Agregate functie (Samenvoegen)

Inleiding

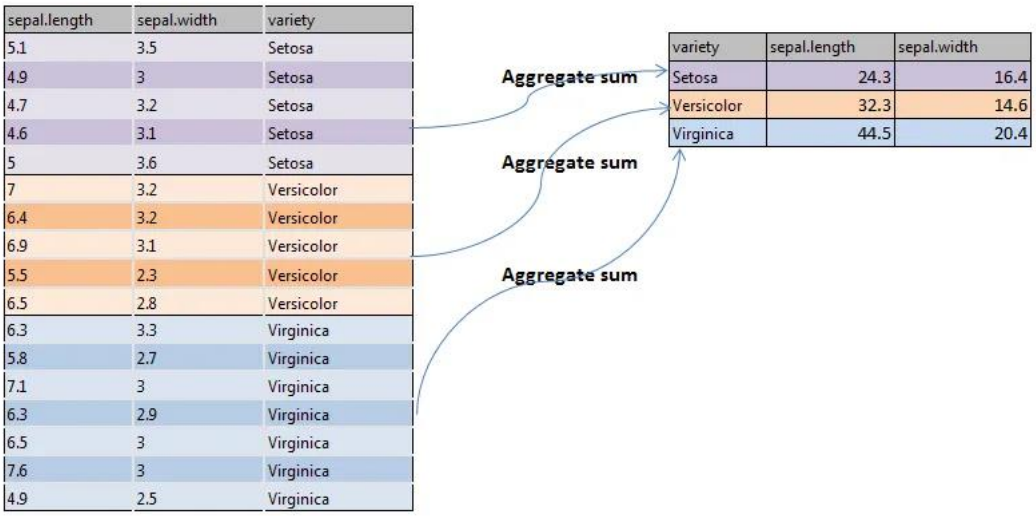
Aggregate() Functie in R Splitst de gegevens in subsets, berekent samenvattende (*summary*) statistieken voor elke subset en retourneert het resultaat in een groep op vorm. De aggregatiefunctie in R is vergelijkbaar met groeperen op in SQL.

De functie **Aggregate()** is handig bij het uitvoeren van alle aggregatiebewerkingen zoals *som*, *aantal*, *gemiddelde*, *minimum* en *maximum*.

Bekijk het volgende voorbeeld:

- **Aggregate()** die *groepssom* berekent
- **Aggregate()** functie die de groep *max* en *minimum* berkenet.
- **Aggregate()** functie die het *groepsgemiddelde* berekent
- **Aggregate()** functie die groepsaantallen ophaalt

Een pictografische weergave van de **aggregaat()**-functie, d.w.z. de geaggregeerde som, wordt hieronder weergegeven:



In deze tutorial leer je hoe je de aggregatiefunctie toepast in de programmeertaal R. De inhoudsopgave ziet er als volgt uit:

1. Definitie en basis R-syntaxis van geaggregeerde functie
2. Aanmaken van voorbeeldgegevens
3. **Voorbeeld 1**: Bereken gemiddelde per groep met behulp van aggregatiefunctie
4. **Voorbeeld 2**: Bereken de som per groep met behulp van de aggregatiefunctie

5. **Voorbeeld 3**: Geaggregeerde functie toepassen op gegevens die NA's bevatten

Definitie & Basic R Syntax van de aggregate Functie

Definitie: De geaggregeerde R-functie berekent samenvattende statistieken van subgroepen van een gegevens-set.

Basis R-syntaxis: Je kunt de basis R-programmeersyntaxis van de aggregatiefunctie hieronder vinden.

```
# Basic R syntax of aggregate function
aggregate(x = any_data, by = group_list, FUN = any_function)
```

aggregate(x, by, FUN, ..., simplify = TRUE, drop = TRUE)

X	an R object, Mostly a dataframe
by	a list of grouping elements, by which the subsets are grouped by
FUN	a function to compute the summary statistics
simplify	a logical indicating whether results should be simplified to a vector or matrix if possible
drop	a logical indicating whether to drop unused combinations of grouping values.

Er is ook een andere versie van de aggregate functie gebaseerd op de formula method.

```
# Basic R syntax of aggregate function (formula method)
aggregate(formula, data, FUN, ..., subset, na.action = na.omit)
```

formula	a formula, such as y ~ x or cbind(y1, y2) ~ x1 + x2, where the y variables are numeric data to be split into groups according to the grouping x variables (usually factors)
data	A dataframe object were the elemets will be pulled
FUN	a function to compute the summary statistics
…	further arguments passed to or used by methods. (i.e. simplify, drop, etc.)

In het volgende scrip zal doormiddel van drie voorbeelden uitgewerkt worden hoe de aggregatiefunctie in R toegepast zou kunnen worden.

Bereken gemiddelde per groep met behulp van aggregatiefunctie

In **Voorbeeld 1** zal ik uitleggen hoe je de aggregatiefunctie kunt gebruiken om het gemiddelde van elke subgroep en van elke variabele van onze voorbeeldgegevens te retourneren.

Binnen de aggregatiefunctie moeten we drie argumenten specificeren:

- De invoergegevens.
- De groeperingsindicator.
- De functie die we op elke subgroep willen toepassen.

Hieronder zie je het resultaat van de data die aangemaakt is voor uit uitvoeren van de agreggate functie.

Console Terminal Jobs

~/

```
> data
  x1 x2 x3 group
1  1  2  1     A
2  2  3  1     A
3  3  4  1     B
4  4  5  1     C
5  5  6  1     C
>
```

Na uitvoer van de code, in *de regels 9 t/m 10* van de code, van het eerste voorbeeld ziet de uitvoer er als volgt uit:

Console Terminal Jobs

~/

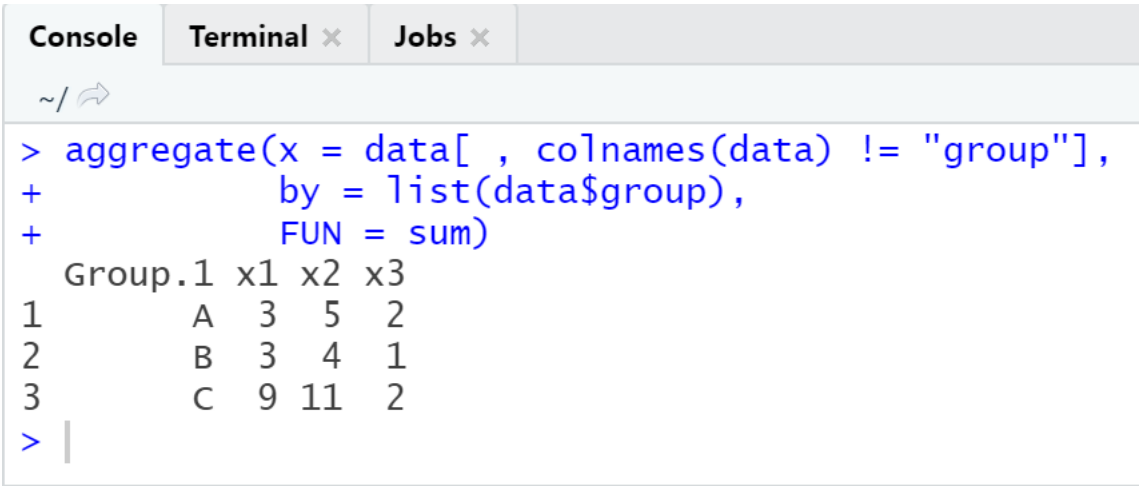
```
> aggregate(x = data[ , colnames(data) != "group"],
+           by = list(data$group),
+           FUN = mean)
  Group.1  x1  x2 x3
1       A 1.5 2.5  1
2       B 3.0 4.0  1
3       C 4.5 5.5  1
>
```

Zoals je kunt zien, heeft de RStudio-console het gemiddelde geretourneerd voor elke subgroep (d.w.z. A, B en C) voor elk van onze numerieke variabelen (d.w.z. x1, x2 en x3).

Merk op dat we de groeperingsindicator moesten uitsluiten van ons dataframe en merk ook op dat we de groeperingsindicator moesten converteren naar een lijst. Dit zijn nodig voorwaarden van de aggregatiefunctie.

In **Voorbeeld 2** bereken we de som per groep met behulp van de aggregatiefunctie.

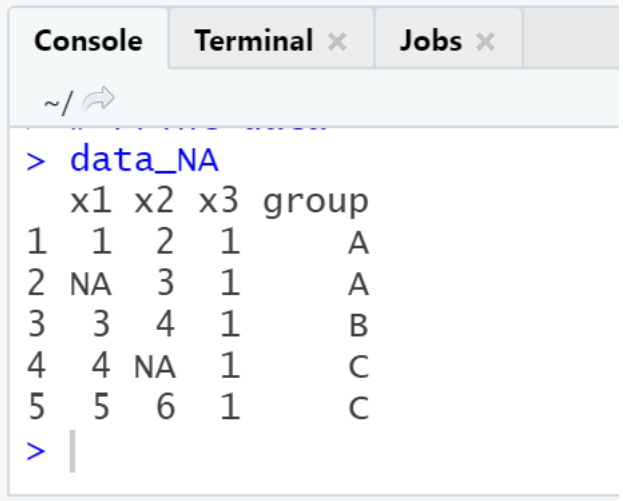
In het vorige voorbeeld hebben we het gemiddelde van elk subgroep berekend over meerdere kolommen van ons dataframe. Echter, het is gemakkelijk mogelijk om andere functies toe te passen binnen de aggregaat instructie. In voorbeeld 2 wordt geïllustreerd hoe je de som per groep retourneert met behulp van de aggregatiefunctie, in *de regels 13 t/m 15*. Hieronder zie je het resultaat van deze uitvoer.



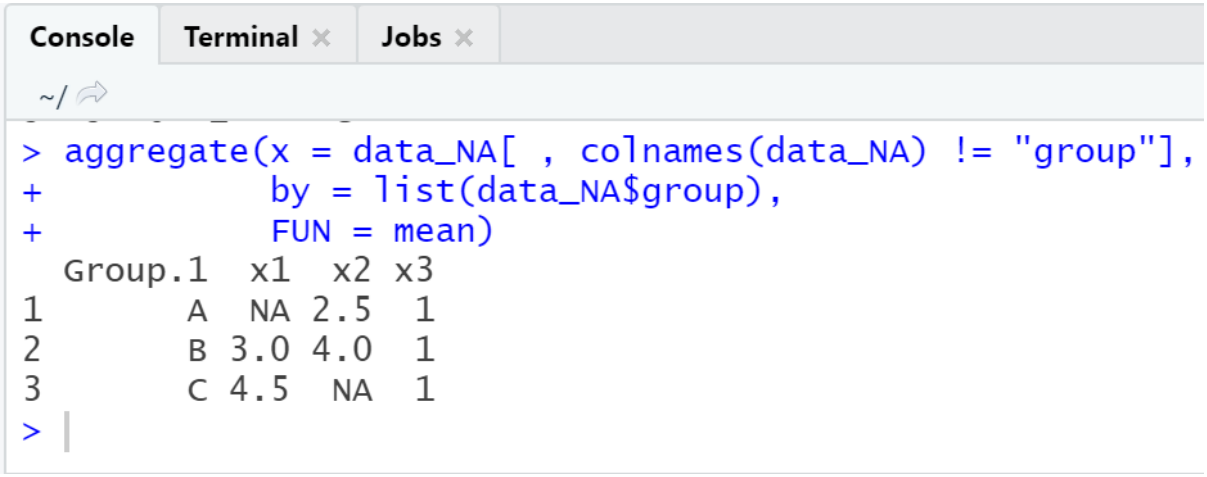
Alles wat we moesten veranderen was het FUN-argument binnen de aggregatiefunctie. De vorige uitvoer toont de telling door **de telling per groep** *(count by group)* van de voorbeeldgegevens.

In **voorbeeld 3** wordt de aggregatiefunctie toegepast op gegevens die NA waarden bevatten. Een typisch probleem bij het toepassen van de aggregatiefunctie is: ontbrekende waarden in het invoergegevensframe. *Voorbeeld 3* dus legt uit hoe om te gaan met NA-waarden met de aggregatiefunctie.

Laten we eerst enkele NA-waarden invoegen in onze voorbeeldgegevens. Dit gebeurd in *de regels 17 t/m 22* van de code. Toon daar na de nieuwe data. Dit gebeurd in regel 26. Het resultaat is dan zoals hiernaast te zien is:



De vorige uitvoer van de RStudio-console laat zien hoe de bijgewerkte gegevens eruit zien. Zoals je kunt zien, hebben sommige gegevenscellen de waarde NA gekregen.
Laten we proberen de aggregatiefunctie toe te passen zoals we eerder deden. Het resultaat van de aggregaat functie te gebruiken, *de regels 25 t/m 26* in de code, is dan als volgt:



Zoals je kunt zien, zijn sommige waarden in de uitvoer NA. Gelukkig kunnen we onze NA-waarden gewoon tijdelijk verwijderen met het argument **na.rm** binnen de aggregatiefunctie.
Het resultaat van de aggregatiefunctie met het argument **na.rm**, in *de regels 29 t/m 30* in de code, is dan als volgt.

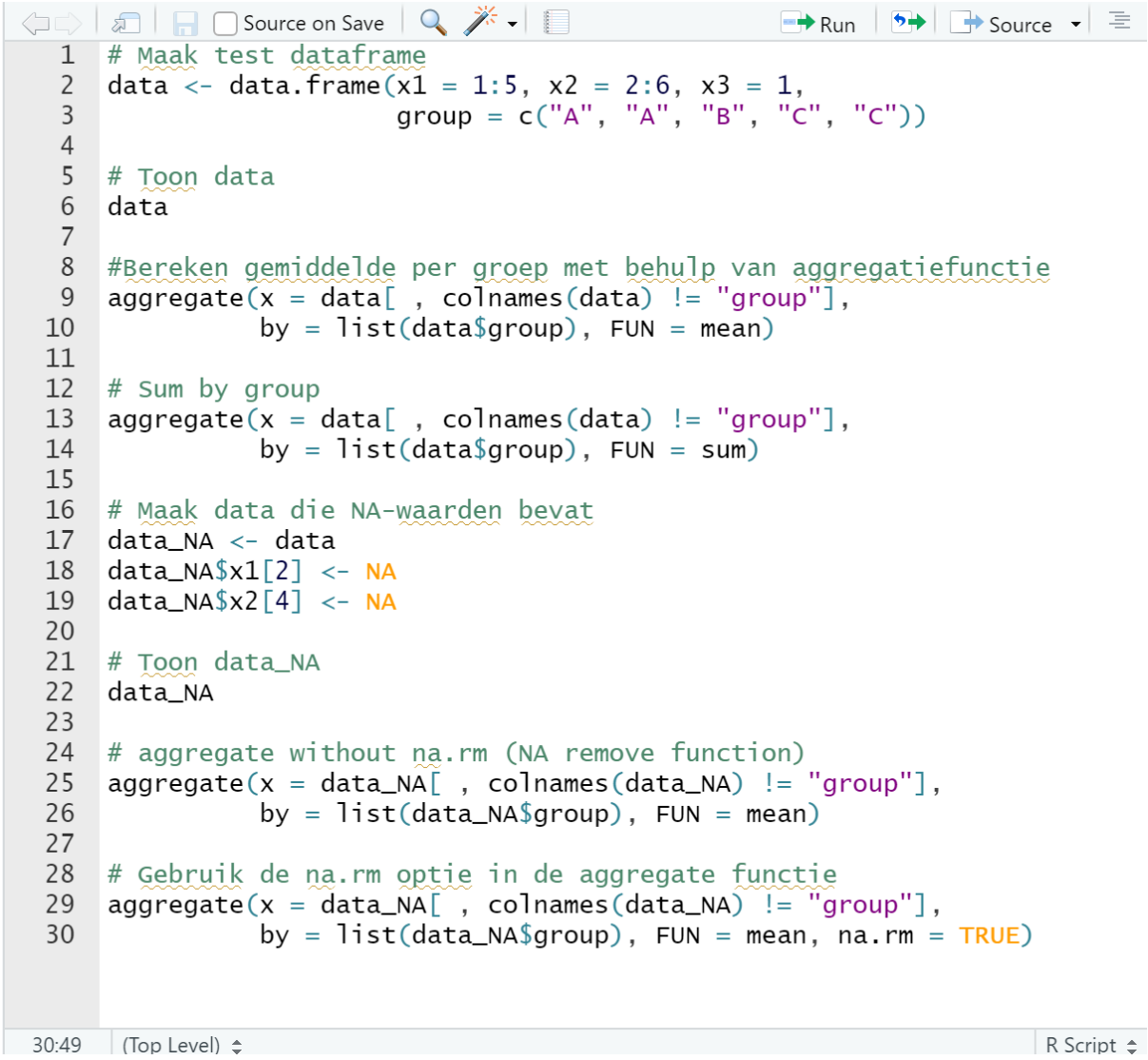
```
Console  Terminal  Jobs
~/
> aggregate(x = data_NA[ , colnames(data_NA) != "group"],
+           by = list(data_NA$group),
+           FUN = mean,
+           na.rm = TRUE)
  Group.1  x1  x2 x3
1       A 1.0 2.5  1
2       B 3.0 4.0  1
3       C 4.5 6.0  1
>
```

In het laatste **voorbeeld** kunnen we de formulemethode gebruiken om gegevens te aggregeren, met **~** kunt u aan de linkerkant de gegevens bepalen die u wilt aggregeren en aan de rechterkant de groep waarmee u wilt aggregeren.
Het moet duidelijk zijn om uw dataframe te selecteren waaruit de gegevens worden gebruikt en de functie die op de elementen wordt toegepast.

```
Console  Jobs
R 4.0.2 · ~/
> aggregate(x1~group, data = data, FUN = mean)
  group  x1
1     A 1.5
2     B 3.0
3     C 4.5
>
```

1. Open RStudio en klik vervolgens op **File**, **New File**, **R Script** of druk op de toetsen **Ctrl + Shift + N** om een nieuw Code Editor venster te openen.

2. Voer de onderstaande broncode in de Code Editor in. Laat de groene commentaarregels weg. Neem deze wel door want daar staat de uitleg over wat er gebeurd bij elke instructie.



3 Sla het **R Script** op met de naam **Agregate.R** en klik vervolgens op de knop **Source** of druk op **Ctrl + Shift + S** om de rekenkundige uitvoer te zien.