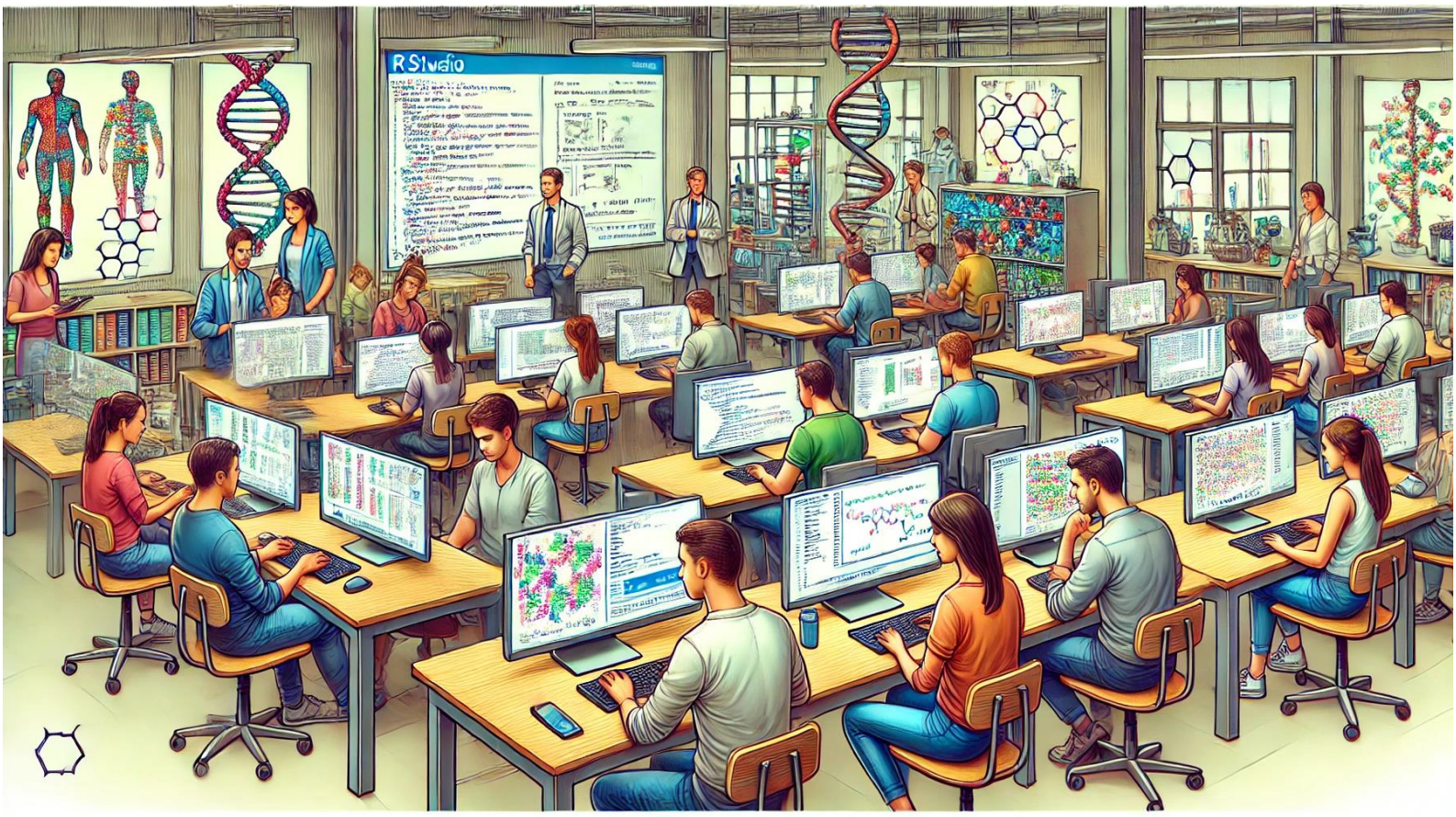
J2P3
Werkcolleges Data-analyse in R

Biologie en Medisch Laboratoriumonderzoek & Biotechnologie



NHL STENDEN hogeschool

van hall larenstein university of applied sciences

Overzicht Data-analyse in R

Lesnummer	DataCamp lessen	Opgave in document
1	**Introduction to R** 1. Intro to basics 2. Vectors 5. Data frames	**Opgave 1** **Opgave 2**
2	**Introduction to the Tidyverse** 1. Data wrangling 2. Data visualization	**Opgave 3** **Opgave 4**
3	**Introduction to the Tidyverse** 3. Grouping and summarizing 4. Types of visualizations	**Opgave 5** **Opgave 6**
4	**Introduction to Data Visualization with ggplot2** 1. Introduction 2. Aesthetics	**Opgave 7** **Opgave 8**
5	**Introduction to Data Visualization with ggplot2** 3. Geometries 4. Themes	**Opgave 9** **Opgave 10**
6	**Intermediate Data Visualization with ggplot2** 3. Facets	**Opgave 11**

In de tabel staan lesnummers weergeven. Wanneer deze lessen plaatsvinden verschilt per klas. Houd goed voor jezelf bij wanneer je elke les hebt zodat je bij blijft met de lesstof en de mogelijkheid hebt om vragen te stellen bij werkcolleges.

Na afloop van het tentamen lever je een script in. Het is dus handig om alle opgaven uit te werken in een script zodat je hiermee kunt oefenen. Daarnaast mag je uitgewerkte opdrachten meenemen.

Opgave 1: Hersen- en lichaamsgewicht

Van een aantal dieren is het lichaamsgewicht in kg en het gewicht van de hersenen in g bekend. Deze informatie staat in een tabel op BlackBoard: animal_weight.csv. Download dit bestand. Bekijk het gedownloade bestand met Kladblok (of een andere soortgelijke text editor). Doe dit **niet** in Word, Excel etc. zodat de opmaak niet veranderd wordt! Kijk welk scheidingsteken je kan vinden en of je tabel headers bevat.

a. Laad de dataset animal_weight.csv in R onder een zelfgekozen naam (zoals bijvoorbeeld `animals`) met `read.csv`. Geef hierbij aan wat de *separator* (scheidingsteken) is, en of er *headers* (rijnamen) in het bestand staan. Bekijk of het inladen goed is gegaan met `head()`.

b. Het is handig om te weten welk gewicht bij welk dier hoort. Maak een vector `animal_names` door het onderstaande hierin te zetten. Voeg deze vector aan de dataset toe als rijnamen met behulp van `rownames()`.

```
("Mountain beaver", "Cow", "Grey wolf", "Goat", "Guinea pig",
"Dipliodocus", "Asian elephant", "Donkey", "Horse", "Potar monkey",
"Cat", "Giraffe", "Gorilla", "Human", "African elephant",
"Triceratops", "Rhesus monkey", "Kangaroo", "Golden hamster", "Mouse",
"Rabbit", "Sheep", "Jaguar", "Chimpanzee", "Rat", "Brachiosaurus",
"Mole", "Pig")
```

c. Bekijk of de rijnamen goed zijn toegevoegd met behulp van `head()`. Bekijk ook de opbouw van de dataset met `str()`. Wat voor klassen hebben de variabelen? Van hoeveel dieren is het lichaamsgewicht bekend?

d. Bereken het gemiddelde lichaamsgewicht van alle dieren met behulp van `mean()`.

e. Laten we in de dataset gaan inzoomen op de lichte dieren. Selecteer uit de dataset alle dieren die een lichaamsgewicht van **maximaal** 1 kg hebben en sla dit op in `light_animals`. Bekijk hierna wat je in `light_animals` hebt gezet door de eerste tien bovenste rijen te bekijken.

f. Bekijk welk van deze dieren de zwaarste hersenen heeft in verhouding met hun lichaam. Let op: hersenen staan in g weergeven, lichaamsgewicht in kg. Je kan dan dus de hoeveelheid hersenen in gram per kg lichaamsgewicht van het dier berekenen. Deel hiervoor de `brain` variabele door de `body` variabele en sla dit op in een nieuwe variabele in de dataframe `light_animals$brain_body`. Welk dier heeft de zwaarste hersenen in verhouding met het lichaam?

Opgave 2: Groeisnelheid bacteriën

Van verschillende bacteriën is de groeisnelheid gemeten onder verschillende omstandigheden. De gegevens staan in bacteria.csv. Er zijn verschillende temperaturen en media gebruikt. Binnen deze media zijn er verschillende pH's getest en is er bij een aantal bacteriën wel of geen antibiotica toegevoegd. Download de data en bekijk deze in een text editor.

a. Laad de dataset bacteria.csv in R onder een zelfgekozen naam. Bekijk of het inladen goed is gegaan met `tail()`.

b. We gaan de tabel in wat meer detail bekijken. Je kunt de variabelen, in dit geval gecodeerd als kolomnamen, apart bekijken. Bekijk de variabelen met behulp van `colnames()` of `names()`. Kijk vervolgens hoe de kolom `Antibiotic` en `Temperature` gecodeerd staan met behulp van `class()`.

c. Verander de rijnamen naar de variabele `ExperimentID`. Verwijder vervolgens de eerste kolom volledig uit de dataset door deze te deselecteren. Zorg ervoor dat je de wijzigingen opslaat in R onder de naam waarin je data is opgeslagen.

We hebben nu te maken met een relatief kleine dataset. We kunnen daarom tellen hoevaak een bepaalde bacterie getest is. Als je meer data hebt wordt dit lastiger, en is het dus handig dit soort zaken te automatiseren.

d. Laat R uitrekenen hoevaak je *Lactobacillus casei* getest hebt door `sum()` te gebruiken.

e. Bekijk welke verschillende temperaturen getest zijn door `factor()` te gebruiken en de uitkomst hiervan te sorteren.

f. Maak een subset van alle bacteriën die blootgesteld zijn aan antibioticum en sla dit op onder een nieuwe, zelfgekozen naam. De laatste kolom (waarin staat of een bacterie is blootgesteld staat aan antibioticum) is niet meer nodig. Sla de eerste vijf kolommen (met alle bijbehorende rijen) op zodat de laatste verwijderd wordt onder je eerder zelfgekozen naam.

g. Iemand voert nog een aantal testen uit om te kijken of er evenveel groei plaatsvindt in een zuurdere omgeving na toevoeging van antibiotica. Voeg onderstaande vectoren toe in R.

```
Bacteria <- c("Bacillus subtilis", "Staphylococcus aureus")
GrowthRate <- c("0.2", "0.1", "0.3", "0.2")
Temperature <- c("39", "35", "35", "38")
Medium <- c("Agar", "Agar", "Broth", "Agar")
pH <- c("3.3", "3.3", "3.3", "3.3")
```

Het blijkt dat twee bacterienamen ontbreken. Dat zijn *Bacillus subtilis* en *Escherichia coli*. Voeg deze toe aan de vector `Bacteria`. Gebruik vervolgens `data.frame()` om de gegevens in een tabel op te slaan onder een nieuwe, zelfgekozen naam.

h. Het kan handig zijn om verschillende tabellen samen op te slaan in een *list*. Zo kan je bijvoorbeeld overzicht bewaren als je met veel tabellen werkt. Gebruik `list()` om de twee tabellen uit f en g samen te voegen en geef deze componenten (tabellen) handige, nieuwe namen in de *list*. Check vervolgens of het gelukt is.

Opgave 3: Hersen- en lichaamsgewicht deel 2

De vorige keer is een dataset gebruikt, waarbij van 28 dieren het lichaamsgewicht in kg en het gewicht van hersenen in g bekend was. Dit keer gebruiken we een grotere dataset, waarbij deze informatie voor 65 diersoorten beschikbaar is.

a. Download animal_weight2.csv van BlackBoard en laad de data in onder een zelfgekozen naam. Dit doe je nadat je hebt bekeken wat voor scheidingsteken erin staat en of de data headers bevat. Dit keer staan de rijnamen al bij de dataset in (moest je de vorige keer zelf invoegen). Check of het inladen goed is gegaan door de bovenste 15 rijen te bekijken.

b. Bij het werken met Datacamp zijn alle nodige packages al geïnstalleerd. Als je een nieuw package op je eigen laptop/computer wil gebruiken, is het dus nodig om deze eerst te installeren. Op een schoolcomputer moet je packages altijd eerst installeren. Op je eigen laptop hoeft dit maar één keer.
Installeer `dplyr` en `ggplot2` met behulp van `install.packages()`. Let erop dat je de namen van de packages tussen aanhalingstekens zet. Laad vervolgens de packages in met `library()`.

c. Maak een pipe (%>%), waarbij je filtert op een hersengewicht van 1 gram of meer. Sorteer de gefilterde data ook op basis van hersengewicht (hoog naar laag). Welk dier heeft de zwaarste hersenen? En welk dier in de gefilterde dataset de lichtste?

d. Maak een puntdiagram voor alle dieren, waarbij je het lichaamsgewicht op de x-as uitzet, en het gewicht van de hersenen op de y-as.

e. De puntdiagram is nog niet duidelijk af te lezen. Maak nog een keer dezelfde puntdiagram, maar nu door zowel voor de x-as, als de y-as, een logaritmische schaal in te voegen.

f. Het valt op dat de drie zwaarste dieren niet op een lijn liggen met de rest van de dieren. Als je in de dataset kijkt, valt het op dat dit de dinosaurussen zijn. Gebruik een pipe waarbij je de dinosaurussen (lichaamsgewicht > 9000 kg) uit de dataset filtert en sla dit op in een nieuwe dataset `animals_nodino`. Maak met deze nieuwe dataset weer een puntdiagram.

Opgave 4: Groeisnelheid planten

Een aantal planten zijn onder verschillende omstandigheden gegroeid. De hoeveelheid toegevoegd water verschilde net als de hoeveelheid lichtintensiteit (Lux). De totale hoogtes van de veschillende planten zijn gemeten na het aantal weken dat de plant heeft kunnen groeien (variërend). De dataset is te vinden op BlackBoard onder de naam plantheight.csv.

a. Bekijk wat voor scheidingsteken gebruikt is en of de data headers bevat. Laad vervolgens de gegevens in R onder een zelfgekozen naam. Check of het inladen goed is gegaan door de onderste rijen te bekijken. Zorg ervoor dat `dplyr` en `ggplot2` zijn ingeladen als je dit nog niet hebt gedaan.

Decimale scheidingstekens in R

In voorgaande datasets werd er gebruik gemaakt van een `.` als decimaal scheidingsteken. In veel Europese landen wordt echter vaak gebruik gemaakt van een `,` als decimaal scheidingsteken. R herkent je gegevens alleen als getallen als er gebruik wordt gemaakt van een `.` als scheidingsteken. Het is dus belangrijk om hier rekening mee te houden!

Als je gegevens hebt waarbij er gebruik wordt gemaakt van een `,` als decimaal scheidingsteken dan moet je dit bij het inladen correct aanpassen zodat R er een `.` van kan maken. Dit kan op verschillende manieren. Hieronder zijn twee voorbeelden weergeven:

```
data <-    read.csv("C:/pad/naar/data.csv",
           header = TRUE, sep = ";", dec = ",")
```

Of met behulp van `read.csv2`, welke rekening houdt met hoe Excel in Europese landen gegevens opslaat in CSV format.

```
data <-    read.csv2("C:/pad/naar/data.csv",
           header = TRUE, sep = ";")
```

b. Enkele lampen leken tijdens het experiment niet goed te werken. De lampen die niet goed werken hebben een gemeten `Lux` van 300 of minder. Gebruik een pipe om alle planten met werkende lampen over te houden en sla dit op onder dezelfde naam als die je bij a hebt gekozen.

Alleen de totale hoogte van de planten is gemeten. Achteraf blijkt dat planten minimaal 15 weken moeten groeien om een goed beeld van de groei te krijgen. Daarbij is het voor het vergelijken van de groei handig als deze wordt weergeven als gemiddelde groei per week.

c. Maak een pipe, waarin je filtert zodat alle planten overblijven die minimaal 15 weken hebben kunnen groeien. Voeg in dezelfde pipe met behulp van `mutate` een kolom toe waarin je de gemiddelde groei per

week berekent en geef deze nieuwe kolom een zelfgekozen naam. Sla het geheel op onder dezelfde naam als die je bij a hebt gekozen.

d. Voeg een puntdiagram toe van de data uit opdracht c met behulp van `ggplot2`. Zet daarin de groeiduur op de x-as, totale planthoogte op de y-as en gebruik de plantensoort als verschillende kleuren van de punten.

e. De visualisatie bij d gaf niet veel duidelijkheid. Voeg opnieuw een puntdiagram in met behulp van `ggplot2`. Zet de groeiduur weer op de x-as en de totale hoogte op de y-as. Voeg een kleurengradiënt van de punten in op basis van de lichtintensiteit. Laat de grootte van de punten afhangen van hoeveel water de plant heeft gekregen. Zet de plantensoorten in losse plotjes met behulp van `facet_wrap()`.

Opgave 5: Grootte van irissen

Bij deze opgave gebruik je een dataset van bloemen: irissen. Deze dataset staat al in 'basic' R. Om de dataset op te slaan onder de naam `iris`, typ je het volgende in: `data("iris")`. Bekijk de dataset door `str(iris)` in te typen. Je ziet dat de dataset bestaat uit informatie van 150 bloemen. Van deze bloemen is de lengte en breedte van zowel het kelkblad, als het bloemblad bekend. Alle maten zijn weergeven in cm. Ook staat in de dataset om welke soorten irissen het gaat: 3 soorten van elk 50 samples.

a. Laad de `dplyr` en `ggplot2` packages in R. Als je op een schoolcomputer werkt, kan het zijn dat je deze eerst opnieuw moet downloaden.

b. Maak een pipe, waarin je `summarize()` gebruikt om de gemiddelde lengte en de gemiddelde breedte van de kelkbladeren te samenvatten in: `mean_sepal_length` en `mean_sepal_width`.

c. Er zou een verschil in de grootte van de kelkbladeren tussen de soorten kunnen zitten. Maak een puntdiagram, waarbij je de lengte van de kelkbladeren op de x-as zet en breedte van de kelkbladeren op de y-as. Geef de punten een kleur op basis van welk soort iris het is. Wat zijn de verschillen tussen de soorten, kijkend naar de puntdiagram?

d. Zet de breedte van de kelkbladeren uit in een boxplot om de verschillen in breedte tussen de soorten beter te bekijken. Weergeef de soorten weer in verschillende kleuren. Welke soort heeft de breedste kelkbladeren?

e. Het is ook interessant om naar de grootte van de bloemblaadjes te kijken. Maak een pipe waarin je eerst de verschillende soorten irissen groepeert, waarna je met `summarize()` de mediane lengte en breedte van de bloemblaadjes opslaat in `median_petal_length` en `median_petal_width`. Welke soort heeft de grootste blaadjes? En welke de kleinste?

f. Vergelijk de lengte van de bloemblaadjes van de setosa en virginica soorten in een histogram. Maak daarvoor eerst een pipe waarin je alle soorten behalve de versicolor (met `!=`) filtert en opslaat in `iris_setosa_virginica`. Maak vervolgens een histogram voor de lengte van de bloemblaadjes, waarin je de twee soorten een aparte kleur geeft en de `binwidth` (= breedte van de staven) aanpast naar 0.5.

Opgave 6: Studietijd en scores

Van 50 studenten is bijgehouden hoeveel tijd zij aan studeren hebben besteed en wat de prestaties na afloop waren. Van de studenten is wat persoonlijke informatie (naam, leeftijd), informatie over de studie (welke, hoeveelste jaar) en de scores bekend. Scores van tentamen, opdrachten en deelname zijn bekend. De totale score is een combinatie van bovengenoemde scores. De data is te vinden op BlackBoard onder de naam student_performance.csv.

a. Download de data en laad dit in R. Check ook of het inladen goed is gegaan met behulp van `head()` en `str()`. Laad `dplyr` een `ggplot2` in R als je dit nog niet hebt gedaan.

b. Het kan interessant zijn om de scores per opleiding te bekijken. Naast het gemiddelde kan de spreiding van scores ook handige informatie weergeven. Maak een pipe, waarin je een groepering maakt voor de verschillende opleidingen (`major`). Vat vervolgens de gemiddelde totale score per opleiding samen als `avg_score`, de minimumscore als `min_score` en de maximale totale score als `total_score`. Gebruik hiervoor `summarize()`.

c. Laten we analyseren of we verschillende invloeden kunnen zien op de gemiddelde studieduur. Om meerdere factoren tegelijk te bekijken, kunnen we meerdere groeperingen maken in één pipe. Groepeer de studenten op studiejaren en vervolgens het geslacht. Vat vervolgens samen hoeveel uren deze groeperingen gemiddeld gestudeerd hebben in `avg_hours`.

Visualisaties kunnen helpen ter verduidelijking. Zo kan dit bijvoorbeeld inzicht geven in de gemiddelde score per studiejaar (en de mogelijke verandering hiervan in de verschillende jaren).

d. Maak een pipe waarin je groepeert op basis van studiejaar en de gemiddelde toetsscores opslaat in `avg_test_score`. Voeg vervolgens een lijndiagram in, waarbij je het studiejaar weergeeft op de x-as en de gemiddelde toetsscore per studiejaar op de y-as. De y-as wordt automatisch aangepast op basis van de opgegeven waardes. Laat de y-as van 0 tot 100 lopen om alle mogelijke scores mee te nemen.

In voorheengaande opdrachten hebben we altijd een pipe apart gebruik voordat we begonnen aan visualisaties. Je kunt een visualisatie ook deel uit laten maken van een pipe.

e. Maak een pipe waarin je een groepering maakt per opleiding en de gemiddelde studieduur samenvat in `avg_hours`. Voeg in dezelfde pipe een staafdiagram toe waarbij je de studie op de x-as zet, de gemiddelde studieduur op de y-as en de opleiding gebruikt als opvulkleur voor de staven.

Introduction to Data Visualization with ggplot2 deel 1/2

Opgave 7: slaap van zoogdieren

Bij deze opdracht gebruik je een dataset waar informatie over slaap van zoogdieren in staat. Hierin staat onder andere informatie over hoe lang verschillende dieren slapen, hoe lang hun remslaap duurt, hoe lang ze wakker zijn en hoe zwaar ze zijn. Laad het `ggplot2` package in R en laad de data in met data `data("msleep")` en bekijk de data met `str(msleep)`.

a. Sla een basis voor een plot op onder de naam `sleep_total_rem`, waarbij je de totale hoeveelheid slaap op de x-as zet, en de hoeveelheid remslaap op de y-as. Voeg nog geen grafiektype toe.

b. Maak een puntdiagram met `sleep_total_rem`. Voeg om een eventueel verband te bekijken `geom_smooth()` in. Met geom_smooth() wordt (met de standaardinstellingen) een 95% betrouwbaarheidsinterval van het lineaire model toegevoegd.

c. Maak dezelfde grafiek als je bij b hebt gedaan, maar voeg nu labels op de x-as en y-as toe, waarin je de variabele met de correcte eenheid, in dit geval uren, weergeeft. Geef de `geom_smooth` de kleur `darkolivegreen3`.

d. Je kan ook naar andere variabelen kijken. Maak een nieuwe puntdiagram, waarbij je het dieet (`vore`) van de dieren op de x-as zet, en de totale hoeveelheid slaap op de y-as. Voeg hierbij de duur van de remslaap als kleur toe. Voeg ook een `jitter` toe van 0.1, zodat de punten niet allemaal precies op één lijn liggen (en je plekken met veel punten dus beter kan zien).

e. Maak dezelfde plot, maar met een paar toevoegingen. Verander de grootte van de punten naar het aantal kilogram hersenen per kilogram lichaamsgewicht. Doe dit door een deling te maken (binnen de `aes()`). Voeg ook een `alpha` van 0.4 in (mate van doorzichtigheid), zodat overlappende punten nog zichtbaar blijven. Verander ook de jitter van 0.1 naar 0.2, zodat je minder overlap krijgt tussen de punten. Verander als laatste nog het label van de y-as naar: *total amount of sleep in hours*.

Opgave 8: verdeling sport studenten

Er zijn gegevens verzameld van een aantal sportende studenten en opgeslagen in studenten_sport.csv op BlackBoard. Bekend is in welk studiejaar de studenten zitten, welke sport ze beoefenen, hoe vaak de studenten per week sporten, een eigen gegeven gezondheidsscore (1-10), het gemiddelde cijfer binnen de opleiding en of de student lid is van een sportclub.

a. Download studenten_sport.csv en check hoe je data eruitziet in een tekst editor. Laad de gegevens in R en sla het op onder een zelfgekozen naam. Check of het inladen goed is gegaan met behulp van `str()`. Laad `dplyr` en `ggplot2` in (download indien nodig).

b. De variabele `Studiejaar` ga je gebruiken voor visualisaties. Het is daarvoor nodig dit eerst in een `factor` te veranderen, zodat R dit gesorteerd kan visualiseren. Vervang om dit te doen de variabele `Studiejaar` in je dataset met de `factor` hiervan. Voeg als levels de namen toe (`Eerstejaar, Tweedejaar` etc.). Zet het sorteren op `TRUE`.

c. Voeg een barplot in om het aantal studenten per studiejaar die lid zijn van een sportclub te visualiseren. Zet hiervoor het studiejaar op de x-as en gebruik of iemand lid is van een sportclub als opvulling voor de balken. Gebruik `position =` om de balken naast elkaar te weergeven in plaats van gestapeld. Voeg een zelfgekozen titel toe aan je plot.

d. Bereken het gemiddelde cijfer van personen in de verschillende studiejaren waarbij je rekening houdt of ze lid zijn van een sportclub. Gebruik hiervoor een pipe en groepeer op basis van studiejaar en sportclub. Sla de uitkomst op onder een zelfgekozen naam.

e. Voeg een lijndiagram toe van de uitkomst uit d. Zet het studiejaar op de x-as, het gemiddelde op de y-as en voeg de sportclub als groep toe. Sla de uitkomst op onder een zelfgekozen naam en bekijk je gemaakt plot.

f. In de lijndiagram uit e kan je nog niet aflzezen of iemand bij een sportclub zit of niet. Voeg aan je gemaakte lijndiagram als kleur toe of iemand bij een sportclub zit of niet.

In de praktijk voeg je steeds elementen toe aan je plot om je figuur op te helderen. Check bij elke toegevoegde stap hoe je plot eruitziet en voeg dan het volgende toe.

g. Visualiseer de gezondheidsscore tegen het gemiddelde cijfer in een puntdiagram:
   - Zet het gemiddelde cijfer op de x-as en de gezondheidsscore op de y-as van een puntdiagram;
   - Gebruik het geslacht als kleur voor de punten, het soort sport als vorm van de punten en vergroot de puntgrootte naar `5`;

- Voeg de IDs van de studenten toe met behulp van `geom_text()`. Verander de grootte van de labels naar `5`, geef de labels een zwarte kleur en geef ze een `alpha` van `0.5` zodat de punten nog goed zichtbaar blijven;
- Verander het bereik van de y-as zodat deze van 0-10 loopt;
- Verander de label van `Sportsoort` naar “sport” in je legenda;
- Geberuik `scale_color_manual` om zelf kleuren toe te voegen. Gebruik hiervoor `cyan3`, `gold2` en `hotpink2`.

Introduction to Data Visualization with ggplot2 deel 3/4

Opgave 9: Visualisatie grootte kelkbladeren

Voor deze opdracht gebruik je dezelfde dataset als je bij opdracht 3 hebt gebruikt. Laad de dataset `iris` weer op dezelfde manier in als je bij opdracht 3 gedaan hebt. Laad ook `ggplot2` in R. Omdat er erg veel `elements` zijn in R die veranderd kunnen worden (met `theme`), is het handig om hier een overzicht van te hebben. **Op deze site** kan je het overzicht van `elements` terugvinden. Het is handig om deze pagina bijvoorbeeld als favoriet op te slaan in je browser of de link toe te voegen aan je script zodat je het later weer kunt vinden.

a. Maak een puntdiagram, waarbij je de soort iris op de x-as zet, de lengte van de kelkbladeren op de y-as en de breedte van de kelkbladeren gebruikt als grootte.

b. Verander de gemaakte puntdiagram, waarbij je de grootte van de punten verandert door de lengte van de bloemblaadjes en de punten een beetje doorzichtig maakt (een `alpha` van `0.4`). Verander ook de kleuren van de punten naar `darkorchid3`. Verander als laatste de positie van de punten zodat ze minder overlappen met `position_jitterdodge()`, waarbij je voor zowel `jitter.width` als `dodge.width` een waarde van 0.3 gebruikt. Geef je plot een toepasselijke titel en labels op de assen.

c. Nu de plot er al redelijk uitziet, is het tijd om naar wat details te kijken met `themes()`. De verticale witte strepen zijn niet perse nodig, verwijder alleen de verticale strepen door hiervoor de goede optie te zoeken in `panel.grid` in de tabel (zie site onder opdrachtbeschrijving). Het is misschien ook mooi om de kleuren van de titel en de assen te veranderen naar `darkorchid4`. Verander alle tekst in de plot naar deze kleur met `element_text()`, en verander ook hierbij de grootte naar `20`.

d. Er zijn natuurlijk eindeloos veel manieren hoe je een plot op kan maken. Gelukkig zijn er ook wat voorgekauwde `themes`. Voeg de al bestaande `theme_classic()` toe aan de plot in plaats van de eigen toevoegingen.

Opgave 10: gedrag dieren in verschillende temperatuurzones

Van een aantal diersoorten is gedrag in verschillende temperatuurzones (warm, gematigd, koud) bijgehouden. Onder andere actitiveit, voedselconsumptie en gemiddelde beweginssnelheid is gemeten in de verschillende dieren. Download de dataset temperatuurzones.csv van BlackBoard.

a. Bekijk in een text editor hoe de dataset eruitziet. Laad de gegevens vervolgens in R onder een zelfgekozen naam. Check of het inladen goed is gegaan door de bovenste 10 rijen te bekijken. Zorg ervoor dat `ggplot2` en `dplyr` zijn ingeladen.

b. Bekijk welke variabelen nummerieken zijn binnen je dataset. Gebruik vervolgens twee keer `summarise_all()`. Sla de twee uitkomsten op onder een nieuwe, zelfgekozen naam. Bereken het gemiddelde voor alle variabelen. Bereken ook de standaarddeviatie voor al je variabelen. Bekijk wat je hebt opgeslagen

c. Analyseer de relatie tussen temperatuurzone (x-as) en activiteit (y-as) door dit uit te zetten in een puntdiagram. Gebruik de diersoorten als kleuren voor je punten. Zorg dat de punten elkaar niet overlappen door gebruik te maken van `geom_jitter()` met een breedte van `0.1`. Voeg een passend bijschrift toe voor de x-as, y-as en geef je figuur een titel. Voeg daarna de bestaande `theme_minimal` toe.

d. Voeg een histogram toe om de voedselconsumptie per temperatuurzone te analyseren. Zet hiervoor je voedselconsumptie uit op de x-as en gebruik de temperatuurzone als opvulkleur. Voeg onderstaande toe:
   - Voeg binnen `geom_histogram()` toe dat je balkjes een breedte moeten hebben van `50` en naast elkaar moeten komen te staan (dus niet gestapeld). Zorg er ook voor dat ze uitgelijnd worden (met `center`) op `25` voor het gemakkelijk aflezen;
   - Voeg het `Pastel1` toe met behulp van `scale_fill_brewer()`;
   - Voeg een passend bijschrift toe voor de x-as en y-as;

e. Visualiseer de gemiddelde activiteit van de dieren per temperatuurzone. Doe dit door:

1. Een pipe te maken. Sla de uitkomst op onder een zelfgekozen naam. Maak eerst een groepering van de temperatuurzones en diersoorten. Gebruik `summarize()` om de gemiddelde activiteit te berekeken en sla dit op als `GemActiviteit`. Bereken de standaardfout van de gemiddeldes door gebruik te maken van: `SEM = sd(Activiteit) / sqrt(n()`. (Ter herinnering: formule was S / wortel(n)).
2. Vul vervolgens onderstaande code aan op de rood gemarkeerde plekken om foutbalken toe te voegen.

```
ggplot(samenvatting, aes(x = Temperatuurzone, y = GemActiviteit, fill =
Soort)) +
  geom_col(position = "dodge") +
  geom_errorbar(aes(ymin = ..., ymax = ... + SEM),
                position = position_dodge(0.9), width = 0.2)
```

In Snelheden.csv op BlackBoard staat aanvullende informatie. De snelheid van geboren dieren is getraceerd en op verschillende momenten gemeten gedurende 12 maanden. De gemiddelde snelheid en standaarddeviatie hiervan is in de dataset te vinden.

f. Laad Snelheden.csv in onder een zelfgekozen naam. Voeg een lijndiagram in met de tijd op de x-as en de gemiddelde snelheid op de y-as. Maak de lijn de kleur `darkolivegreen4`. Voeg een betrouwbaarheidsinterval aan de lijn toe met behulp van `geom_ribbon()`. Gebruik hiervoor de SE uit je ingeladen data als minimum en maximum (verschil met het gemiddelde). Gebruik als opvulkleur `darkolivegreen4` en een `alpha` van `0.5` (zodat je de lijn erachter nog goed kunt zien).

g. Vul onderstaande code (rood gemarkeerd) aan om de bewegingssnelheid van dieren per temperatuurzone te visualiseren in boxplots. Maak de gridlines grijs van kleur en voeg een stippellijn toe. Zorg ervoor dat de achtergrond van je paneel de kleur `gray95` krijgt. Zorg ervoor dat je legenda boven je figuur (maar nog wel onder de titel) weergeven wordt.

```
ggplot(temperatuurzones, aes(x = Soort, y = Bewegingssnelheid, fill =
Temperatuurzone)) +
  geom_boxplot() +
  theme( axis.title = element_text(face = "bold"),
         panel.grid.major = element_line(...,
                                         ...),
         panel.background = ...(fill = "gray95"),
         ... = "...",
         legend.margin = margin(t = 10, r = 10, b = 10, l = 10) ) +
  labs( title = "Bewegingssnelheid per Diersoort en Temperatuurzone",
        x = "Soort",
        y = "Bewegingssnelheid (km/u)" )
```

Intermediate Data Visualization with ggplot2 (facets)

Opgave 11: Tolerantie gras tegen kou

Bij deze opdracht gebruik je een dataset waarin resultaten staan van een onderzoek naar de tolerantie van gras tegen kou. Gras samples zijn in twee verschillende regio's (Quebec en Mississippi) onder verschillende condities (gekoeld en niet gekoeld) gekweekt en vervolgens getest op CO_2 opname. Ook deze dataset staat in basic R, en kan je inladen met `data("CO2")`. Bekijk na het inladen hoe deze dataset eruit ziet door de eerste 6 rijen te bekijken. Laad ook het ggplot2 package in.

a. Maak een puntdiagram, waarbij de concentratie van CO_2 in de omgeving op de x-as staat, de CO_2 opname van de plant op de y-as staat en de behandeling als kleur is weergeven. Plot met behulp van `facet_grid()` de planten die gekweekt zijn op de verschillende locaties los van elkaar (in kolommen).

b. Het lijkt er op dat er nog een variabele is die invloed heeft op de CO_2 opname. Naast de behandeling, die in kleur is weergeven, is er namelijk nog meer spreiding binnen de verschillende behandelingen te zien. Kijk in de dataset welke variabele nog meer invloed zou kunnen hebben op de CO_2 opname van de planten, en gebruik deze variabele binnen de `facet_grid()` om dit los van elkaar op rijen te plotten.

c. Je weet nu welke variabelen allemaal invloed hebben op de CO_2 opname van de planten. De manier hoe dit bij *b* gevisualiseerd is, is alleen nog niet handig om hiermee een vergelijking te kunnen doen. Maak een plot, waarbij de concentratie CO_2 in de lucht op de x-as staat, de opname van CO_2 door de planten op de y-as en de verschillende planten als kleur zijn weergeven. Voeg dit keer een lijndiagram toe in plaats van een puntdiagram. Plot de plek waar de planten gekweekt zijn apart in kolommen, en de verschillende behandelingen apart in rijen.

d. De plot lijkt nu al wat overzichtelijker, maar het is handig om nog enkele dingen toe te voegen zodat er een vergelijking gedaan kan worden. De lijnen zijn smal, en daardoor soms lastig te lezen. Maak de lijnen wat dikker. Ook is het handig om de kleuren van de lijnen aan te passen, omdat ze nu erg veel op elkaar lijken. Voeg hiervoor `+ scale_color_discrete()` toe aan de plot. Voeg voor de variabelen die je in de `facet_grid()` hebt gezet labels toe, zodat je weet op basis van welke variabelen de rijen en kolommen zijn gesplitst. Voeg als laatste nog `margins` toe waarin de verschillende (en alle) combinaties van gesplitste variabelen samen in staan. Wat valt je op aan de plot?


NHL
STENDEN
hogeschool
van hall
larenstein
university of applied sciences