Seminar 2

Lise Rødland

March 15, 2021

I dag skal vi se på følgende:

3. Organisering av arbeidet
4. Pakker
5. Laste inn data (prosjekt og working directory)
6. Målenivå
7. Klasser og målenivå
8. Utforske data
9. Plotting

Organisering av arbeidet

Når vi jobber R så kan vi organisere arbeidet vårt på to måter: 1. Bruke prosjekter 2. Sette working directory

Working directory angir den mappen vi ønsker å hente og lagre filer til. Du kan tenke på det som den mappen du lagret prosjektfilen din i på første seminar. Når du åpner prosjektfilen din så husker R hvilken mappe dette er, men dersom du åpner et vanlig script må du fortelle R det.

Her kan man enten bruke `setwd("filbane")` eller så kan man trykke seg frem via verktøylinjen. Da velger du Session -> Set Working Directory -> Choose Directory og klikker deg frem til mappen din.

Her er et eksempel på bruk av `setwd()`:

```
setwd("~/Dokumenter/STV1020")
```

I dette seminaret skal vi bruke et datasett fra European Social Survey Round 9 (2018), og dettedatasettet inneholder svarene fra norske respondenter. Filen heter "ESS9NO.dta" og ligger i Canvas.

Pass på at du har lagret datasettet vi skal bruke i dag i samme mappe som den du har satt som working directory.

Overskrifter og tekst

Hvordan man organiserer et R-script kommer an på hva man selv synes er mest oversiktlig, men det er viktig at man klarer å holde oversikt over hva man har kodet og forstår hva man har gjort når man kommer tilbake til et script.

Det er lurt å lage overskrifter for å huske hva du tenker at koden din skal gjøre. Dersom du velger overskrift-formatet "# Overskrift —-" så vil R automatisk gi deg muligheten til å gjemme koden under overskriften.

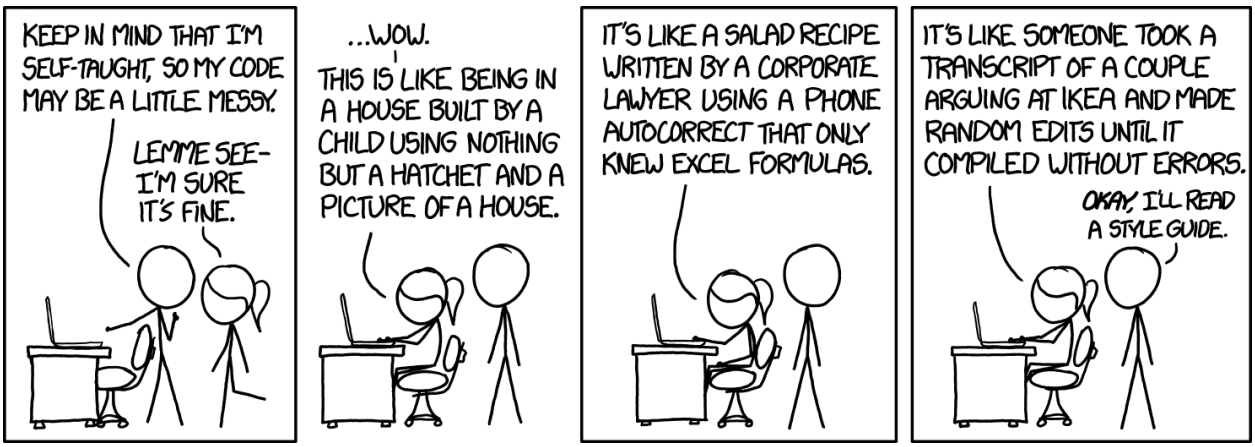


Figure 1: Ryddig kode gjør det enklere for deg selv og andre når du står fast. Tegneserie fra XKCD: https://xkcd.com/1513/

Om du trykker på "Show document outline" i menylinjen til høyre over scriptet ditt kan du også få opp en innholdsfortegnelse basert på overskriftene dine.

Det kan også være lurt å inkludere kommentarer i scriptet ditt som forklarer hva du gjør. Tekst som ikke skal leses av R skriver man etter emneknagg (#). Vi glemmer fort så det er en god idé å tenke at du skal kunne se tilbake på dette scriptet om et år igjen og skjønne hva du har gjort.

Til sist så er det lurt å ikke skrive for mange tegn før du bytter linje. The tidyverse style guide anbefaler at en begrenser antall tegn til 80 per linje. R teller for hvor mange tegn du har per linje til venstre nedenfor scriptet ditt. Tidyverse sin stilguide inneholder også flere tips til hvordan man kan skrive lettleselig kode.

Pakker

R-pakker er utvidelser til programmeringsspråket R. De inneholder kode, data, og dokumentasjon som gir oss tilgang til funksjoner som løser ulike problemer og gjør koding enklere. Første gang man skal bruke en pakke må man installere den. Etter at vi har installert pakken så må vi "hente den fra biblioteket" for å fortelle R at vi ønsker å bruke pakken. Dette må vi gjøre hver gang vi åpner R på nytt og ønsker å bruke pakken.

Den første pakken vi skal installere er Tidyverse. Tidyverse er et sett med pakker som gjør databehandling mye, mye enklere. Føst installerer vi pakken. Om dere har gjort dette på forhånd trenger dere ikke gjøre dette på nytt. Å installere gjør vi kun en gang. Vi installerer pakker ved hjelp av funksjonen `install.packages("pakkenavn")`:

```
install.packages("tidyverse")
```

Det neste vi gjør er å laste inn pakken ved hjelp av `library()`:

```
library(tidyverse)
```

Merk at pakkenavnet ikke står i hermetegn når vi bruker `library()`. Hermetegn rundt pakkenavnet er bare nødvendig når vi bruker `install.packages()`.

Laste inn data

Dersom dere skal gjøre statistisk analyse, er som regel den første seksjonen import og forberedelse av data. En styrke ved R, er at det er mulig å importere mange ulike filtyper, både fra en mappe på pcen din og fra en url på internett. Det er også mulig å ha flere datasett oppe i R samtidig. Jeg går gjennom import av filer fra excel, stata, spss og R, men vit at det finnes mange andre muligheter. Hvis man lurer på hvordan man skal laste inn en bestemt filtype og har glemt hvordan man gjør det så er dette veldig lett å finne på internett

Når du skal laste inn eller lagre noe lokalt på pc-en så vil R til enhver tid forvente at filnavn du refererer til befinner seg i working directory. Som vi husker så er working directory en mappe på pcen din; enten den du har lagret prosjektet ditt i eller den du har satt som working directory ved hjelp av `setwd()`. For å sjekke hva nåværende working directory er, og hvilke filer som finnes i den mappen, kan du skrive følgende kode (jeg har gjemt egen output):

```
getwd()
```

```
## [1] "C:/Users/liserod/OneDrive - Universitetet i Oslo/Projects/Undervisning/STV1020/doc/seminar2"
```

```
list.files()
```

```
## [1] "seminar2.md"    "seminar2.pdf"   "seminar2.Rmd"   "seminar2_files"
```

Datasett kommer i mange ulike filformater. Noen vanlige formater er csv, dta (Stata-datasett), sav (SPSS-datasett) og Rdata. Hvilket format dataene dine har bestemmer hvilken funksjon du må bruke for å laste inn datasettet. For det meste så følger funksjonene dette formatet:

```
# Laster inn og lagrer datasettet som et objekt:
datasett <- read_filtype("filbane/filnavn.filtype")
```

For eksempel så er datasettet vi skal bruke i dag en dta-fil. Pakken `haven` inneholder funksjoner for å lese dta-filer og sav-filer. Det første vi gjør er derfor å installere og laste inn `haven`.

```
install.packages("haven")
```

```
library(haven)

ess <- read_dta("../../data/ESS9NO.dta")
```

Her er eksempler på noen andre funksjoner for å laste inn data:

```
# For .csv:
read_csv("data/filnavn.csv")

# For filer i R-format:
load("data/filnavn.Rdata")
```

For å laste inn en excel-filer bruker vi pakken `readxl`:

```
install.packages("readxl")

library(readxl)

df <- read_excel("data/filnavn.xlsx")
```

Organisering av data

Når man bruker større datasett som ESS, så inneholder datasettet ofte mange flere variabler enn de vi ønsker å bruke i våre analyser og variablene har navn som kan være vanskelig å huske, f.eks. nwspol. For å finne ut hvilken informasjon variablene inneholder så kan vi slå opp i kodeboken. Kodeboken til ESS finner dere her. ESS inneholder også mange landspesisifkke variabler og kodeboken er derfor veldig lang.

Når vi skal jobbe videre med data så kan det være lurt å fjerne de variablene vi ikke skal bruke og gi variablene navn som er lette for oss å forstå og huske. For å gjøre dette skal vi benytte oss av funksjoner i `tidyverse`. Først bruker vi `select()` til å velge de variablene vi vil beholde og så bruker vi `rename` til å endre navnene. Vi bruker en pipe `%>%` mellom funksjonene, som tar outputen til et utsagn og gjør det til inputen til det neste utsagnet. Pipen kan sees på som ordet "så". Rename-funksjonen lar oss forandre navnet til variabler og bruker syntaksen nytt_navn = gammelt_navn.

```
ess_subset <- ess %>%
  select(nwspol, polintr, vote, yrbrn) %>%
  rename(
        news = nwspol,
        interest = polintr,
        age = yrbrn
  )
```

Dersom du har sjekket kodeboken på forhånd så kan du også endre variablenavnene når du bruker `select()`:

```
ess_subset <- ess %>%
  select(
    vote,
    news = nwspol,
    interest = polintr,
    year_born = yrbrn
    )
```

Vi kan regne oss frem til alder ved å bruke variabelen `year_born`og informasjonen om at undersøkelsen ble gjennomført i 2018:

```
ess_subset$age <- 2018 - ess_subset$year_born
```

Klasser og målenivå

I forelesning gikk dere gjennom tre ulike målenivåer; kategorisk, ordinalt og kontinuerlig. Disse sammen faller til en viss grad med noen av klassene i R. Men det er viktig å huske at R ikke vet hvilken klasse en variabel har så dere kan ikke nødvendigvis bruke dette til å sjekke variabelens målenivå. Det viktige er at dere gir R riktig informasjon om hvilket målenivå en variabel har.

Kategorisk

Når variabler er kategoriske så kan egenskapen deles i to eller flere gjensidig utelukkende kategorier. I ESS datasettet vårt er variabelen "vote" kategorisk; man har enten stemt, ikke stemt, eller så er man ikke berettiget til å stemme. Dette kan vi se i utklippet fra kodeboken.

vote - Voted last national election

Type	Code
vote	Some people don't vote nowadays for one reason or another. Did you vote in the last [country] national election in [month/year]?
Location	B13
Question	Some people don't vote nowadays for one reason or another. Did you vote in the last [country] national election in [month/year]?

1	Yes
2	No
3	Not eligible to vote

Figure 2: Utdrag fra ESS sin kodebok for variabelen vote

Vi kan sjekke hvilken klasse variabelen har ved hjelp av `class()`

```
class(ess_subset$vote)
```

```
## [1] "haven_labelled" "vctrs_vctr"     "double"
```

Her får vi opp flere klasser; "haven_labelled", "vctrs_vctr" og "double". Dette skyldes blant annet at datasett i dta-format ofte kommer med labels. I mange datasett får kategorisk variabler ofte tall istedenfor kategorinavn som verdier. Labels inneholder informasjon om hvilke kategorier disse tallene representerer. Denne informasjonen finner vi også i kodeboken. Dette gjør at kategoriske variabler ofte fremstår som at de har et høyere målenivå enn de faktisk har i R. For å få frem poenget så kan vi spørre R om variabelen `vote` er registrert som numerisk:

```
is.numeric(ess_subset$vote)
```

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

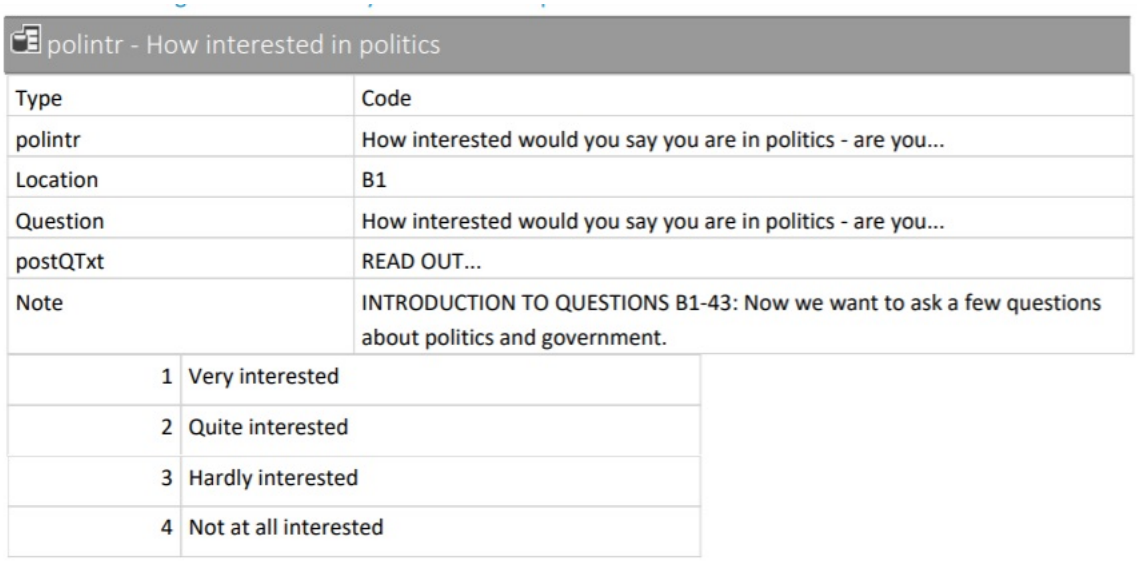
Før vi bruker denne variabelen i en analyse bør vi endre klassen ved å bruke for eksempel `as.factor()`:

```
ess_subset$vote2 <- as.factor(ess_subset$vote)

class(ess_subset$vote2)
```

```
## [1] "factor"
```

Ordinalnivå



polintr - How interested in politics

Type	Code
polintr	How interested would you say you are in politics - are you...
Location	B1
Question	How interested would you say you are in politics - are you...
postQTxt	READ OUT...
Note	INTRODUCTION TO QUESTIONS B1-43: Now we want to ask a few questions about politics and government.

1	Very interested
2	Quite interested
3	Hardly interested
4	Not at all interested

Figure 3: Utdrag fra ESS sin kodebok for variabelen interest (opprinnelig navn polintr)

Når variabler er på ordinalnivå kan de deles i to eller flere kategorier som kan rangeres, men vi kan ikke si noe om avstanden mellom verdiene og en enhets økning har ikke samme betydning. I ESS datasettet vår så er variabelen "interest" et eksempel på en variabel på ordinalnivå; i utdraget fra kodeboken ser vi at man kan være ikke interessert, lite interessert, ganske interessert, eller veldig interessert i politikk.

```
class(ess_subset$interest)
```

```
## [1] "haven_labelled" "vctrs_vctr"     "double"
```

```
is.numeric(ess_subset$interest)
```

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

Som vi ser er også denne variabelen registrert som numerisk av R. Denne bør vi omkode til en faktor. Faktorer bevarer informasjon om rangering. I og med at kategoriene i dette tilfelle har verdiene 1 til 4 i stigende rekkefølge så trenger vi ikke å angi en egen rangering når vi lager faktoren:

```
ess_subset$interest2 <- as.factor(ess_subset$interest)

class(ess_subset$interest2)
```

```
## [1] "factor"
```

```
levels(ess_subset$interest2)
```

```
## [1] "1" "2" "3" "4"
```

Kontinuerlig

Kontinuerlige variabler kan ragneres, har samme avstand mellom alle verdier og en enhets økning betyr alltid det samme. Her er det altså snakk om variabler med faktiske tallverdier. I ESS datasettet vårt så

nwspol - News about politics and current affairs, watching, reading or listening, in minutes	
Type	Numeric (Integer)
nwspol	On a typical day, about how much time do you spend watching, reading or listening to news about politics and current affairs?
Location	A1
Question	On a typical day, about how much time do you spend watching, reading or listening to news about politics and current affairs?
postQTxt	Please give your answer in hours and minutes. INTERVIEWER: If no time spent, enter 00 00. TYPE IN DURATION

Figure 4: Utdrag fra ESS sin kodebok for variabelen news (opprinnelig navn nwspol)

er variabelen "news" kontinuerlig. Som vi kan se i utdraget fra kodeboken så måler variabelen hvor mange minutter man bruker på nyheter hver dag. Det er et minutts avstand mellom hver verdi, og en økning på en enhet vil alltid bety en økning på et minutt.

Vi kan sjekke klassen her også:

```
class(ess_subset$news)
```

```
## [1] "haven_labelled" "vctrs_vctr"     "double"
```

```
is.numeric(ess_subset$news)
```

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

Denne variabelen er numerisk og skal være det så her gjør vi ingen endringer.

Utforske data

Det er mange ulike måter å utforske datasett og variabler på. Vi skal se på funksjonene: `summary()`, `str()`, `head()` og `tail()`.

For å få et deskriptivt sammendrag av et objekt kan vi bruke `summary()` eller str().

```
summary(ess_subset)
```

```
##       vote            news           interest     year_born         age          vote2      interest2
##  Min.   :1.000   Min.   :   0.0   Min.   :1.0   Min.   :1928   Min.   :15.00   1   :1156   1   :184
##  1st Qu.:1.000   1st Qu.:  30.0   1st Qu.:2.0   1st Qu.:1957   1st Qu.:32.00   2   : 124   2   :564
##  Median :1.000   Median :  60.0   Median :2.0   Median :1971   Median :47.00   3   : 125   3   :568
##  Mean   :1.266   Mean   : 104.1   Mean   :2.4   Mean   :1971   Mean   :46.54   NA's:   1   4   : 89
##  3rd Qu.:1.000   3rd Qu.: 120.0   3rd Qu.:3.0   3rd Qu.:1986   3rd Qu.:61.00               NA's:  1
##  Max.   :3.000   Max.   :1109.0   Max.   :4.0   Max.   :2003   Max.   :90.00
##  NA's   :1       NA's   :34       NA's   :1     NA's   :32     NA's   :32
```

```
str(ess_subset)
```

```
## tibble [1,406 x 7] (S3: tbl_df/tbl/data.frame)
##  $ vote     : dbl+lbl [1:1406] 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 3, 1, 1, 1,
##    ..@ label       : chr "Voted last national election"
##    ..@ format.stata: chr "%20.0g"
##    ..@ labels      : Named num [1:6] 1 2 3 NA NA NA
##    .. ..- attr(*, "names")= chr [1:6] "Yes" "No" "Not eligible to vote" "Refusal" ...
##  $ news     : dbl+lbl [1:1406]    60,    60,   540,    30,    60,   120,    60,    90,   120,    60,
##    ..@ label       : chr "News about politics and current affairs, watching, reading or listening, i
##    ..@ format.stata: chr "%12.0g"
##    ..@ labels      : Named num [1:3] NA NA NA
##    .. ..- attr(*, "names")= chr [1:3] "Refusal" "Don't know" "No answer"
##  $ interest : dbl+lbl [1:1406] 3, 2, 2, 2, 2, 1, 2, 3, 2, 2, 1, 2, 2, 3, 1, 2, 2, 3, 2, 2, 3, 2, 2, 1
##    ..@ label       : chr "How interested in politics"
##    ..@ format.stata: chr "%21.0g"
##    ..@ labels      : Named num [1:7] 1 2 3 4 NA NA NA
##    .. ..- attr(*, "names")= chr [1:7] "Very interested" "Quite interested" "Hardly interested" "Not a
##  $ year_born: dbl+lbl [1:1406] 1961, 1960, 1956, 1967, 1972, 1964, 1959, 2000, 1950, 1975, 1934, 1965
##    ..@ label       : chr "Year of birth"
##    ..@ format.stata: chr "%12.0g"
##    ..@ labels      : Named num [1:3] NA NA NA
##    .. ..- attr(*, "names")= chr [1:3] "Refusal" "Don't know" "No answer"
##  $ age      : num [1:1406] 57 58 62 51 46 54 59 18 68 43 ...
##  $ vote2    : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 3 1 1 ...
##  $ interest2: Factor w/ 4 levels "1","2","3","4": 3 2 2 2 2 1 2 3 2 2 ...
```

Hvis man vil se de første eller siste radene i et datasett, kan man bruke henholdsvis head- og tail-funksjonene. Man kan også velge for eksempel å bare se på de første eller siste verdiene til en bestemt variabel.

```
head(ess_subset)
```

```
## # A tibble: 6 x 7
##        vote      news              interest year_born   age vote2 interest2
##   <dbl+lbl> <dbl+lbl>             <dbl+lbl> <dbl+lbl> <dbl> <fct> <fct>
## 1   1 [Yes]        60 3 [Hardly interested]      1961    57 1     3
## 2   1 [Yes]        60 2 [Quite interested]       1960    58 1     2
## 3   1 [Yes]       540 2 [Quite interested]       1956    62 1     2
## 4   1 [Yes]        30 2 [Quite interested]       1967    51 1     2
## 5   1 [Yes]        60 2 [Quite interested]       1972    46 1     2
## 6   1 [Yes]       120 1 [Very interested]        1964    54 1     1
```

```
tail(ess_subset)
```

```
## # A tibble: 6 x 7
##                       vote              news              interest year_born   age vote2 interest2
##                  <dbl+lbl>          <dbl+lbl>             <dbl+lbl> <dbl+lbl> <dbl> <fct> <fct>
## 1 1 [Yes]                    90              1 [Very interested]        1955    63 1     1
## 2 3 [Not eligible to vote]   20              2 [Quite interested]       2003    15 3     2
## 3 1 [Yes]                    30              3 [Hardly interested]      1994    24 1     3
## 4 1 [Yes]                    60              2 [Quite interested]       1984    34 1     2
## 5 2 [No]                  NA(c) [Don't know] 3 [Hardly interested]      1974    44 2     3
## 6 1 [Yes]                    30              3 [Hardly interested]      1988    30 1     3
```

Alle disse kan også bruker på enkeltvariabler.

Deskriptiv statistikk

Som dere husker fra forelesning og fra kapittel seks i Kellsted og Whitten så er det variabelens målenivå som avgjør hvilken deskriptiv statistikk som er fornuftig.

Kategoriske variabler

R har ingen innebygd funksjon for å finne modusverdien. Ved å søke på internett så finner du fort mange ulike funksjoner du kan bruke, men for å gjøre det enkelt bruker vi bare `table()`. Funksjonen `table()` gir oss en frekvenstabell, mens `prop.table` gjør om frekvenstabellen til andeler. ESS datasettet mangler data for noen observasjoner. ved å ta med `useNA = "always"` så får vi ogås denne informasjonen i tabellen:

```
table(ess_subset$vote, useNA = "always")
```

```
##
##    1    2    3 <NA>
## 1156  124  125    1
```

```
prop.table(table(ess_subset$vote))
```

```
##
##          1          2          3
## 0.82277580 0.08825623 0.08896797
```

```
prop.table(table(ess_subset$vote, useNA = "always"))
```

```
##
##            1            2            3         <NA>
## 0.8221906117 0.0881934566 0.0889046942 0.0007112376
```

Kontinuerlige variabler

```
# Finner minimumsverdi (det laveste antall minutter brukt på nyheter)
min(ess_subset$news, na.rm = TRUE) # na.rm = TRUE sier at missing skal droppes i beregningen
```

```
## <labelled<double>[1]>: News about politics and current affairs, watching, reading or listening, in mi
## [1] 0
##
## Labels:
##  value      label
##  NA(b)    Refusal
##  NA(c) Don't know
##  NA(d)  No answer
```

```r
# Finner maksimumsveriden (den høyeste antall minutter brukt på nyheter)
max(ess_subset$news, na.rm = TRUE)
```

```
## <labelled<double>[1]>: News about politics and current affairs, watching, reading or listening, in mi
## [1] 1109
##
## Labels:
##  value      label
##  NA(b)    Refusal
##  NA(c) Don't know
##  NA(d)  No answer
```

```r
# Finner gjennomsnittlig antall minutter
mean(ess_subset$news, na.rm = TRUE)
```

```
## [1] 104.1006
```

```r
# Finner median
median(ess_subset$news, na.rm = TRUE)
```

```
## [1] 60
```

```r
# Finner standardavviket
sd(ess_subset$news, na.rm = TRUE)
```

```
## [1] 155.5571
```

```r
# Finner varians
var(ess_subset$news, na.rm = TRUE)
```

```
## [1] 24198.01
```

```r
# Finner kvantilverdiene
quantile(ess_subset$news, na.rm = TRUE)
```

```
##   0%  25%  50%  75% 100%
##    0   30   60  120 1109
```

```r
# Finner forskjellig deskriptiv statistikk for en variabel
summary(ess_subset$news)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     0.0    30.0    60.0   104.1   120.0  1109.0      34
```

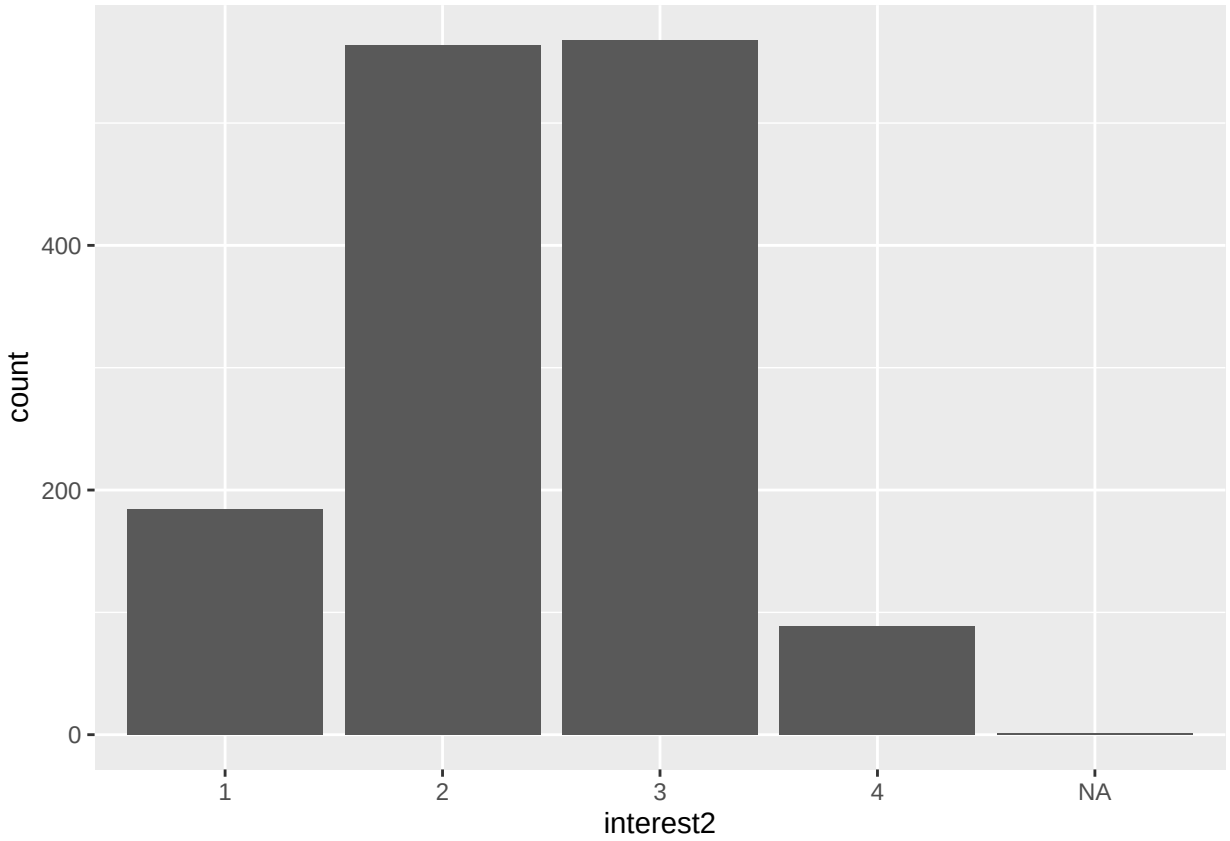
Plotting

Vi skal kort introdusere hvordan man kan visualisere data i dette seminaret, og så vil dere få en mer grundig gjennomgang neste seminar. Det er gøy å kunne visualisere dataene våre, både for vår egen del, men også for de som skal lese oppgavene våre. For å få fine grafer kan man bruke funksjonen `ggplot()`.

Kategoriske variabeler

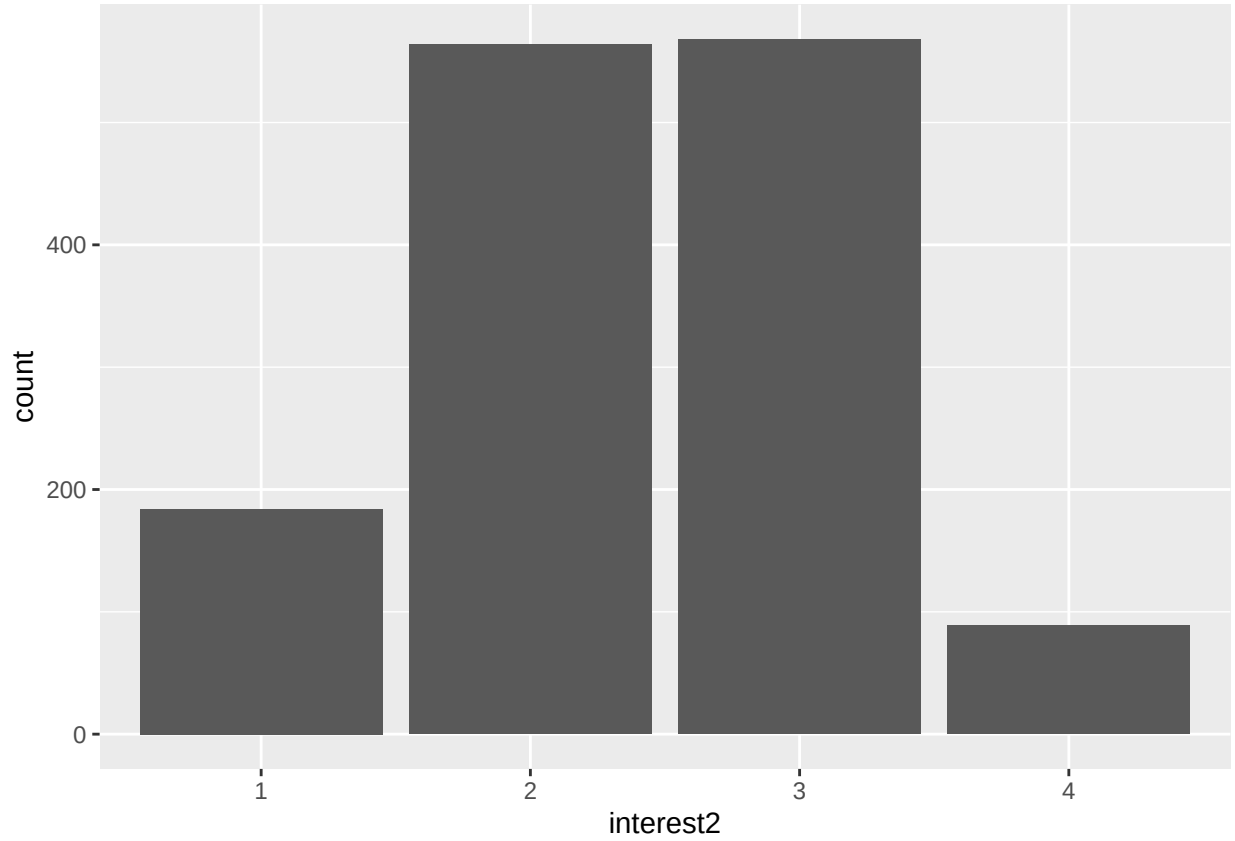
Søylediagram med en variabel Hvordan kan vi visualisere hvordan fordelingen av politisk interesse er? Her kan vi bruke `geom_bar` til å lage et søylediagram (bar chart). Et søylediagram viser antall observasjoner av hver verdi.

```
ggplot(data = ess_subset, aes(x = interest2)) +
  geom_bar()
```



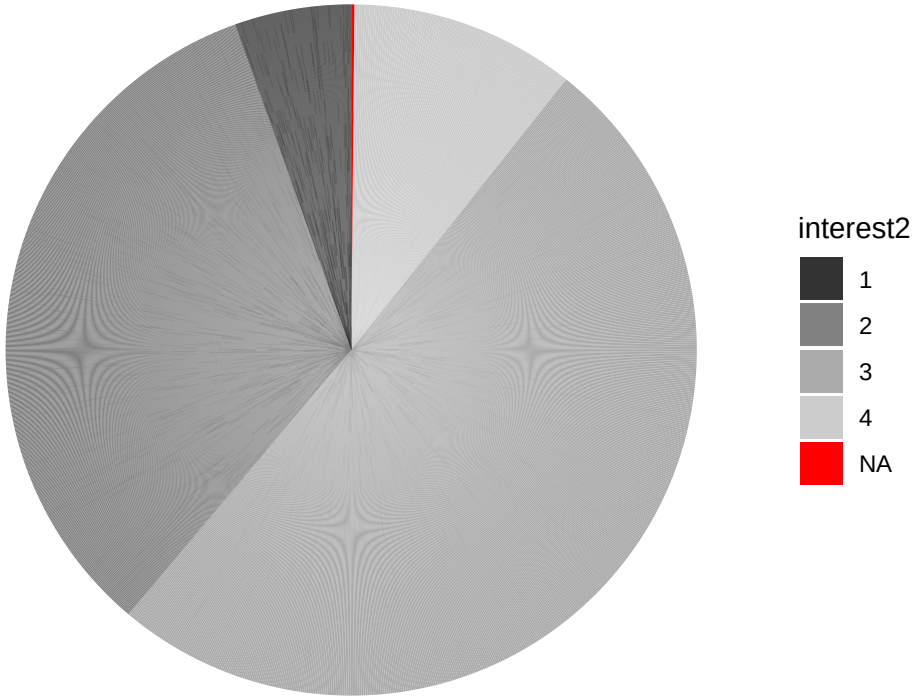
Dersom vi ikke ønsker å gi missingverdiene (NA) en egen søyle så kan vi bruke `filter()` til å fjerne disse:

```
ggplot(data = ess_subset %>%
         filter(!is.na(interest2)),
       aes(x = interest2)) +
  geom_bar()
```



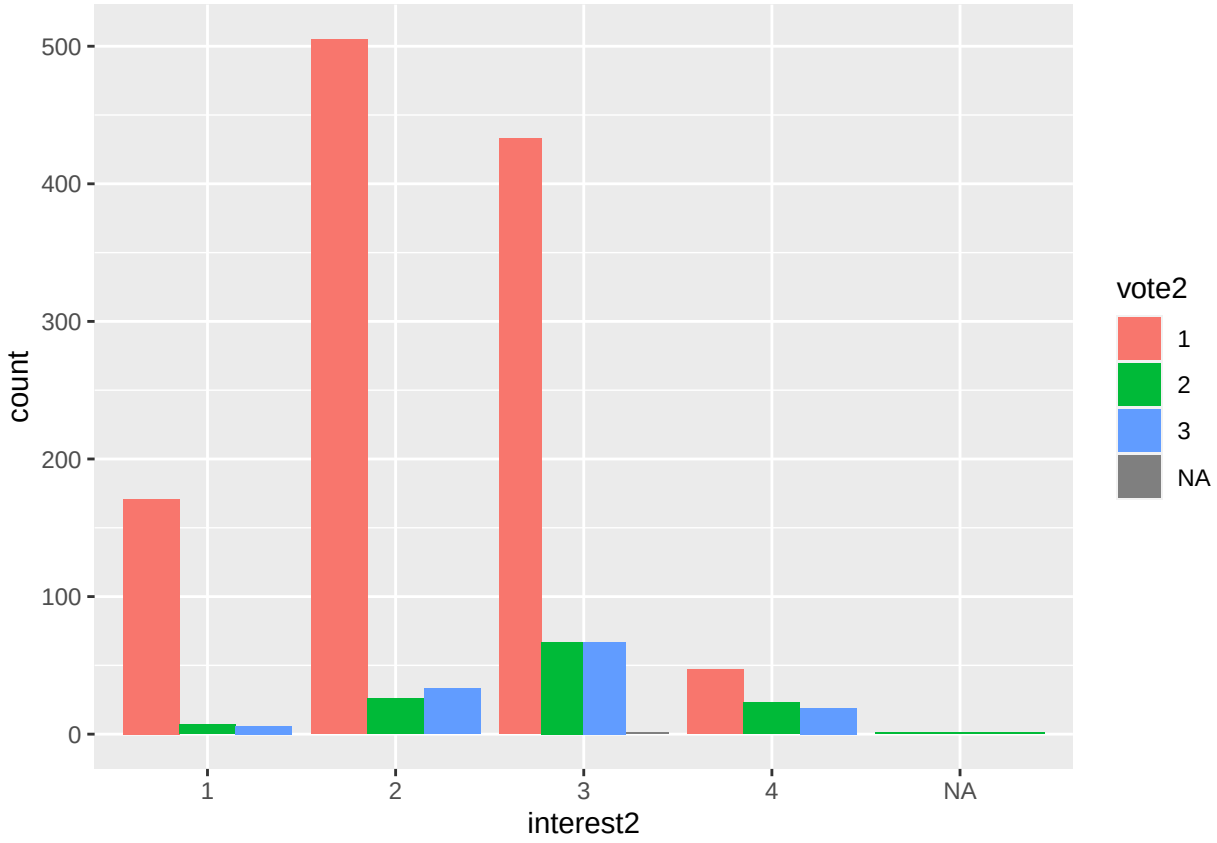
Et alternativ til søylediagram er kakediagram (pie chart):

```
ggplot(ess_subset, aes(x = "", y = interest2, fill = interest2)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  theme_void() +
  scale_fill_grey()
```



Søylediagram med to variabler Hvor mange innenfor hvert nivå av politisk interesse stemte? Vi kan bruke `geom_bar()` igjen, men vi sier at vi også vil se fordelingen av hvordan respondentene stemte innenfor hvert nivå av politisk interesse med `(aes(fill = vote2))`. Så sier vi at vi vil at det skal være en søyle for de ulike alternativene for vote med `position = "dodge"`.

```
ggplot(data = ess_subset,
       aes(x = interest2)) +
  geom_bar(aes(fill=vote2),
           position = "dodge")
```



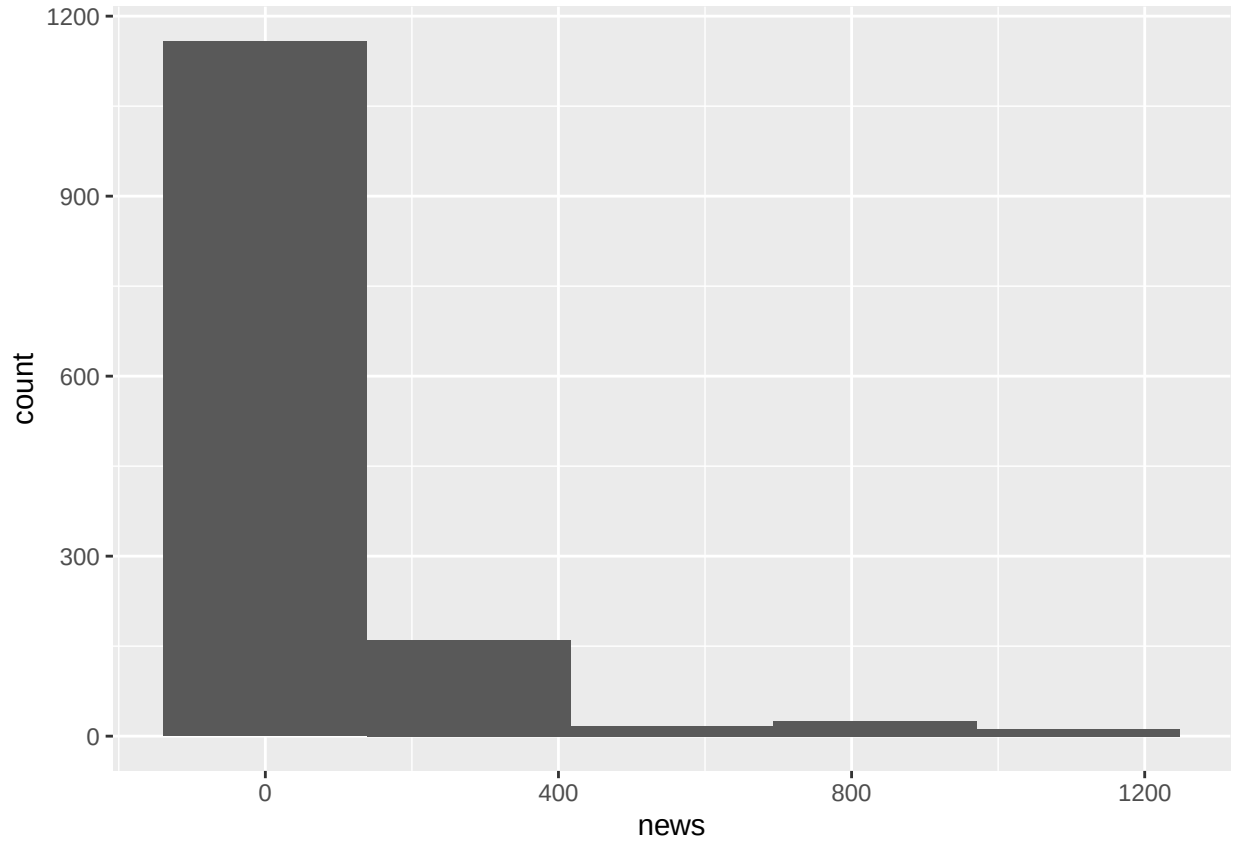
Kontinuerlige variabler

Histogram Hvordan fordeler respondentenes alder og tiden de bruker på nyheter seg? Disse variablene er kontinuerlige, så vi kan bruke `geom_histogram` for å lage et histogram. Her gjør jeg det med variabelen news.

```
ggplot(data = ess_subset, aes(x = news)) +
      geom_histogram(bins = 5)
```

```
## Don't know how to automatically pick scale for object of type haven_labelled/vctrs_vctr/double. Defau
```

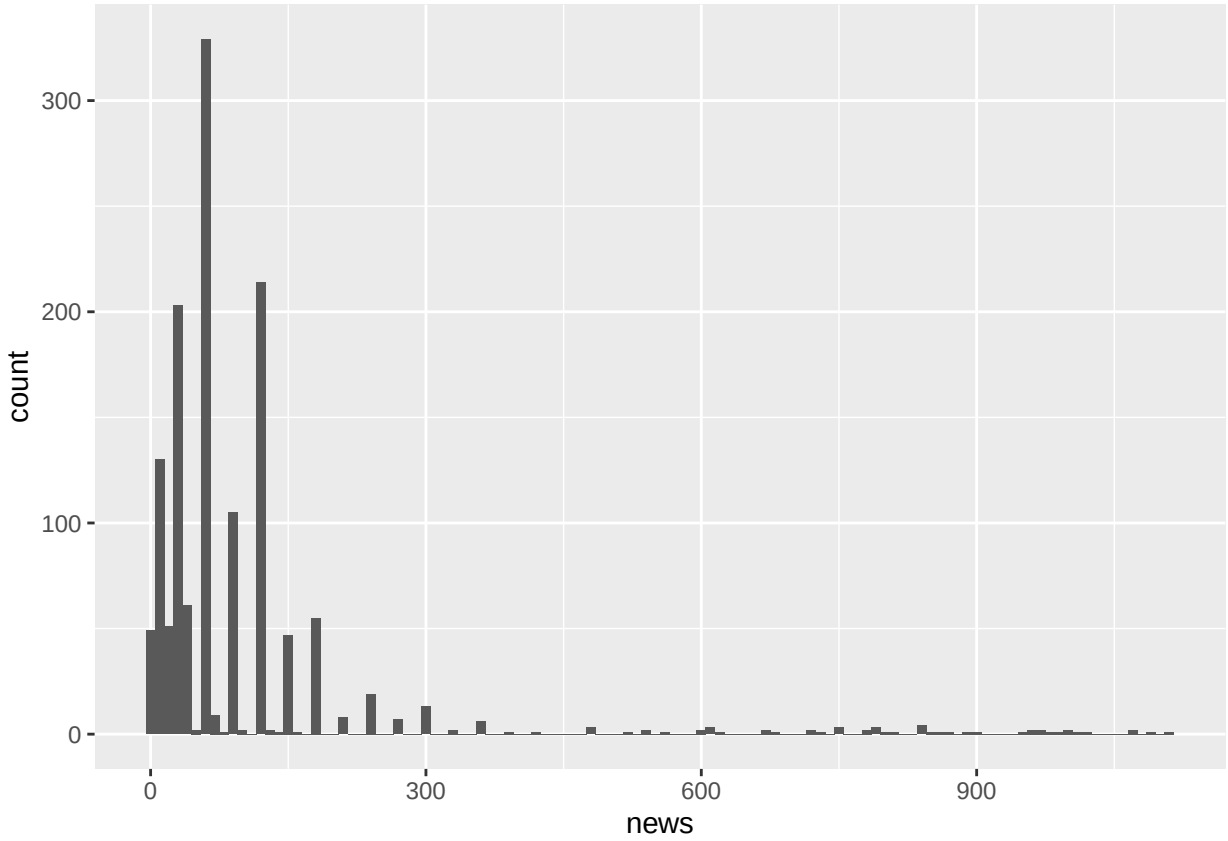
```
## Warning: Removed 34 rows containing non-finite values (stat_bin).
```




```
ggplot(data = ess_subset, aes(x = news)) +
  geom_histogram(binwidth = 10)
```

```
## Don't know how to automatically pick scale for object of type haven_labelled/vctrs_vctr/double. Defau
```

```
## Warning: Removed 34 rows containing non-finite values (stat_bin).
```

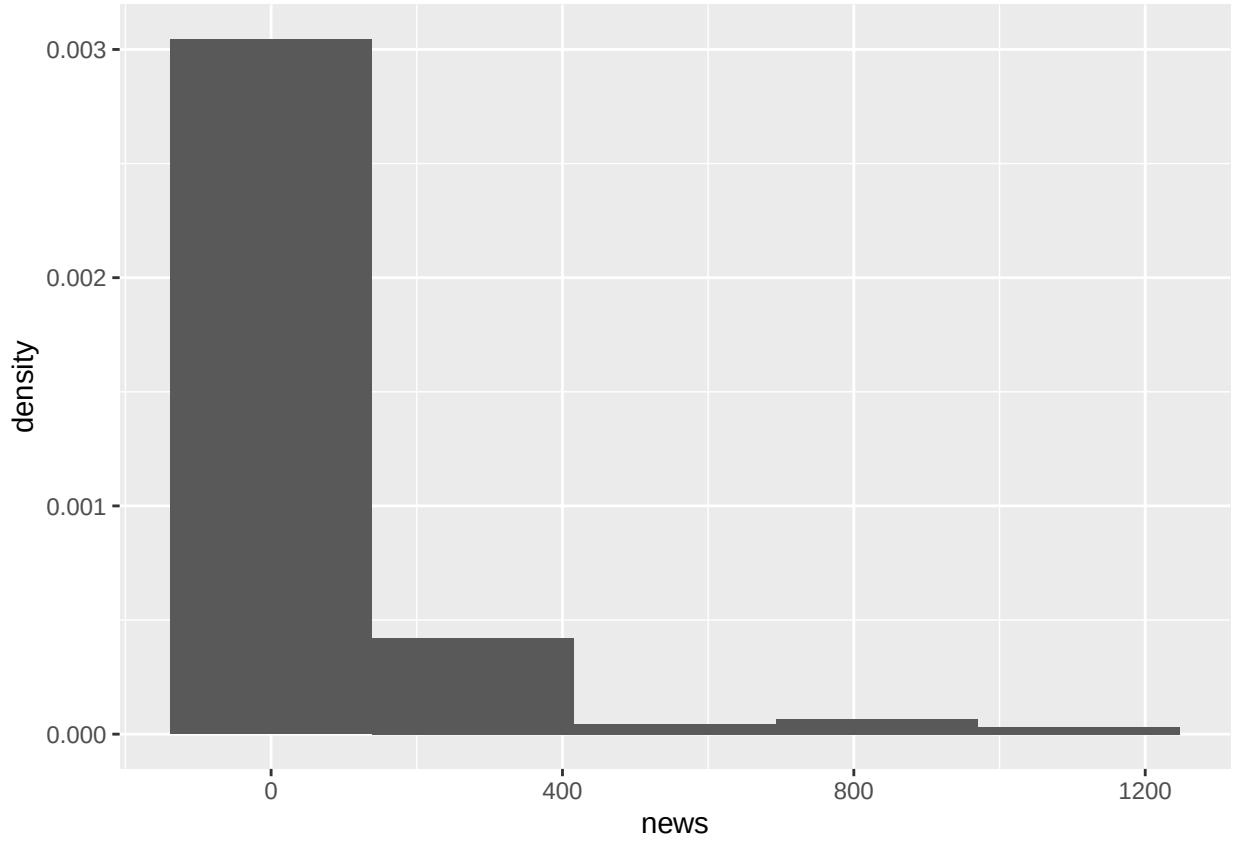


Et histogram viser hvor mange enheter det er i hver kategori. Vi kan enten spesifisere hvor mange søyler vi vil ha (bins) eller hvor stor hver søyle skal være (bindwidth). Vi kan også velge å plotte density fremfor count. Da får vi histogrammer tilsvarende figur 6.5 i Kellsted og Whitten:

```
ggplot(data = ess_subset, aes(x = news, y = ..density..)) +
      geom_histogram(bins = 5)
```

```
## Don't know how to automatically pick scale for object of type haven_labelled/vctrs_vctr/double. Defau
```

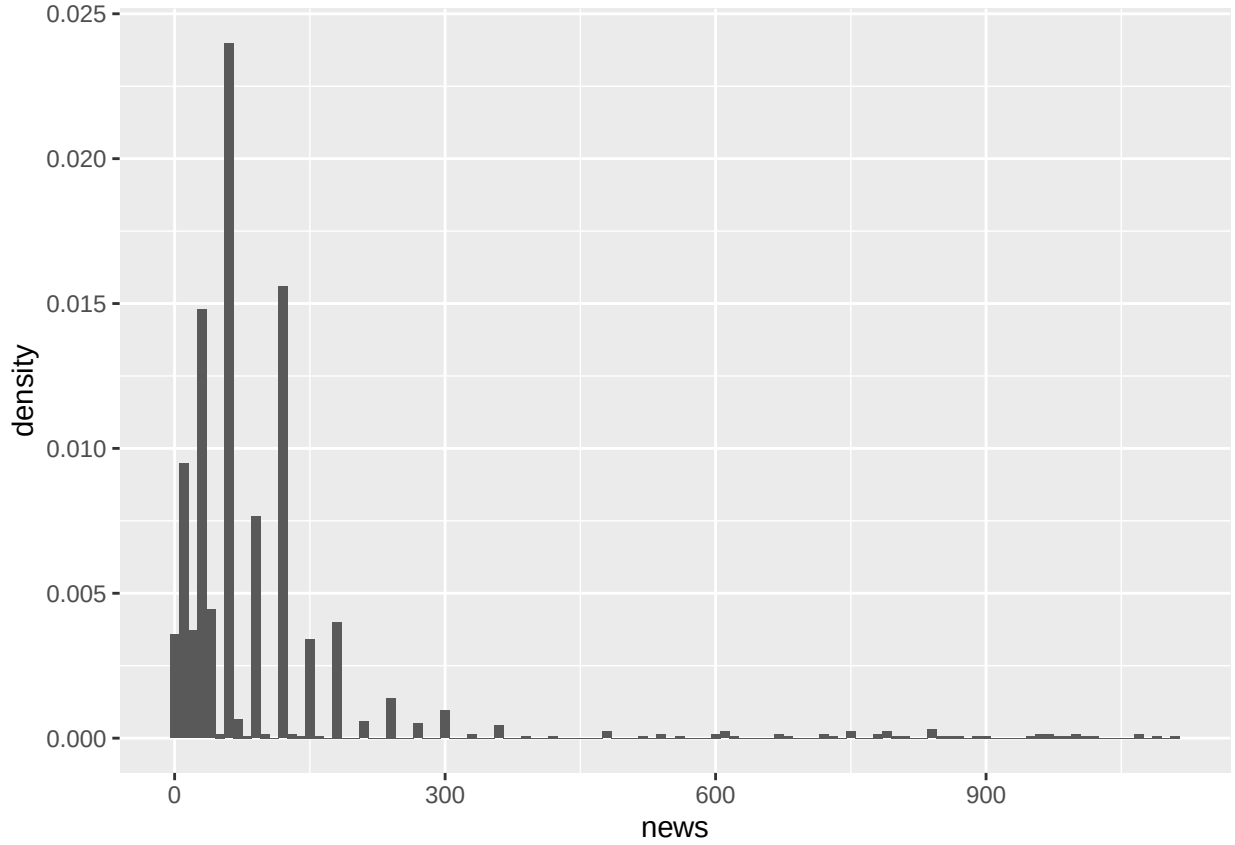
```
## Warning: Removed 34 rows containing non-finite values (stat_bin).
```




```
ggplot(data = ess_subset, aes(x = news, y = ..density..)) +
  geom_histogram(binwidth = 10)
```

```
## Don't know how to automatically pick scale for object of type haven_labelled/vctrs_vctr/double. Defau
```

```
## Warning: Removed 34 rows containing non-finite values (stat_bin).
```

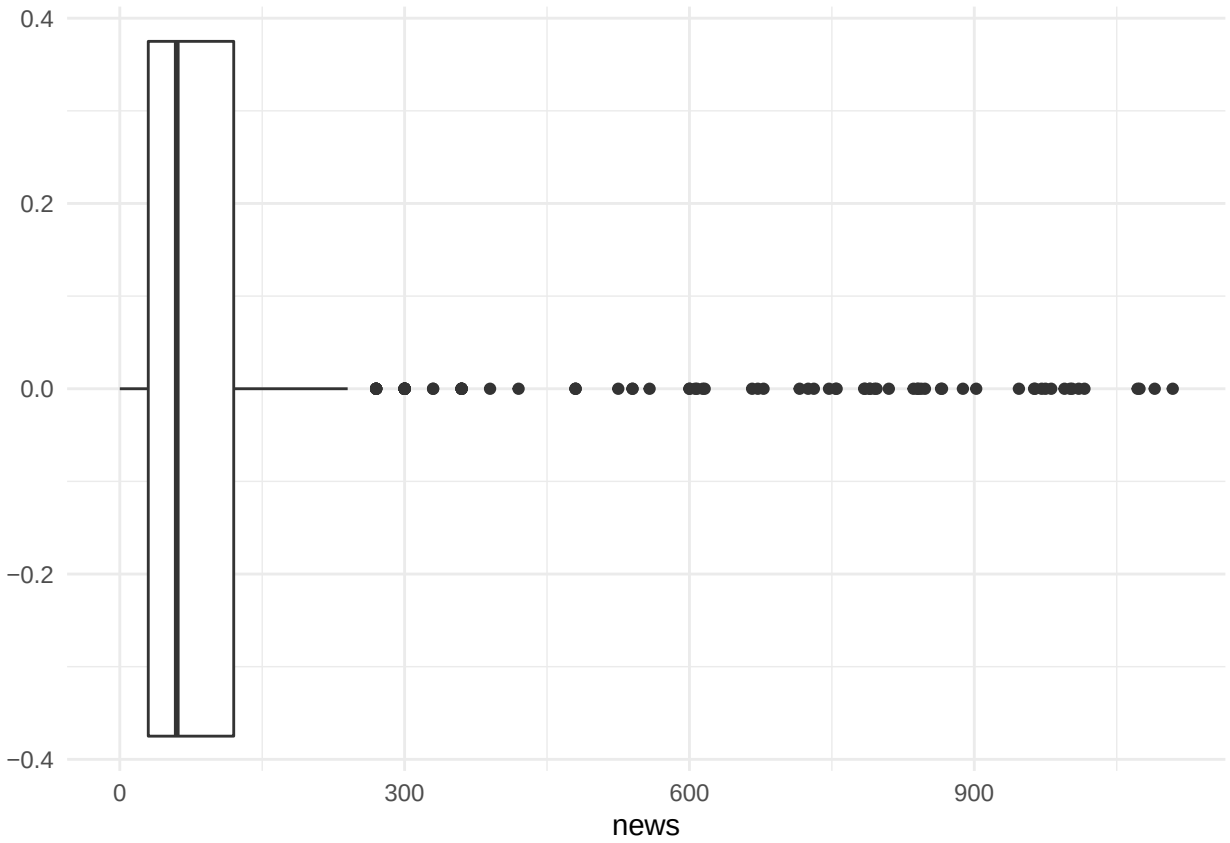


Boksplott Hvordan fordeler alder seg på interesse? Vi kan lage et boksplott med `geom_boxplot`. Et boksplott kan vise hvordan en kontinuerlig variabel (alder) er fordelt innenfor en annen kategorisk variabel (politisk interesse). Boksen i midten representerer spennet til de 50% vanligste verdiene (andre og tredje kvartil), mens strekene viser spennet til verdiene i nedre og øvre kvartil.

```
ggplot(data = ess_subset, aes(x = news)) +
  geom_boxplot() +
  theme_minimal()
```

```
## Don't know how to automatically pick scale for object of type haven_labelled/vctrs_vctr/double. Defau
```

```
## Warning: Removed 34 rows containing non-finite values (stat_boxplot).
```



Hvis dere vil utforske hvordan man kan tilpasse de ulike diagrammene vi har sett på og mange andre, kan denne siden være nyttig: https://www.r-graph-gallery.com/index.html

BONUS: hente ut labels

Ved hjelp av pakken `labelled` kan vi hente ut informasjon om labels og bruke det i plots. NB. dette er en bonus for dere som vil lage fine plot og vil ikke komme på prøven. Først installerer og laste vi inn pakken:

```
install.packages("labelled")
```

```
library(labelled)
```

Så lager vi et datasett med informasjon om verdier og labels ved hjelp av funksjonen `val_labels`. I koden under skjer flere ting. Først henter jeg ut informasjon om verdier og labels og lagrer i et objekt. Så endrer jeg navnet på den variabelen med verdiene så navnet matcher variabelnavnet i det opprinnelige datasettet mitt. Så gjør jeg om radnavnene til en variabel fordi det er her informasjonen om labels er lagret. Til slutt filtrerer jeg ut labels for missing.

```
labels_vote <- data.frame(val_labels(ess_subset$vote)) %>%
  rename(vote = val_labels.ess_subset.vote.) %>%
  mutate(labels_vote = rownames(.)) %>%
  filter(!is.na(vote))
```

Etter å ha gjort dette så slår jeg sammen de to datasettene og plotter på nytt. Denne gangen bruker jeg variabelen `labels_vote` i tabell og plot:

```r
ess_subset <- ess_subset %>%
  full_join(labels_vote, by = "vote")

table(ess_subset$labels_vote)
```

```
##
##                   No Not eligible to vote                  Yes
##                  124                  125                 1156
```

```r
ggplot(data = ess_subset, aes(x = labels_vote)) +
  geom_bar()
```