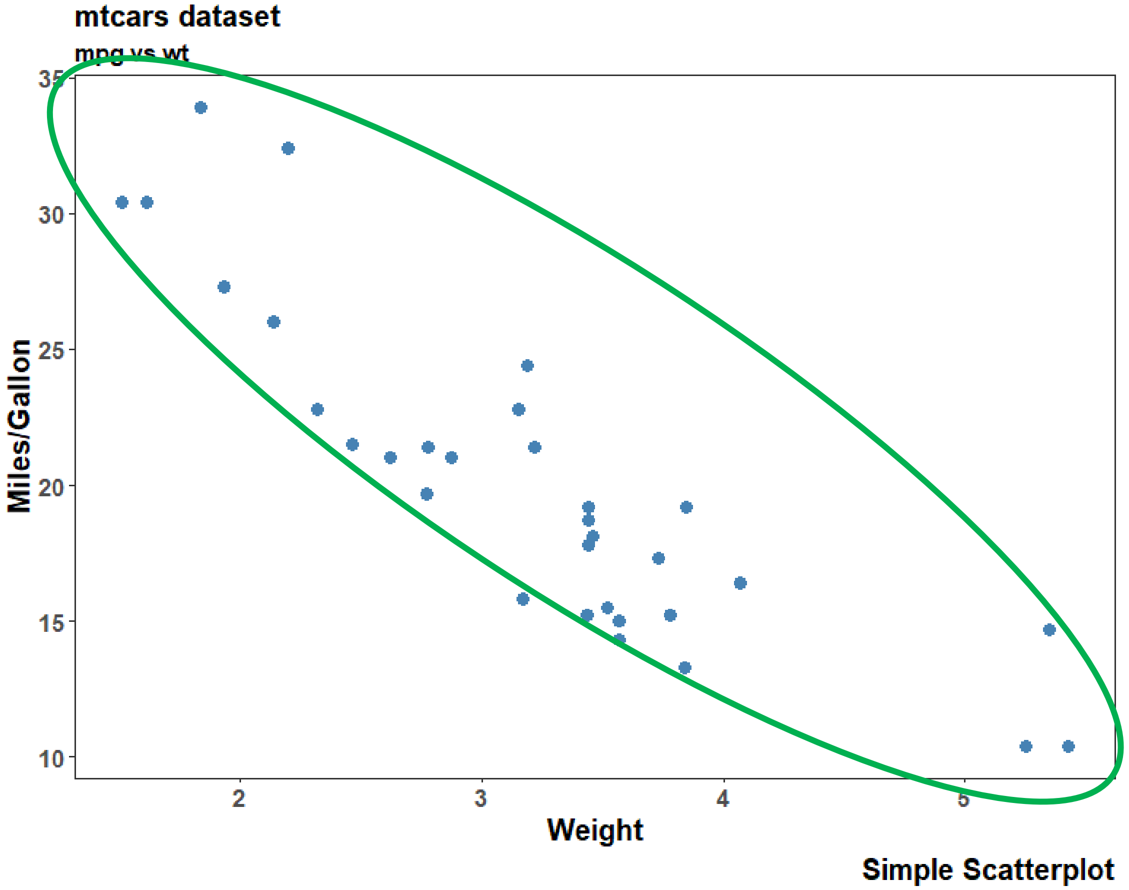
```
> summary(lm.mod)

Call:
lm(formula = mpg ~ wt, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.5432 -2.3647 -0.1252  1.4096  6.8727 

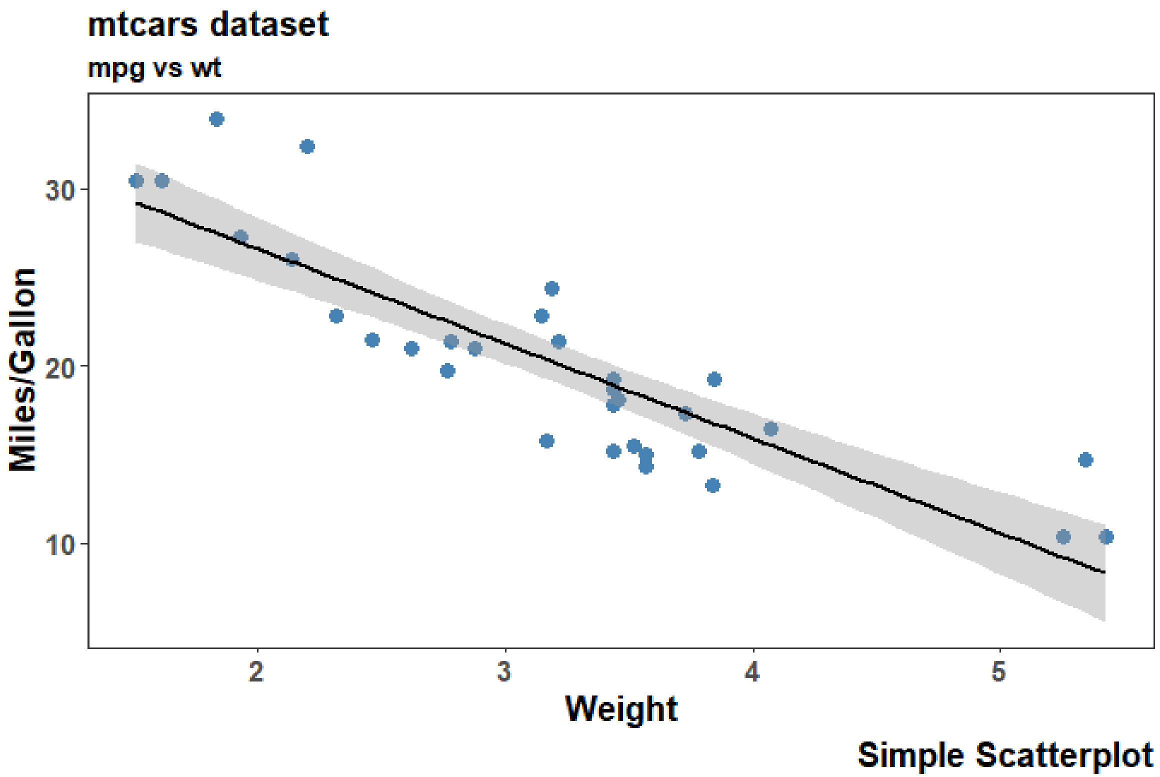
Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
wt           -5.3445     0.5591  -9.559 1.29e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.046 on 30 degrees of freedom
Multiple R-squared:  0.7528,	Adjusted R-squared:  0.7446 
F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```



Visualizing Results

- Best fit line on a scatterplot
- Linear regression workflow:
  - Assign the data and aesthetic mapping
  - Plot scatterplot using geom_point()
  - Add fit line using geom_smooth() and use lm as method.



Linear Regression as glm

- Linear regression is the same as GLM with a “Gaussian” family.
- Assumption: residuals are distributed in a “Gaussian” distribution.
- Two different functions
  - lm.fit=lm(mpg~wt, mtcars)
  - glm.fit=glm(mpg~wt, mtcars, family="gaussian")

R Outputs – lm()

```
> summary(lm.fit)

Call:
lm(formula = mpg ~ wt, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.5432 -2.3647 -0.1252  1.4096  6.8727 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
wt           -5.3445     0.5591  -9.559 1.29e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.046 on 30 degrees of freedom
Multiple R-squared:  0.7528,	Adjusted R-squared:  0.7446 
F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson saMRC advancinglife

R Outputs – glm()

```
> summary(glm.fit)

Call:
glm(formula = mpg ~ wt, family = "gaussian", data = mtcars)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.5432  -2.3647  -0.1252   1.4096   6.8727

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
wt           -5.3445     0.5591  -9.559 1.29e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

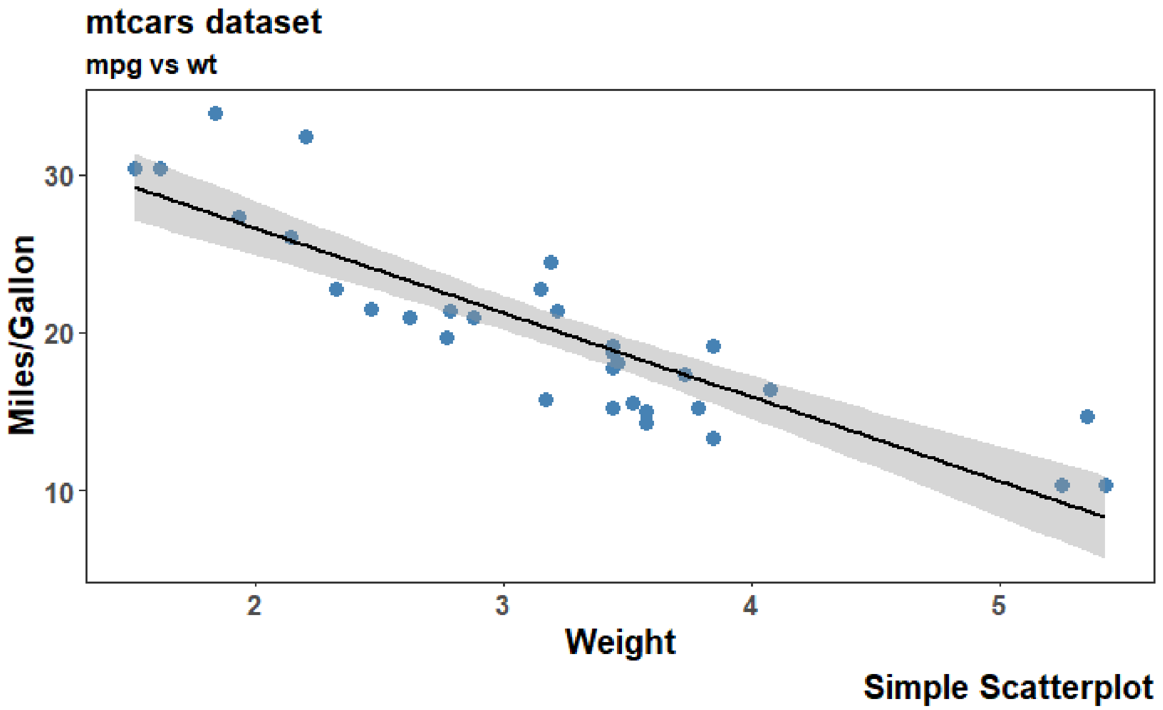
(Dispersion parameter for gaussian family taken to be 9.277398)

    Null deviance: 1126.05  on 31  degrees of freedom
Residual deviance:  278.32  on 30  degrees of freedom
AIC: 166.03

Number of Fisher Scoring iterations: 2
```

Visualizing Results

- Best fit line on a scatterplot
- Linear regression workflow:
  - Assign the data and aesthetic mapping
  - Plot scatterplot using geom_point()
  - Add fit line using geom_smooth() and use glm as method.



Analysis 2: Comparing a Continuous Variable across Groups

- Analysis of Variance (ANOVA) compares whether groups differ in some measured continuous variable.
- ANOVA partitions variation into within-group and across-group variation.
- Conclude that groups are different if the variation across groups is much larger than the variation within groups.

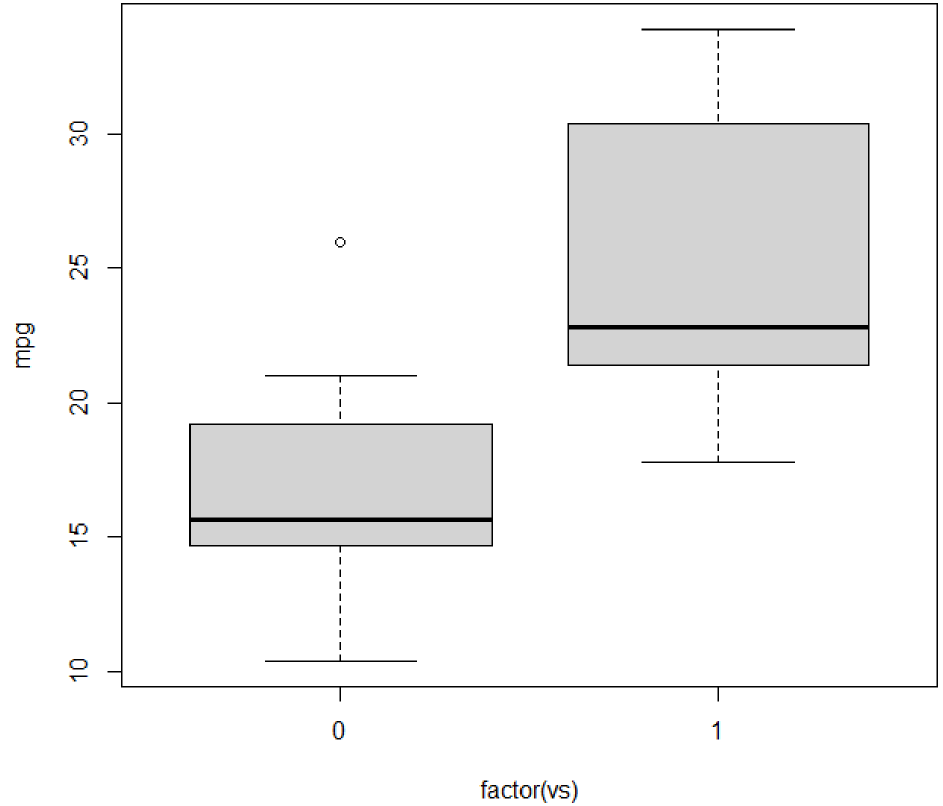
Analysis 2: Comparing a Continuous Variable across Groups

- Effect of shape of the engine (“vs”) on fuel efficiency (“mpg”)
  - vs = 0 if v-shape and 1 = straight
  - class(mtcars$vs) = “numeric”
  - should be considered as a “factor” variable
  - Visualization: boxplot

mpg	Miles/(US) gallon
cyl	Number of cylinders
disp	Displacement (cu.in.)
hp	Gross horsepower
drat	Rear axle ratio
wt	Weight (1000 lbs)
qsec	1/4 mile time
vs	Engine (0 = V-shaped, 1 = straight)
am	Transmission (0 = automatic, 1 = manual)
gear	Number of forward gears

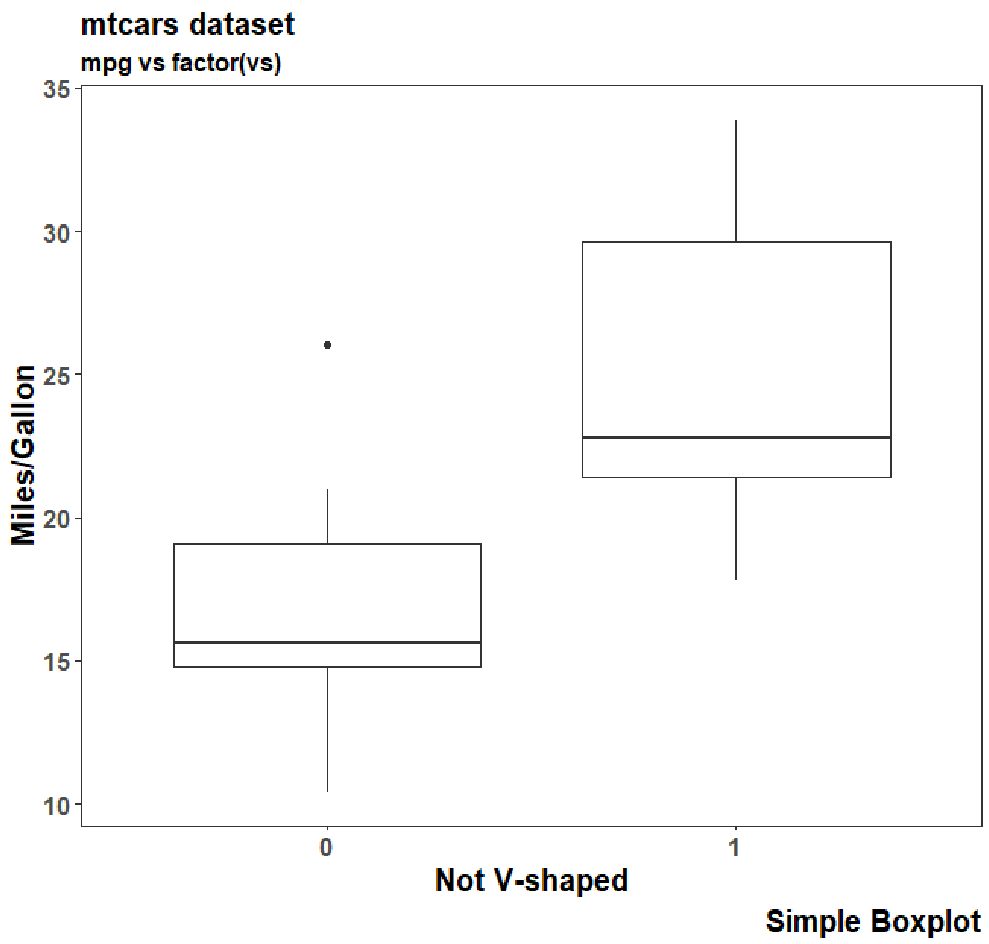
Boxplot: simple

- plot(mpg~factor(vs), data=mtcars)



Boxplot: ggplot2

- ggplot(mtcars, aes(x=factor(vs), y=mpg)) + geom_boxplot() + …

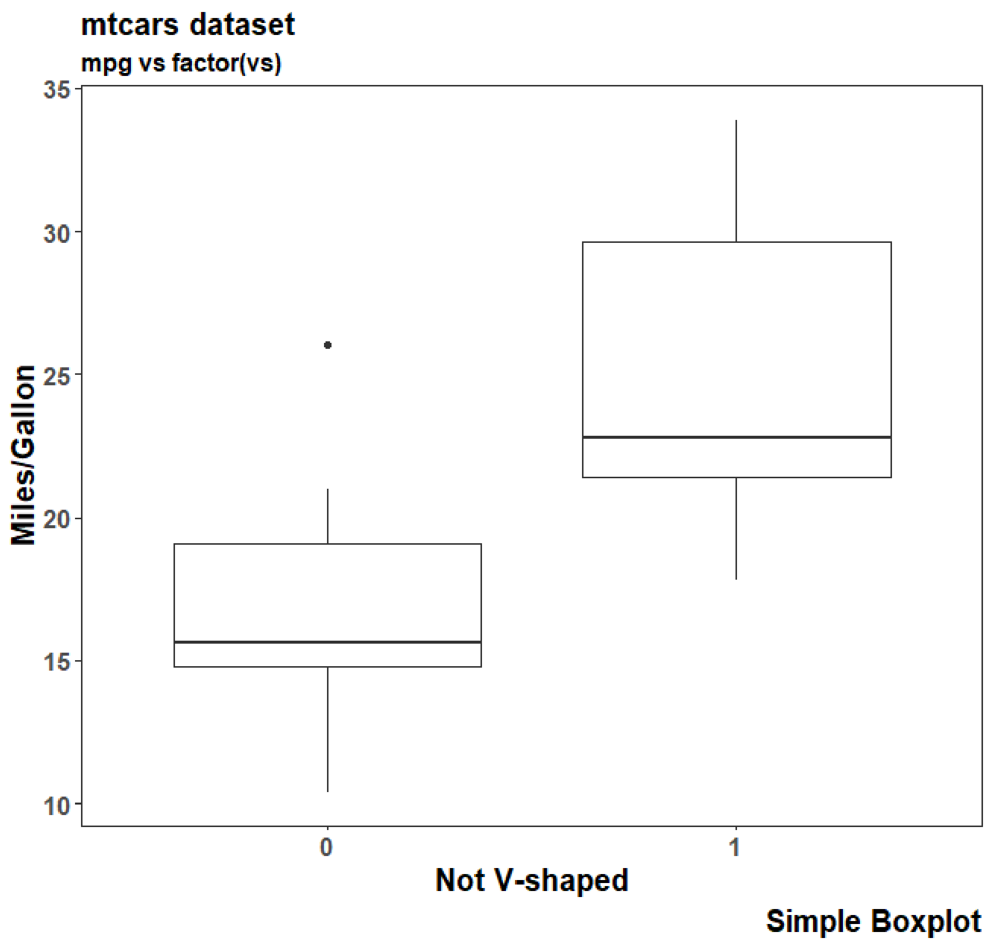


Johnson&Johnson

Running ANOVA

- ggplot(mtcars, aes(x=factor(vs), y=mpg)) + geom_boxplot() + …
- Probable differences in fuel efficiency based on engine shape.
- How to test this hypothesis:

  => ANOVA



Running ANOVA

- aov.mod=aov(mpg~factor(vs), data=mtcars)
- summary(aov.mod)

```
> summary(aov.mod)
            Df Sum Sq Mean Sq F value   Pr(>F)    
factor(vs)   1  496.5   496.5   23.66 3.42e-05 ***
Residuals   30  629.5    21.0                     
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- significant difference in fuel efficiency based on engine shape

ANOVA ⇔ Linear Model

- ANOVA is a type of linear model where the predictor variable is categorical.
- run ANOVA using the lm() function:
  - lm.mod2=lm(mpg~factor(vs), data=mtcars)
  - lm.mod2

```
> lm.mod2

Call:
lm(formula = mpg ~ factor(vs), data = mtcars)

Coefficients:
(Intercept)  factor(vs)1
      16.62         7.94
```

Johnson&Johnson

ANOVA ⇔ Linear Model

- summary(lm.mod2)
- anova(lm.mod2)

```
> summary(aov.mod)
            Df Sum Sq Mean Sq F value   Pr(>F)    
factor(vs)   1  496.5   496.5   23.66 3.42e-05 ***
Residuals   30  629.5    21.0                     
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> summary(lm.mod2)

Call:
lm(formula = mpg ~ factor(vs), data = mtcars)

Residuals:
   Min     1Q Median     3Q    Max 
-6.757 -3.082 -1.267  2.828  9.383 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   16.617      1.080  15.390 8.85e-16 ***
factor(vs)1    7.940      1.632   4.864 3.42e-05 ***

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.581 on 30 degrees of freedom
Multiple R-squared:  0.4409,	Adjusted R-squared:  0.4223 
F-statistic: 23.66 on 1 and 30 DF,  p-value: 3.416e-05
```

t-test instead of F-test

- to test a linear model with categorical variables, summary() picks a ‘reference value’ of that variable and then conducts pairwise comparisons between that reference value and other values to generate t-statistics.

ANOVA ⇔GLM

- glm.mod2=glm(mpg~factor(vs), data=mtcars, family="gaussian")
- anova(glm.mod2, test="F")

```
> anova(glm.mod2, test="F")
Analysis of Deviance Table

Model: gaussian, link: identity

Response: mpg

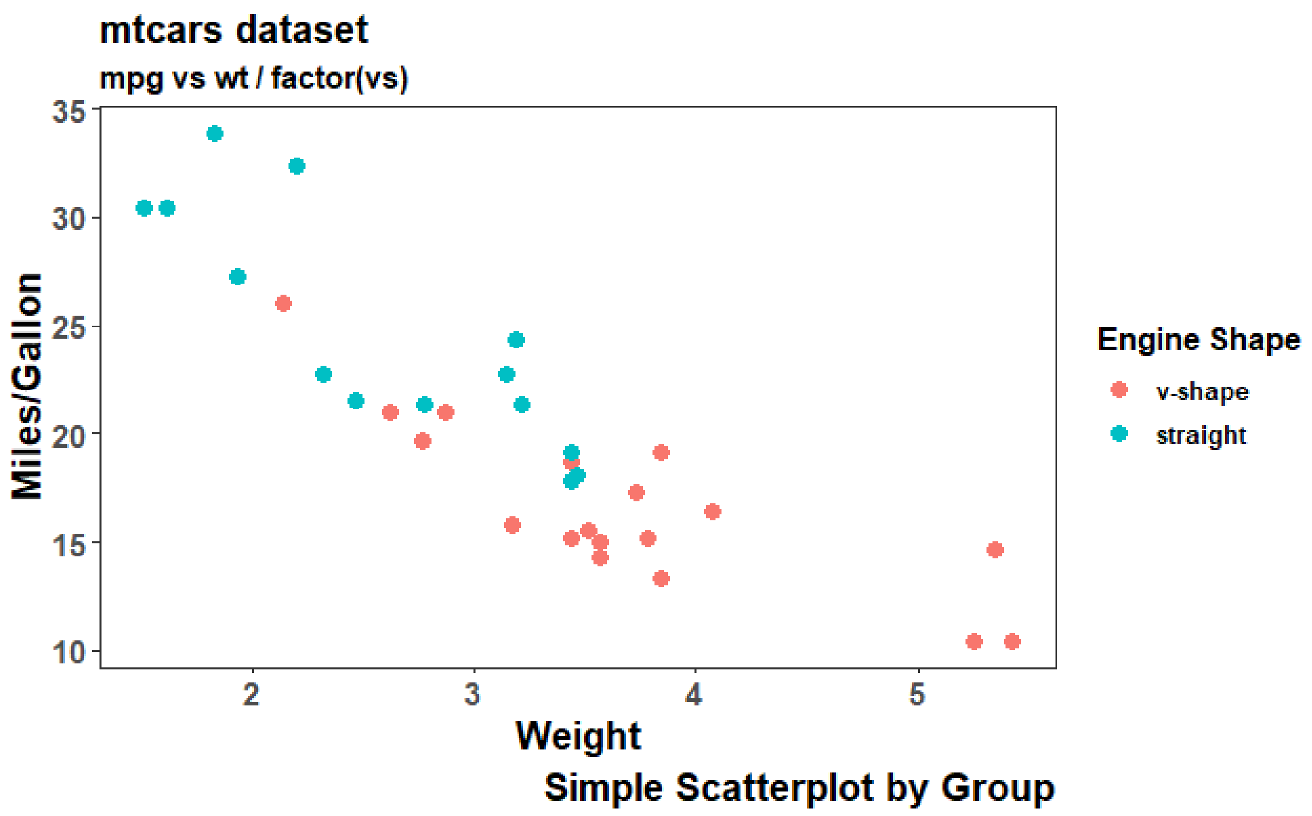
Terms added sequentially (first to last)


           Df Deviance Resid. Df Resid. Dev      F    Pr(>F)
NULL                          31    1126.05
factor(vs)  1   496.53        30     629.52 23.662 3.416e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Multiple Predictor Variables in a Linear Model

- Combine the two previous analysis with Continuous and Categorical predictor variable.

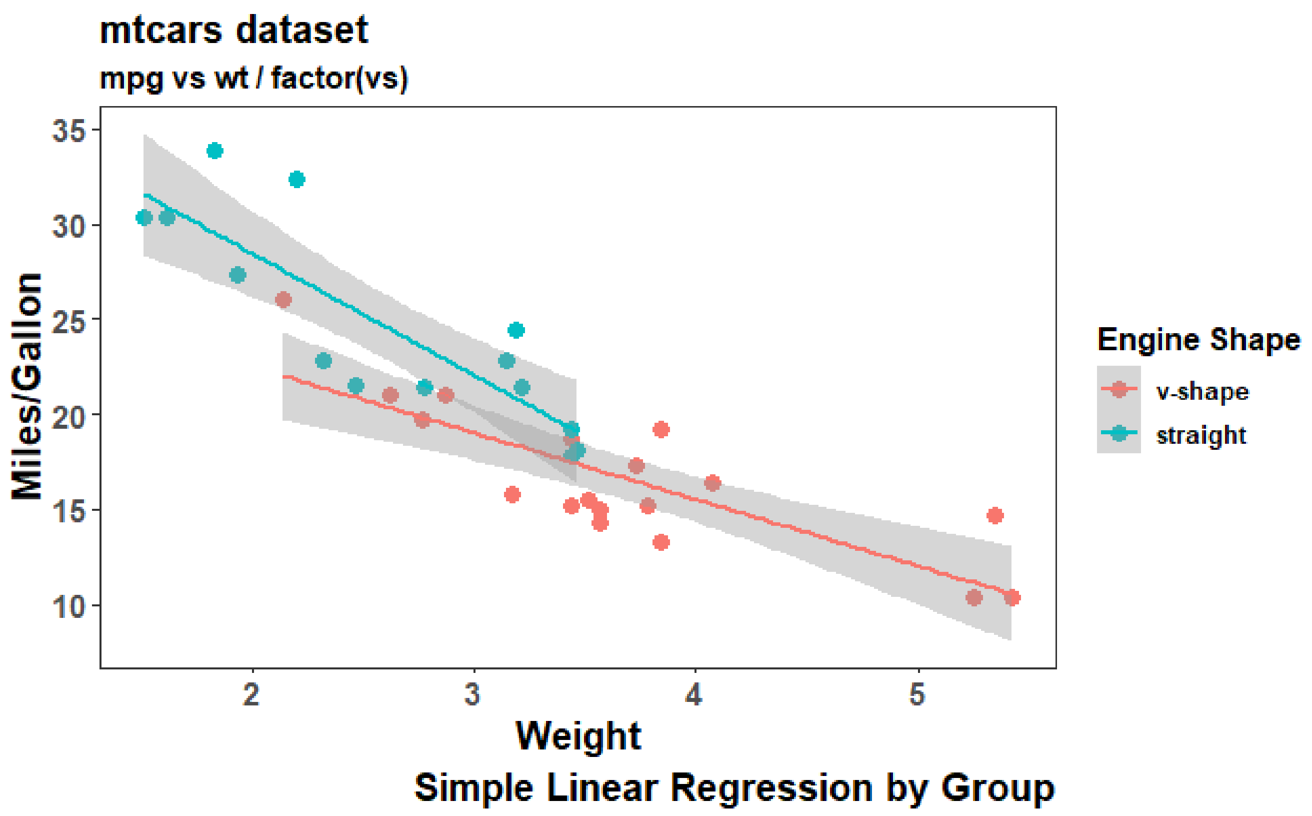
- Grouped scatterplot



Johnson&Johnson

Multiple Predictor Variables in a Linear Model

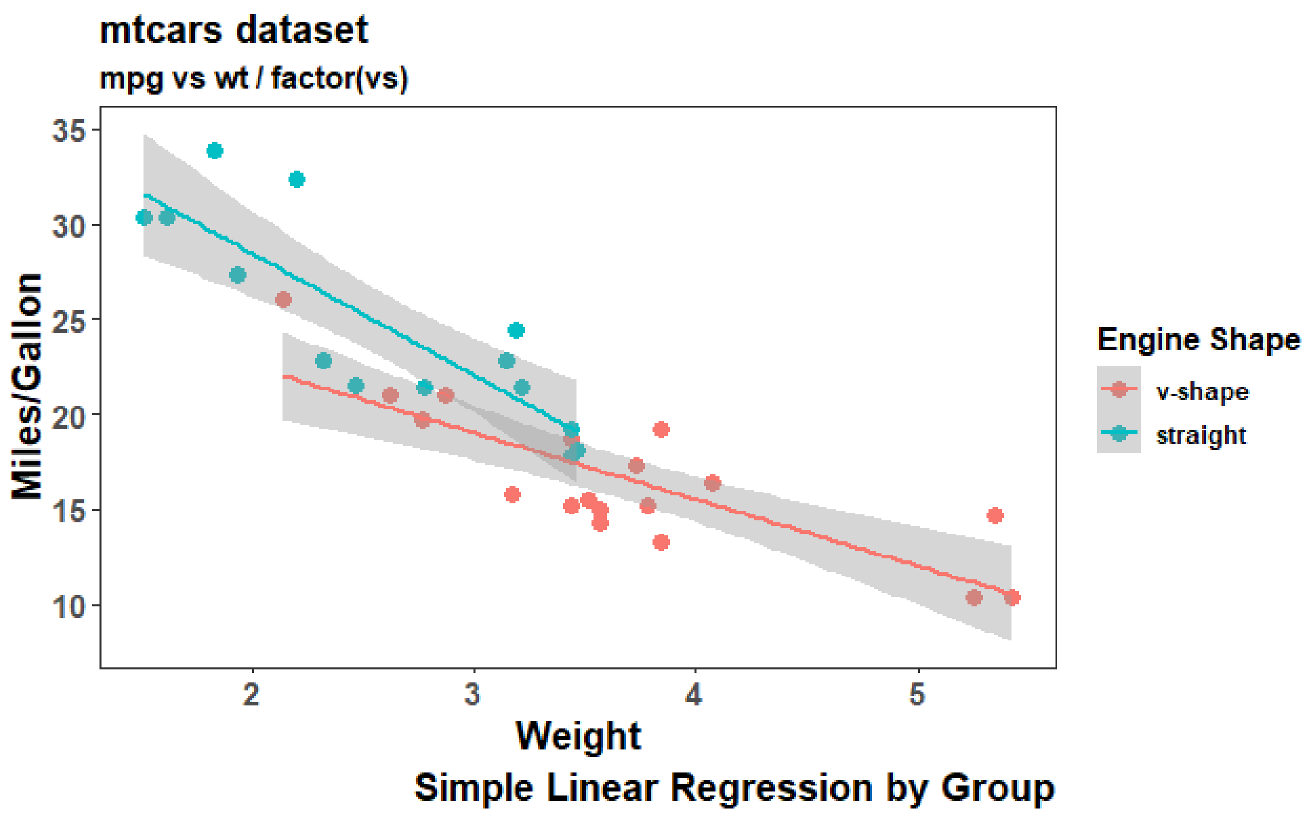
- Combine the two previous analysis with Continuous and Categorical predictor variable.
- Grouped scatterplot
- Grouped lm



Johnson&Johnson

Multiple Predictor Variables in a Linear Model

- v-shaped engines are primarily used in heavy cars => is the difference we saw in mpg between the engine shapes actually driven by vehicle weight?
- mpg is systematically lower for v-shaped cars => maybe there still is an effect after accounting for engine weight?
- Are slopes of the two lines different? => the relationship between weight and mpg contingent on engine shape?



Multiple Predictor Variables in a Linear Model

- v-shaped engines are primarily used in heavy cars => is the difference we saw in mpg between the engine shapes actually driven by vehicle weight?
- mpg is systematically lower for v-shaped cars => maybe there still is an effect after accounting for engine weight?

```
> lm.mod3=lm(mpg~ wt + factor(vs), data=mtcars)
> anova(lm.mod3)
Analysis of Variance Table

Response: mpg
           Df Sum Sq Mean Sq  F value    Pr(>F)    
wt          1 847.73  847.73 109.7042 2.284e-11 ***
factor(vs)  1  54.23   54.23   7.0177   0.01293 *  
Residuals  29 224.09    7.73                       
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

most of the variation in mpg is explained by vehicle weight, but that there is also additional effect of engine shape

ANCOVA

- The relationship between weight and mpg contingent on engine shape?
  - Need to identify "interaction term" for weight and engine shape

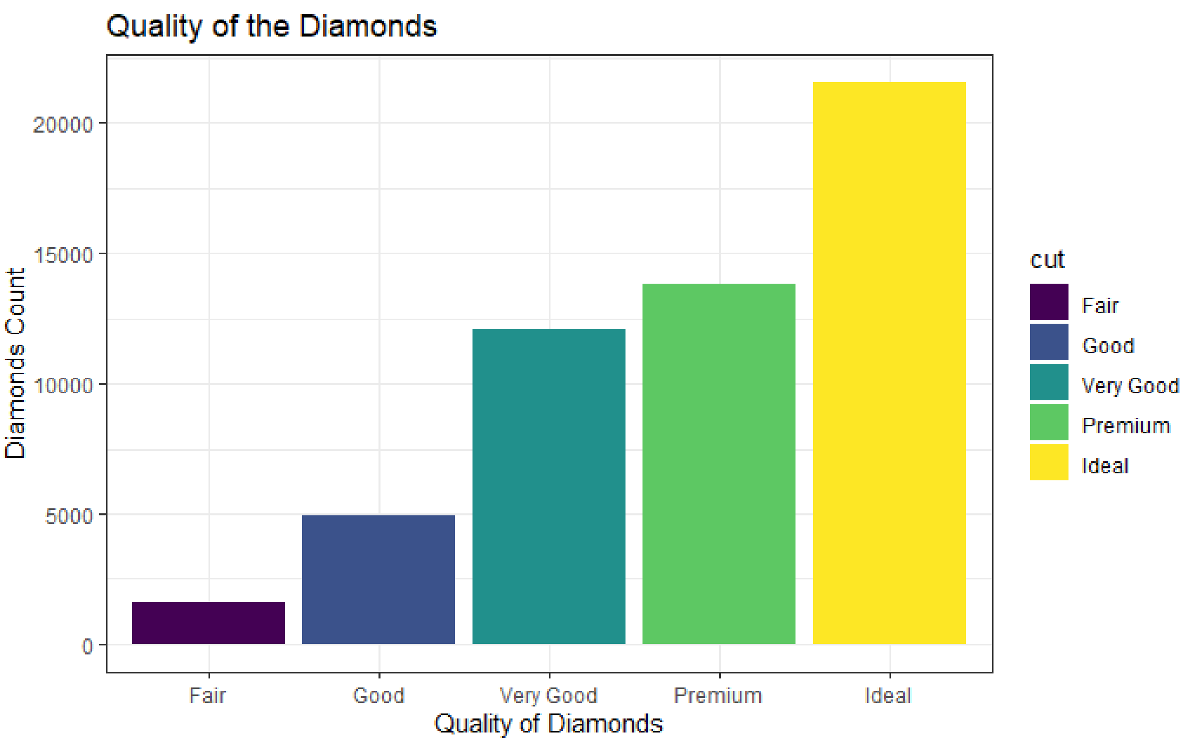
```
> lm.mod4=lm(mpg~ wt * factor(vs), data=mtcars)
> anova(lm.mod4)
Analysis of Variance Table

Response: mpg
               Df  Sum Sq Mean Sq  F value    Pr(>F)    
wt              1  847.73  847.73 127.5924 6.121e-12 ***
factor(vs)      1   54.23   54.23   8.1619  0.007978 ** 
wt:factor(vs)   1   38.06   38.06   5.7287  0.023634 *  
Residuals      28  186.03    6.64                       
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

there is a significant interaction between vehicle weight and engine shape on their effect on fuel efficiency

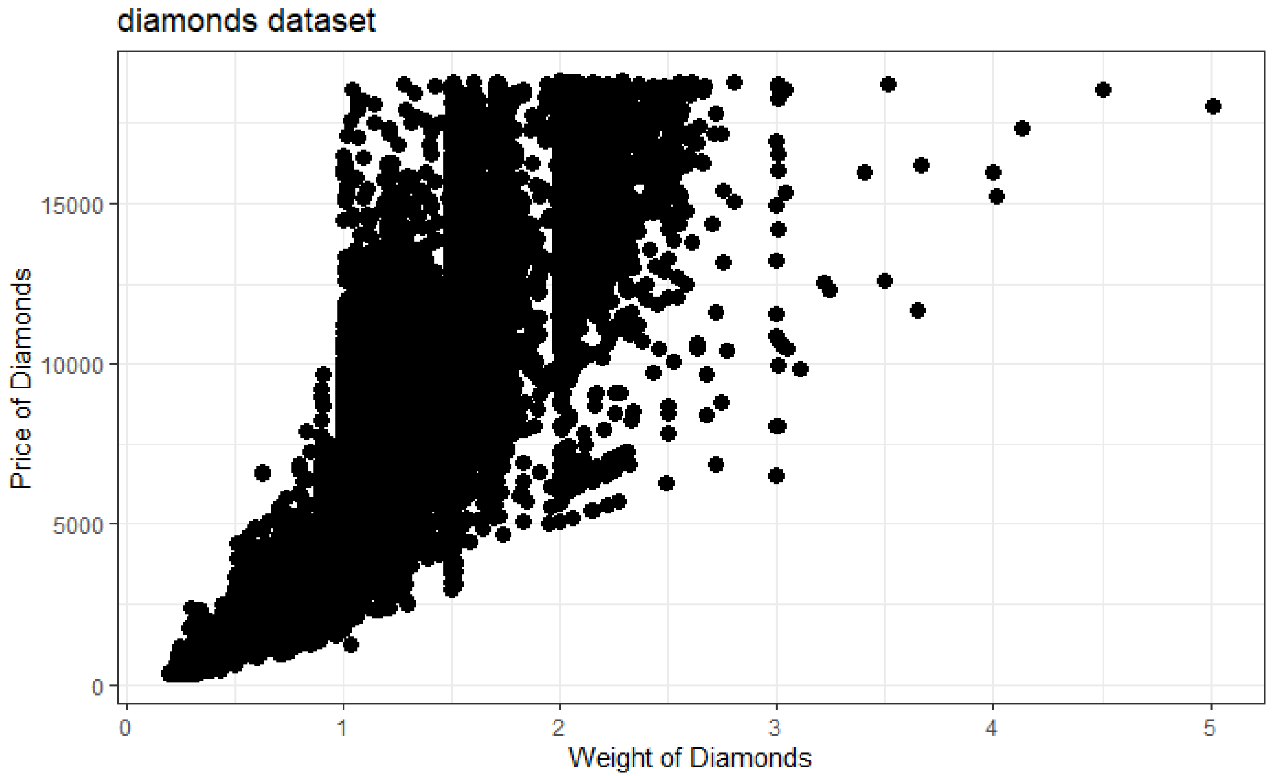
DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson saMRC advancinglife

diamonds Dataset



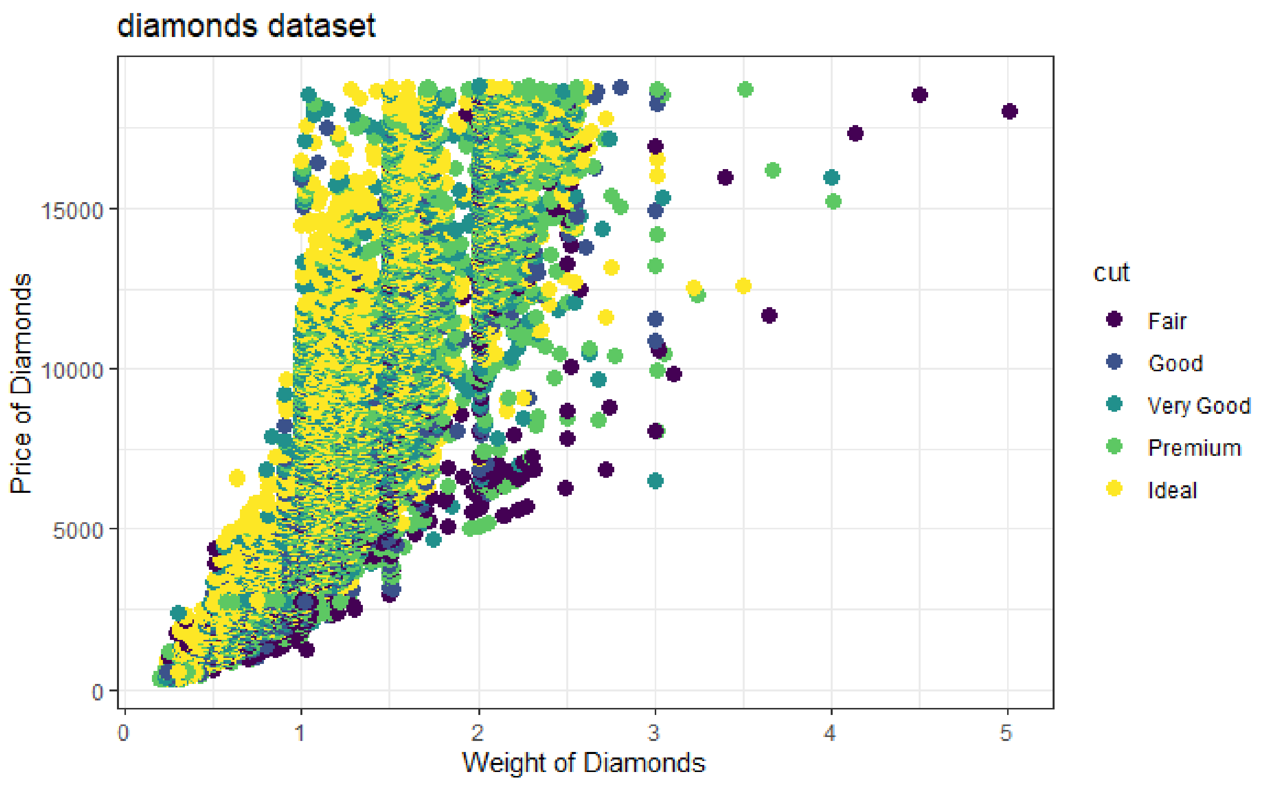
price	price in US dollars
carat	weight of the diamond
cut	quality of the cut
color	diamond color
clarity	measurement of how clear the diamond is
x	length in mm
y	width in mm
z	depth in mm
depth	total depth percentage
table	width of top of diamond relative to widest point

diamonds Dataset



price	price in US dollars
carat	weight of the diamond
cut	quality of the cut
color	diamond color
clarity	measurement of how clear the diamond is
x	length in mm
y	width in mm
z	depth in mm
depth	total depth percentage
table	width of top of diamond relative to widest point

diamonds Dataset



price	price in US dollars
carat	weight of the diamond
cut	quality of the cut
color	diamond color
clarity	measurement of how clear the diamond is
x	length in mm
y	width in mm
z	depth in mm
depth	total depth percentage
table	width of top of diamond relative to widest point

DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson SAMRC advancinglife

GLM with Binary Response

- Question: predict when a diamond will cost above a certain threshold price (let's say $10,000)
- Does the probability that a diamond is worth > $10,000 is dependent on its "carat", or unit weight?
  - Logistic regression

GLM with Binary Response

- glm.fit=glm(expensive~carat, data=diamonds)
- summary(glm.fit)

```
> glm.fit=glm(expensive~carat, data=diamonds)
> summary(glm.fit)

Call:
glm(formula = expensive ~ carat, data = diamonds)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.03657  -0.17896   0.00801   0.09337   0.82104

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.227495   0.001891  -120.3   <2e-16 ***
carat        0.406453   0.002038   199.4   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.05033729)

    Null deviance: 4717.3  on 53939  degrees of freedom
Residual deviance: 2715.1  on 53938  degrees of freedom
AIC: -8148.1

Number of Fisher Scoring iterations: 2
```

For an F-test:

```
> anova(glm.fit, test="F")
Analysis of Deviance Table

Model: gaussian, link: identity

Response: expensive

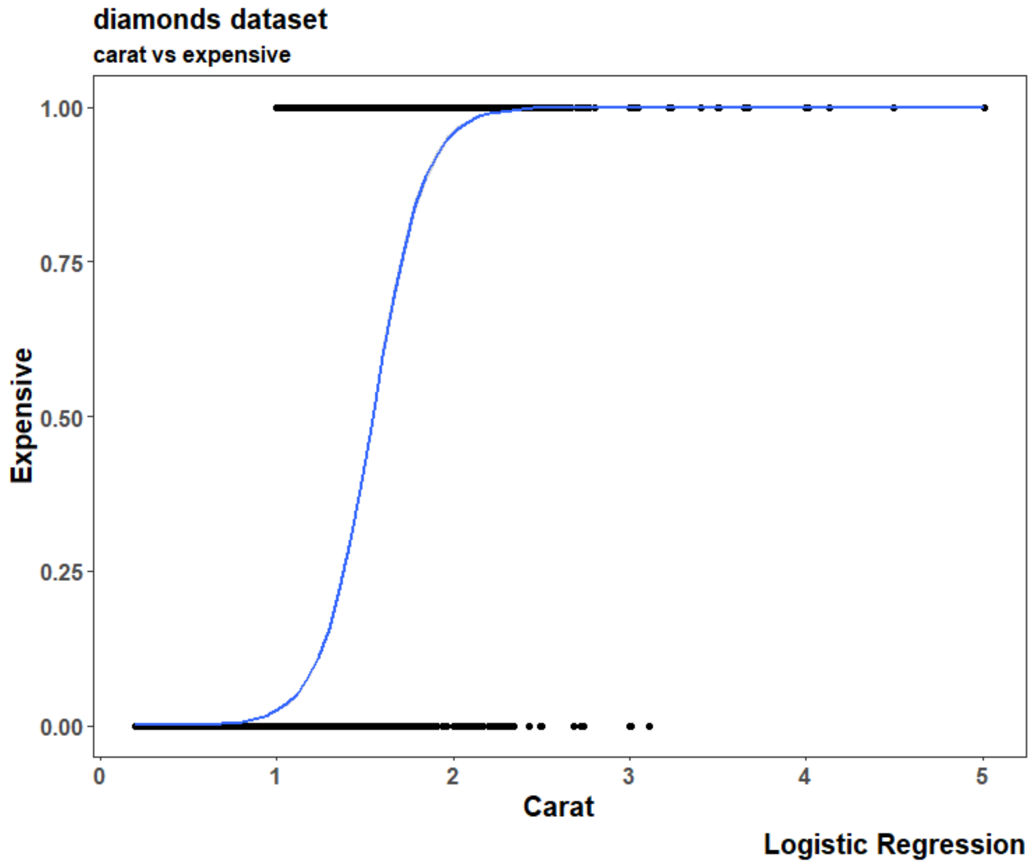
Terms added sequentially (first to last)


      Df Deviance Resid. Df Resid. Dev     F    Pr(>F)
NULL                  53939     4717.3
carat  1   2002.2     53938     2715.1 39775 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Visualization of the Result

Specify Binomial Family

```
ggplot(diamonds, aes(x=carat, y=expensive))+
  geom_point() +
  geom_smooth(method="glm", method.args = list(family = "binomial")) +
      labs(title="diamonds dataset", subtitle="carat vs expensive",
           y="Expensive", x="Carat", caption="Logistic Regression") +
  theme_bw()+
  theme(axis.title = element_text(size = title_fontsize, face = "bold"),
        axis.text.y = element_text(size = axisText_fontsize, face = "bold"),
        axis.text.x = element_text(size = axisText_fontsize, face = "bold"),
        panel.grid = element_blank(),
        plot.title = element_text(size = title_fontsize, face = "bold"),
        plot.subtitle = element_text(size = subtitle_fontsize, face = "bold"),
        plot.caption = element_text(size = title_fontsize, face = "bold"))
```



GLM with Count and Proportional Data

- snail dataset from MASS package: includes the results of an experiment in which snails were held for 1,2, 3, or 4 weeks under controlled temperature and relative humidity.
- “Deaths” can be converted to proportions using “N”.

```
> tibble(snails)
# A tibble: 96 x 6
   Species Exposure Rel.Hum  Temp Deaths     N
   <fct>      <int>   <dbl> <int>  <int> <int>
 1 A              1    60      10      0    20
 2 A              1    60      15      0    20
 3 A              1    60      20      0    20
 4 A              1    65.8    10      0    20
 5 A              1    65.8    15      0    20
 6 A              1    65.8    20      0    20
 7 A              1    70.5    10      0    20
 8 A              1    70.5    15      0    20
 9 A              1    70.5    20      0    20
10 A              1    75.8    10      0    20
# i 86 more rows
# i Use `print(n = ...)` to see more rows
```

GLM with Count Data - Model

- Poisson regression using the count of deaths as the response variable.
- Expect the effect to be less apparent if the snails are kept in the experiment for a short amount of time.
  - restrict the analysis to snails held for 3 or 4 weeks

- count.fit=glm(Deaths~Temp, data=snails, subset=which(Exposure>=3), family="poisson“)

GLM with Count Data - Results

```
> summary(count.fit)

Call:
glm(formula = Deaths ~ Temp, family = "poisson", data = snails,
    subset = which(Exposure >= 3))

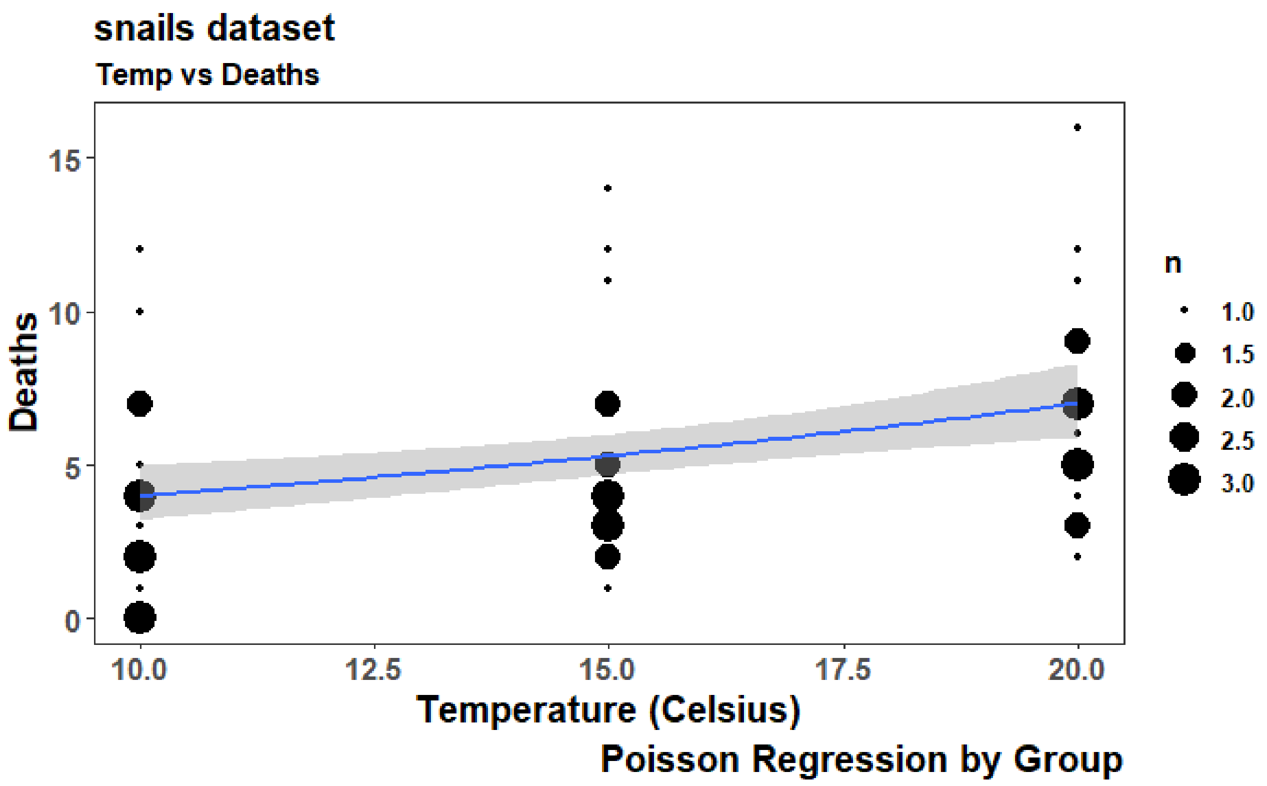
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8309  -1.1110  -0.4584   0.7083   3.2154

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.82914    0.25375   3.268  0.00108 **
Temp         0.05589    0.01546   3.615  0.00030 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 130.86  on 47  degrees of freedom
Residual deviance: 117.53  on 46  degrees of freedom
AIC: 276.11

Number of Fisher Scoring iterations: 5
```



GLM with Proportional Data - Model

- Model proportion of deaths instead of the number of deaths.

- prop.fit=glm(cbind(Deaths, N)~Temp, data=snails, subset=which(Exposure>=3), family=binomial(link="logit"))
- summary(prop.fit)

```
> summary(prop.fit)

Call:
glm(formula = cbind(Deaths, N) ~ Temp, family = binomial(link = "logit"),
    data = snails, subset = which(Exposure >= 3))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7016  -1.0290  -0.4111   0.6136   2.8184

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.16897    0.28286  -7.668 1.75e-14 ***
Temp         0.05604    0.01742   3.216   0.0013 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 104.606  on 47  degrees of freedom
Residual deviance:  94.093  on 46  degrees of freedom
AIC: 241.71

Number of Fisher Scoring iterations: 4
```

Overview

1. Introduction & Motivation
2. R Software: Basic Introduction
3. R Studio Interface
4. Installing and Loading R Packages
5. Projects in R and developing R packages
6. Case Study and analysis with R
7. **RMarkdown**
8. Advanced RMarkdown
9. Dashboard for Covid-19 using RMarkdown

Markdown: What?

- Markdown allows you to write a file format independent document using an easy-to-read and easy-to-write plain text format.

- Instead of marking up text so that is easy for a computer to read
  - e.g. HTML: <html><body>Name</body></html>

- The goal is to mark down text so that it is easy and human readable (instead of machine readable):
  - e.g. **Name**

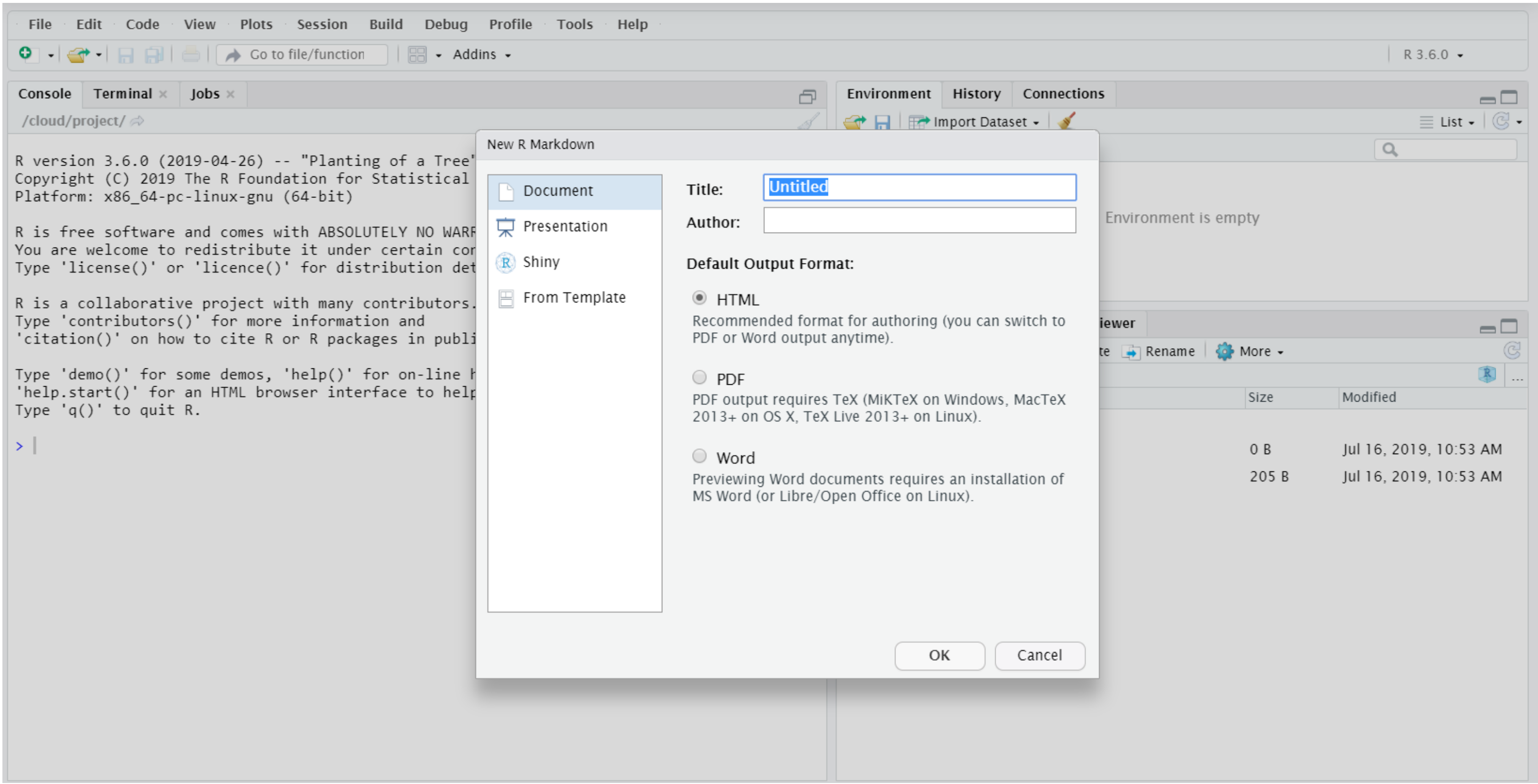
- Markdown is a specific Markup language which is structured very loosely => any file format can be generated using pandoc

- From one Markdown document you can generate different file formats: html, PDF, docx, slideshows, rtf, etc.
  - The downside is that there is slightly less control over formatting.

RMarkdown: What? Why?

- Extension of Markdwon via R:
  - Allowing *R* code and its results to be merged with Markdown
  - Ensuring that RMarkdown documents are fully reproducible
  - Enabling extra modifications to original markdown specification
- Provides an unified authoring framework for data science, combining your code, its results, etc.
  - e.g., just by changing the dataset then entire analysis can be rerun and the new report can be produced.
- Integrates a number of R packages and external tools
- RMarkdown Cheat Sheet: *Help > Cheatsheets > R Markdown Cheat Sheet (*https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf*)*
- RMarkdown Reference Guide: *Help > Cheatsheets > R Markdown Reference Guide*
- Both cheatsheets are also available at http://rstudio.com/cheatsheets
- *Help > Markdown Quick Reference*

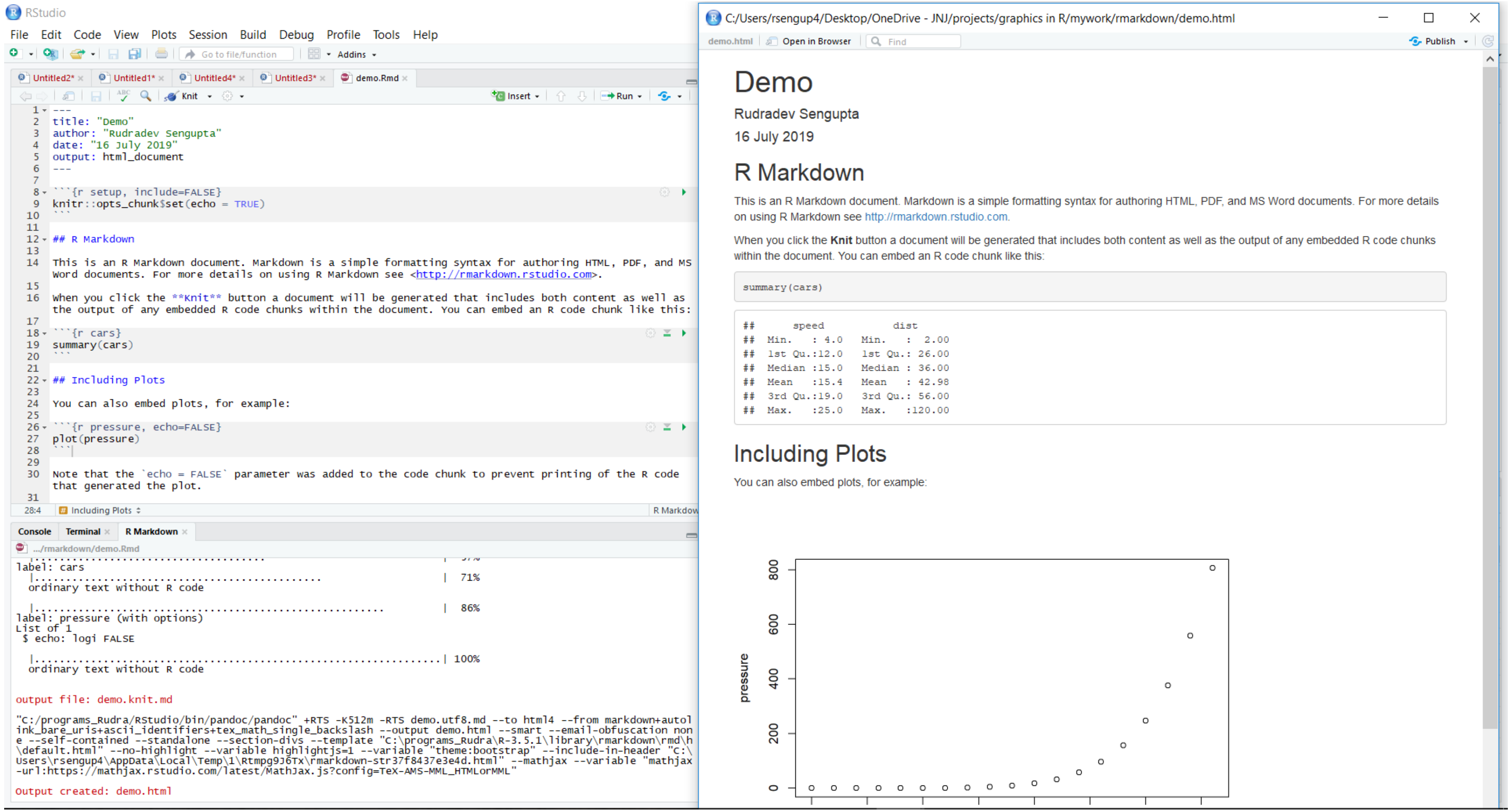
RMarkdown in RStudio Cloud



- File -> New File -> R Markdown...

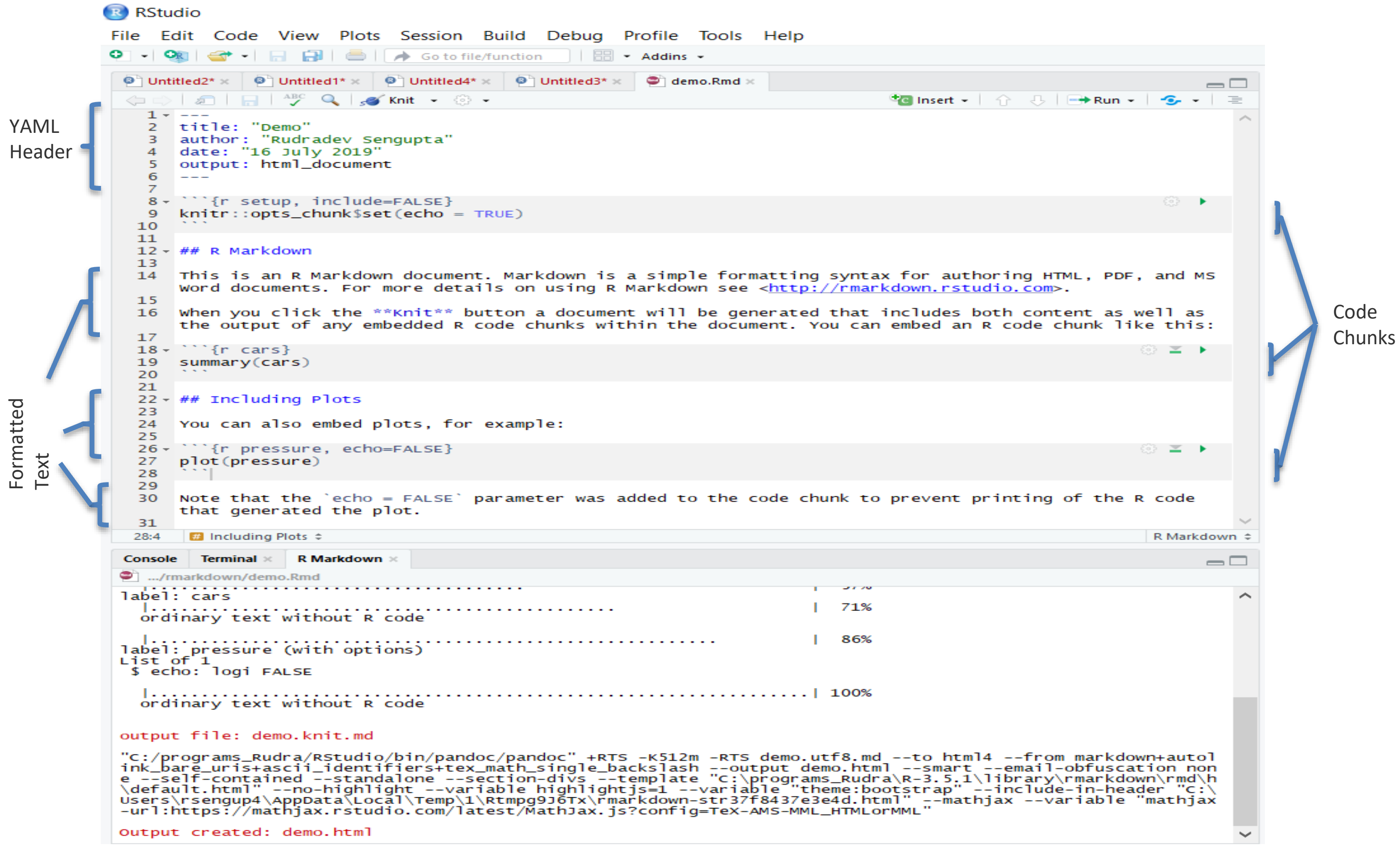
DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson saMRC advancinglife

RMarkdown: First Look



DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson SAMRC advancing life

RMarkdown: Components



- There are principally three sections to an RMarkdown document:
- YAML header surrounded by ---
- Code chunks surrounded by ```
- Free text mixed with simple text formatting like #heading and _italics_
- Also possible to include inline code to make it more dynamic:
  - `r pressure$temperature[5] * pressure$pressure[5]`

DSI DATA SCIENCE INSTITUTE UHASSELT Johnson&Johnson saMRC advancing life

RMarkdown: Workflow

- **“knit”**ing the .Rmd document:
  - — RMarkdown sends the .Rmd file to knitr (http://yihui.name/knitr/)
  - — knitr executes all of the code chunks and creates a new markdown (.md) document which includes the code and its output
  - — .md file is then processed by **pandoc** (http://pandoc.org/) which is responsible for creating the final file

- The advantage of this two step workflow is that you can create a very wide range of output formats.

Running RMarkdown

- A notebook interface where code and output are interleaved

- run each code chunk (RStudio executes the code and displays the results inline with the code):
  - by clicking the Run icon OR,
  - by pressing Cmd/Ctrl + Shift + Enter

- To produce a complete report:
  - click “Knit” or press Cmd/Ctrl + Shift + K
  - rmarkdown::render(“***.Rmd”) – can specify additional options here

YAML Header

- Can control different settings for the "whole document" by tweaking the parameters of the YAML header

- YAML : "yet another markup language" => designed to represent hierarchical data in easy way to read and write
  - RMarkdown uses it to control many details of the fine output, e.g., document parameters, bibliographies, etc.

  - Easy to create several similar reports using different values of the parameters and then combine them together to create the final report

YAML Header

```
---
output: html_document
params:
  my_class: "suv"
---

```{r setup, include = FALSE}
library(ggplot2)
library(dplyr)

class <- mpg %>% filter(class == params$my_class)
```

Fuel economy for `r params$my_class`s

```{r, message = FALSE}
ggplot(class, aes(displ, hwy)) +
  geom_point() +
  geom_smooth(se = FALSE)
```
```

- RMarkdown documents can include one or more parameters whose values can be set when you render the report.
- Useful when you want to re-render the same report with distinct values for various key inputs.
  - Doing similar exploratory analysis for different compounds or subject specific analysis for different patients.
  - Declaration: use the params field.
  - parameters are available within the code chunks as a read-only list named params

my_class parameter determines which class of cars to display

- Possible to write atomic vectors directly into the YAML header.
- Possible to run R expressions by prefacing the parameter value with !r.

```
params:
  start: !r lubridate::ymd("2015-01-01")
  snapshot: !r lubridate::ymd_hms("2015-01-01 12:30:00")
```

- Click the “Knit with Parameters” option in the Knit dropdown menu

Code Chunks

- To run code inside an R Markdown document, you need to insert a chunk. There are three ways to do so:
  - — The keyboard shortcut Cmd/Ctrl + Alt + I
  - — The “Insert” button icon in the editor toolbar
  - — By manually typing the chunk delimiters ```{r} and ```
- Function-like features.
- A chunk should be relatively self-contained, and focussed around a single task.
- Chunk Name: optional, but helpful – easy to nagivate or call chunks at different parts of the script.
- Chunk options: which types of output each option supressess:

Option	Run code	Show code	Output	Plots	Messages	Warnings
`eval = FALSE`	-		-	-	-	-
`include = FALSE`		-	-	-	-	-
`echo = FALSE`		-				
`results = "hide"`			-			
`fig.show = "hide"`				-		
`message = FALSE`					-	
`warning = FALSE`						-

- .Rmd prints tables by default in the console if you specify a dataframe
- Options to do Caching:
  - — Normally, each knit of a document starts from a completely clean slate.
  - — cache=TRUE will save the output of the chunk to a specially named file on disk. On subsequent runs, knitr will check to see if the code has changed, and if it hasn’t, it will reuse the cached results.

Text Formatting

- .Rmd files is written in Markdown, a lightweight set of conventions for formatting plain text files.

- Markdown is designed to be easy to read and easy to write. It is also very easy to learn.

- R Markdown uses Pandoc's Markdown, a slightly extended version of Markdown.

- The best way to learn these is simply to try them out.

- Refer to the Cheatsheets or Markdown Quick Reference.

Overall (short) Discussion

- Modeling and reporting.
- Publicly available data sources.
- For short-term prediction: many methods are available.

Contact Us

- Rudradev Sengupta: rsengup4@its.jnj.com
- Ziv Shkedy: ziv.shkedy@uhasselt.be