Seminar 4

Lise Rødland

March 29, 2021

I løpet av dette seminaret skal vi:

1. Repetere litt om innlastning av data.
2. Missingverdier (NA).
3. Statistiske mål.
4. Univariat analyse.
5. Bivariat analyse.

Først installerer vi nye pakker og laster inn pakkene vi skal bruke i dagens seminar:

```
# Installerer nye pakker
install.packages("stargazer")
```

```
# Laster inn pakker
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(stargazer)
```

```
##
## Please cite as:

##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.

##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

Laste inn data og ulike typer av data.

Det neste vi skal gjøre er å laste inn data. Som vi allerede har snakket om så finnes det mange typer data og hver type krever egne koder for innlastning. På Canvas har jeg lastet opp et dokument som oppsummerer litt av dette. I dag skal vi laste inn data i csv.-format ved hjelp av funksjonen `read.csv`:

```
# For å laste inn .csv-filer
data <- read.csv("../../data/internett.csv")

# For å lagre .csv-filer
write.csv(data, file = "internett_ed.csv")
```

Datasettet heter internettbruk og omhandler internettbruken til italienere. Det betstår av et utvalg variabler hentet fra European Social Survey (ESS) runde 9 (2018). Enhetene er italienske statsborgere og samlet innholder datasettet 2745 observasjoner og 5 variable:

- Kjonn – Mann = 1, Kvinne = 2
- Alder – Alder til respondenten
- Utdanning – Antall år med fullført utdanning
- Tillit – Tillit til det italienske parlament (0-10), 0 = ingen tillit, 10 = fullstendig tillit
- Internettbruk – Hvor ofte bruker respondenten internett? (1-5), 1 = aldri, 5 = hver dag.

Før vi går videre vil vi se på dataene våre. Disse kodene har vi sett på tidligere, men vi repeterer det igjen.

Vi kan bruke `View()` til å åpne en egen fane med datasettet vårt

```
View(data)
```

Vi kan bruke `head()` til å undersøke de seks første observasjonene og `tail()` til å undersøke de seks siste observasjonene:

```
head(data)
```

```
##   X internettbruk kjonn alder utdanning tillit
## 1 1             5     2    67        18      8
## 2 2             5     1    45        11      6
## 3 3             1     2    73         8      0
## 4 4             5     1    21         8     NA
## 5 5             1     2    86         3      6
## 6 6             5     2    53        17      6
```

```
tail(data)
```

```
##         X internettbruk kjonn alder utdanning tillit
## 2740 2740             2     1    78         2      8
## 2741 2741             1     2    82        12      8
## 2742 2742             3     1    44        10      6
## 2743 2743             5     1    52        13      3
## 2744 2744             5     2    58        13      7
## 2745 2745             4     2    56         8      6
```

Vi kan bruker `summary()` på et datasett-objekt for å se målenivå, antall missingverdier og beskrivende statistikk:

```
summary(data)
```

```
##        X         internettbruk       kjonn           alder         utdanning
##  Min.   :   1   Min.   :1.000   Min.   :1.000   Min.   :16.00   Min.   : 0.0
##  1st Qu.: 687   1st Qu.:2.000   1st Qu.:1.000   1st Qu.:36.00   1st Qu.: 8.0
##  Median :1373   Median :5.000   Median :2.000   Median :52.00   Median :12.0
##  Mean   :1373   Mean   :3.629   Mean   :1.527   Mean   :51.28   Mean   :11.5
##  3rd Qu.:2059   3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:67.00   3rd Qu.:14.0
##  Max.   :2745   Max.   :5.000   Max.   :2.000   Max.   :90.00   Max.   :37.0
##                 NA's   :5                       NA's   :21      NA's   :85
##      tillit
##  Min.   : 0.000
##  1st Qu.: 2.000
##  Median : 5.000
##  Mean   : 4.251
##  3rd Qu.: 6.000
##  Max.   :10.000
##  NA's   :89
```

Missing (NA/Not available)

Det finnes mange grunner til at det er tomme celler/manglende verdier/svar i dataene. I datasett basert på spørreundersøkelser som ESS så kan det hende at noen respondenter ikke har ønsket å svare på alle spørsmål. I andre tilfeller kan det hende vi rett og slett mangler dataen. Vi skal nå se på hvordan vi kan finne missing-verdier og hva vi kan gjøre med dem i R. Men når dere skal gjøre egne analyser så er det er viktig å teoretisk begrunne hvordan man håndterer NA-verdier på bakgrunn av utvalget av populasjonen. Er missing-verdiene systematiske eller er de tilfeldige?

Når vi skal finne missing er det mest vanlig er å bruke funksjonen `is.na()`. `is.na()` tar utgangspunkt i en logisk test. For hver observasjon i et datasett eller en variabel så sjekker `is.na()` om det finnes missingverdier. Under ser dere et eksempel der jeg spør R om rad 1:6 har missingverdier:

```
is.na(data %>%
        slice_head(n = 6))
```

```
##          X internettbruk kjonn alder utdanning tillit
## [1,] FALSE         FALSE FALSE FALSE     FALSE  FALSE
## [2,] FALSE         FALSE FALSE FALSE     FALSE  FALSE
## [3,] FALSE         FALSE FALSE FALSE     FALSE  FALSE
## [4,] FALSE         FALSE FALSE FALSE     FALSE   TRUE
## [5,] FALSE         FALSE FALSE FALSE     FALSE  FALSE
## [6,] FALSE         FALSE FALSE FALSE     FALSE  FALSE
```

Som dere ser er verdiene i datasettet nå byttet ut med `FALSE` og `TRUE`. `TRUE` indikerer at observasjonen mangler data om denne variabelen. Et eksempel er observasjon/rad nummer 4 som mangler data på tillit-variabelen. `FALSE` indikerer at det finnes data.

Vi kan kombinere `is.na()` med `sum()` for å finne totalt antall missingverdier. `sum()` kan brukes til å summere, for eksempel:

```
sum(2,2,6)
```

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

Når `sum()` kombineres med `is.na()` så kan vi tenke oss at `TRUE` gir verdien 1 og `FALSE` gir verdien 0. Det betyr at en observasjon får verdien 1 på en gitt variabel dersom informasjonen mangler og 0 om den ikke mangler. R summerer så opp alle disse verdiene og summen angir antall observasjoner med verdien 1, altså med manglende informasjon.

```
# Finner ut om de seks første observasjonene har missing på variabelen tilitt
is.na(data %>%
        slice_head(n = 6) %>%  # Beholder bare de seks første observasjonene
        select(tillit))        # Beholder bare variabelen tillit
```

```
##      tillit
## [1,]  FALSE
## [2,]  FALSE
## [3,]  FALSE
## [4,]   TRUE
## [5,]  FALSE
## [6,]  FALSE
```

```
# Omdanner TRUE/FALSE til 1/0 ved hjelp av as.numeric
as.numeric(is.na(data %>%
                   slice_head(n = 6) %>%
                   select(tillit)))
```

```
## [1] 0 0 0 1 0 0
```

```
# Teller antall missing blant de seks første observasjonene
sum(is.na(data %>%
            slice_head(n = 6) %>%
            select(tillit)))
```

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

Vi kan kombinere `sum()` og `is.na()` til å telle totalt antall missingverdier i datasettet vårt. En rad, eller en observasjon, kan ha missing på flere variabler. Denne funksjonen vil telle alle disse.

```
sum(is.na(data))
```

```
## [1] 200
```

Vi kan kombinere `sum()`, `is.na()` og indeksering ved hjelp av `$` til å hente ut antall missingverdier på en variabel:

```
sum(is.na(data$internettbruk)) # Viser hvor mange missing det er på en variabel
```

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

Vi kan kombinere `sum()` med en funksjon som heter `complete.cases()` for å finne ut hvor mange observasjoner vi *ikke* mangler noe informasjon på noen variabler om:

```
complete.cases(data %>%
                 slice_head(n = 6))
```

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

```
sum(complete.cases(data))
```

```
## [1] 2562
```

Du kan også lese antall missing ut fra `summary`. Legg merke til at NAs (missingverdier) kommer til sist:

```
# For alle variabler i datasettet
summary(data)
```

```
##        X         internettbruk       kjonn           alder         utdanning
##  Min.   :   1   Min.   :1.000   Min.   :1.000   Min.   :16.00   Min.   : 0.0
##  1st Qu.: 687   1st Qu.:2.000   1st Qu.:1.000   1st Qu.:36.00   1st Qu.: 8.0
##  Median :1373   Median :5.000   Median :2.000   Median :52.00   Median :12.0
##  Mean   :1373   Mean   :3.629   Mean   :1.527   Mean   :51.28   Mean   :11.5
##  3rd Qu.:2059   3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:67.00   3rd Qu.:14.0
##  Max.   :2745   Max.   :5.000   Max.   :2.000   Max.   :90.00   Max.   :37.0
##                 NA's   :5                       NA's   :21      NA's   :85
##      tillit
##  Min.   : 0.000
##  1st Qu.: 2.000
##  Median : 5.000
##  Mean   : 4.251
##  3rd Qu.: 6.000
##  Max.   :10.000
##  NA's   :89
```

```
# For variabelen internettbrukt
summary(data$internettbruk)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   2.000   5.000   3.629   5.000   5.000       5
```

I en oppgave så vil en vanligvis kommenterer missingverdier. Vi må også velge hva vi skal gjøre med dem. Det er vanelig å fjerne NA hvis de er 'missing at random' eller 'missing completely at random'. Du kan velge å fjerne alle missing verdier eller bare missing verdier på spesifikke variable. Når vi begynner med analyser så vil R ta høyde for missingverdier og fjerne dem automatisk. Det er som når vi beregner gjennomsnittet - man kan ikke regne gjennomsnittet av missing, derfor må vi si til R hvordan R skal håndtere missing.

Dersom en ønsker et datasett uten missingverdier så kan man bruke funksjonen `drop_na()`.

```
# Fjerne alle observajoner med minst en missing
data1 <- data %>%
  drop_na()

# Fjerne alle observasjoner med missing på en variabel (eller fler)
data2 <- data %>%
```

```
  drop_na(internettbruk) # Du kan legge til flere variable med komma

# Vi skal ikke bruke data1 og data2 mer så jeg fjerner dem fra environment
rm(data1, data2)
```

Når dere skal skrive egne oppgaver så er det viktig å lese kodeboken nøye. Noen ganger kommer manglende data kodet som for eksempel 999 eller 9999. Da kan R tror at dette er en faktisk verdi og ikke en missingverdi. I slike tilfeller må en omkode variabelen før en bruker den i analyser.

Statistiske mål

Statistiske mål forteller oss noe om fordelingen til ulike variabler, som for eksempel gjennomsnitt, median og standardavvik, men også minimum- og maksimumverdier, typetall og frekvens. Statistiske mål på sentraltendens er gjennomsnitt, median og modus. Som vi har sett tidligere så kan vi få informasjon om dette ved å bruke samlefunksjonen `summary()` eller ved å bruke enkeltfunksjoner som for eksempel `mean()`, `min()` og `max()`. Ved bruk av enkeltfunksjonene så må vi huske å spesifisere `na.rm = TRUE` så R vet hvordan vi ønsker å håndtere missingverdier.

Statistiske mål på spredning i dataene er standardavviket og varians. Standardavviket viser respondentenes gjennomsnittlige avstand fra gjennomsnittet. Vi kan bruke funksjonen `sd()`. Også her må vi spesifisere `na.rm = TRUE` for at R skal vite hvordan vi ønsker å håndtere missingverdier. Syntaksen for `sd()` er som `mean()`:

```
sd(data$variabel, na.rm = TRUE)
```

Om vi ønsker å regne ut standardavviket for internettbruk så skriver vi:

```
sd(data$internettbruk, na.rm = TRUE)
```

```
## [1] 1.645191
```

Variansen er standardavviket opphøyd i annen. Dermed er standardavviket kvadratroten av variansen. Det er enklere å tolke standardavvik enn varians. Jeg viser likevel hvordan man finner variansen.

```
# Lagrer variansen i et eget objekt
varians <- var(data$internettbruk, na.rm = TRUE)

# bruker funksjonen sqrt() for å finne kvadratroten
sqrt(varians)
```

```
## [1] 1.645191
```

Vi kan bruke en logisk test for å sjekke om kvadratroten av variansen gir samme tall som standardavviket:

```
# Lagrer først standardavviket i et objekt:
stdavvik <- sd(data$internettbruk, na.rm = TRUE)

# Bruker logisk test for å spørre R om standardavviket er det samme som kvadrat-
# roten av variansen
stdavvik == sqrt(varians)
```

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

Vi skal ikke bruke objektene `stdavvik` og `varians` noe mer så vi fjerner dem fra environment:

```
rm(stdavvik, varians)
```

Univariat analyse: Deskriptiv statistikk med én variabel.

Når vi kun har én variabel vi vil beskrive, har vi å gjøre med univariate fordelinger. Da blir vi kjent med variablene hver for seg. En univariat fordeling gir oss informasjon om hvordan observasjonene fordeler seg på en variabels ulike verdier. Igjen gir `summary()`-funksjonen en rask oversikt over statistiske mål og deskriptiv statistikk. Det er her nyttig å gjøre seg godt kjent med de ulike statistiske målene. Men den univariate analysen kan ta ting et skritt videre, med for eksempel tabeller og histogrammer.

```
summary(data)
```

```
##        X        internettbruk       kjonn           alder         utdanning
##  Min.   :   1   Min.   :1.000   Min.   :1.000   Min.   :16.00   Min.   : 0.0
##  1st Qu.: 687   1st Qu.:2.000   1st Qu.:1.000   1st Qu.:36.00   1st Qu.: 8.0
##  Median :1373   Median :5.000   Median :2.000   Median :52.00   Median :12.0
##  Mean   :1373   Mean   :3.629   Mean   :1.527   Mean   :51.28   Mean   :11.5
##  3rd Qu.:2059   3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:67.00   3rd Qu.:14.0
##  Max.   :2745   Max.   :5.000   Max.   :2.000   Max.   :90.00   Max.   :37.0
##                 NA's   :5                       NA's   :21      NA's   :85
##      tillit
##  Min.   : 0.000
##  1st Qu.: 2.000
##  Median : 5.000
##  Mean   : 4.251
##  3rd Qu.: 6.000
##  Max.   :10.000
##  NA's   :89
```

Det er også lurt å gjøre seg kjent med klassen til variablene. Til det kan vi bruke `tibble()`, `str()` eller `class()`:

```
# tibble og str gir informasjon om hele datasettet
tibble(data)
```

```
## # A tibble: 2,745 x 6
##        X internettbruk kjonn alder utdanning tillit
##    <int>         <int> <int> <int>     <int>  <int>
##  1     1             5     2    67        18      8
##  2     2             5     1    45        11      6
##  3     3             1     2    73         8      0
##  4     4             5     1    21         8     NA
##  5     5             1     2    86         3      6
##  6     6             5     2    53        17      6
##  7     7             1     1    77        18      0
##  8     8             5     2    35        18      3
##  9     9             1     2    66        16      6
## 10    10             4     1    52        10      6
## # ... with 2,735 more rows
```

```
str(data)
```

```
## 'data.frame':    2745 obs. of  6 variables:
##  $ X            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ internettbruk: int  5 5 1 5 1 5 1 5 1 4 ...
##  $ kjonn        : int  2 1 2 1 2 2 1 2 2 1 ...
##  $ alder        : int  67 45 73 21 86 53 77 35 66 52 ...
##  $ utdanning    : int  18 11 8 8 3 17 18 18 16 10 ...
##  $ tillit       : int  8 6 0 NA 6 6 0 3 6 6 ...
```

```
# class gir informasjon om en variabel
class(data$internettbruk)
```

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

For kategoriske variabler, på nominalnivå eller ordinalnivå, kan vi bruke frekvenstabeller for å beskrive dataene med tall, og kake- og søylediagram for å beskrive dataene grafisk. Kake- og søylediagrammer lager vi ved hjelp av `ggplot()`. Dette så vi nærmere på i seminar tre så se tilbake til notatene derfra for å reptere dette.

Variabelen for kjønn er en kategorisk variabel. En frekvenstabell forteller oss hvor mange respondenter som er menn og hvor mange som er kvinner. Vi kan bruke funksjonen `table()`. `table()` gir kan brukes på en eller to variabler. Her skal vi se på tilfellet med en variabel:

```
table(data$kjonn)
```

```
##
##    1    2
## 1298 1447
```

```
# Lagrer table i et objekt, som vi kan eksportere til Word
tabell <- table(data$kjonn)
tabell1 <- data.frame(tabell)

# Bruker stargazer-pakken til å eksportere tabellen
stargazer(tabell1,
          type = "html",
          out = "../../output/sem4_tabkjonn.htm",
          summary = FALSE)
```

```
##
## <table style="text-align:center"><tr><td colspan="3" style="border-bottom: 1px solid black"></td></t
## <tr><td colspan="3" style="border-bottom: 1px solid black"></td></tr><tr><td style="text-align:left"
## <tr><td style="text-align:left">2</td><td>2</td><td>1,447</td></tr>
## <tr><td colspan="3" style="border-bottom: 1px solid black"></td></tr></table>
```

Vi kan gjøre det samme for internettbruk, som er på ordinalnivå.

```
tabell2 <- table(data$internettbruk)
tabell2
```

```
##
##    1    2    3    4    5
##  598  209  189  360 1384
```

Disse viser den absolutte fordelingen, altså totalt antall observasjoner for hver verdi. Vi kan også få den relative fordelingen mellom kategoriene, som viser prosentvis fordeling. Vi bruker prop.table()-funksjonen

```
tabell3 <- prop.table(table(data$internettbruk))
tabell3
```

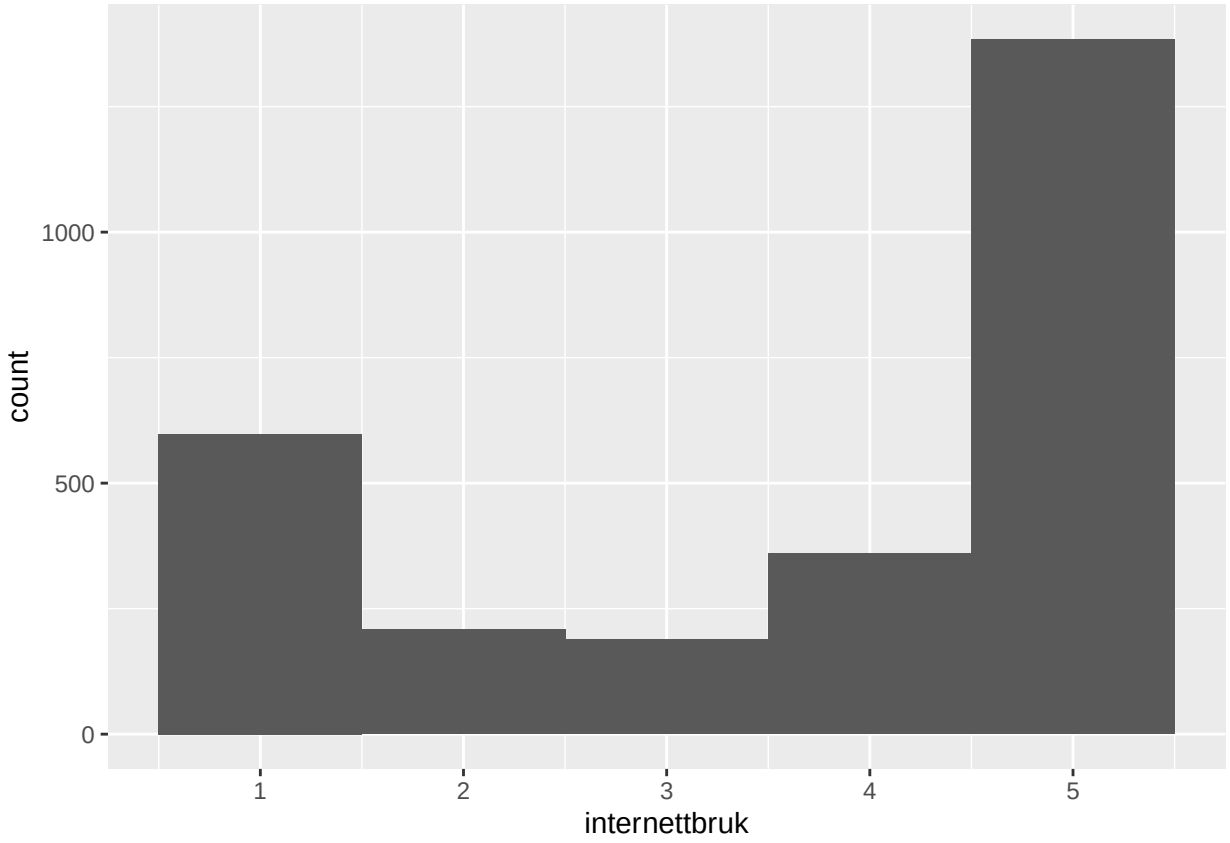
```
##
##          1          2          3          4          5
## 0.21824818 0.07627737 0.06897810 0.13138686 0.50510949
```

Det er alltid et poeng å lage grafer og figurer for å beskrive dataen. Det gir nemlig et godt visuelt og mer intutitvt inntrykk av dataene. For kategoriske variabler kan vi lage kake- og søylediagram for å beskrive frekvensfordelingene til variablene.

For å få søylediagram bruker vi funksjonen ggplot-funksjonen som er i pakken tidyverse.

```
# Søylediagram for internettbruk
        ggplot(data, aes(internettbruk)) +
  geom_bar(width = 1)
```

```
## Warning: Removed 5 rows containing non-finite values (stat_count).
```



```
# Prøv å legg på titler osv...

 # Her ser vi tydelig at det er flest som oppgir 5 som alternativ
```

For kontinuerlige variabler så kan vi også bruke frekvenstabeller, men da må vi først omkode variabelen. Dersom vi lager en frekvenstabell for alder, må vi omkode den til kategorier ved hjelp av `cut()`-funksjonen. I argumentet `breaks =` forteller jeg R hvor mange kategorier det skal være, og hvor skjæringspunktet skal være. Først undersøker jeg variabelen ved hjelp av `summary()`.

```
summary(data$alder) # Min er 16 år, og maks er 90 år. Lager så kategorier
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   16.00   36.00   52.00   51.28   67.00   90.00      21
```

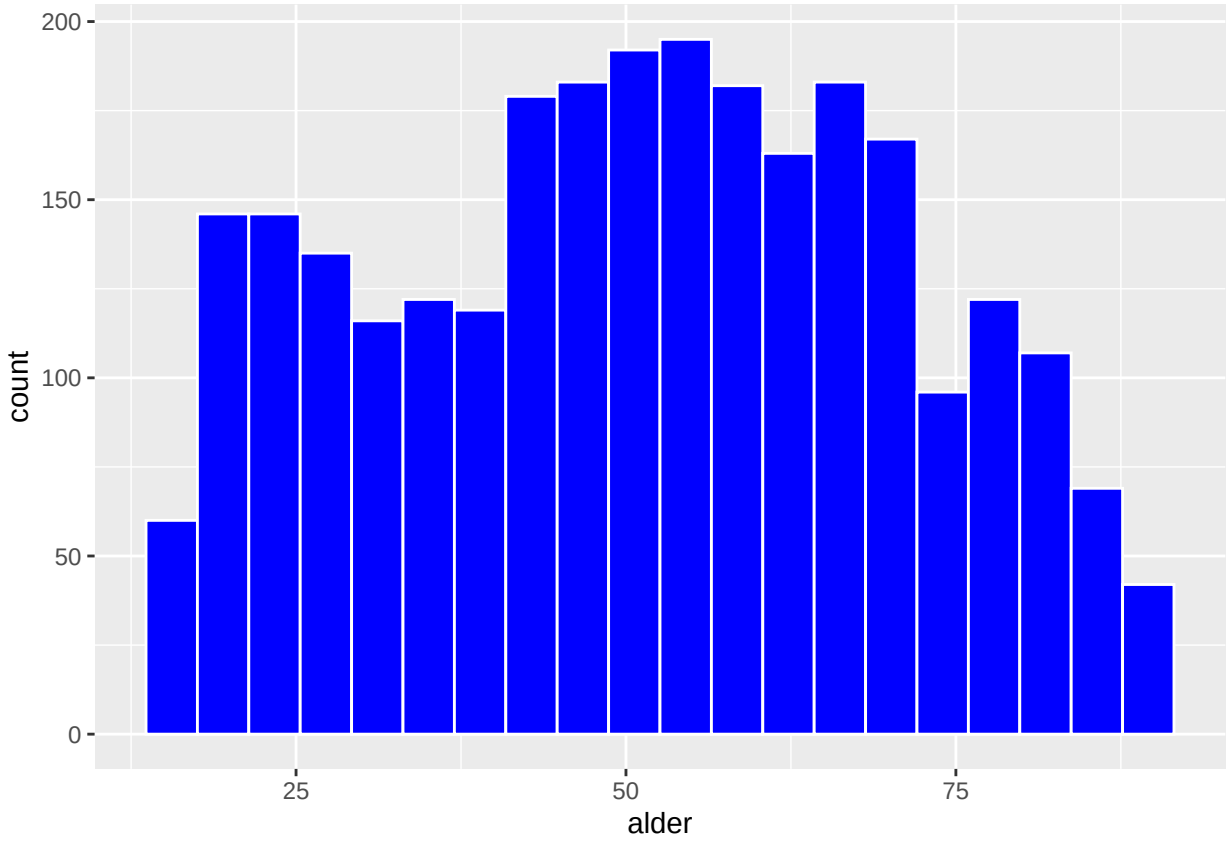
```
data$alder_kat <- cut(data$alder,
                          breaks = c(16, 30, 45, 60, 75, 90))

table(data$alder_kat)
```

```
##
## (16,30] (30,45] (45,60] (60,75] (75,90]
##     487     548     712     609     340
```

Grafiske fremstillinger er også nyttig med kontinuerlige variabler. Da kan vi blant annet bruke histogrammer. Også her må vi dele opp i kategorier. Vi bruker ggplot, men endrer `geom_`. Legg merke til argumentet `bins =` - dette bestemmer hvor mange søyler vi ønsker. Prøv å endre argumentet å se hva som skjer.

```
ggplot(data, aes(alder)) +
        geom_histogram(bins = 20,
                       color = "white",
                       fill = "blue")
```

```
## Warning: Removed 21 rows containing non-finite values (stat_bin).
```



I en større oppgave ønsker man ofte å presentere alle variablenes deskriptive statistikk i en felles tabell. Funksjonen `stargazer()` er fin til å gjøre dette.

```
stargazer(data,
          type = "text")
```

```
##
## ================================================================================
## Statistic       N     Mean    St. Dev.  Min   Pctl(25) Pctl(75)  Max
## --------------------------------------------------------------------------------
## X             2,745 1,373.000 792.558    1      687     2,059   2,745
## internettbruk 2,740   3.629    1.645   1.000   2.000    5.000   5.000
## kjonn         2,745   1.527    0.499     1       1        2       2
## alder         2,724  51.277    19.429  16.000  36.000   67.000  90.000
## utdanning     2,660  11.504    4.331   0.000   8.000    14.000  37.000
## tillit        2,656   4.251    2.525   0.000   2.000    6.000   10.000
## --------------------------------------------------------------------------------
```

```
# Vis hvordan du gjøre dette om til html.
```

Bivariat hypotesetesting

Kellstedt og Whitten presenterer tre typer hypotesetester for bivariate sammenhenger som vi skal se på i seminar i dag. Hvilken test som passer avhenger av målenivået på avhengig og uavhengig variabel:

- Kategorisk avhengig og uavhengig variabel: tabellanalyse
- Kontinuelig avhengig og kategorisk uavhengig variabel: sammenligne gjennomsnitt
- Kontinuerlig avhengig og uavhengig variabel: korrelasjonskoeffisient

Bivariat analyse brukes når man analyserer to variabler. Bivariat analyse er nyttig for å få oversikt over sammenhengen mellom to variabler, i tillegg til at det forteller oss noe om hvor mye to variabler korrelerer, altså hvor mye de henger sammen. Bivariat statistikk er også nyttig for å teste korrelasjonens statistiske signifikans.

Tabellanalyse

Dersom vi har to kategoriske variabler vi ønsker å sammenlikne så kan vi presentere dem i en krysstabell. Da bruker vi funksjonen `table()`. Vi kan opprette en krysstabell mellom internettbruk og kjønn i et nytt objekt kalt krysstabell. La oss si at vi har en hypotese om at menn

Det første vi gjør er å laste inn datasettet `ANES2016small`. Dette er samme datasett som Kellstedt og Whitten bruker. Datasettet er i .Rdata-format så da bruker vi funksjonen `load` til å laste inn data:

```
# Bytt ut det som står i hermetegn med filbanen og filnavnet på din maskin:
load("../../data/ANES2016small.RData")
```

Det første vi skal gjøre er å lage en krysstabell med absolutte antall. Da bruker vi funksjonen `table()`. I dag skal vi lagre krysstabellen i et eget objekt for å kunne bruke den videre. Jeg kaller objektet `krysstabell`:

```
krysstabell <- table(ANES2016small$V2Trump, ANES2016small$female)
```

Denne tabellen oppgir frekvensfordelingen i absolutte tall. Vi kan også finne relative tall, altså andeler ved hjelp av `prop.table`. `prop.table` krever et table-objekt og vi bruker derfor

```
prop.table(krysstabell, margin = 1)
```

```
##
##             0         1
##   0 0.4192277 0.5807723
##   1 0.5089667 0.4910333
```

```
# margin = 1 brukes for å regne ut fordelingen per linje, feks hvor mange
# menn relativt til kvinner som oppgir 1 på skalaen for internettbruk
```

Kjikvadrattesten tester sammenhengen mellom to kategoriske variabler. Den sammenlikner krysstabellen vi har, men en hypotetisk tabell fra et annet utvalg der det ikke er noen sammeheng mellom variablene. Så tester den sannsynligheten for at tabellen vår er generert ved en tilfeldighet. Vi bruker funksjonen chisq.test()

```
chisq.test(krysstabell)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  krysstabell
## X-squared = 19.371, df = 1, p-value = 1.076e-05
```

```
# X-squared, altså kjikvadratet er på 31.18, og
```

En alternativ måte å gjøre det på er å bruke funksjonen `CrossTable` fra pakken `gmodels`. Da må vi først installere og laste inn pakken:

```
install.packages("gmodels")
```

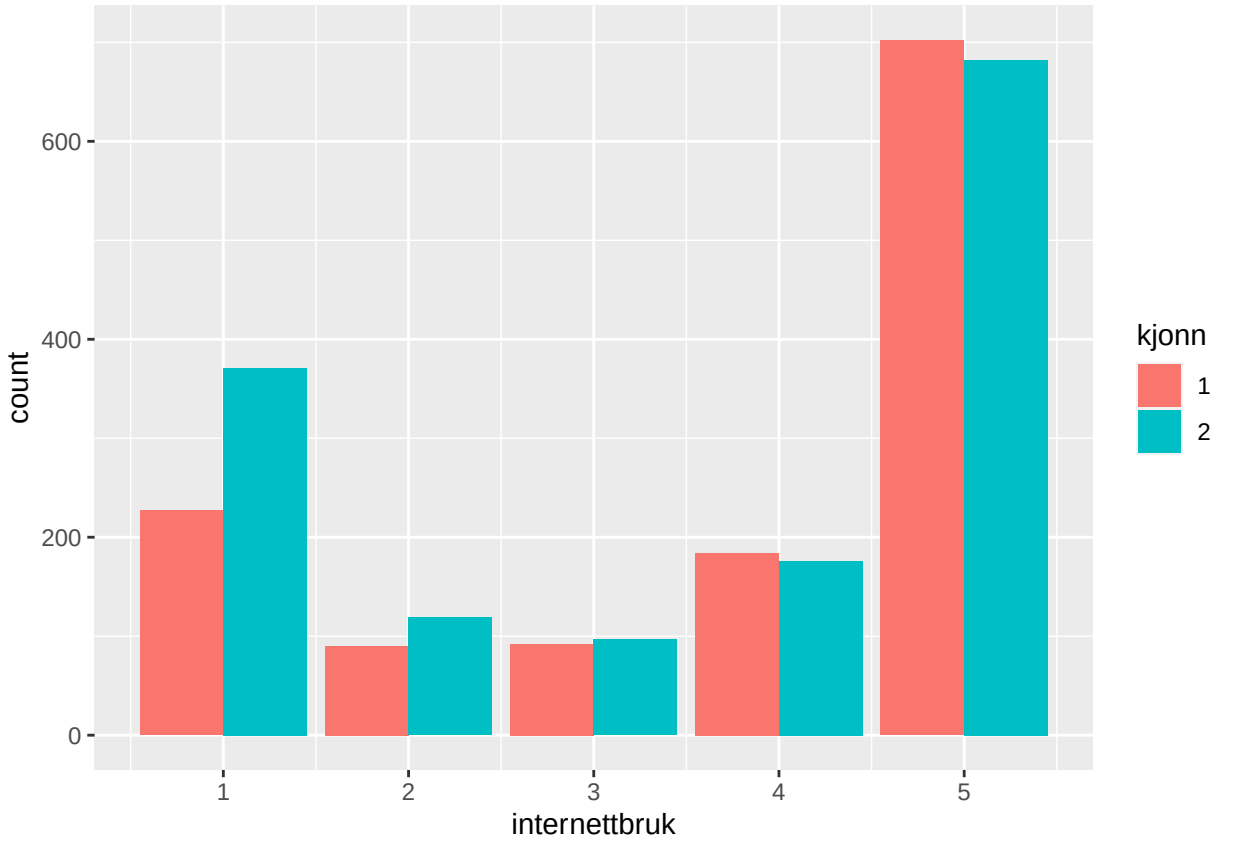
```
library(gmodels)

CrossTable(ANES2016small$V2Trump, ANES2016small$female)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2440
##
##
##                       | ANES2016small$female
## ANES2016small$V2Trump |         0 |         1 | Row Total |
## ----------------------|-----------|-----------|-----------|
##                     0 |       532 |       737 |      1269 |
##                       |     5.091 |     4.377 |           |
##                       |     0.419 |     0.581 |     0.520 |
##                       |     0.472 |     0.562 |           |
##                       |     0.218 |     0.302 |           |
## ----------------------|-----------|-----------|-----------|
##                     1 |       596 |       575 |      1171 |
##                       |     5.518 |     4.744 |           |
##                       |     0.509 |     0.491 |     0.480 |
##                       |     0.528 |     0.438 |           |
##                       |     0.244 |     0.236 |           |
## ----------------------|-----------|-----------|-----------|
##          Column Total |      1128 |      1312 |      2440 |
##                       |     0.462 |     0.538 |           |
## ----------------------|-----------|-----------|-----------|
##
##
```

Vi kan lage søylediagrammer for å presentere sammenhengen grafisk. Igjen, det er alltid lurt, også for deg selv. Det er mer intuitivt å tolke, og lettere å se sammenhenger raskt.

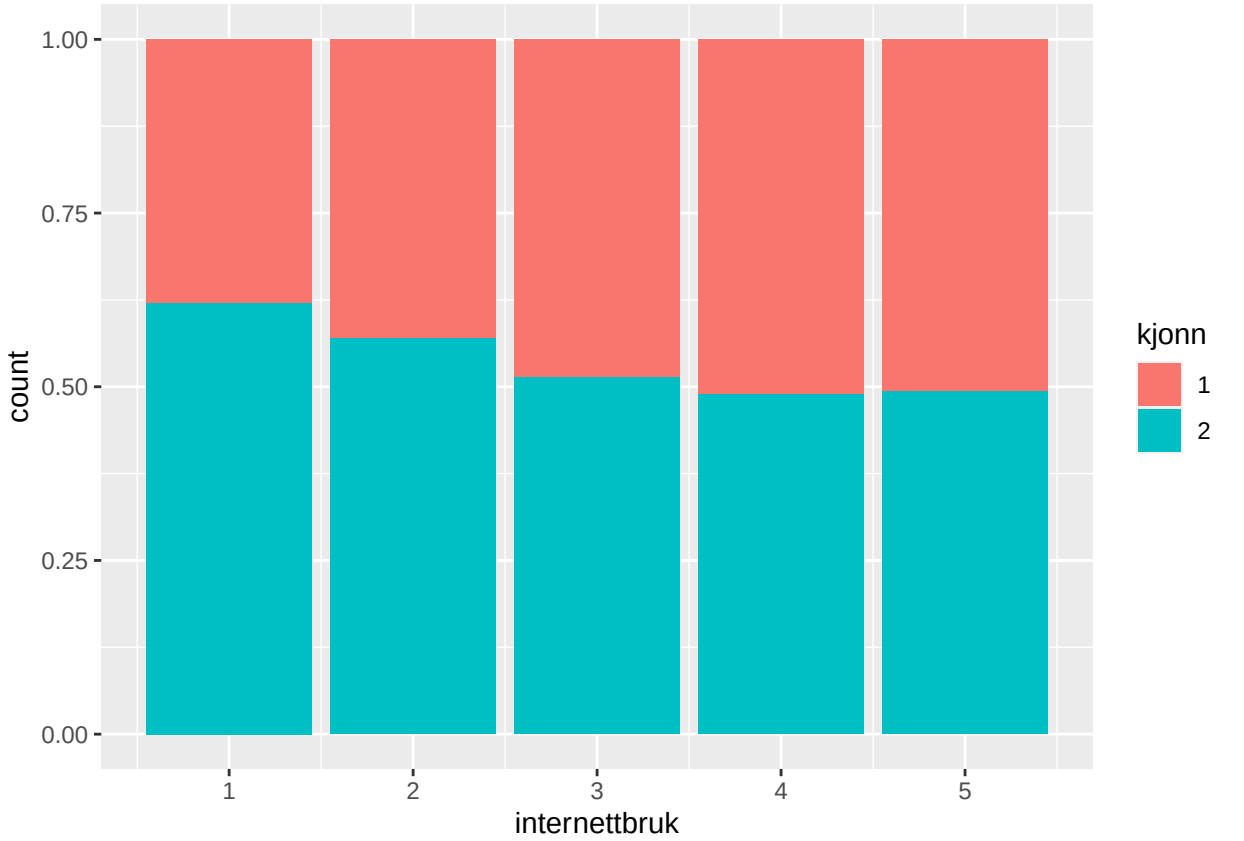
```
ggplot(data, aes(x = internettbruk,
                 fill = as.factor(kjonn))) +
 geom_bar(position = "dodge") +
labs(fill = "kjonn")
```

```
## Warning: Removed 5 rows containing non-finite values (stat_count).
```




```
# For relative tall

ggplot(data, aes(x = internettbruk,
                 fill = as.factor(kjonn))) +
 geom_bar(position = "fill") +
labs(fill = "kjonn")
```

```
## Warning: Removed 5 rows containing non-finite values (stat_count).
```



Vi avslutter med bivariat analyse med to kontinuerlige variabler. Dette er en forsmak på bivariat regresjonsanalyse som vi skal se mer på neste gang. Hensikten med dette er å beskrive korrelasjonen eller samvariasjonen mellom variablene. Vi kan beskrive denne sammenhengen med Pearsons r eller teste om korrelasjonen er statistisk signifikant.

Pearsons r beskriver styrken og retningen til korrelasjonen mellom to variabler. Den varierer fra -1 (negativ sammenheng) til 1 (positiv sammenheng). 0 indikerer ingen sammenheng. La oss teste med alder og utdanning. Vi bruker funksjonen `cor()`.

```
R <- cor(x = data$alder,
         y = data$utdanning,
         use = "pairwise.complete.obs")
R
```

```
## [1] -0.4090912
```

```
# Hva forteller dette oss?
```

Vi kan også sette opp en korrelasjonsmatrise for å utforske alle de bivariate korrelasjonene i datasettet mellom de akutelle variablene.

```
cor(data %>%
      select_if(is.numeric), use = "pairwise.complete.obs")
```

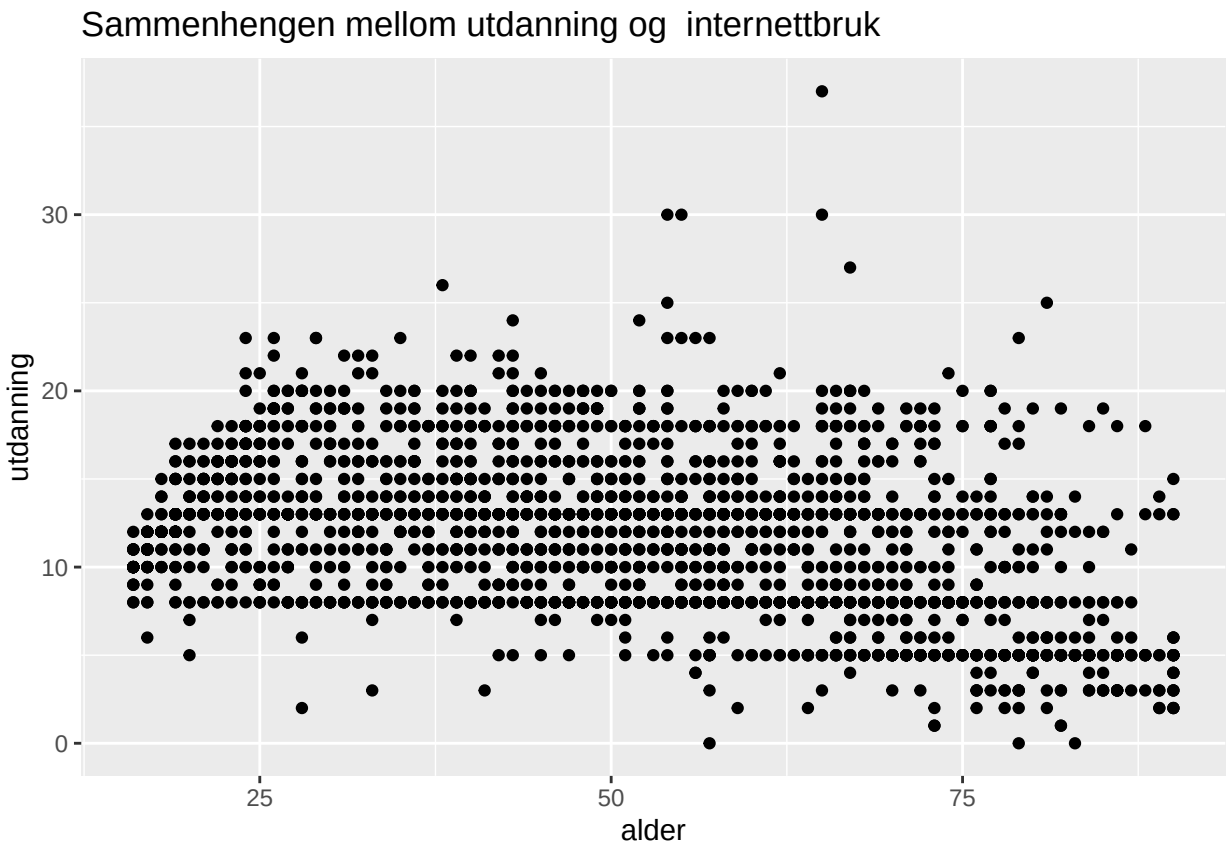
```
##                              X internettbruk       kjonn       alder
```

```
## X              1.0000000000   0.004242174 -0.02648290 -0.002426273
## internettbruk  0.0042421745   1.000000000 -0.10206670 -0.643609483
## kjonn         -0.0264829020  -0.102066704  1.00000000  0.066887813
## alder         -0.0024262730  -0.643609483  0.06688781  1.000000000
## utdanning     -0.0003138939   0.558348858 -0.05282830 -0.409091157
## tillit         0.0042579995   0.155872190 -0.04115814 -0.098498608
##                   utdanning       tillit
## X             -0.0003138939  0.004257999
## internettbruk  0.5583488578  0.155872190
## kjonn         -0.0528283014 -0.041158137
## alder         -0.4090911568 -0.098498608
## utdanning      1.0000000000  0.139119011
## tillit         0.1391190115  1.000000000
```

Spredningsdiagrammer egner seg godt for å grafisk fremstille sammenhengen mellom to kontinuerlige variabler. Den viser hvor hver respondent (observasjonsenhet) plasserer seg på x-aksen og y-aksen. Vi bruker ggplot med et annet geom-argument.

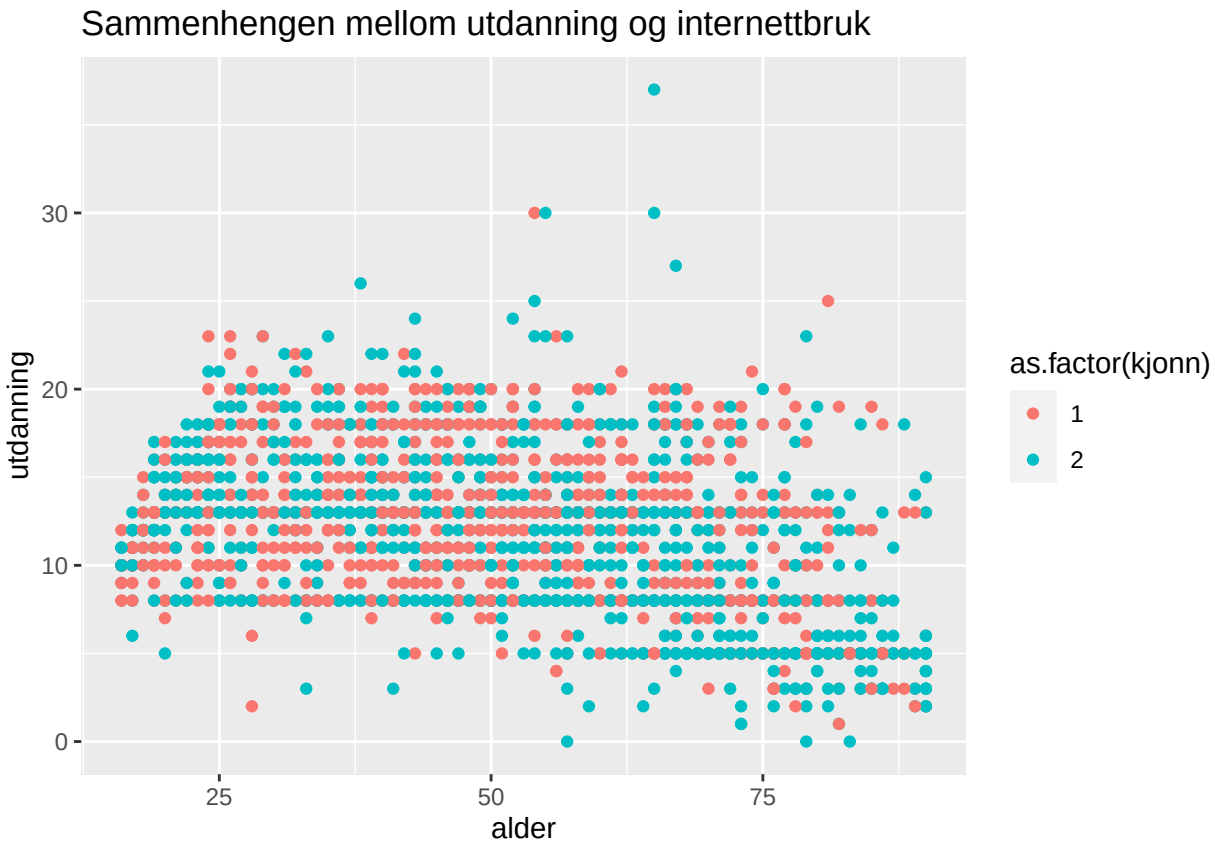
```
ggplot(data, aes(alder, utdanning)) +
geom_point() +
labs(title = "Sammenhengen mellom utdanning og  internettbruk")
```

```
## Warning: Removed 100 rows containing missing values (geom_point).
```



```
# Hva viser spredningsdiagrammet oss?

ggplot(data, aes(alder, utdanning,
color = as.factor(kjonn))) +
geom_point() +
labs(title = "Sammenhengen mellom utdanning og internettbruk")
```

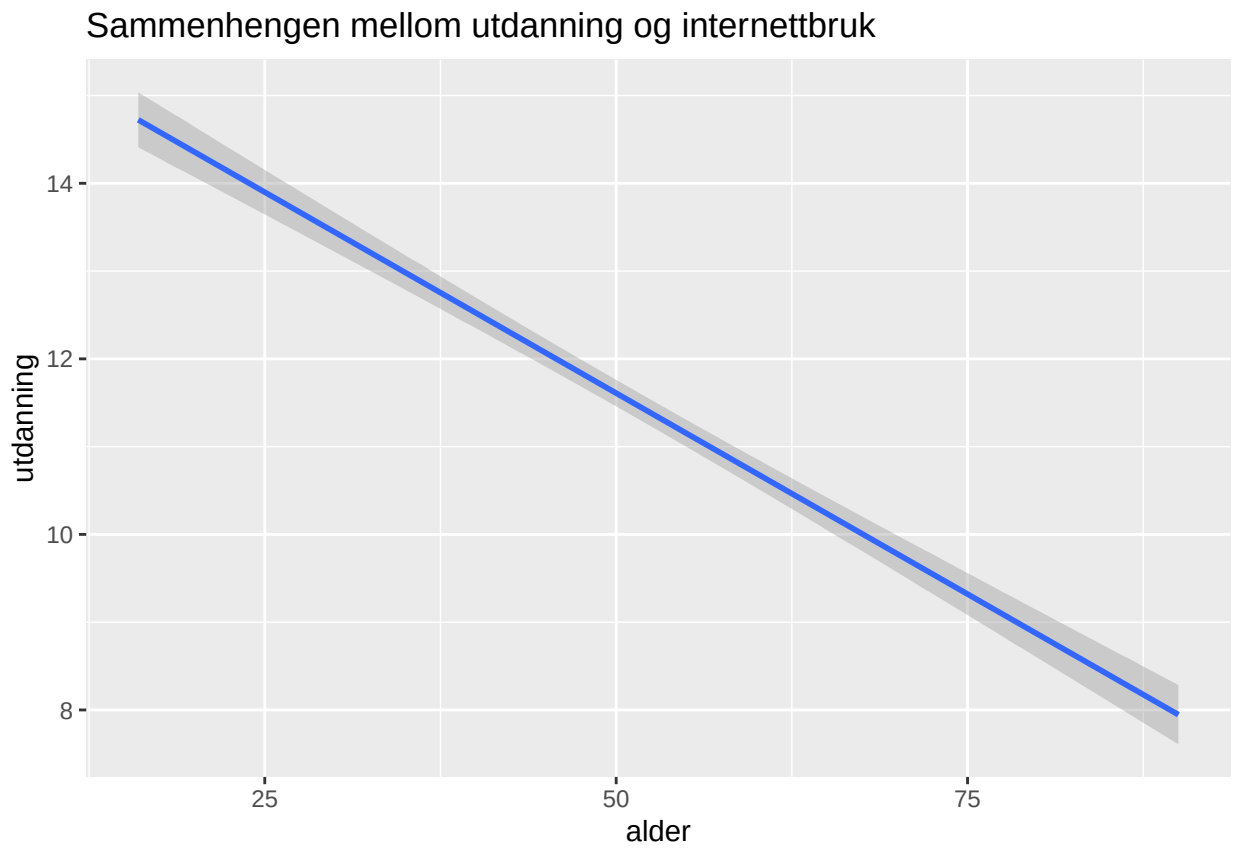
```
## Warning: Removed 100 rows containing missing values (geom_point).
```




```
# Plott med linje
ggplot(data, aes(alder, utdanning)) +
geom_smooth(method = "lm")+
labs(title = "Sammenhengen mellom utdanning og internettbruk")
```

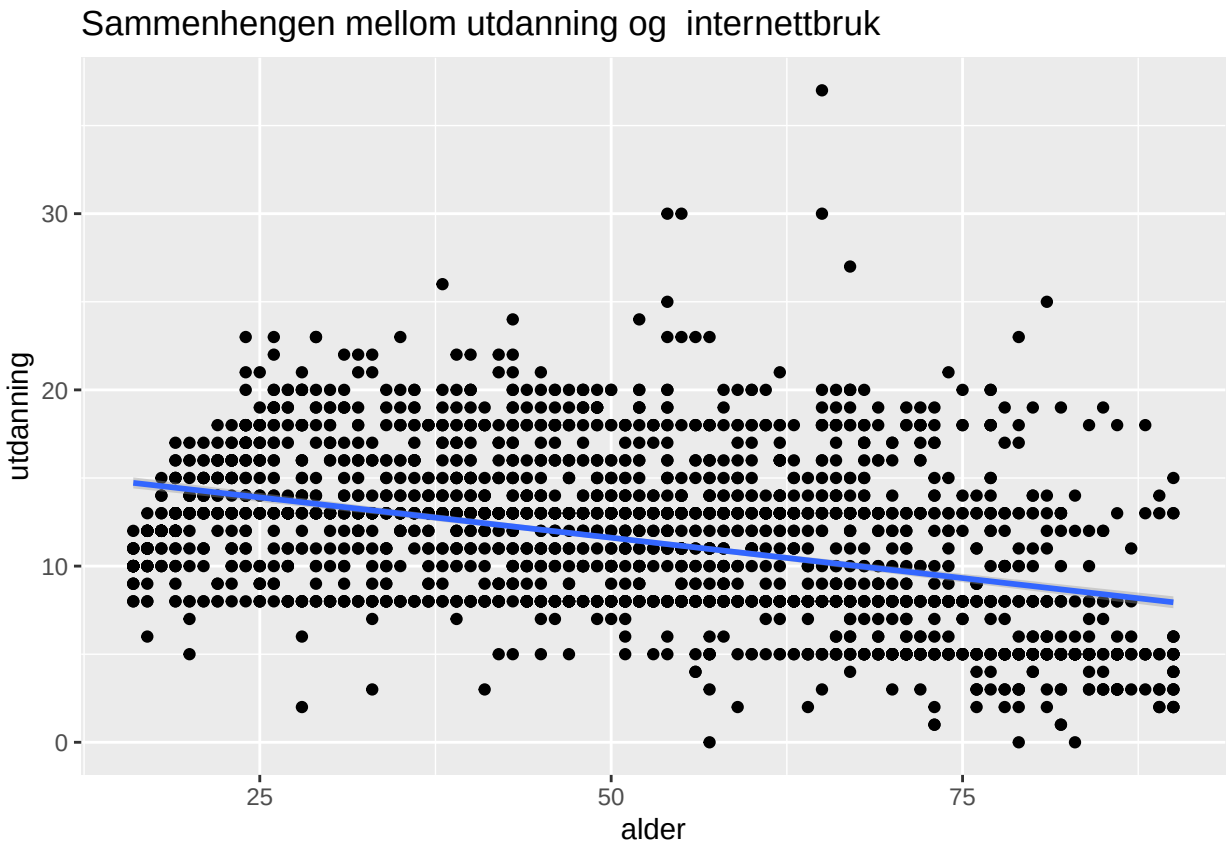
```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 100 rows containing non-finite values (stat_smooth).
```




```
# Plott med linje og punktestimater

ggplot(data, aes(alder, utdanning)) +
geom_point() +
geom_smooth(method = "lm") +
labs(title = "Sammenhengen mellom utdanning og  internettbruk")
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 100 rows containing non-finite values (stat_smooth).

## Warning: Removed 100 rows containing missing values (geom_point).
```



Dette er begynnelsen på en regresjonsanalyse, som er tema for neste seminar.