Introductie R & Rstudio

Robin Khalfa

2025-05-20

Inleiding

R is een krachtige en veelzijdige programmeertaal die specifiek ontwikkeld werd voor statistische data analyse en datavisualisatie. Door zijn *open source* en uitgebreid ecosysteem aan packages wordt R wereldwijd gebruikt in academisch onderzoek, overheidsinstellingen, private bedrijven en in de industrie. R biedt ondersteuning voor zowel eenvoudige beschrijvende statistiek als voor het ontwikkelen en toepassen van complexe statistische modellen en algoritmes, waardoor het bijzonder geschikt is voor verschillende datawetenschappelijke toepassingen.

RStudio daarentegen is een geïntegreerde ontwikkelomgeving (IDE) die ontworpen is om 'werken met R' te vergemakkelijken. Het biedt een gebruiksvriendelijke interface, ingebouwde hulpmiddelen voor scriptbeheer, versiecontrole, visualisatie en interactieve rapportage.

Dit document biedt een introductie tot de basisfunctionaliteiten van R en RStudio. Aan de hand van concrete voorbeelden wordt getoond hoe men data kan inladen, manipuleren, visualiseren en analyseren. De nadruk ligt op het verwerven van praktische vaardigheden die de gebruiker in staat moet stellen om zelfstandig analyses uit te voeren en resultaten op een transparante manier te rapporteren.

De structuur van dit document is als volgt:

1. **Gebruik van packages**
   In dit deel wordt uitgelegd hoe packages geïnstalleerd, geladen en beheerd worden. Er wordt aandacht besteed aan CRAN, `install.packages()`, en `library()`.

2. **Working directory**
   Hier wordt besproken hoe men de working directory instelt, controleert en aanpast, en waarom dit belangrijk is voor het beheren van bestanden en datasets.

3. **R basics**
   In deel 3 geven we een overzicht van basiselementen m.b.t. programmeren in R, waaronder soorten variabelen, datatypes, operatoren, en functies.

4. **Werken met data in R**
   Hier behandelen we het inladen en manipuleren van datasets. Er wordt gewerkt met `readr, dplyr` en `tidyr` uit het tidyverse-ecosysteem.

5. **Graphics en visualisaties**
   Introductie tot datavisualisatie in R, met focus op `ggplot2`. Zowel eenvoudige plots als meerlagige grafieken komen aan bod.

6. **Geavanceerde functies**
   In deze 'gevorderde sectie' wordt ingegaan op het schrijven van functies en het gebruik van loops in R.

---

1. Gebruik van packages

R biedt via zijn package-ecosysteem toegang tot een enorm scala aan aanvullende functionaliteiten. Packages kunnen afkomstig zijn van verschillende bronnen, waaronder:

- **CRAN**: het centrale distributieplatform voor R-packages.
- **Bioconductor**: gespecialiseerd in bioinformatica en statistische genomica.
- **GitHub**: vaak gebruikt voor experimentele of in ontwikkeling zijnde packages.

1.1 Installatie van packages via CRAN

De meeste packages zijn beschikbaar via het CRAN-netwerk. Ze worden als volgt geïnstalleerd:

Een package moet na installatie telkens opnieuw worden geladen in een nieuwe R-sessie:

Meerdere packages tegelijk installeren kan als volgt:

1.2 Installatie van packages via Bioconductor (optioneel, ter info, niet uitvoeren)

Bioconductor is een apart platform voor packages gericht op computationele biologie en statistische genomica. Voor toegang tot Bioconductor gebruikt men het `BiocManager` package:

Bioconductor-packages worden ook onderhouden via een centrale infrastructuur en zijn onderhevig aan kwaliteitscontrole.

1.3 Installatie van packages via GitHub (optioneel, ter info, niet uitvoeren)

Sommige packages zijn (nog) niet beschikbaar op CRAN of Bioconductor, maar worden gedeeld via GitHub. Hiervoor gebruikt men het `devtools` of `remotes` package:

```
# Eerst devtools installeren indien nodig
#if (!require("remotes")) {
  #install.packages("remotes")
#}

# Installatie vanaf GitHub
#remotes::install_github("hadley/emo")
#library(emo)
```

Let op: GitHub-packages zijn niet noodzakelijk stabiel en zijn vaak in ontwikkeling.

1.4 Controle en beheer

Om te controleren of een package reeds geïnstalleerd is:

```
# if (!requireNamespace("dplyr", quietly = TRUE)) {
#   install.packages("dplyr")
#   message("Package 'dplyr' werd geïnstalleerd.")
# } else {
#   message("Package 'dplyr' is reeds geïnstalleerd.")
# }
#
# library(dplyr)
```

Om een overzicht van alle geïnstalleerde packages te verkrijgen:

```
# installed.packages()[, "Package"]
```

1.5 Belang van reproduceerbaarheid

Bij het delen van scripts of rapporten is het handig om expliciet te vermelden welke packages gebruikt worden, inclusief versienummers. Dit kan via:

```
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22631)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
## [3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
## [5] LC_TIME=Dutch_Netherlands.utf8
##
## time zone: Europe/Paris
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggplot2_3.5.2
##
## loaded via a namespace (and not attached):
##  [1] vctrs_0.6.5        cli_3.6.1          knitr_1.43         rlang_1.1.1
##  [5] xfun_0.40          generics_0.1.3     glue_1.6.2         htmltools_0.5.8.1
##  [9] scales_1.4.0       fansi_1.0.4        rmarkdown_2.24     grid_4.3.1
## [13] evaluate_0.21      tibble_3.2.1       fastmap_1.1.1      yaml_2.3.7
## [17] lifecycle_1.0.3    compiler_4.3.1     dplyr_1.1.4        RColorBrewer_1.1-3
## [21] pkgconfig_2.0.3    rstudioapi_0.15.0  farver_2.1.1       digest_0.6.33
## [25] R6_2.5.1           tidyselect_1.2.0   utf8_1.2.3         pillar_1.9.0
## [29] magrittr_2.0.3     withr_2.5.0        tools_4.3.1        gtable_0.3.4
```

of

```
devtools::session_info()
```

```
## - Session info ---------------------------------------------------------------
##  setting  value
##  version  R version 4.3.1 (2023-06-16 ucrt)
##  os       Windows 11 x64 (build 22631)
##  system   x86_64, mingw32
##  ui       RTerm
```

```
##  language (EN)
##  collate  Dutch_Netherlands.utf8
##  ctype    Dutch_Netherlands.utf8
##  tz       Europe/Paris
##  date     2025-05-20
##  pandoc   3.2 @ C:/Program Files/RStudio/resources/app/bin/quarto/bin/tools/ (via rmarkdown)
##
## - Packages ------------------------------------------------------------------
##  package      * version date (UTC) lib source
##  cachem         1.0.8   2023-05-01 [1] CRAN (R 4.3.1)
##  callr          3.7.3   2022-11-02 [1] CRAN (R 4.3.1)
##  cli            3.6.1   2023-03-23 [1] CRAN (R 4.3.1)
##  crayon         1.5.2   2022-09-29 [1] CRAN (R 4.3.1)
##  devtools       2.4.5   2022-10-11 [1] CRAN (R 4.3.1)
##  digest         0.6.33  2023-07-07 [1] CRAN (R 4.3.1)
##  dplyr          1.1.4   2023-11-17 [1] CRAN (R 4.3.3)
##  ellipsis       0.3.2   2021-04-29 [1] CRAN (R 4.3.1)
##  evaluate       0.21    2023-05-05 [1] CRAN (R 4.3.1)
##  fansi          1.0.4   2023-01-22 [1] CRAN (R 4.3.1)
##  farver         2.1.1   2022-07-06 [1] CRAN (R 4.3.1)
##  fastmap        1.1.1   2023-02-24 [1] CRAN (R 4.3.1)
##  fs             1.6.3   2023-07-20 [1] CRAN (R 4.3.1)
##  generics       0.1.3   2022-07-05 [1] CRAN (R 4.3.1)
##  ggplot2      * 3.5.2   2025-04-09 [1] CRAN (R 4.3.1)
##  glue           1.6.2   2022-02-24 [1] CRAN (R 4.3.1)
##  gtable         0.3.4   2023-08-21 [1] CRAN (R 4.3.1)
##  htmltools      0.5.8.1 2024-04-04 [1] CRAN (R 4.3.3)
##  htmlwidgets    1.6.2   2023-03-17 [1] CRAN (R 4.3.1)
##  httpuv         1.6.11  2023-05-11 [1] CRAN (R 4.3.1)
##  knitr          1.43    2023-05-25 [1] CRAN (R 4.3.1)
##  later          1.3.1   2023-05-02 [1] CRAN (R 4.3.1)
##  lifecycle      1.0.3   2022-10-07 [1] CRAN (R 4.3.1)
##  magrittr       2.0.3   2022-03-30 [1] CRAN (R 4.3.1)
##  memoise        2.0.1   2021-11-26 [1] CRAN (R 4.3.1)
##  mime           0.12    2021-09-28 [1] CRAN (R 4.3.0)
##  miniUI         0.1.1.1 2018-05-18 [1] CRAN (R 4.3.1)
##  pillar         1.9.0   2023-03-22 [1] CRAN (R 4.3.1)
##  pkgbuild       1.4.2   2023-06-26 [1] CRAN (R 4.3.1)
##  pkgconfig      2.0.3   2019-09-22 [1] CRAN (R 4.3.1)
##  pkgload        1.3.2.1 2023-07-08 [1] CRAN (R 4.3.1)
##  prettyunits    1.1.1   2020-01-24 [1] CRAN (R 4.3.1)
##  processx       3.8.2   2023-06-30 [1] CRAN (R 4.3.1)
##  profvis        0.3.8   2023-05-02 [1] CRAN (R 4.3.1)
##  promises       1.2.1   2023-08-10 [1] CRAN (R 4.3.1)
##  ps             1.7.5   2023-04-18 [1] CRAN (R 4.3.1)
##  purrr          1.0.2   2023-08-10 [1] CRAN (R 4.3.1)
##  R6             2.5.1   2021-08-19 [1] CRAN (R 4.3.1)
##  RColorBrewer   1.1-3   2022-04-03 [1] CRAN (R 4.3.0)
##  Rcpp           1.0.11  2023-07-06 [1] CRAN (R 4.3.1)
##  remotes        2.4.2.1 2023-07-18 [1] CRAN (R 4.3.1)
##  rlang          1.1.1   2023-04-28 [1] CRAN (R 4.3.1)
##  rmarkdown      2.24    2023-08-14 [1] CRAN (R 4.3.1)
##  rstudioapi     0.15.0  2023-07-07 [1] CRAN (R 4.3.1)
##  scales         1.4.0   2025-04-24 [1] CRAN (R 4.3.1)
```

```
##  sessioninfo    1.2.2   2021-12-06 [1] CRAN (R 4.3.1)
##  shiny          1.7.5   2023-08-12 [1] CRAN (R 4.3.1)
##  stringi        1.7.12  2023-01-11 [1] CRAN (R 4.3.0)
##  stringr        1.5.0   2022-12-02 [1] CRAN (R 4.3.1)
##  tibble         3.2.1   2023-03-20 [1] CRAN (R 4.3.1)
##  tidyselect     1.2.0   2022-10-10 [1] CRAN (R 4.3.1)
##  urlchecker     1.0.1   2021-11-30 [1] CRAN (R 4.3.1)
##  usethis        2.2.2   2023-07-06 [1] CRAN (R 4.3.1)
##  utf8           1.2.3   2023-01-31 [1] CRAN (R 4.3.1)
##  vctrs          0.6.5   2023-12-01 [1] CRAN (R 4.3.3)
##  withr          2.5.0   2022-03-03 [1] CRAN (R 4.3.1)
##  xfun           0.40    2023-08-09 [1] CRAN (R 4.3.1)
##  xtable         1.8-4   2019-04-21 [1] CRAN (R 4.3.1)
##  yaml           2.3.7   2023-01-23 [1] CRAN (R 4.3.0)
##
##  [1] C:/Users/rkhalfa/AppData/Local/R/win-library/4.3
##  [2] C:/Program Files/R/R-4.3.1/library
##
## ------------------------------------------------------------------------------
```

---

2. Working directory

Een correcte configuratie van de **working directory** is essentieel voor het betrouwbaar en reproduceerbaar uitvoeren van scripts in R. De working directory bepaalt namelijk het pad van waaruit bestanden ingelezen en weggeschreven worden.

2.1 Wat is de working directory?

De working directory is het bestandspad waar R zoekt naar bestanden of waar het bestanden opslaat als er geen expliciet pad wordt opgegeven:

```
# Toon huidige working directory
getwd()
```

```
## [1] "C:/Users/rkhalfa/OneDrive - UGent/Robin Khalfa/12. OPLEIDINGEN/SNA IN R/SNA met R"
```

Je kunt de working directory tijdelijk instellen met:

```
# Stel de working directory in
setwd("C:/Users/rkhalfa/OneDrive - UGent/Robin Khalfa/12. OPLEIDINGEN/SNA IN R/SNA met R")
getwd()
```

```
## [1] "C:/Users/rkhalfa/OneDrive - UGent/Robin Khalfa/12. OPLEIDINGEN/SNA IN R/SNA met R"
```

> Het gebruik van `setwd()` wordt afgeraden in gedeelde scripts, omdat dit afhankelijk is van het lokale bestandssysteem van de gebruiker.

2.2 Het probleem met `setwd()` in gedeelde projecten

Scripts die `setwd()` gebruiken met absolute paden zijn **niet reproduceerbaar** op andere systemen. Zo zal een pad als:

```
setwd("C:/Users/rkhalfa/OneDrive - UGent/Robin Khalfa/12. OPLEIDINGEN/SNA IN R")
```

niet werken op een andere computer of besturingssysteem. Dit leidt tot fouten bij het inladen of wegschrijven van bestanden.

2.3 RStudio-projecten (.Rproj) als oplossing

Een **RStudio-project** (`.Rproj`) creëert een afzonderlijke werkomgeving waarbij de projectmap automatisch ingesteld wordt als working directory. Dit biedt meerdere voordelen:

- Reproduceerbaarheid: scripts verwijzen relatief naar bestanden binnen de projectstructuur.
- Bestandsorganisatie: analyses, data en scripts worden binnen één consistente structuur bewaard.
- Integratie: RStudio onthoudt de laatste werksessie, open bestanden, enz.

Aanmaken van een project In RStudio:

> File → New Project → New Directory → R Project
> Kies een map en een naam, bijvoorbeeld: `MyFirstSNA`

Dit maakt automatisch een `.Rproj`-bestand aan. Bij het openen van dit bestand wordt de working directory automatisch op de projectmap ingesteld:

```
getwd()
```

```
## [1] "C:/Users/rkhalfa/OneDrive - UGent/Robin Khalfa/12. OPLEIDINGEN/SNA IN R/SNA met R"
```

```
# [1] ".../MyFirstSNA"
```

Vanaf dan kun je werken met **relatieve paden**, zoals:

```
data <- read.csv("Data/1100_affective_w1.csv")
print(data)
```

```
##        X X1101 X1103 X1104 X1105 X1106 X1107 X1108 X1109 X1110 X1111 X1112 X1113
## 1   1101    NA     1     0     0     1     0     0    -1    -1    -1     0     2
## 2   1103     0    NA     0     0     0     1    -2    -1     1     0     1     0
## 3   1104     1     0    NA     0     0     0     0     0     0     0     0     1
## 4   1105     0     0     0    NA     0     2     0     2     2     2     0     0
## 5   1106     1     1     1     0    NA     0     1     0     0     0     0     1
## 6   1107     0     1     0     2     0    NA     1     2     2     2     1     0
## 7   1108     0    -2     1     1     0     1    NA    -1     0     0     2     1
## 8   1109     0     0    -1     2     0     2     0    NA     1     2     0     0
## 9   1110     0     1     0     2     0     2     1     1    NA     2     1     0
## 10  1111    -2     1     0     2     0     2     1     2     2    NA     1     0
## 11  1112     1     1     0     1     1     1     2     0     1     1    NA     1
```

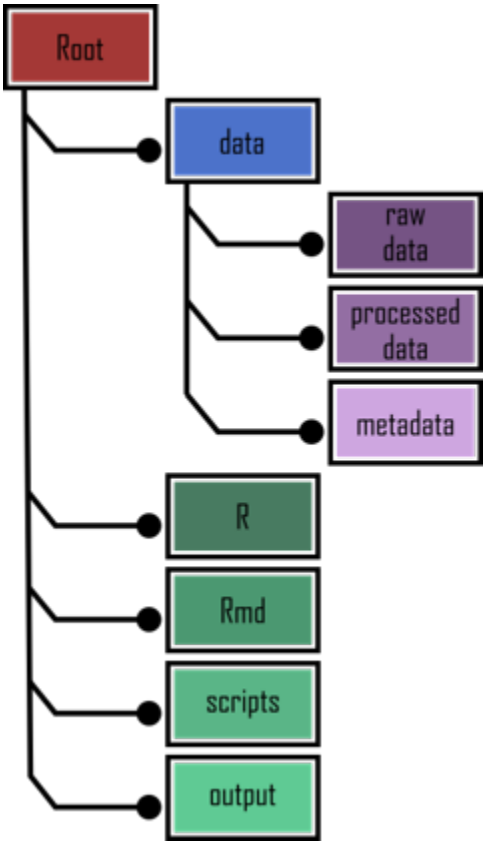
```
## 12 1113     1     1     0     0     2     1     1    -1     0     0    -1    NA
## 13 1114    -2     1     0     1     0     1     0     0     1     2     0     0
## 14 1115     1     1    -1     0     1     1     1     0     1     0    -1     2
## 15 1116     1     0     0     0     2     0     0     0     0     0     0     0
## 16 1117     2     1     2     0     1     0     1     0     0     0     1     2
## 17 1118     2     1     2     0     1     0     1     0     0     0     1     2
## 18 1119     0     0     1     0     0     1     2    -2    -1    -2     2     1
## 19 1120     1     1     0     1     0     1     0     0     2     1     0     0
## 20 1121     0     0     1     1     0     2     1     0     2     1     1     0
## 21 1122    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 22 1123    -1     2     0     1     0     1     0     0     1     1     0     0
## 23 1124     1     1     1     0     2     0     1     0     0     0     1     0
## 24 1125     2     1     0     0     0     0     0    -1     0     0    -1     0
## 25 1127     1     1     0     1     1     1     1    -1     0     0     1     2
## 26 1128    -1     0     0     0     0     0     0    -2     1    -1     2     0
## 27 1129     2     1     1     1     1     0     0     0     0     0     0     0
## 28 1130    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 29 1131     2     1     0     0     1     0     1     0     0    -1     0     1
## 30 1132     1     0     1     0     0     0     0     0     0     0     0     2
## 31 1133     0     1     0     1     0     1     0     0     1     0     0     0
##    X1114 X1115 X1116 X1117 X1118 X1119 X1120 X1121 X1122 X1123 X1124 X1125
## 1      0     1     0     2     2    -1     0     0    -1     0     1     2
## 2      0     0    -1     0     0    -2     2     2    -1     2     1     1
## 3      0    -2     0     2     2     1     0     0     0     0     0    -1
## 4      2     0     0     0     0    -1     2     2     0     2     0     0
## 5      0     1     2     1     1    -1     0     0     1     0     1     0
## 6      2     0     0     0     0     0     2     2     0     2     0     0
## 7      0     0    -1     1     1     2     0     1     0    -1     0     0
## 8      1     0     0     0     0    -1     0     1     0     1     0     0
## 9      1     0    -1     0     0    -1     1     2     0     1     0     1
## 10     2     0    -2     0     0    -2     1     1     0     1     0     0
## 11     0     0     0     0     0     2     1     1     0    -1     1    -1
## 12     0     2     0     2     2     0     0     1     0     0     1     1
## 13    NA     0    -1     0     0    -1     1     1     0     1     0     0
## 14     1    NA     0     1     1     1     0     1     0     0     0     0
## 15     0     1    NA     1     1    -1     0     0     1     0     2     1
## 16     0     2     1    NA     2     1     0     0     2     0     1     2
## 17     0     2     1     2    NA     1     0     0     1     0     1     2
## 18     0     1     1     1     1    NA     0     0    -2     0     0     1
## 19     1     0    -2     0     0     1    NA     2    -1     2     0     1
## 20     1     0    -2     0     0     1     2    NA     2     2     0     1
## 21    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 22     1     0     0     0     0     0     2     2     1    NA     0     0
## 23     0     1     2     2     2     0     0     0     1     0    NA     2
## 24     0     0     0     1     1    -2     0     1     0     0     1    NA
## 25     0     2     0     2     2     0     1     1     0    -1     1     2
## 26     0    -1    -2     2     1    -2     0     0    -1    -1     0    -1
## 27     1     1     1     1     1    -2     0     1     0    -1     1     2
## 28    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 29     0     0     0     1     1    -1     0     0     0     0     1     2
## 30     0     2     1     2     2    -1     0     0     0     0     0     2
## 31     1     0     0     0     0    -2     0     0     0    -1     0     0
##    X1127 X1128 X1129 X1130 X1131 X1132 X1133
## 1      2     1     2     0     2     1     0
```

```
## 2      1     1     0     0     0    -1    -1
## 3      1     0     0     0     0     1     0
## 4      0    -1     0     1     0     0     2
## 5      1     1     1    -1     1     0     0
## 6      0     1     0     1     0     0     2
## 7      1     1     0     0     1     0     0
## 8      0     0     0     0     0     0     0
## 9      0     1     0     2     0     0     0
## 10     0     0     0     0     0     0     0
## 11     1     2     0     0     0     0     0
## 12     2     1     1     0     1     2     0
## 13     0     0     0     0     0     0    -1
## 14     2     0     1     0     0     2     1
## 15     1     1     2     0     1     0     0
## 16     2     2     2     0     2     2     0
## 17     2     1     1     0     0     0     0
## 18     1     0    -2    -2     1    -1     0
## 19     1     1     1     2     1     0    -2
## 20     1     1     1     1     1     0    -2
## 21    NA    NA    NA    NA    NA    NA    NA
## 22     0     0     0     0     0    -1     0
## 23     2     1     2     0     2     1     0
## 24     1     1     2    -1     2     0     0
## 25    NA     1     2    -1     1     2     0
## 26     0    NA    -2    -2     2    -2     0
## 27     1     0    NA     0     2     0     0
## 28    NA    NA    NA    NA    NA    NA    NA
## 29     1     2     2     0    NA     0     0
## 30     2     0     2     0     1    NA     0
## 31     0     0     0     1     0     0    NA
```

in plaats van absolute paden.

2.4 Best practices voor deelbare projecten

Voor projecten die gedeeld worden via e-mail, cloud (bv. OneDrive), of versiebeheer (zoals GitHub), is het aanbevolen om de volgende structuur aan te houden:



Enkele aanbevelingen:

- Gebruik **relatieve paden** binnen scripts.
- Houd ruwe data in een aparte `data/` map.
- Voeg een `README.md` toe met uitleg over de structuur en inhoud.
- Zet geen `setwd()`-commando's in gedeelde scripts.
- Gebruik `.Rproj` als ingangspunt voor de werkomgeving.

---

3. R Basics

In wat volgt gaat we in op enkele fundamentele eigenschappen van R als programmeertaal en rekenomgeving. R is een expressieve en flexibele taal, met een eigen syntax en gedragingen.

3.1 Basiskenmerken van de R-console

- **R is 'case-sensitive'**: `Data`, `data`, en `DATA` zijn verschillende objecten.
- **`#` duidt op een commentaarregel**: alles na een `#` wordt genegeerd door de interpreter.
- Een **`+`-symbool in de console** betekent dat R wacht op verdere input, omdat de opdracht nog niet syntactisch volledig is
- **Spaties zijn belangrijk** in bepaalde gevallen (zoals in bestandsnamen), maar de meeste spaties binnen expressies zijn toegestaan.
- Indien de console vastloopt (bijv. bij een onafgewerkte haak), druk op `Esc` om het proces af te breken en opnieuw te beginnen (of druk op `stop` in de console).

3.2 Simpele wiskundige bewerkingen

R kan rechtstreeks gebruikt worden als een 'rekenmachine':

```
3 + 5       # Optelling
```

```
## [1] 8
```

```
10 - 4      # Aftrekking
```

```
## [1] 6
```

```
6 * 7       # Vermenigvuldiging
```

```
## [1] 42
```

```
8 / 2       # Deling
```

```
## [1] 4
```

```
2^3         # Exponentiële macht
```

```
## [1] 8
```

```
sqrt(25)    # Vierkantswortel
```

```
## [1] 5
```

```
log(100)    # Natuurlijke logaritme
```

```
## [1] 4.60517
```

```
log10(100)  # Logaritme in basis 10
```

```
## [1] 2
```

3.3 Objecten en naamgeving in R

In R worden gegevens opgeslagen in **objecten**. Objecten kunnen variabelen, vectoren, tabellen, functies, enz. bevatten. Objecten worden toegekend met het `<-` (aanbevolen) of `=` operator:

```r
getal <- 42
woord <- "Statistiek" # (kan ook met '')
getal
```

```
## [1] 42
```

```r
woord
```

```
## [1] "Statistiek"
```

R is objectgeoriënteerd: alles wat je aanmaakt of gebruikt is een object.

Naamgeving

- **Snake case** wordt vaak gebruikt: `gemiddelde_waarde`
- Andere stijlen zoals `camelCase` of `dot.notatie` zijn ook mogelijk, maar wees consistent.
- Namen mogen geen spaties bevatten, starten met een letter, en mogen geen gereserveerde woorden omvatten.

3.4 Functies in R

R gebruikt een grote hoeveelheid ingebouwde functies. De syntaxis is: `functie(argumenten)`. Bijvoorbeeld:

```r
c(1, 2, 3, 5)  # Combineert getallen tot een vector
```

```
## [1] 1 2 3 5
```

```r
mean(c(1, 2, 3, 5))  # Berekening van het gemiddelde
```

```
## [1] 2.75
```

Hulpfunctie Voor documentatie over een functie:

```r
?mean
```

```
## starting httpd help server ... done
```

```r
help(mean)
```

Om te zoeken binnen alle helpbestanden:

```r
??"linear model"
```

3.5 Vectoren in R

Vectoren zijn één van de meest belangrijke datastructuren in R. Een vector is een geordende reeks elementen van hetzelfde datatype.

```
x <- c(10, 20, 30)           # Numerieke vector
y <- c("A", "B", "C")        # Karaktervector
z <- c(TRUE, FALSE, TRUE)    # Logische vector
x
```

Aanmaak van vectoren

```
## [1] 10 20 30
```

```
y
```

```
## [1] "A" "B" "C"
```

```
z
```

```
## [1]  TRUE FALSE  TRUE
```

```
x + 1        # Elk element verhoogd met 1
```

Vectoroperaties

```
## [1] 11 21 31
```

```
x * 2        # Vermenigvuldiging per element
```

```
## [1] 20 40 60
```

```
length(x)    # Aantal elementen
```

```
## [1] 3
```

> Let op: vectoren zijn homogeen (alle elementen moeten van hetzelfde type zijn). Gebruik `list()` voor heterogene verzamelingen.

Extractie van elementen Elementen kunnen worden geselecteerd op basis van hun **positie** (indexering) of via **logische expressies**:

```r
x[2]          # Tweede element (20)
```

```
## [1] 20
```

```r
x[c(1,3)]     # Eerste en derde element (10, 30)
```

```
## [1] 10 30
```

```r
x[x > 15]     # Elementen groter dan 15
```

```
## [1] 20 30
```

- R gebruikt **1-gebaseerde indexering** (dus de eerste positie is 1, niet 0 zoals in Python).
- Logische expressies evalueren elk element individueel tot `TRUE` of `FALSE`.

Combinatie van voorwaarden Meerdere voorwaarden kunnen gecombineerd worden met **logische operatoren**:

```r
x[x > 10 & x < 30]    # Tussen 10 en 30 (exclusief)
```

```
## [1] 20
```

```r
x[x == 10 | x == 30] # Gelijk aan 10 of 30
```

```
## [1] 10 30
```

```r
x[!is.na(x)]          # Alle niet-NA elementen
```

```
## [1] 10 20 30
```

- `&`: logische EN
- `|`: logische OF
- `!`: logische NIET

Ordenen van vectoren Vectoren kunnen gesorteerd worden:

```r
sort(x)               # Oplopende volgorde
```

```
## [1] 10 20 30
```

```r
sort(x, decreasing = TRUE)  # Aflopende volgorde
```

```
## [1] 30 20 10
```

Via **positionele ordening** kun je andere vectoren herordenen op basis van een sorteersleutel:

```r
namen <- c("Alice", "Bob", "Chris")
scores <- c(88, 95, 82)

order_index <- order(scores)   # Posities voor oplopende ordening
namen[order_index]             # Namen gesorteerd volgens score
```

```
## [1] "Chris" "Alice" "Bob"
```

Dit is handig om de **relatie tussen kolommen te behouden** in bijvoorbeeld dataframes of gekoppelde vectoren.

Vectorisatie in R De meeste functies in R zijn **vectorized**: ze worden impliciet toegepast op elk element van een vector, zonder dat hiervoor expliciete loops nodig zijn.

```r
x <- c(1, 2, 3)
y <- c(10, 20, 30)
x + y       # [11 22 33]
```

```
## [1] 11 22 33
```

Indien vectoren ongelijke lengte hebben, past R automatisch '**recycling**' toe:

```r
x <- c(1, 2, 3, 4)
y <- c(10, 20)
x + y       # [11 22 13 24]  → y wordt gerecycled
```

```
## [1] 11 22 13 24
```

> Als de lengte van de langere vector **geen veelvoud** is van de kortere, geeft R een waarschuwing.

Omgaan met ontbrekende waarden Ontbrekende waarden worden aangeduid met `NA`:

```r
waarden <- c(1, 2, NA, 4)
mean(waarden)             # Resultaat: NA
```

```
## [1] NA
```

```r
mean(waarden, na.rm = TRUE)  # Resultaat: 2.33
```

```
## [1] 2.333333
```

Veel functies bieden het argument `na.rm = TRUE` om `NA`-waarden te negeren bij berekeningen.

Functionele flexibiliteit R-functies zijn doorgaans **configureerbaar via argumenten**, wat toelaat om gedrag aan te passen aan specifieke behoeften:

```r
round(3.14159, digits = 2)  # Afronden tot 2 decimalen
```

```
## [1] 3.14
```

```r
rep(1:3, times = 2)         # Herhaal vector
```

```
## [1] 1 2 3 1 2 3
```

```r
seq(from = 1, to = 5, by = 0.5)  # Genereer sequentie
```

```
## [1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
```

3.6 Opslaan en laden van objecten

Om objecten op te slaan:

```r
save(x, y, file = "objecten.RData")  # Specifieke objecten
save.image(file = "workspace.RData") # Volledige werksessie
```

Om objecten terug in te laden:

```r
load("objecten.RData")
```

Bestanden worden doorgaans opgeslagen in `.RData` of `.rda`-formaat. Zorg dat je working directory correct ingesteld is (zie vorige sectie).

---

4. Werken met data in R

Werken met data is de belangrijkste activiteit binnen data-analyse in R. Dit deel biedt een overzicht van de belangrijkste concepten en toepassingen.

4.1 Datatypes in R

R ondersteunt zes basis datatypes:

```r
class(3.14)        # numeric
```

```
## [1] "numeric"
```

```r
class(4L)          # integer
```

```
## [1] "integer"
```

```r
class("tekst")      # character
```

```
## [1] "character"
```

```r
class(TRUE)         # logical
```

```
## [1] "logical"
```

```r
class(2+3i)         # complex
```

```
## [1] "complex"
```

```r
class(as.raw(2))    # raw
```

```
## [1] "raw"
```

- **Numeric**: decimale getallen (vb. 3.14)
- **Integer**: gehele getallen (vb. 4L)
- **Character**: tekstuele waarden
- **Logical**: TRUE of FALSE
- **Complex**: getallen met een imaginaire component
- **Raw**: ruwe bytes

4.2 Datastructuren in R

De meest gebruikte datastructuren in R zijn:

Structuur	Beschrijving
Vector	Reeks van elementen van hetzelfde datatype
Matrix	2D-array met homogeen datatype
Array	Multidimensionale uitbreiding van matrices
List	Heterogene container voor objecten
Data frame	Tabelvormige structuur met kolommen van verschillende types

```r
v <- c(1, 2, 3)                          # vector
m <- matrix(1:6, nrow = 2)               # matrix
a <- array(1:8, dim = c(2,2,2))          # array
l <- list(getal = 1, tekst = "abc")      # list
df <- data.frame(id = 1:3, naam = c("A", "B", "C"))  # data frame

#Print out
v
```

```
## [1] 1 2 3
```

```r
m
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

```
a
```

```
## , , 1
##
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
##
## , , 2
##
##      [,1] [,2]
## [1,]    5    7
## [2,]    6    8
```

```
l
```

```
## $getal
## [1] 1
##
## $tekst
## [1] "abc"
```

```
df
```

```
##   id naam
## 1  1    A
## 2  2    B
## 3  3    C
```

```
#saveRDS(v, file = 'Data/v_object.rds')
```

4.3 Importeren van data

R ondersteunt verschillende methoden voor het importeren van data. Veelgebruikte functies zijn:

```
read.csv("Data/1100_affective_w1.csv", header = TRUE, sep = ",")
```

```
##       X X1101 X1103 X1104 X1105 X1106 X1107 X1108 X1109 X1110 X1111 X1112 X1113
## 1  1101    NA     1     0     0     1     0     0    -1    -1    -1     0     2
## 2  1103     0    NA     0     0     0     1    -2    -1     1     0     1     0
## 3  1104     1     0    NA     0     0     0     0     0     0     0     0     1
## 4  1105     0     0     0    NA     0     2     0     2     2     2     0     0
## 5  1106     1     1     1     0    NA     0     1     0     0     0     0     1
## 6  1107     0     1     0     2     0    NA     1     2     2     2     1     0
## 7  1108     0    -2     1     1     0     1    NA    -1     0     0     2     1
## 8  1109     0     0    -1     2     0     2     0    NA     1     2     0     0
## 9  1110     0     1     0     2     0     2     1     1    NA     2     1     0
## 10 1111    -2     1     0     2     0     2     1     2     2    NA     1     0
## 11 1112     1     1     0     1     1     1     2     0     1     1    NA     1
## 12 1113     1     1     0     0     2     1     1    -1     0     0    -1    NA
```

```
## 13 1114    -2     1     0     1     0     1     0     0     1     2     0     0
## 14 1115     1     1    -1     0     1     1     1     0     1     0    -1     2
## 15 1116     1     0     0     0     2     0     0     0     0     0     0     0
## 16 1117     2     1     2     0     1     0     1     0     0     0     1     2
## 17 1118     2     1     2     0     1     0     1     0     0     0     1     2
## 18 1119     0     0     1     0     0     1     2    -2    -1    -2     2     1
## 19 1120     1     1     0     1     0     1     0     0     2     1     0     0
## 20 1121     0     0     1     1     0     2     1     0     2     1     1     0
## 21 1122    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 22 1123    -1     2     0     1     0     1     0     0     1     1     0     0
## 23 1124     1     1     1     0     2     0     1     0     0     0     1     0
## 24 1125     2     1     0     0     0     0     0    -1     0     0    -1     0
## 25 1127     1     1     0     1     1     1     1    -1     0     0     1     2
## 26 1128    -1     0     0     0     0     0     0    -2     1    -1     2     0
## 27 1129     2     1     1     1     1     0     0     0     0     0     0     0
## 28 1130    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 29 1131     2     1     0     0     1     0     1     0     0    -1     0     1
## 30 1132     1     0     1     0     0     0     0     0     0     0     0     2
## 31 1133     0     1     0     1     0     1     0     0     1     0     0     0
##    X1114 X1115 X1116 X1117 X1118 X1119 X1120 X1121 X1122 X1123 X1124 X1125
## 1      0     1     0     2     2    -1     0     0    -1     0     1     2
## 2      0     0    -1     0     0    -2     2     2    -1     2     1     1
## 3      0    -2     0     2     2     1     0     0     0     0     0    -1
## 4      2     0     0     0     0    -1     2     2     0     2     0     0
## 5      0     1     2     1     1    -1     0     0     1     0     1     0
## 6      2     0     0     0     0     0     2     2     0     2     0     0
## 7      0     0    -1     1     1     2     0     1     0    -1     0     0
## 8      1     0     0     0     0    -1     0     1     0     1     0     0
## 9      1     0    -1     0     0    -1     1     2     0     1     0     1
## 10     2     0    -2     0     0    -2     1     1     0     1     0     0
## 11     0     0     0     0     0     2     1     1     0    -1     1    -1
## 12     0     2     0     2     2     0     0     1     0     0     1     1
## 13    NA     0    -1     0     0    -1     1     1     0     1     0     0
## 14     1    NA     0     1     1     1     0     1     0     0     0     0
## 15     0     1    NA     1     1    -1     0     0     1     0     2     1
## 16     0     2     1    NA     2     1     0     0     2     0     1     2
## 17     0     2     1     2    NA     1     0     0     1     0     1     2
## 18     0     1     1     1     1    NA     0     0    -2     0     0     1
## 19     1     0    -2     0     0     1    NA     2    -1     2     0     1
## 20     1     0    -2     0     0     1     2    NA     2     2     0     1
## 21    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 22     1     0     0     0     0     0     2     2     1    NA     0     0
## 23     0     1     2     2     2     0     0     0     1     0    NA     2
## 24     0     0     0     1     1    -2     0     1     0     0     1    NA
## 25     0     2     0     2     2     0     1     1     0    -1     1     2
## 26     0    -1    -2     2     1    -2     0     0    -1    -1     0    -1
## 27     1     1     1     1     1    -2     0     1     0    -1     1     2
## 28    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 29     0     0     0     1     1    -1     0     0     0     0     1     2
## 30     0     2     1     2     2    -1     0     0     0     0     0     2
## 31     1     0     0     0     0    -2     0     0     0    -1     0     0
##    X1127 X1128 X1129 X1130 X1131 X1132 X1133
## 1      2     1     2     0     2     1     0
## 2      1     1     0     0     0    -1    -1
```

```
## 3      1     0     0     0     0     1     0
## 4      0    -1     0     1     0     0     2
## 5      1     1     1    -1     1     0     0
## 6      0     1     0     1     0     0     2
## 7      1     1     0     0     1     0     0
## 8      0     0     0     0     0     0     0
## 9      0     1     0     2     0     0     0
## 10     0     0     0     0     0     0     0
## 11     1     2     0     0     0     0     0
## 12     2     1     1     0     1     2     0
## 13     0     0     0     0     0     0    -1
## 14     2     0     1     0     0     2     1
## 15     1     1     2     0     1     0     0
## 16     2     2     2     0     2     2     0
## 17     2     1     1     0     0     0     0
## 18     1     0    -2    -2     1    -1     0
## 19     1     1     1     2     1     0    -2
## 20     1     1     1     1     1     0    -2
## 21    NA    NA    NA    NA    NA    NA    NA
## 22     0     0     0     0     0    -1     0
## 23     2     1     2     0     2     1     0
## 24     1     1     2    -1     2     0     0
## 25    NA     1     2    -1     1     2     0
## 26     0    NA    -2    -2     2    -2     0
## 27     1     0    NA     0     2     0     0
## 28    NA    NA    NA    NA    NA    NA    NA
## 29     1     2     2     0    NA     0     0
## 30     2     0     2     0     1    NA     0
## 31     0     0     0     1     0     0    NA
```

```
read.table("Data/attributes_clubs.txt", header = TRUE, sep = "\t")
```

```
##                                  ids type missing96 missing97
## 1                  Academic decathalon club         0         0
## 2                            Art Club club         0         0
## 3                          Asian Club club         0         0
## 4                           Band, 8th club         0         0
## 5                          Band, Jazz club         0         0
## 6            Band, Marching (Symphonic) club         0         0
## 7                  Baseball, JV (10th) club         0         0
## 8                          Baseball, V club         0         0
## 9                Basketball, boys 8th club         0         0
## 10               Basketball, boys 9th club         0         0
## 11                Basketball, boys JV club         0         0
## 12                 Basketball, boys V club         0         0
## 13               Basketball, girls 8th club         0         0
## 14               Basketball, girls 9th club         0         0
## 15                Basketball, girls JV club         0         0
## 16                Basketball, girls V club         0         0
## 17                  Cheerleaders, 8th club         0         0
## 18                  Cheerleaders, 9th club         0         0
## 19                   Cheerleaders, JV club         0         0
## 20           Cheerleaders, Spirit Squad club         0         0
## 21                     Cheerleaders, V club         0         0
```

```
## 22                           Chess Club club         0         0
## 23                       Choir, a capella club         0         0
## 24     Choir, barbershop quartet (4 men) club         0         0
## 25                 Choir, chamber singers club         0         0
## 26                         Choir, concert club         0         0
## 27                          Choir, treble club         0         0
## 28       Choir, vocal ensemble (4 women) club         0         0
## 29                 Choir, women's ensemble club         0         0
## 30                              Close-up club         0         0
## 31                Cross Country, boys 8th club         0         0
## 32                  Cross Country, boys V club         0         0
## 33                Cross Country, girls 8th club         0         0
## 34                  Cross Country, girls V club         0         0
## 35                                Debate club         0         0
## 36                             Drill Team club         0         0
## 37                           Drunk Driving club         0         0
## 38                  Drunk Driving Officers club         0         0
## 39                           Football, 8th club         0         0
## 40                           Football, 9th club         0         0
## 41                             Football, V club         0         0
## 42                               Forensics club         0         0
## 43 Forensics (National Forensics League) club         0         0
## 44                       French Club (high) club         0         0
## 45                        French Club (low) club         0         0
## 46                               French NHS club         0         0
## 47       Full IB Diploma Students (12th) club         0         0
## 48                             German Club club         0         0
## 49                              German NHS club         0         0
## 50                            Golf, boys V club         0         0
## 51                           Hispanic Club club         0         0
## 52                             Internships club         0         0
## 53                     Junior Class Board club         0         0
## 54                                Key Club club         0         0
## 55                              Latin Club club         0         0
## 56                          Newspaper Staff club         0         0
## 57                                    NHS club         0         0
## 58                          Orchestra, 8th club         0         0
## 59                 Orchestra, Full Concert club         0         0
## 60                    Orchestra, Symphonic club         0         0
## 61                                   PEER club         0         0
## 62                               Pep Club club         0         0
## 63                         Pep Club Officers club         0         0
## 64                           Quiz-Bowl (all) club         0         0
## 65                         Science Olympiad club         0         0
## 66                                Soccer, V club         0         0
## 67                     Softball, JV (10th) club         0         0
## 68                             Softball, V club         0         0
## 69                            Spanish Club club         0         0
## 70                     Spanish Club (high) club         0         0
## 71                             Spanish NHS club         0         0
## 72                                   STUCO club         0         0
## 73                  Swim & Dive Team, boys club         0         0
## 74                 Swim & Dive Team, girls club         0         0
## 75                    Teachers of Tomorrow club         0         0
```

```
## 76                       Tennis girls V club         0         0
## 77                       Tennis, boys V club         0         0
## 78                  Theatre Productions club         0         0
## 79                Thespian Society (ITS) club         0         0
## 80                      Track, boys 8th club         0         0
## 81                        Track, boys V club         0         0
## 82                      Track, girls 8th club         0         0
## 83                        Track, girls V club         0         0
## 84                       Volleyball, 8th club         0         0
## 85                       Volleyball, 9th club         0         0
## 86                        Volleyball, JV club         0         0
## 87                         Volleyball, V club         0         0
## 88                       Wrestling, 8th club         0         0
## 89                         Wrestling, V club         0         0
## 90                 Yearbook Contributors club         0         0
## 91                      Yearbook Editors club         0         0
##      club_type_detailed club_type_general club_type_gender club_profile
## 1  Academic Competition          Academic       boys_girls          low
## 2       Performance Art   Performance Art       boys_girls          low
## 3       Ethnic Interest   Ethnic Interest       boys_girls          low
## 4       Performance Art   Performance Art       boys_girls          low
## 5       Performance Art   Performance Art       boys_girls     moderate
## 6       Performance Art   Performance Art       boys_girls     moderate
## 7           Team Sports       Team Sports             boys          low
## 8           Team Sports       Team Sports             boys     moderate
## 9           Team Sports       Team Sports             boys          low
## 10          Team Sports       Team Sports             boys          low
## 11          Team Sports       Team Sports             boys     moderate
## 12          Team Sports       Team Sports             boys    very_high
## 13          Team Sports       Team Sports            girls          low
## 14          Team Sports       Team Sports            girls          low
## 15          Team Sports       Team Sports            girls     moderate
## 16          Team Sports       Team Sports            girls         high
## 17          Team Sports       Team Sports            girls         high
## 18          Team Sports       Team Sports            girls         high
## 19          Team Sports       Team Sports            girls    very_high
## 20          Team Sports       Team Sports            girls         high
## 21          Team Sports       Team Sports            girls    very_high
## 22 Academic Competition          Academic       boys_girls          low
## 23      Performance Art   Performance Art       boys_girls     moderate
## 24      Performance Art   Performance Art             boys     moderate
## 25      Performance Art   Performance Art       boys_girls     moderate
## 26      Performance Art   Performance Art       boys_girls          low
## 27      Performance Art   Performance Art            girls          low
## 28      Performance Art   Performance Art            girls     moderate
## 29      Performance Art   Performance Art            girls     moderate
## 30    Academic Interest          Academic       boys_girls          low
## 31    Individual Sports Individual Sports             boys          low
## 32    Individual Sports Individual Sports             boys          low
## 33    Individual Sports Individual Sports            girls          low
## 34    Individual Sports Individual Sports            girls          low
## 35 Academic Competition          Academic       boys_girls     moderate
## 36      Performance Art   Performance Art            girls    very_high
## 37              Service           Service       boys_girls          low
```

```
## 38              Service           Service       boys_girls          low
## 39         Team Sports       Team Sports             boys          low
## 40         Team Sports       Team Sports             boys     moderate
## 41         Team Sports       Team Sports             boys    very_high
## 42 Academic Competition          Academic       boys_girls     moderate
## 43 Academic Competition          Academic       boys_girls     moderate
## 44    Academic Interest          Academic       boys_girls          low
## 45    Academic Interest          Academic       boys_girls          low
## 46    Academic Interest          Academic       boys_girls          low
## 47    Academic Interest          Academic       boys_girls         high
## 48    Academic Interest          Academic       boys_girls          low
## 49    Academic Interest          Academic       boys_girls          low
## 50    Individual Sports Individual Sports             boys          low
## 51      Ethnic Interest   Ethnic Interest       boys_girls          low
## 52    Academic Interest          Academic       boys_girls          low
## 53           Leadership          Academic       boys_girls         high
## 54              Service           Service       boys_girls          low
## 55    Academic Interest          Academic       boys_girls          low
## 56                Media          Academic       boys_girls         high
## 57              Service           Service       boys_girls          low
## 58      Performance Art   Performance Art       boys_girls          low
## 59      Performance Art   Performance Art       boys_girls          low
## 60      Performance Art   Performance Art       boys_girls          low
## 61              Service           Service       boys_girls          low
## 62              Service           Service       boys_girls          low
## 63              Service           Service       boys_girls     moderate
## 64 Academic Competition          Academic       boys_girls          low
## 65 Academic Competition          Academic       boys_girls          low
## 66         Team Sports       Team Sports             boys     moderate
## 67         Team Sports       Team Sports            girls          low
## 68         Team Sports       Team Sports            girls     moderate
## 69    Academic Interest          Academic       boys_girls          low
## 70    Academic Interest          Academic       boys_girls          low
## 71    Academic Interest          Academic       boys_girls          low
## 72           Leadership          Academic       boys_girls    very_high
## 73    Individual Sports Individual Sports             boys          low
## 74    Individual Sports Individual Sports            girls          low
## 75    Academic Interest          Academic       boys_girls          low
## 76    Individual Sports Individual Sports            girls          low
## 77    Individual Sports Individual Sports             boys          low
## 78      Performance Art   Performance Art       boys_girls         high
## 79      Performance Art   Performance Art       boys_girls         high
## 80    Individual Sports Individual Sports             boys          low
## 81    Individual Sports Individual Sports             boys     moderate
## 82    Individual Sports Individual Sports            girls          low
## 83    Individual Sports Individual Sports            girls     moderate
## 84         Team Sports       Team Sports            girls          low
## 85         Team Sports       Team Sports            girls          low
## 86         Team Sports       Team Sports            girls     moderate
## 87         Team Sports       Team Sports            girls         high
## 88    Individual Sports Individual Sports             boys          low
## 89    Individual Sports Individual Sports             boys     moderate
## 90                Media          Academic       boys_girls     moderate
## 91                Media          Academic       boys_girls         high
```

```
##    club_season club_commitment      club_type_grade club_feeder
## 1       spring        not_high           all_grades          no
## 2     all_year        not_high           all_grades          no
## 3     all_year        not_high           all_grades          no
## 4     all_year            high               eighth         yes
## 5     all_year            high               ninth+          no
## 6     all_year            high               ninth+          no
## 7       spring            high ninth_tenth_eleventh         yes
## 8       spring            high               ninth+          no
## 9       winter            high               eighth         yes
## 10      winter            high                ninth         yes
## 11      winter            high ninth_tenth_eleventh         yes
## 12      winter            high               ninth+          no
## 13      winter            high               eighth         yes
## 14      winter            high                ninth         yes
## 15      winter            high ninth_tenth_eleventh         yes
## 16      winter            high               ninth+          no
## 17    all_year            high               eighth         yes
## 18    all_year            high                ninth         yes
## 19    all_year            high ninth_tenth_eleventh         yes
## 20    all_year            high ninth_tenth_eleventh         yes
## 21    all_year            high               ninth+          no
## 22    all_year        not_high           all_grades          no
## 23    all_year            high           all_grades          no
## 24    all_year            high           all_grades          no
## 25    all_year            high           all_grades          no
## 26    all_year            high           all_grades          no
## 27    all_year            high           all_grades          no
## 28    all_year            high           all_grades          no
## 29    all_year            high           all_grades          no
## 30      winter        not_high           all_grades          no
## 31        fall            high               eighth         yes
## 32        fall            high               ninth+          no
## 33        fall            high           all_grades          no
## 34        fall            high               ninth+          no
## 35        fall            high           all_grades          no
## 36    all_year            high           all_grades          no
## 37    all_year        not_high           all_grades          no
## 38    all_year        not_high           all_grades          no
## 39        fall            high               eighth         yes
## 40        fall            high                ninth         yes
## 41        fall            high               ninth+          no
## 42      spring            high           all_grades          no
## 43    all_year            high           all_grades          no
## 44    all_year        not_high           all_grades          no
## 45    all_year        not_high           all_grades         yes
## 46    all_year        not_high           all_grades          no
## 47    all_year            high           all_grades          no
## 48    all_year        not_high           all_grades          no
## 49    all_year        not_high           all_grades          no
## 50      spring            high           all_grades          no
## 51    all_year        not_high           all_grades          no
## 52    all_year            high           all_grades          no
## 53    all_year        not_high           all_grades          no
```

```
## 54    all_year        not_high           all_grades          no
## 55    all_year        not_high           all_grades          no
## 56    all_year            high           all_grades          no
## 57    all_year        not_high           all_grades          no
## 58    all_year            high               eighth         yes
## 59    all_year            high           all_grades          no
## 60    all_year            high               ninth+          no
## 61    all_year        not_high           all_grades          no
## 62    all_year        not_high           all_grades          no
## 63    all_year        not_high           all_grades          no
## 64      winter        not_high           all_grades          no
## 65        fall        not_high           all_grades          no
## 66        fall            high           all_grades          no
## 67      spring            high ninth_tenth_eleventh         yes
## 68      spring            high               ninth+          no
## 69    all_year        not_high           all_grades          no
## 70    all_year        not_high           all_grades          no
## 71    all_year        not_high           all_grades          no
## 72    all_year        not_high           all_grades          no
## 73      winter            high           all_grades          no
## 74      winter            high           all_grades          no
## 75    all_year        not_high           all_grades          no
## 76      spring            high           all_grades          no
## 77      spring            high           all_grades          no
## 78    all_year            high           all_grades          no
## 79    all_year            high           all_grades          no
## 80      spring            high               eighth         yes
## 81      spring            high               ninth+          no
## 82      spring            high           all_grades          no
## 83      spring            high               ninth+          no
## 84    all_year            high               eighth         yes
## 85        fall            high                ninth         yes
## 86        fall            high ninth_tenth_eleventh         yes
## 87        fall            high               ninth+          no
## 88      winter            high               eighth         yes
## 89      winter            high               ninth+          no
## 90    all_year        not_high           all_grades          no
## 91    all_year            high           all_grades          no
```

```
readRDS("Data/v_object.rds")
```

```
## [1] 1 2 3
```

```
load("Data/talk_nets_undirected.RData")
```

Bestandstype	Functie	Belangrijke argumenten
CSV	`read.csv()`	`sep`, `header`, `stringsAsFactors`
TXT	`read.table()`	`sep`, `na.strings`, `colClasses`
Excel (via `readxl`)	`read_excel()`	`sheet`, `col_types`, `range`
RDS	`readRDS()`	-
RData	`load()`	-

Gebruik `readr::read_csv()` voor snellere import via tidyverse.

4.4 Data Wrangling

Data wrangling verwijst naar het proces van het transformeren, structureren en manipuleren van ruwe data tot een geschikt formaat voor analyse.

```
#install.packages('tidyverse')
#install.packages('dplyr')
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.2
```

```
## Warning: package 'tidyr' was built under R version 4.3.3
```

```
## Warning: package 'readr' was built under R version 4.3.3
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
```

4.4.1 Selectie van gegevens: indexering 1. Positionele indexering (via rijen en kolommen):

```
df[1, 2]        # Rij 1, kolom 2
```

```
## [1] "A"
```

```
df[ , "naam"]   # Kolom 'naam'
```

```
## [1] "A" "B" "C"
```

```
df[1:2, ]       # Eerste twee rijen
```

```
##   id naam
## 1  1    A
## 2  2    B
```

2. Logische indexering (gebaseerd op voorwaarden):

```r
df[df$id > 1, ]           # Rijen met id > 1
```

```
##   id naam
## 2  2    B
## 3  3    C
```

Alternatief:

```r
df %>% filter(id > 1)
```

```
##   id naam
## 1  2    B
## 2  3    C
```

4.4.2 Manipuleren van rijen en kolommen Toevoegen of verwijderen van kolommen:

```r
df$score <- c(70, 65, 35)  # Nieuwe kolom
df
```

```
##   id naam score
## 1  1    A    70
## 2  2    B    65
## 3  3    C    35
```

```r
#df$score <- NULL                    # Verwijderen van kolom
```

Met `dplyr`:

```r
df <- df %>%
  mutate(new_var = c(70, 65, 35))
df
```

```
##   id naam score new_var
## 1  1    A    70      70
## 2  2    B    65      65
## 3  3    C    35      35
```

```r
df <- df %>%
  mutate(new_var = c(70, 65, 35)) %>% # Opnieuw verwijderen nieuwe variabele
  select(-new_var)
df
```

```
##   id naam score
## 1  1    A    70
## 2  2    B    65
## 3  3    C    35
```

Toevoegen of verwijderen van rijen:

```r
nieuwe_rij <- data.frame(id = 4, naam = "Robin", score = 89)
df <- rbind(df, nieuwe_rij)        # Rij toevoegen
df
```

```
##   id  naam score
## 1  1     A    70
## 2  2     B    65
## 3  3     C    35
## 4  4 Robin    89
```

```r
#df <- df[-1, ]                    # Verwijder eerste rij
```

Tidyverse:

```r
df <- bind_rows(df, nieuwe_rij) %>% slice(-5)
df
```

```
##   id  naam score
## 1  1     A    70
## 2  2     B    65
## 3  3     C    35
## 4  4 Robin    89
```

4.4.3 Ordenen van data Sorteren van rijen:

```r
df[order(df$score), ] #Oplopend
```

```
##   id  naam score
## 3  3     C    35
## 2  2     B    65
## 1  1     A    70
## 4  4 Robin    89
```

```r
df[order(-df$score), ]  # Aflopend
```

```
##   id  naam score
## 4  4 Robin    89
## 1  1     A    70
## 2  2     B    65
## 3  3     C    35
```

Met `dplyr::arrange()`:

```r
df %>% arrange(desc(score))
```

```
##   id  naam score
## 1  4 Robin    89
## 2  1     A    70
## 3  2     B    65
## 4  3     C    35
```

Herordenen van kolommen:

```
df <- df[ , c("id", "score", "naam")]
df
```

```
##   id score  naam
## 1  1    70     A
## 2  2    65     B
## 3  3    35     C
## 4  4    89 Robin
```

Of met select:

```
df %>% select(id, naam, score)
```

```
##   id  naam score
## 1  1     A    70
## 2  2     B    65
## 3  3     C    35
## 4  4 Robin    89
```

```
df2 <- data.frame(
  id = 5:20,  # 13 waarden
  naam = c("Eva", "Tom", "Lena", "Max", "Sara", "Joris", "Anna", "Paul",
           "Lily", "Hans", "Marie", "Niels", "Laura", "Wim", "Jonas", "Rein"),
  score = sample(50:100, 16, replace = TRUE)
)
df2
```

4.4.4 Samenvoegen van datasets

```
##    id  naam score
## 1   5   Eva    78
## 2   6   Tom    85
## 3   7  Lena    63
## 4   8   Max    86
## 5   9  Sara    71
## 6  10 Joris    64
## 7  11  Anna    85
## 8  12  Paul    66
## 9  13  Lily    51
## 10 14  Hans    95
## 11 15 Marie    75
## 12 16 Niels    97
## 13 17 Laura    77
## 14 18   Wim    76
## 15 19 Jonas    67
## 16 20  Rein    94
```

Verticaal (rijen toevoegen):

```r
df_combined <- rbind(df, df2)
df_combined
```

```
##    id score  naam
## 1   1    70     A
## 2   2    65     B
## 3   3    35     C
## 4   4    89 Robin
## 5   5    78   Eva
## 6   6    85   Tom
## 7   7    63  Lena
## 8   8    86   Max
## 9   9    71  Sara
## 10 10    64 Joris
## 11 11    85  Anna
## 12 12    66  Paul
## 13 13    51  Lily
## 14 14    95  Hans
## 15 15    75 Marie
## 16 16    97 Niels
## 17 17    77 Laura
## 18 18    76   Wim
## 19 19    67 Jonas
## 20 20    94  Rein
```

Horizontaal (kolommen toevoegen):

```r
geslaagd <- df_combined$score >= 50
df_combined <- cbind(df_combined, geslaagd)
df_combined
```

```
##    id score  naam geslaagd
## 1   1    70     A     TRUE
## 2   2    65     B     TRUE
## 3   3    35     C    FALSE
## 4   4    89 Robin     TRUE
## 5   5    78   Eva     TRUE
## 6   6    85   Tom     TRUE
## 7   7    63  Lena     TRUE
## 8   8    86   Max     TRUE
## 9   9    71  Sara     TRUE
## 10 10    64 Joris     TRUE
## 11 11    85  Anna     TRUE
## 12 12    66  Paul     TRUE
## 13 13    51  Lily     TRUE
## 14 14    95  Hans     TRUE
## 15 15    75 Marie     TRUE
## 16 16    97 Niels     TRUE
## 17 17    77 Laura     TRUE
## 18 18    76   Wim     TRUE
## 19 19    67 Jonas     TRUE
## 20 20    94  Rein     TRUE
```

Kan natuurlijk veel gemakkelijker:

```
df_combined <- df_combined %>% select(-geslaagd)
df_combined$geslaagd <- df_combined$score >= 50
df_combined
```

```
##    id score  naam geslaagd
## 1   1    70     A     TRUE
## 2   2    65     B     TRUE
## 3   3    35     C    FALSE
## 4   4    89 Robin     TRUE
## 5   5    78   Eva     TRUE
## 6   6    85   Tom     TRUE
## 7   7    63  Lena     TRUE
## 8   8    86   Max     TRUE
## 9   9    71  Sara     TRUE
## 10 10    64 Joris     TRUE
## 11 11    85  Anna     TRUE
## 12 12    66  Paul     TRUE
## 13 13    51  Lily     TRUE
## 14 14    95  Hans     TRUE
## 15 15    75 Marie     TRUE
## 16 16    97 Niels     TRUE
## 17 17    77 Laura     TRUE
## 18 18    76   Wim     TRUE
## 19 19    67 Jonas     TRUE
## 20 20    94  Rein     TRUE
```

Join-operaties (zoals in SQL):

```
df1 <- data.frame(
  id = 1:5,
  naam = c("Eva", "Tom", "Lena", "Max", "Sara")
)

df3 <- data.frame(
  id = c(3, 4, 5, 6, 7),
  score = c(78, 88, 91, 65, 72)
)

# Inner join: enkel rijen met overeenkomende 'id'
df_merged_inner <- merge(df1, df3, by = "id", all = FALSE)

# Left join: behoud alle rijen uit df1
df_merged_left <- merge(df1, df3, by = "id", all.x = TRUE)

# Full join: behoud alle rijen uit beide
df_merged_full <- merge(df1, df3, by = "id", all = TRUE)

# Print out
df_merged_inner
```

```
##   id naam score
```

```
## 1  3 Lena    78
## 2  4  Max    88
## 3  5 Sara    91
```

```
df_merged_left
```

```
##   id naam score
## 1  1  Eva    NA
## 2  2  Tom    NA
## 3  3 Lena    78
## 4  4  Max    88
## 5  5 Sara    91
```

```
df_merged_full
```

```
##   id naam score
## 1  1  Eva    NA
## 2  2  Tom    NA
## 3  3 Lena    78
## 4  4  Max    88
## 5  5 Sara    91
## 6  6 <NA>    65
## 7  7 <NA>    72
```

Tidyverse:

```
# Inner join
df_merged_inner_tidy <- df1 %>%
  inner_join(df3, by = "id")

# Left join
df_merged_left_tidy <- df1 %>%
  left_join(df3, by = "id")

# Full join
df_merged_full_tidy <- df1 %>%
  full_join(df3, by = "id")

# Anti join: toon rijen in df1 die géén match hebben in df2
df_anti_tidy <- df1 %>%
  anti_join(df3, by = "id")

# Print out
df_merged_inner_tidy
```

```
##   id naam score
## 1  3 Lena    78
## 2  4  Max    88
## 3  5 Sara    91
```

```
df_merged_left_tidy
```

```
##   id naam score
## 1  1  Eva    NA
## 2  2  Tom    NA
## 3  3 Lena    78
## 4  4  Max    88
## 5  5 Sara    91
```

```
df_merged_full_tidy
```

```
##   id naam score
## 1  1  Eva    NA
## 2  2  Tom    NA
## 3  3 Lena    78
## 4  4  Max    88
## 5  5 Sara    91
## 6  6 <NA>    65
## 7  7 <NA>    72
```

```
df_anti_tidy
```

```
##   id naam
## 1  1  Eva
## 2  2  Tom
```

```
df4 <- data.frame(
  id = 1:3,
  naam = c("Eva", "Tom", "Lena"),
  jaar_2022 = c(75, 88, 93),
  jaar_2023 = c(82, 90, 95)
)

df4
```

4.4.5 Hervormen van data

```
##   id naam jaar_2022 jaar_2023
## 1  1  Eva        75        82
## 2  2  Tom        88        90
## 3  3 Lena        93        95
```

Van breed naar lang formaat (pivoting):

```
df_long <- df4 %>% pivot_longer(cols = starts_with("jaar"), names_to = "jaar",
                                values_to = "waarde")

df_long
```

```
## # A tibble: 6 x 4
##      id naam  jaar      waarde
##   <int> <chr> <chr>      <dbl>
## 1     1 Eva   jaar_2022     75
## 2     1 Eva   jaar_2023     82
## 3     2 Tom   jaar_2022     88
## 4     2 Tom   jaar_2023     90
## 5     3 Lena  jaar_2022     93
## 6     3 Lena  jaar_2023     95
```

Van lang naar breed formaat:

```
df_breed <- df_long %>% pivot_wider(names_from = "jaar", values_from = "waarde")

df_breed
```

```
## # A tibble: 3 x 4
##      id naam  jaar_2022 jaar_2023
##   <int> <chr>     <dbl>     <dbl>
## 1     1 Eva          75        82
## 2     2 Tom          88        90
## 3     3 Lena         93        95
```

Splitsen en samenvoegen van kolommen:

```
df_datum <- data.frame(
  datum = c("2023-05-12", "2024-03-08", "2024-11-20"),
  voornaam = c("Eva", "Tom", "Lena"),
  achternaam = c("Peeters", "Janssens", "De Smet")
)

df_datum
```

```
##        datum voornaam achternaam
## 1 2023-05-12      Eva    Peeters
## 2 2024-03-08      Tom   Janssens
## 3 2024-11-20     Lena    De Smet
```

```
df_gesplitst <- df_datum %>%
  separate(col = datum, into = c("jaar", "maand", "dag"), sep = "-")

df_samengevoegd <- df_gesplitst %>%
  unite("volledige_naam", voornaam, achternaam, sep = " ")

df_gesplitst
```

```
##   jaar maand dag voornaam achternaam
## 1 2023    05  12      Eva    Peeters
## 2 2024    03  08      Tom   Janssens
## 3 2024    11  20     Lena    De Smet
```

```r
df_samengevoegd
```

```
##   jaar maand dag volledige_naam
## 1 2023    05  12    Eva Peeters
## 2 2024    03  08   Tom Janssens
## 3 2024    11  20   Lena De Smet
```

4.4.6 Samenvatten en groeperen Gebruik van `dplyr` voor samenvattende statistieken:

```r
df_scores <- data.frame(
  id = 1:6,
  categorie = c("A", "A", "B", "B", "C", "C"),
  score = c(75, 82, 91, 85, 44, 63)
)

df_scores
```

```
##   id categorie score
## 1  1         A    75
## 2  2         A    82
## 3  3         B    91
## 4  4         B    85
## 5  5         C    44
## 6  6         C    63
```

```r
df_scores %>%
  group_by(categorie) %>%
  summarise(
    gemiddelde = mean(score, na.rm = TRUE),
    aantal = n()
  )
```

```
## # A tibble: 3 x 3
##   categorie gemiddelde aantal
##   <chr>          <dbl>  <int>
## 1 A               78.5      2
## 2 B               88        2
## 3 C               53.5      2
```

Meerdere samenvattingen combineren:

```r
df_scores %>%
  group_by(categorie) %>%
  summarise(across(
    where(is.numeric),
    list(mean = mean, sd = sd),
    na.rm = TRUE
  ))
```

```
## Warning: There was 1 warning in `summarise()`.
```

```
## i In argument: ‘across(where(is.numeric), list(mean = mean, sd = sd), na.rm =
##   TRUE)‘.
## i In group 1: ‘categorie = "A"‘.
## Caused by warning:
## ! The ‘...‘ argument of ‘across()‘ is deprecated as of dplyr 1.1.0.
## Supply arguments directly to ‘.fns‘ through an anonymous function instead.
##
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))
```

```
## # A tibble: 3 x 5
##   categorie id_mean id_sd score_mean score_sd
##   <chr>       <dbl> <dbl>      <dbl>    <dbl>
## 1 A             1.5 0.707       78.5     4.95
## 2 B             3.5 0.707       88       4.24
## 3 C             5.5 0.707       53.5    13.4
```

---

5. Graphics en visualisaties

R is heel krachtig wat betreft het visualiseren van data. Grafieken bieden een intuïtieve manier om patronen, trends en anomalieën te identificeren in datasets. In R bestaan er drie hoofdbenaderingen voor het aanmaken van grafieken:

- **Base R graphics**: het originele, flexibele maar minder gestructureerde systeem
- **Lattice graphics**: gericht op multi-panel visualisaties (behandelen we niet hier)
- **ggplot2**: gebaseerd op het principe ‘Grammar of Graphics’

Elke methode heeft haar eigen filosofie, sterktes en typische toepassingsgebieden.

5.1 Base R Graphics

Het base graphics-systeem in R biedt eenvoudige maar flexibele grafische functies. Het systeem is opgebouwd rond twee groepen functies:

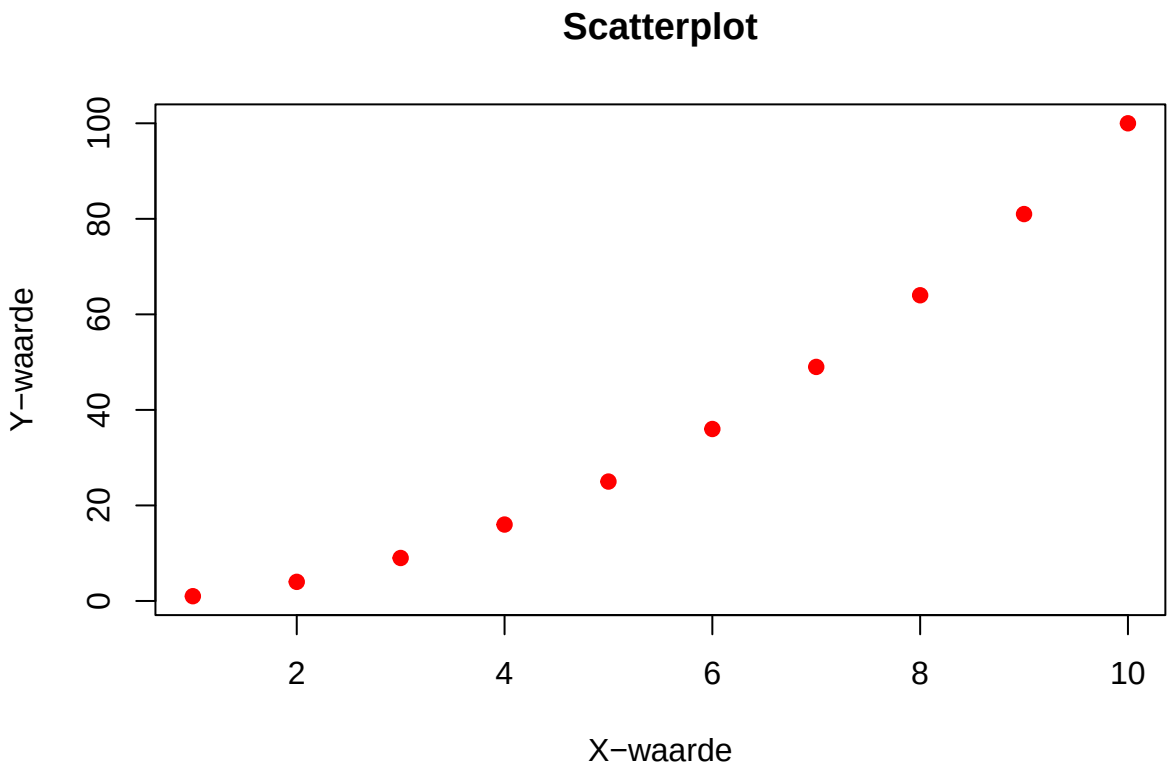
- **High-level functies**: creëren een nieuwe grafiek (zoals `plot()`, `hist()`, `boxplot()`).
- **Low-level functies**: voegen elementen toe aan een bestaande grafiek (zoals `lines()`, `points()`, `text()`, `legend()`).

Base R is veelzijdig, maar vereist manuele controle en is minder consistent in syntax tussen verschillende plottypes.

- Basis plot (bv. scatterplot):

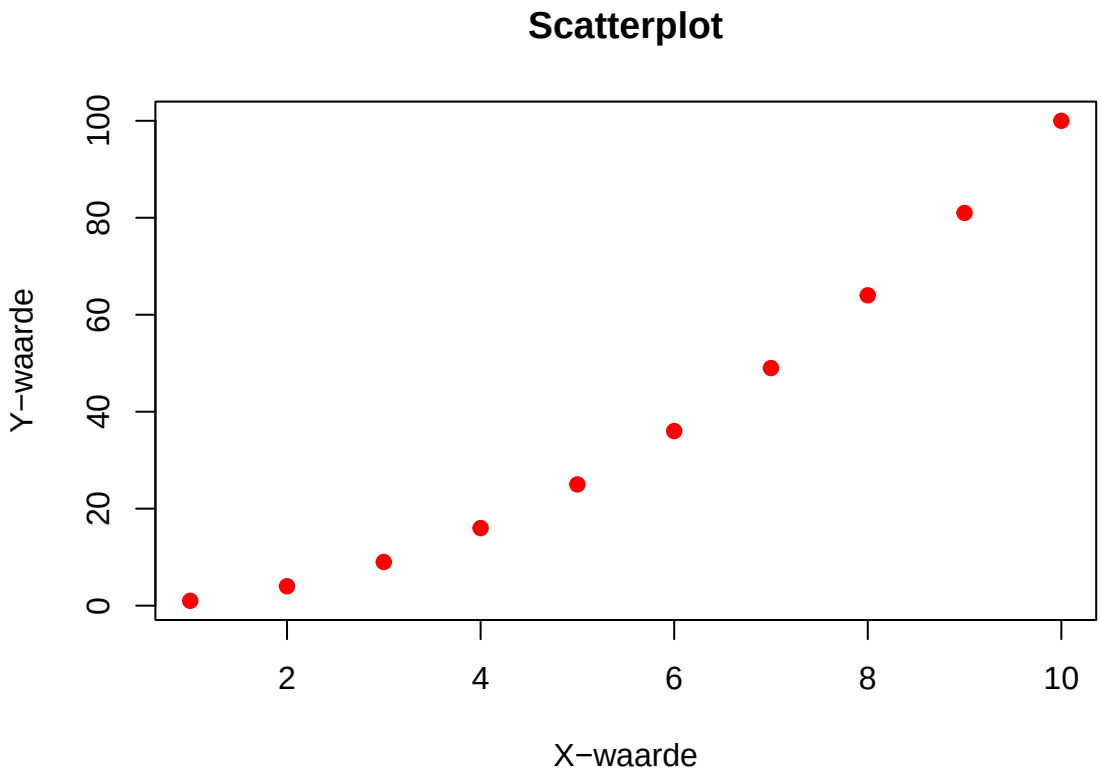
```r
# Basis scatterplot
x <- 1:10
y <- x^2

plot(x, y, main = "Scatterplot", xlab = "X-waarde", ylab = "Y-waarde", pch = 19,
     col = "red")
```



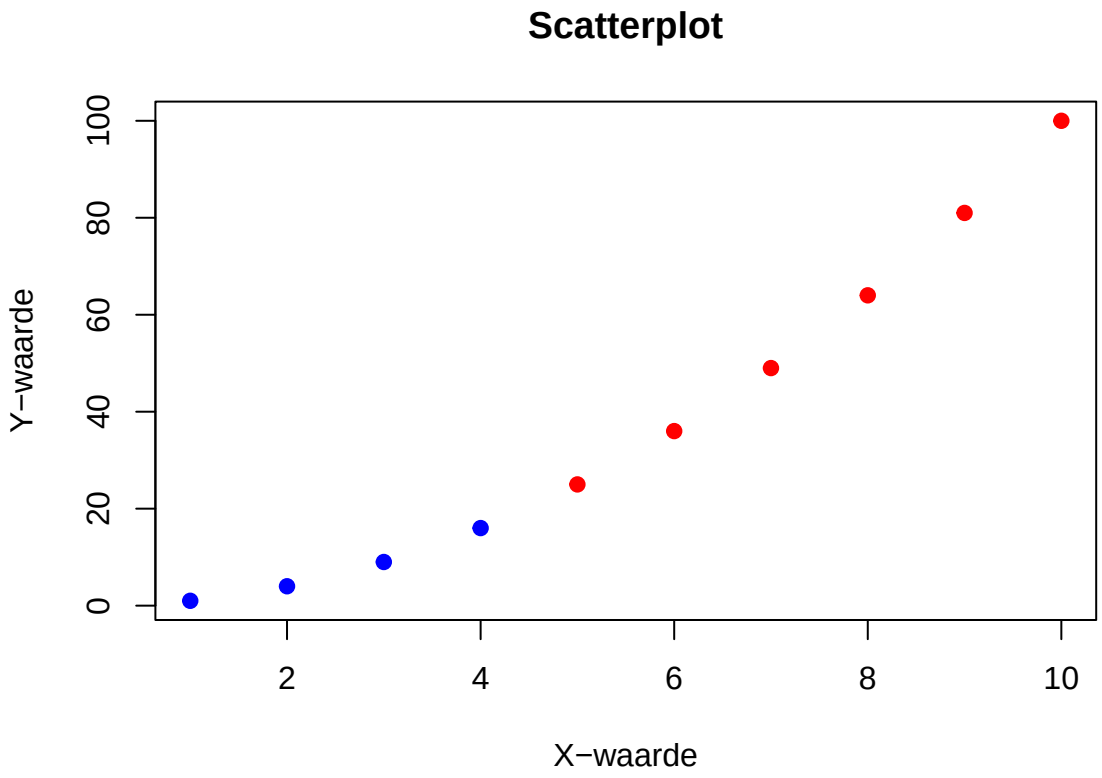
- Aanpassen van de marges:

```r
par(mar = c(5, 6, 4, 2))# Instellingen marges: c(boven, links, onder, rechts)
plot(x, y, main = "Scatterplot", xlab = "X-waarde", ylab = "Y-waarde", pch = 19,
     col='red')
```



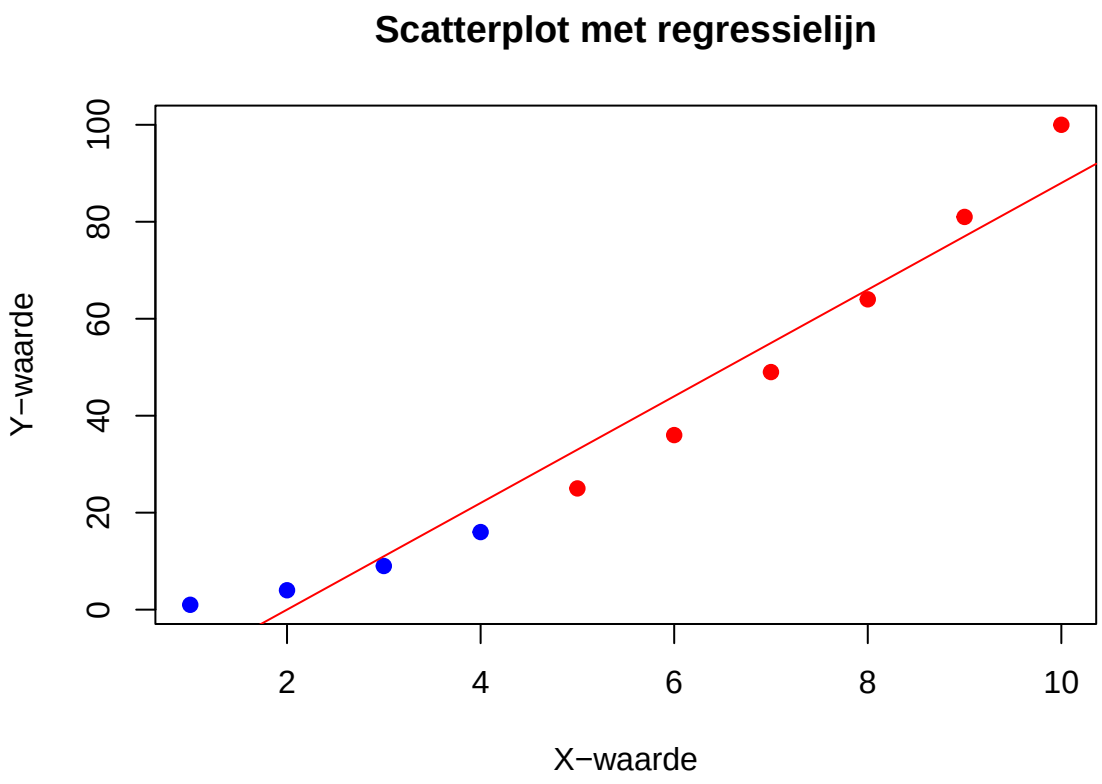
- Kleuren op basis van numerieke variabele:

```
kleuren <- ifelse(x < 5, "blue", "red")
par(mar = c(5, 6, 4, 2))
plot(x, y, main = "Scatterplot", xlab = "X-waarde", ylab = "Y-waarde", pch = 19,
     col = kleuren)
```



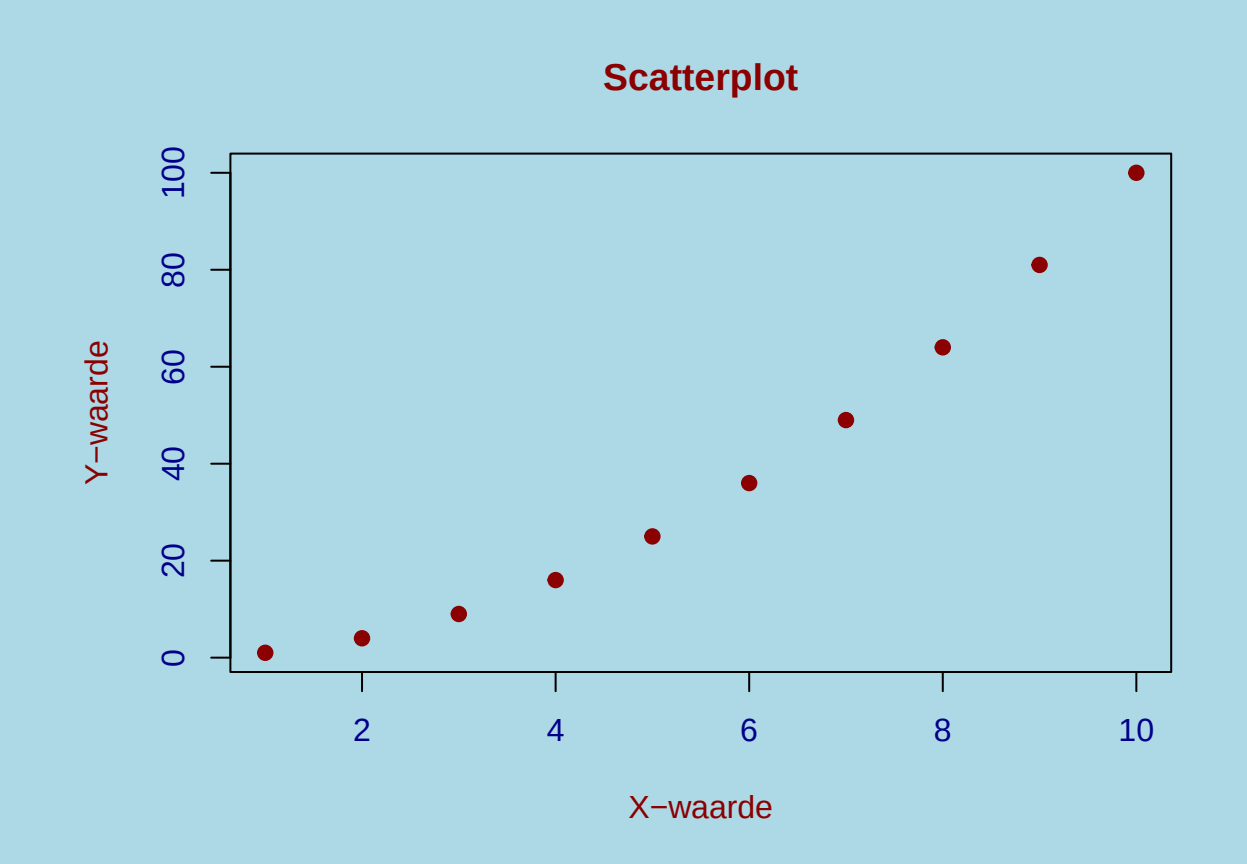
- Lijnen en achtergrond

```
par(mar = c(5, 6, 4, 2))
plot(x, y, main = "Scatterplot met regressielijn", xlab = "X-waarde",
     ylab = "Y-waarde", pch = 19, col = kleuren)
abline(lm(y ~ x), col = "red")  # Regressielijn in rood
```



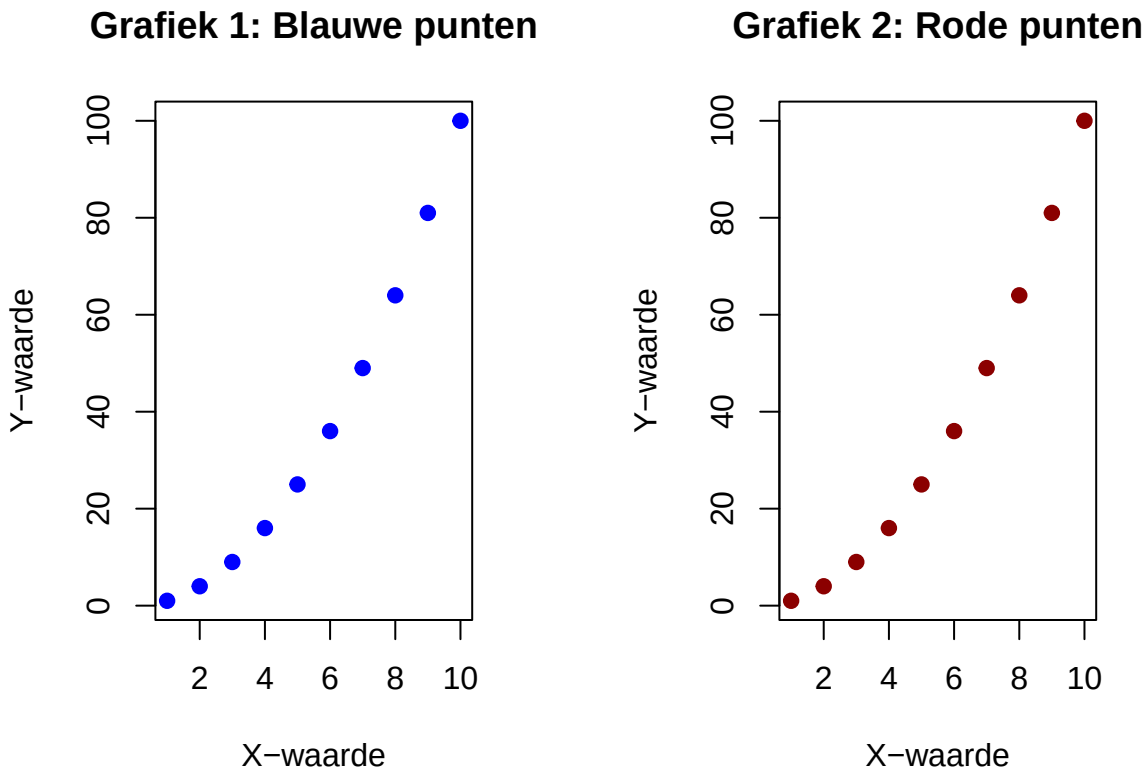
- Instellen van achtergrond en askleuren

```
par(mar = c(5, 6, 4, 2))
par(bg = "lightblue", col.axis = "darkblue", col.lab = "darkred")
plot(x, y, main = "Scatterplot", xlab = "X-waarde", ylab = "Y-waarde", pch = 19,
     col = 'darkred', col.main = 'darkred')
```



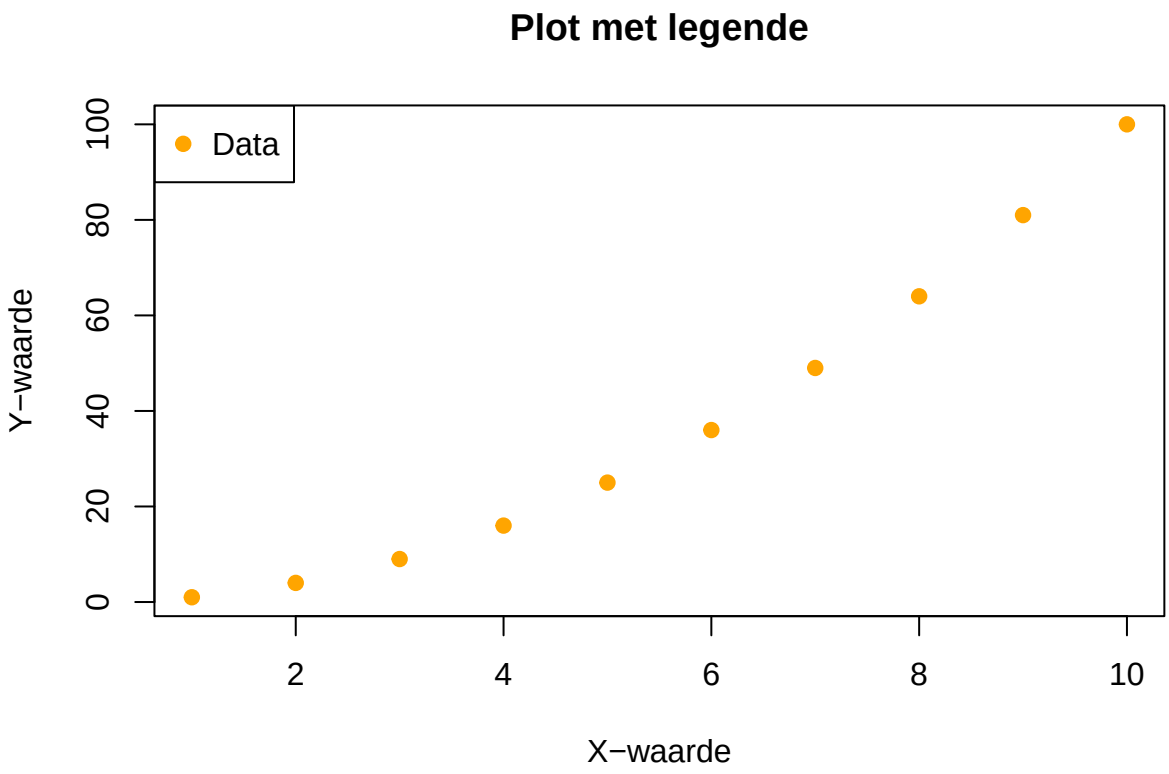
- Meerdere grafieken tegelijk plotten

```
par(mfrow = c(1, 2))  # Twee grafieken naast elkaar
par(mar = c(5, 6, 4, 2))
plot(x, y, main = "Grafiek 1: Blauwe punten", xlab = "X-waarde",
     ylab = "Y-waarde", pch = 19, col = "blue")
plot(x, y, main = "Grafiek 2: Rode punten", xlab = "X-waarde",
     ylab = "Y-waarde", pch = 19, col = "darkred")
```



- Legendes

```
# Plot met legende
plot(x, y, main = "Plot met legende", xlab = "X-waarde", ylab = "Y-waarde",
     pch = 19, col = "orange")
legend("topleft", legend = "Data", col = "orange", pch = 19)
```



- Exporteren van base R visualisatie (voorbeeld)

```
# Exporteren van grafiek naar PNG met kleuren
png("scatterplot.png", width = 800, height = 600, res = 150)
plot(x, y, main = "Scatter", xlab = "X-waarde", ylab = "Y-waarde",
     pch = 19, col = "darkgreen")
dev.off()
```

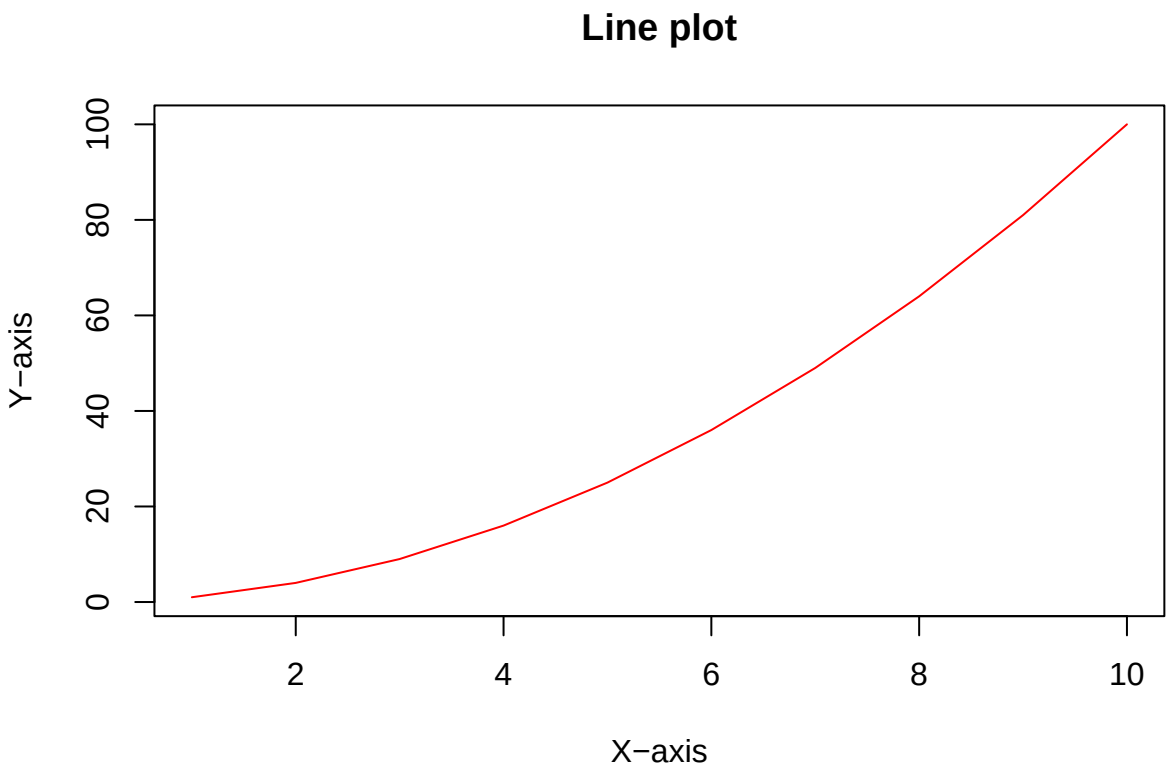
```
## pdf
##   2
```

- Enkele veel gebruikte plots en diens functie in R

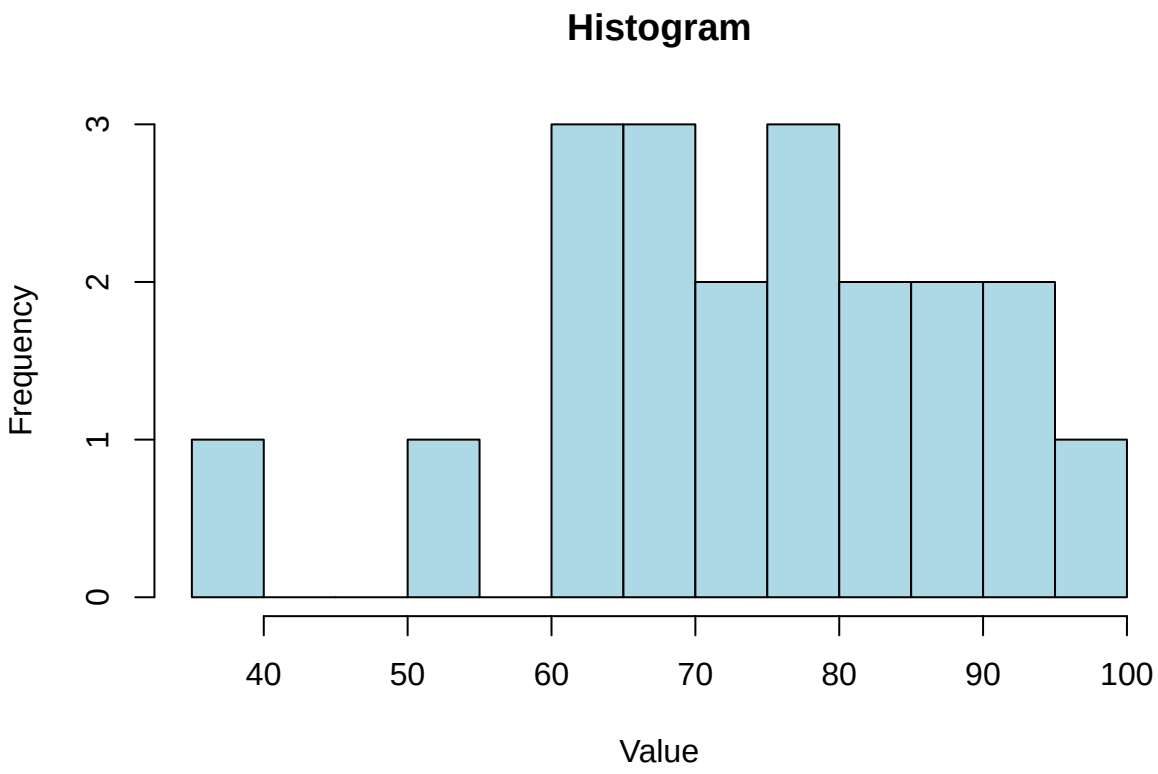
Plot type	Functie	Beschrijving	Voorbeeld code
Scatter plot	`plot()`	Basisfunctie voor het tekenen van een scatter plot tussen twee variabelen.	`plot(x, y)`
Lijn plot	`plot()`	Gebruikt voor het tekenen van een lijnplot, vooral als de argumenten `type = "l"` zijn.	`plot(x, y, type = "l")`

Plot type	Functie	Beschrijving	Voorbeeld code
Histogram	`hist()`	Teken een histogram voor een enkele variabele.	`hist(data)`
Boxplot	`boxplot()`	Teken een boxplot om de spreiding van de data en de aanwezigheid van outliers te visualiseren.	`boxplot(data)`
Barplot	`barplot()`	Teken een barplot voor het visualiseren van de frequenties van categorieën.	`barplot(table(data))`
Pie chart	`pie()`	Teken een taartdiagram om proporties van een totaal te visualiseren.	`pie(c(10, 20, 30, 40))`
Density plot	`plot(density())`	Plot de dichtheidsfunctie van een continue variabele.	`plot(density(data))`
Lijn plot met meerdere lijnen	`matplot()`	Teken meerdere lijnen in één grafiek.	`matplot(x, y, type = "l")`
Stem-and-leaf plot	`stem()`	Visualiseer de verdeling van een numerieke variabele als een stem-and-leaf plot.	`stem(data)`
Scatterplot matrix	`pairs()`	Maak een matrix van scatter plots om de relaties tussen meerdere variabelen te bekijken.	`pairs(~ var1 + var2 + var3, data = df)`
Heatmap	`heatmap()`	Maak een heatmap van een matrix of dataset om patronen te visualiseren.	`heatmap(matrix_data)`
Correlogram	`cor()` + `image()`	Visualiseer de correlatie van variabelen in een matrixvorm.	`image(cor(data))`
QQ-Plot	`qqnorm()` / `qqline()`	Visualiseer de normaalverdeling van data met een QQ-plot.	`qqnorm(data); qqline(data)`

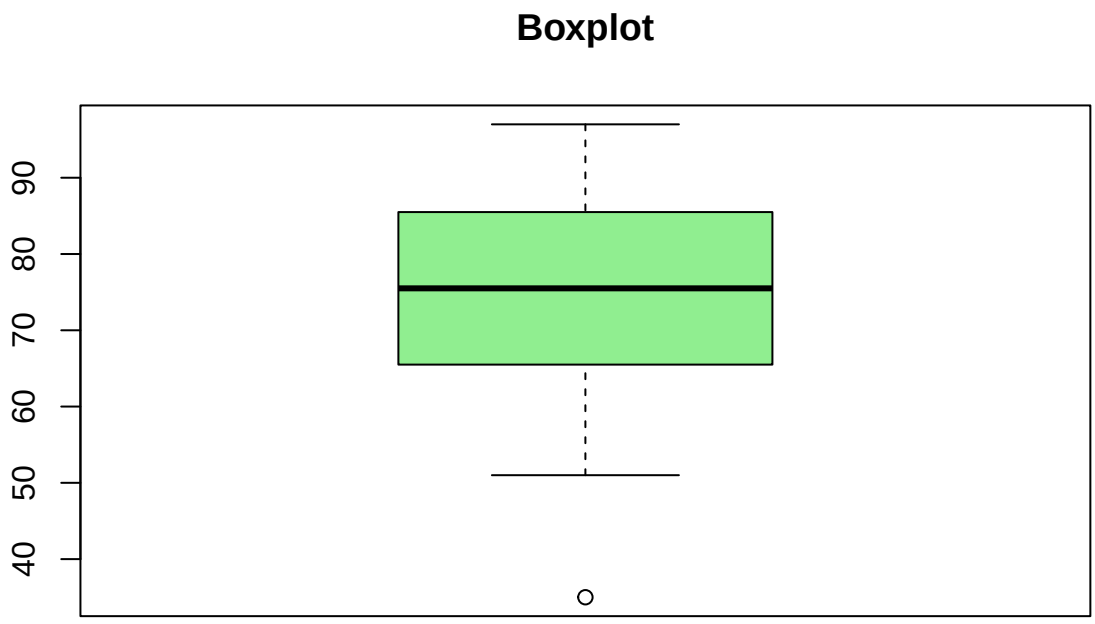
```r
# Line Plot
plot(x, y, type = "l", col = "red", main = "Line plot", xlab = "X-axis",
     ylab = "Y-axis")
```



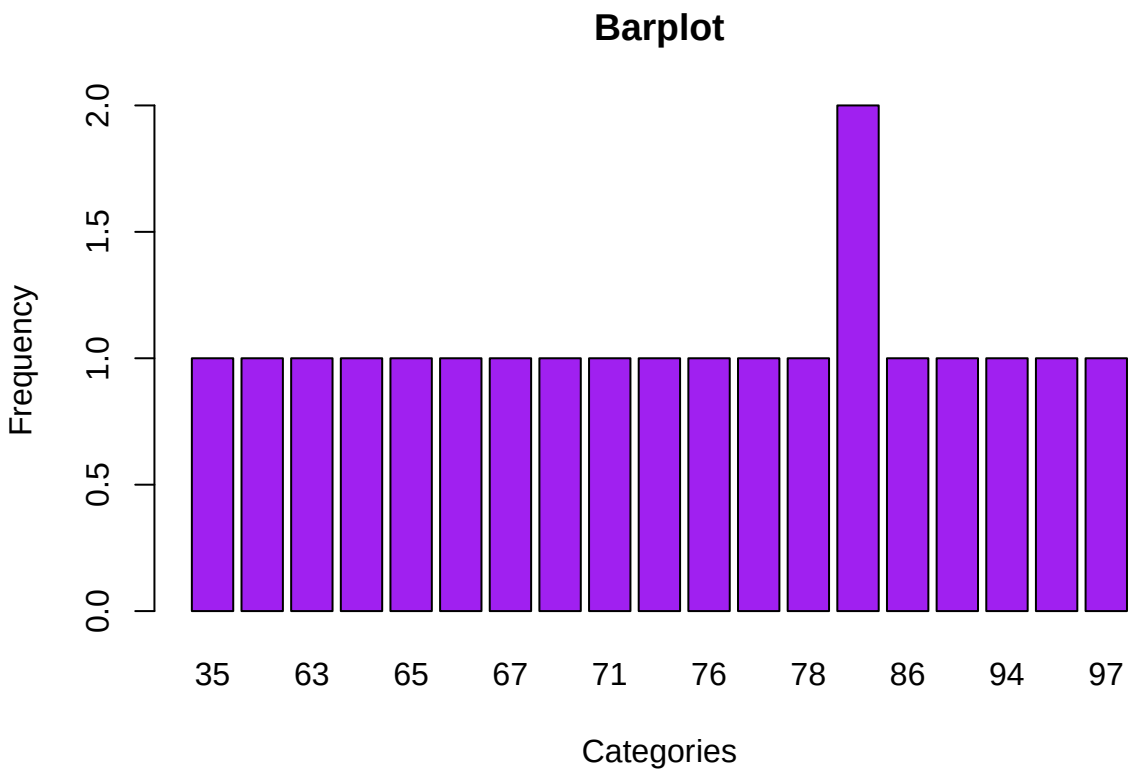

```
# Histogram
hist(df_combined$score, col = "lightblue", main = "Histogram", xlab = "Value",
     breaks = 10)
```




```r
# Boxplot
boxplot(df_combined$score, main = "Boxplot", col = "lightgreen")
```




```r
# Barplot
barplot(table(df_combined$score), col = "purple", main = "Barplot",
        xlab = "Categories", ylab = "Frequency")
```



5.2. ggplot

`ggplot` is een krachtig en consistent systeem voor datavisualisatie in R, gebaseerd op het principe van Grammar of Graphics. Het werkt modulair: grafieken worden opgebouwd in lagen.

```
library(ggplot2)
library(dplyr)
```

5.2.1. Basisstructuur van een ggplot Elke plot in `ggplot2` begint met de functie `ggplot()`, die de dataset en esthetiek (aesthetic mappings) definieert. Vervolgens voeg je een geometrie toe (zoals een punt, lijn, of boxplot) via `geom_*()` functies.

De algemene structuur is:

```
ggplot(data = dataset, aes(x = x_variable, y = y_variable, color = group_variable)) +
  geom_*()
```

Belangrijke componenten:

- `data`: De dataset die je gebruikt.
- `aes()`: De esthetiek die bepaalt hoe de variabelen worden weergegeven (bijv. x-as, y-as, kleur).
  - `x`: Variabele voor de x-as.

- `y`: Variabele voor de y-as.
- `color`: Kleur van de punten, lijnen, etc. (vaak gebruikt voor categorische variabelen).
- `size`: Grootte van de punten of lijnen.
- `shape`: Vorm van de punten.
- `fill`: Vulkleur voor objecten zoals bars of gebieden.

5.2.2. `geom_*` functies De geometrieën bepalen wat er wordt 'getekend' op een plot. Elk type plot in ggplot2 wordt gemaakt door een geometrie toe te voegen via een geom_*() functie.

Belangrijke geometrieën zijn:

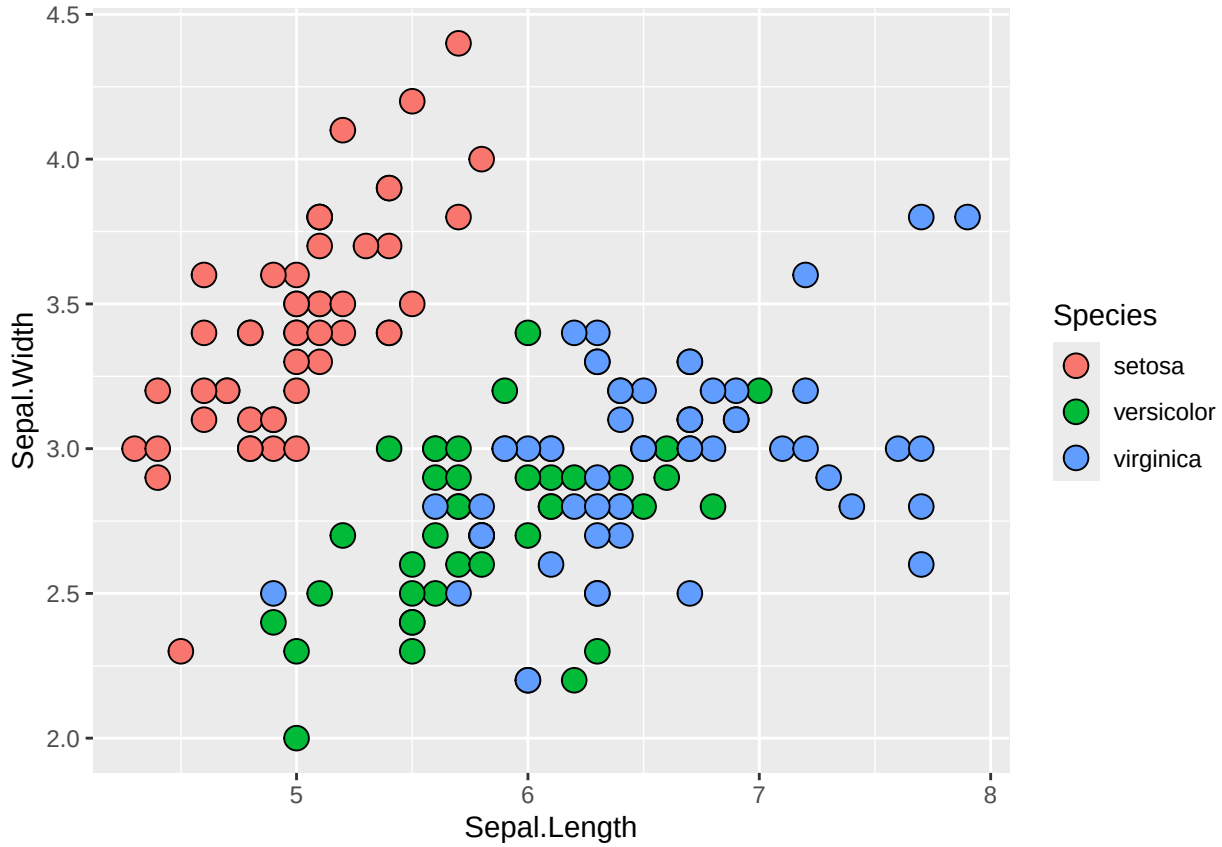
- `geom_point()`: Voor een scatterplot, waarbij punten worden getekend.
- `geom_line()`: Voor een lijngrafiek.
- `geom_histogram()`: Voor een histogram.
- `geom_boxplot()`: Voor een boxplot.
- `geom_bar()`: Voor een staafdiagram (meestal voor categorische variabelen).
- `geom_smooth()`: Voor het toevoegen van een trendlijn

Voorbeeld

```
# We gebruiken de standaard Iris dataset die beschikbaar is in base R
head(iris, 6)
```

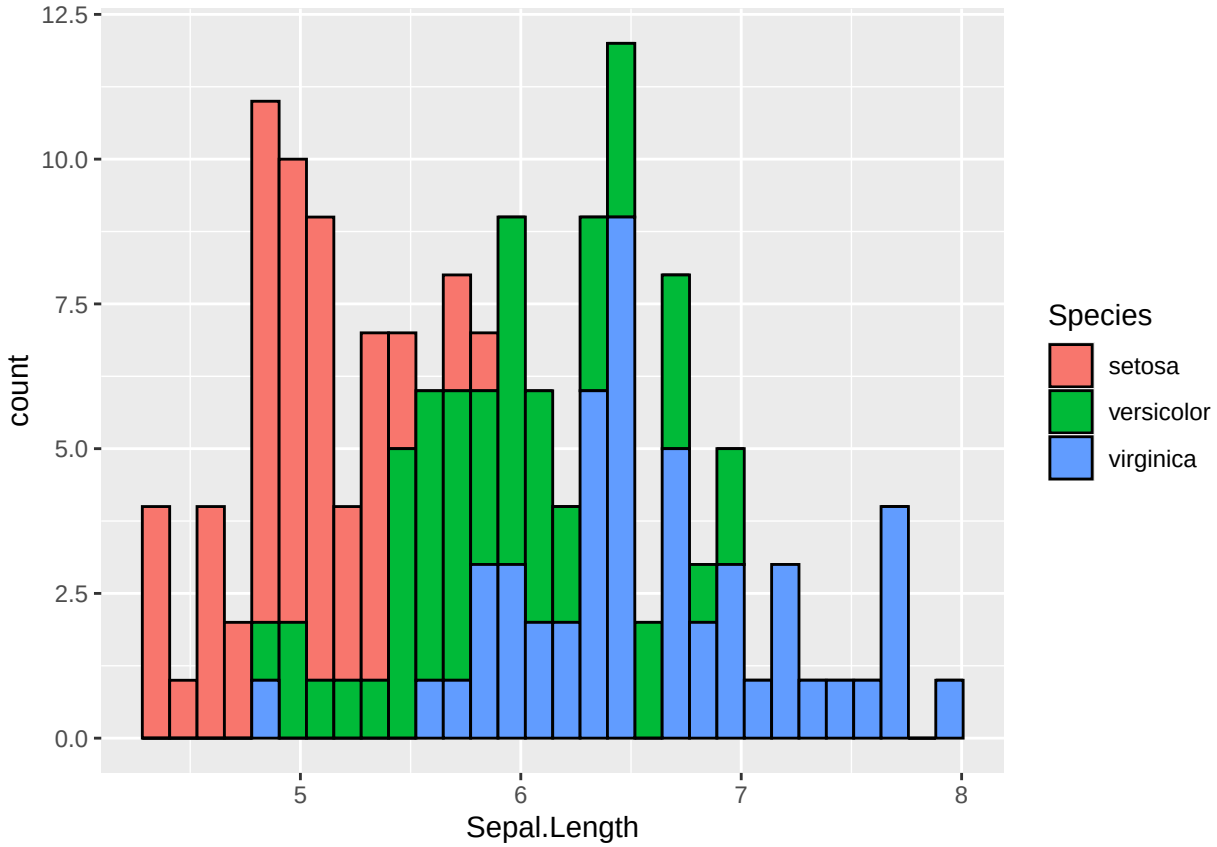
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
# Basis scatterplot van naam tegen score
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, fill = Species)) +
  geom_point(size = 4, shape = 21, color = "black")
```




```
# Basis scatterplot van naam tegen score
ggplot(data = iris, aes(x = Sepal.Length, color = Species, fill = Species)) +
  geom_histogram(size = 0.5, color = "black", bins = 30)
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

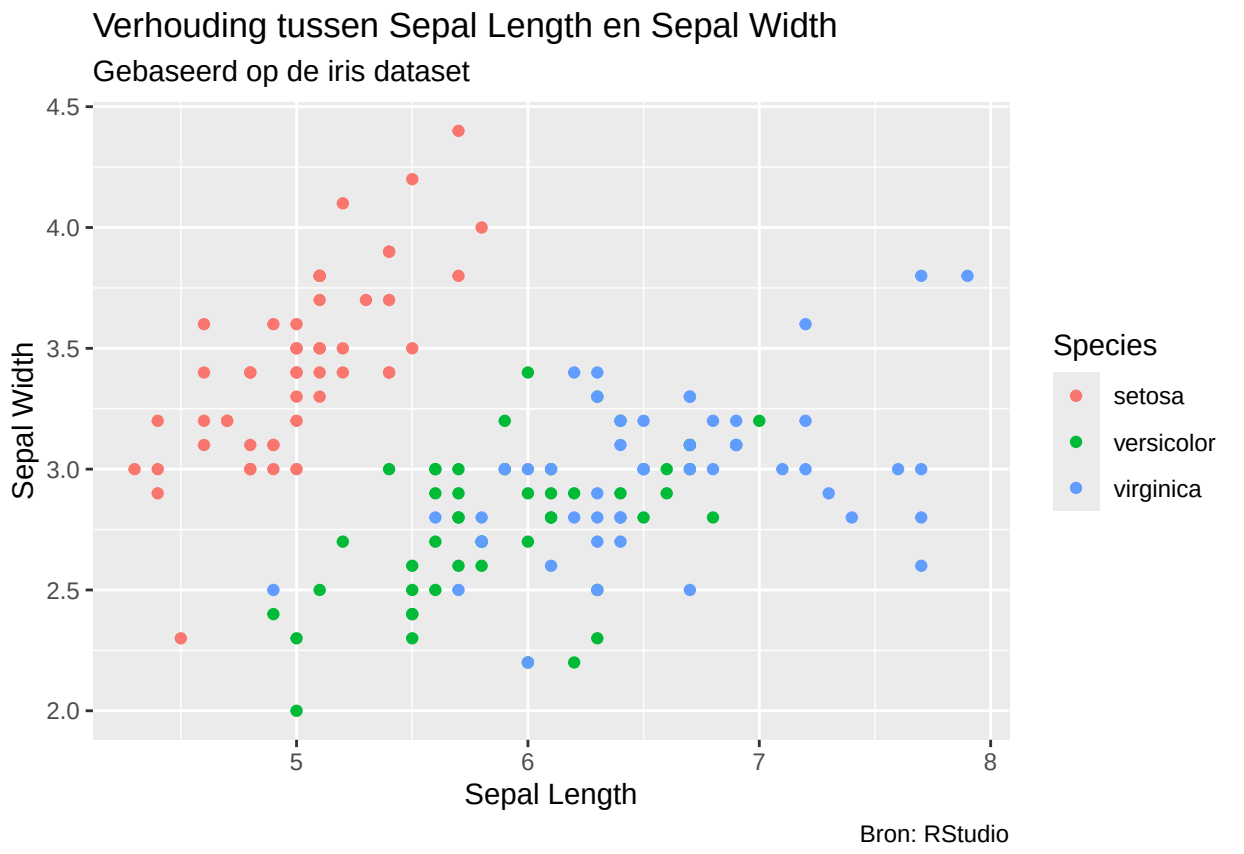


5.2.3. Labels en titels Met de functie labs() kan je belangrijke tekstuele elementen aan je grafiek toevoegen, zoals de plot titel, de as-labels, een ondertitel, en de tekst voor de legenda.

Belangrijke argumenten van `labs()`:

- `title`: De titel van de plot.
- `subtitle`: De ondertitel van de plot.
- `x`: Het label voor de x-as.
- `y`: Het label voor de y-as.
- `caption`: Een bijschrift onderaan de plot.
- `color`: Het label voor de legenda die wordt gebruikt bij een color esthetiek.

```
# Basis scatterplot van naam tegen score
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point() +
  labs(
    title = "Verhouding tussen Sepal Length en Sepal Width",
    subtitle = "Gebaseerd op de iris dataset",
    x = "Sepal Length",
    y = "Sepal Width",
    caption = "Bron: RStudio"
  )
```



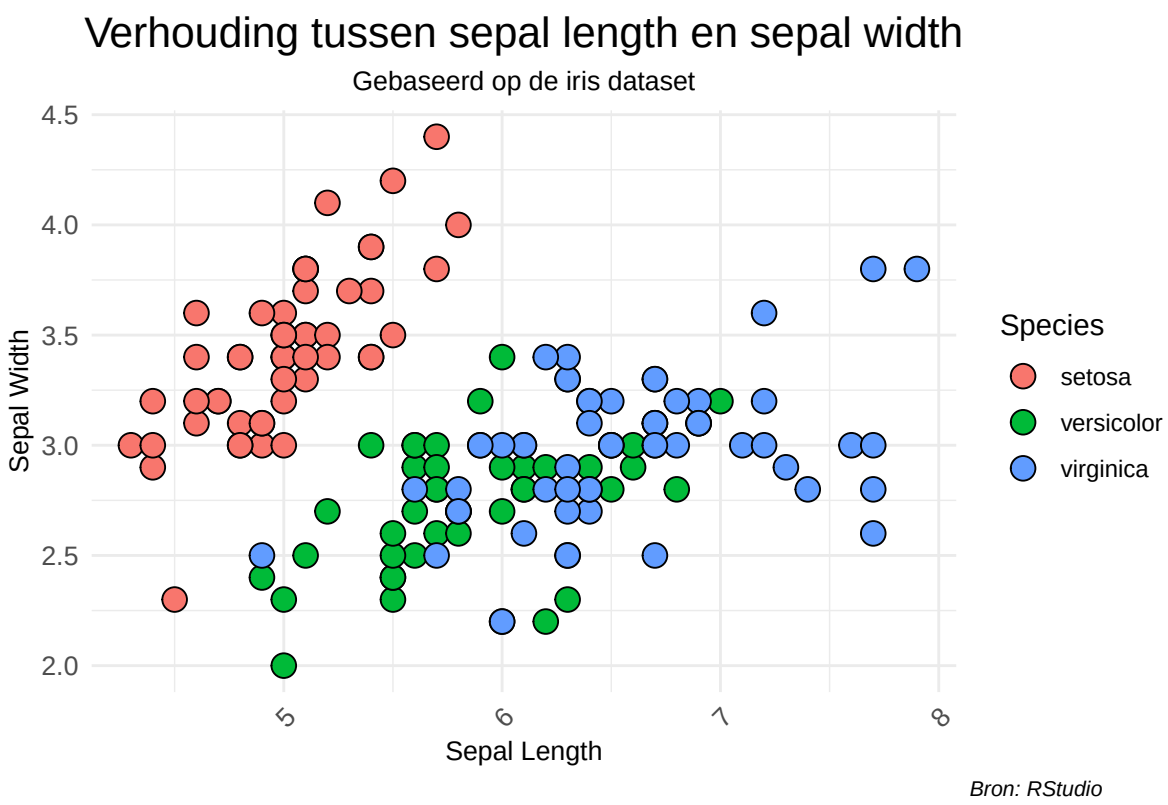
5.2.4. Thema's en stijlen Met `theme()` kun je de algehele uitstraling van je grafiek aanpassen. Dit stelt je in staat om de achtergrond, as-titels, as-etiketten, en andere visuele elementen aan te passen.

Belangrijke argumenten:

- `theme_minimal()`, `theme_classic()`, `theme_dark()`: Voorgeconfigureerde thema's die een plot een andere visuele stijl geven.
- `axis.text`: Aangepaste tekst voor de assen.
- `axis.title`: Aangepaste titel voor de assen.
- `plot.title`: Aangepaste titel voor plot.
- `panel.background`: Achtergrondkleur van plot.

```
# Basis scatterplot van naam tegen score
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, fill = Species)) +
  geom_point(size = 4, shape = 21, color = 'black') +
  labs(
    title = "Verhouding tussen sepal length en sepal width",
    subtitle = "Gebaseerd op de iris dataset",
    x = "Sepal Length",
    y = "Sepal Width",
    caption = "Bron: RStudio"
  ) +
```

```r
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, hjust = 0.5),
    plot.subtitle = element_text(size = 10, hjust = 0.5),
    axis.title.x = element_text(size = 10, color = "black"),
    axis.title.y = element_text(size = 10, color = "black"),
    axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10),
    plot.caption = element_text(hjust = 1.2, size = 8, face = "italic"),
    plot.margin = margin(t = 10, r = 10, b = 20, l = 20)
  )
```



Bron: RStudio

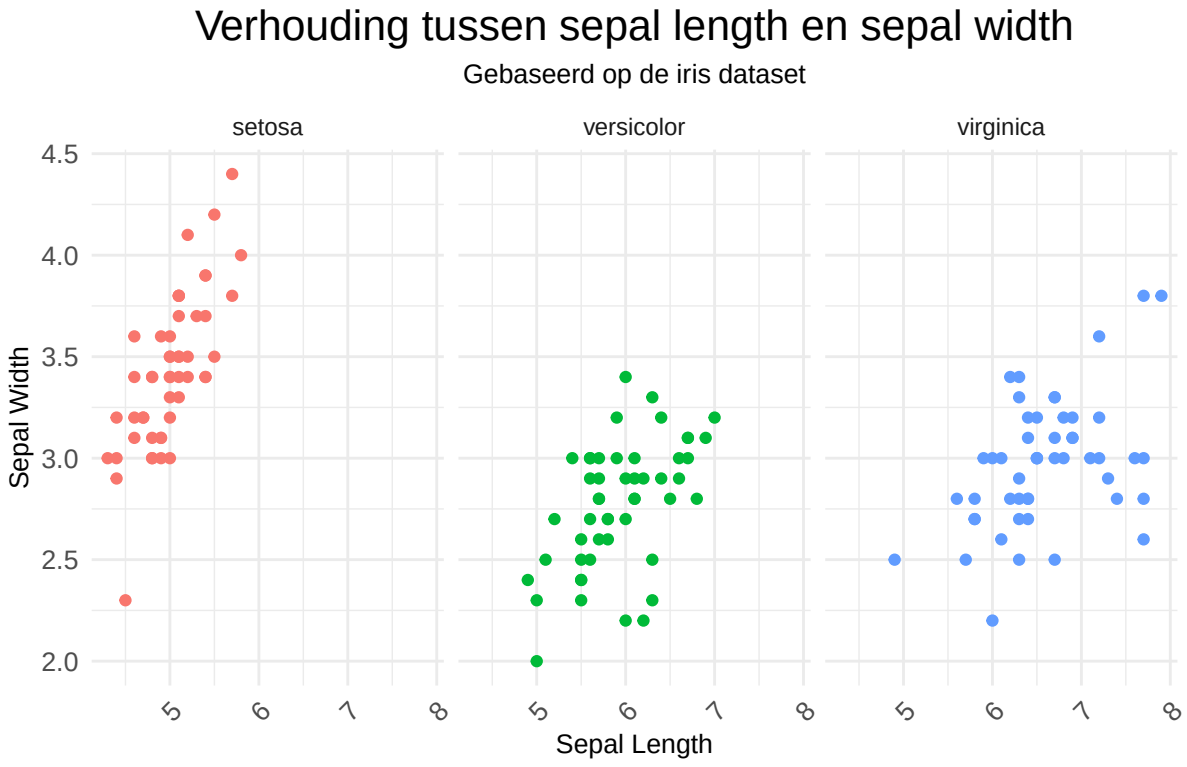
5.2.5. Facetting Facetting helpt je om meerdere subplots te maken, bijvoorbeeld door je data te splitsen op basis van een categorische variabele.

Belangrijke functies:

- `facet_wrap()`: Maakt facetten op basis van één variabele.
- `facet_grid()`: Maakt facetten op basis van twee variabelen (rijen en kolommen).

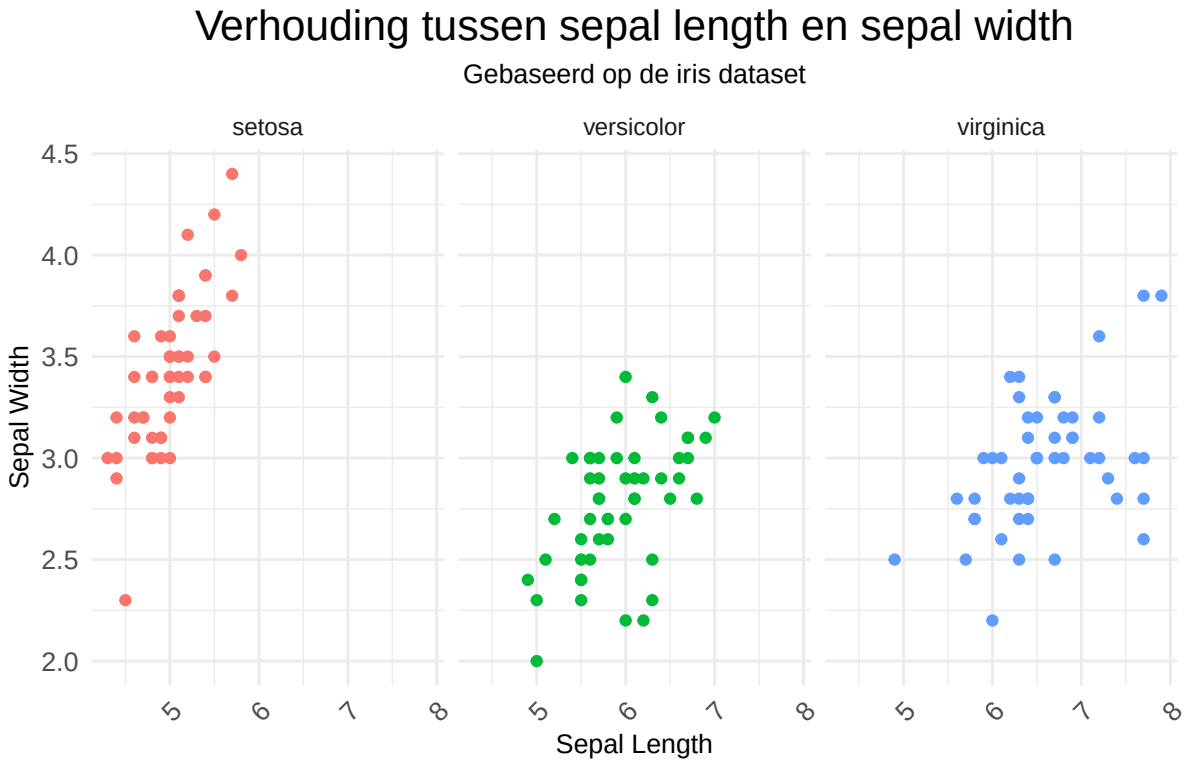
```r
# Basis scatterplot met facetten per soort
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_point(aes(color = Species)) +
  facet_wrap(~ Species, ncol = 3) +  # Horizontale facettering op basis van Species
```

```
  labs(
    title = "Verhouding tussen sepal length en sepal width",
    subtitle = "Gebaseerd op de iris dataset",
    x = "Sepal Length",
    y = "Sepal Width",
    caption = "Bron: RStudio"
  ) +
  theme_minimal()  +
  theme(
    plot.title = element_text(size = 16, hjust = 0.5),
    plot.subtitle = element_text(size = 10, hjust = 0.5),
    axis.title.x = element_text(size = 10, color = "black"),
    axis.title.y = element_text(size = 10, color = "black"),
    axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10),
    plot.caption = element_text(hjust = 1.2, size = 8, face = "italic"),
    plot.margin = margin(t = 10, r = 10, b = 20, l = 20),
    legend.position = "none"
  )
```




```
# Basis scatterplot met facetten per soort
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_point(aes(color = Species)) +
  facet_grid(. ~ Species) +  # Horizontale facettering op basis van Species
  labs(
```

```
    title = "Verhouding tussen sepal length en sepal width",
    subtitle = "Gebaseerd op de iris dataset",
    x = "Sepal Length",
    y = "Sepal Width",
    caption = "Bron: RStudio"
  ) +
  theme_minimal()  +
  theme(
    plot.title = element_text(size = 16, hjust = 0.5),
    plot.subtitle = element_text(size = 10, hjust = 0.5),
    axis.title.x = element_text(size = 10, color = "black"),
    axis.title.y = element_text(size = 10, color = "black"),
    axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10),
    plot.caption = element_text(hjust = 1.2, size = 8, face = "italic"),
    plot.margin = margin(t = 10, r = 10, b = 20, l = 20),
    legend.position = "none"
  )
```



---

6. Geavanceerde functies in R (basis)

R laat, net zoals andere programmeertalen, toe om ook zelf meer geavanceerde functies en algoritmen te schrijven. Zeker wanneer er gewerkt wordt met grote datasets, kan dit zeer handig zijn.

6.1. Even achter het gordijn kijken:

Door de naam van een functie te type zonder (), kan je nagaan wat de logica en source code is van een functie in R

```
theme_classic
```

```
## function (base_size = 11, base_family = "", base_line_size = base_size/22,
##     base_rect_size = base_size/22)
## {
##     theme_bw(base_size = base_size, base_family = base_family,
##         base_line_size = base_line_size, base_rect_size = base_rect_size) %+replace%
##         theme(panel.border = element_blank(), panel.grid.major = element_blank(),
##             panel.grid.minor = element_blank(), axis.line = element_line(colour = "black",
##                 linewidth = rel(1)), strip.background = element_rect(fill = "white",
##                 colour = "black", linewidth = rel(2)), complete = TRUE)
## }
## <bytecode: 0x000001c9734bc1b8>
## <environment: namespace:ggplot2>
```

6.2. Zelf functies schrijven in R

Een functie is een herbruikbare eenheid van code die een specifieke taak uitvoert. Dit bevordert modulariteit, leesbaarheid en reproduceerbaarheid.

Basisstructuur

```
mijn_functie <- function(argument1, argument2) {
  resultaat <- argument1 + argument2
  return(resultaat)
}

# Voorbeeldtoepassing
mijn_functie(3, 5)
```

```
## [1] 8
```

Voorbeeld met de `iris` dataset

```
# Gemiddelde lengte van kelkblad per soort
mean_sepal <- function(data, soort) {
  subset_data <- subset(data, Species == soort)
  mean(subset_data$Sepal.Length)
}

mean_sepal(iris, "setosa")
```

```
## [1] 5.006
```

6.3. Loops in R

Loops voeren een blok code herhaaldelijk uit op basis van iteraties of voorwaarden. Ze zijn nuttig voor repetitieve taken.

For-loop structuur

```
getallen <- c(1, 2, 3, 4, 5)
for (i in getallen) {
  print(i^2)
}
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
```

Voorbeeld met `iris` dataset

```
soorten <- unique(iris$Species)

for (soort in soorten) {
  gem <- mean(iris[iris$Species == soort, "Petal.Length"])
  cat("Gemiddelde bloembladlengte voor", soort, ":", gem, "\n")
}
```

```
## Gemiddelde bloembladlengte voor setosa : 1.462
## Gemiddelde bloembladlengte voor versicolor : 4.26
## Gemiddelde bloembladlengte voor virginica : 5.552
```

6.4. Conditionele logica

Conditionele statements sturen de uitvoering van code op basis van `TRUE`/`FALSE` condities.

`if`, `else`, en `ifelse`

```
x <- 10
if (x > 5) {
  print("x is groter dan 5")
} else {
  print("x is kleiner of gelijk aan 5")
}
```

```
## [1] "x is groter dan 5"
```

```
# Vectorized alternatief
vec <- c(3, 7, 9)
ifelse(vec > 5, "Groot", "Klein")
```

```
## [1] "Klein" "Groot" "Groot"
```

Toepassing op `iris` dataset

```r
iris$Breedte_categorie <- ifelse(iris$Sepal.Width > 3, "Breed", "Smal")
head(iris[, c("Sepal.Width", "Breedte_categorie")])
```

```
##   Sepal.Width Breedte_categorie
## 1         3.5             Breed
## 2         3.0              Smal
## 3         3.2             Breed
## 4         3.1             Breed
## 5         3.6             Breed
## 6         3.9             Breed
```

6.5. Combineren van functies, loops en condities

De kracht ligt er vaak in om functies, loops en condities te combineren. Dit maakt het mogelijk om dynamische analyses te automatiseren.

Voorbeeld: functie met loop én conditie

```r
analyse_soorten <- function(data) {
  soorten <- unique(data$Species)
  resultaten <- list()

  for (soort in soorten) {
    subset_data <- data[data$Species == soort, ]
    gem <- mean(subset_data$Petal.Length)

    if (gem > 4) {
      resultaten[[soort]] <- paste("Lang bloemblad (", gem, ")")
    } else {
      resultaten[[soort]] <- paste("Kort bloemblad (", gem, ")")
    }
  }
  return(resultaten)
}

analyse_soorten(iris)
```

```
## $setosa
## [1] "Kort bloemblad ( 1.462 )"
##
## $versicolor
## [1] "Lang bloemblad ( 4.26 )"
##
## $virginica
## [1] "Lang bloemblad ( 5.552 )"
```

7. Recap

Dit document biedt een overzichtelijke introductie tot het werken met R en RStudio, met als doel de gebruiker vertrouwd te maken met fundamentele concepten, workflows en technieken die essentieel zijn voor data-analyse en statistisch programmeren in R.

We zijn gestart met het gebruik van packages, waarbij het belang van het R-ecosysteem en de rol van CRAN werden toegelicht. Vervolgens kwam het correct instellen van de working directory en het gebruik van R Projects aan bod, als cruciale voorwaarden voor een reproduceerbare en gestructureerde werkomgeving.

Daarna besteedden we aandacht aan de basisprincipes van R, zoals variabelen, datatypes en functies. Op basis van deze fundamenten verkenden we meer toegepaste vaardigheden in het manipuleren van data aan de hand van het tidyverse, met specifieke aandacht voor dplyr, tidyr en readr. Hiermee legden we de basis voor het uitvoeren van gestructureerde en efficiënte datatransformaties.

Het onderdeel over datavisualisatie behandelde het gebruik van ggplot2, met voorbeelden van zowel eenvoudige als meerlagige grafieken. Deze technieken maken het mogelijk om complexe data op een inzichtelijke en visueel aantrekkelijke manier te presenteren.

Tot slot kwamen enkele geavanceerde functionaliteiten aan bod, waaronder het schrijven van eigen functies en het gebruik van control flow-structuren zoals for-loops en if-statements. Deze bouwstenen maken het mogelijk om analyses te automatiseren en schaalbaar op te zetten.

Deze notebook kan dienen als naslagwerk en vertrekpunt voor verder zelfstandig gebruik van R, zowel in een academische als in een professionele omgeving.