DATA SOCIETY®

"If you can't explain it simply, you don't understand it well enough."

- Albert Einstein

Course syllabus

1. What is data science?
2. Manipulating data in R
3. Visualization in R

Setting expectations

Data science takes dedication! You will need to:

1. Take this course ☺
2. Practice
3. Review class material on your own
4. Practice
5. Complete exercises outside of class
6. Practice
7. Share and read latest news

Outline

- What is data science?
- A data scientist's approach
- Introduction to R
  - Calculations in R
  - Reading data into R
  - Manipulating data in R
- Visualization in R
  - Basic plotting

How is data being used?

Retail	Finance	Marketing	Real estate	Cool
Target:	Kabbage:	Netflix:	Zillow:	Andrew Ng:
The store knows you're pregnant based on what you buy	Makes lending decisions based on Amazon product reviews, etc.	What movie should you watch?	Calculates Zestimate (value of your home)	Machine learning techniques recognize cat faces online using pictures and videos
	amazon reviews	NETFLIX		

What is "Big Data"?

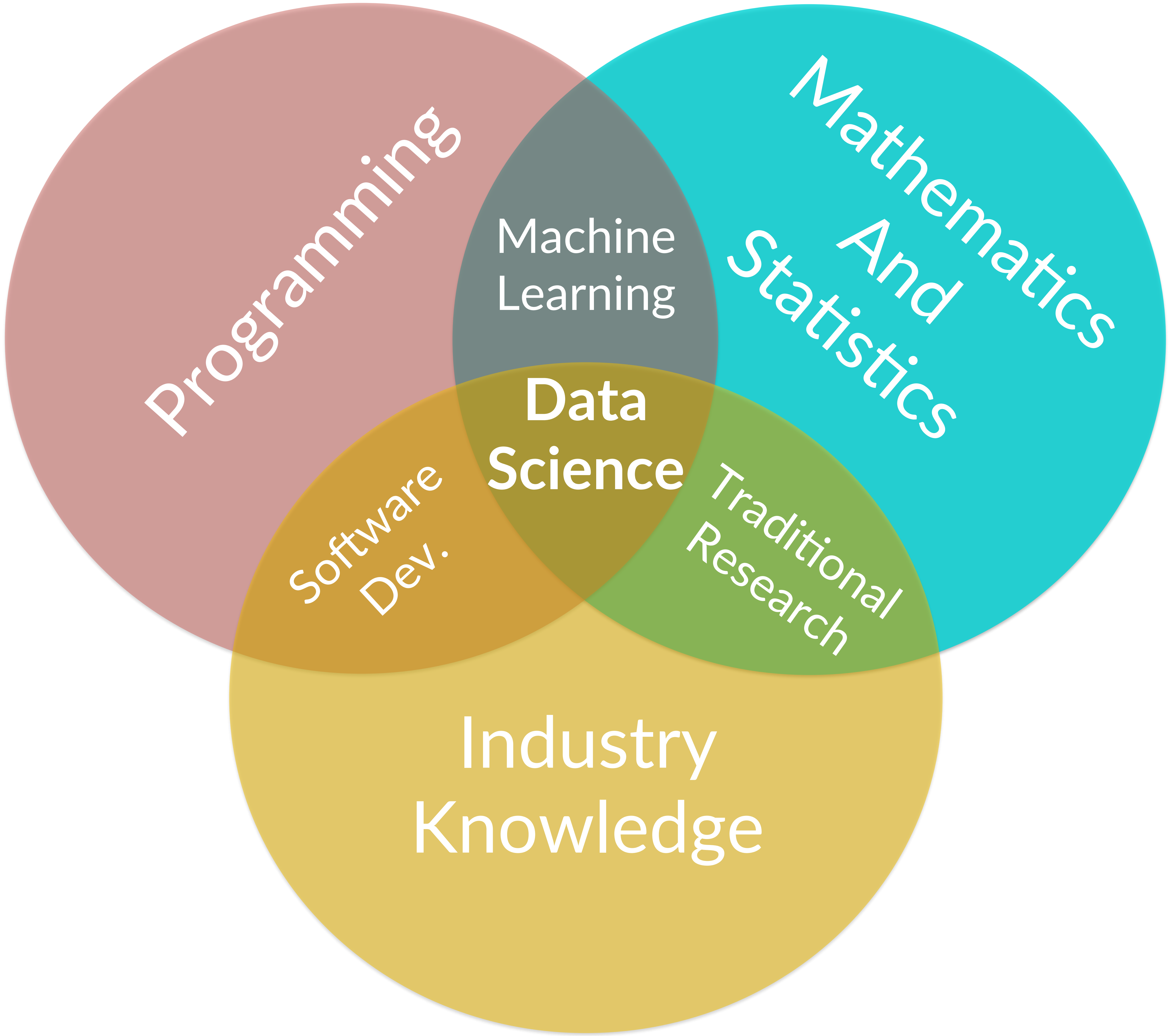
- Big Data is large volumes of information
  - Moving
  - Storing
  - Manipulating
  - Accessing

- It is not:
  - Analysis or insights

That's why you're in this class!

What is data science?

- Data science applies the scientific method to analyzing data
- It lies at the intersection of several disciplines
- It draws on domain specific knowledge that makes the analysis of Big Data possible



Who is a data scientist?

- An analyst who can:

1. Pose the right question

2. Wrangle the data (gather, clean, and sample data to get a suitable data set)

3. Manage the data for easy access by the organization

4. Explore the data to generate a hypothesis

5. Make predictions using statistical methods such as regression and classification

6. Communicate the results using visualizations, presentations, and products

Levels of expertise

Data analyst

- Wrangles the data
- Manages the data
- Creates basic analyses and visualizations

Data modeler

- Models to answer specific questions
- Understands the data, its source and structure

Data scientist

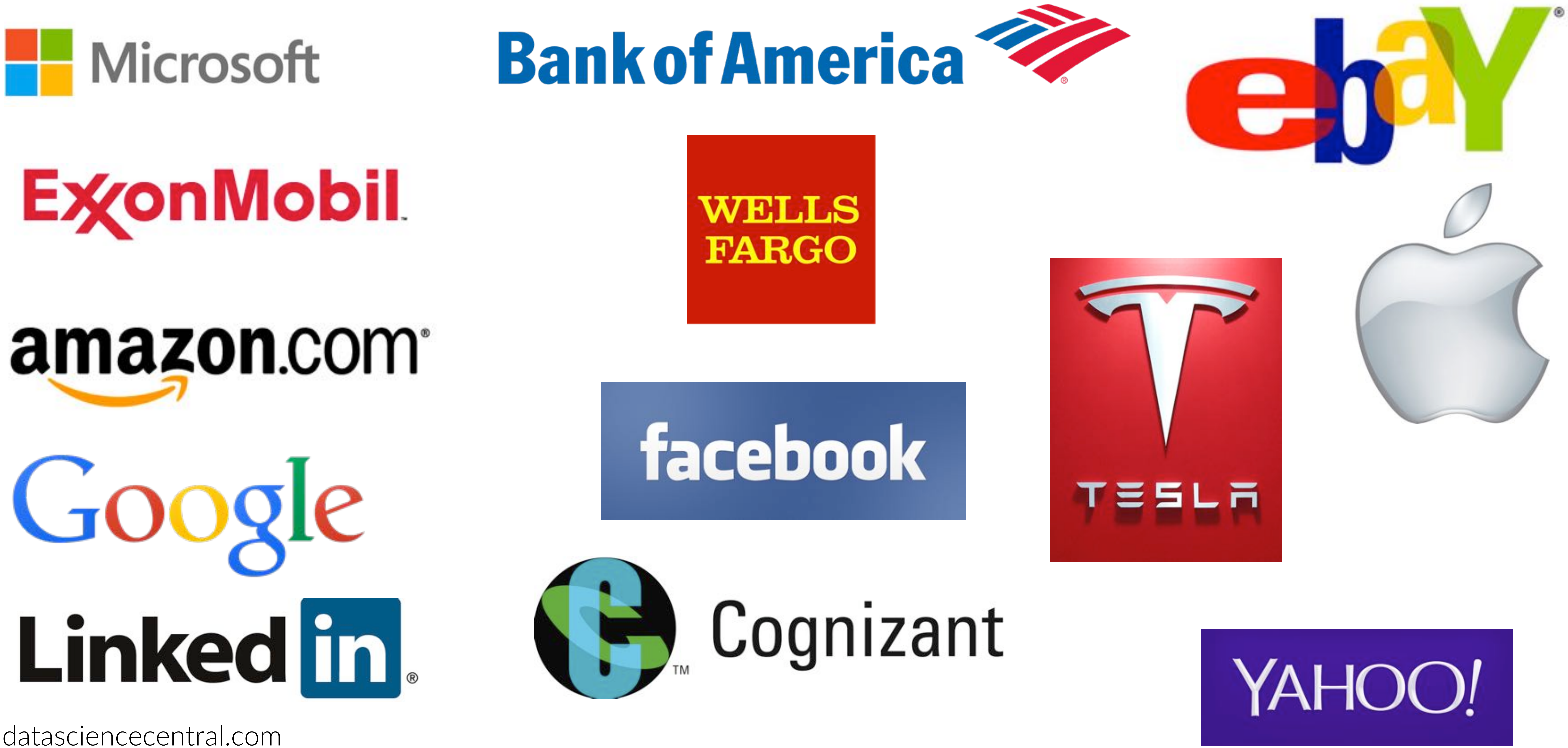
- Asks the right questions
- Looks for patterns in data
- Interprets results critically

Data science job market

Somewhat important ✓ Very important ✔

	A non-data-driven company	The business is just starting to collect data	Data is the product of the company	Company uses data to make decisions
Basic tools	✔	✔	✔	✔
Software engineering		✔	✓	✓
Statistics	✓	✓	✔	✔
Machine learning			✔	✔
Data processing		✔	✓	✔
Data visualization and communication	✔	✓	✓	✔
Thinking like a data scientist	✓	✓	✔	✔

Who hires data scientists?



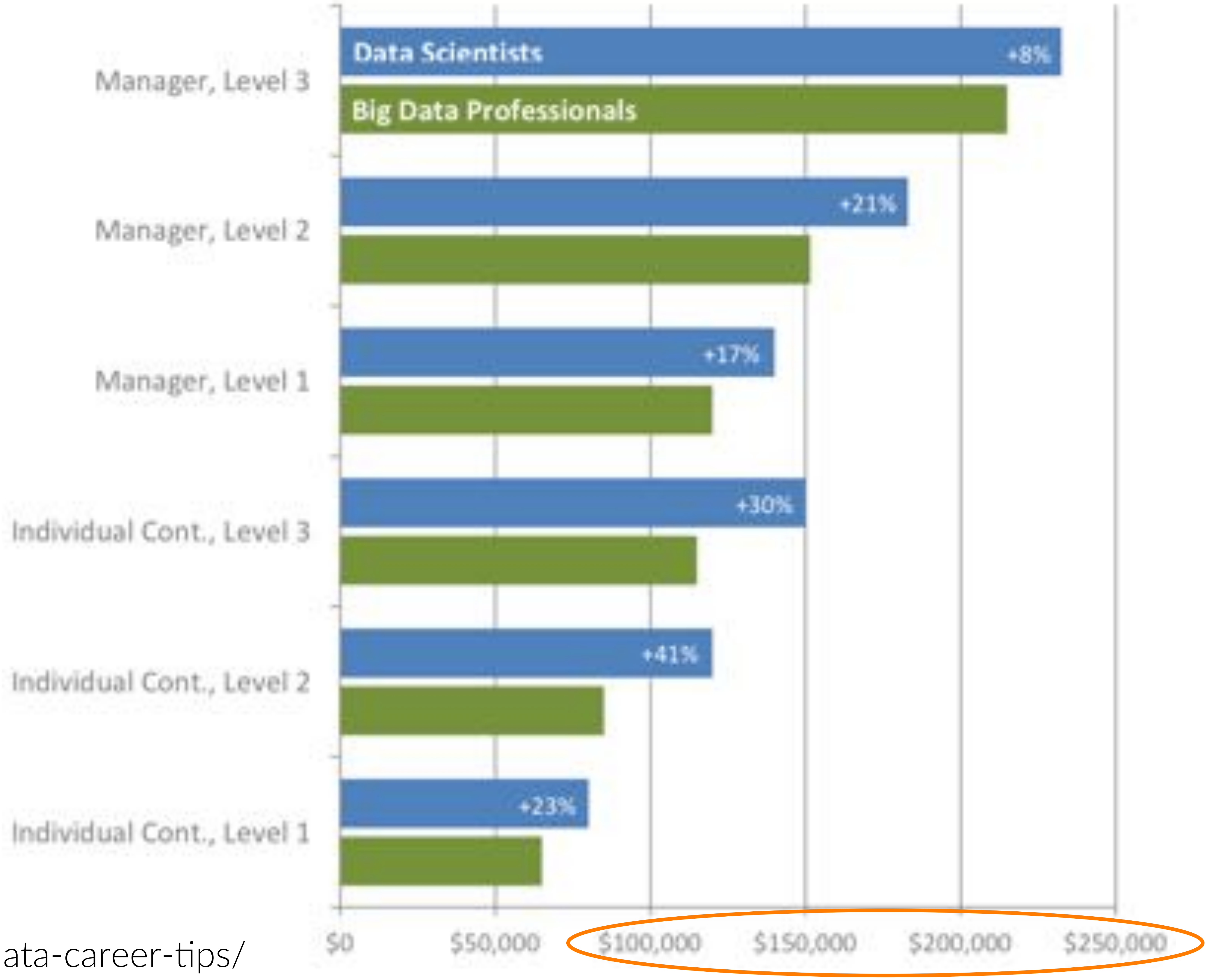
Source: datasciencecentral.com

How much do data scientists make?

- According to a Burtch Works 2014 data science job market survey:

“Data scientists earn a median salary that can be up to 40% higher than predictive analytics professionals at the same job level”

- The graphic on the right provides detail on median salaries by experience level

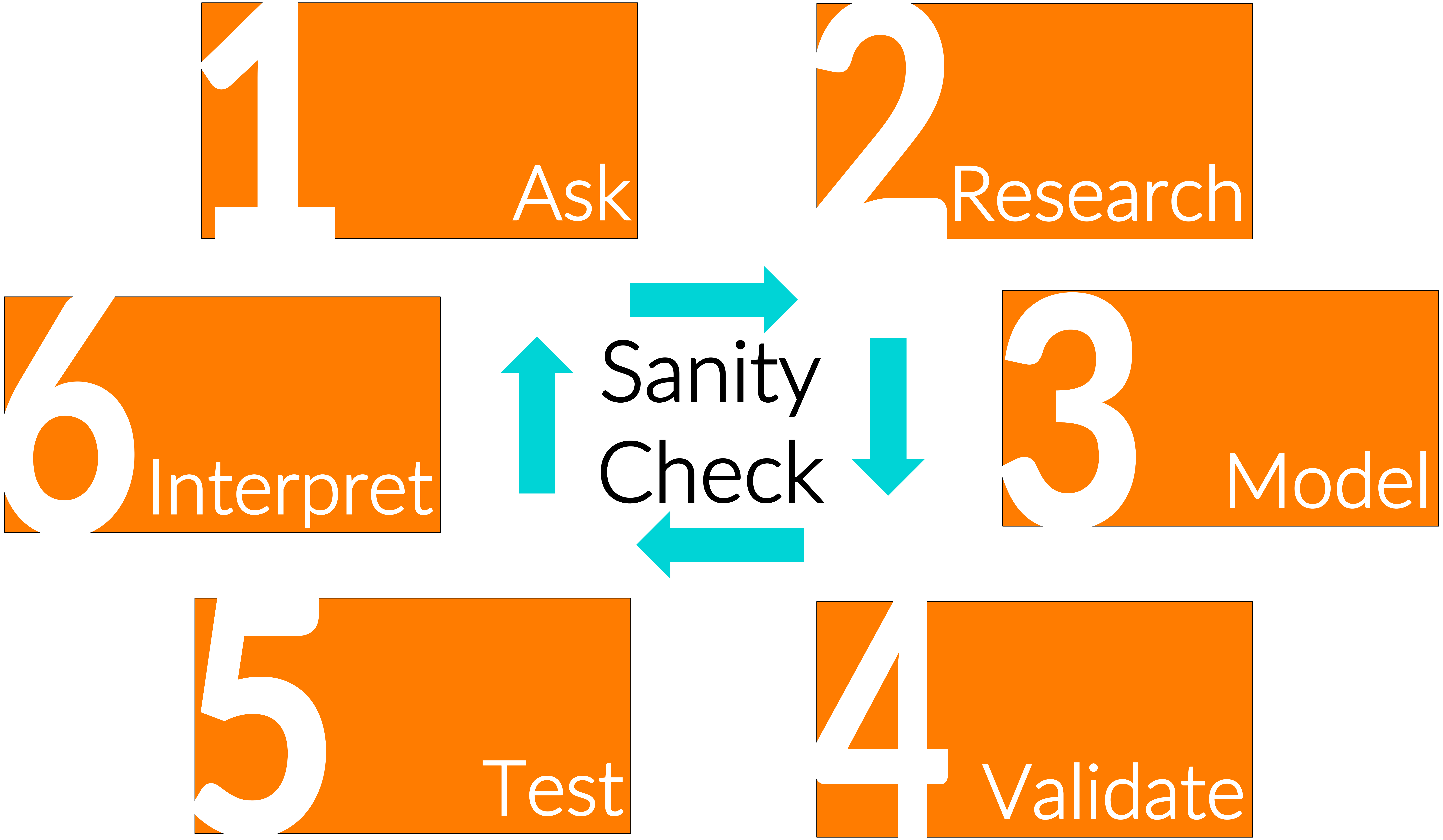


Source: http://www.burtchworks.com/big-data-analyst-salary/big-data-career-tips/

Outline

- What is data science?
- A data scientist's approach
- Introduction to R
  - Calculations in R
  - Reading data into R
  - Manipulating data in R
- Visualization in R
  - Basic plotting

Data science control cycle



1 Ask

2 Research

3 Model

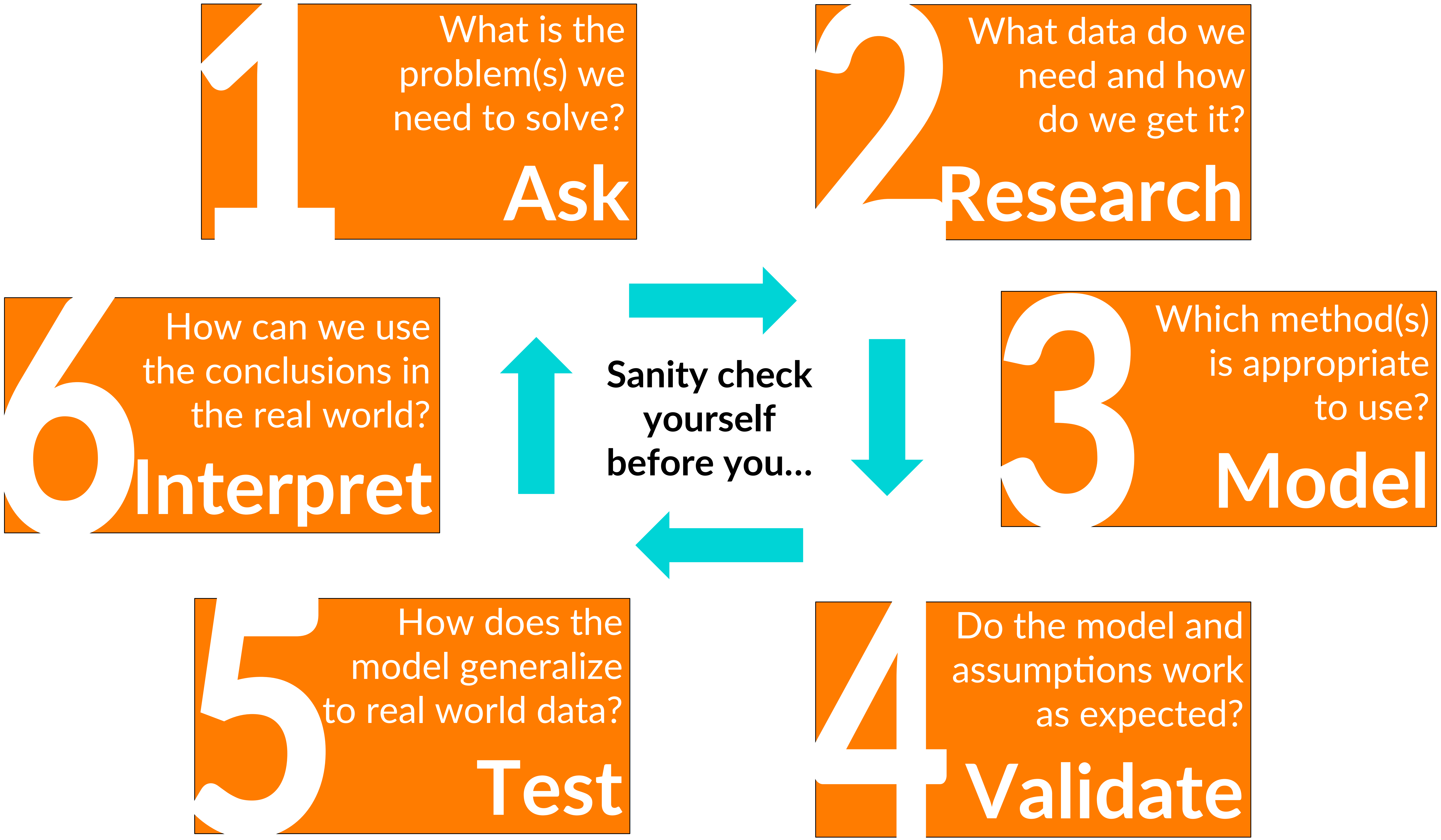
4 Validate

5 Test

6 Interpret

Sanity Check

Data science control cycle



1 What is the problem(s) we need to solve? **Ask**

2 What data do we need and how do we get it? **Research**

3 Which method(s) is appropriate to use? **Model**

4 Do the model and assumptions work as expected? **Validate**

5 How does the model generalize to real world data? **Test**

6 How can we use the conclusions in the real world? **Interpret**

Sanity check yourself before you…

For every job there is a tool

Data storage

- Hadoop
- Spark
- SQL
- …

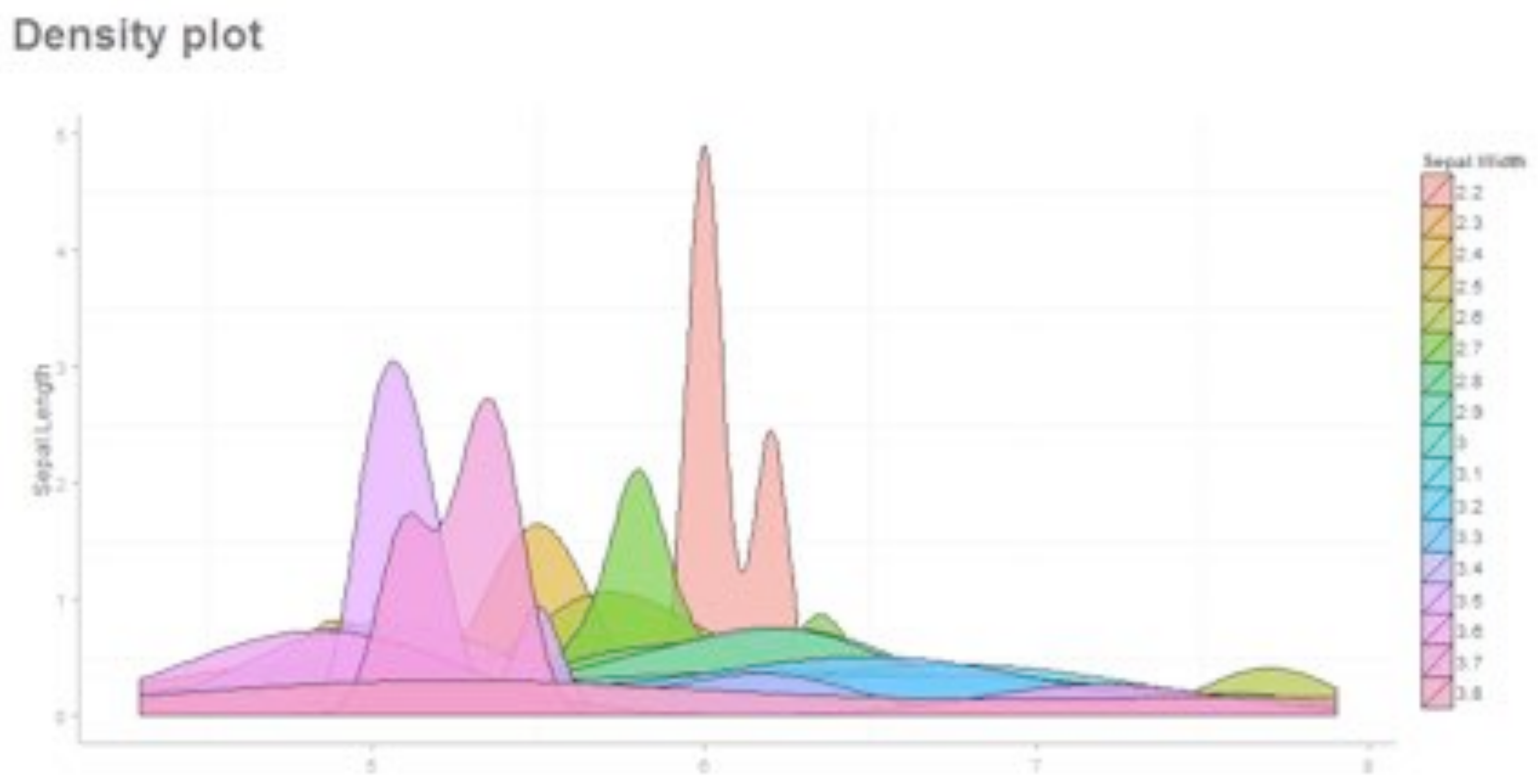
Data analysis

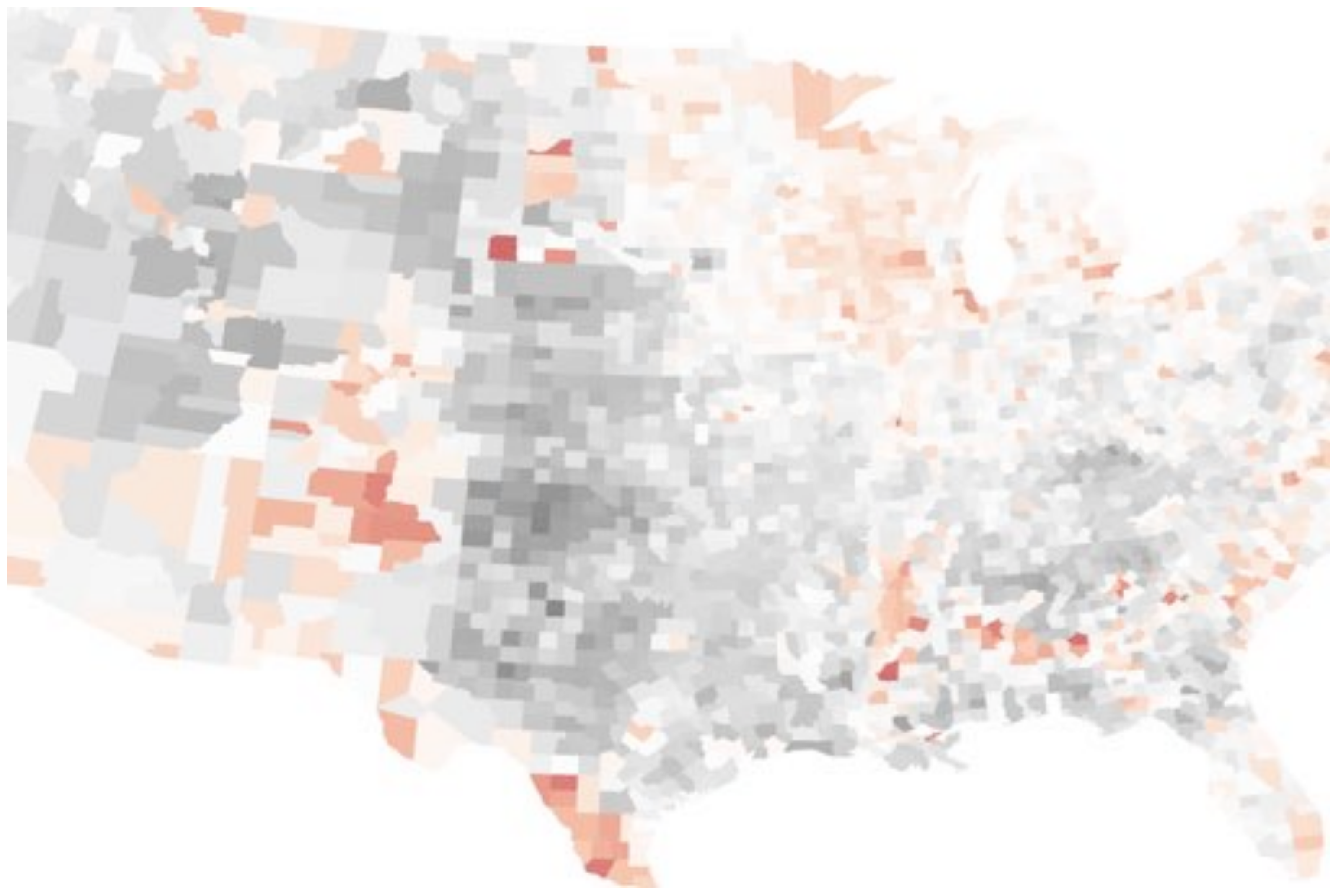
- R
- SAS
- SPSS
- Matlab
- Python
- Google Prediction API
- …

For every job there is a tool

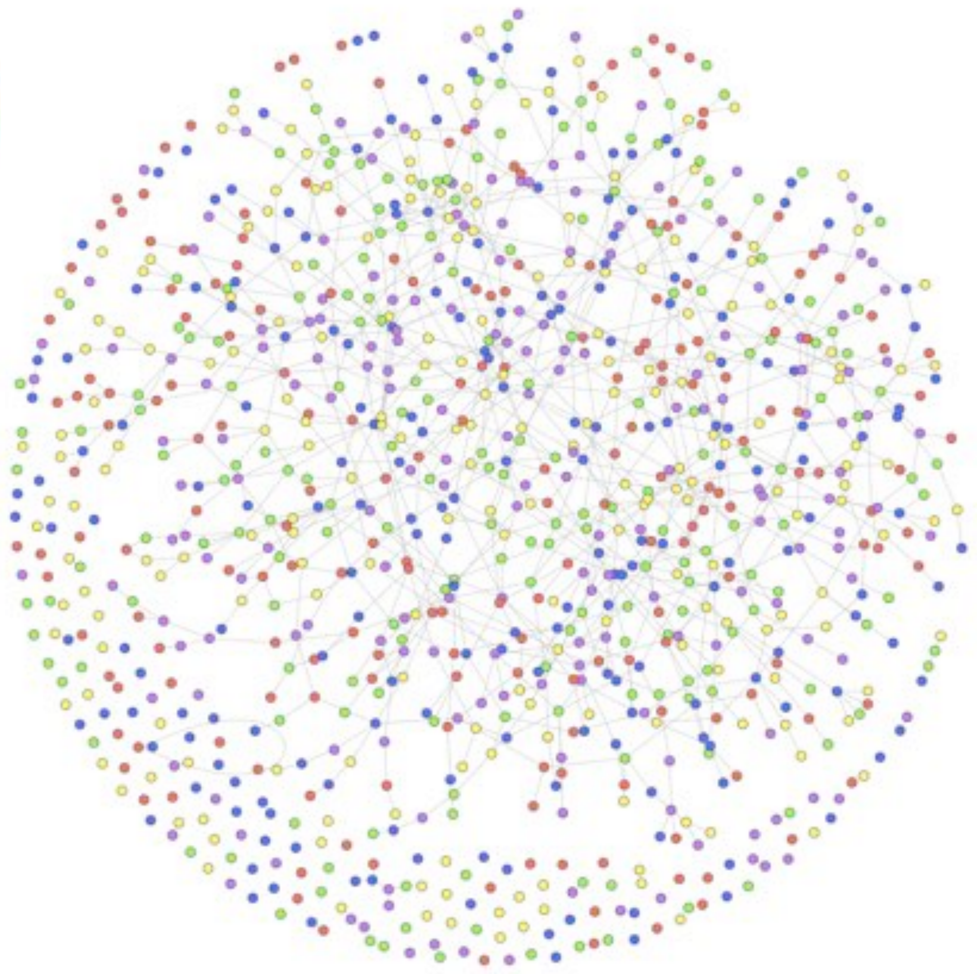
Visualization

- R
- Tableau
- iVEDiX





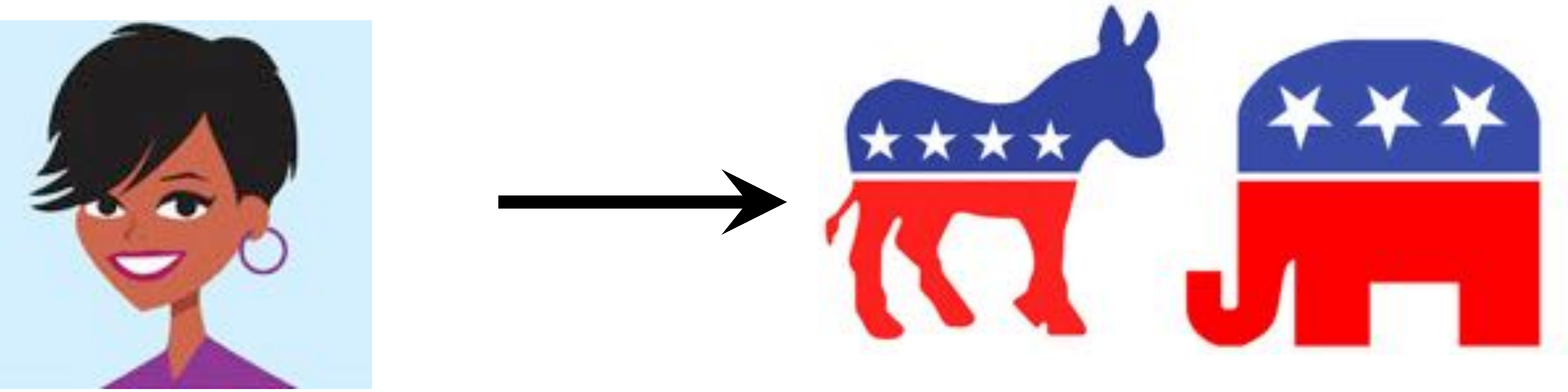
All of these visualizations were created in R



Supervised machine learning

Pattern discovery when inputs (x) and outputs (y) are known

Input x:
Voter



Output y:
Political affiliation

Examples: Classification and regression are supervised machine learning

Unsupervised machine learning

The data inputs *(x)* have no target outputs *(y)*

Input x:
Voter

→ **?**

Output y:
Not given
(to be discovered)

We want to impose structure on the inputs *(x)* to say something meaningful about the data

Machine vs. human

	Machine	Human
Understanding context		✓
Thinking through the problem		✓
Asking the right questions		✓
Selecting the right tools		✓
Performing calculations quickly	✓	
Performing repetitive tasks	✓	
Following pre-defined rules	✓	
Interpreting results		✓

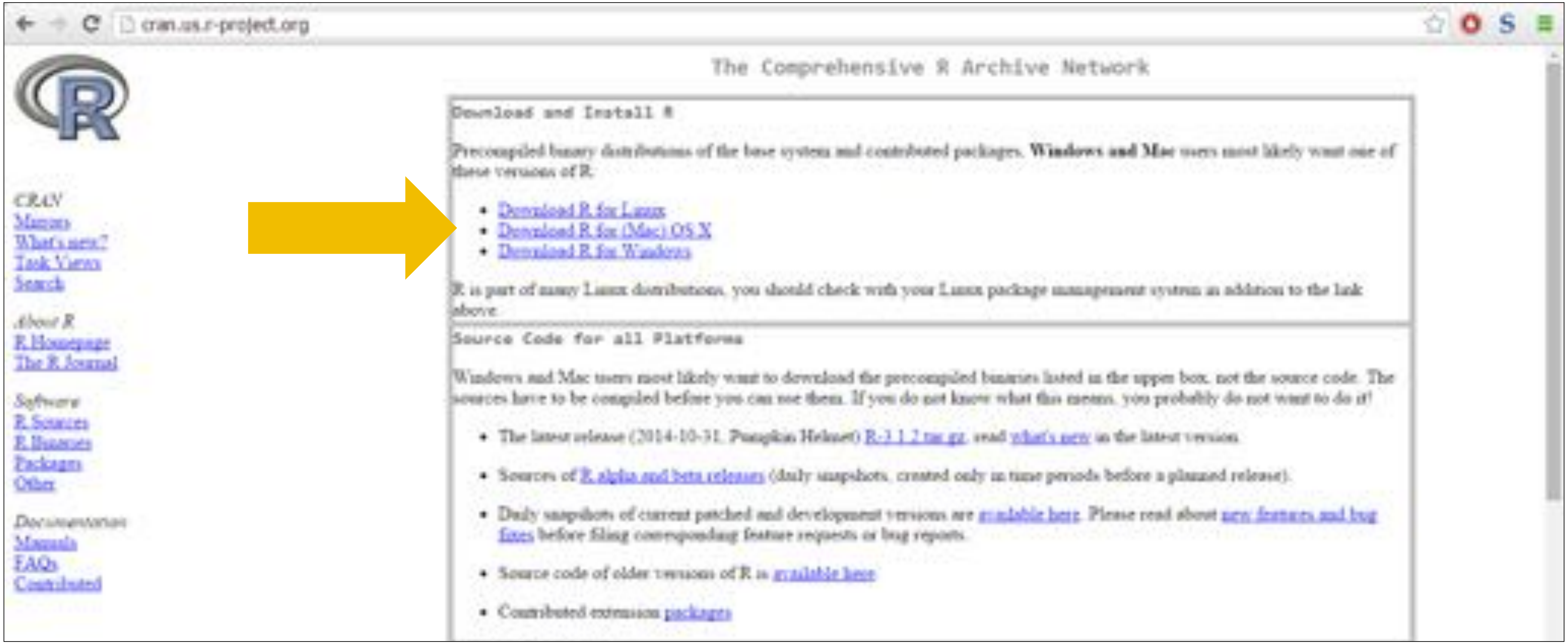
Setting up R: overview

1. What is R?
2. Download R from the CRAN website (http://cran.us.r-project.org/)
   - R for Windows
   - R for Mac
3. Install R Studio (http://www.rstudio.com/products/rstudio/download/)
   - RStudio a brief tour
4. Running a script
   - Variables
5. Reading in a data
   - Manually
   - Through the script

What is R?

- R is a statistical programming software
  - Has many similar features to SAS, Excel and SPSS
  - Scripting language
- It’s free and open source
- Has lots of helpful pre-built functions
  - You can build your models quicker
- Easy to learn

Install R



The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages. **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

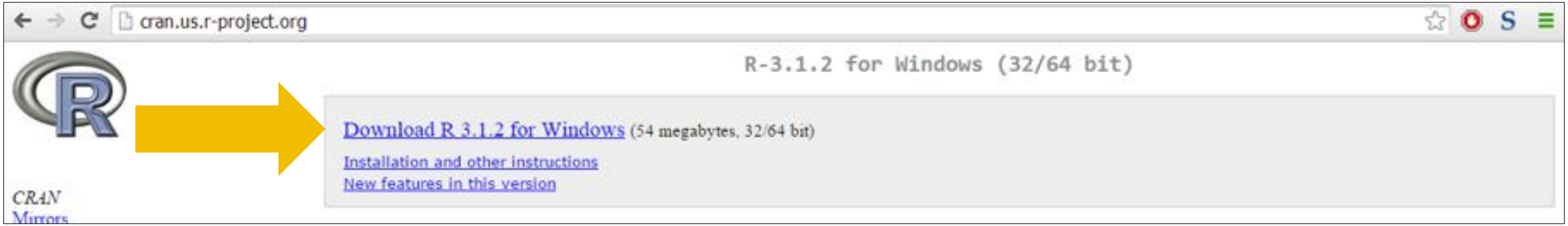
R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2014-10-31, Pumpkin Helmet) R-3.1.2.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

R for Windows



cran.us.r-project.org

R-3.1.2 for Windows (32/64 bit)

Download R 3.1.2 for Windows (54 megabytes, 32/64 bit)

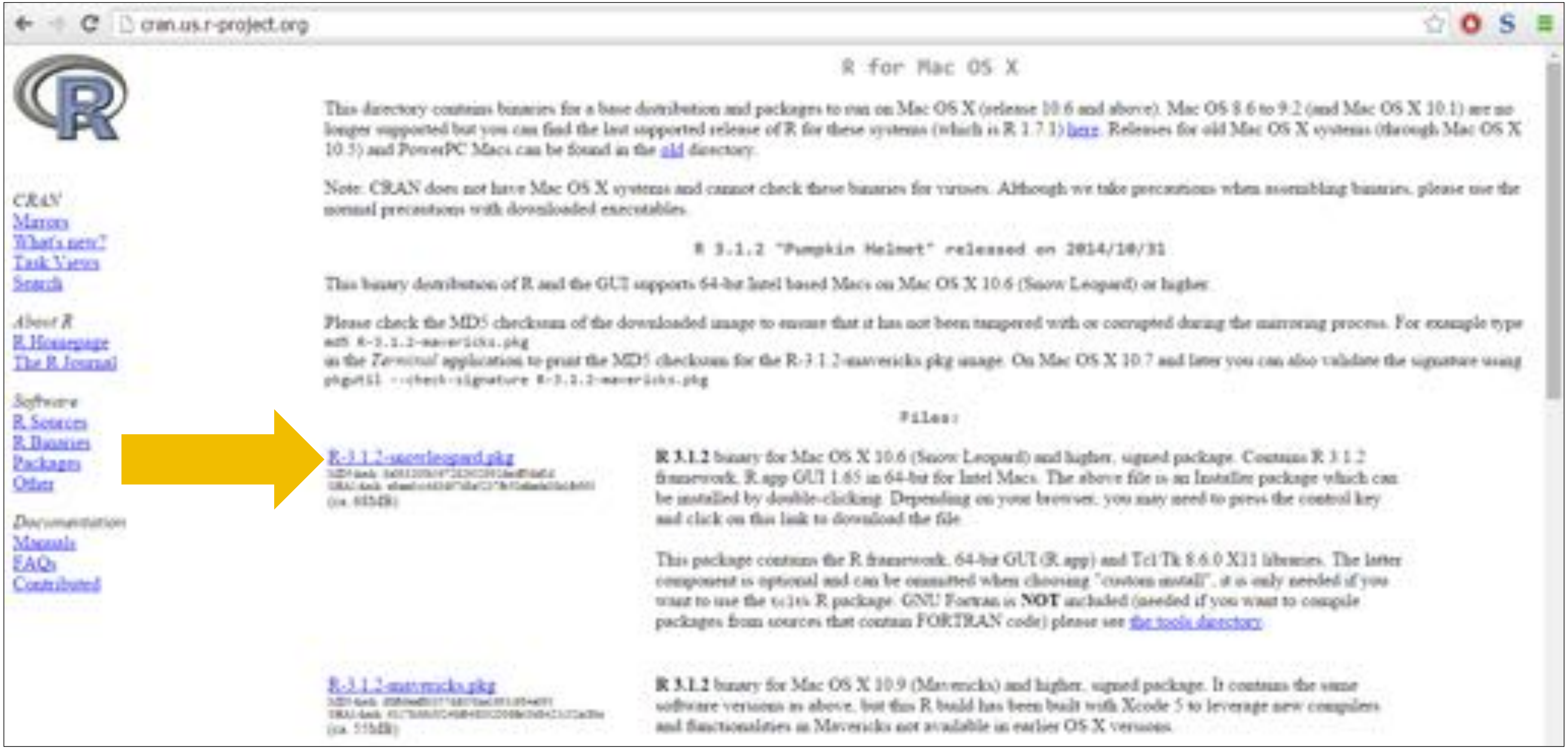
Installation and other instructions

New features in this version

CRAN

Mirrors

R for Mac



CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

R for Mac OS X

This directory contains binaries for a base distribution and packages to run on Mac OS X (release 10.6 and above). Mac OS 8.6 to 9.2 (and Mac OS X 10.1) are no longer supported but you can find the last supported release of R for these systems (which is R 1.7.1) here. Releases for old Mac OS X systems (through Mac OS X 10.5) and PowerPC Macs can be found in the old directory.

Note: CRAN does not have Mac OS X systems and cannot check these binaries for viruses. Although we take precautions when assembling binaries, please use the normal precautions with downloaded executables.

R 3.1.2 "Pumpkin Helmet" released on 2014/10/31

This binary distribution of R and the GUI supports 64-bit Intel based Macs on Mac OS X 10.6 (Snow Leopard) or higher.

Please check the MD5 checksum of the downloaded image to ensure that it has not been tampered with or corrupted during the mirroring process. For example type
`md5 R-3.1.2-mavericks.pkg`
in the *Terminal* application to print the MD5 checksum for the R-3.1.2-mavericks.pkg image. On Mac OS X 10.7 and later you can also validate the signature using
`pkgutil --check-signature R-3.1.2-mavericks.pkg`

Files:

R-3.1.2-snowleopard.pkg
MD5-hash: [illegible]
SHA1-hash: [illegible]
(ca. 65MB)

R 3.1.2 binary for Mac OS X 10.6 (Snow Leopard) and higher, signed package. Contains R 3.1.2 framework, R.app GUI 1.65 in 64-bit for Intel Macs. The above file is an Installer package which can be installed by double-clicking. Depending on your browser, you may need to press the control key and click on this link to download the file.

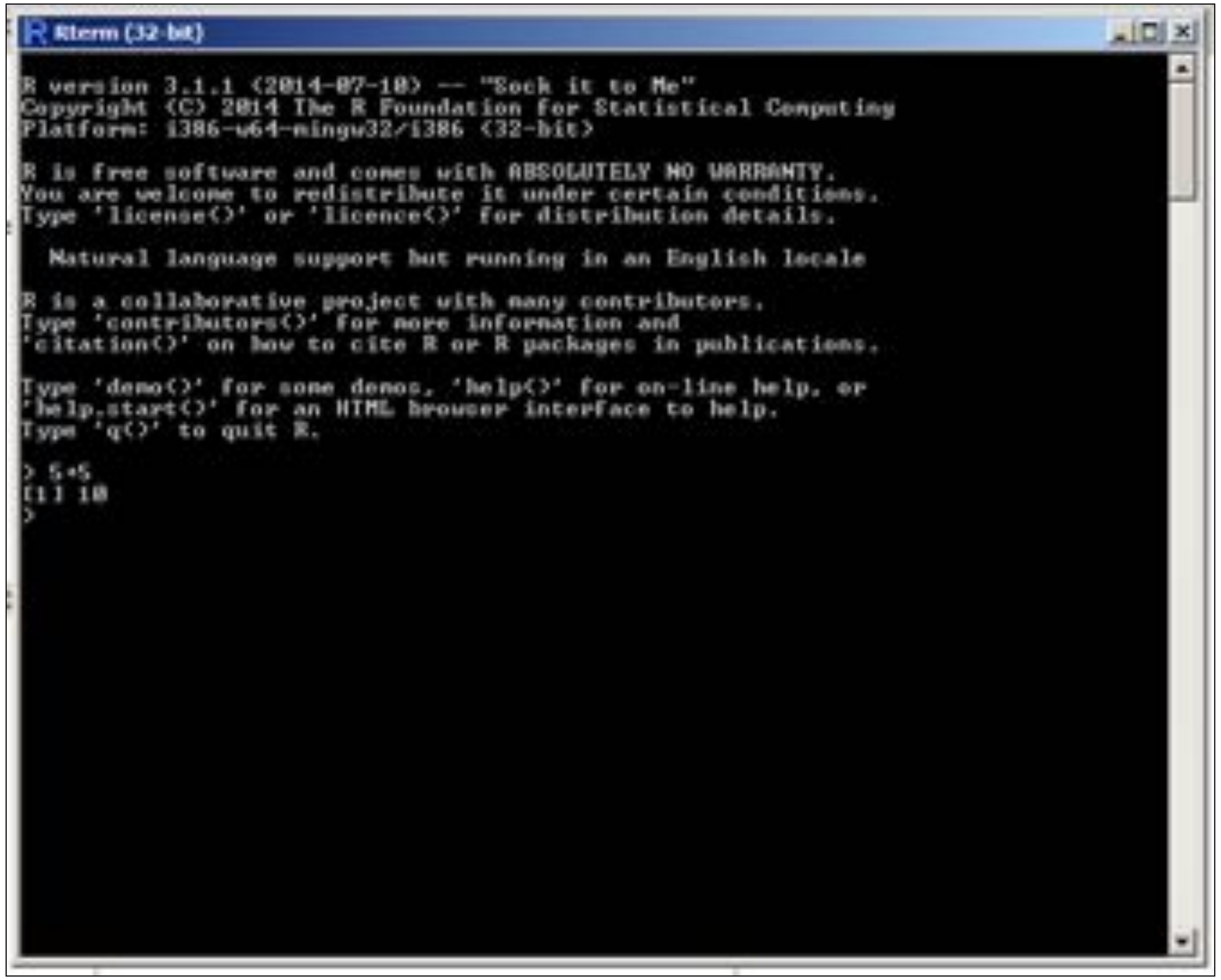
This package contains the R framework, 64-bit GUI (R.app) and Tcl/Tk 8.6.0 X11 libraries. The latter component is optional and can be ommitted when choosing "custom install", it is only needed if you want to use the tcltk R package. GNU Fortran is **NOT** included (needed if you want to compile packages from sources that contain FORTRAN code) please see the tools directory.

R-3.1.2-mavericks.pkg
MD5-hash: [illegible]
SHA1-hash: [illegible]
(ca. 55MB)

R 3.1.2 binary for Mac OS X 10.9 (Mavericks) and higher, signed package. It contains the same software versions as above, but this R build has been built with Xcode 5 to leverage new compilers and functionalities in Mavericks not available in earlier OS X versions.

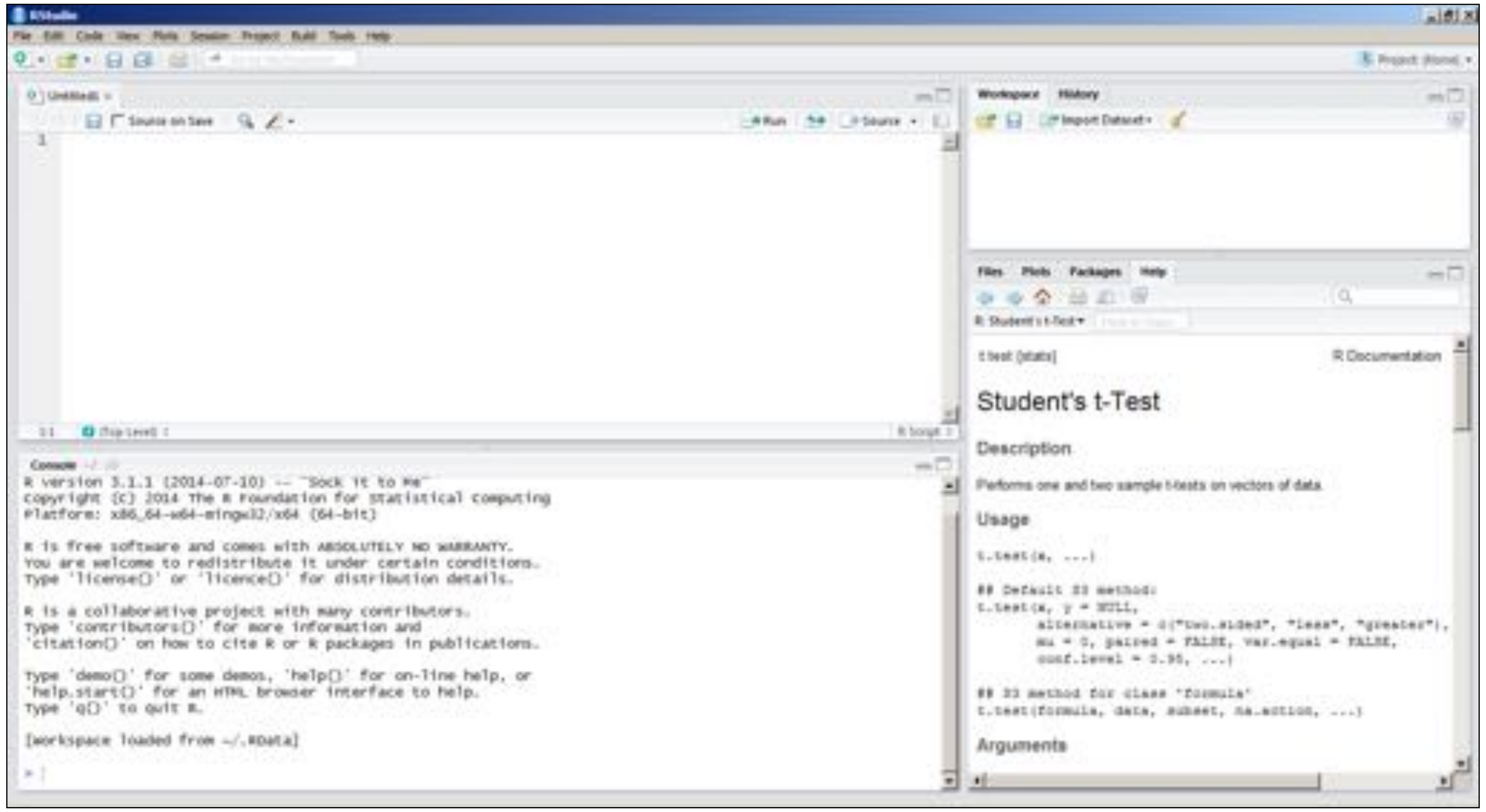
What is RStudio?

The user interface in the R terminal is bulky and non-intuitive



```
R version 3.1.1 (2014-07-10) -- "Sock it to Me"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 5+5
[1] 10
>
```

RStudio provides a better interface for a more intuitive user experience



```
R version 3.1.1 (2014-07-10) -- "Sock it to Me"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[workspace loaded from ~/.RData]

>
```

t.test {stats} R Documentation

Student's t-Test

Description

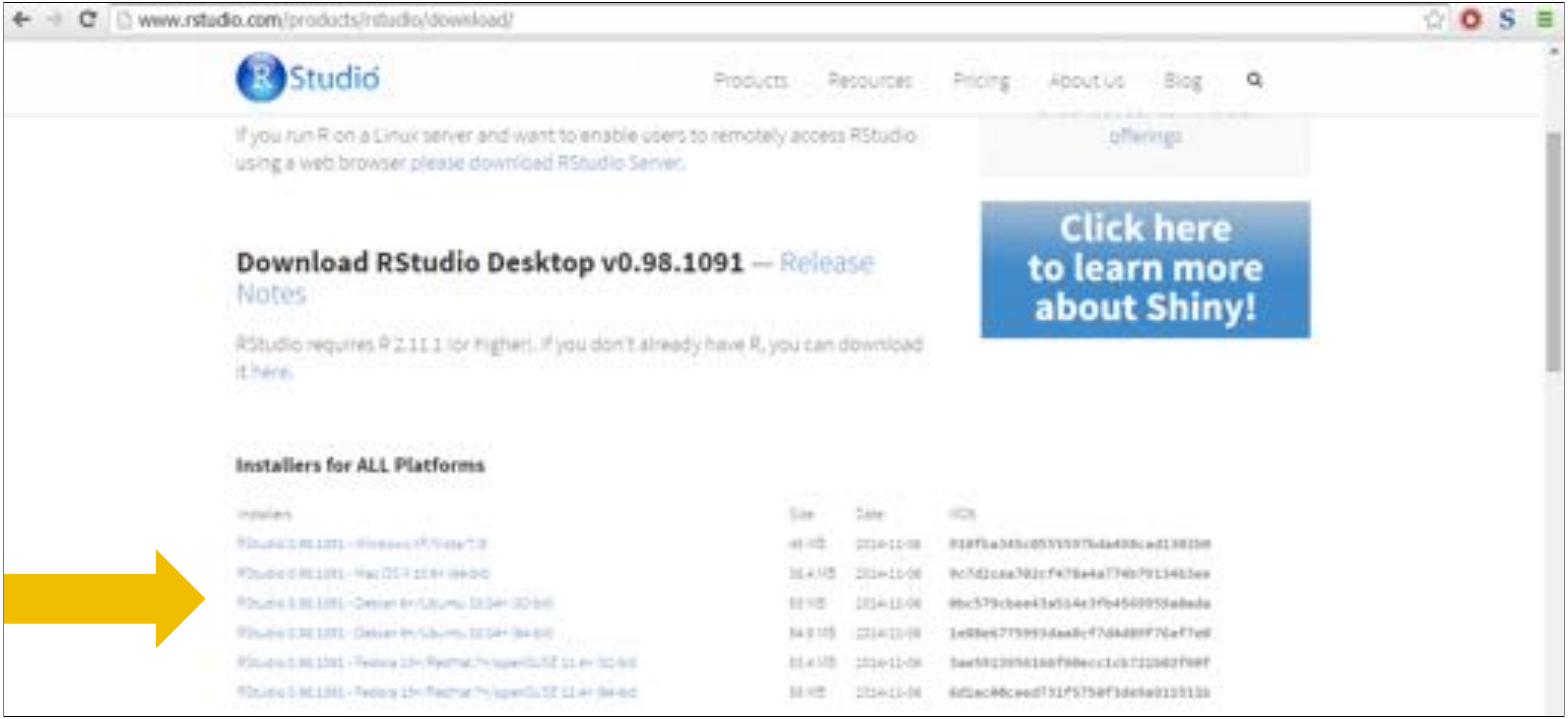
Performs one and two sample t-tests on vectors of data.

Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

## S3 method for class 'formula'
t.test(formula, data, subset, na.action, ...)
```

Arguments

Install R Studio



www.rstudio.com/products/rstudio/download/
RStudio
Products
Resources
Pricing
About us
Blog
offerings

If you run R on a Linux server and want to enable users to remotely access RStudio using a web browser please download RStudio Server.

Click here to learn more about Shiny!

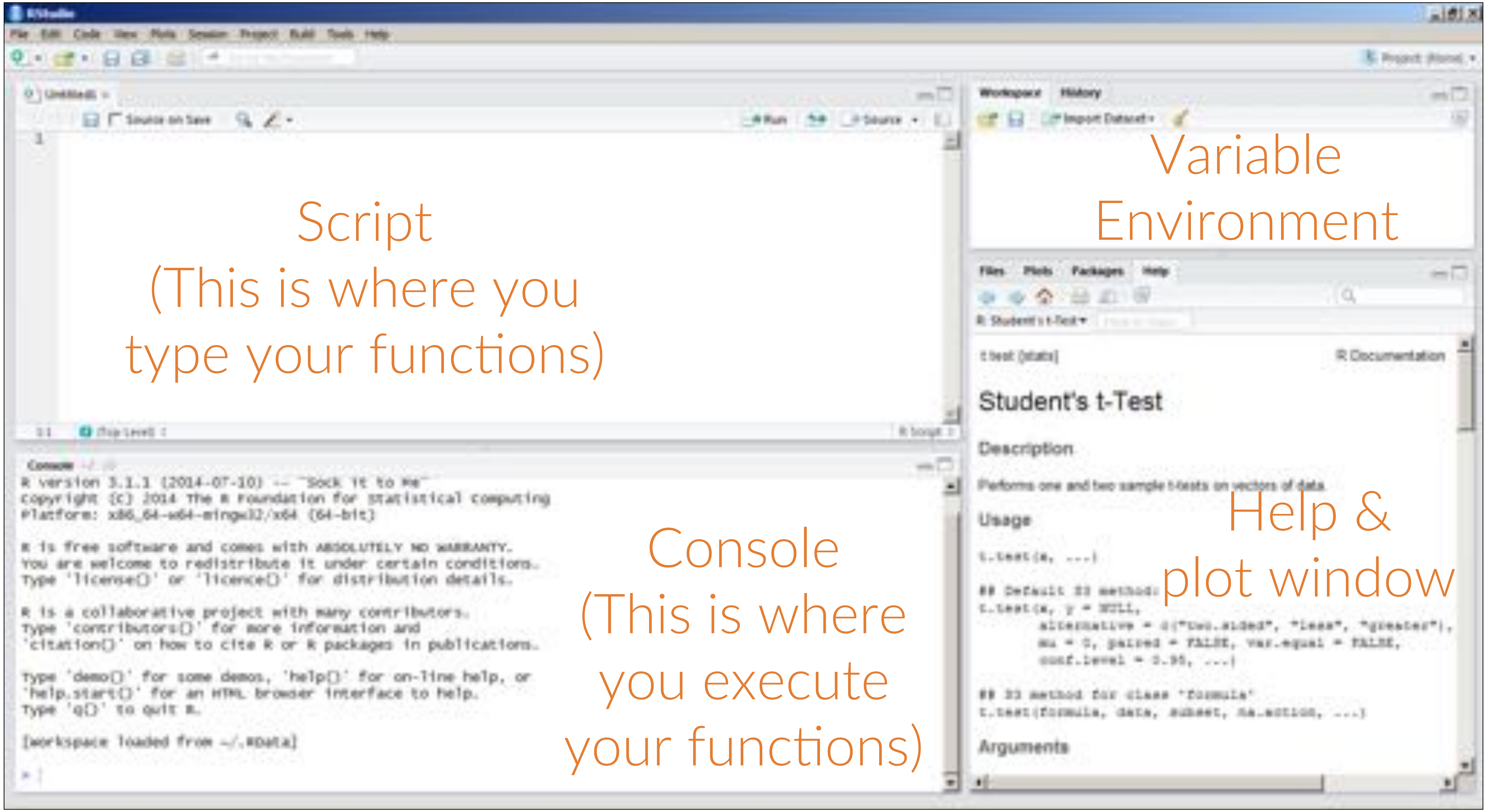
Download RStudio Desktop v0.98.1091 — Release Notes

RStudio requires R 2.11.1 (or higher). If you don't already have R, you can download it here.

Installers for ALL Platforms

Installers	Size	Date	MD5
RStudio 0.98.1091 - Windows XP/Vista/7/8	[illegible]	2014-11-06	[illegible]
RStudio 0.98.1091 - Mac OS X 10.6+ (64-bit)	[illegible]	2014-11-06	[illegible]
RStudio 0.98.1091 - Debian 6+/Ubuntu 10.04+ (32-bit)	[illegible]	2014-11-06	[illegible]
RStudio 0.98.1091 - Debian 6+/Ubuntu 10.04+ (64-bit)	[illegible]	2014-11-06	[illegible]
RStudio 0.98.1091 - Fedora 13+/RedHat 7+/openSUSE 11.4+ (32-bit)	[illegible]	2014-11-06	[illegible]
RStudio 0.98.1091 - Fedora 13+/RedHat 7+/openSUSE 11.4+ (64-bit)	[illegible]	2014-11-06	[illegible]

RStudio



Script
(This is where you
type your functions)

Variable
Environment

Console
(This is where
you execute
your functions)

Help &
plot window

```
R version 3.1.1 (2014-07-10) -- "Sock it to Me"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

>
```

t.test {stats} R Documentation

Student's t-Test

Description

Performs one and two sample t-tests on vectors of data.

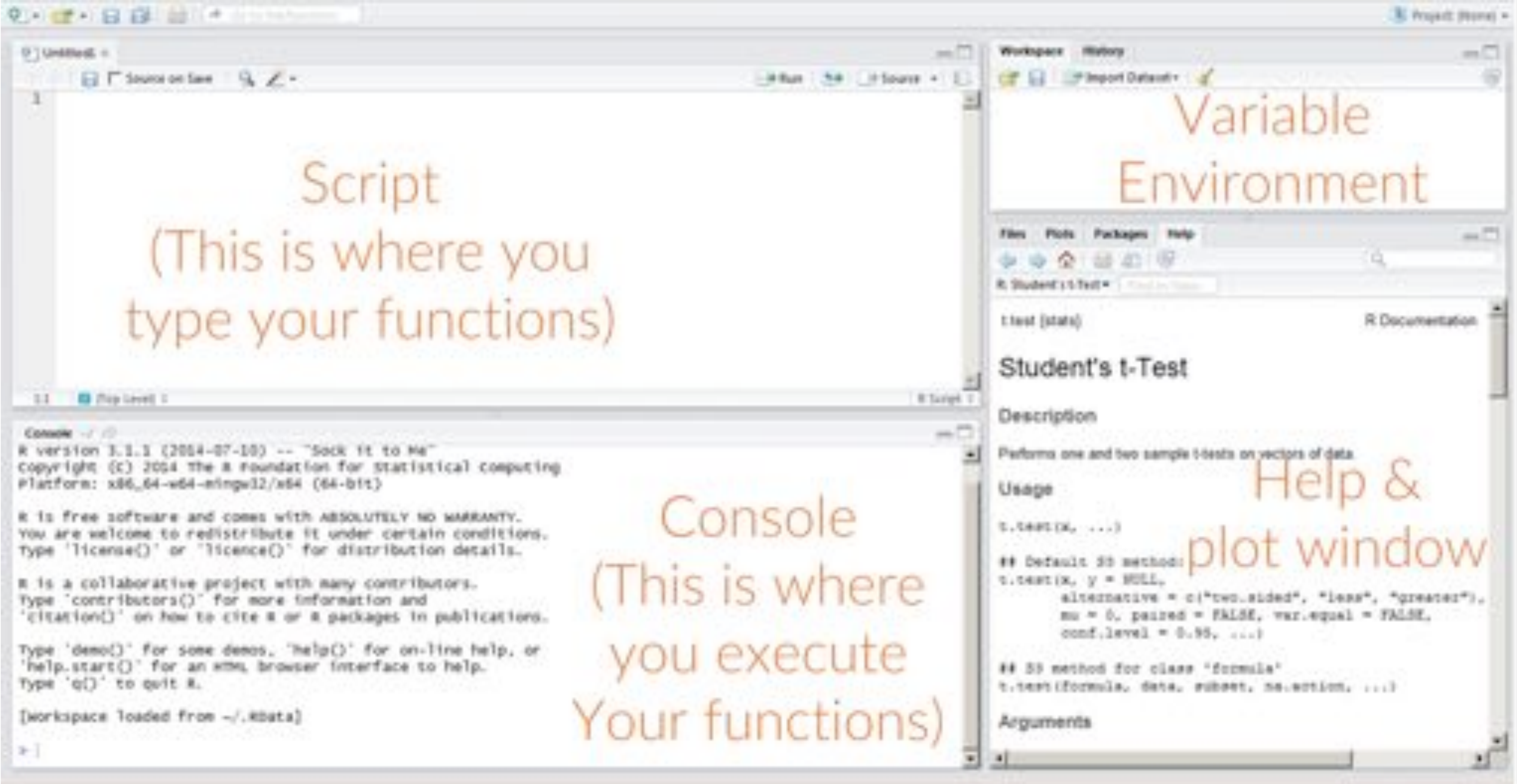
Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

## S3 method for class 'formula'
t.test(formula, data, subset, na.action, ...)
```

Arguments

Script and Console

- Script
  - We use scripts to generate re-usable code
  - They are like macros
  - Code is not executed here unless you press run (see next slide)
  - <u>Good coding practice:</u>
    - When saving a script use a name that describes what the script does i.e. counter
    - Comment your code using #
- Console
  - This is where the code executes
  - It is the actual R program (backend)
  - Anything that you type in here will be executed



```
R version 3.1.1 (2014-07-10) -- "Sock it to Me"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

>
```

t.test {stats} R Documentation

Student's t-Test

Description

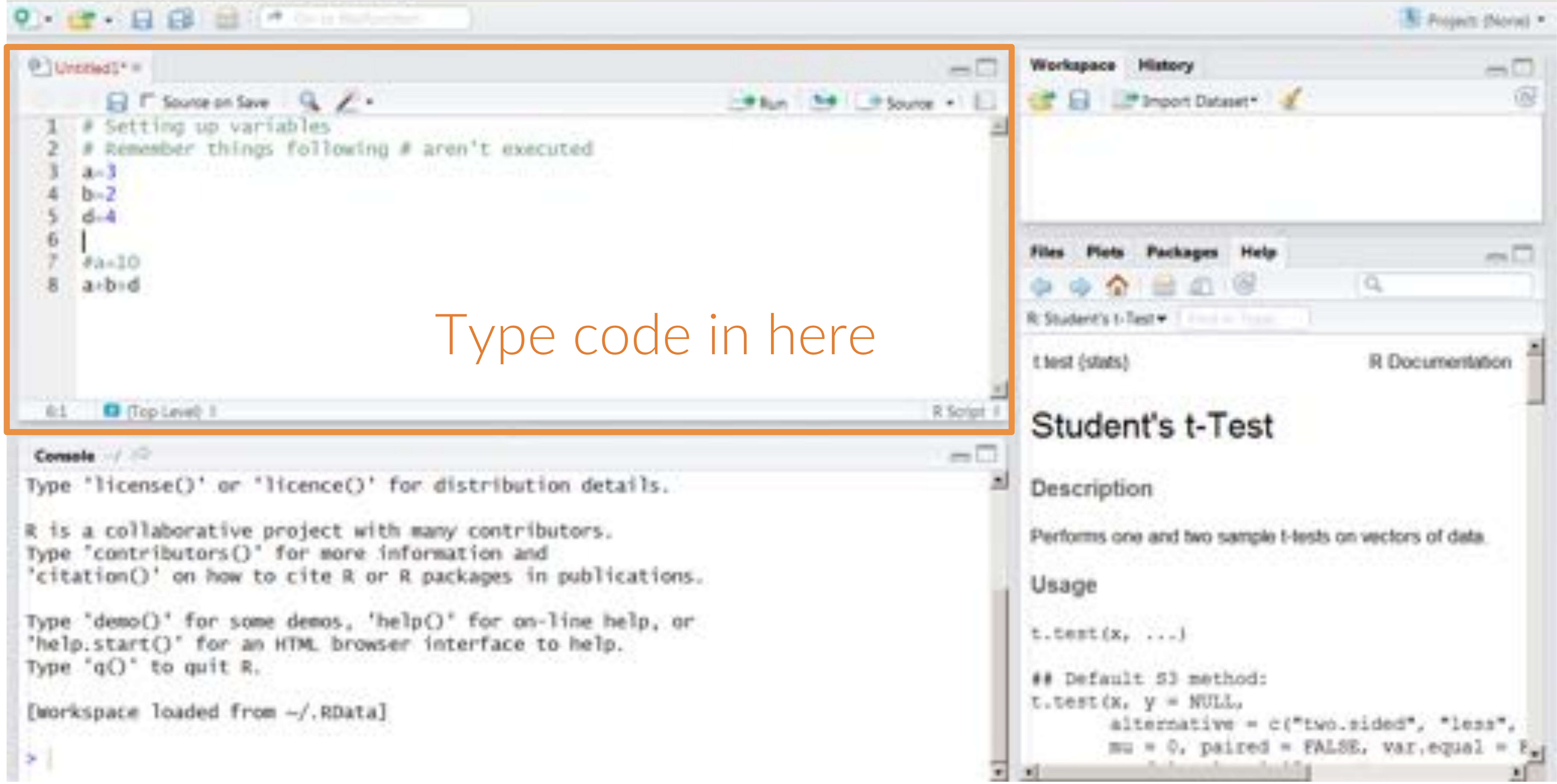
Performs one and two sample t-tests on vectors of data.

Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

## S3 method for class 'formula'
t.test(formula, data, subset, na.action, ...)
```

Arguments

Running code from script



```
# Setting up variables
# Remember things following # aren't executed
a<-3
b<-2
d<-4

#a<-10
a+b+d
```

Type code in here

```
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

>
```

t.test {stats} R Documentation

Student's t-Test

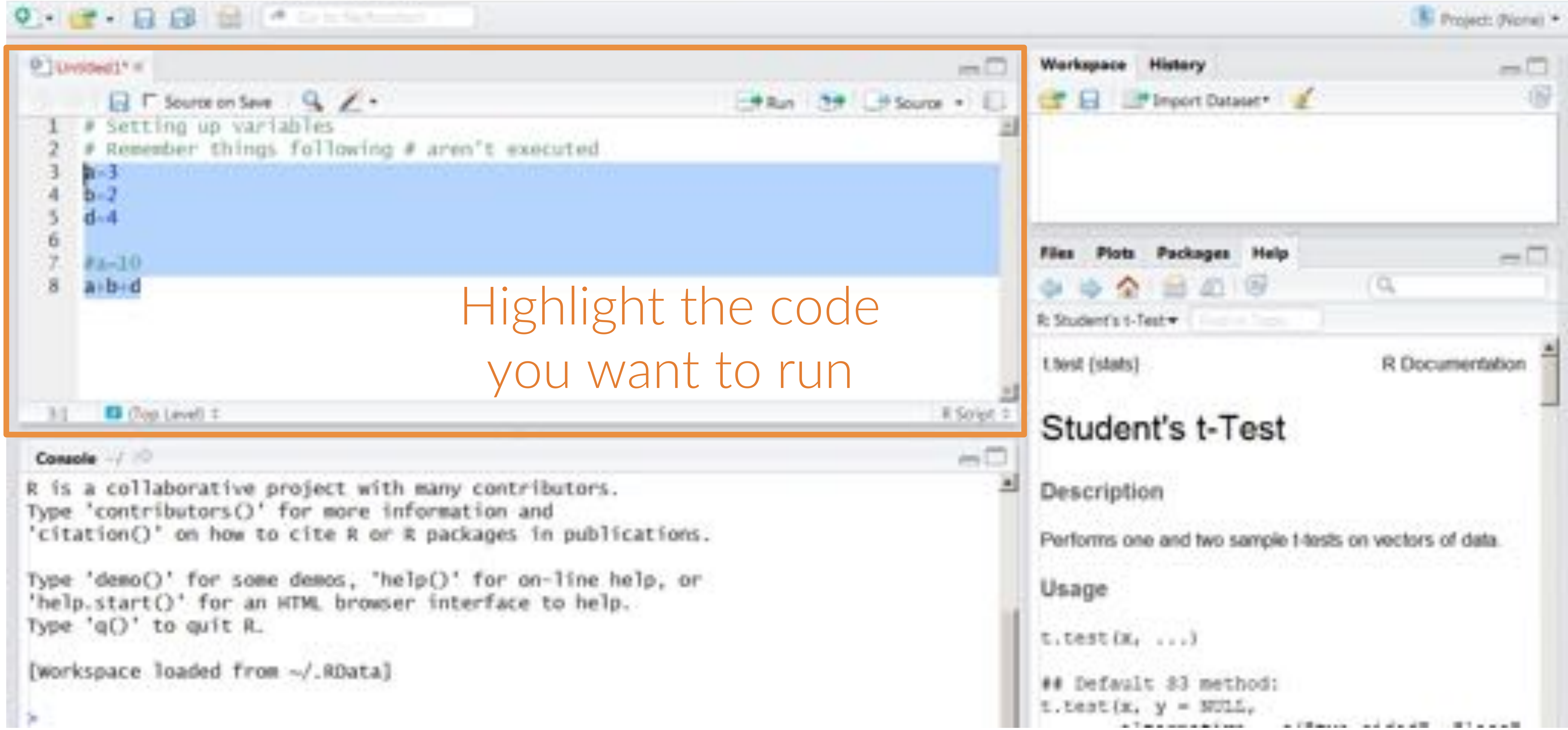
Description

Performs one and two sample t-tests on vectors of data.

Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
       alternative = c("two.sided", "less",
       mu = 0, paired = FALSE, var.equal = F
```

Running code from a script



```
# Setting up variables
# Remember things following # aren't executed
a<-3
b<-2
d<-4

#a<-10
a+b+d
```

Highlight the code you want to run

```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

>
```

t.test {stats} R Documentation

Student's t-Test

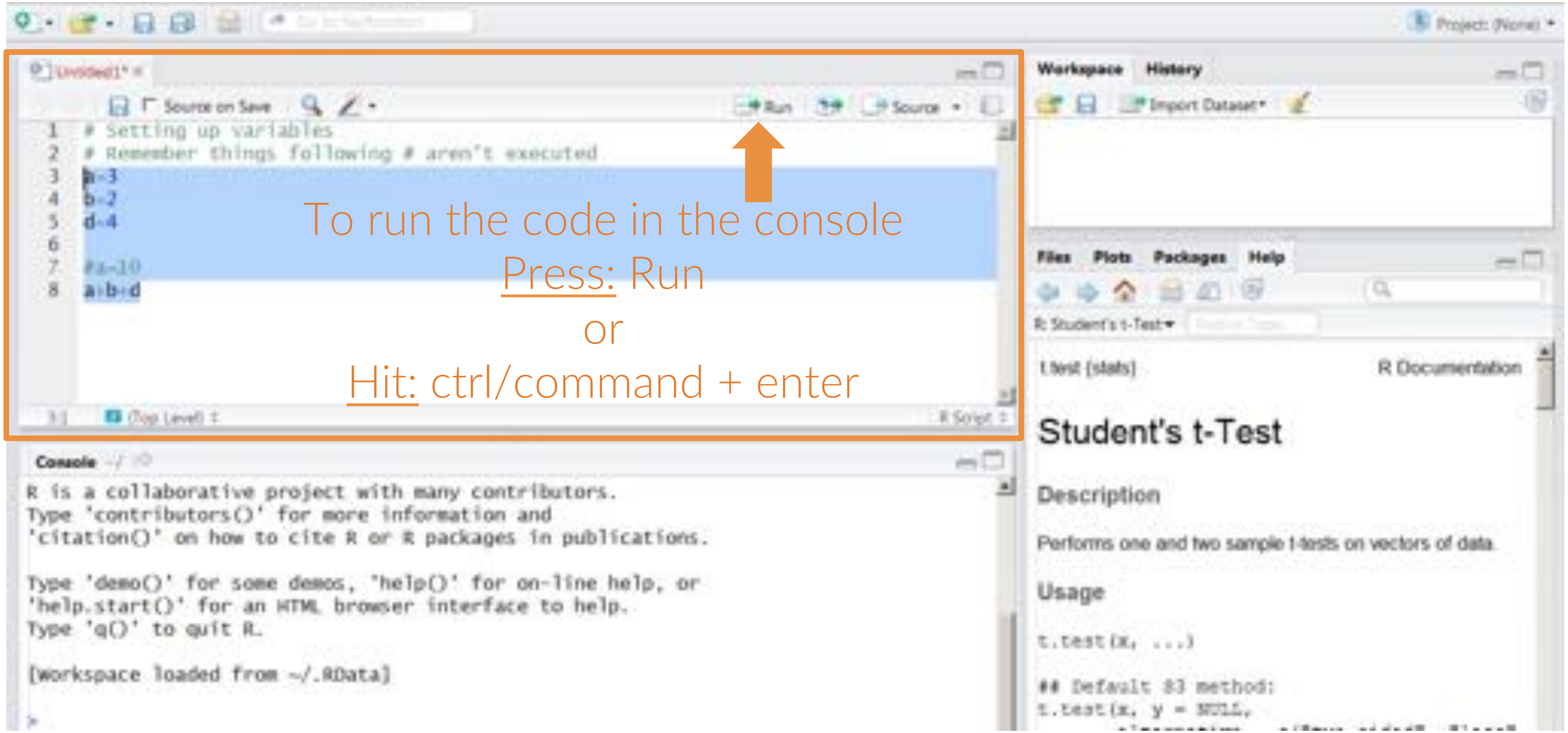
Description

Performs one and two sample t-tests on vectors of data.

Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
```

Running code from a script



```
# Setting up variables
# Remember things following # aren't executed
a<-3
b<-2
d<-4

#a<-10
a+b+d
```

To run the code in the console

Press: Run

or

Hit: ctrl/command + enter

```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

>
```

t.test {stats} R Documentation

Student's t-Test

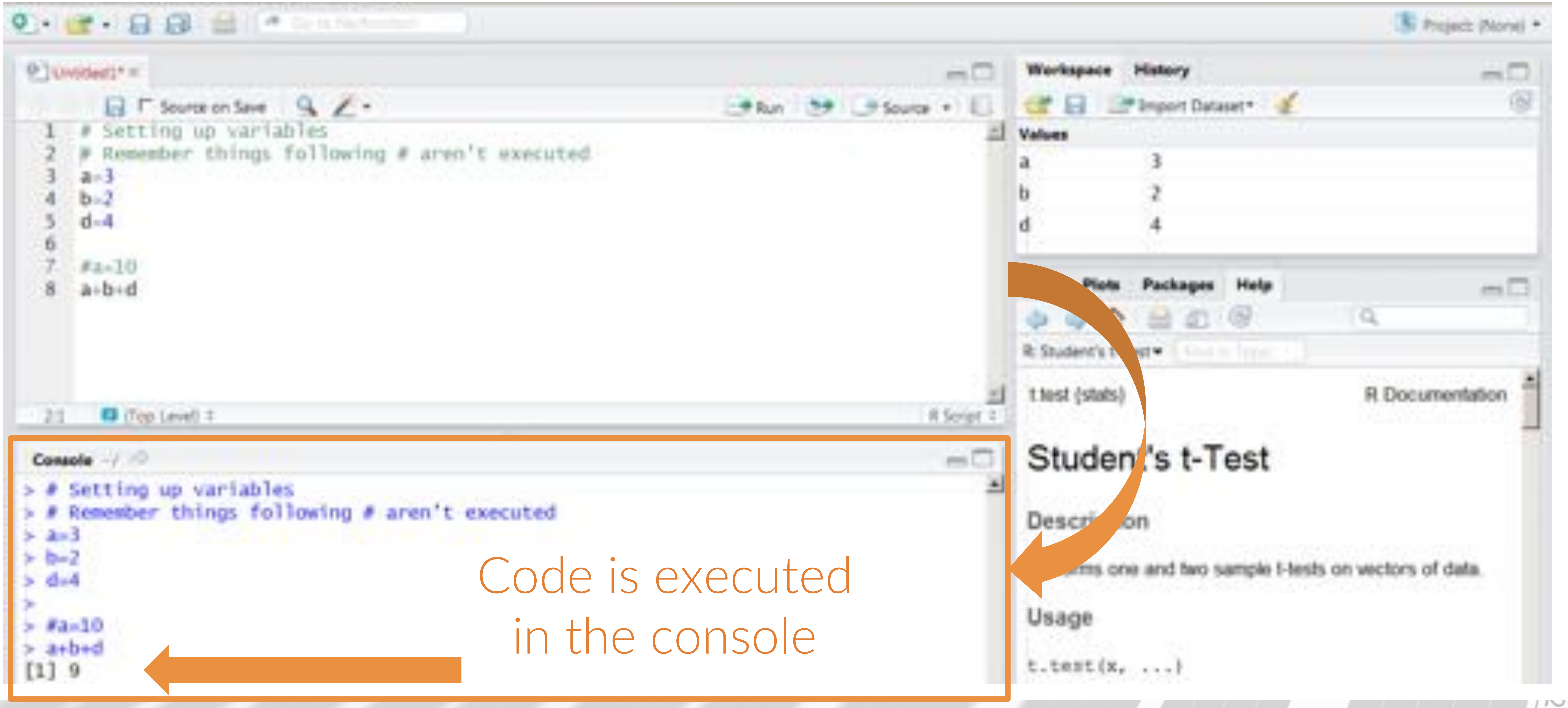
Description

Performs one and two sample t-tests on vectors of data.

Usage

```
t.test(x, ...)

## Default S3 method:
t.test(x, y = NULL,
```

Running code from a script



```
# Setting up variables
# Remember things following # aren't executed
a=3
b=2
d=4

#a=10
a+b+d
```

Values	
a	3
b	2
d	4

t.test {stats}   R Documentation

Student's t-Test

Description

...rms one and two sample t-tests on vectors of data.

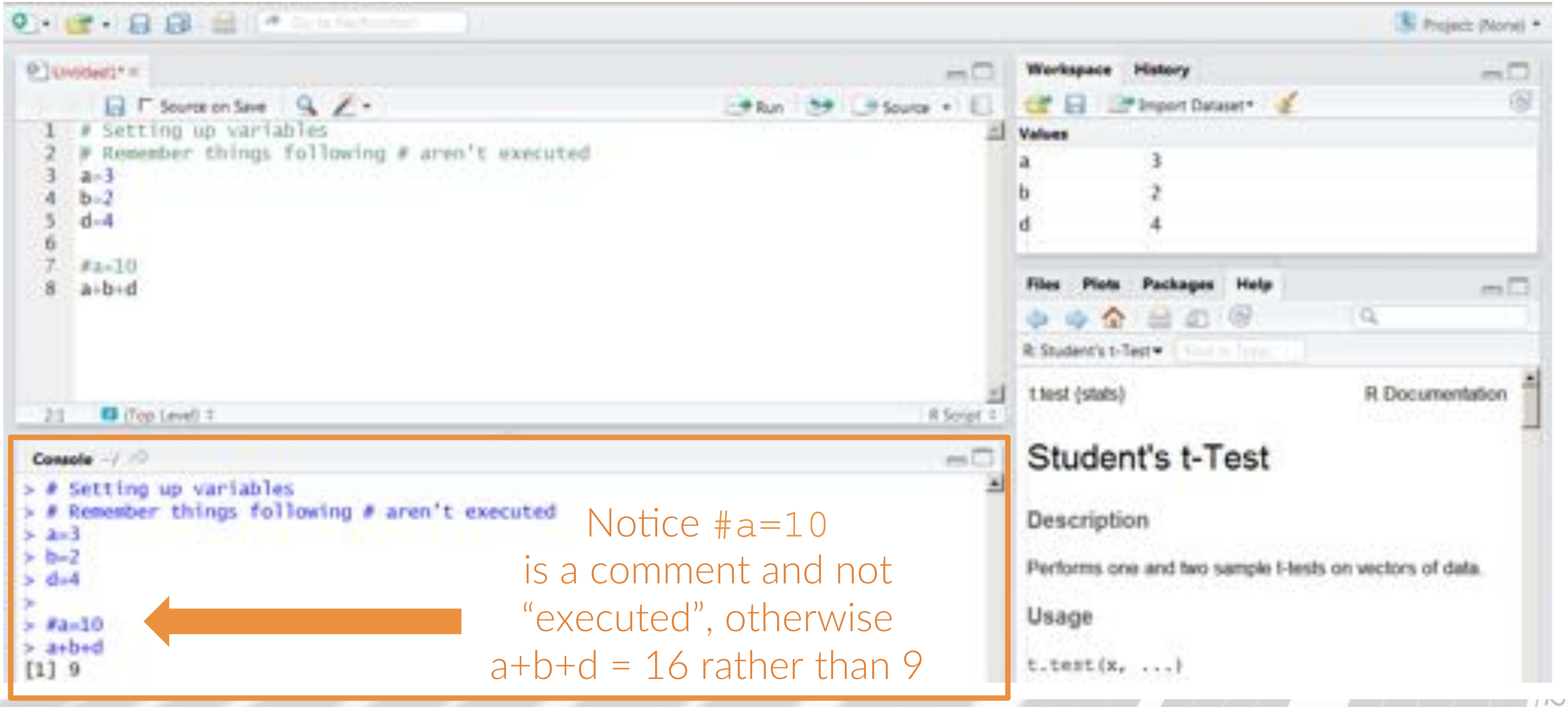
Usage

```
t.test(x, ...)
```

```
> # Setting up variables
> # Remember things following # aren't executed
> a=3
> b=2
> d=4
>
> #a=10
> a+b+d
[1] 9
```

Code is executed in the console

Running code from a script



```
# Setting up variables
# Remember things following # aren't executed
a=3
b=2
d=4

#a=10
a+b+d
```

Values

a	3
b	2
d	4

t.test {stats}   R Documentation

Student's t-Test

Description

Performs one and two sample t-tests on vectors of data.

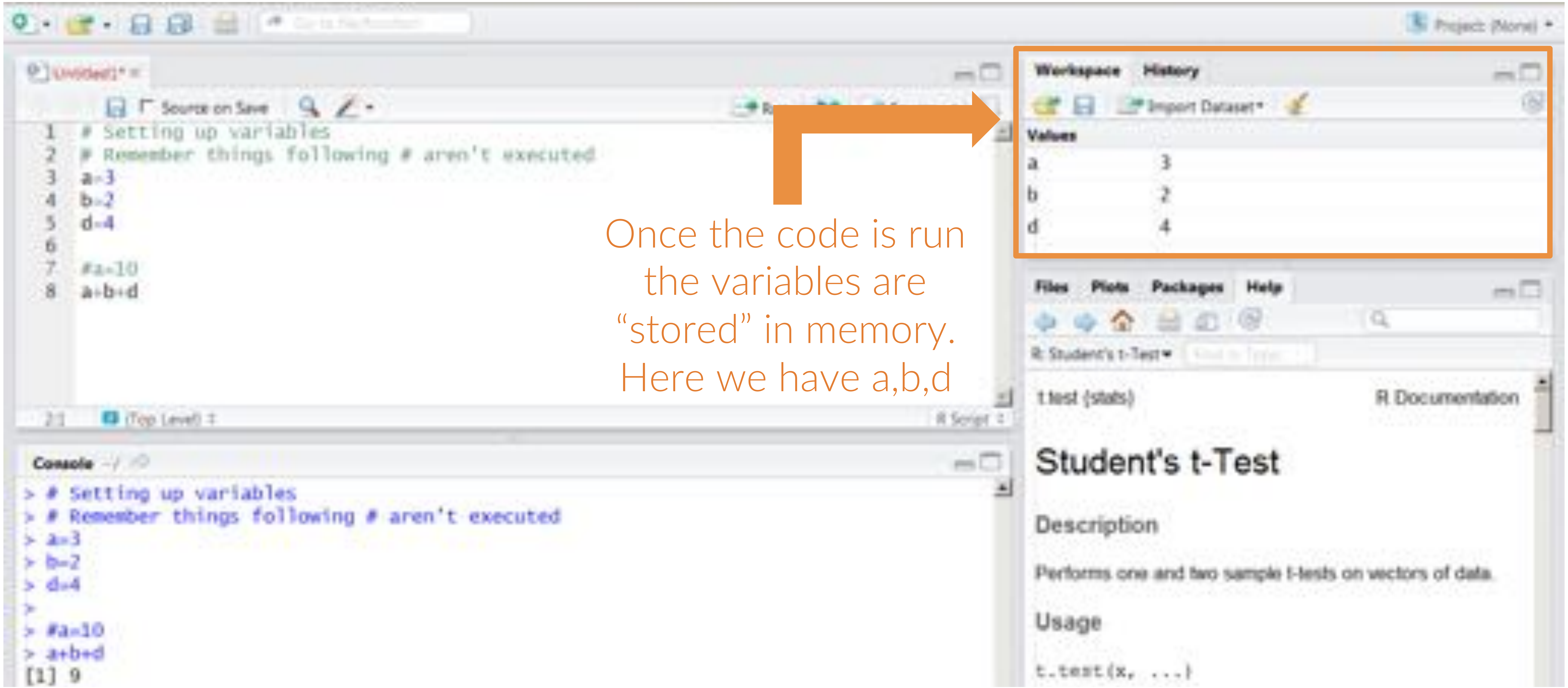
Usage

`t.test(x, ...)`

```
> # Setting up variables
> # Remember things following # aren't executed
> a=3
> b=2
> d=4
>
> #a=10
> a+b+d
[1] 9
```

Notice #a=10
is a comment and not
"executed", otherwise
a+b+d = 16 rather than 9

Variables



```
# Setting up variables
# Remember things following # aren't executed
a=3
b=2
d=4

#a=10
a+b+d
```

```
> # Setting up variables
> # Remember things following # aren't executed
> a=3
> b=2
> d=4
>
> #a=10
> a+b+d
[1] 9
```

Values

a	3
b	2
d	4

t.test {stats} R Documentation

Student's t-Test

Description

Performs one and two sample t-tests on vectors of data.

Usage

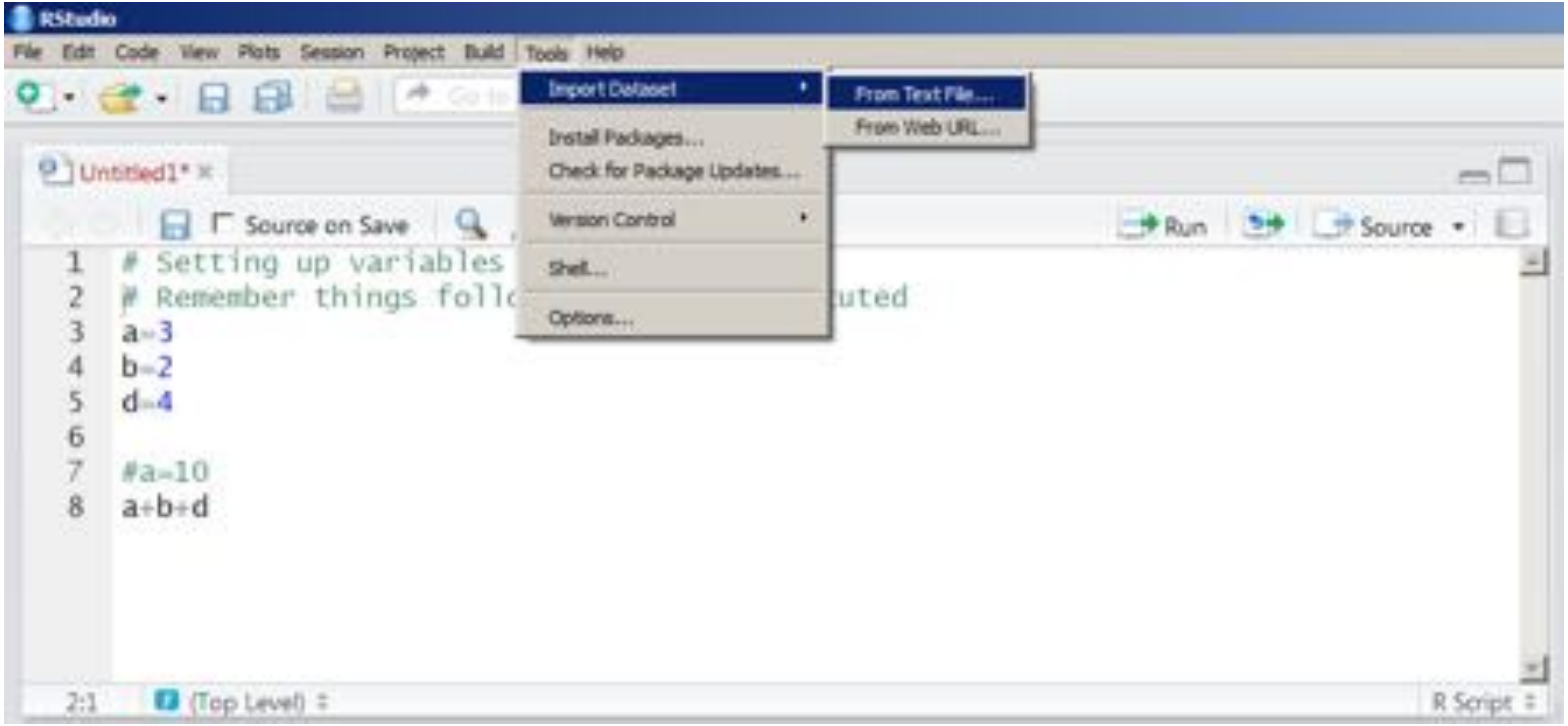
```
t.test(x, ...)
```

Once the code is run the variables are "stored" in memory. Here we have a,b,d

Reading in data

- Data comes in many forms
  - R can read in most forms
- The most common to work with now days are
  - `csv` files – comma separated value files
  - `tsv` files – tab separated value files
- R can also read in others (excel and json) but that is not covered here

Manually

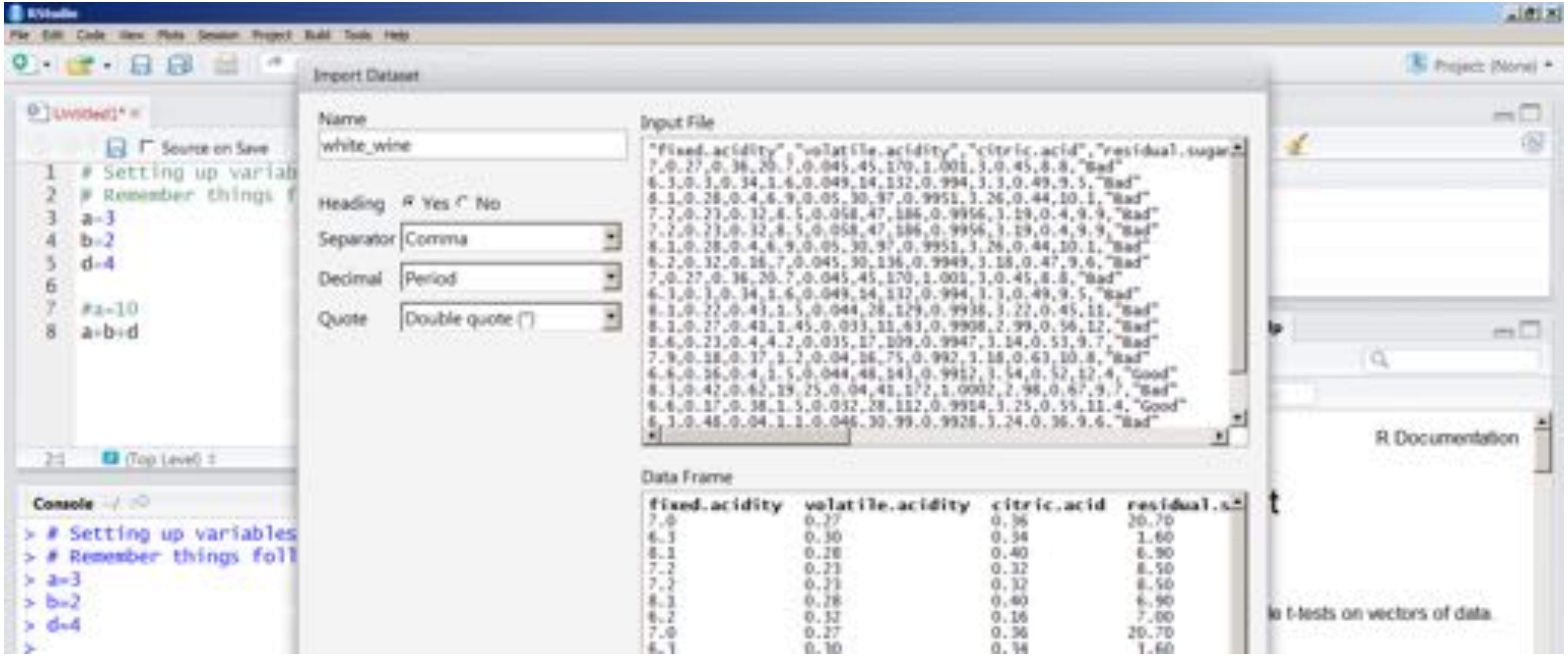


```
# Setting up variables
# Remember things foll                   uted
a=3
b=2
d=4

#a=10
a+b+d
```

Tools > Import Dataset > From Text File...

Manually

- R will automatically select options for you such as heading, sometimes it makes mistakes so be sure to double check



```
# Setting up variab
# Remember things f
a=3
b=2
d=4

#a=10
a+b+d
```

Console

```
> # Setting up variables
> # Remember things foll
> a=3
> b=2
> d=4
>
```

Import Dataset

Name: white_wine

Heading: ● Yes ○ No

Separator: Comma

Decimal: Period

Quote: Double quote (")

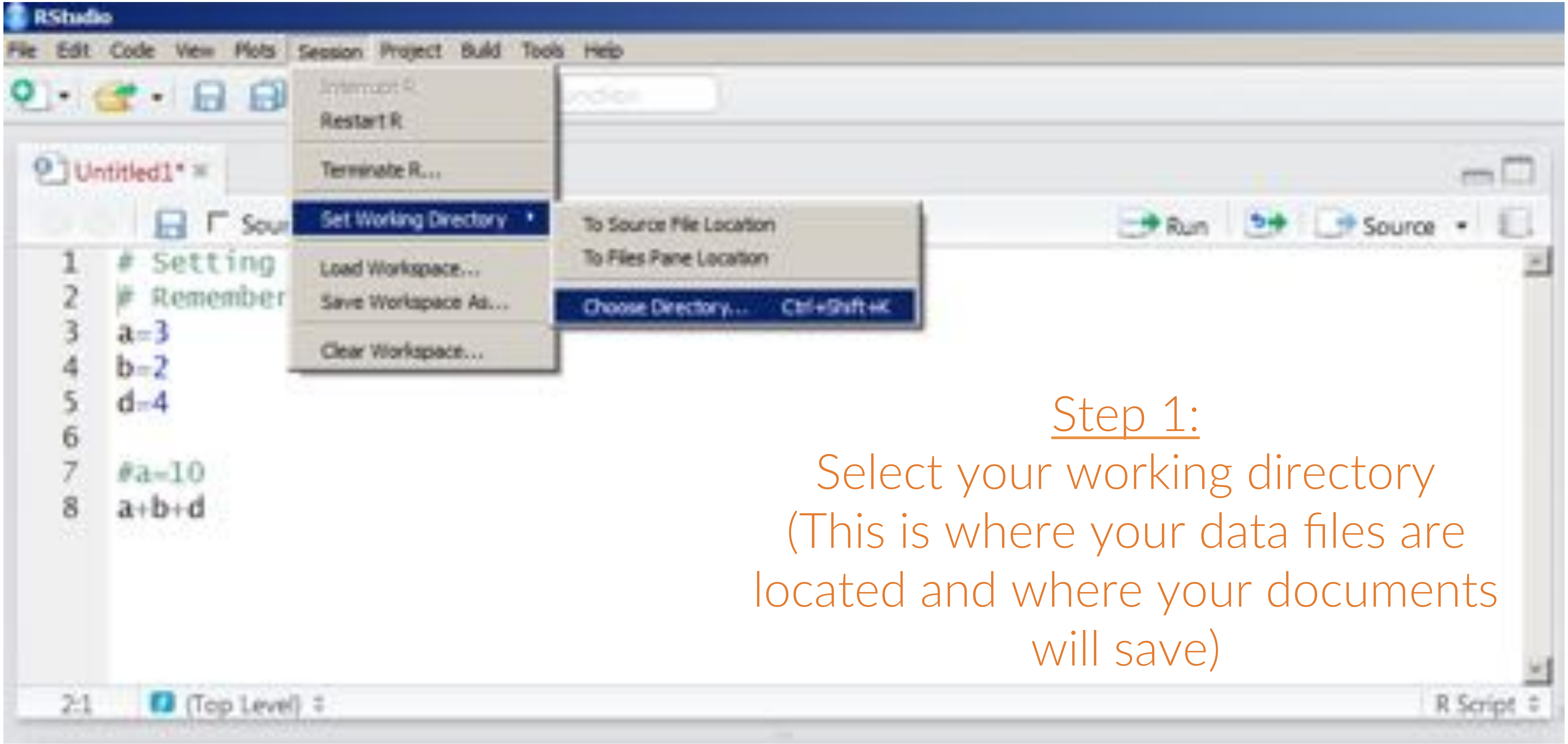
Input File

```
"fixed.acidity","volatile.acidity","citric.acid","residual.sugar
7,0.27,0.36,20.7,0.045,45,170,1.001,3,0.45,8.8,"Bad"
6.3,0.3,0.34,1.6,0.049,14,132,0.994,3.3,0.49,9.5,"Bad"
8.1,0.28,0.4,6.9,0.05,30,97,0.9951,3.26,0.44,10.1,"Bad"
7.2,0.23,0.32,8.5,0.058,47,186,0.9956,3.19,0.4,9.9,"Bad"
7.2,0.23,0.32,8.5,0.058,47,186,0.9956,3.19,0.4,9.9,"Bad"
8.1,0.28,0.4,6.9,0.05,30,97,0.9951,3.26,0.44,10.1,"Bad"
6.2,0.32,0.16,7,0.045,30,136,0.9949,3.18,0.47,9.6,"Bad"
7,0.27,0.36,20.7,0.045,45,170,1.001,3,0.45,8.8,"Bad"
6.3,0.3,0.34,1.6,0.049,14,132,0.994,3.3,0.49,9.5,"Bad"
8.1,0.22,0.43,1.5,0.044,28,129,0.9938,3.22,0.45,11,"Bad"
8.1,0.27,0.41,1.45,0.033,11,63,0.9908,2.99,0.56,12,"Bad"
8.6,0.23,0.4,4.2,0.035,17,109,0.9947,3.14,0.53,9.7,"Bad"
7.9,0.18,0.37,1.2,0.04,16,75,0.992,3.18,0.63,10.8,"Bad"
6.6,0.16,0.4,1.5,0.044,48,143,0.9912,3.54,0.52,12.4,"Good"
8.3,0.42,0.62,19.25,0.04,41,172,1.0002,2.98,0.67,9.7,"Bad"
6.6,0.17,0.38,1.5,0.032,28,112,0.9914,3.25,0.55,11.4,"Good"
6.3,0.48,0.04,1.1,0.046,30,99,0.9928,3.24,0.36,9.6,"Bad"
```

Data Frame

```
fixed.acidity  volatile.acidity  citric.acid  residual.s
7.0            0.27              0.36         20.70
6.3            0.30              0.34          1.60
8.1            0.28              0.40          6.90
7.2            0.23              0.32          8.50
7.2            0.23              0.32          8.50
8.1            0.28              0.40          6.90
6.2            0.32              0.16          7.00
7.0            0.27              0.36         20.70
6.1            0.30              0.34          1.60
```

Through the Script

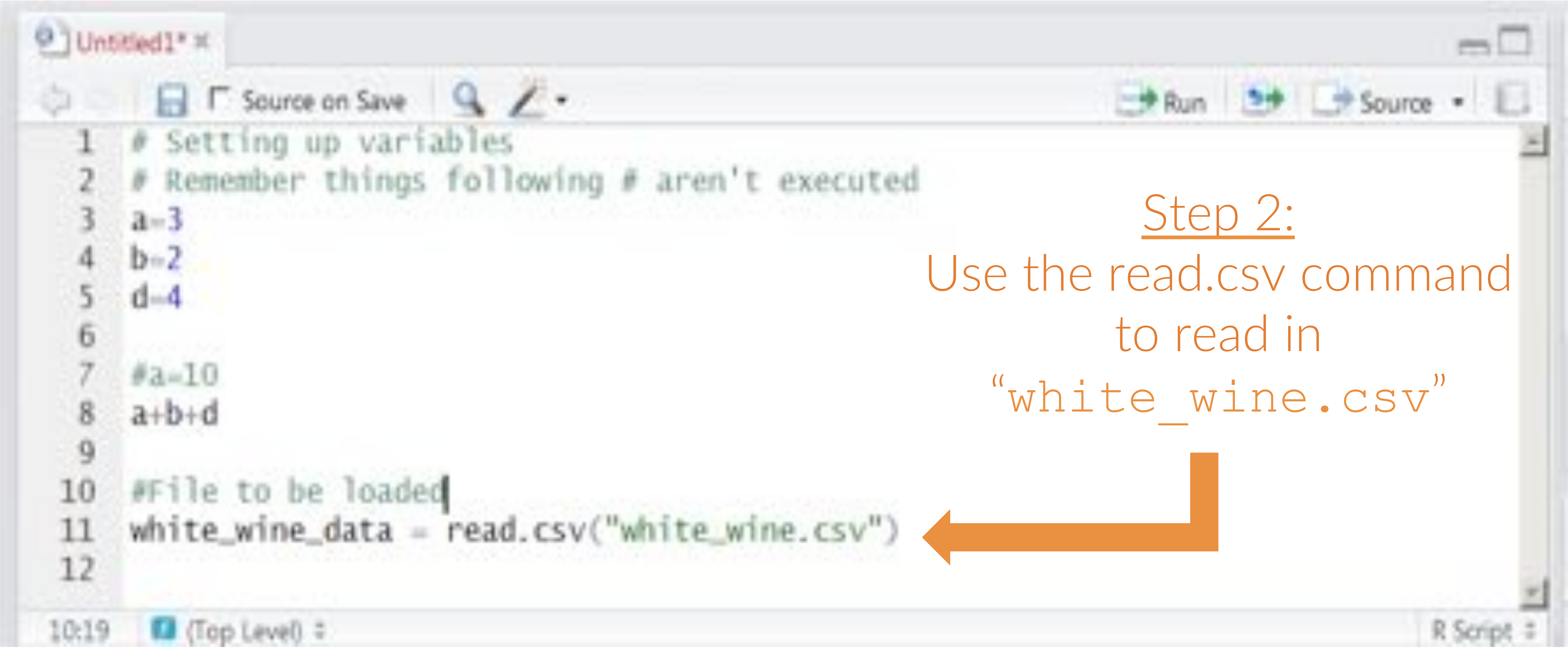


```
# Setting
# Remember
a=3
b=2
d=4

#a=10
a+b+d
```

Step 1:

Select your working directory (This is where your data files are located and where your documents will save)

Through the script

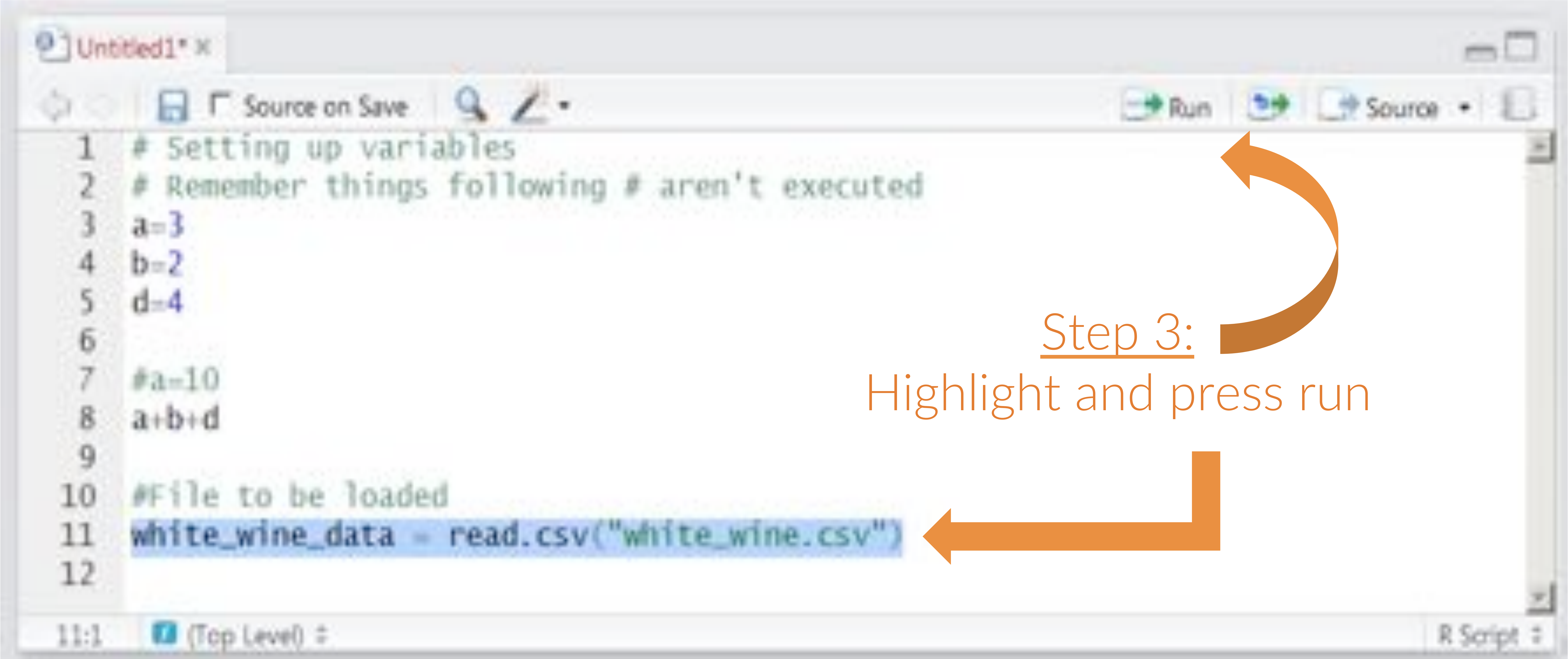


```
# Setting up variables
# Remember things following # aren't executed
a=3
b=2
d=4

#a=10
a+b+d

#File to be loaded
white_wine_data = read.csv("white_wine.csv")

```

Step 2:

Use the read.csv command to read in “`white_wine.csv`”

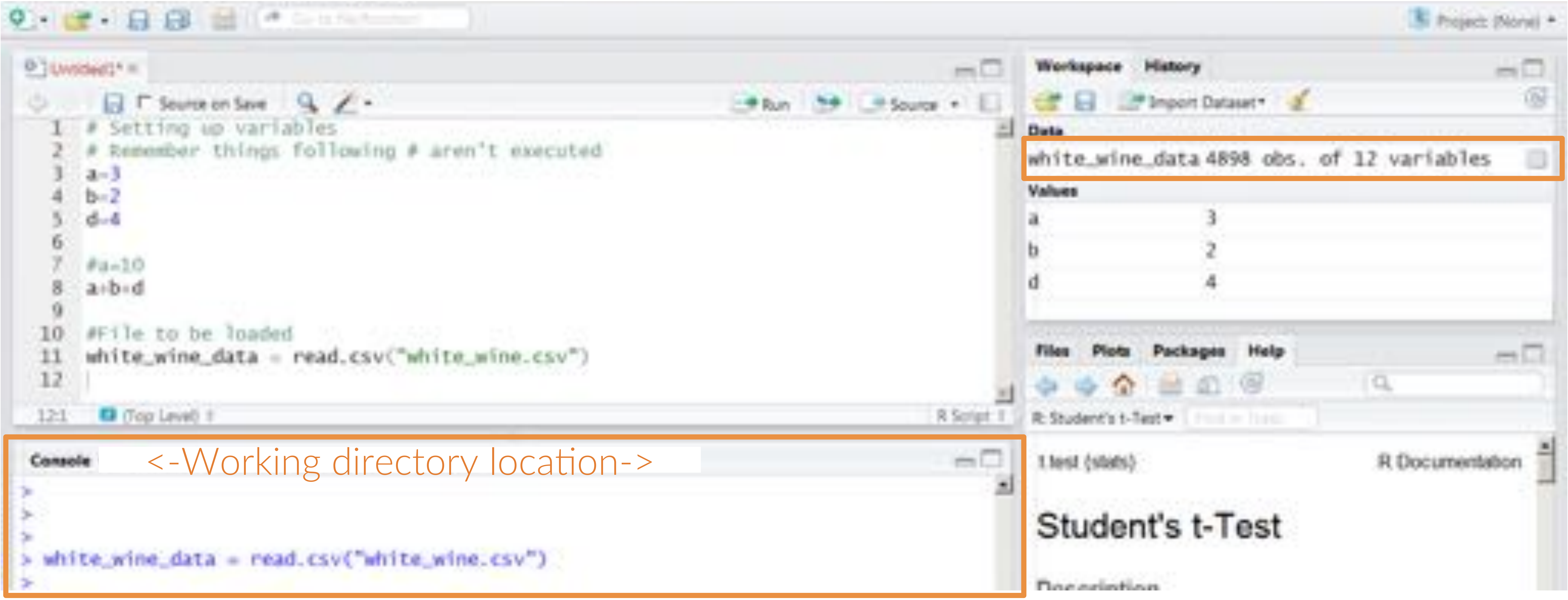
Through the script



```
# Setting up variables
# Remember things following # aren't executed
a=3
b=2
d=4

#a=10
a+b+d

#File to be loaded
white_wine_data = read.csv("white_wine.csv")

```

Step 3:
Highlight and press run

Through the script



```
# Setting up variables
# Remember things following # aren't executed
a=3
b=2
d=4

#a=10
a+b+d

#File to be loaded
white_wine_data = read.csv("white_wine.csv")
```

Data

white_wine_data 4898 obs. of 12 variables

Values

a	3
b	2
d	4

Console <-Working directory location->

```
>
>
>
> white_wine_data = read.csv("white_wine.csv")
>
```

Outline

- What is data science?
- A data scientist's approach
- Introduction to R
  - Calculations in R
  - Reading data into R
  - Manipulating data in R
- Visualization in R
  - Basic plotting

Why use R

1. De facto standard among professional statisticians
2. Comparable and often superior in power to commercial products (SAS, SPSS)
3. Available for the Windows, Mac, and Linux operating systems
4. R is a general-purpose programming language, so you can use it to automate analyses
5. Create dynamic graphics and visualization
6. Large community of users, many are prominent scientists: www.r-bloggers.com
7. Pre-made packages to run data analyses contributed by user base (over 6,500 packages)

Source: http://cran.r-project.org/web/packages/

Uses of R

1. Can be used to analyze and visualize data

2. Can be used to write software

3. Can be used to create data products and applications

In this course, we will focus on how to analyze and visualize data

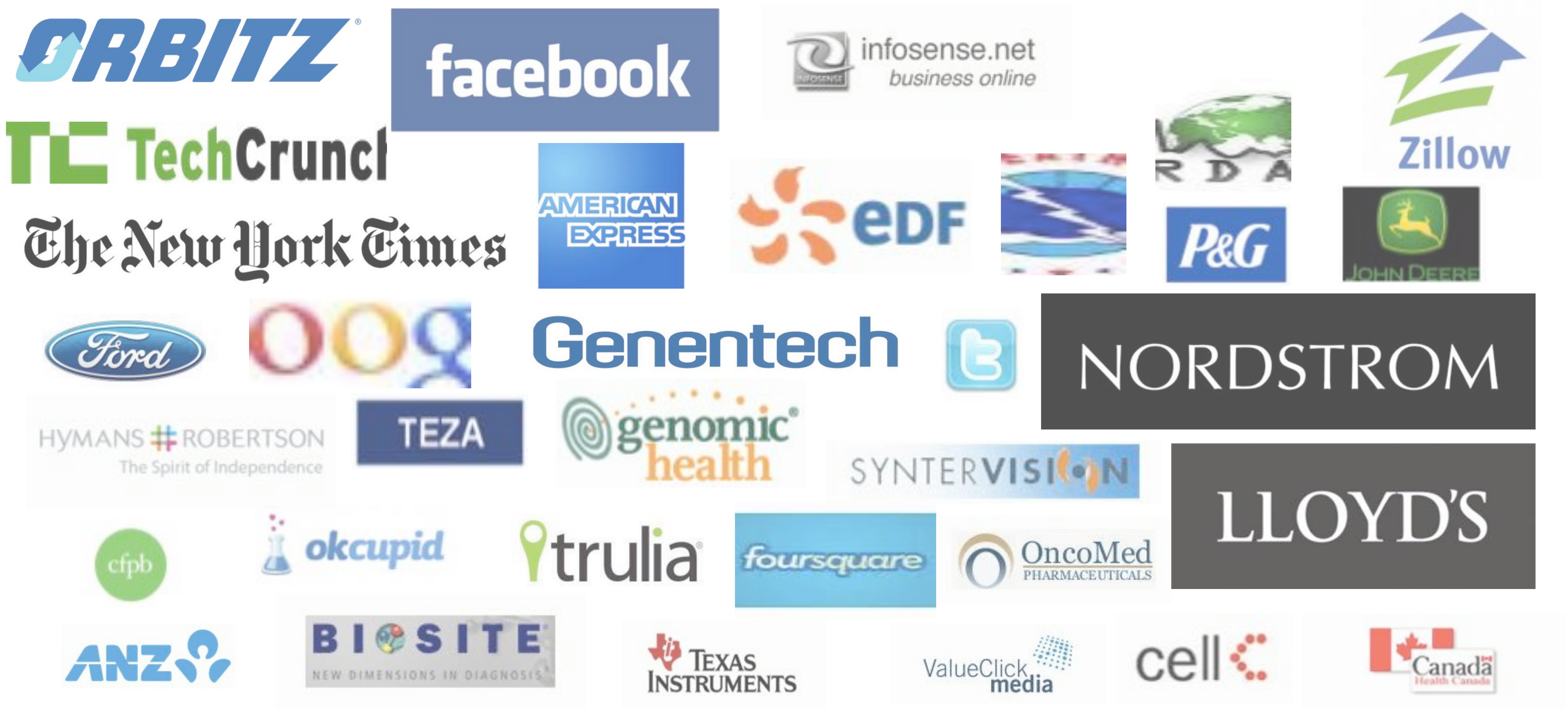
THE TWO STATES OF EVERY PROGRAMMER
I AM A GOD.
I HAVE NO IDEA WHAT I'M DOING.

Data formats R can read

- Can work with many types of data

Microsoft® SQL Server®
Excel
unstructured information data
{JSON}
XML
<HTML>
sas®

Companies that use R



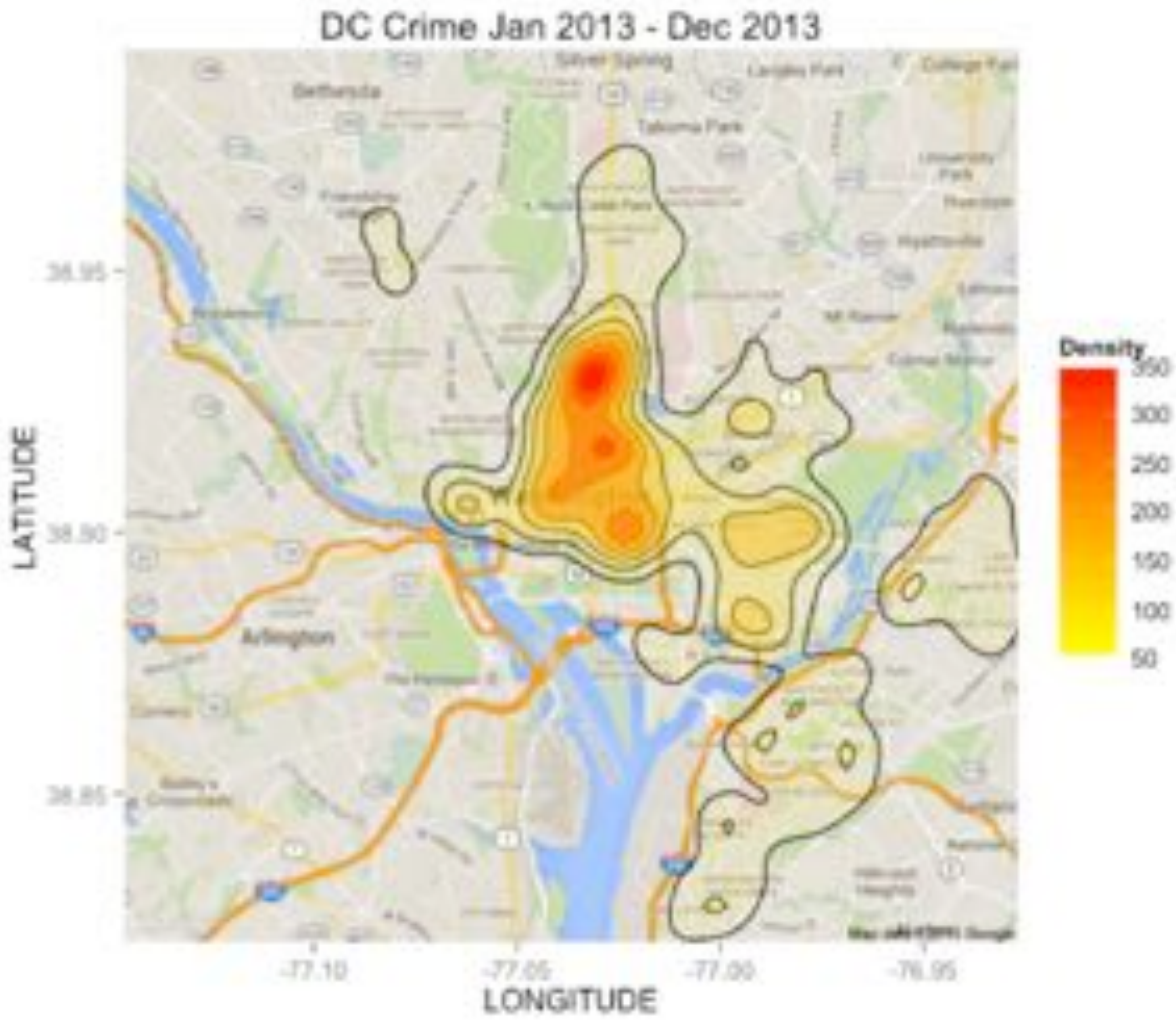
R vs. Excel

	R	Excel
Data capacity	R can read files as big as several gigabytes and trillions of data points; only limitation is your RAM	Excel can't read more than 1,048,576 rows and 16,384 columns (2011 version), files over ~300 megabytes can be very slow to work with
Customization	Can create custom visualizations through code, very flexible	Drop down menus limit ability to manipulate charts and graphs
Analyzing data	Powerful, pre-built packages that speed up work flow	Less flexible built-in analytic abilities that can be augmented by macros
Modeling	Data analysis and statistical models	Complex financial and accounting models
Seeing data	Built-in spreadsheet viewer	Easy to use spreadsheet interface
Usability	Direct commands similar to Excel "if-statements"	Keyboard shortcuts and slower point-and-click functionality

Visualizations in R

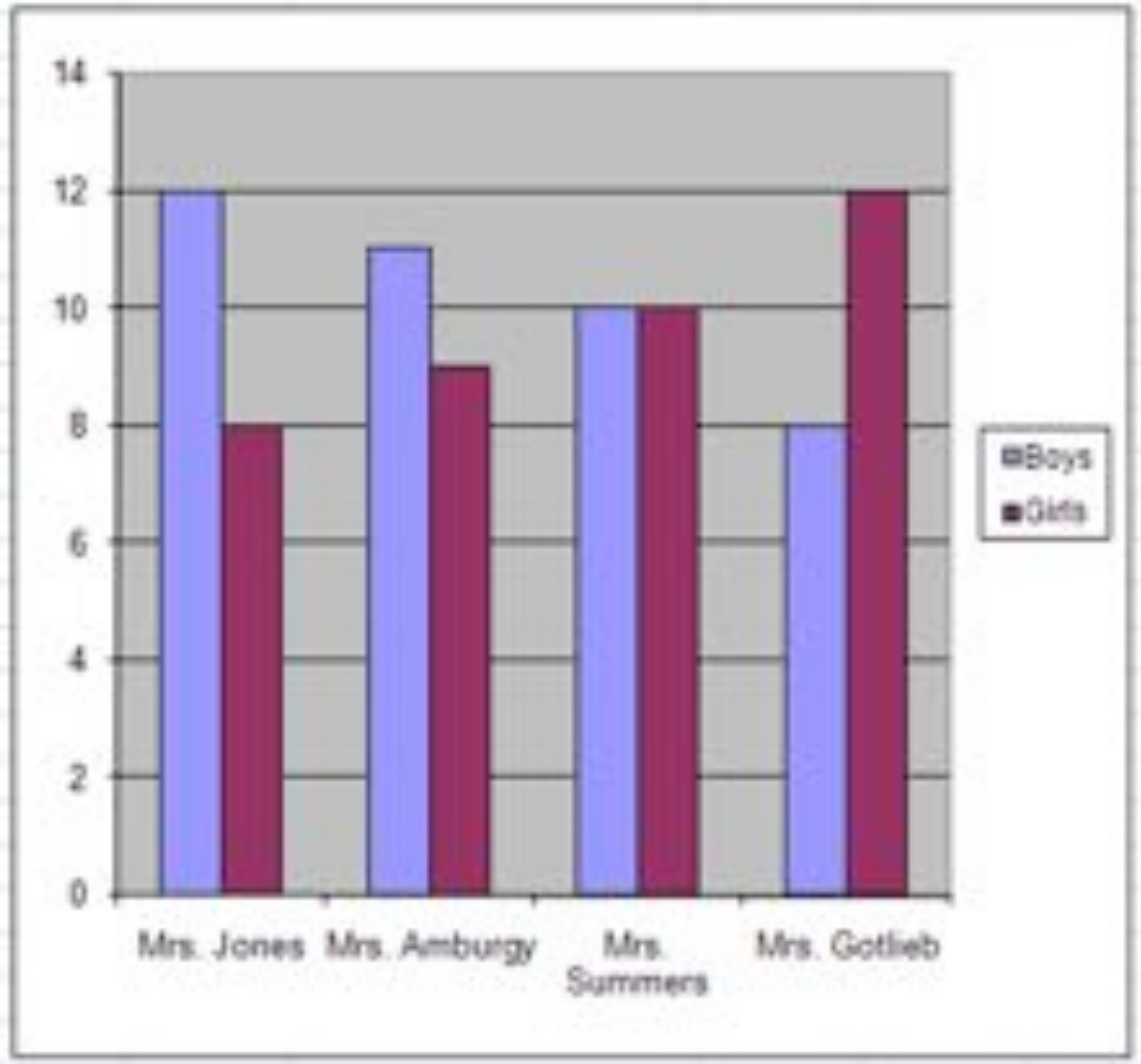
R

Simple customizable code: flexible



Excel

Drag and drop: rigid

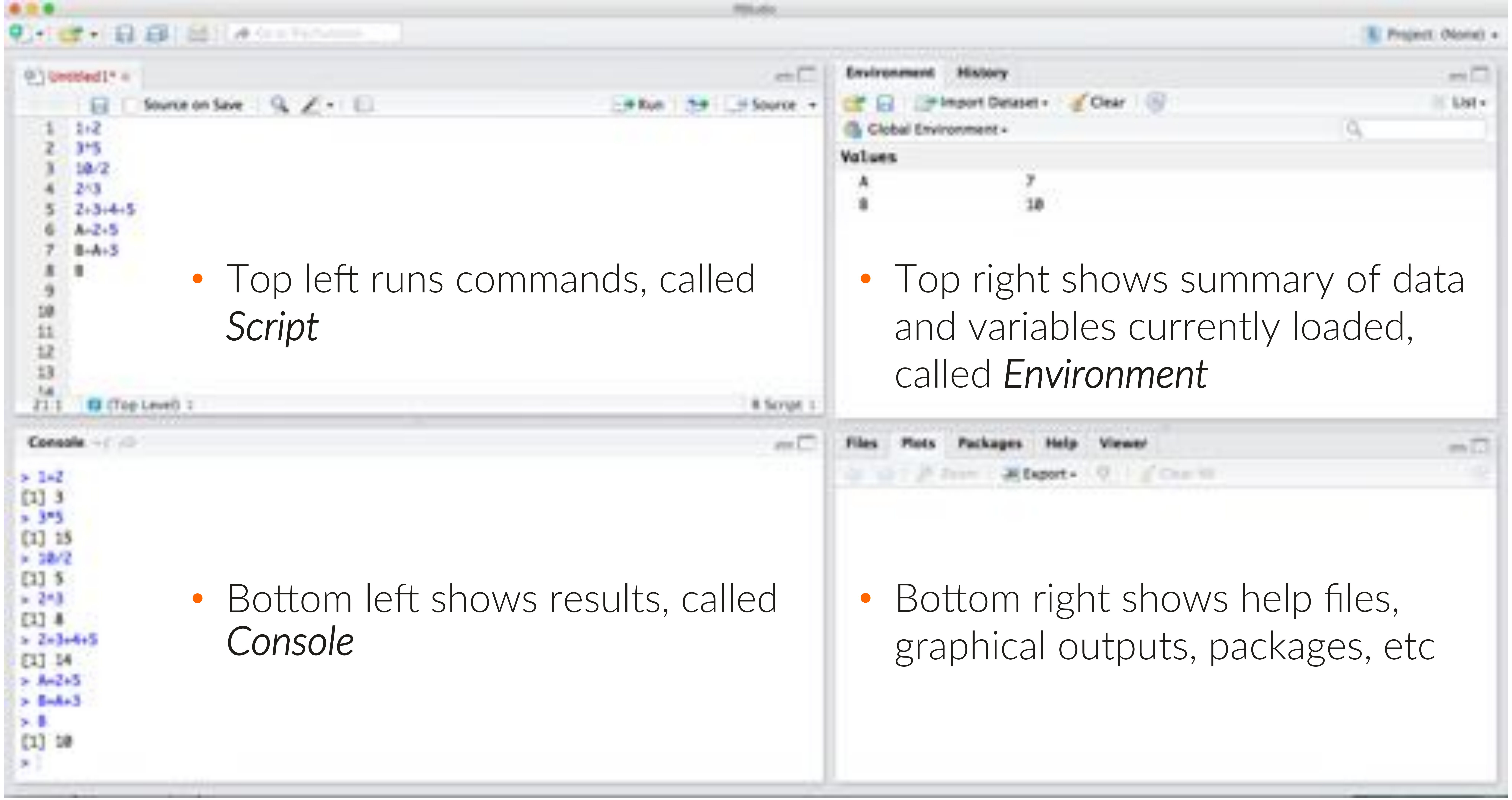


R vs. Python

- R has more convenient statistical packages to analyze data than Python
  - More than any other software tool, over 6,500 as of April 2015
- R is easier to learn for non-programmers than Python, less code is required to perform tasks
- Python is used by many data scientists to build data products (they also tend to be computer scientists)
- Python can be easier to integrate into web applications

Source: http://cran.r-project.org/web/packages/

RStudio overview



```
1+2
3*5
10/2
2^3
2+3+4+5
A=2+5
B=A+3
B
```

Values

A	7
B	10

```
> 1+2
[1] 3
> 3*5
[1] 15
> 10/2
[1] 5
> 2^3
[1] 8
> 2+3+4+5
[1] 14
> A=2+5
> B=A+3
> B
[1] 10
>
```

- Top left runs commands, called *Script*
- Top right shows summary of data and variables currently loaded, called *Environment*
- Bottom left shows results, called *Console*
- Bottom right shows help files, graphical outputs, packages, etc

Working with R: Comments

- Hashmarks are used to add comments and annotate your code

Script

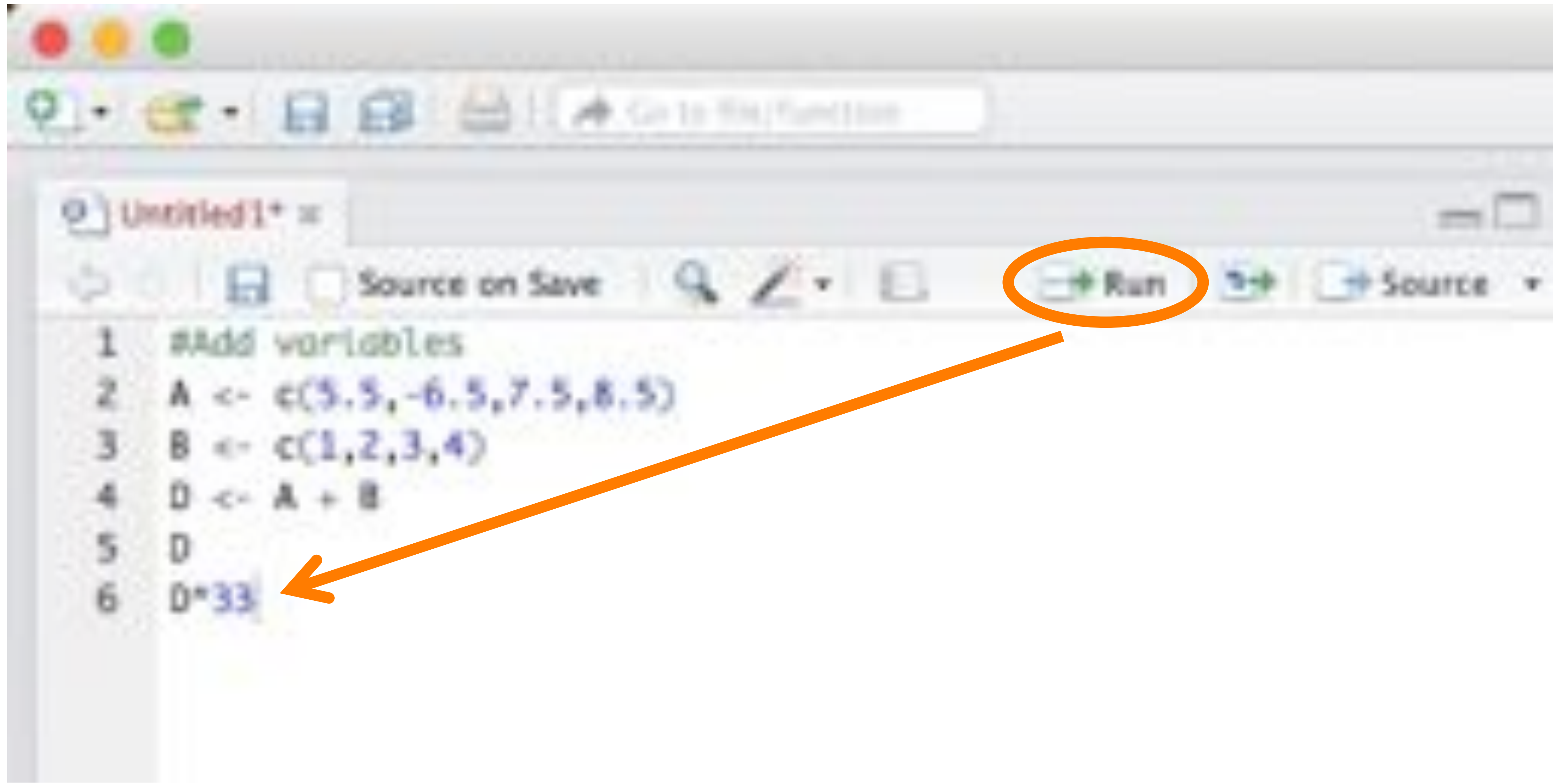
```
# Comments need to start with a hashmark, but don’t need to end with one

# Hashmarks show up in green and are included to explain your code
```

- It’s good practice to annotate your code
  - You can go back later and understand what you were doing

Executing commands in R

- Code is executed when you press “Run” in the top right hand corner of the script window
- R runs the line of code where your cursor is located
- You can also highlight multiple lines to run at once

Note: R is case sensitive



```
#Add variables
A <- c(5.5,-6.5,7.5,8.5)
B <- c(1,2,3,4)
D <- A + B
D
D*33
```

Outline

- What is data science?
- A data scientist's approach
- Introduction to R
  - Calculations in R
  - Reading data into R
  - Manipulating data in R
- Visualization in R
  - Basic plotting

Working with R: variables

- A series of numbers (think columns in Excel) can be defined using the arrow (**<-**) or equals (**=**) sign

Script

```
# Define variables with arrow
A <- c(5.5, -6.5, 7.5, 8.5)
B <- c(1, 2, 3, 4)
```

or

Script

```
# Define variables with equals sign
A = c(5.5, -6.5, 7.5, 8.5)
B = c(1, 2, 3, 4)
```

- The command `c( )` stands for “concatenate” (join) a series of numbers

Basic operations in R

Adding

- Just use + sign

Script

```
# Add variables
 A = c(5.5, -6.5, 7.5, 8.5)
 B = c(1, 2, 3, 4)
 D = A + B
 D
```

Console

```
> D
 [1] 6.5  -4.5  10.5  12.5
```

Multiplying

- Just use * sign

Script

```
# Multiply variables
 E = D*33
 E
```

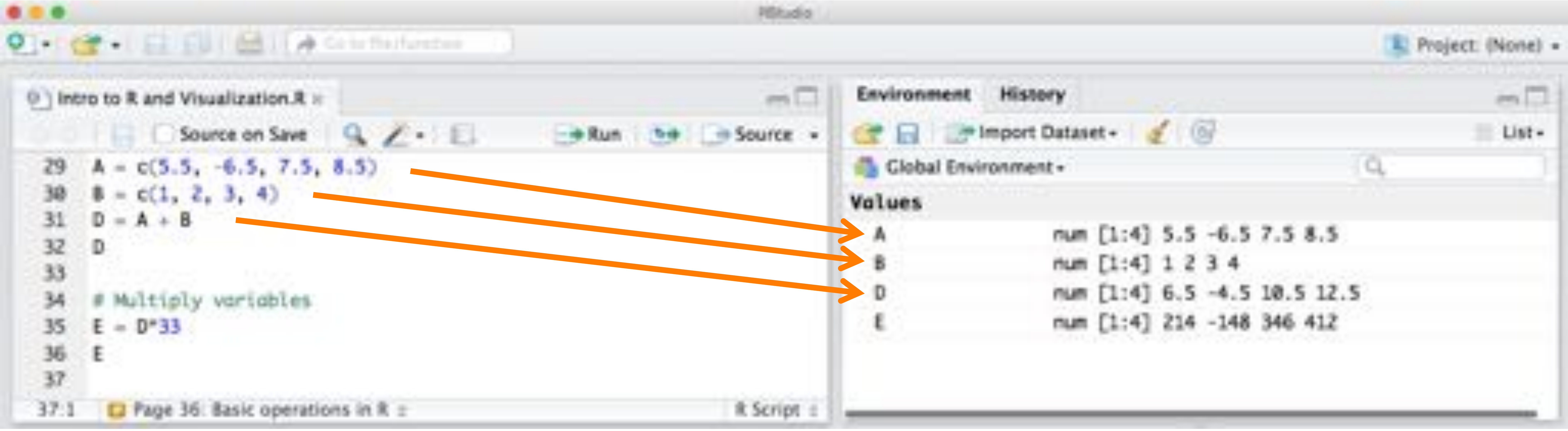
Console

```
> E
 [1] 214.5  -148.5  346.5  412.5
```

Enter formulas in top left window (script)
Output is shown in bottom left window (console)

Working with R: variables

- When a variable is named (instantiated), R stores it in its “environment” and can use it for subsequent operations



```
A = c(5.5, -6.5, 7.5, 8.5)
B = c(1, 2, 3, 4)
D = A + B
D

# Multiply variables
E = D*33
E
```

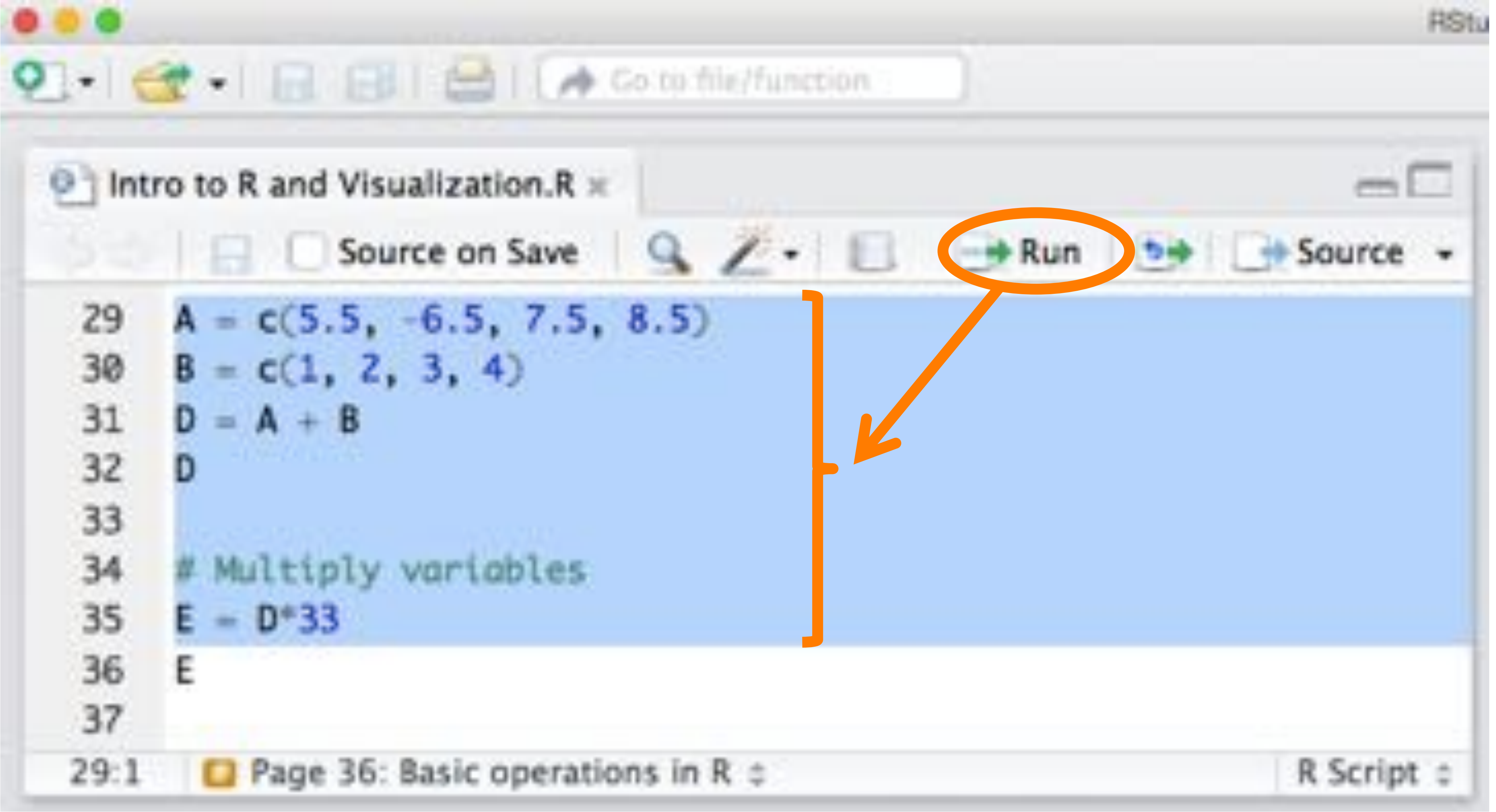
```
Values
A                num [1:4] 5.5 -6.5 7.5 8.5
B                num [1:4] 1 2 3 4
D                num [1:4] 6.5 -4.5 10.5 12.5
E                num [1:4] 214 -148 346 412
```

R can run several lines of code

- You can highlight several lines of code and press “Run” to execute all of them

- Highlighting can be done either with the mouse or by holding “Shift” and using the arrow keys

- You can execute a command by pressing “Ctrl” + “Enter” for PCs or “Command” + “Enter” for Macs


RStu
Go to file/function
Intro to R and Visualization.R
Source on Save
Run
Source
29:1
Page 36: Basic operations in R
R Script

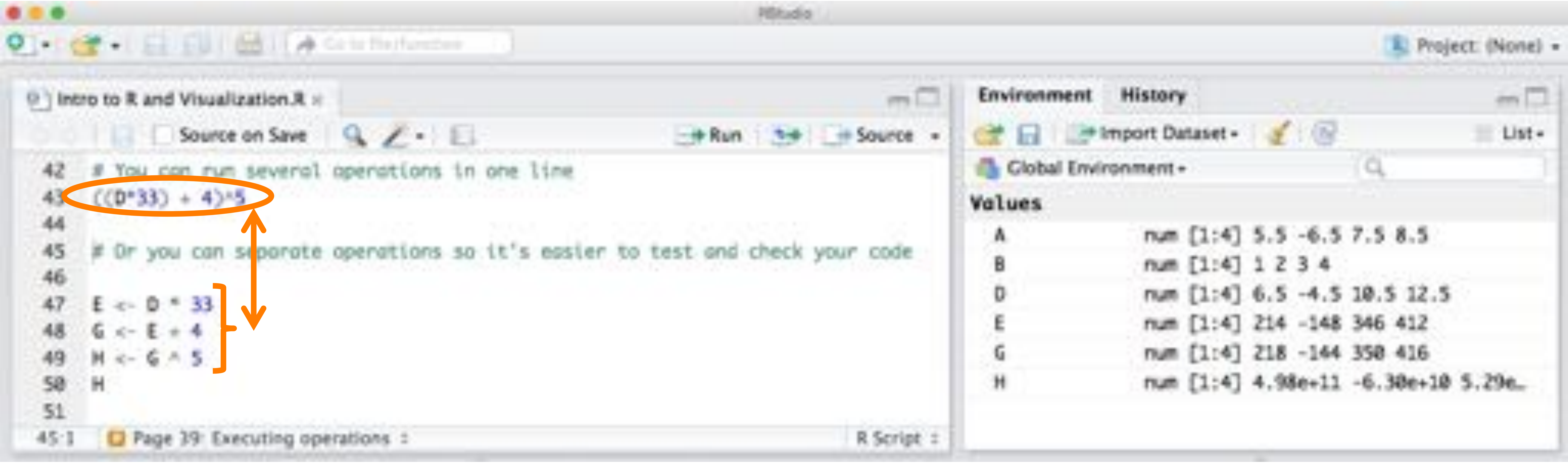
```
A = c(5.5, -6.5, 7.5, 8.5)
B = c(1, 2, 3, 4)
D = A + B
D

# Multiply variables
E = D*33
E
```

Troubleshooting: if you have trouble with this, try restarting R, restarting your computer, or reinstalling R

Executing operations

- You can run several operations in 1 line of code

- Or you can separate steps and instantiate new variables to check your code more easily



```
# You can run several operations in one line
((D*33) + 4)^5

# Or you can separate operations so it's easier to test and check your code

E <- D * 33
G <- E + 4
H <- G ^ 5
H

```

```
Values
  A          num [1:4] 5.5 -6.5 7.5 8.5
  B          num [1:4] 1 2 3 4
  D          num [1:4] 6.5 -4.5 10.5 12.5
  E          num [1:4] 214 -148 346 412
  G          num [1:4] 218 -144 350 416
  H          num [1:4] 4.98e+11 -6.30e+10 5.29e…
```

Outline

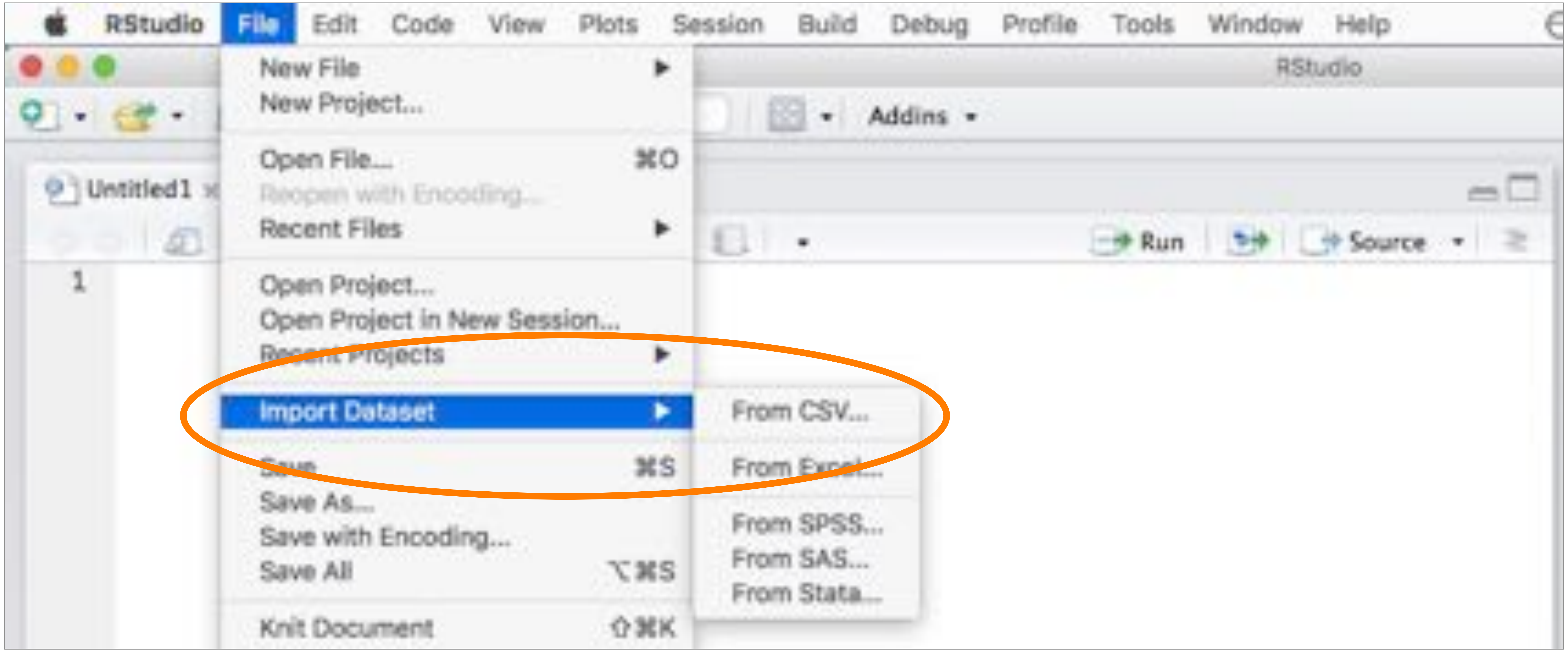
- What is data science?
- A data scientist's approach
- Introduction to R
  - Calculations in R
  - Reading data into R
  - Manipulating data in R
- Visualization in R
  - Basic plotting

A note about data

- Data can be found on a variety of sites on the internet
- Processing data stored in different formats is covered in a separate course
- For the purposes of this course, we will provide all the data sets already cleaned

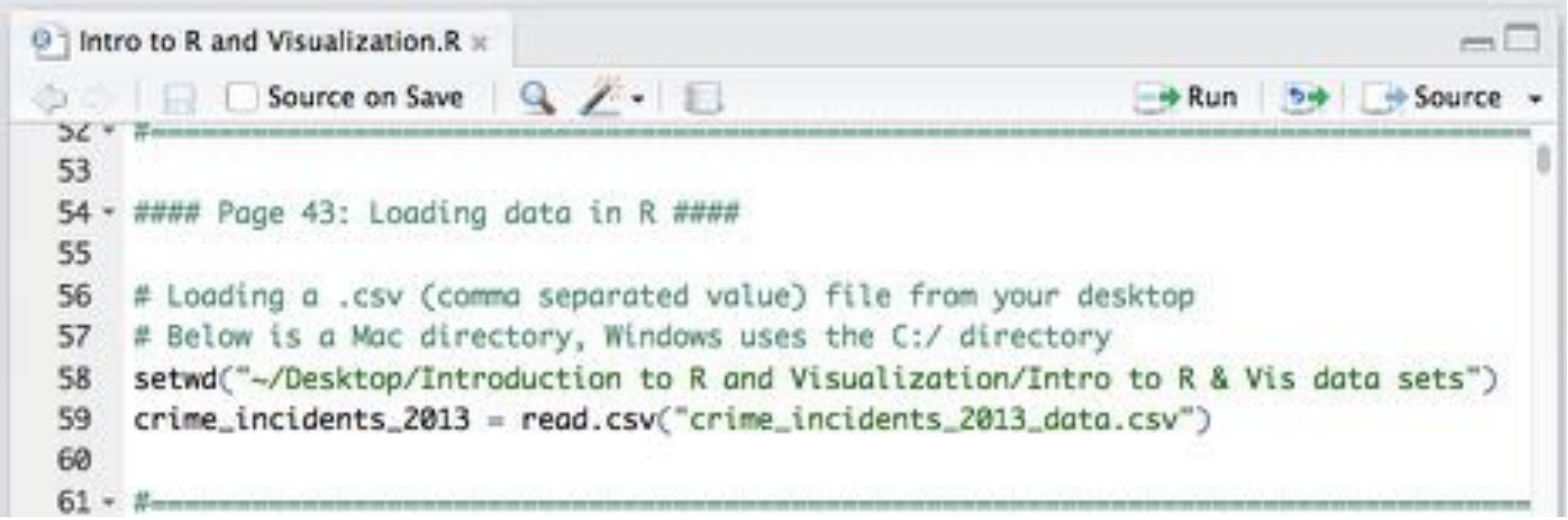
Loading data in R

- Loading data from your computer
  - Point and click



Loading data in R

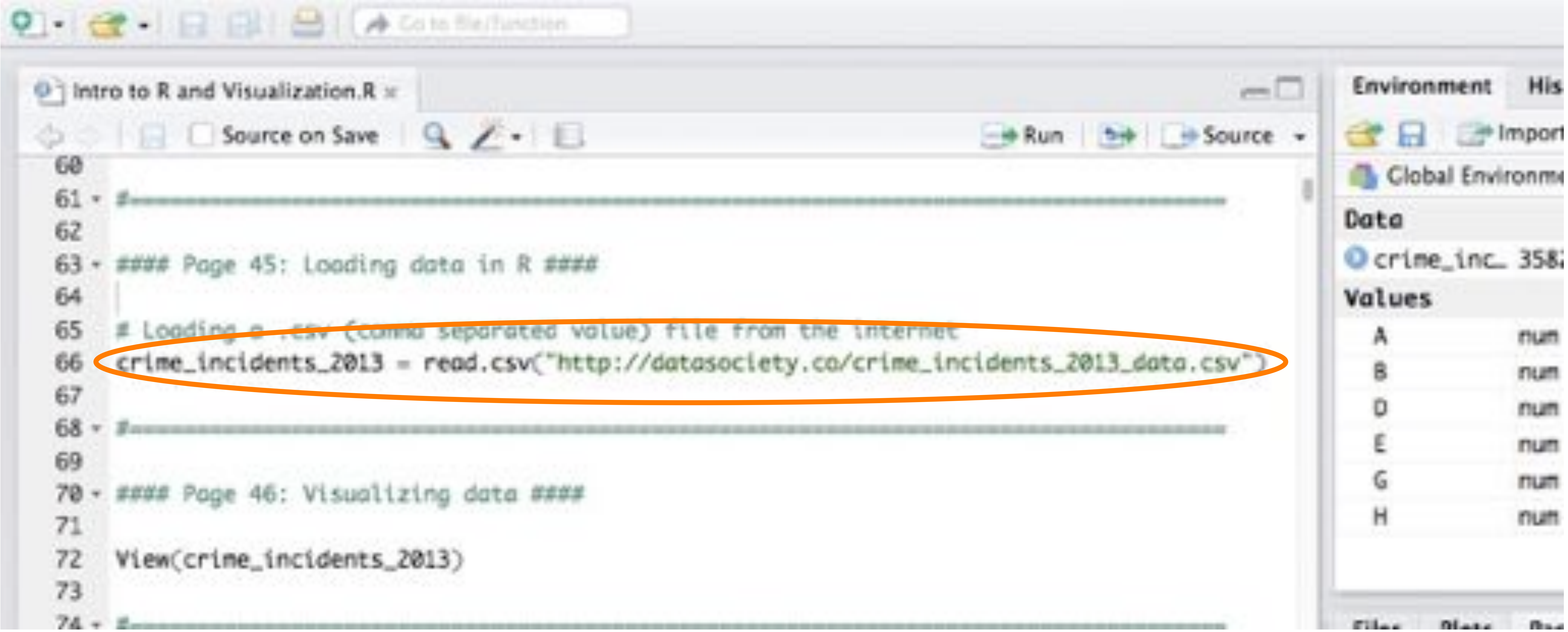
- Loading data from your computer
  - Enter code into script window
  - `crime_incidents_2013` is instantiated as the label of the data set



```
#-----------------------------------------------------------------------------

#### Page 43: Loading data in R ####

# Loading a .csv (comma separated value) file from your desktop
# Below is a Mac directory, Windows uses the C:/ directory
setwd("~/Desktop/Introduction to R and Visualization/Intro to R & Vis data sets")
crime_incidents_2013 = read.csv("crime_incidents_2013_data.csv")

#-----------------------------------------------------------------------------
```

Loading data in R

- Loading data from the internet
  - Enter code into script window
  - `crime_incidents_2013` is instantiated as the label of the data set

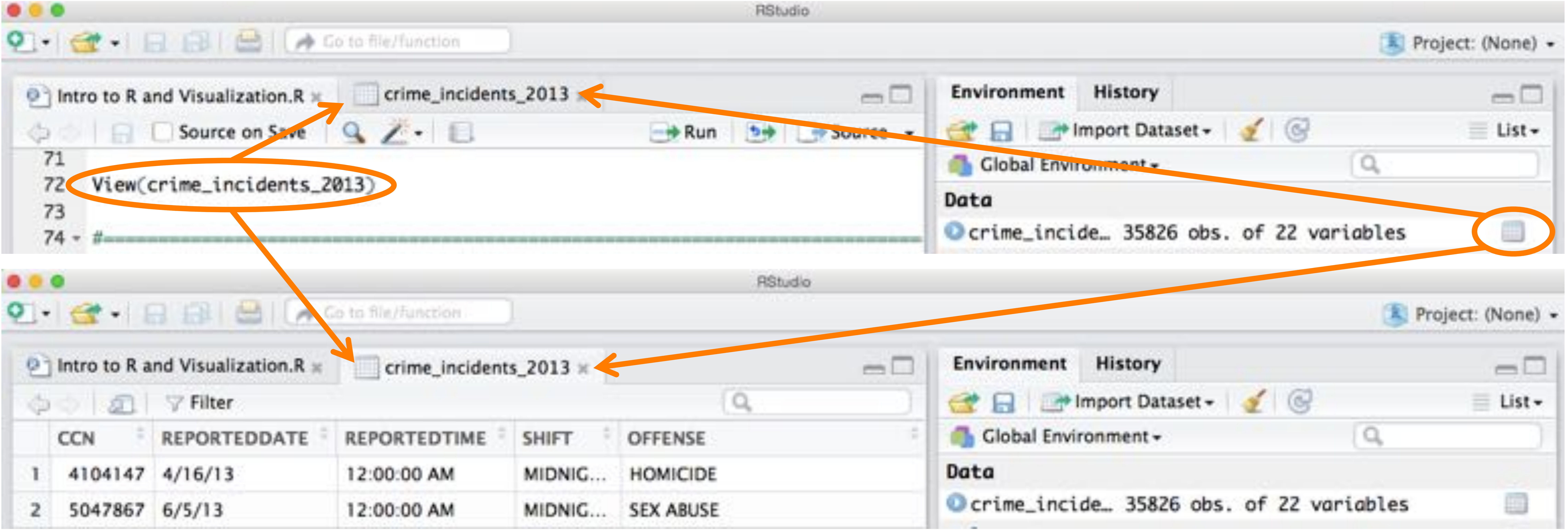


```
#----------------------------------------------------------------------------

#### Page 45: Loading data in R ####

# Loading a .csv (comma separated value) file from the internet
crime_incidents_2013 = read.csv("http://datasociety.co/crime_incidents_2013_data.csv")

#----------------------------------------------------------------------------

#### Page 46: Visualizing data ####

View(crime_incidents_2013)

#----------------------------------------------------------------------------
```

```
Data
crime_inc… 358
Values
A        num
B        num
D        num
E        num
G        num
H        num
```

Visualizing data

- Once data is loaded, you can see it as a spreadsheet by either:
  - Pressing the “spreadsheet” button in the top right window
  - Using the `View()` function in the script window



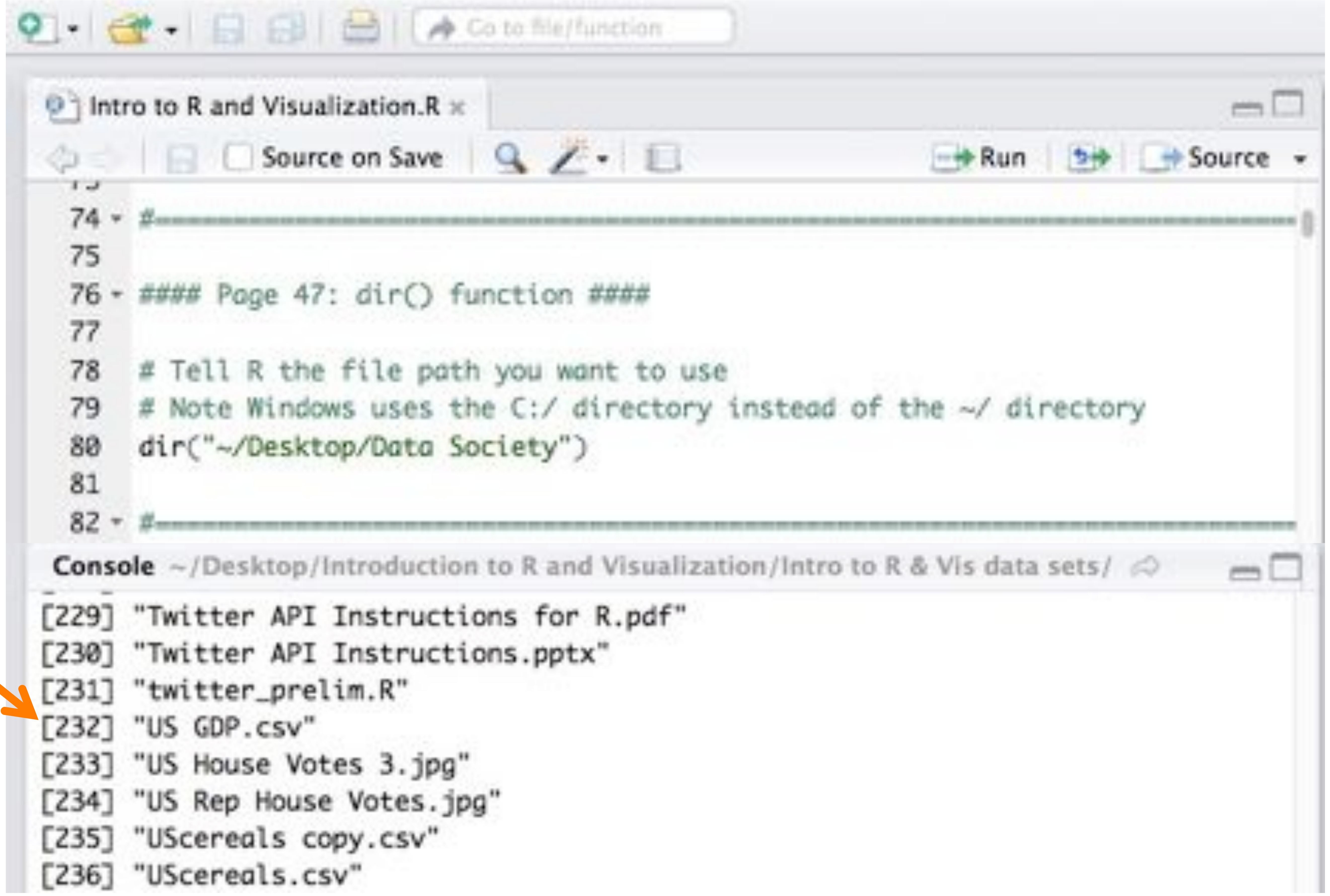
```
View(crime_incidents_2013)
#
```

crime_incide… 35826 obs. of 22 variables

	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE
1	4104147	4/16/13	12:00:00 AM	MIDNIG…	HOMICIDE
2	5047867	6/5/13	12:00:00 AM	MIDNIG…	SEX ABUSE

`dir()` function

- Lists all the files in a particular directory



Intro to R and Visualization.R
Source on Save
Run
Source
Go to file/function
Console ~/Desktop/Introduction to R and Visualization/Intro to R & Vis data sets/

```
#---------------------------------------------------------------------------

#### Page 47: dir() function ####

# Tell R the file path you want to use
# Note Windows uses the C:/ directory instead of the ~/ directory
dir("~/Desktop/Data Society")

#---------------------------------------------------------------------------
```

```
[229] "Twitter API Instructions for R.pdf"
[230] "Twitter API Instructions.pptx"
[231] "twitter_prelim.R"
[232] "US GDP.csv"
[233] "US House Votes 3.jpg"
[234] "US Rep House Votes.jpg"
[235] "UScereals copy.csv"
[236] "UScereals.csv"
```

Number of the file in the list

R can read many types of files

Script

```
read.csv("filename.csv")          # read Excel files converted to csv format
read.table("filename")            # reads a table from a text file
read.spss("filename.spss")        # reads SPSS files
read.dta("filename.dta")          # reads Stata files
read.ssd("filename.ssd")          # reads SAS files
read.octave("filename.octave")    # read Octave files
read.mtp("filename.mtp")          # read Minitab files
read.systat("filename.systat")    # read Systat files
read.JPEG("filename.jpg")         # read JPEG image files
```

Note: this requires us to install package called 'jpeg', we will cover packages later

Types of data used in R

Basic (units)

- Integers (-1, 5, 100)
- Numerics (2.54)
- Characters (“Hello”)
- Logicals (TRUE)
- Factors (A B C)

Composites

- Vectors
- Matrices (arrays)
- Lists
- Data frames

Basics make up composites

Note: R works only with types of data it “understands.” When loading new data, you need to tell R what kind of data you are loading

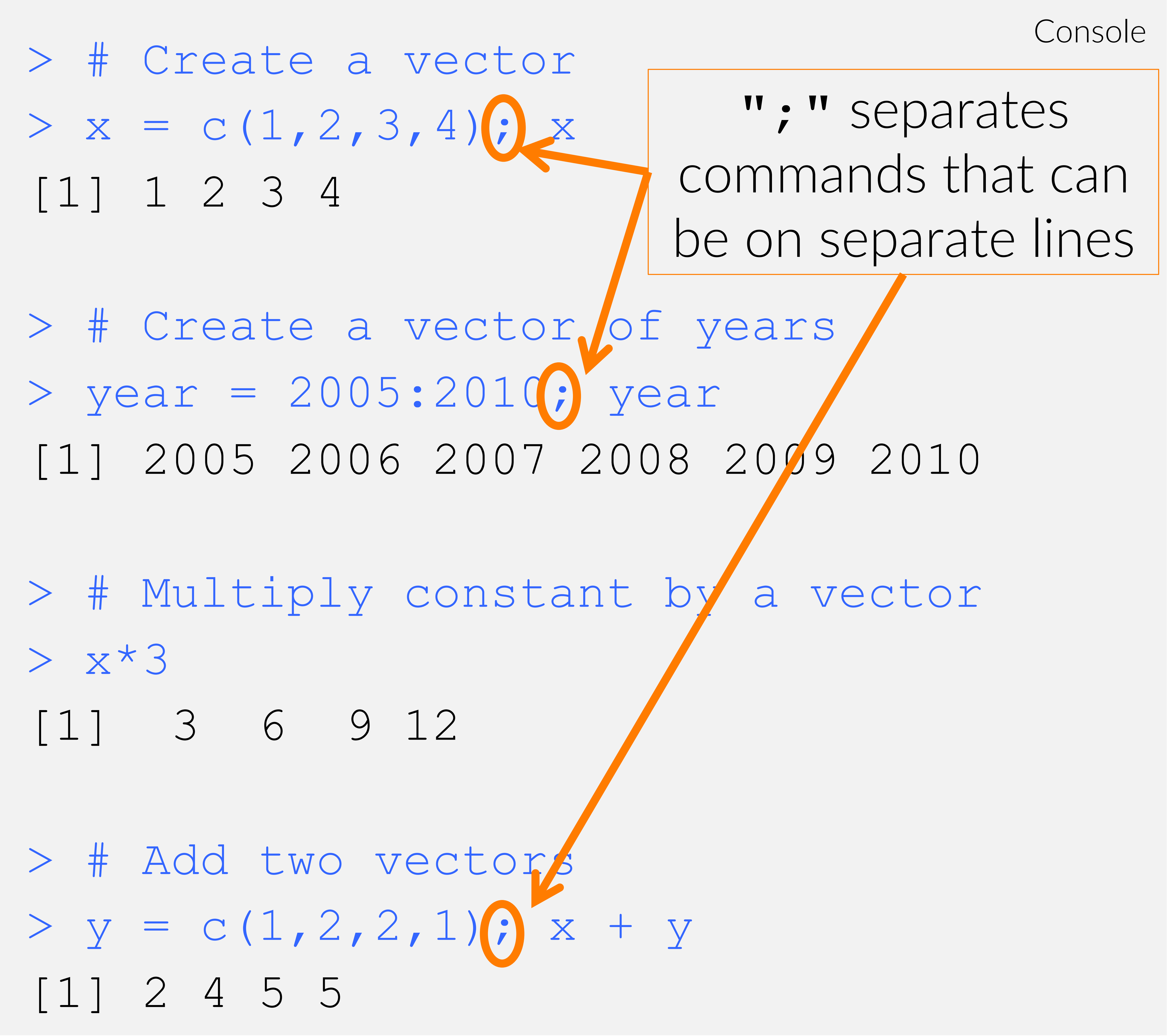
Why do data types matter?

- Different functions and commands in R can only work with certain types of data
- You may need to tell R that data from a spreadsheet is either a vector a list or another type of data before you can perform analysis on it
- Note that a spreadsheet that only includes numbers can be called either a list or a matrix
- You should tell R how to interpret the data based on what analysis you'd like to perform

Vectors in R

Vectors allow you to automatically sort text documents, images, etc.

- A vector is a collection of elements of the same type (a column with either all numbers or all letters)
- R reads data as vectors
- Vectors allow you to manipulate a lot of data with a single command
- Operation between two vectors requires that vectors contain the same number of entries (same length)



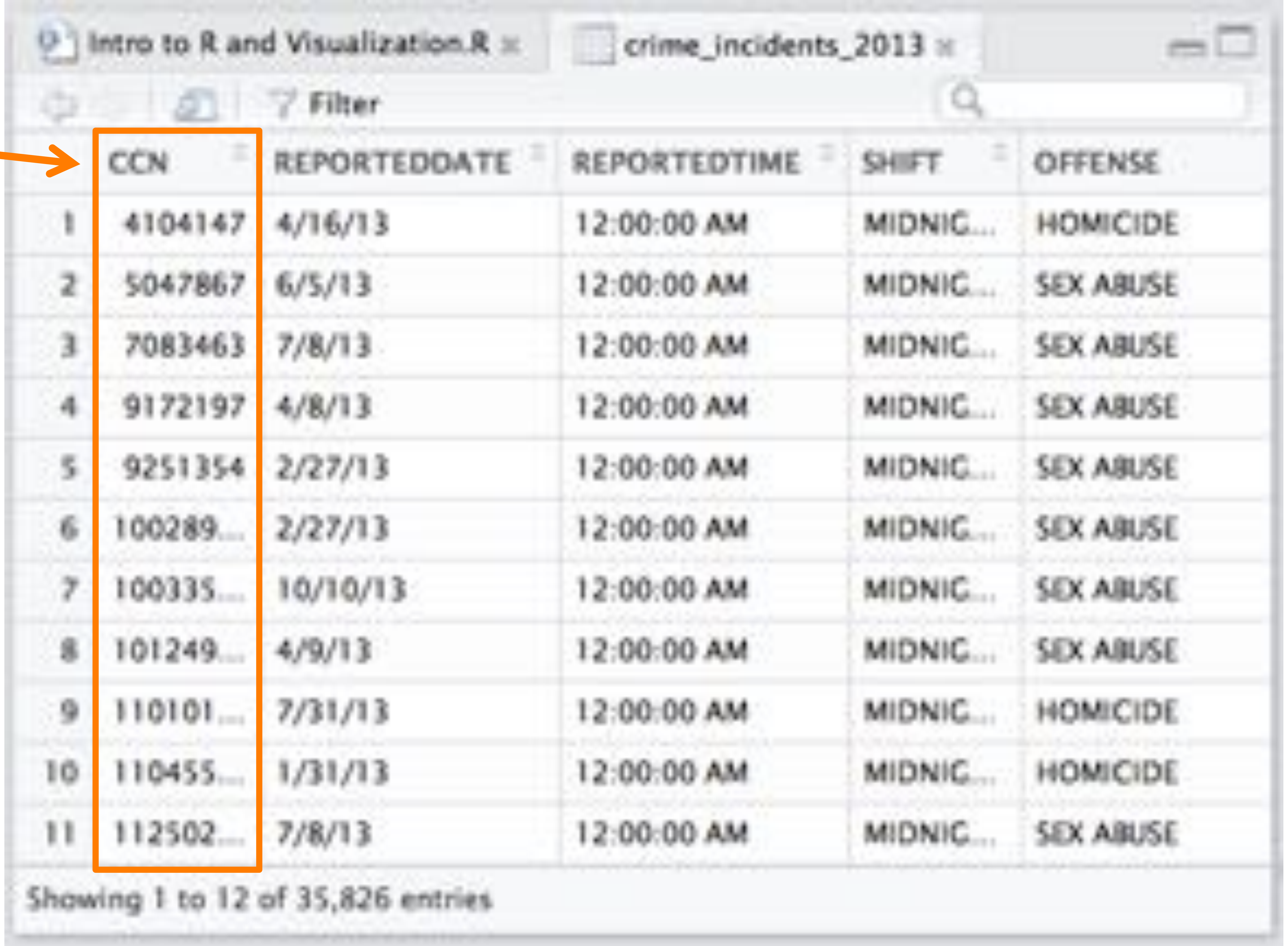
Console

```
> # Create a vector
> x = c(1,2,3,4); x
[1] 1 2 3 4

> # Create a vector of years
> year = 2005:2010; year
[1] 2005 2006 2007 2008 2009 2010

> # Multiply constant by a vector
> x*3
[1]  3  6  9 12

> # Add two vectors
> y = c(1,2,2,1); x + y
[1] 2 4 5 5
```

";" separates commands that can be on separate lines

Vectors in R

A vector



	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
6	100289...	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
7	100335...	10/10/13	12:00:00 AM	MIDNIG...	SEX ABUSE
8	101249...	4/9/13	12:00:00 AM	MIDNIG...	SEX ABUSE
9	110101...	7/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
10	110455...	1/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
11	112502...	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE

Showing 1 to 12 of 35,826 entries

Matrices in R

Matrices allow us to work with several columns of data at once

- A matrix is one or more vectors stacked next to each other
- Matrices can have row and column names, which can be determined and/or assigned (i.e. a customer list) by `dimnames, rownames` or `colnames` functions
- A matrix is a table where all the data is of the same type (i.e. numbers or letters)

Console

```
> # Create a matrix
> mat = matrix(1:6, nrow = 3, ncol = 2)
> mat
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

Script

```
# Get matrix information
class(mat)        # "matrix"
is.vector(mat)    # FALSE
is.matrix(mat)    # TRUE
length(mat)       # 6
dim(mat)          # 3 2
```

Customizing matrices in R

Add names and adjust data order

- If you want to put in your data by row instead of by column, type `byrow = TRUE`

- You can add names for rows and columns by using the function `dimnames`

Console

```
> mat1 = matrix(1:6, nrow = 3, ncol = 2,
byrow = TRUE)
> mat1
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6

> mat2 = matrix(1:6, nrow = 3, ncol = 2,
byrow = TRUE, dimnames = list(c("Row1",
"Row2", "Row3"), c("Col1", "Col2")))
> mat2
     Col1 Col2
Row1    1    2
Row2    3    4
Row3    5    6
```

Lists in R

Lists allow you to work with different types of data mixed together

- A list is a vector with different types of elements (data)
- The elements of a list can be numeric vectors, character vectors, matrices and other lists
- List components are determined by $ signs

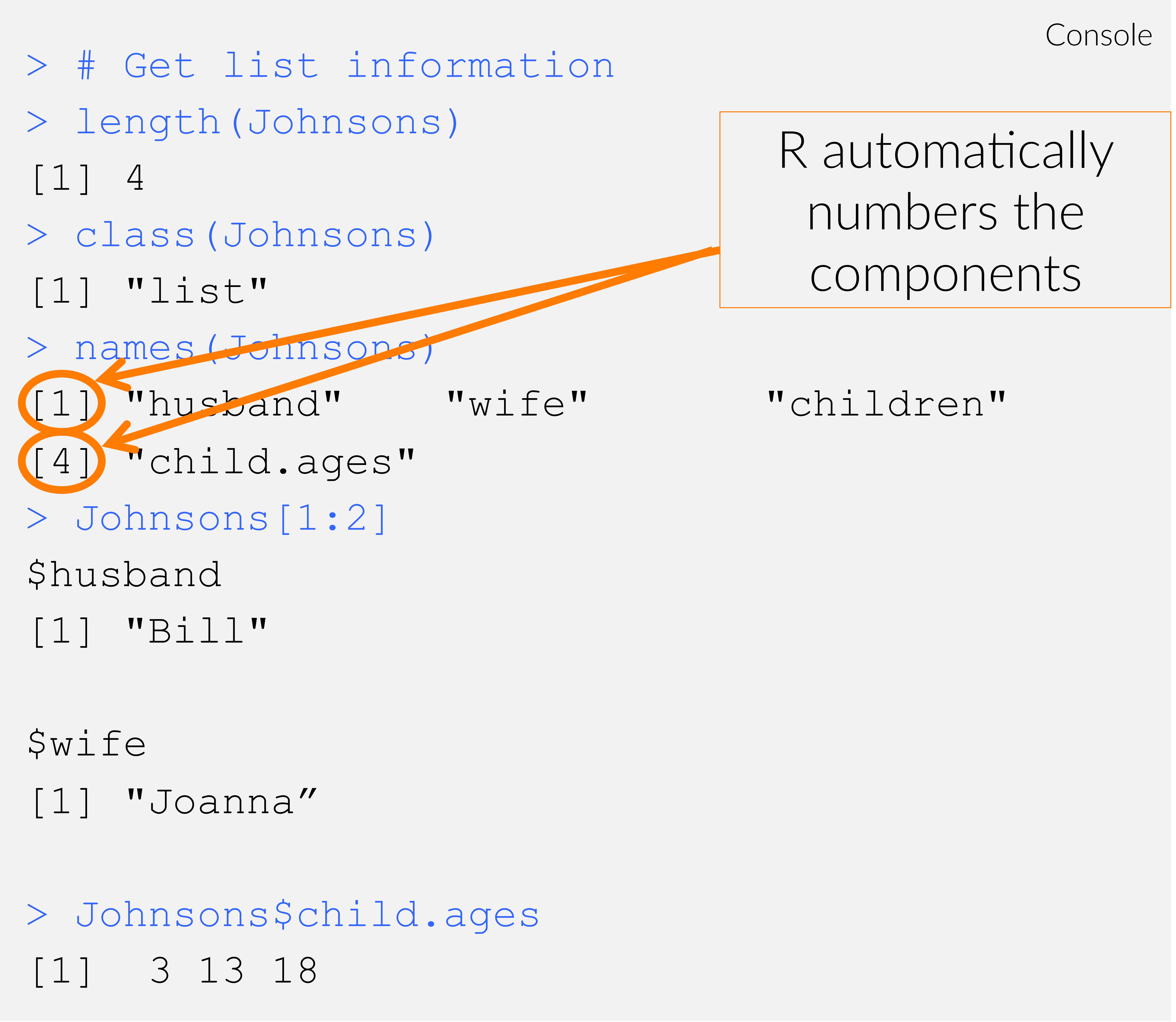
Console

```
> # Create a list
> Johnsons = list(husband = "Bill", wife =
"Joanna", children = TRUE, child.ages = c(3,
13, 18))
> Johnsons
$husband
[1] "Bill"

$wife
[1] "Joanna"

$children
[1] TRUE

$child.ages
[1]  3 13 18
```

Lists in R

You can pull out detailed information from lists

- `length("list")`: gives you the number of components
- `class("list")`: identifies the data type of the information
- `names("list")`: identifies the name of each component
- `"list"[1:2]`: identifies the specific data in those components
- `"list"$"component_name"`: identifies the specific data in that component

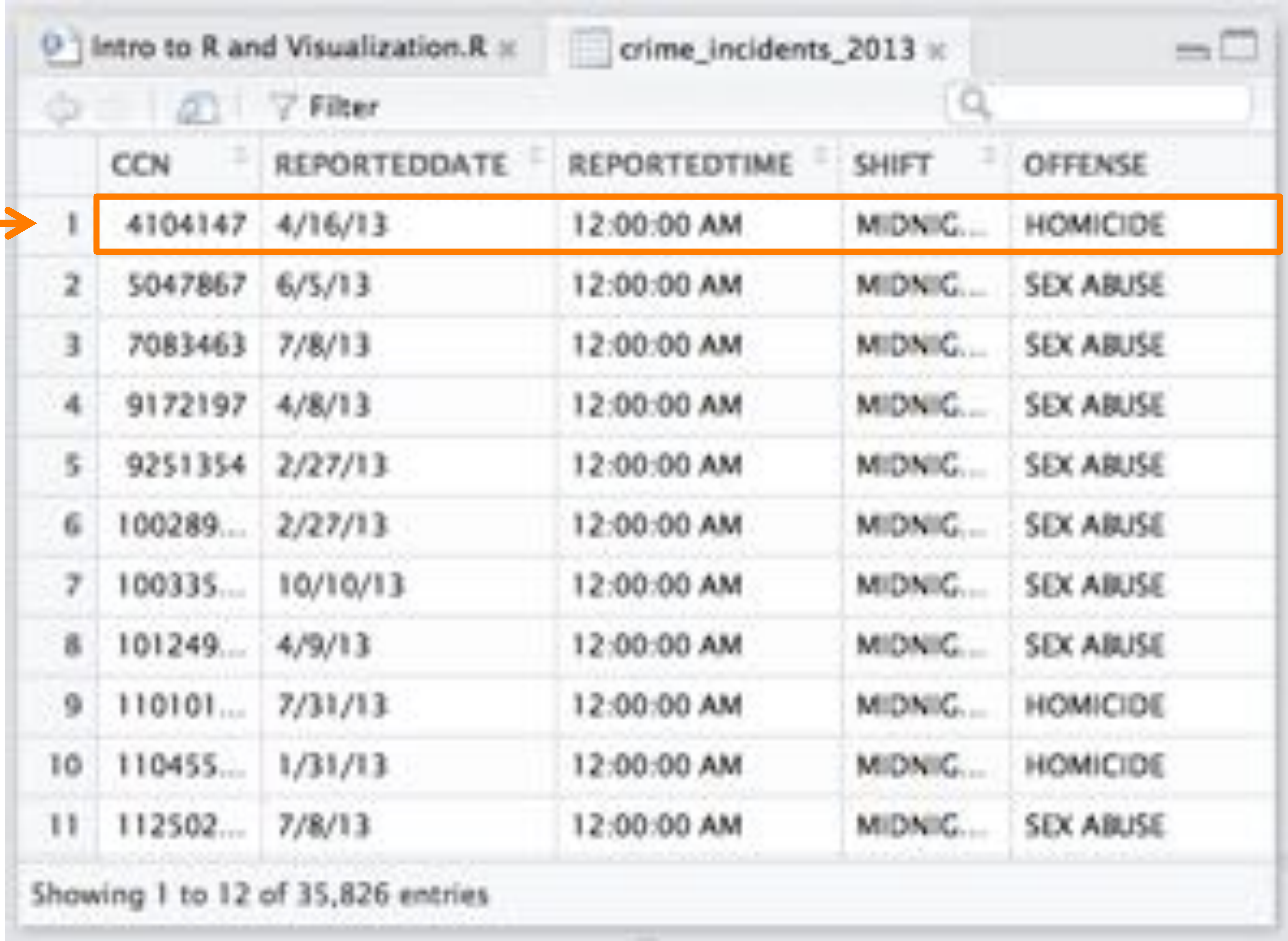


```
> # Get list information
> length(Johnsons)
[1] 4
> class(Johnsons)
[1] "list"
> names(Johnsons)
[1] "husband"    "wife"        "children"
[4] "child.ages"
> Johnsons[1:2]
$husband
[1] "Bill"

$wife
[1] "Joanna”

> Johnsons$child.ages
[1]  3 13 18
```

R automatically numbers the components

Lists in R

A list includes several data types and unstructured data



	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
6	100289...	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
7	100335...	10/10/13	12:00:00 AM	MIDNIG...	SEX ABUSE
8	101249...	4/9/13	12:00:00 AM	MIDNIG...	SEX ABUSE
9	110101...	7/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
10	110455...	1/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
11	112502...	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE

Showing 1 to 12 of 35,826 entries

Data frames in R

Data frames allow you to work with alpha-numeric and other data types

- Data frames are matrices with different data types
- Each column is a vector of the same length, types may differ
- The elements of a data frame can be numeric vectors, factor vectors, and logical vectors
- Consist of observations (rows), variables (columns)

Console

```
> # Setting up variables for data.frame
> first.name = c('Joe', 'Bob', 'Jill')
> last.name = c('Li', 'Dan', 'Smith')
> age = c(45, 20, 37)

> # Setting up data.frame
> df = data.frame(first.name,
> last.name, age)

> # Output
> df
  first.name last.name age
1        Joe        Li  45
2        Bob       Dan  20
3       Jill     Smith  37
```

Combining lists into data frames

You can combine lists and create data frames

- Note that these lists must have the same number of components

- Use function `rbind` (row bind) to match the categories and combine lists
  - When using `rbind`, the names of the columns must be the same

Note: the `rbind` function won't work with multiple data points within an element of a list

Script

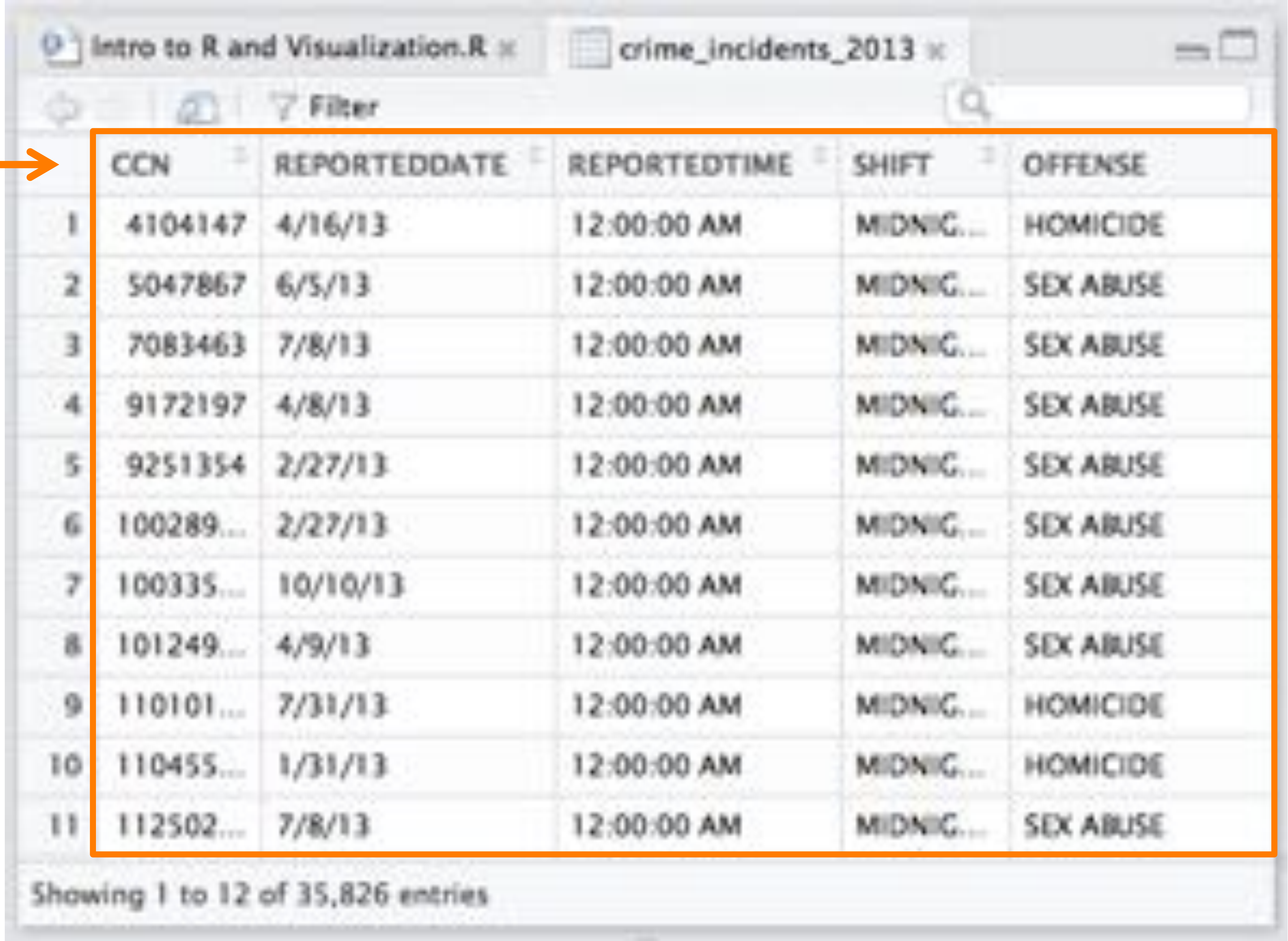
```
# Combining lists into a data frame

Simpsons = list(husband = "Homer", wife =
                "Marge", children = TRUE,
                num.kids = 3)
Belchers = list(husband = "Bob", wife =
                "Linda", children = TRUE,
                num.kids = 3)
rbind(Simpsons, Johnsons)
```

Console

```
> rbind(Simpsons, Belchers)
         husband wife    children num.kids
Simpsons "Homer" "Marge" TRUE     3
Belchers "Bob"   "Linda" TRUE     3
```

Data frames in R

A data frame includes several data types



	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
6	100289...	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
7	100335...	10/10/13	12:00:00 AM	MIDNIG...	SEX ABUSE
8	101249...	4/9/13	12:00:00 AM	MIDNIG...	SEX ABUSE
9	110101...	7/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
10	110455...	1/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
11	112502...	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE

Showing 1 to 12 of 35,826 entries

Creating data types in R

Composite data types	Instantiating (creating)	Appropriate use
Vector	`c( . . . )`	Store simple data types in a row or column
Matrix	`matrix(data, nrows, ncols)`	Store multiple rows and columns of a single data type
Data Frame	`data.frame(data matrix)`	Work with alphanumeric and other data types
List	`list(element_1,...,element_k)`	Work with different data types simultaneously

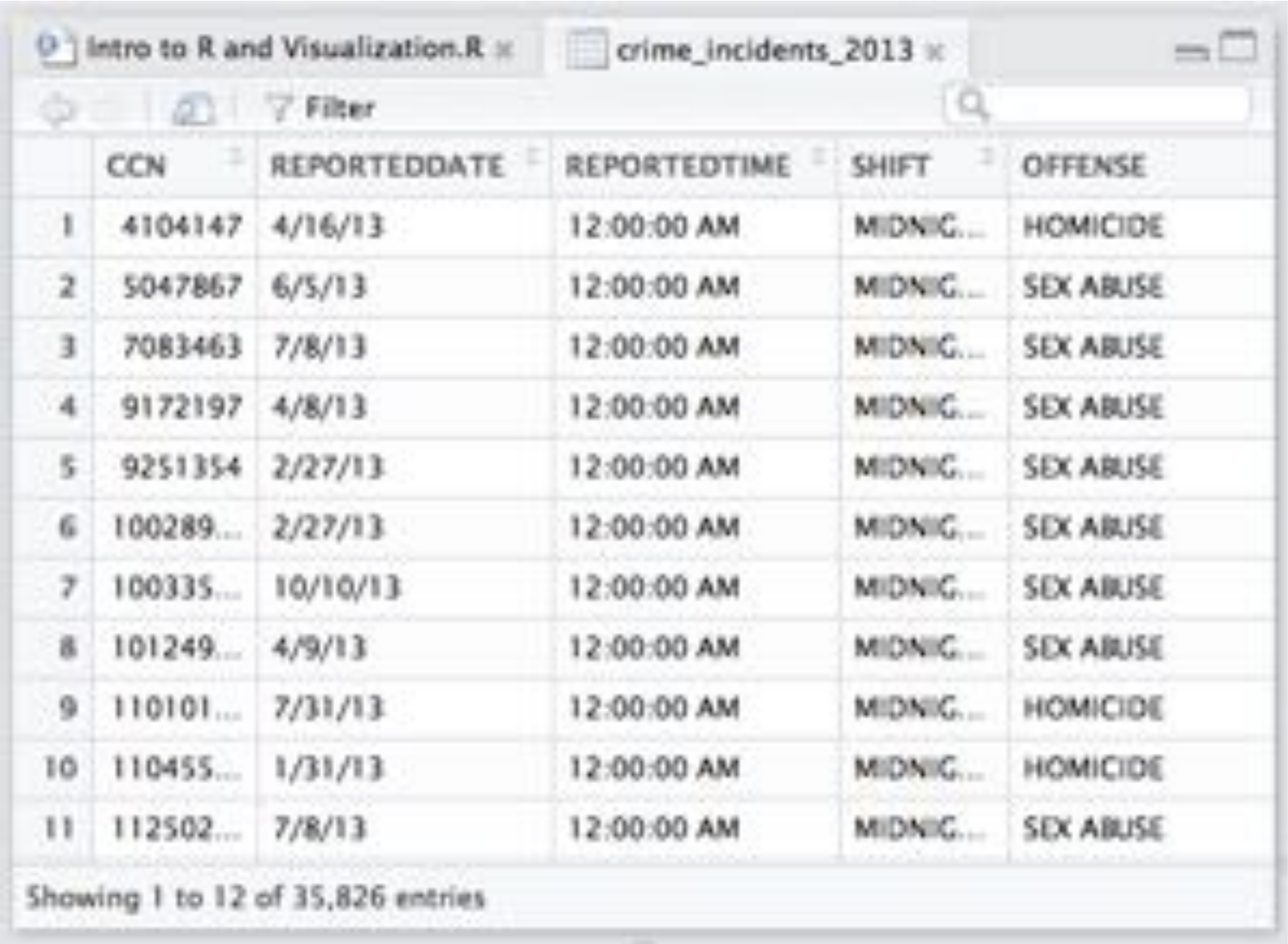
Outline

- What is data science?
- A data scientist's approach
- Introduction to R
  - Calculations in R
  - Reading data into R
  - Manipulating data in R
- Visualization in R
  - Basic plotting

DC crime data 2013

What does the data look like?

- Almost 36,000 rows, each row represents a crime
- Information includes date, time, offense, method, location, precinct, etc.
- Always take the time to look over your data – this habit will help you understand the information you are working with

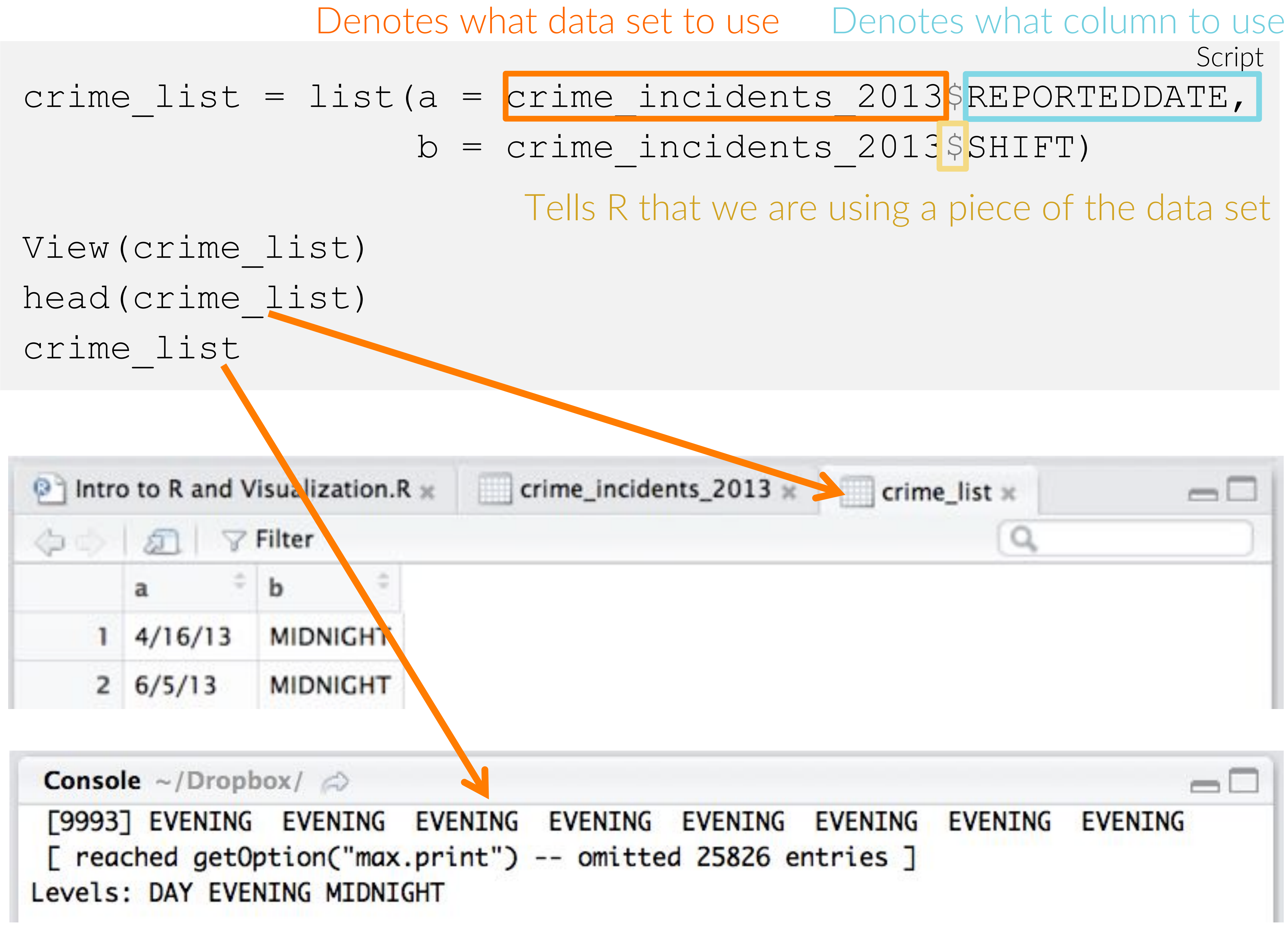


	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
6	100289...	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE
7	100335...	10/10/13	12:00:00 AM	MIDNIG...	SEX ABUSE
8	101249...	4/9/13	12:00:00 AM	MIDNIG...	SEX ABUSE
9	110101...	7/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
10	110455...	1/31/13	12:00:00 AM	MIDNIG...	HOMICIDE
11	112502...	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE

Showing 1 to 12 of 35,826 entries

Subsetting data

- You can separate out certain columns of the data by using a $ sign
- The format is `"data_set"$"name_of_column"`
- If the list you are creating has columns of different lengths, the **`View()`** function will not work
  - Use the `head()` function or just enter the new variable you created instead



Denotes what data set to use

Denotes what column to use

Script

```
crime_list = list(a = crime_incidents_2013$REPORTEDDATE,
                  b = crime_incidents_2013$SHIFT)
```

Tells R that we are using a piece of the data set

```
View(crime_list)
head(crime_list)
crime_list
```

	a	b
1	4/16/13	MIDNIGHT
2	6/5/13	MIDNIGHT

```
[9993] EVENING  EVENING  EVENING  EVENING  EVENING  EVENING  EVENING  EVENING
 [ reached getOption("max.print") -- omitted 25826 entries ]
Levels: DAY EVENING MIDNIGHT
```

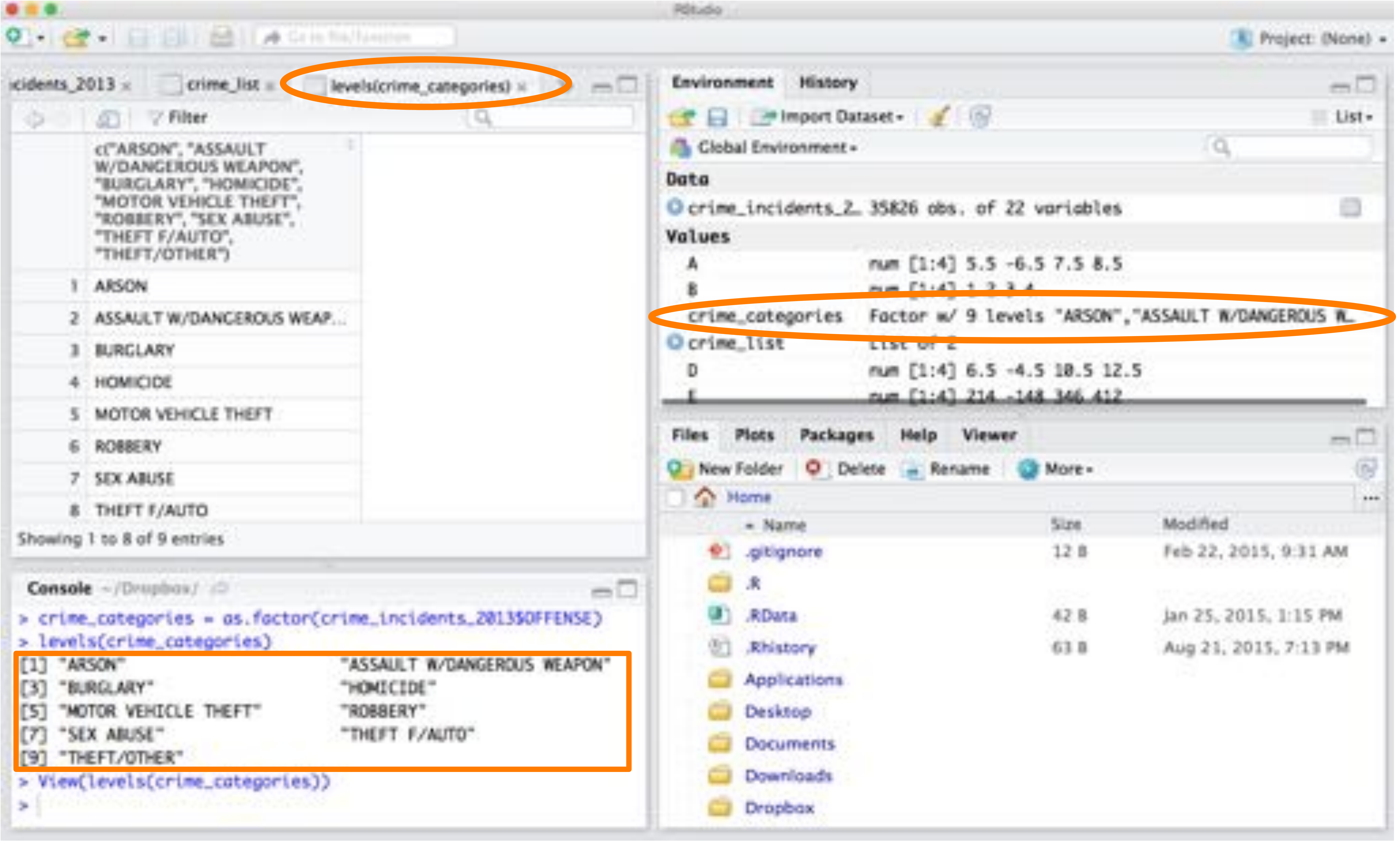
Factors in R

- A factor is a unique value in a data set
- For example, if five crimes are labeled as “arson, burglary, burglary, burglary, robbery”, there would be three factors: arson, burglary, and robbery
- Use the `as.factor()` function to tell R to read data as a factor

Script

```
# Set a category of our crime data set as a factor using as.factor()
crime_categories = as.factor(crime_incidents_2013$OFFENSE)

# Visualize the unique values in the OFFENSE category of the crime data set
levels(crime_categories)
View(levels(crime_categories))
```

Factors in R



	c("ARSON", "ASSAULT W/DANGEROUS WEAPON", "BURGLARY", "HOMICIDE", "MOTOR VEHICLE THEFT", "ROBBERY", "SEX ABUSE", "THEFT F/AUTO", "THEFT/OTHER")
1	ARSON
2	ASSAULT W/DANGEROUS WEAP...
3	BURGLARY
4	HOMICIDE
5	MOTOR VEHICLE THEFT
6	ROBBERY
7	SEX ABUSE
8	THEFT F/AUTO

Showing 1 to 8 of 9 entries

```
Data
crime_incidents_2… 35826 obs. of 22 variables
Values
  A                  num [1:4] 5.5 -6.5 7.5 8.5
  B                  num [1:4] 1 2 3 4
  crime_categories   Factor w/ 9 levels "ARSON","ASSAULT W/DANGEROUS W…
crime_list           List of 2
  D                  num [1:4] 6.5 -4.5 10.5 12.5
  E                  num [1:4] 214 -148 346 412
```

```
> crime_categories = as.factor(crime_incidents_2013$OFFENSE)
> levels(crime_categories)
[1] "ARSON"                       "ASSAULT W/DANGEROUS WEAPON"
[3] "BURGLARY"                    "HOMICIDE"
[5] "MOTOR VEHICLE THEFT"         "ROBBERY"
[7] "SEX ABUSE"                   "THEFT F/AUTO"
[9] "THEFT/OTHER"
> View(levels(crime_categories))
>
```

Name	Size	Modified
.gitignore	12 B	Feb 22, 2015, 9:31 AM
.R		
.RData	42 B	Jan 25, 2015, 1:15 PM
.Rhistory	63 B	Aug 21, 2015, 7:13 PM
Applications		
Desktop		
Documents		
Downloads		
Dropbox		

Searching for terms in your data

- The function `grep()` can help you check if your data set includes a given term

Script

```
# grep() only works on a data type called a character vector
# Recall that a vector is a single column
crime = as.character(crime_incidents_2013$METHOD)

# Create a new variable that includes the output of grep()
gun_rows = grep("GUN", crime)

# View gun_rows, this includes all the row numbers with the word "GUN"
# Recall that the View() function displays data frames so tell R that "gun_rows"
# is a data frame
View(as.data.frame(gun_rows))
```

Select the “METHOD” column

- You can search the entire data set by combining all the columns into one or searching each column individually

Searching for terms in your data

- How many rows include “GUN” as the method?

Console

```
> # Find the number of rows
> length(gun_rows)
[1] 2156
```

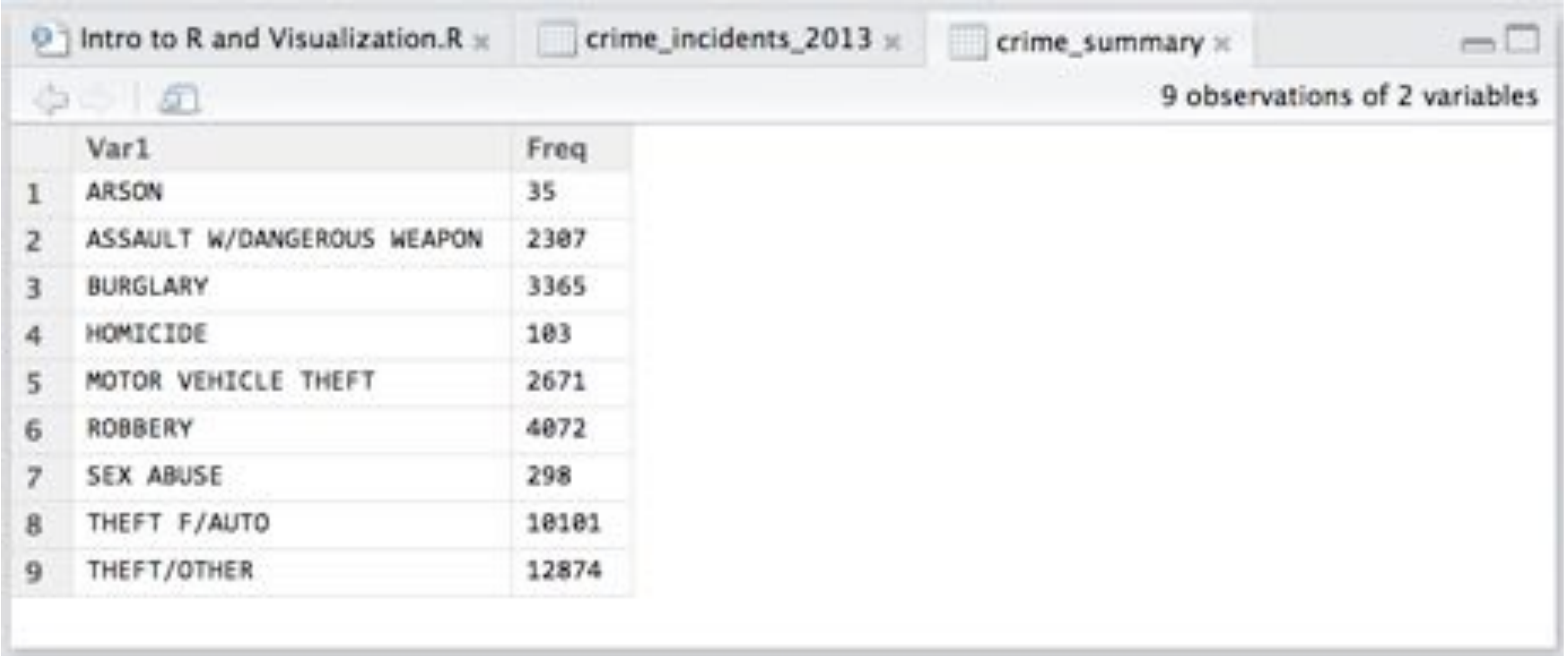
Summarizing your data

- The `table()` function is a convenient way to see unique categories in your data and determine the frequency of occurrence

Script

```
crime_summary = table(crime_incidents_2013$OFFENSE)

View(crime_summary)
```

Column to summarize



9 observations of 2 variables

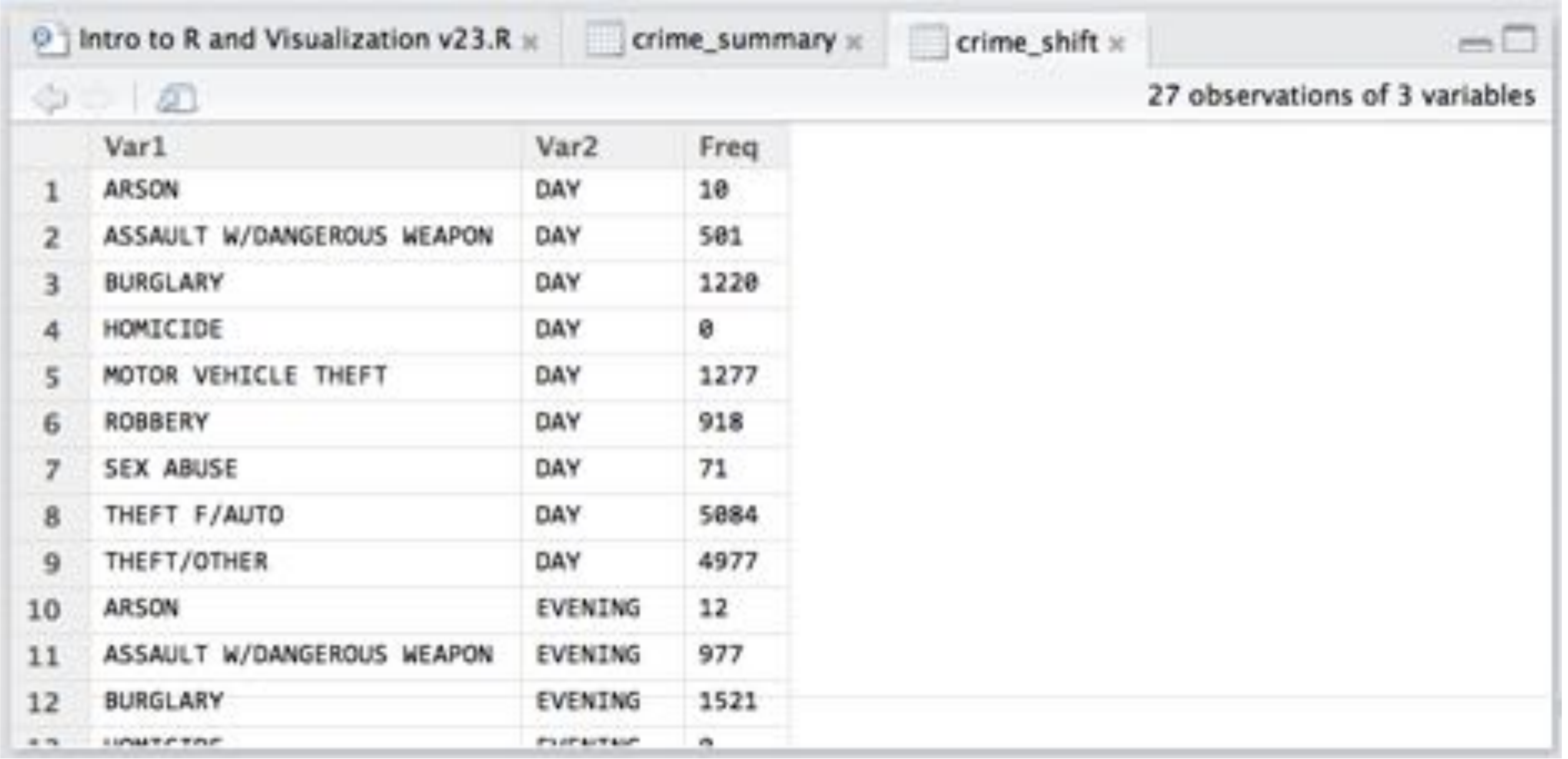
	Var1	Freq
1	ARSON	35
2	ASSAULT W/DANGEROUS WEAPON	2307
3	BURGLARY	3365
4	HOMICIDE	103
5	MOTOR VEHICLE THEFT	2671
6	ROBBERY	4072
7	SEX ABUSE	298
8	THEFT F/AUTO	10101
9	THEFT/OTHER	12874

Summarizing your data

- What if I want to see how crime is distributed over 24 hours?

Script

```
crime_time = table(crime_incidents_2013$OFFENSE,
                   crime_incidents_2013$SHIFT)

View(crime_time)
```

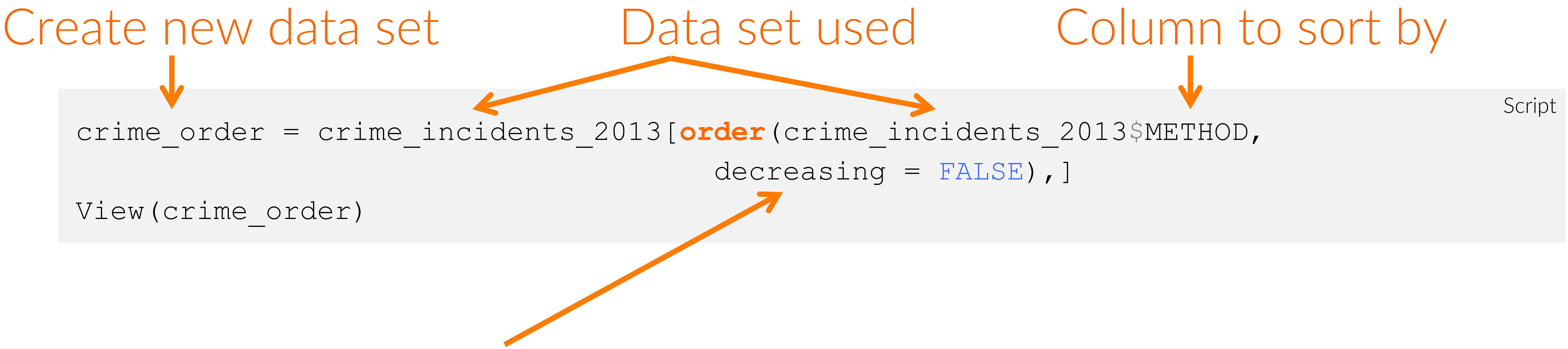


27 observations of 3 variables

	Var1	Var2	Freq
1	ARSON	DAY	10
2	ASSAULT W/DANGEROUS WEAPON	DAY	501
3	BURGLARY	DAY	1220
4	HOMICIDE	DAY	0
5	MOTOR VEHICLE THEFT	DAY	1277
6	ROBBERY	DAY	918
7	SEX ABUSE	DAY	71
8	THEFT F/AUTO	DAY	5084
9	THEFT/OTHER	DAY	4977
10	ARSON	EVENING	12
11	ASSAULT W/DANGEROUS WEAPON	EVENING	977
12	BURGLARY	EVENING	1521
13	HOMICIDE	EVENING	0

Sorting your data

- The `order()` function is a convenient way to sort your data
  - For more details of what this function can do, run the `?order` command



Create new data set

Data set used

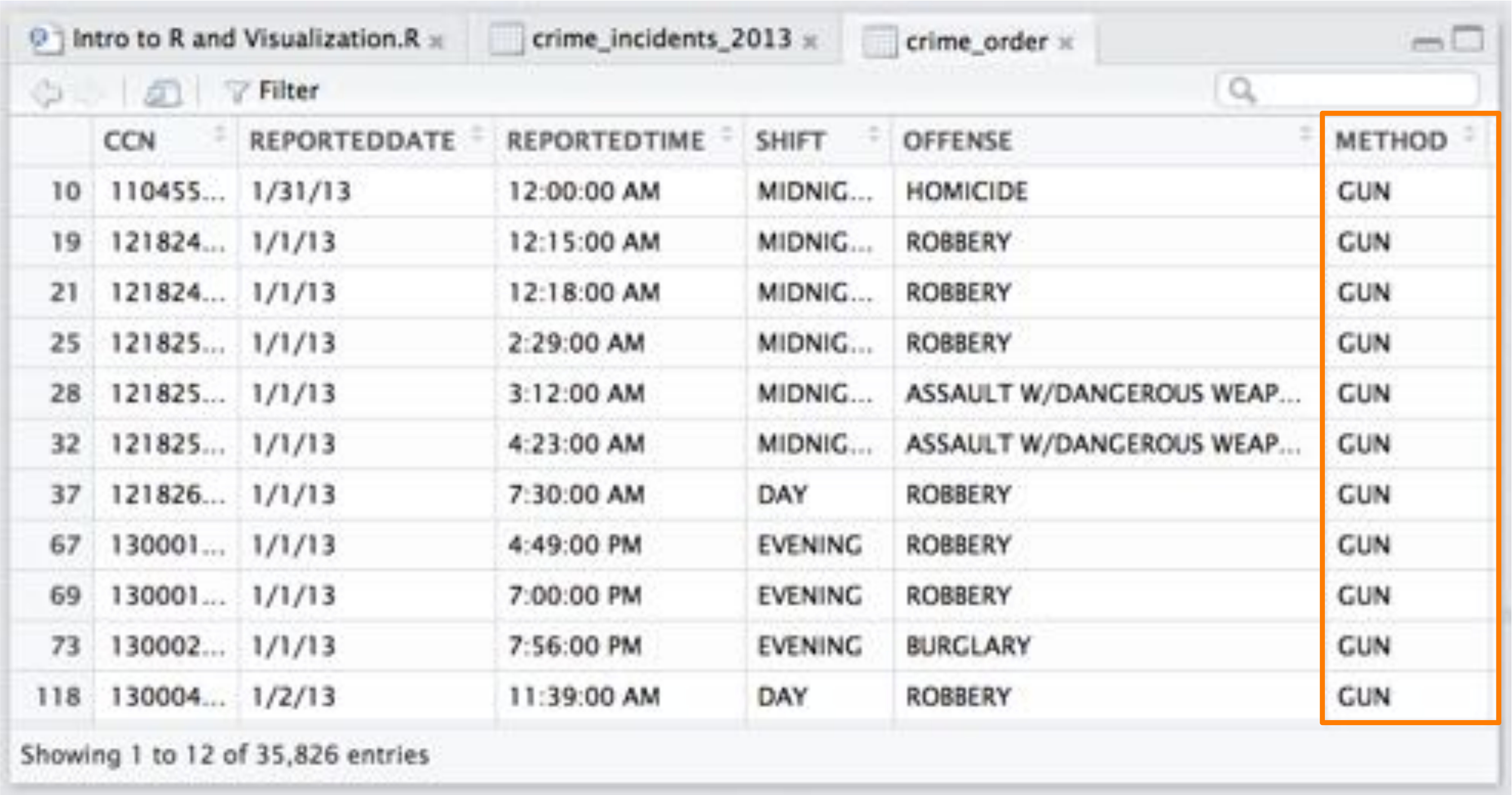
Column to sort by

Script

```
crime_order = crime_incidents_2013[order(crime_incidents_2013$METHOD,
                                          decreasing = FALSE),]
View(crime_order)
```

- Note that the `decreasing` operator can be set to `TRUE` or `FALSE` depending on the order you are looking for (i.e. ascending or descending)

Sorting your data



	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE	METHOD
10	110455...	1/31/13	12:00:00 AM	MIDNIG...	HOMICIDE	GUN
19	121824...	1/1/13	12:15:00 AM	MIDNIG...	ROBBERY	GUN
21	121824...	1/1/13	12:18:00 AM	MIDNIG...	ROBBERY	GUN
25	121825...	1/1/13	2:29:00 AM	MIDNIG...	ROBBERY	GUN
28	121825...	1/1/13	3:12:00 AM	MIDNIG...	ASSAULT W/DANGEROUS WEAP...	GUN
32	121825...	1/1/13	4:23:00 AM	MIDNIG...	ASSAULT W/DANGEROUS WEAP...	GUN
37	121826...	1/1/13	7:30:00 AM	DAY	ROBBERY	GUN
67	130001...	1/1/13	4:49:00 PM	EVENING	ROBBERY	GUN
69	130001...	1/1/13	7:00:00 PM	EVENING	ROBBERY	GUN
73	130002...	1/1/13	7:56:00 PM	EVENING	BURGLARY	GUN
118	130004...	1/2/13	11:39:00 AM	DAY	ROBBERY	GUN

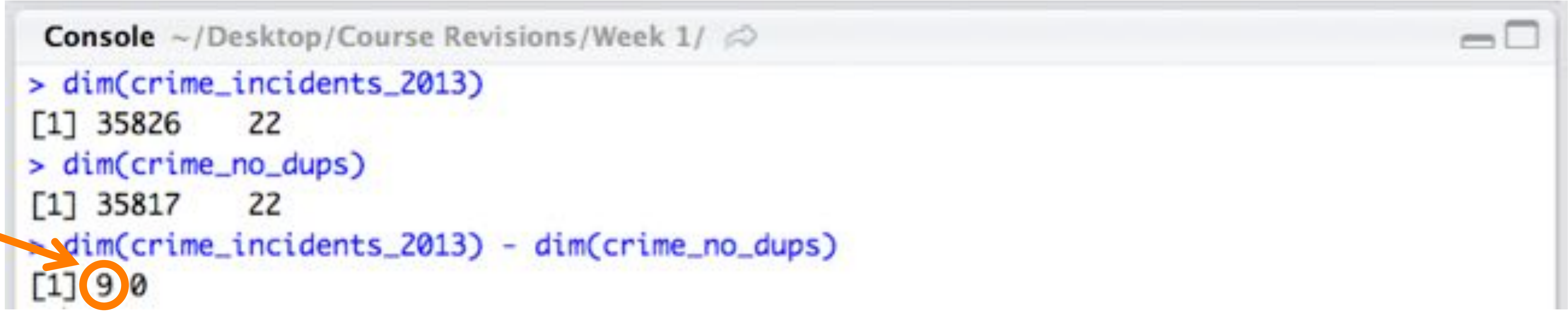
Showing 1 to 12 of 35,826 entries

Eliminating duplicate records

- You can eliminate repetitive instances of your data using `unique()`
  - Eliminates rows where every entry is the same as another row

Script

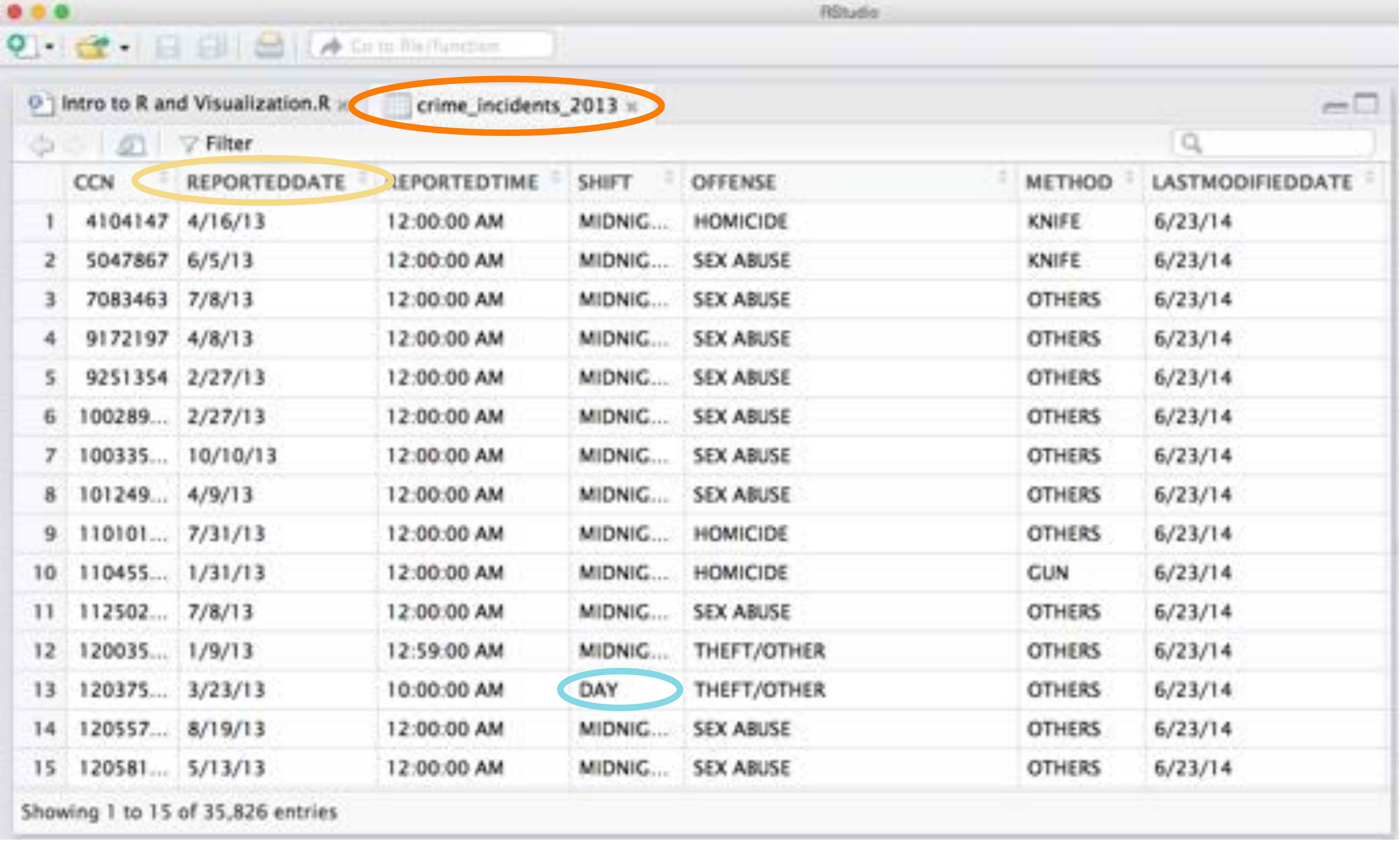
```
crime_no_dups = unique(crime_incidents_2013)

# Check how many rows are identical by looking at the dimensions of each data set
dim(crime_incidents_2013)
dim(crime_no_dups)

# You can just compute the difference in the dimensions of the data set
dim(crime_incidents_2013) - dim(crime_no_dups)
```



```
> dim(crime_incidents_2013)
[1] 35826    22
> dim(crime_no_dups)
[1] 35817    22
> dim(crime_incidents_2013) - dim(crime_no_dups)
[1] 9 0
```

9 rows were eliminated

Subsetting: real data example



	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE	METHOD	LASTMODIFIEDDATE
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE	KNIFE	6/23/14
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE	KNIFE	6/23/14
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
6	100289...	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
7	100335...	10/10/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
8	101249...	4/9/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
9	110101...	7/31/13	12:00:00 AM	MIDNIG...	HOMICIDE	OTHERS	6/23/14
10	110455...	1/31/13	12:00:00 AM	MIDNIG...	HOMICIDE	GUN	6/23/14
11	112502...	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
12	120035...	1/9/13	12:59:00 AM	MIDNIG...	THEFT/OTHER	OTHERS	6/23/14
13	120375...	3/23/13	10:00:00 AM	DAY	THEFT/OTHER	OTHERS	6/23/14
14	120557...	8/19/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
15	120581...	5/13/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14

Showing 1 to 15 of 35,826 entries

Subsetting: selects only values you want

- When working with large data sets, you may want to use or visualize only a portion of your data at any given time

Script

```
# Vector subsetting
v = c(10, 20, 30, 40, 50, 60)    # define your vector with 6 numbers
v[3]                             # select the 3rd term
v[1:4]                           # select the first 4 terms
v[3:6]                           # select the 3rd through the 6th term
v[c(1, 3, 6)]                    # select the 1st, 3rd and 6th term
2*v[3:6]                         # multiply the 3rd through the 6th term by 2
```

Console

```
[1] 30
[1] 10 20 30 40
[1] 30 40 50 60
[1] 10 30 60
[1] 60  80 100 120
```

Subsetting: selects only values you want

- When working with large data sets, you may want to use or visualize only a portion of your data at any given time

Script

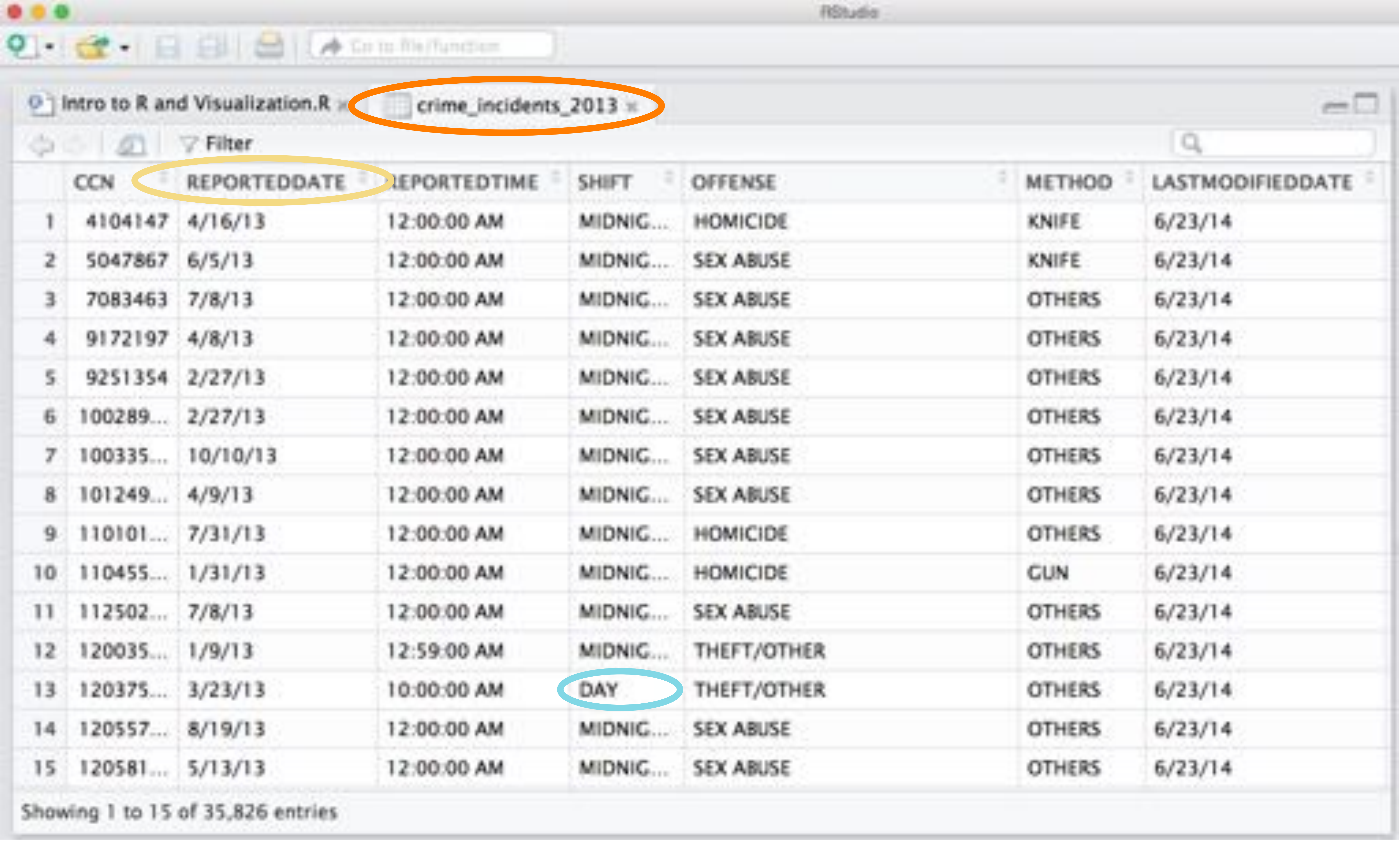
```
# Matrix subsetting
m = matrix(c(5, 6, 7, 8), 2, 2) # define your matrix with 4 numbers in
                                  2 rows and 2 columns
m                   # view your matrix
m[2,]               # select the numbers in the 2nd row
m[,1]               # select the numbers in the 1st column
m[1,2]              # select the number in the 1st row and 2nd column
m[1:3]              # select the first 3 terms in the matrix
m[1:2,1:2]          # select the #s in 1st and 2nd rows and 1st and 2nd cols
2*m[1:3]            # multiply the first 3 terms in the matrix by 2
```

Subsetting: selects only values you want

Console

```
> m                   # view your matrix
     [,1] [,2]
[1,]    5    7
[2,]    6    8
> m[2,]               # select the numbers in the 2nd row
[1] 6 8
> m[,1]               # select the numbers in the 1st column
[1] 5 6
> m[1,2]              # select the number in the 1st row and 2nd column
[1] 7
> m[1:3]              # select the first 3 terms in the matrix
[1] 5 6 7
> m[1:2,1:2]          # select the #s in 1st and 2nd rows and 1st and 2nd cols
      [,1] [,2]
[1,]    5    7
[2,]    6    8
> 2*m[1:3]            # multiply the first 3 terms in the matrix by 2
[1] 10 12 14
```

Subsetting: real data example



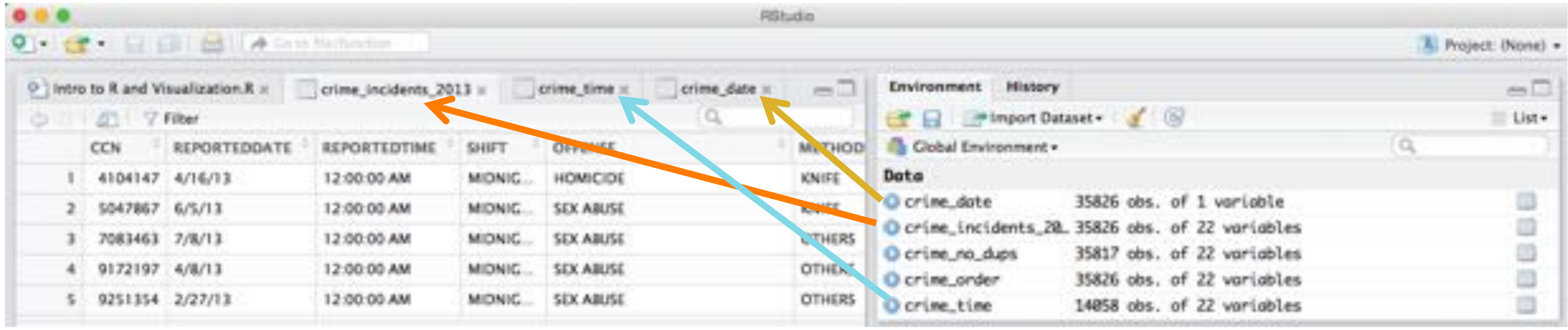
	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE	METHOD	LASTMODIFIEDDATE
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE	KNIFE	6/23/14
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE	KNIFE	6/23/14
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
6	100289...	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
7	100335...	10/10/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
8	101249...	4/9/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
9	110101...	7/31/13	12:00:00 AM	MIDNIG...	HOMICIDE	OTHERS	6/23/14
10	110455...	1/31/13	12:00:00 AM	MIDNIG...	HOMICIDE	GUN	6/23/14
11	112502...	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
12	120035...	1/9/13	12:59:00 AM	MIDNIG...	THEFT/OTHER	OTHERS	6/23/14
13	120375...	3/23/13	10:00:00 AM	DAY	THEFT/OTHER	OTHERS	6/23/14
14	120557...	8/19/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14
15	120581...	5/13/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS	6/23/14

Showing 1 to 15 of 35,826 entries

Subsetting: real data example

Script

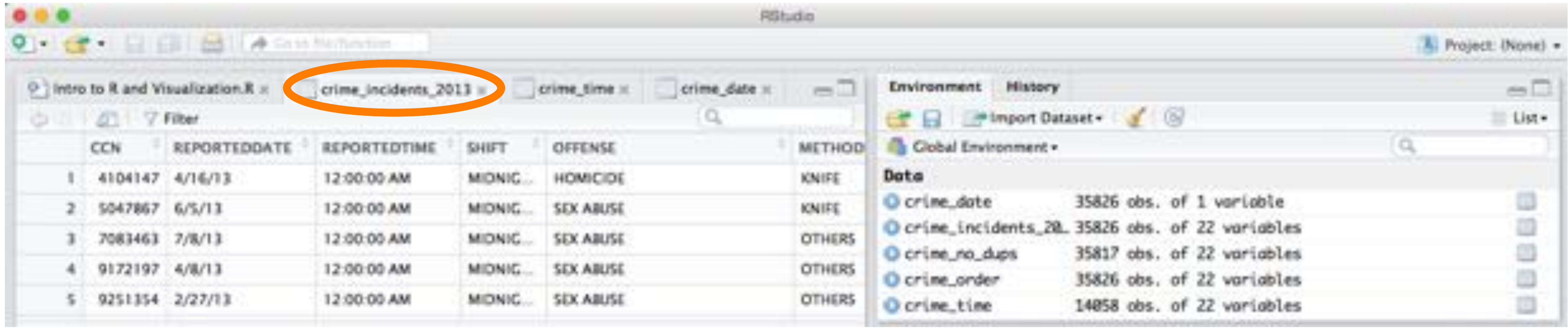
```
# Select data from the crime data set based on time of day the crime occurred
crime_time = subset(crime_incidents_2013, SHIFT == "DAY")

# Or you can also use
crime_time = crime_incidents_2013[crime_incidents_2013$SHIFT == "DAY",]
View(crime_time)

# Select 1 column from the data set
crime_date = subset(crime_incidents_2013, select = REPORTEDDATE)
View(crime_date)
```



	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE	METHOD
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE	KNIFE
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE	KNIFE
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS

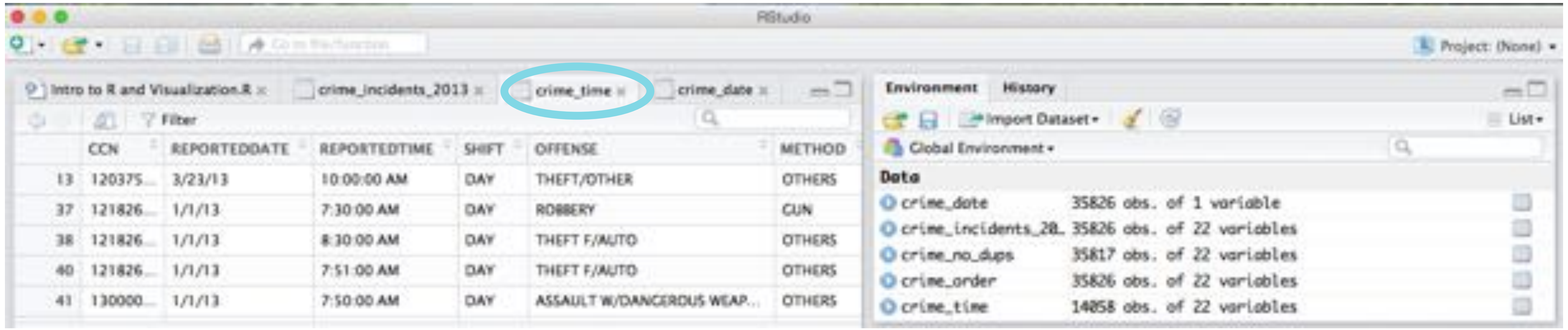
```
Data
crime_date           35826 obs. of 1 variable
crime_incidents_20…  35826 obs. of 22 variables
crime_no_dups        35817 obs. of 22 variables
crime_order          35826 obs. of 22 variables
crime_time           14058 obs. of 22 variables
```

Subsetting: real data example



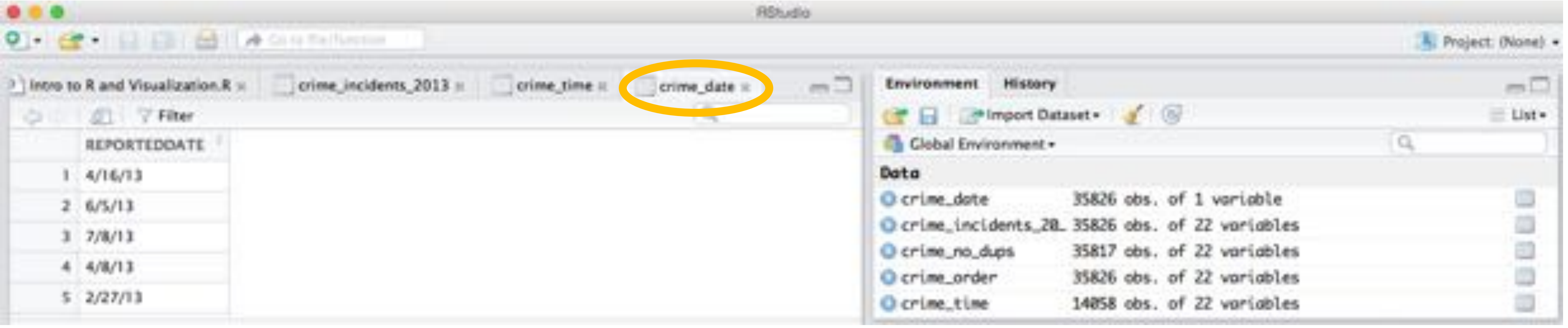
	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE	METHOD
1	4104147	4/16/13	12:00:00 AM	MIDNIG...	HOMICIDE	KNIFE
2	5047867	6/5/13	12:00:00 AM	MIDNIG...	SEX ABUSE	KNIFE
3	7083463	7/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS
4	9172197	4/8/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS
5	9251354	2/27/13	12:00:00 AM	MIDNIG...	SEX ABUSE	OTHERS

```
Data
crime_date           35826 obs. of 1 variable
crime_incidents_20.. 35826 obs. of 22 variables
crime_no_dups        35817 obs. of 22 variables
crime_order          35826 obs. of 22 variables
crime_time           14058 obs. of 22 variables
```



	CCN	REPORTEDDATE	REPORTEDTIME	SHIFT	OFFENSE	METHOD
13	120375...	3/23/13	10:00:00 AM	DAY	THEFT/OTHER	OTHERS
37	121826...	1/1/13	7:30:00 AM	DAY	ROBBERY	GUN
38	121826...	1/1/13	8:30:00 AM	DAY	THEFT F/AUTO	OTHERS
40	121826...	1/1/13	7:51:00 AM	DAY	THEFT F/AUTO	OTHERS
41	130000...	1/1/13	7:50:00 AM	DAY	ASSAULT W/DANGEROUS WEAP...	OTHERS

```
Data
crime_date           35826 obs. of 1 variable
crime_incidents_20.. 35826 obs. of 22 variables
crime_no_dups        35817 obs. of 22 variables
crime_order          35826 obs. of 22 variables
crime_time           14058 obs. of 22 variables
```



	REPORTEDDATE
1	4/16/13
2	6/5/13
3	7/8/13
4	4/8/13
5	2/27/13

```
Data
crime_date           35826 obs. of 1 variable
crime_incidents_20.. 35826 obs. of 22 variables
crime_no_dups        35817 obs. of 22 variables
crime_order          35826 obs. of 22 variables
crime_time           14058 obs. of 22 variables
```

Subsetting: real data example

- What if we wanted to select only robberies that happened during the day?
  - When performing several operations, keep the steps separate so it's easier to check your code

Script

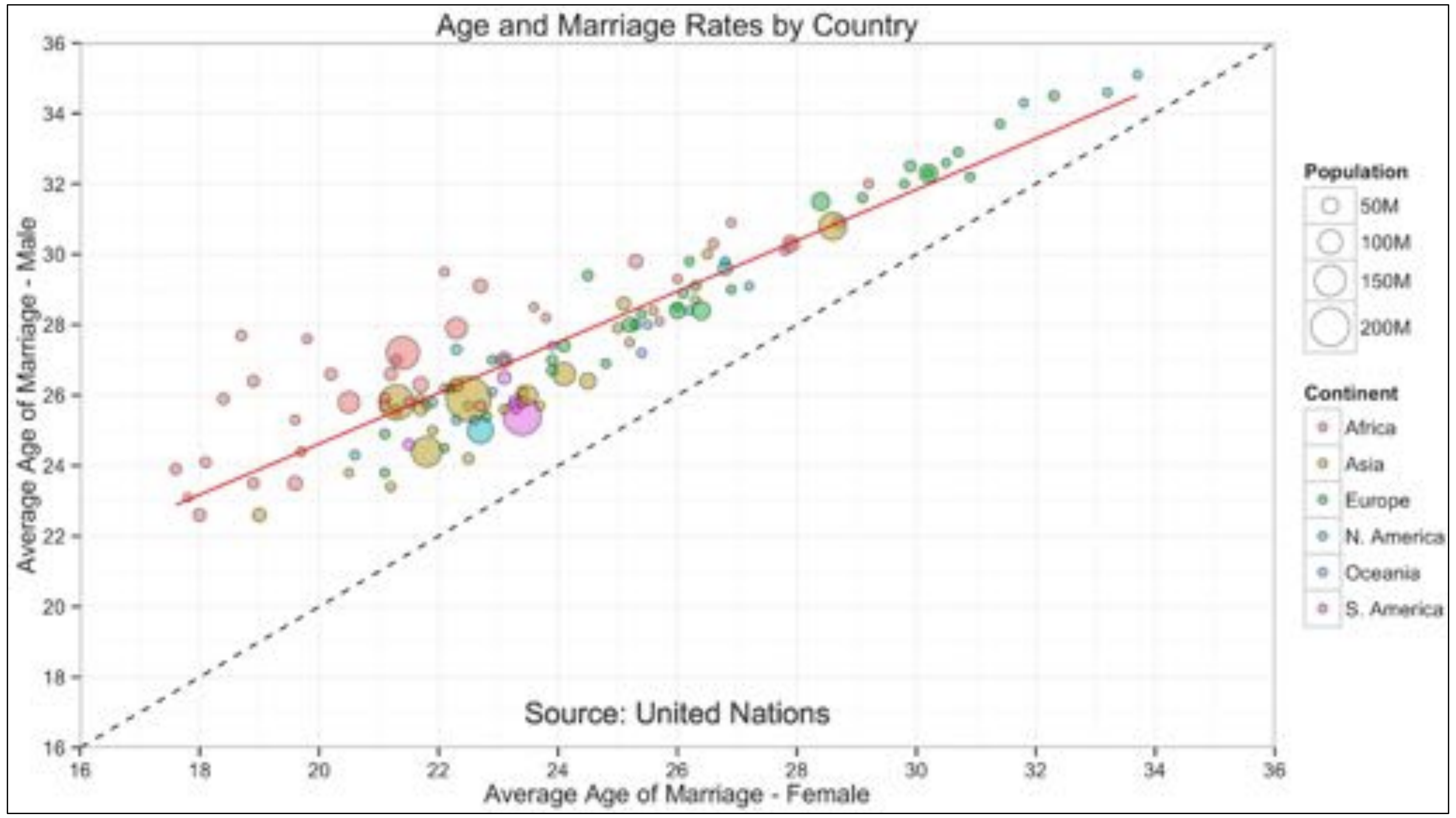
```
# First, only select the crimes that happened during the day
crime_time = subset(crime_incidents_2013, SHIFT == "DAY")

# Second, select only robberies out of the new data set called "crime_time"
crime_time_robbery = subset(crime_time, OFFENSE == "ROBBERY")
View(crime_time_robbery)
```

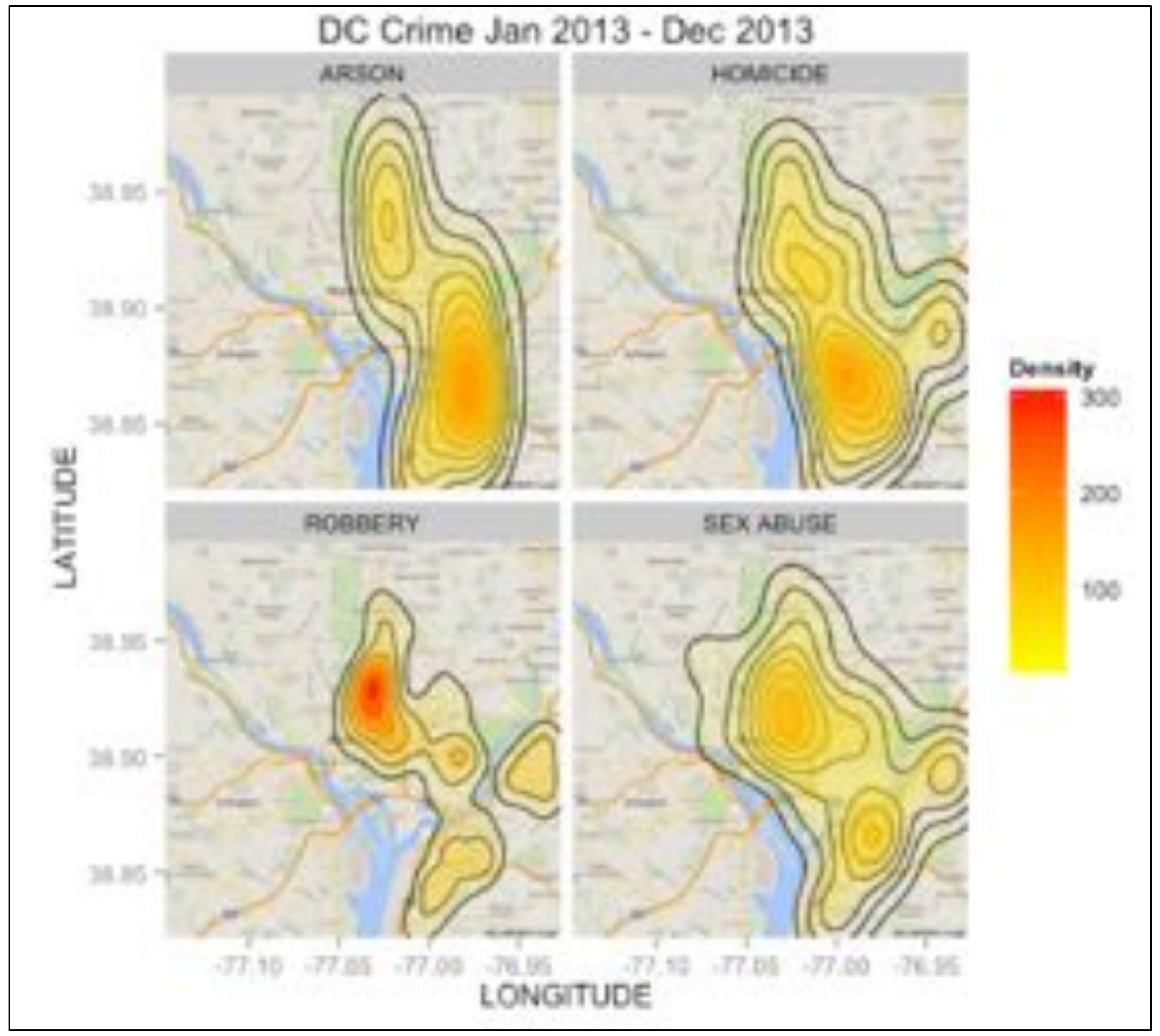
Outline

- What is data science?
- A data scientist's approach
- Introduction to R
  - Calculations in R
  - Reading data into R
  - Manipulating data in R
- Visualization in R
  - Basic plotting

Why build a visualization?

- To communicate your ideas
- To better understand your data
- To discover a new insight
- Visualization is a great way to do exploratory data analysis (“EDA”)



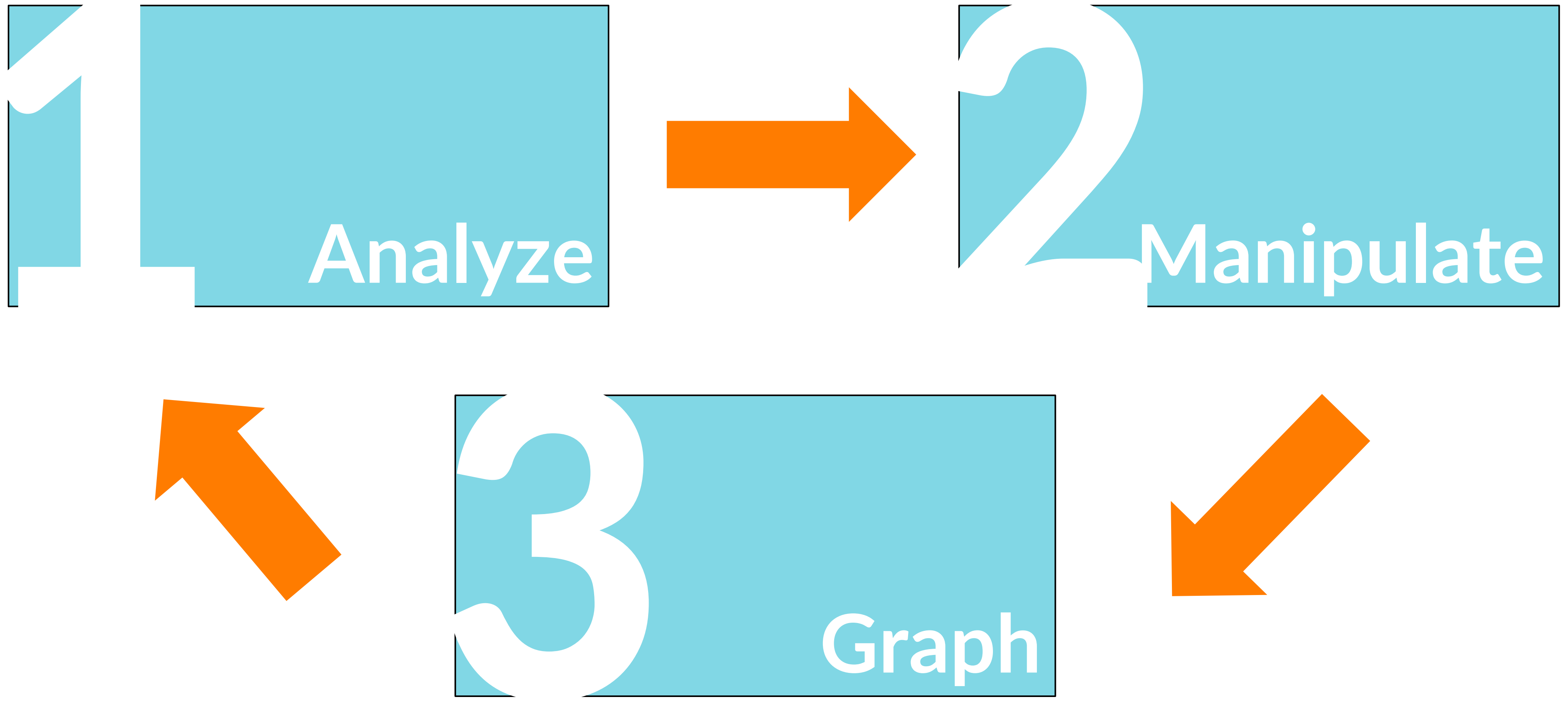


Exploratory data analysis

- R is a powerful tool for EDA because the graphics tie in with the functions used to analyze data

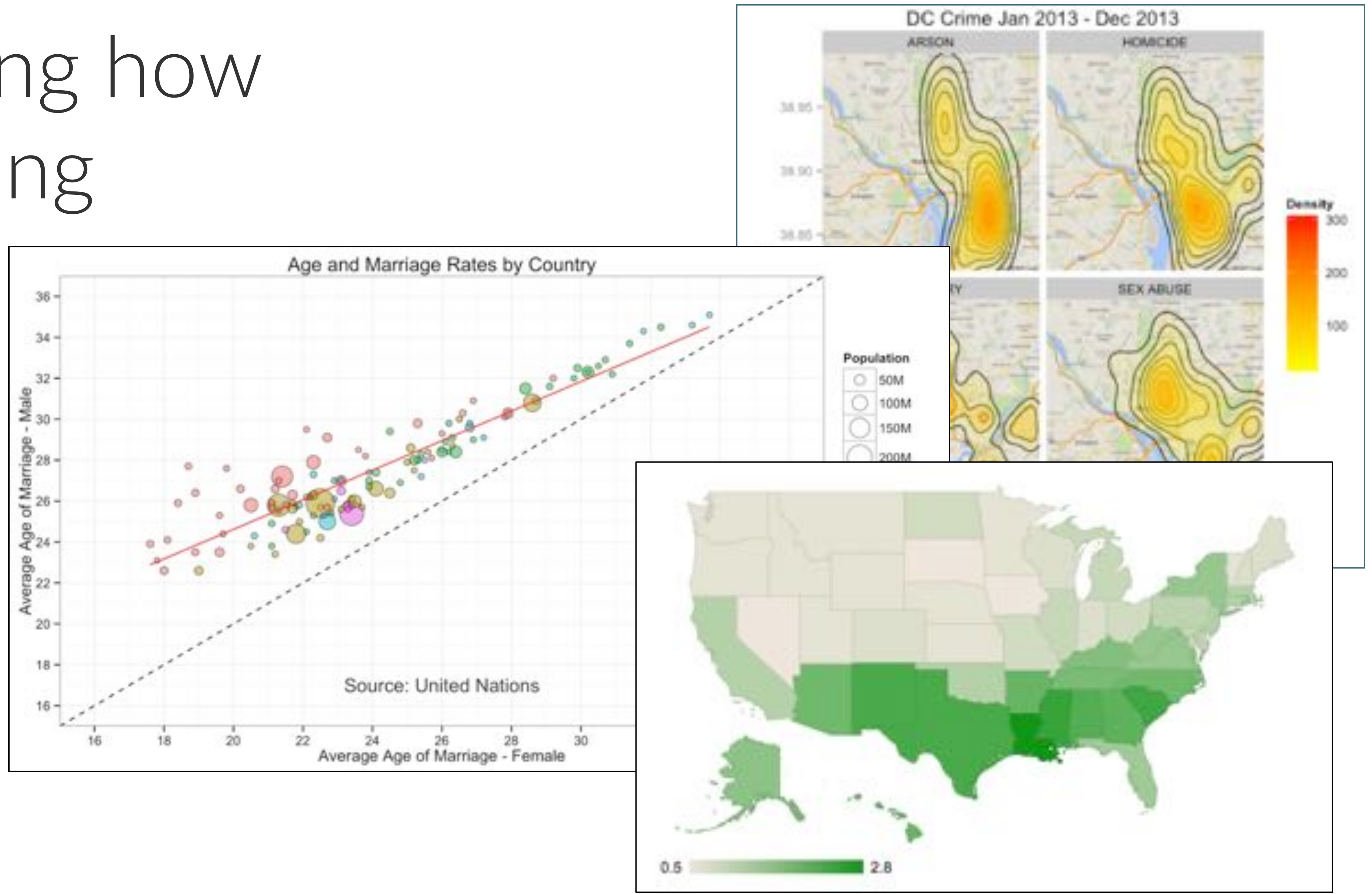
- You can create graphs without breaking your train of thought as you explore your data

- Visualization is an iterative process
  1. Analyze
  2. Manipulate
  3. Graph
  4. Repeat



What can you ultimately do

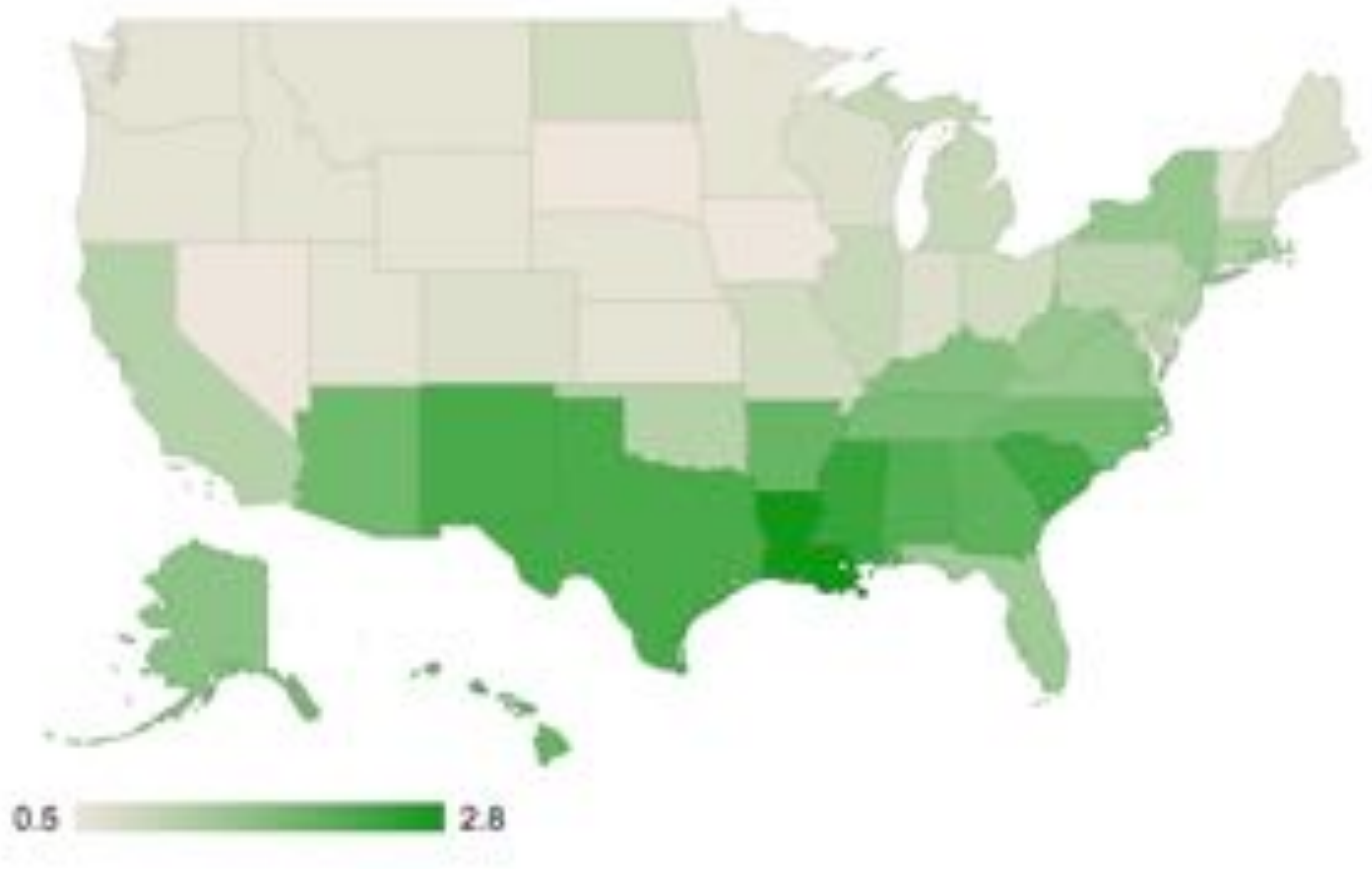
- The key to visualization in R is understanding how your data maps to the image you are creating
  - How R interprets your data
- Visualization types
  - Maps
  - Dynamic visualizations
  - Web based & interactive
  - 3D
  - Composite graphs made of many different elements
- Repeatable (no need to ever re-create a graph, just re-use the commands you already typed out)
- Use `??visualization` to see other examples



Visualization in R

- R comes with basic plotting functionality

- More advanced visualizations are done through packages
  - `ggplot2`
  - `ggpairs`
  - `ggmaps`
  - `rgl`
  - `googleVis`
  - `lattice`

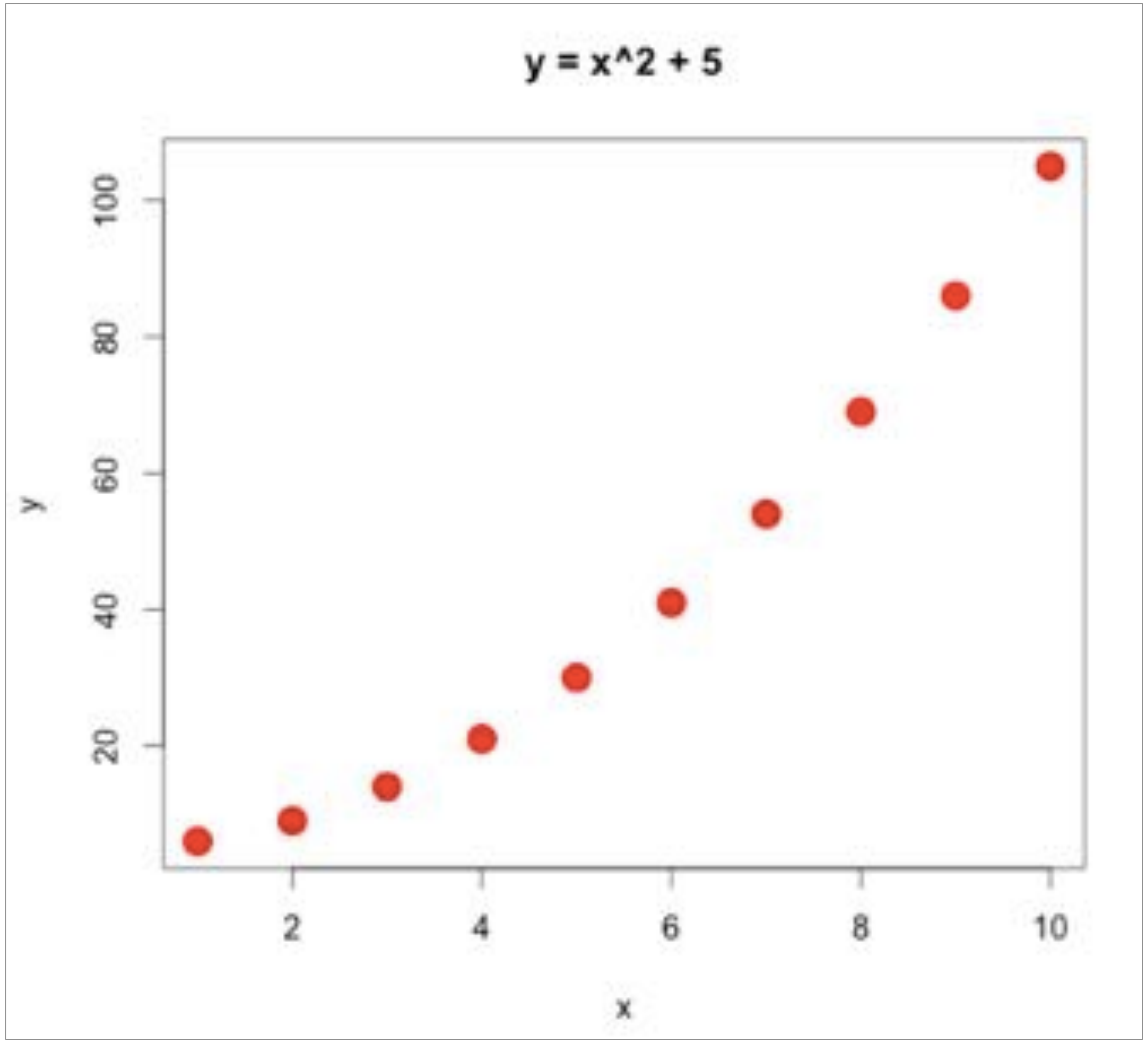


Basic scatter plot

- `x, y` – coordinates
- `xlab, ylab, main` – graph labels
- `lwd` – size of dot / line
- `col` – colors, “red”, ”blue”, “green”,...

Script

```
# Suppose we want to plot y = x^2 + 5
x = seq(1:10)
y = x^2 + 5

plot(x, y,
     lwd = 10,
     col = "red",
     main = "y = x^2 + 5")
```

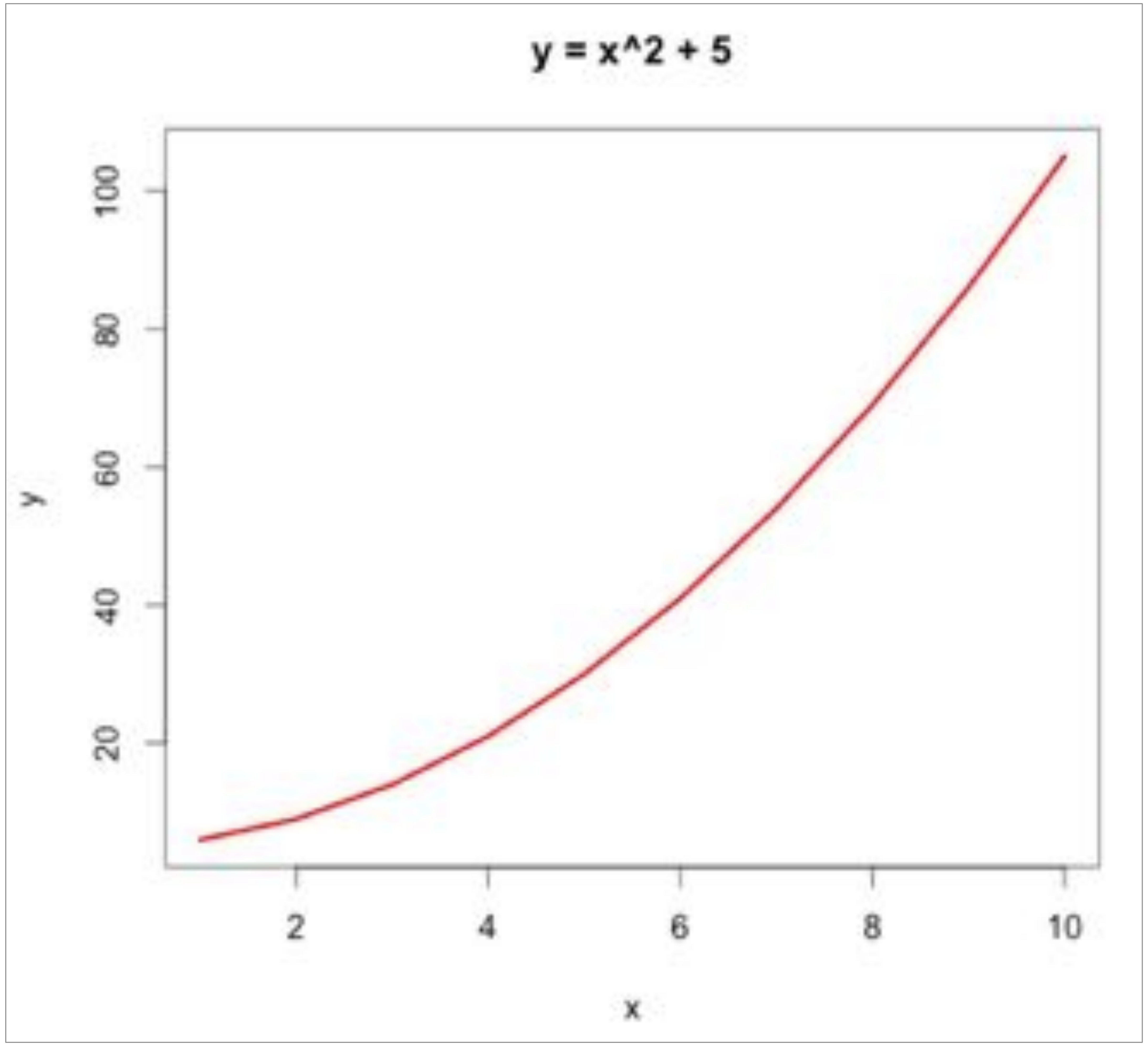


Basic line plot

- x, y – coordinates
- xlab, ylab, main – graph labels
- lwd – size of dot / line
- type – line, points or both
  - "l" – line
  - "b" – line and dots
- col – colors, “red”, ”blue”, “green”,...

Script

```
# Suppose we want to plot y = x^2 + 5
x = seq(1:10)
y = x^2 + 5
plot(x, y,
     xlab = "x",
     ylab = "y",
     main = "y = x^2 + 5",
     lwd = 3,
     type = "l",
     col = "red")
```

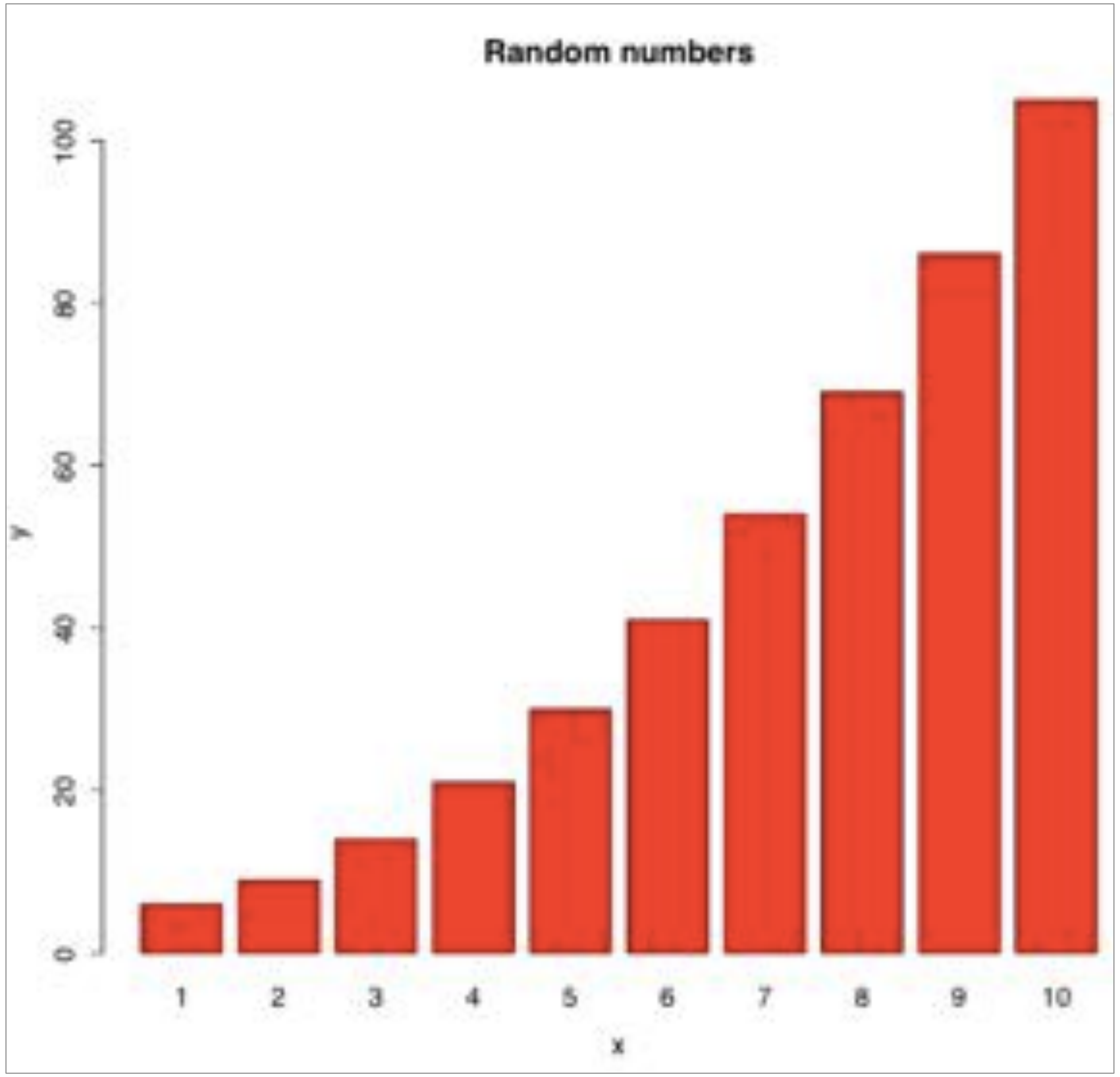


Basic bar plot

- `height` – value of each bar
- `xlab, ylab, main` – graph labels
- `names.arg` – labels to add to the x axis
- `col` – colors, “red”, ”blue”, “green”,…

Script

```
# Suppose we want to plot 10 random
# numbers using the formula y = x^2 + 5
x = seq(1:10)
y = x^2 + 5
barplot(height = y,
        width = 1,
        xlab = "x",
        ylab = "y",
        main = "Random numbers",
        names.arg = x,
        col = "red")
```

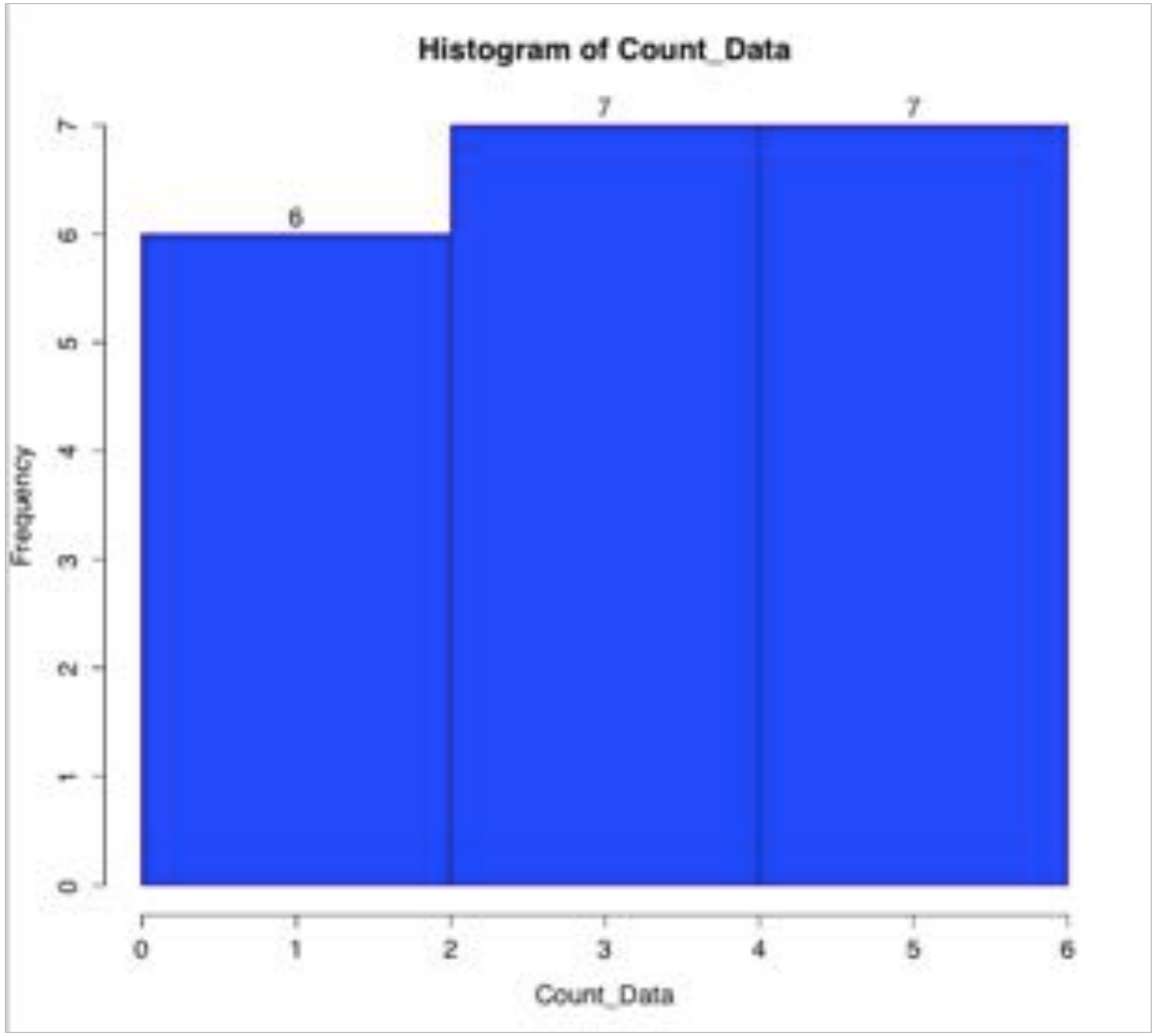


Basic histogram

- col – colors, “red”, ”blue”, “green”,…
- xlab, ylab, main – graph labels
- labels – labels on top of the columns
- breaks – allows you to specify a custom number of groupings

Script

```
# Let's plot a histogram of a sequence
of numbers
Count_Data = c(1, 1, 1, 1, 1,
               2,
               3, 3, 3, 3,
               4, 4, 4,
               5, 5, 5, 5, 5, 5, 5)
hist(Count_Data,
     col = "blue",
     labels = TRUE,
     breaks = 2)
```



Functions for basic graphs

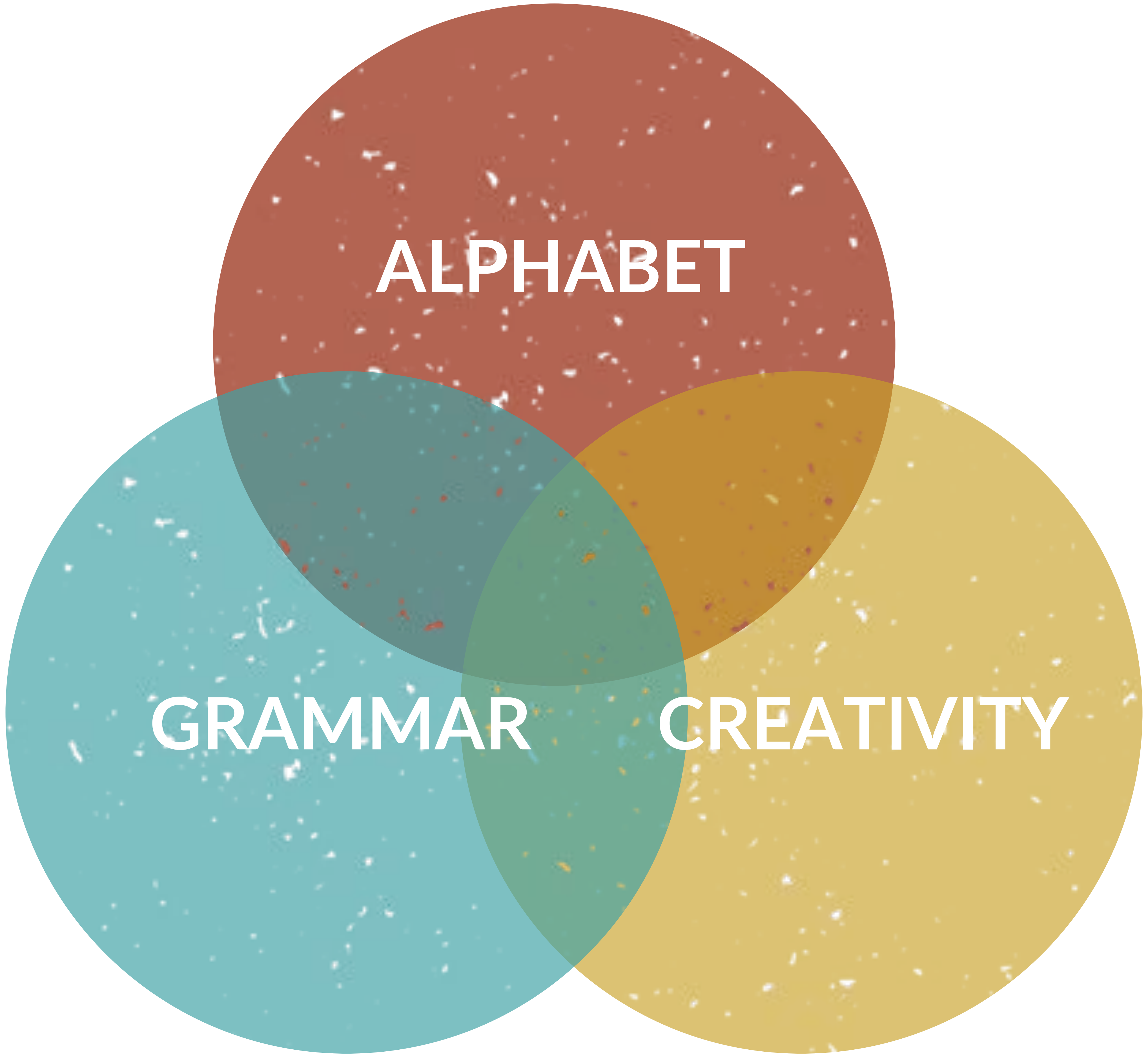
Chart type	Command
Line chart	`plot()`
Bar chart	`barplot()`
Histogram	`hist()`
Boxplot	`boxplot()`
Density plot	`density()`
Pie chart	`pie()`
Scatter plot	`plot()`
Heat map	`image()`

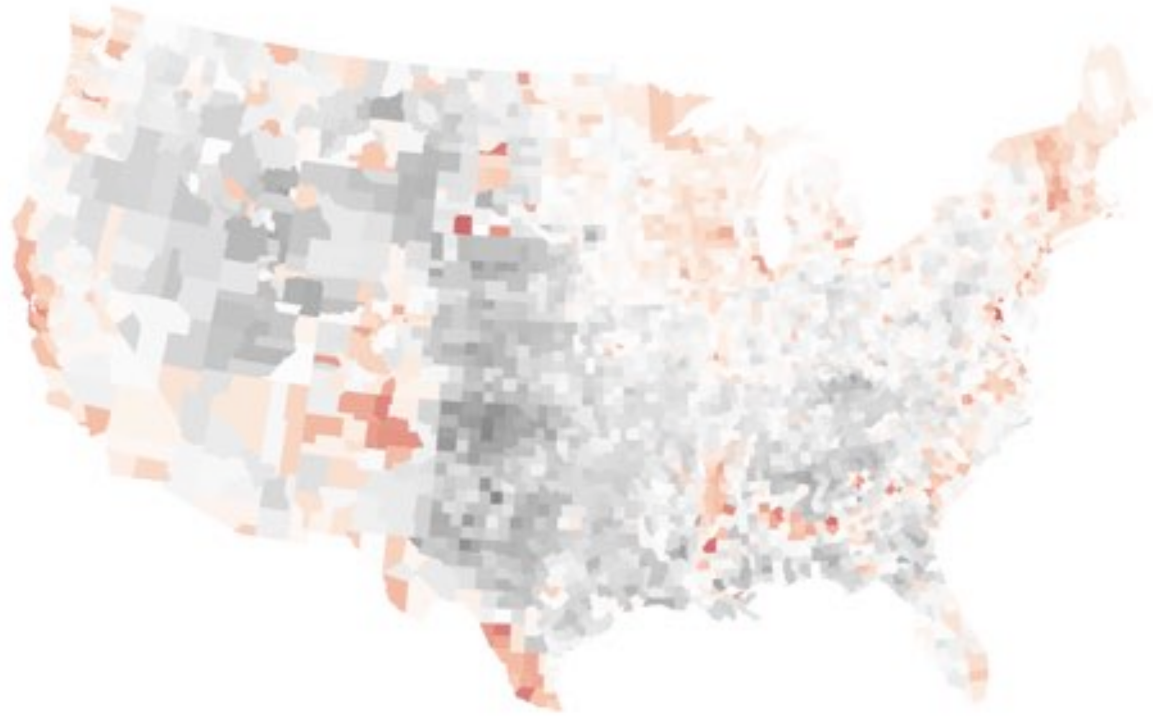
Grammar of graphics: `ggplot2`

- R comes with basic plotting functionality

- More advanced visualizations done through packages: `ggplot2`

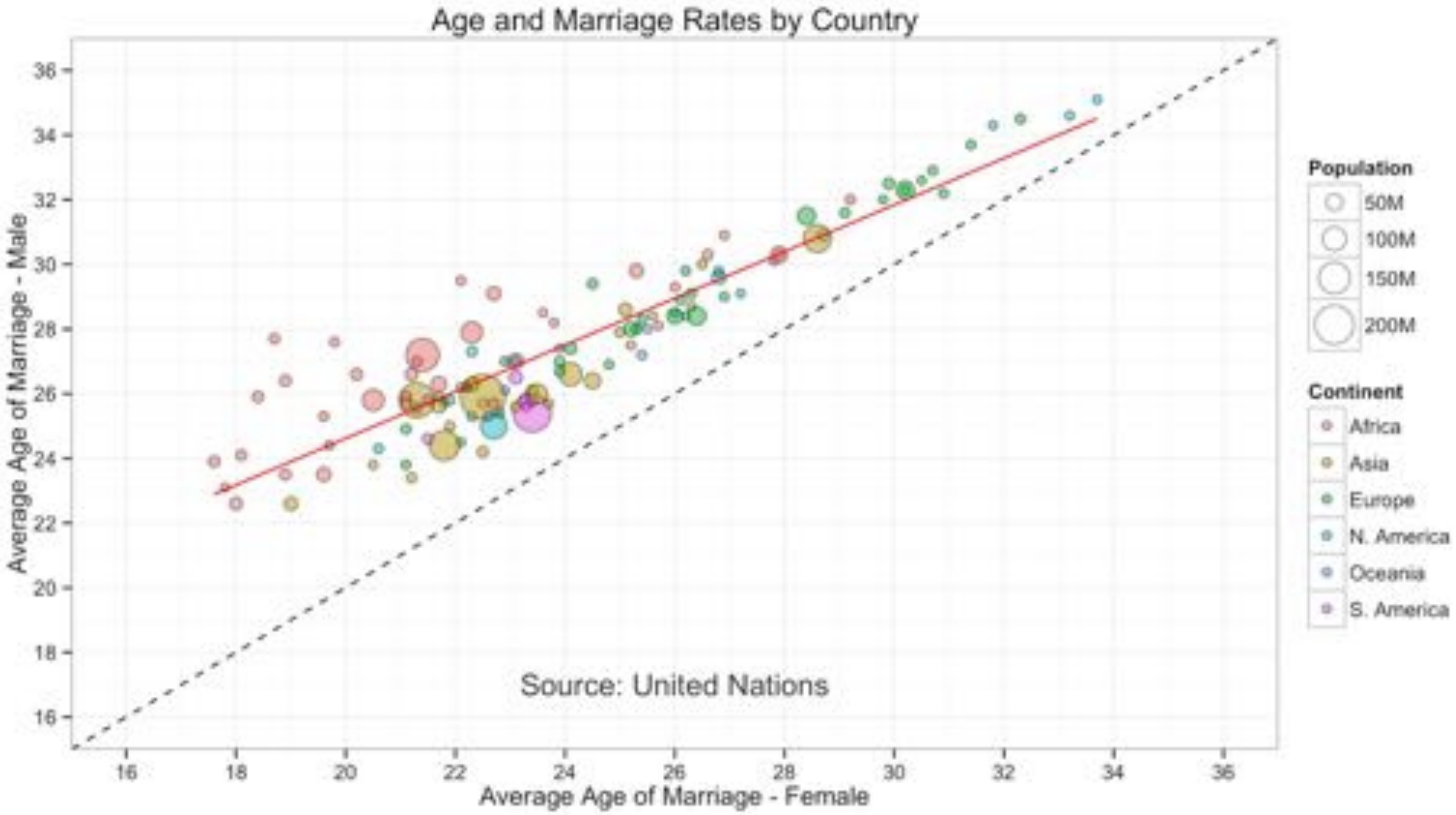
- `ggplot2` allows the user to create flexible visualizations by combining many elements into a single image



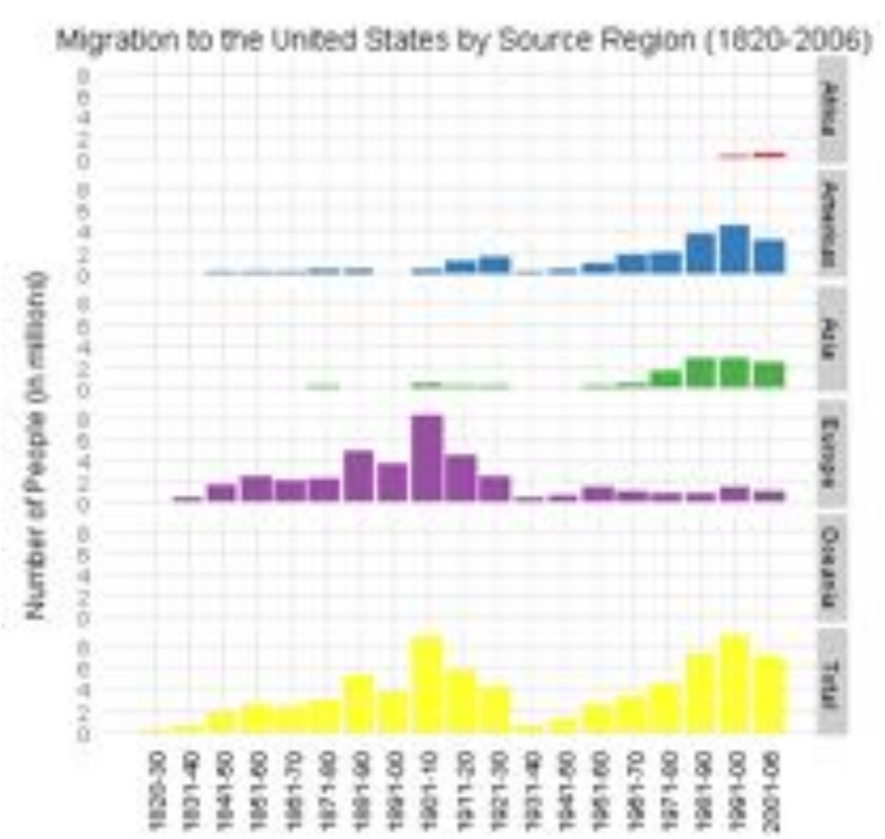


Why `ggplot2`

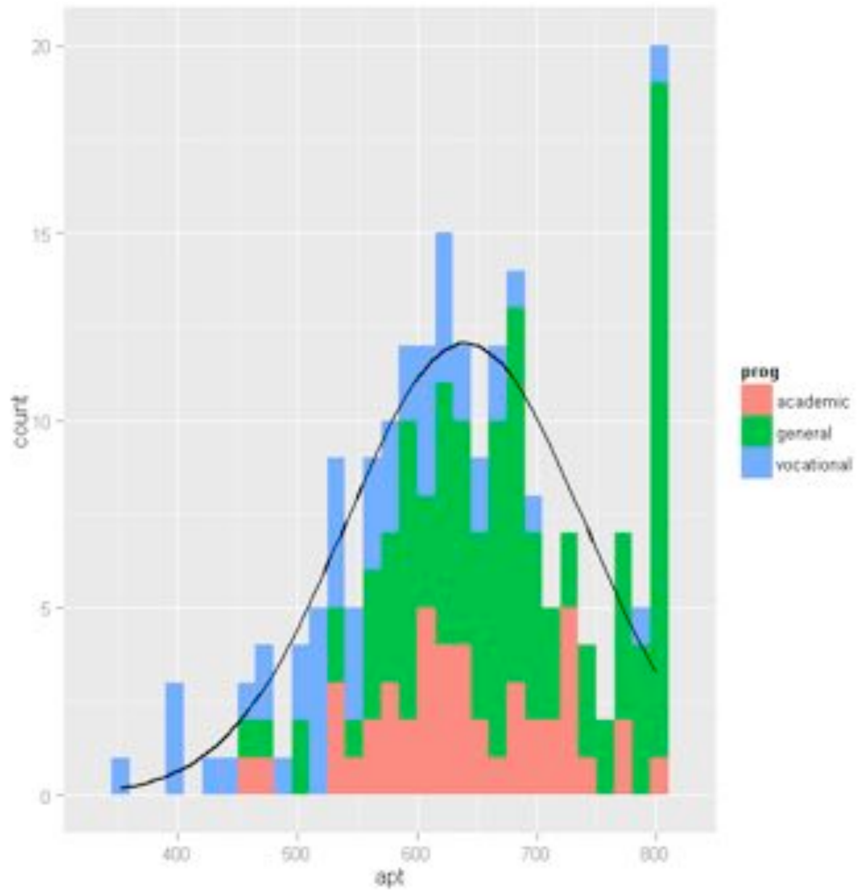
- Explore your data efficiently
- Communicate a visual story flexibly and efficiently
- Layer raw, summarized and contextual data
  - Demonstrate relationships
- Reproduce and extend your work easily



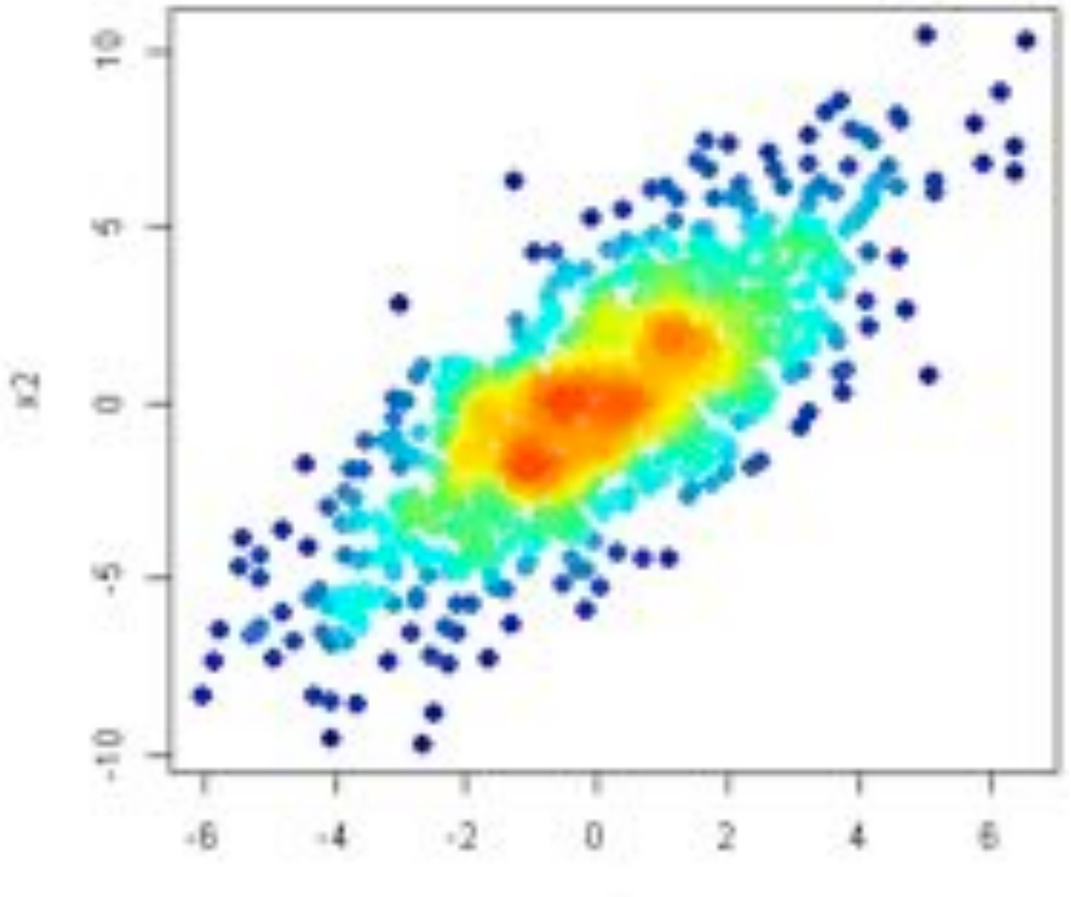
Combine basic plots for enhanced effect



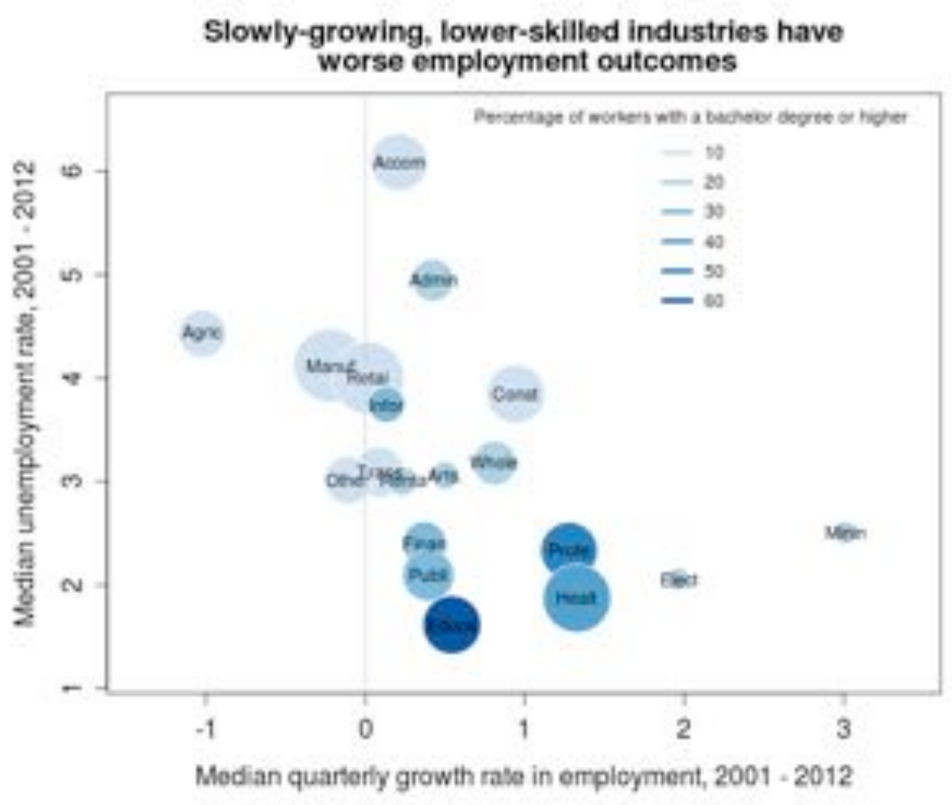
Bar plot



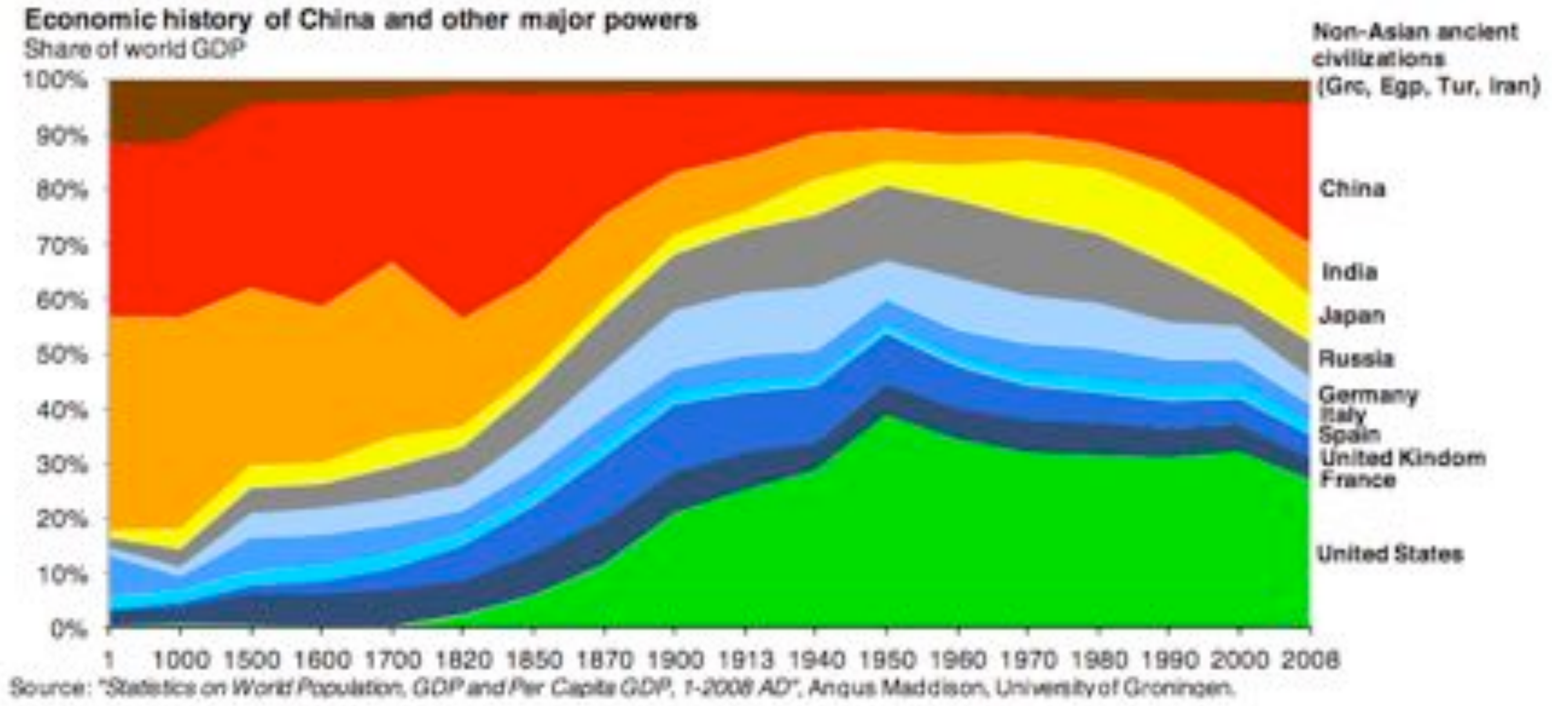
Histogram



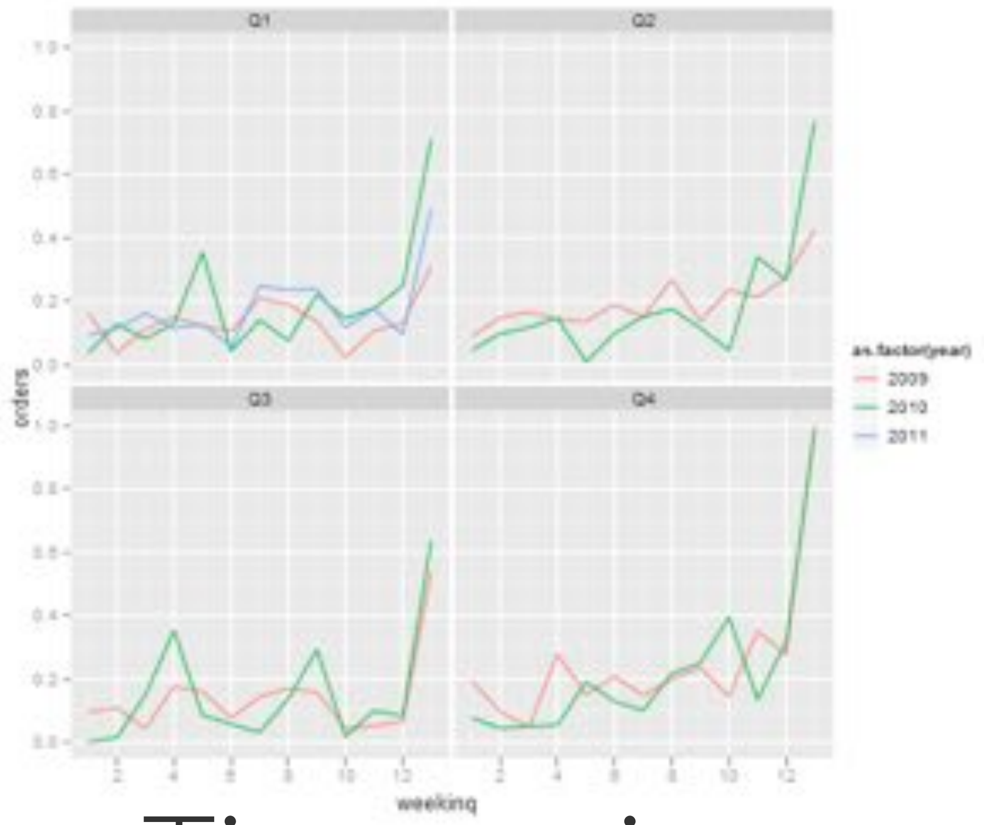
Scatterplot



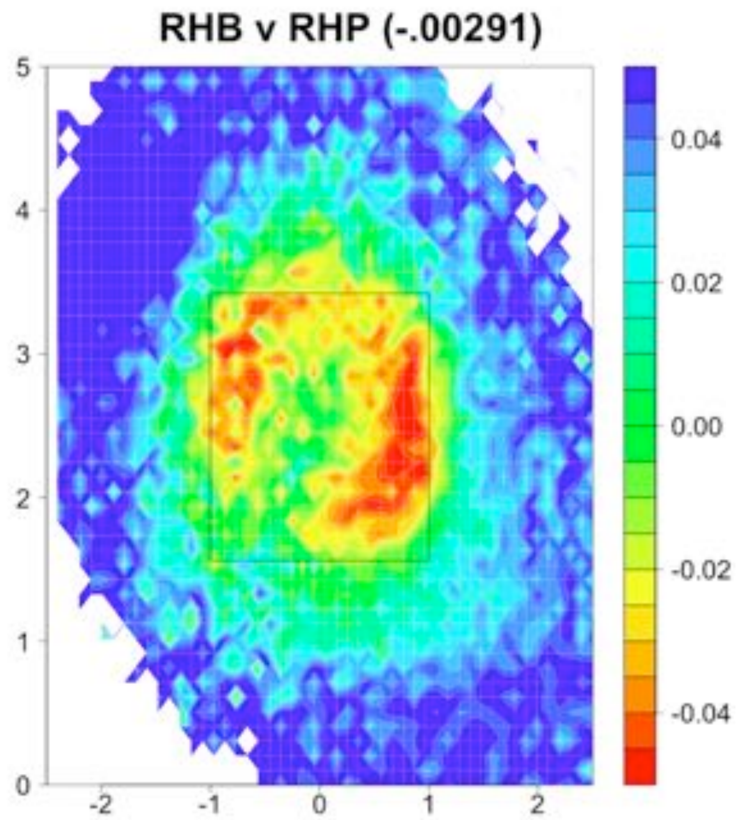
Bubble plot



Area plot

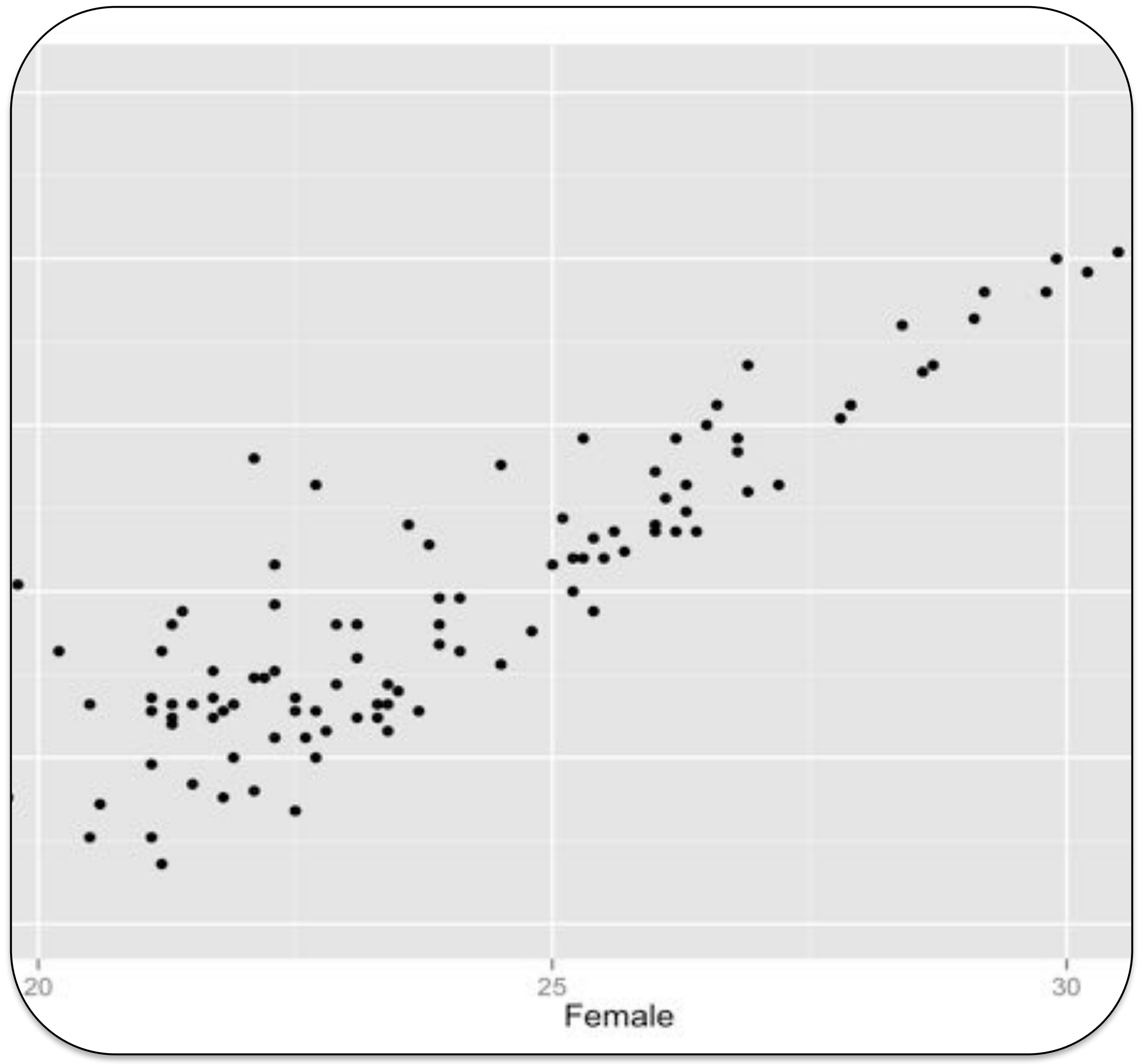


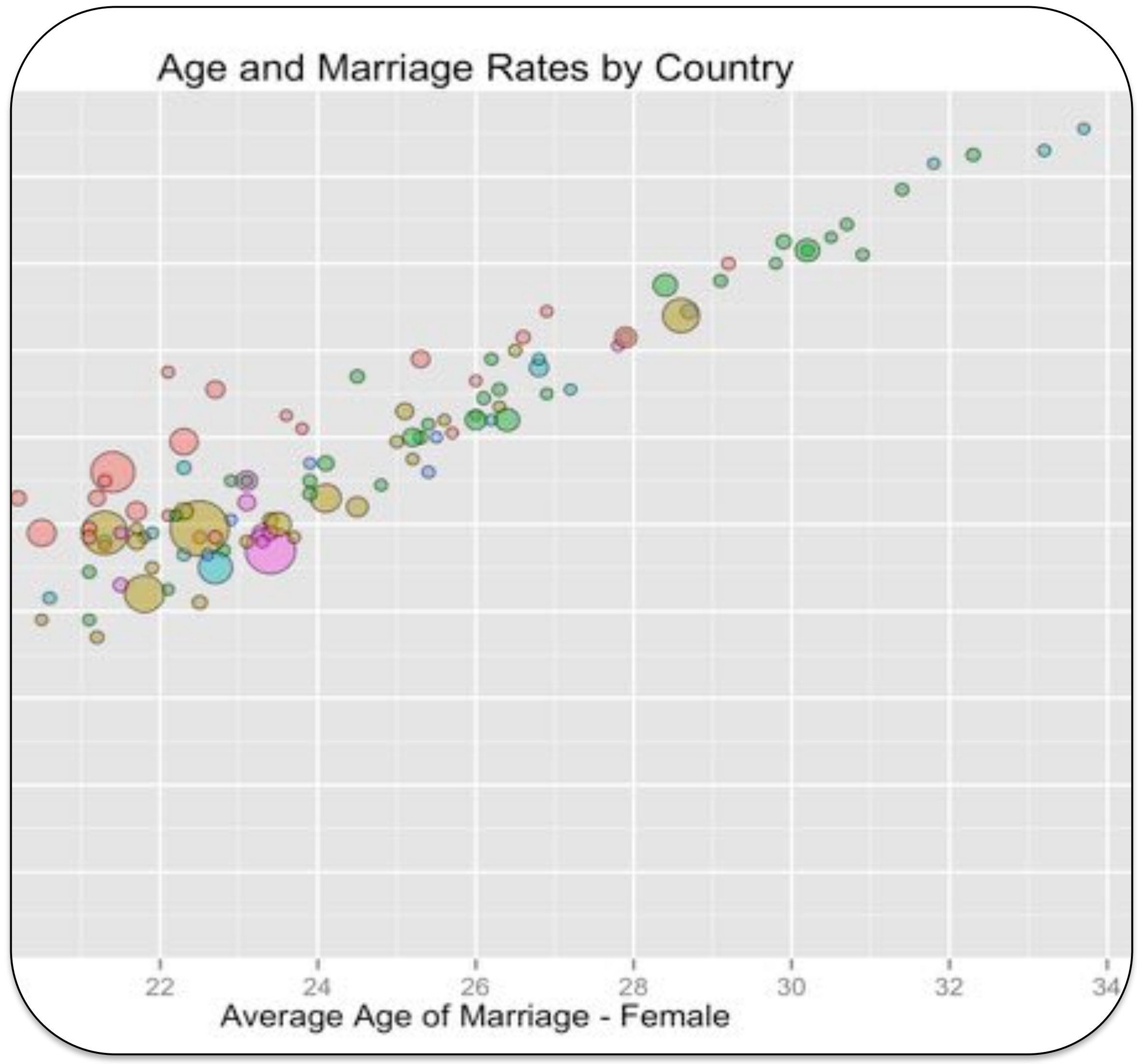
Time series

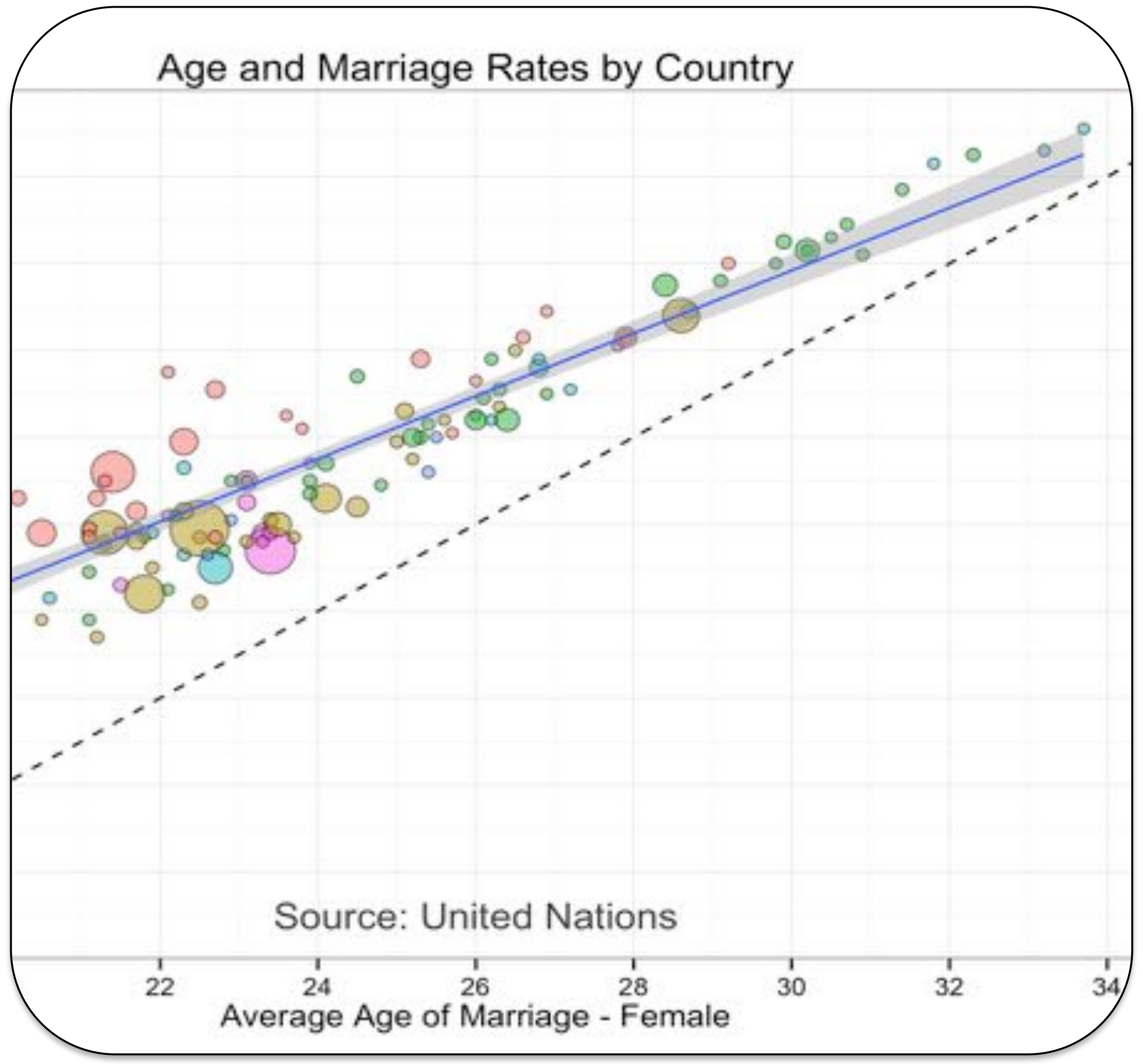


Heat map

`ggplot2`: building a graph







1. Specify data
2. Link data to visuals
3. Assign shapes

4. Adjust vis. effects
5. Adjust axes
6. Adjust legend

7. Customize theme
8. Layer statistics
9. Overlay text

Reading in data and loading library

Script

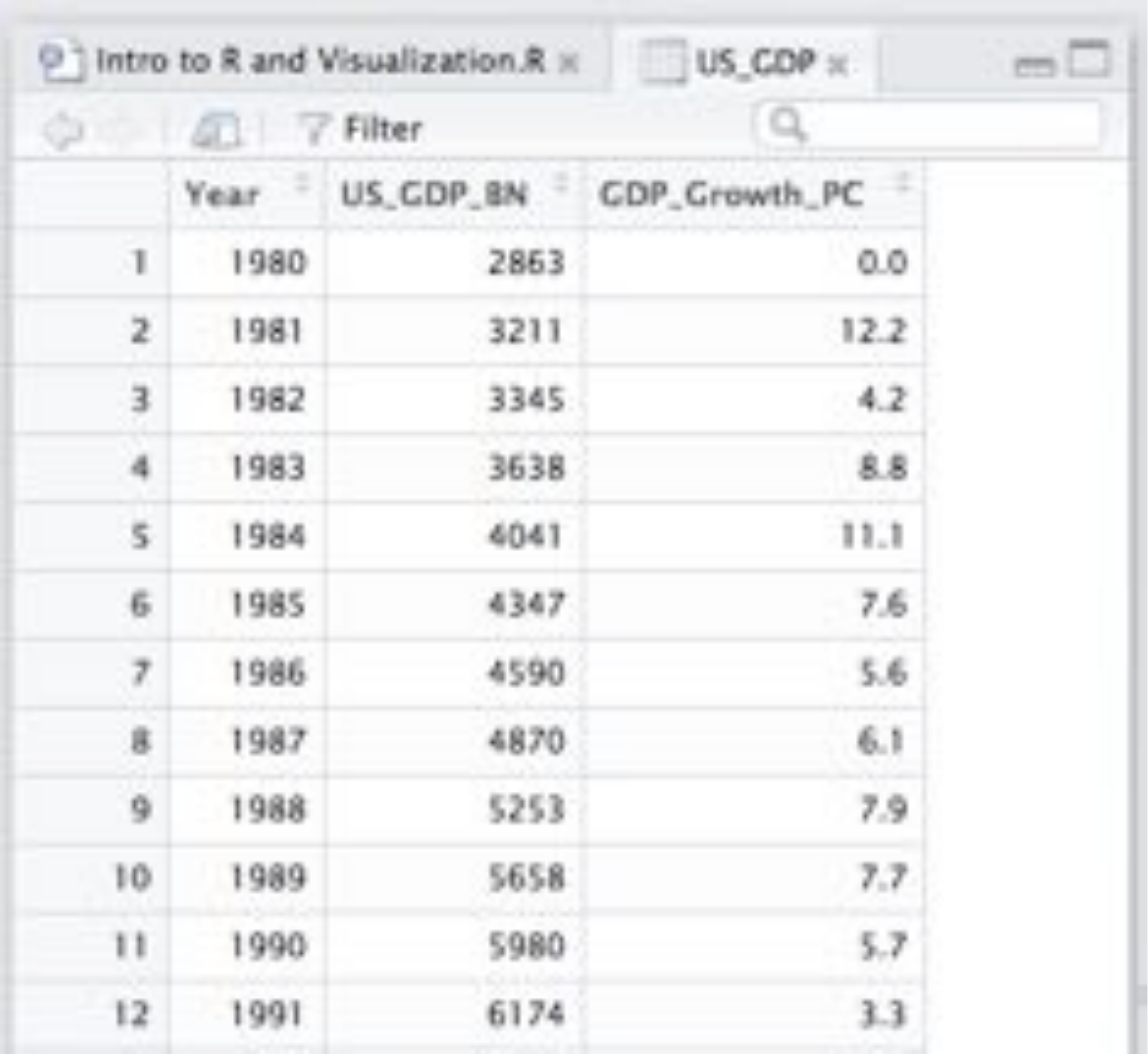
```
# Install ggplot2
install.packages("ggplot2")

# Loading the library ggplot2
library(ggplot2)

# Learning what ggplot2 can do
library(help = ggplot2)

# Reading in files, windows uses the C:/ directory, Mac uses the ~/ directory
setwd("path to files")
US_GDP = read.csv("US GDP.csv")

# View the file
View(US_GDP)
```

- Help: http://docs.ggplot2.org/current/

U.S. GDP data set

What data is available?

- U.S. GDP by year in $USD billions
- U.S. GDP growth rate by year in percentages
- Years: 1980 - 2014
- Source: U.S. Federal Reserve



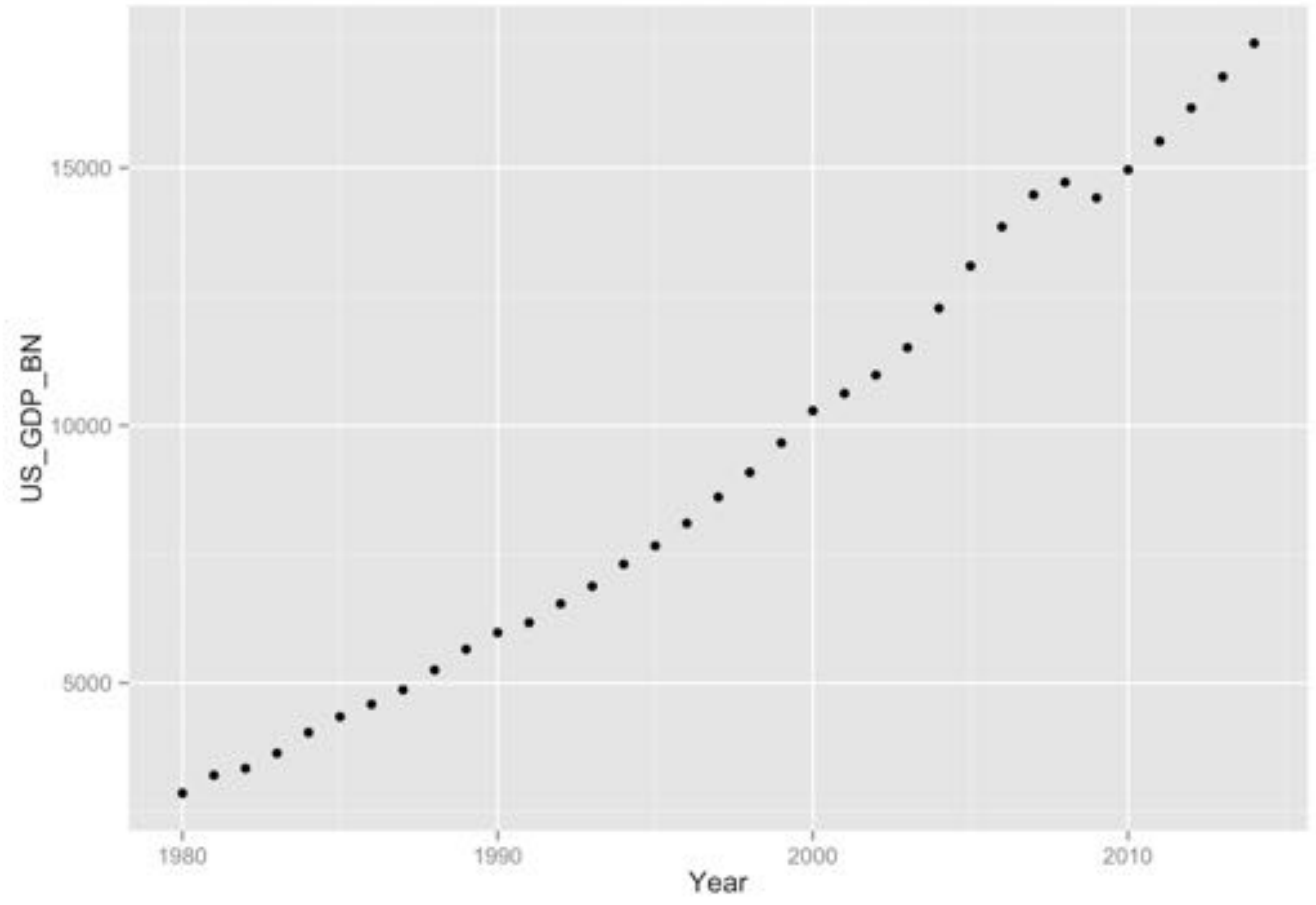
	Year	US_GDP_BN	GDP_Growth_PC
1	1980	2863	0.0
2	1981	3211	12.2
3	1982	3345	4.2
4	1983	3638	8.8
5	1984	4041	11.1
6	1985	4347	7.6
7	1986	4590	5.6
8	1987	4870	6.1
9	1988	5253	7.9
10	1989	5658	7.7
11	1990	5980	5.7
12	1991	6174	3.3

Basic scatter plot

Script

```
ggplot(US_GDP,
       aes(x = Year,
           y = US_GDP_BN)) +
  geom_point()
```

- `aes` – tells R how to map / assign the data
- **`geom_point()`** – tells R to use points to display the data

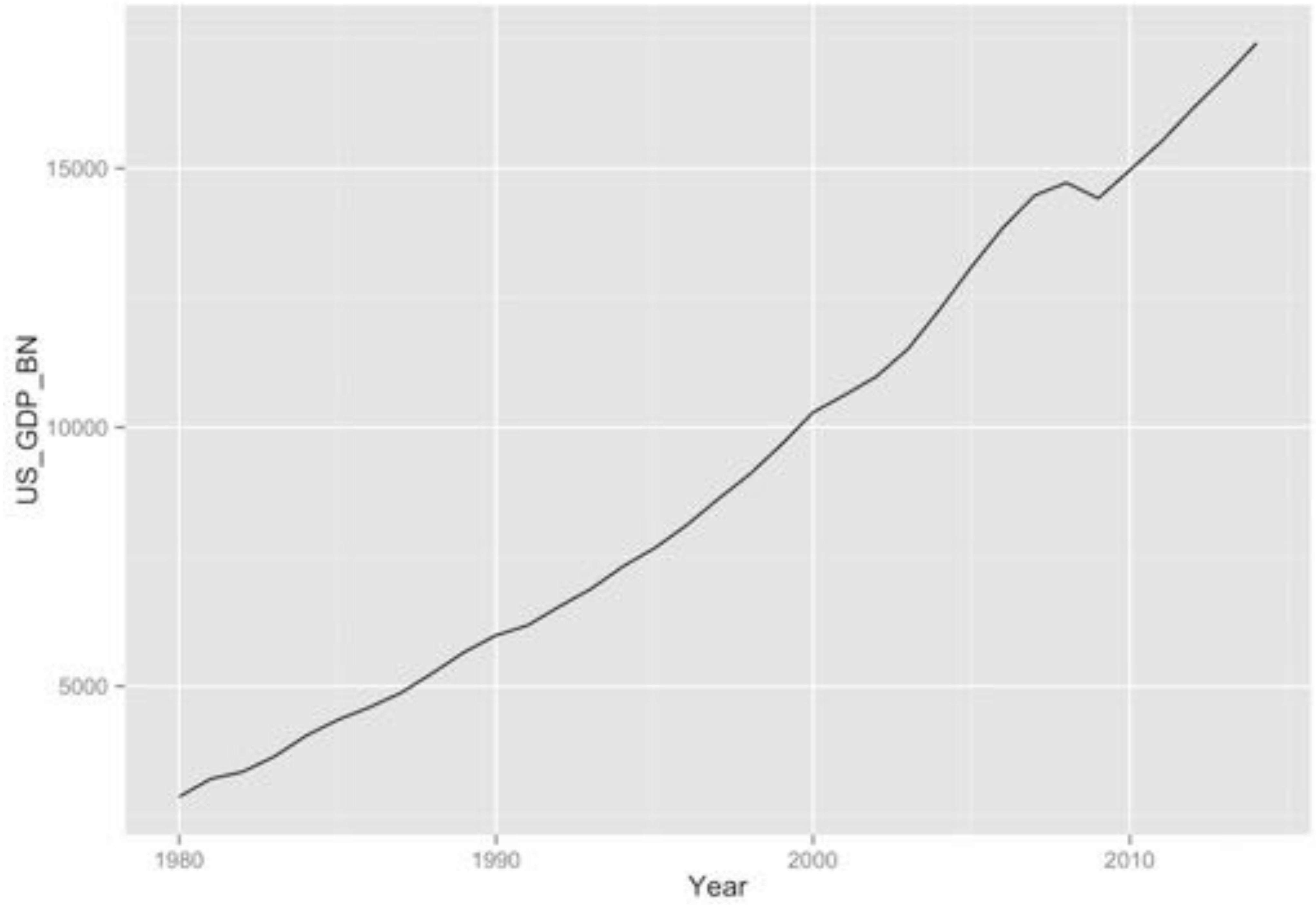


Basic line plot

Script

```
ggplot(US_GDP,
       aes(x = Year,
           y = US_GDP_BN)) +
  geom_line()
```

- `aes` – tells R how to map / assign the data
- **`geom_line()`** – tells R to use a line to display the data

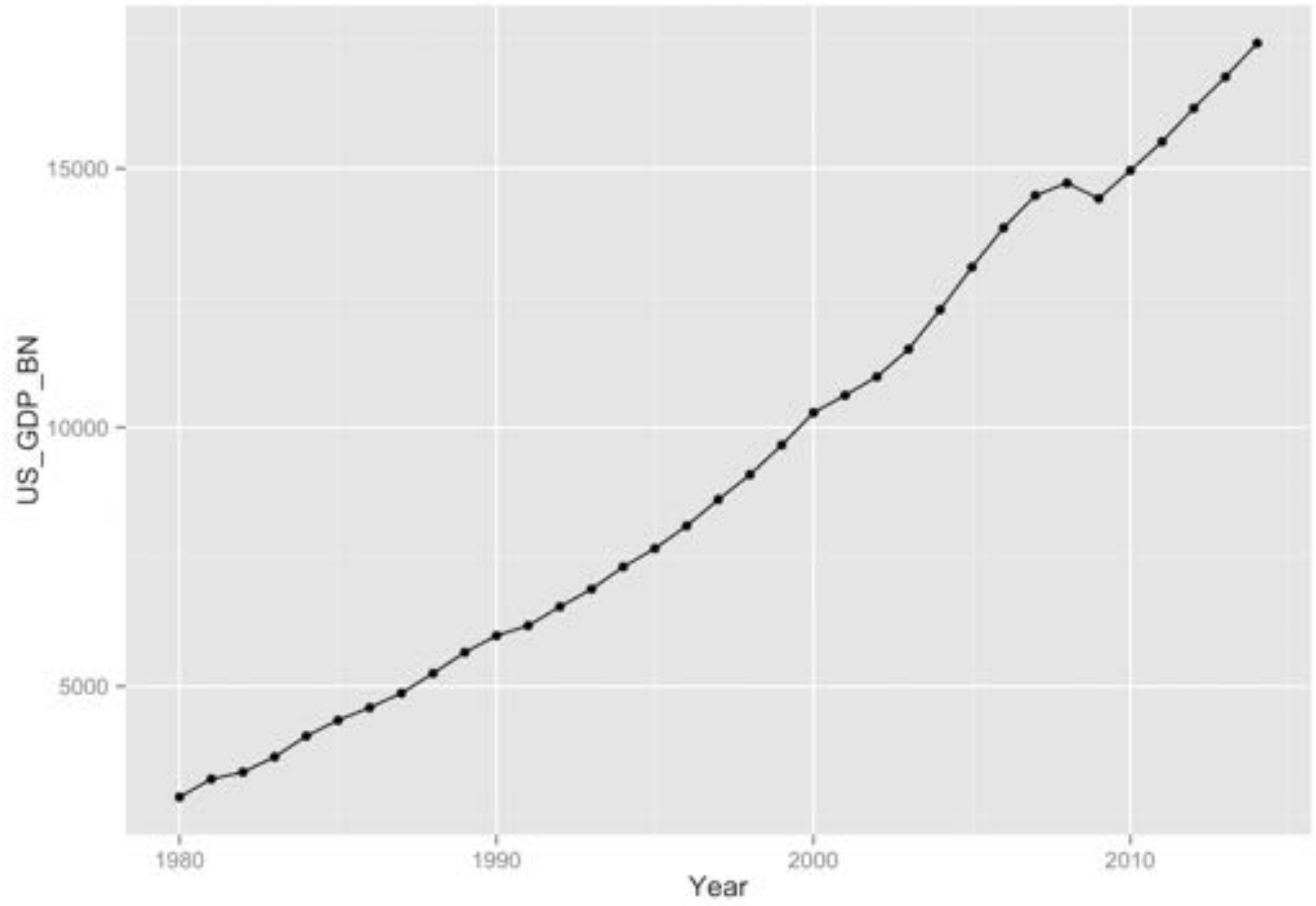


Scatter plot + line plot

Script

```
ggplot(US_GDP,
       aes(x = Year,
           y = US_GDP_BN)) +
  geom_line() +
  geom_point()
```

- **`geom_line()`** – tells R to use a line to display the data
- **`geom_point()`** – tells R to use points to display the data



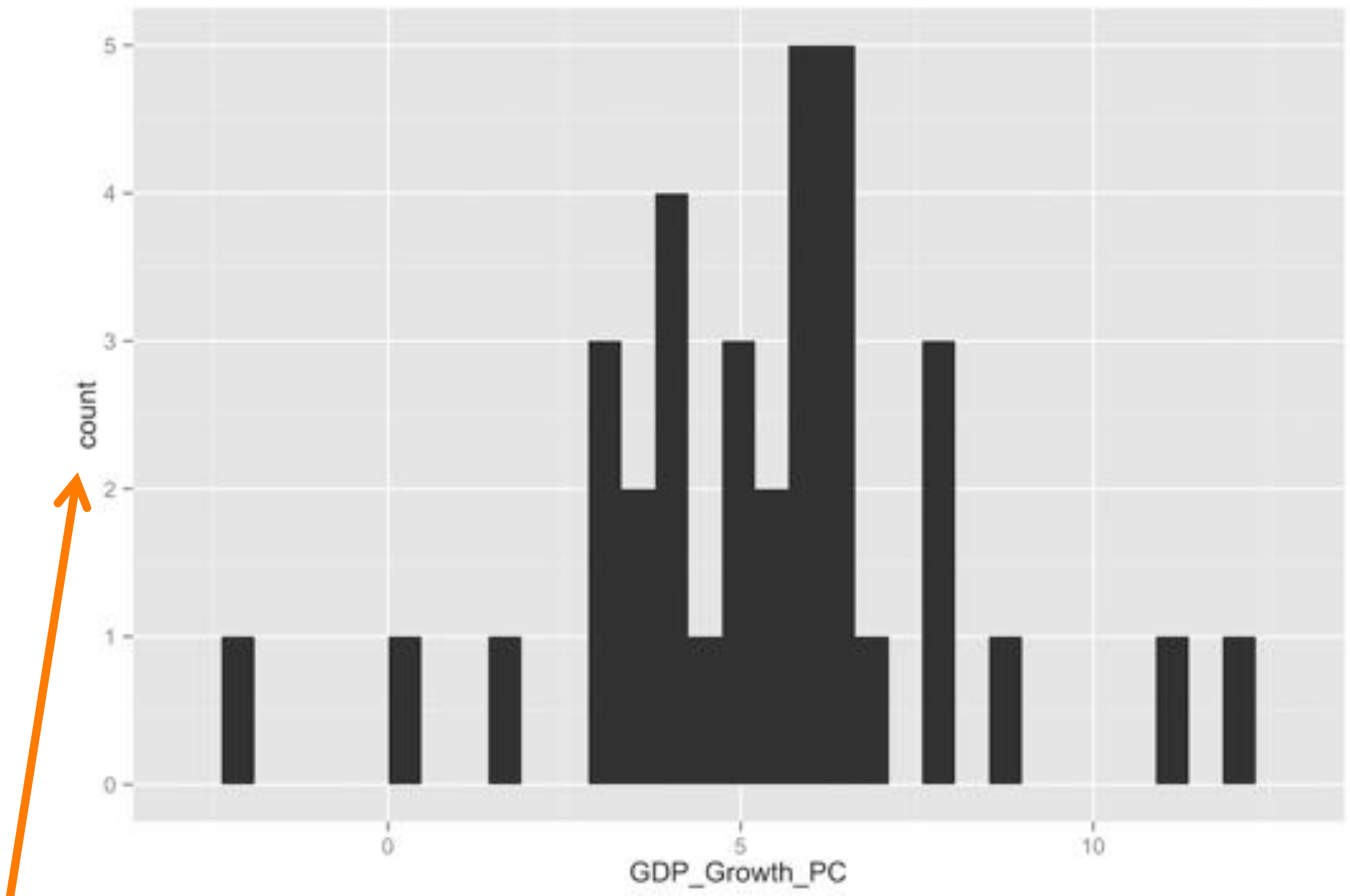
There are now 2 layers on the graph!

Basic histogram

Script

```
ggplot(US_GDP, aes(x = GDP_Growth_PC)) +
  geom_bar()
```

- `aes` – note that we're only including x-axis values
- **`geom_bar()`** – tells R to use bars to represent the data
- When given numeric data, the default setting of `geom_bar()` is to bin (put data into groups) the data and plot the number of instances each bin occurs -- a histogram



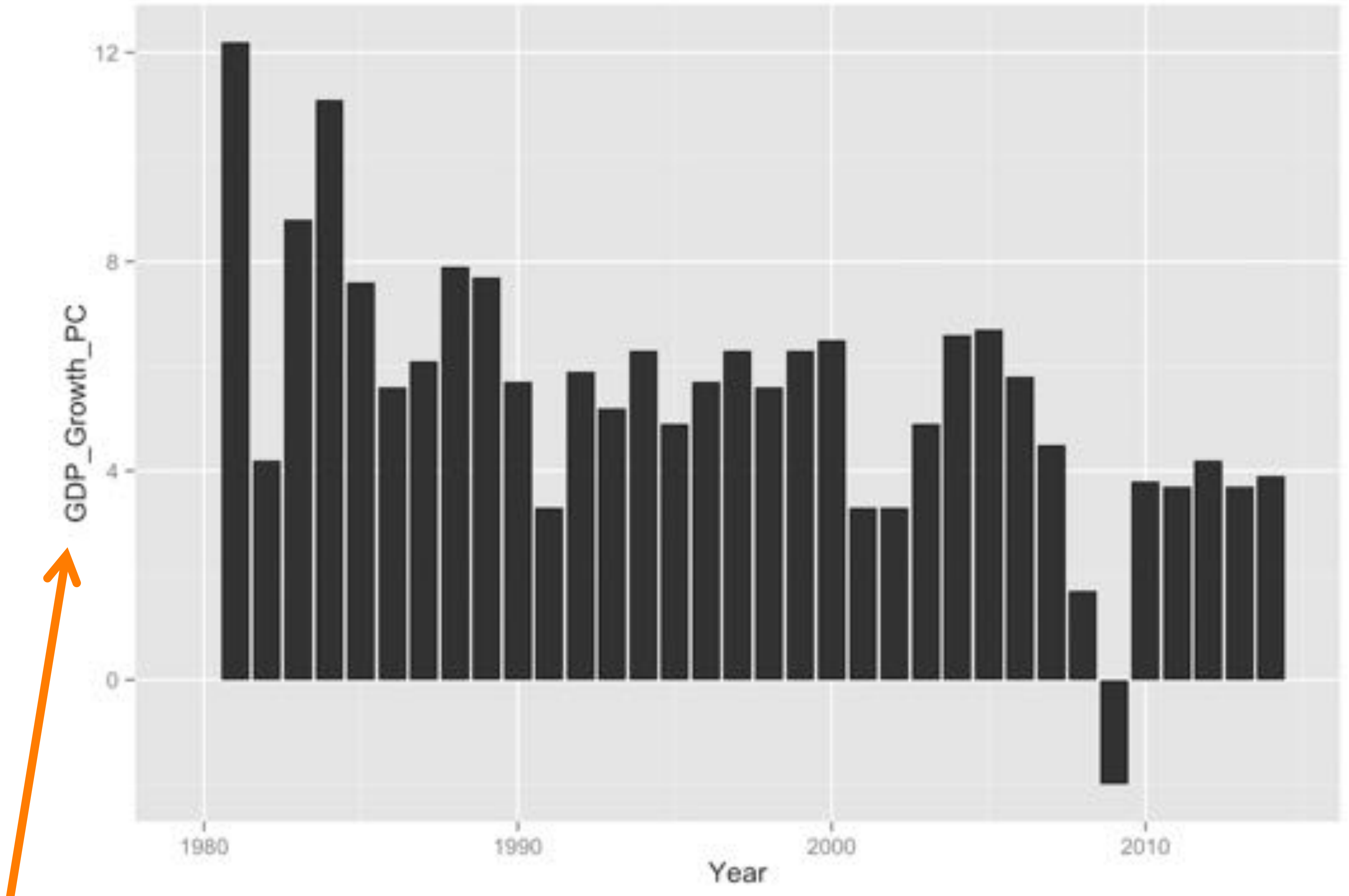
A histogram shows the frequency of occurrence of each value on the x-axis. This is a great way to see how often U.S. GDP grows at various rates.

Basic bar plot

Script

```
ggplot(US_GDP,
       aes(x = Year,
           y = GDP_Growth_PC)) +
  geom_bar(stat = "identity")
```

- **`geom_bar()`** – tells R to use bars to represent the data
- The default setting of `geom_bar()` plots the number of instances of a particular value on the x-axis
- To use the data to determine bar height, we need to tell R to use the `"identity"` statistic (`stat = "identity"`)



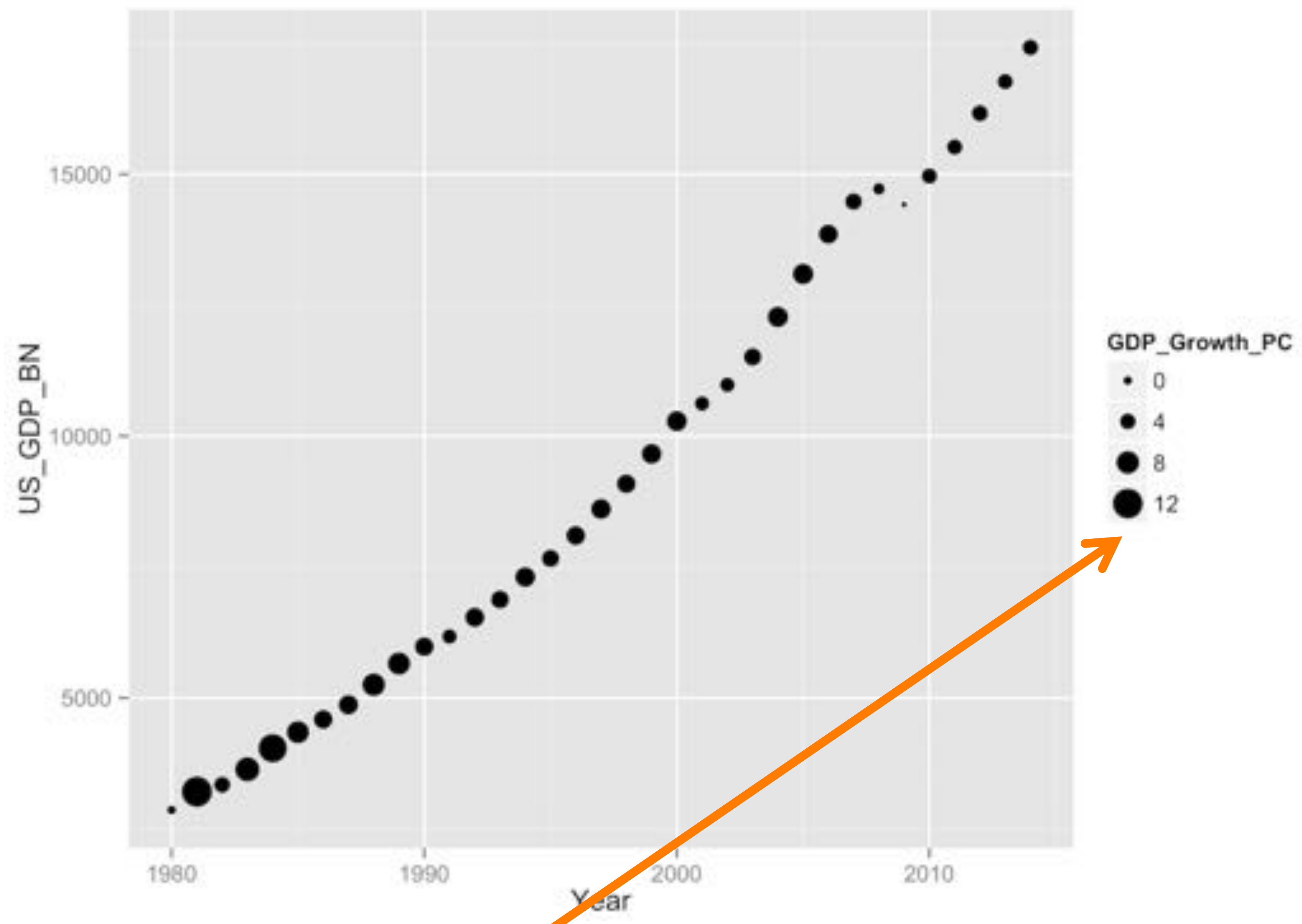
Percentage changes are usually best expressed with bars or area graphs, which are less likely to mislead your audience compared to a line graph

Bubble plot

Script

```
ggplot(US_GDP, aes(x = Year,
                   y = US_GDP_BN,
                   size = GDP_Growth_PC)) +
  geom_point()
```

- The **size** argument tells R to interpret the `GDP_Growth_PC` column as a third variable related to the size of the data markers
- **`geom_point()`** – tells R to use points to represent the data



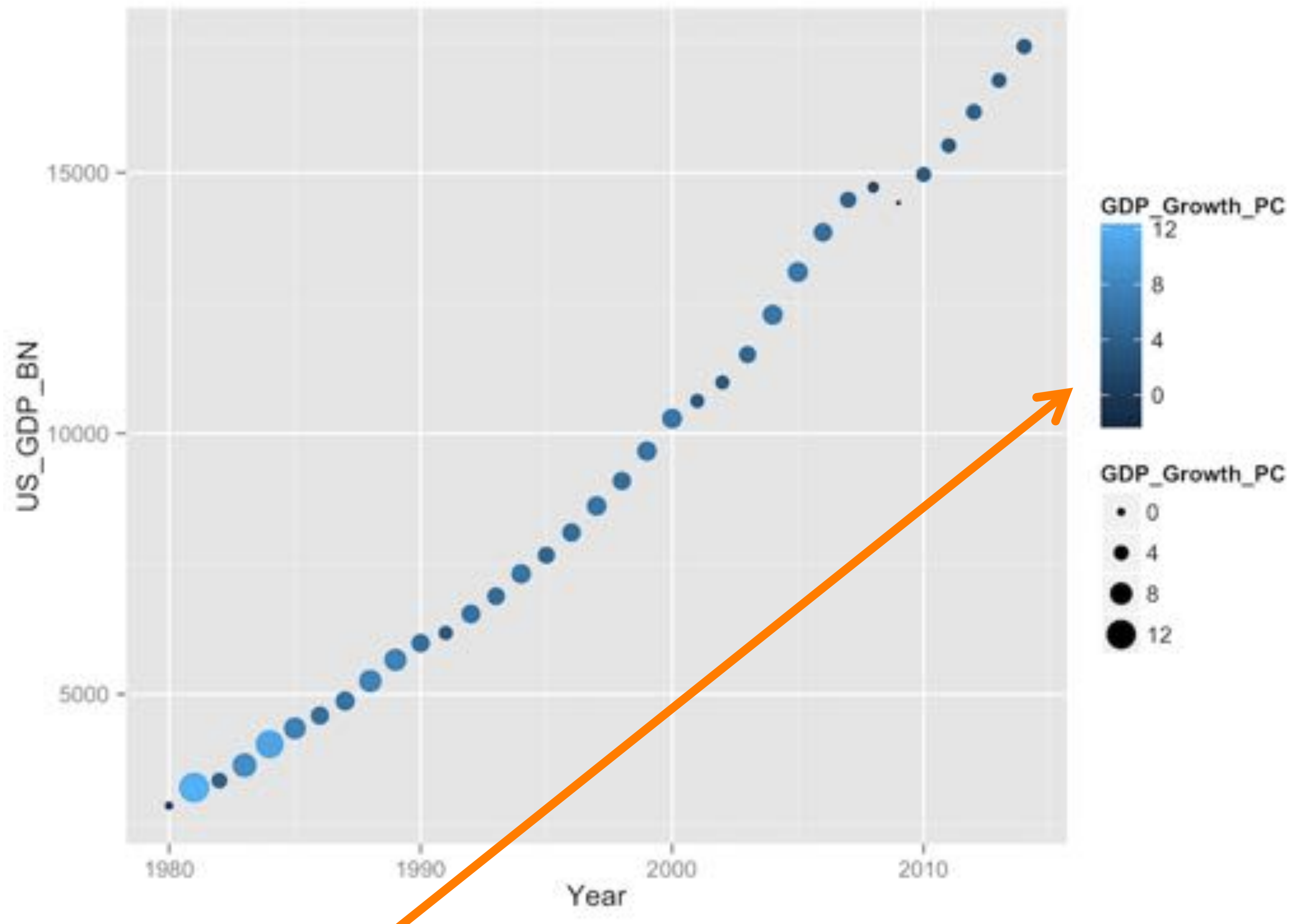
A bubble plot allows you to plot a 3rd variable on a 2D graph

Colored bubble plot

Script

```
ggplot(US_GDP, aes(x = Year,
                   y = US_GDP_BN,
                   size = GDP_Growth_PC,
                   color = GDP_Growth_PC)) +
  geom_point()
```

- The **color** argument tells R to interpret the GDP_Growth_PC column as a third variable distinguished by color
- geom_point() – tells R to use points to represent the data



The prior graph was harder to read because the points were so close in size. Adjusting the color makes the chart more legible.

Additional R resources

1. http://www.statmethods.net
2. http://ggplot2.org
3. https://plot.ly/r/
4. www.r-bloggers.com
5. cran.r-project.org/
6. *The Art of R Programming* by Norman Matloff
7. *R for Everyone* by Jared P. Lander
8. *R Graphics Cookbook* by Winston Chang

R for Everyone
Advanced Analytics and Graphics
JARED P. LA
THE ART OF R PROGRAMMING
A TOUR OF STATISTICAL SOFTWARE DESIGN
NORMAN MATLOFF
Practical Recipes for Visualizing Data
R Graphics Cookbook
O'REILLY
Winston Chang

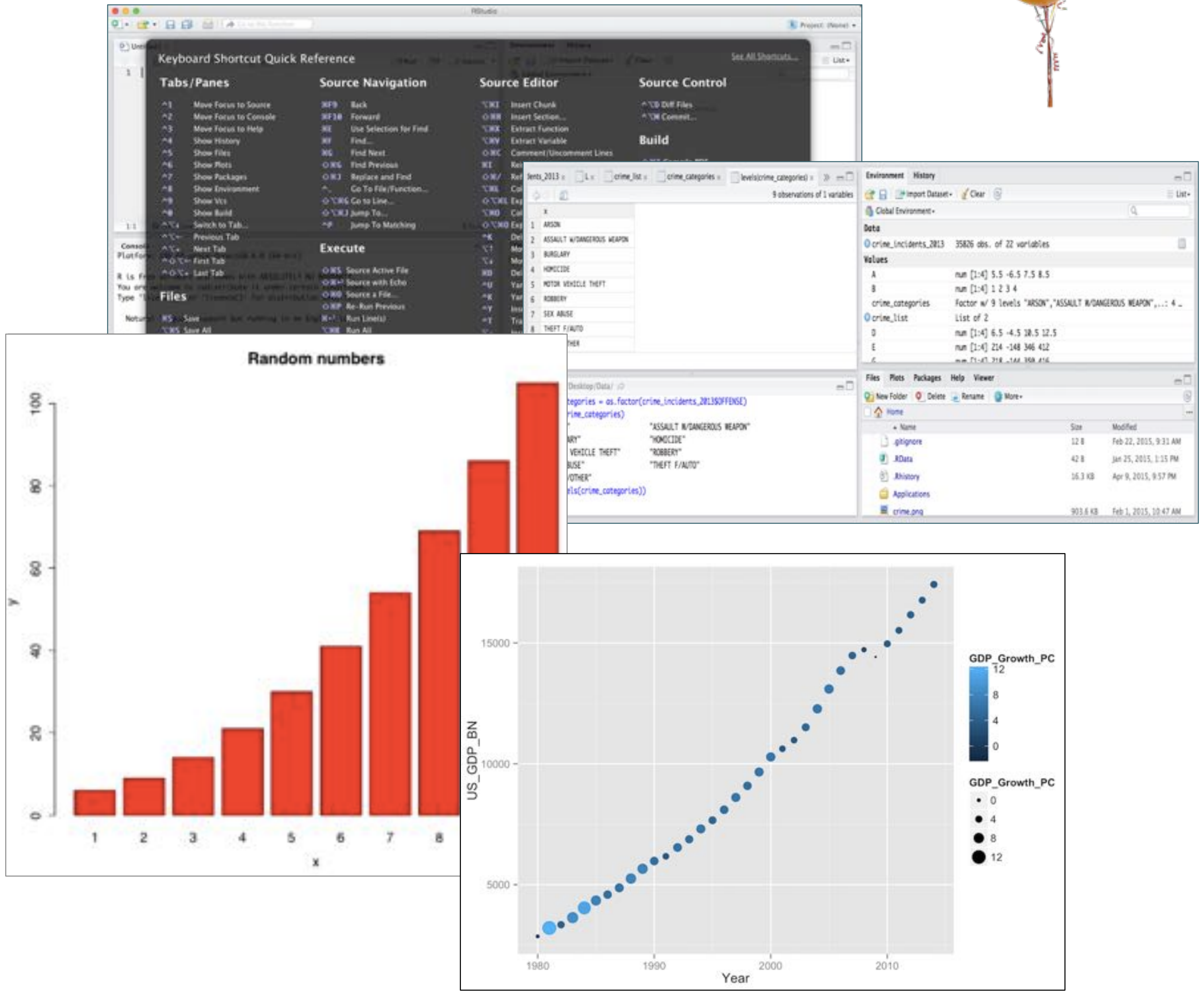
Additional data science resources

1. *Doing Data Science* by Cathy O'Neil & Rachel Schutt
2. *Data Science for Business* by Foster Provost & Tom Fawcett
3. *Data Smart* by John W. Foreman
4. *Mining the Social Web* by Matthew A. Russell
5. *Predictive Analytics* by Eric Siegel
6. *Analyzing the Analyzers* by Harlan Harris, Sean Murphy and Marck Vaisman
7. www.datasciencecentral.com
8. www.kdnuggets.com

O'REILLY
Doing Data Science
STRAIGHT TALK FROM THE FRONTLINE
Data Science for Business
What You Need to Know About Data Mining and Data-Analytic Thinking
O'REILLY
2nd Edition
Mining the Social Web
DATA MINING FACEBOOK, TWITTER, LINKEDIN, GOOGLE+, GITHUB, AND MORE
Matthew A. Russell
PREDICTIVE ANALYTICS
AN INTRODUCTION FOR EVERYONE
THE POWER TO PREDICT WHO WILL CLICK, BUY, LIE, OR DIE
ERIC SIEGEL
Using Data Science to Transform Information into Insight
John W. Foreman
DATA SMART
WILEY
O'REILLY
Analyzing the Analyzers
An Introspective Survey of Data Scientists and Their Work
Harlan D. Harris, Sean Patrick Murphy & Marck Vaisman

Congratulations!!

- ✓ You have learned how to code in R
- ✓ You have learned how to think about data
- ✓ You have learned how to format data
- ✓ You have learned how to build basic visualizations



Don't stop here!

Continue your learning with us at datasociety.co! Learn how to:

Automate data cleaning processes
Build beautiful interactive visualizations
Find new patterns and groups in your data
Predict trends from your data
Analyze networks and find influencers
Prioritize and organize thousands of text documents
Classify new data points to predict behavior

And much more!