

Introduction to R

March, 2022

A data science workflow

1. Retrieve data

- Generate
- Download

1. Store data

- Files
- Databases

3. Process data

- Preparation
- Analysis, Visualisation

4. Publish results

- Reports, Memos, Papers, Books
- Replication Data

What is programming?

Programming is about writing **programs**

- Sets of instructions (algorithms) to be executed by a computer
- written in a computer language (source code, or just code)
- stored in files (and organized in software projects)

Integrated Development Environments (IDEs)

Editor + Console + Tools

What is programming?

How many computer languages exist?

Why are there so many languages?

- Different needs: computation, visualization, robotisation...
- Different hardware: laptops, robots, fridges...
- Versions: improvements, new needs...
- Programming paradigms/ categorizations

Programming concepts are mostly language-agnostic

R

```
library(readr)
mydata <- read_csv("myfile.csv")
```

Python

```
import pandas as pd
mydata = pd.read_csv("myfile.csv")
```

Operators

Arithmetic Operators

- Addition, Subtraction, Multiplication, Division

Comparison Operators

- Equal, Not equal, Greater than, , Less than, Less or equal than

Boolean Operators

- and, or, not

Precedence of Operators

- Parentheses, Exponents and roots, Multiplications and divisions, Additions and subtractions
- Horizontal: left to right
- Vertical: top down

Data Types

Numeric

- integer, floating point, complex number

String

- sequence of characters

Boolean

- True/False, 1/0

Structured (combinations of the others)

- Vectors, Lists Arrays, Dictionaries, Dataframes

Functions are reusable chunks of code

- Each language has a set of **built-in** functions or **commands**

Control flow

Conditional statements

- if-then(-else)

Loop constructs

- for
- do while
- while

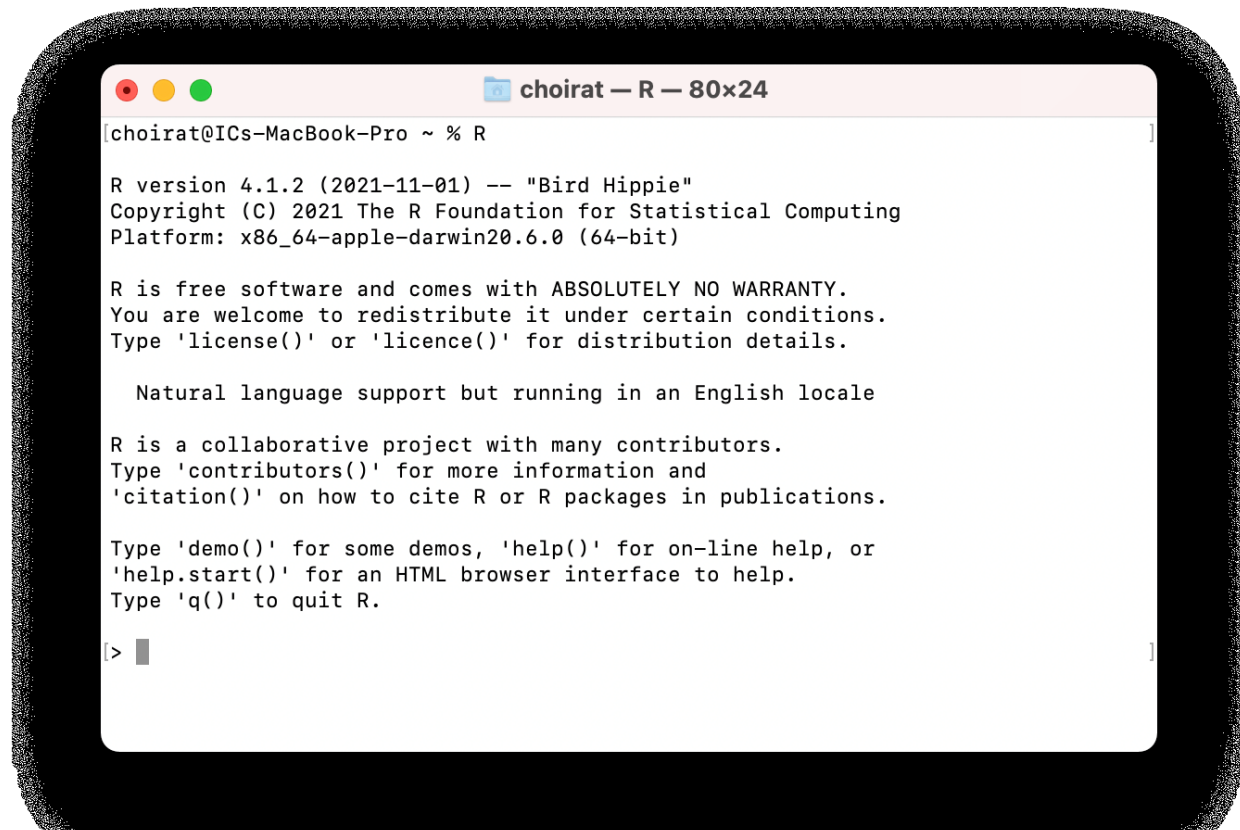
S, Splus, R

R is an implementation of the [S programming language](#) [...]. S was created by [John Chambers](#) in 1976 while at [Bell Labs](#). A commercial version of S was offered as [S-PLUS](#) starting in 1988. Many codes written for S-PLUS run unaltered in R.

In 1991 [Ross Ihaka](#) and [Robert Gentleman](#) at the [University of Auckland](#), New Zealand, embarked on an S implementation, independent of [S-PLUS](#). They began publicizing it in 1993. It was named partly after the first names of the first two R authors and partly as a play on the name of S. In 1995, Martin Maechler convinced Ihaka and Gentleman to make R [free and open-source software](#) under the [GNU General Public License](#).

(Source: [Wikipedia](#))

What does R look like?



```
[choirat@ICs-MacBook-Pro ~ % R]

R version 4.1.2 (2021-11-01) -- "Bird Hippie"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin20.6.0 (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.

[> ]
```

RStudio

A software company

RStudio

Software company



rstudio.com

Founder: [Joseph J. Allaire](#)

Founded: 2009

Headquarters location: [Boston, Massachusetts, United States](#)

Ports

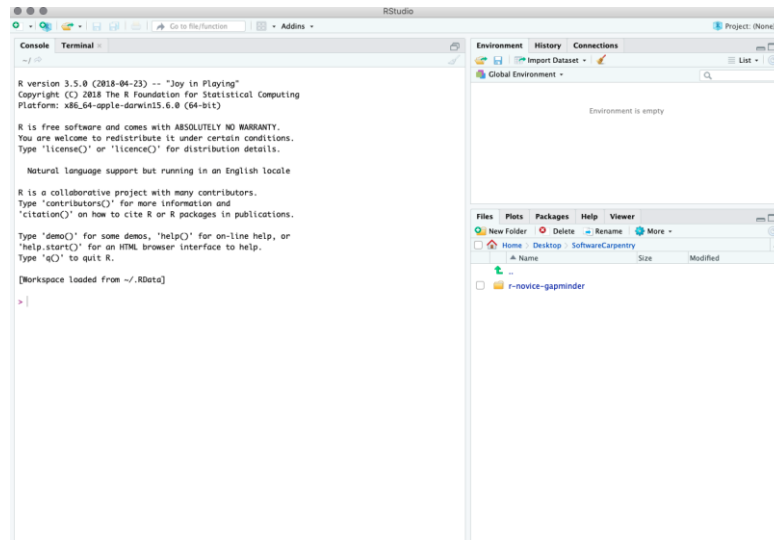
Revenue

Mission

RStudio's mission is **to create free and open-source software for data science, scientific research, and technical communication.**

<https://www.rstudio.com> › [about](#)

An Integrated Development Environment (IDE)



Alternatives to a local installation



+



- Computing power and storage are provided online by remote servers.
- Offers tools for state of the art research reproducibility and dissemination standards.

First steps with Renku



What is Renku?

Renku (連句 “linked verses”), is a Japanese form of popular collaborative linked verse poetry, written by more than one author working together.

The Renku Project is a platform that bundles together various tools for reproducible and collaborative data analysis projects. It is aimed at independent researchers and data scientists as well as labs, collaborations, and courses and workshops. Renku can be used by anyone who deals with data, whether they are a researcher, data analyst, project owner, or data provider.

(Source: [Renku documentation](#))



ETH zürich **EPFL**

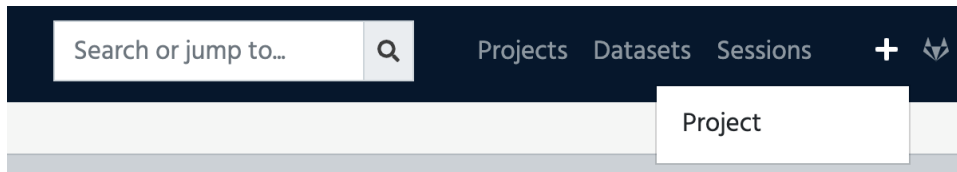
<https://datascience.ch/>

Demo

We give a quick tour, a demo, and then it's your turn!

<https://renkulab.io/>

1. Create an account
2. Login
3. Create an R project
4. Open RStudio



New project


Title

myfirstproject

 There are a few reserved names you cannot use.

Namespace 

cchoirat

 Group namespaces may restrict the visibility options.

Identifier

cchoirat/myfirstproject

 This is automatically derived from Namespace and Title.

Visibility

Public



Template source

RenkuLab

Custom

Template

Renku / Basic R (4.0.5) Project



The simplest R-4.0.5-based renku project with a basic directory structure and necessary supporting files.

Create project



README.md

myfirstproject

Introduction

This is a Renku project - basically a git repository with some bells and whistles. You'll find we have already created some useful things like `data` and `notebooks` directories and a `Dockerfile`.

Working with the project

The simplest way to start your project is right from the Renku platform - just click on the `Environments` tab and start a new session. This will start an interactive environment right in your browser.

To work with the project anywhere outside the Renku platform, click the `Settings` tab where you will find the git repo URLs - use `git` to clone the project on whichever machine you want.

myfirstproject



cchoirat/myfirstproject



[Overview](#) [Collaboration](#) [Files](#) [Datasets](#) [Sessions](#) [Settings](#)

⚙️ [Back to sessions list](#)

Starting session

Checking existing sessions...



myfirstproject



cchoirat/myfirstproject

Start

Overview Collaboration Files Datasets Sessions Settings

Back to sessions list

Branch [master](#)

Commit [8fccde6b](#)

Resources 0.25 cpu | 1G memory | 1G storage

Running since 1 minute ago

Open



File Edit Code View Plots Session Build Debug Profile Tools Help

rstudio

Go to file/function

myfirstproject



Console

Terminal

Jobs

~/work/myfirstproject/

R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (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

Environment

History

Connections

Git

Tutorial

Import Dataset

Global Environment

Environment is empty

Files

Plots

Packages

Help

Viewer

New Folder

Upload

Delete

Rename

More

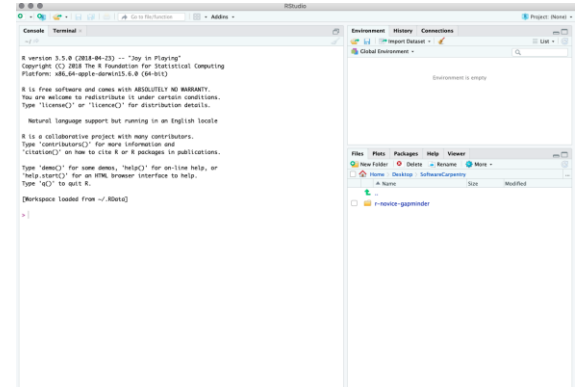
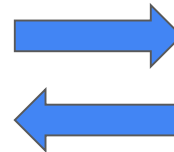
git under the hood



Cloud storage

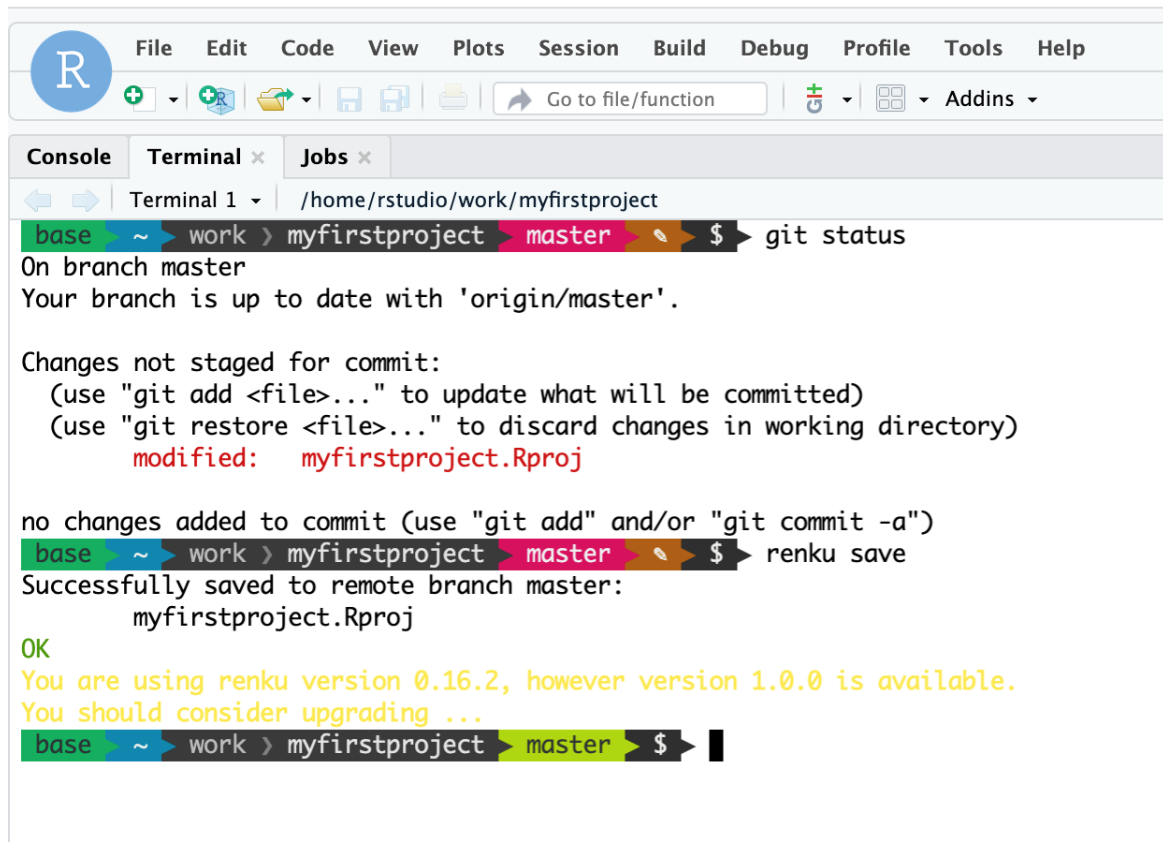


Virtual Machine



Web Browser (e.g., Edge)

renku save



The screenshot shows the RStudio application window. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu bar is a toolbar with icons for file operations and a search bar labeled 'Go to file/function'. The main workspace area is divided into three tabs: Console, Terminal, and Jobs. The Terminal tab is active, showing a terminal window titled 'Terminal 1' with the path '/home/rstudio/work/myfirstproject'. The terminal output shows the following commands and results:

```
base ~ work > myfirstproject master $ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   myfirstproject.Rproj

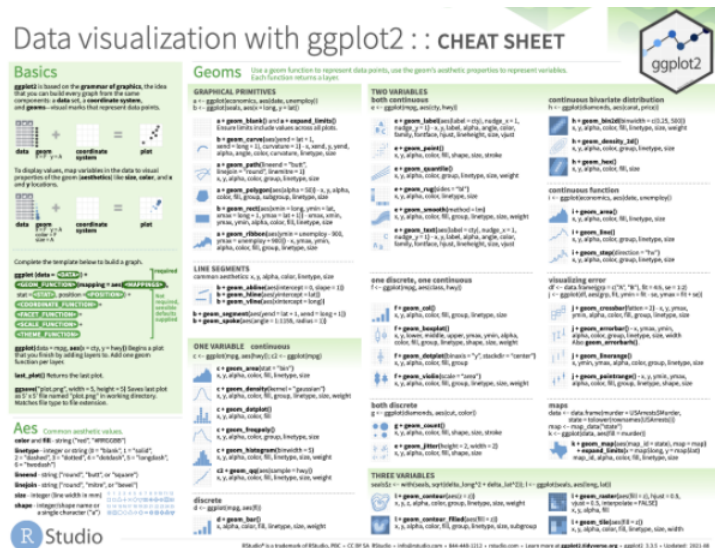
no changes added to commit (use "git add" and/or "git commit -a")
base ~ work > myfirstproject master $ renku save
Successfully saved to remote branch master:
    myfirstproject.Rproj
OK
You are using renku version 0.16.2, however version 1.0.0 is available.
You should consider upgrading ...
base ~ work > myfirstproject master $
```

Demo

Ressources and help

- Rstudio cheatsheets:
<https://www.rstudio.com/resources/cheatsheets/>

- Help from communities websites, e.g., Stackoverflow: <https://stackoverflow.com/>



ggplot add line to scatter

▲

3

⌵

⌵

I have "observed data" which are classical (X, Y) points. X, Y, Z are saved in vectors named X, Y, Z . Typically 10-30 (data).

Then I have "model data": which is a set of (X, Y) points creating a smooth line (I specially made regression line). I used `lm()` and `plot()` to create a smooth line. I added 1000 points to create extra smooth line.

How to scatterplot "observed data" and then plot "result" line on top of this?

Code for scatterplotting "observed data" works well:

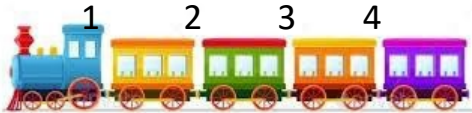
```
library(ggplot2)

data <- c(1,2,3,4,5,6,7,8,9,10,11,12)
y <- c(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1)
model <- lm(y ~ data)
model1 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model2 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model3 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model4 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model5 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model6 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model7 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model8 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model9 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model10 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model11 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model12 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model13 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model14 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model15 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model16 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model17 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model18 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model19 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model20 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model21 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model22 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model23 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model24 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model25 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model26 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model27 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model28 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model29 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model30 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model31 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model32 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model33 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model34 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model35 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model36 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model37 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model38 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model39 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model40 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model41 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model42 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model43 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model44 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model45 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model46 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model47 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model48 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model49 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model50 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model51 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model52 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model53 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model54 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model55 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model56 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model57 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model58 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model59 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model60 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model61 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model62 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model63 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model64 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model65 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model66 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model67 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model68 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model69 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model70 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model71 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model72 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model73 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model74 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model75 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model76 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model77 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model78 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model79 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model80 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model81 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model82 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model83 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model84 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model85 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model86 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model87 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model88 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model89 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model90 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model91 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model92 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model93 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model94 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model95 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model96 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model97 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model98 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model99 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model100 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model101 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model102 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model103 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model104 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model105 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model106 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model107 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model108 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model109 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model110 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model111 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model112 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model113 <- c(1,1,1,1,1,1,1,1,1,1,1,1)
model114 <- c(1,1,1,1,1,1,1,1,1
```

Vectors

- Are a collection of multiple pieces of information. You can think of them as carriages linked together in a train.
- They can be made of numeric, character or factor data
- But each vector can only contain one type of data
- Creating a vector is easy – use the `c` function with brackets and separate the pieces of data using commas: `x <- c(9,19,200,30,45)`

numeric



character



factor



Lists

- Like vectors are a collection of multiple pieces of information.
- They can be made up of individual pieces of data or collections of data: vectors, data frames, matrices, other lists
- Unlike vectors, lists can contain many types of data
- Example: one list made up of 3 vectors:

numeric



character



factor



Matrices & Data Frames

Store data in **rows** and **columns**:

therefore are **2** dimensional (like tables – this means I need both the row and column number to find one piece of info!)

	Column 1	Column 2	Column 3	Column 4	Column 5
Row 1					
Row 2					
Row 3					
Row 4					