

# R and CRAN

## An Introduction

# What is R ?

- R is a programming language and software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing.
- It has been widely used by statisticians, academicians and students for many decades.



# Why R ?

- R is the leading tool for statistics, data analysis, and machine learning.
- Not only is R free, but it's also open-source.
- R allows you to integrate with other languages (C/C++, Java, Python) and enables you to interact with many data sources
- It's platform-independent, so you can use it on any operating system.



# Evolution of R

- R is an implementation of the S programming language combined with lexical scoping semantics inspired by Scheme.
- Firstly, language S was created by John Chambers while at Bell Labs.
- R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and is currently developed by the R Development Core Team, of which John Chambers is a member.
- R is named partly after the first names of the first two R authors and partly as a play on the name of S.

• Courtesy: Wikipedia

# R as a Project

- R is a GNU project.
- The source code for the R software environment is written primarily in C, Fortran, and R.
- R is freely available under the GNU General Public License, and pre-compiled binary versions are provided for various operating systems.
- While R has a command line interface, there are several graphical front-ends available.

• Courtesy: Wikipedia

# Extensibility of R

- A rich collection of libraries are available for R functionality called packages.
- There are some packages provided by default for the basic installation of R
- If you want to add more functionality it can be extended by installing packages to your software environment.
- We can think of packages as add-ins of R.

# Programming Features of R

- R being interpreted language users access it using command line interpreter.
- R supports procedural programming like calling functions as well as object oriented programming like handling objects.
- R provides a matrix approach towards handling the data unlike the data step approach of SAS.

# CRAN

- The “Comprehensive R Archive Network” (CRAN) is a collection of sites which carry identical material, consisting of the R distribution(s), the contributed extensions, documentation for R, and binaries.
- There are several mirror sites to it.
- From CRAN, you can obtain the latest official release of R, daily snapshots of R, documentation to R etc.





*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](#)

## 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 (2015-12-10, Wooden Christmas-Tree) [R-3.2.3.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](#)

### Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.



*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](#)

# Installing R

R-3.2.3 for Windows (32/64 bit)

[Download R 3.2.3 for Windows](#) (62 megabytes, 32/64 bit)

[Installation and other instructions](#)

[New features in this version](#)

If you want to double-check that the package you have downloaded exactly matches the package distributed by R, you can compare the [md5sum](#) of the .exe to the [true fingerprint](#). You will need a version of md5sum for windows: both [graphical](#) and [command line versions](#) are available.

## Frequently asked questions

- [How do I install R when using Windows Vista?](#)
- [How do I update packages in my previous version of R?](#)
- [Should I run 32-bit or 64-bit R?](#)

Please see the [R FAQ](#) for general information about R and the [R Windows FAQ](#) for Windows-specific information.

## Other builds

- Patches to this release are incorporated in the [r-patched snapshot build](#).
- A build of the development version (which will eventually become the next major release of R) is available in the [r-devel snapshot build](#).
- [Previous releases](#)

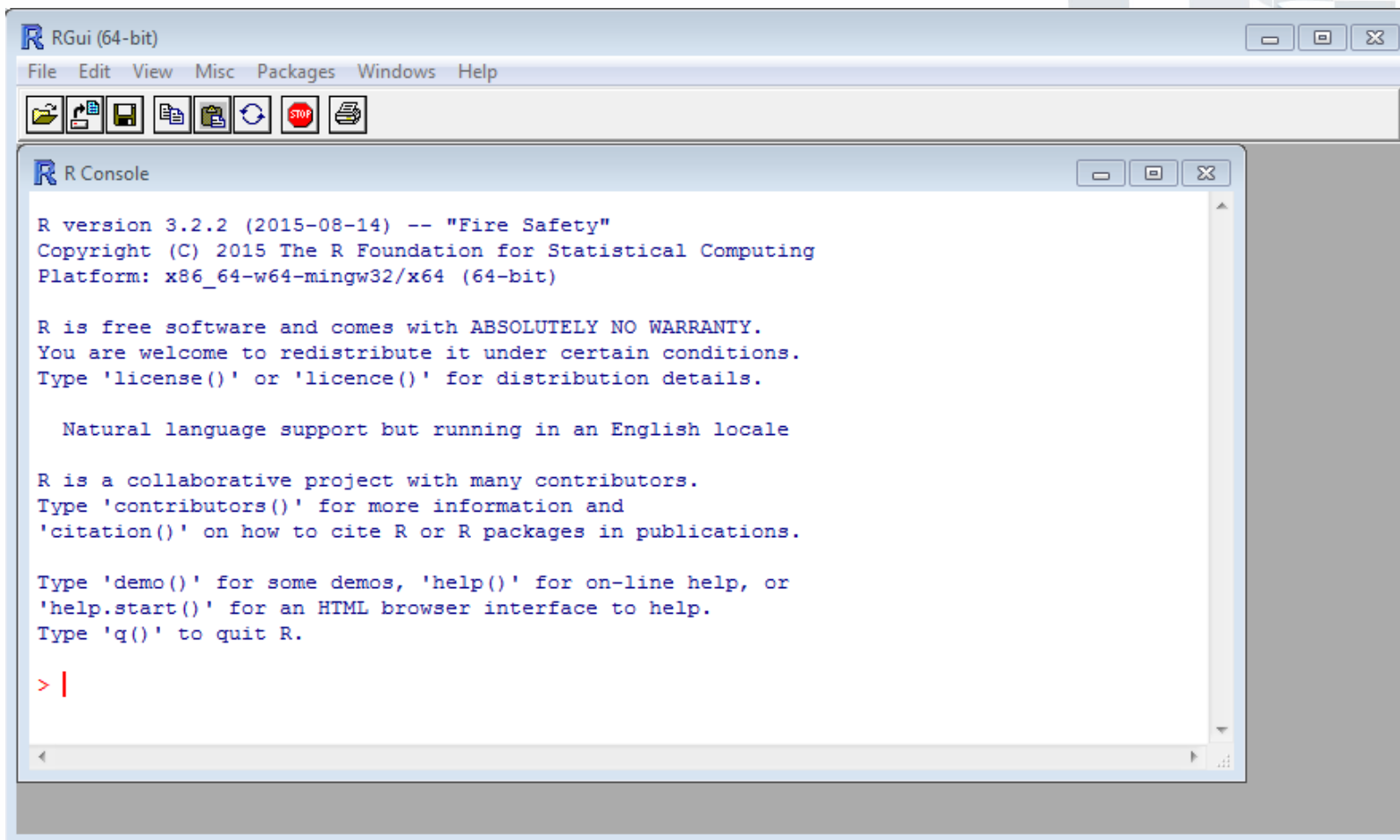
Note to webmasters: A stable link which will redirect to the current Windows binary release is [CRAN.MIRROR>/bin/windows/base/release.htm](http://CRAN.MIRROR>/bin/windows/base/release.htm).

---

Last change: 2015-12-10, by Duncan Murdoch

# R Command Line Interface

# R GUI



# Warming-Up with CLI

## Assigning Values to Variables

```
> a <- 9
> a
[1] 9
> b <- 7
> a+b
[1] 16
> c <- a * b
> c
[1] 63
> s <- "Analytics"
> s
[1] "Analytics"
```

**Up** and **down arrow keys** scroll through your command history.

## Assigning combination of values to Variables

```
> d <- c(1,3,56,17)
> d
[1] 1 3 56 17
```

```
> f <- c("z","r","p","UIJ")
> f
[1] "z" "r" "p" "UIJ"
```

For clearing the screen :  
**Ctrl + L**

# R Workspace

- R workspace is your current **R** working environment and includes all the objects and functions you have created.
- The user can anytime save an image of the current workspace that is automatically reloaded the next time **R** is started.

# Workspace Objects

Viewing list of objects in workspace

```
> ls()  
[1] "a" "b" "c" "d" "f" "s"
```

Saving workspace with name "first"

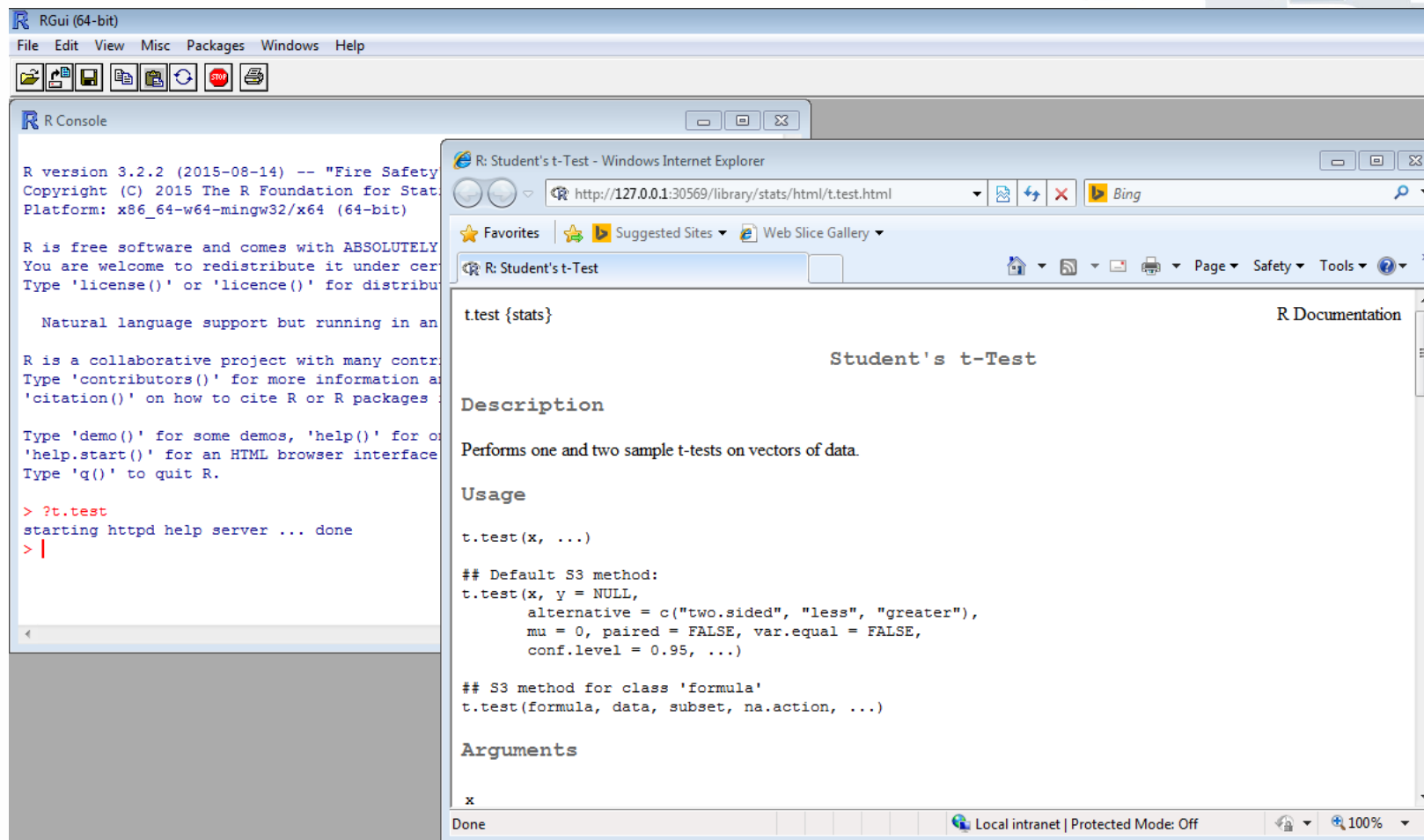
```
> save.image("D:\\first")
```

Loading workspace with name "first"

```
> load("D:\\first")
```

# Getting Help

For getting help for any function you need to type the name of the function after “?”



The screenshot shows the RGui (64-bit) interface with the R Console and a web browser window displaying the R documentation for the `t.test` function.

**R Console:**

```
R version 3.2.2 (2015-08-14) -- "Fire Safety"
Copyright (C) 2015 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.

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.

> ?t.test
starting httpd help server ... done
> |
```

**R: Student's t-Test - Windows Internet Explorer**

Address: <http://127.0.0.1:30569/library/stats/html/t.test.html>

Page Title: R: Student's t-Test

Content:

```
t.test {stats}
```

### 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'

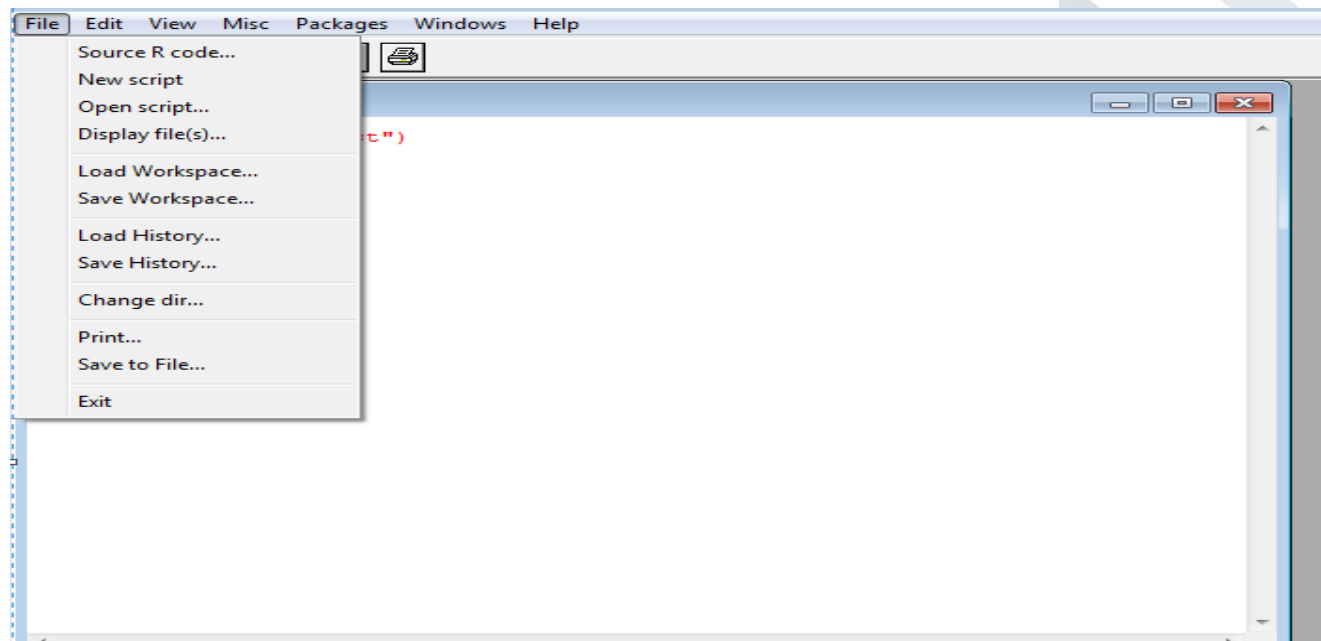
```
t.test(formula, data, subset, na.action, ...)
```

#### Arguments

x

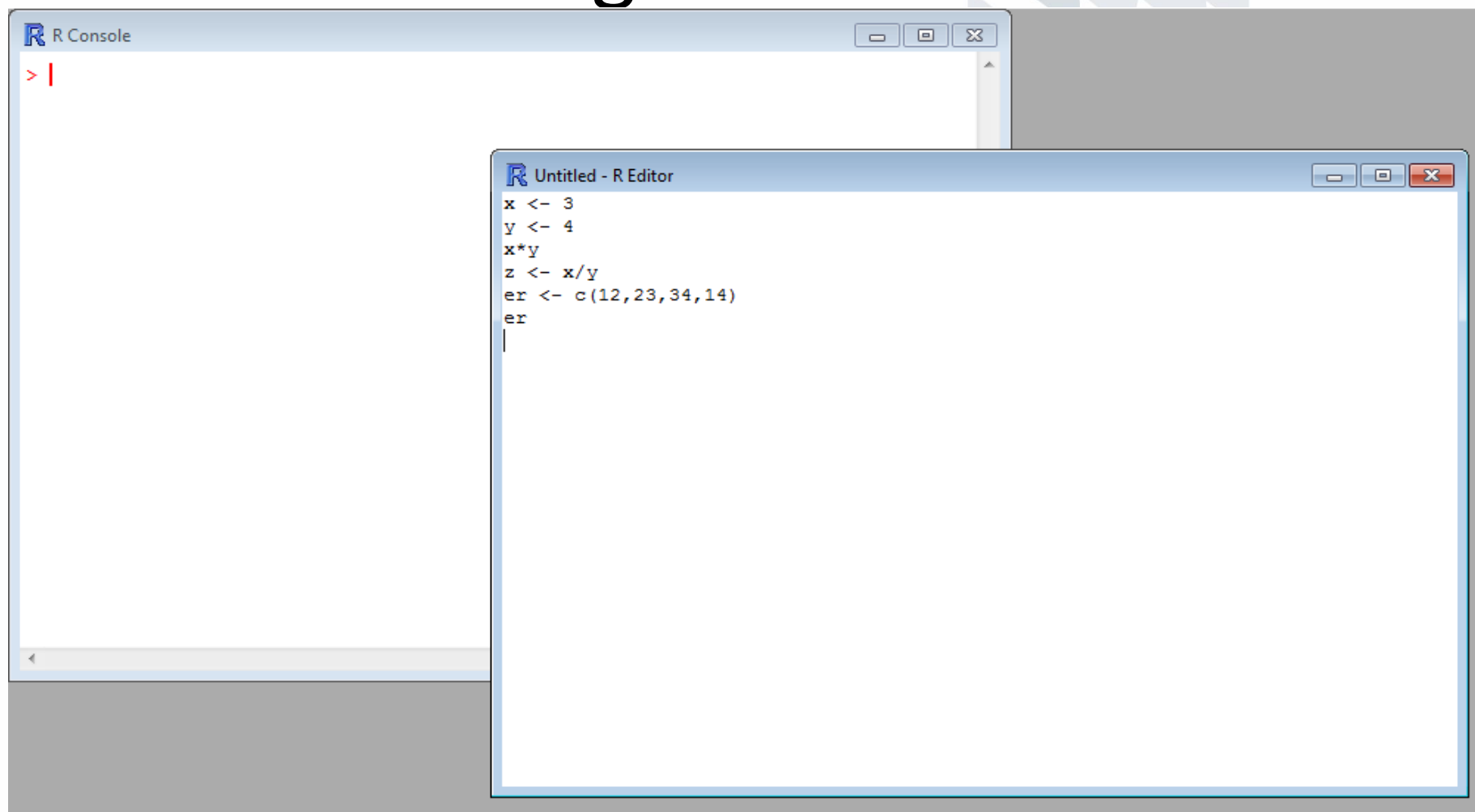


# Working with Program File

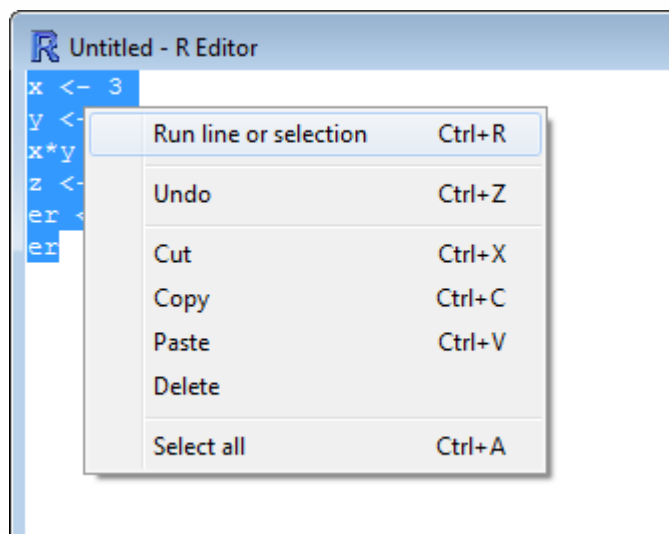


- Choose the option New Script to open the program editor

# Program Editor

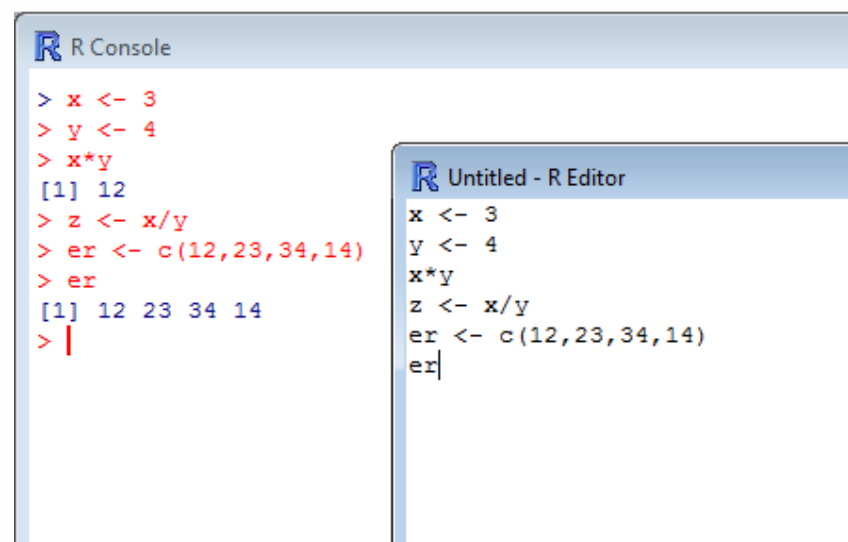


# Running the code



Alternatively, we can press **Ctrl + R** for running the code

The code executed is seen on the command line as line by line command execution.



# IDEs of R

- **R Studio**
- R Commander
- Tinn-R
- Eclipse with plug-in StatET
- VS Editor

# R Studio

## Introduction

# R Studio

- It is a free and open source integrated development environment (IDE) for R

The screenshot shows the RStudio interface with a script editor, environment pane, and console.

**Script Editor (naiveBayes.R):**

```
1 telecom <- read.csv("D:\\Statistics\\NB_K_NN\\Telecom.csv")
2
3 library(caret)
4
5 set.seed(333)
6 intrain <- createDataPartition(y=telecom$Response,p=0.7,list = FALSE)
7
8 training <- telecom[intrain, ]
9 testing <- telecom[-intrain,]
10
11 library(e1071)
12 classifier <- naiveBayes(training[,1:2], training[,3])
13
14 PredY <- predict(classifier, newdata=testing[,3], type="class")
15 PredYProb <- predict(classifier, newdata=testing[,3], type="raw")
16
17 PredY <- factor(PredY, levels = c("Y", "N"))
18 testing[,3] <- factor(testing[,3], levels = c("Y", "N"))
19
20 # PredYRaw <- predict(classifier, newdata=testing[,3], type="raw")
21
22 tbl <- table(PredY, testing[,3], dnn=list('predicted', 'actual'))
23
24 [Top Level]
```

**Environment Pane:**

Global Environment	
intrain	int [1:106, 1] 2 4 5 6 7 8 10 12 13 15 ...
PredYProb	num [1:44, 1:2] 0.0556 0.9424 0.4961 0.0556...
telecom	150 obs. of 3 variables
testing	44 obs. of 3 variables
training	106 obs. of 3 variables
values	
classifier	List of 4
PredY	Factor w/ 2 levels "Y","N": 1 2 1 1 1 1 2 1...
tbl	'table' int [1:2, 1:2] 22 1 8 13

**Files Pane:**

Name	Description	Version
acepack	ace() and avas() for selecting regression transformations	1.3-3.3
arules	Mining Association Rules and Frequent Itemsets	1.3-1
arulesViz	Visualizing Association Rules and Frequent Itemsets	1.1-0
assertthat	Easy pre and post assertions.	0.1
automap	Automatic interpolation package	1.0-14
BH	Boost C++ Header Files	1.58.0-1
bitops	Bitwise Operations	1.0-6
car	Companion to Applied Regression	2.1-0
caret	Classification and Regression Training	6.0-58
caTools	Tools: moving window statistics, GIF, Base64, ROC AUC, etc.	1.17.1

**Console:**

```
McNemar's Test P-value : 0.0455003
Sensitivity : 0.9565
Specificity : 0.6190
Pos Pred Value : 0.7333
Neg Pred Value : 0.9286
Prevalence : 0.5227
Detection Rate : 0.5000
Detection Prevalence : 0.6818
Balanced Accuracy : 0.7878
'Positive' Class : Y
```

The image shows the RStudio interface with several components highlighted by green callouts:

- Smart Code Editor:** Points to the main script editor window where the R code is written.
- Object List:** Points to the Environment pane on the right, which displays the objects in the current session.
- Console:** Points to the console window at the bottom left, which shows the output of the R code.
- Plot Tab:** Points to the 'Plots' tab in the bottom right pane.
- Package List:** Points to the 'Packages' tab in the bottom right pane, which lists installed and available packages.
- Help Tab:** Points to the 'Help' tab in the bottom right pane.

The R code in the Smart Code Editor is as follows:

```
1 telecom <- read.csv("D:\\Statistics\\NB_k_NN\\Telecom.csv")
2
3 library(caret)
4
5 set.seed(333)
6 intrain <- createDataPartition(y=telecom$Response,p=0.7,list = FALSE)
7
8 training <- telecom[intrain, ]
9 testing <- telecom[-intrain,]
10
11 library(e1071)
12 classifier <- naiveBayes(training[,1:2], training[,3])
13
14 PredY <- predict(classifier, newdata=testing[,3], type="class")
15 PredYProb <- predict(classifier, newdata=testing[,3],type="raw")
16
17 PredY <- factor(PredY,levels = c("Y","N"))
18 testing[,3] <- factor(testing[,3],levels = c("Y","N"))
19
20 # PredYRaw <- predict(classifier, newdata=testing[,3], type="raw")
21
22 tbl <- table(PredY, testing[,3], dnn=list("predicted","actual"))
23
19:1 (Top Level) ↕
```

The console output shows the McNemar's Test P-Value and various performance metrics:

```
McNemar's Test P-Value : 0.0455003

Sensitivity : 0.9565
Specificity : 0.6190
Pos Pred Value : 0.7333
Neg Pred Value : 0.9286
Prevalence : 0.5227
Detection Rate : 0.5000
Detection Prevalence : 0.6818
Balanced Accuracy : 0.7878

'Positive' class : Y

> |
```

The Environment pane (Object List) shows the following objects:

Object	Class	Dimensions	Summary
intrain	int	[1:106, 1]	2 4 5 6 7 8 10 12 13 15 ...
PredYProb	num	[1:44, 1:2]	0.0556 0.9424 0.4961 0.0556...
telecom	Data Frame	150 obs. of 3 variables	
testing	Data Frame	44 obs. of 3 variables	
training	Data Frame	106 obs. of 3 variables	
classifier	List	List of 4	
PredY	Factor	w/ 2 levels "Y","N"	1 2 1 1 1 1 1 2 1...
tbl	table	int [1:2, 1:2]	22 1 8 13

The Packages pane (Package List) shows the following installed packages:

Package	Description	Version
acepack	ace() and ace2() for selecting regression transformation	1.3-3.3
arules	Mining Association Rules and Frequent Itemsets	1.3-1
arulesViz	Visualizing Association Rules and Frequent Itemsets	1.1-0
brglm2	Easy pre- and post-processed	0.1
brglm2	Automatic	1.0-14
brglm2	Boost C++	1.58.0-1
bitops	Bitwise Operations	1.0-6
car	Companion to Applied Regression	2.1-0
caret	Classification and Regression Training	6.0-58
caTools	Tools: moving window statistics, GIF, Base64, ROC AUC, etc.	1.17.1