

Statistical learning og programmering

1. Semesterprojekt. Antal ord:

Af Kenneth Gottfredsen, Eva Rauff og Sanne Sørensen

2022-12-25

Indhold

Fra forretningsproblem til datamining problem	1
Import	2
Tidying af vores datasæt	2
Tidy	21
Transformer	22
Visualiser	23
Model	24
Kommunikér/analyse	25
Sessioninformation	25
Litteratur	25
Bilag	25

Fra forretningsproblem til datamining problem

Thise fortæller, at de har svært at forudsige præcist hvor mange liter koldskål produktionsafdelingen skal producere. Dette har resulteret i, at de ikke har kunne producere nok koldskål i år, fordi flere af butikkerne i området har oplevet at deres kølediske har været tomme for koldskål i sommerperioden. Den kvantitative del af analysen er afgrænset til COOP butikker i nærheden af Landbohøjskolen, hvis beliggenhed er i Københavnområdet. Butikkerne afgiver ordre til Thises fjernlager, hvorefter koldskålen bliver leveret ud til butikkerne.

Formålet med analysen er derfor, at beregne en multibel lineær regressionsmodel som bedst kan forudsige butikkernes efterspørgsel på koldskål i området omkring Landbohøjskolen. Derudover vil vi også finde ud af, hvordan vejret og andre vejr-relateret faktorer påvirker butikkernes efterspørgsel på koldskål. Med denne fremgangsmåde kan Thise få løst deres forretningsproblem. Vores datamining problem går ud på, at identificere de forskellige vejr-variablers effekt på efterspørgslen af koldskål.

Dertil vil vi bringe analysens resultater i samspil med Thises grad af datamodenhed, da vi har vurderet virksomhed til, at være i startfasen rent datamodenhedsmæssigt. Vi vil derfor tilbyde dem en række datainitiativer. Disse værktøjer kan de bruge til, ikke kun at forudsige efterspørgslen af koldskål, men også i forbindelse med andre varenumre som fx. Thises græske yoghurt eller skyr fremadrettet. På denne måde kan Thise mejeri vha. en øget datamodenhed forøge deres samlede omsætning betydeligt over tid, såfremt de tager datainitiativerne i anvendelse. I næste afsnit startes der ud med, at importere undersøgelsens datasæt.

Først indlæses `pacman::load()`:

```
# Inden vi går i gang bruger vi pacman() til at installere og indhente relevante  
# pakker på samme tid.
```

```
pacman::p_load("tidyverse", "magrittr", "nycflights13", "gapminder",  
               "Lahman", "maps", "lubridate", "pryr", "hms", "hexbin",  
               "feather", "htmlwidgets", "broom", "pander", "modelr",  
               "XML", "httr", "jsonlite", "lubridate", "microbenchmark",  
               "splines", "ISLR2", "MASS", "testthat", "leaps", "caret",  
               "RSQLite", "class", "babynames", "nasaweather",  
               "fueleconomy", "viridis", "readxl", "timeDate", "tinytex",  
               "ggbeeswarm", "palmerpenguins", "hms", "RColorBrewer",  
               "boot", "openxlsx", "writexl", "PerformanceAnalytics", "car")
```

Import

I første omgang vil vi importere datasættet:

```
# Indlæser datasæt og gemmer det nye datasæt i et objekt.
```

```
data1 <- read_excel("data/stud_exam_data.xlsx")
```

```
# Dernæst undersøges strukturen i datasættet.
```

```
str(data1)
```

```
## tibble [152 x 4] (S3: tbl_df/tbl/data.frame)
```

```
## $ date          : POSIXct[1:152], format: "2022-04-01" "2022-04-02" ...
```

```
## $ efterspørgsel  : num [1:152] 367 361 376 47 367 402 416 355 283 454 ...
```

```
## $ kammerjunkere  : chr [1:152] "0" "0" "0" "1" ...
```

```
## $ forventet_l_lager: chr [1:152] "3" "3" "3" "3" ...
```

Tidying af vores datasæt

Nu har vi fået indlæst datasættet. Det næste skridt er at transformere de forskellige variabler.

I følgende kode-chunk vil vi rekode og transformere de udvalgte variabler så de stemmer overens med eksamensbesvarelsen. Hele kodestumpen vil blive kædet sammen med 'pipe' funktionen' fra dplyr pakken. Omkodningerne bliver til sidst gemt i en ny dataframe som vi kalder data1.

Derefter bruger vi `mutate()` til at lave en ny kolonne ud fra data1. Først laver vi en date-variabel, som vi koder til et date objekt med `ymd()` fra lubridate pakken.

I den næste del anvendes `mutate()` til, at lave en kolonne der hedder dag, som bliver omkodet til en faktor. Dernæst koder vi date til et objekt med `ymd()` funktionen fra lubridate pakken. "lubridate.week.start",1=mandag, istedet for søndag som er standardstillingerne i R.

Dernæst bruger vi `mutate()` igen til at danne en ny weekend-variabel der hedder weekend_1. I denne sammenhæng vælger vi at fredag, lørdag, søndag og fire andre helligdage er 1, ellers er de andre værdier 0. Dette kaldes for en dummyvariabel.

Måned, dag, kamjunk, forvent_lager og weekend_1 er alle kategoriske faktorer. For at gøre det nemmere at forstå hvad de forskellige værdier udtrykker, navngiver vi disse med `fct_recode()` funktionen.

```
data1 <- data1 %>%
  mutate(date = ymd(date), måned = factor(month(date)),
         kamjunk = factor(kammerjunkere), forvent_lager =
         factor(forventet_l_lager)) %>%
  mutate(dag = as.factor(wday(date, week_start =
                           getOption("lubridate.week.start", 1)))) %>%
  mutate(weekend_1 = as.integer(dag %in% c("5", "6", "7") | date %in%
                                ymd("2022-04-14", "2022-04-18", "2022-05-26",
                                    "2022-06-06")) %>%
  mutate(weekend = factor(weekend_1)) %>%
  mutate(data1, kamjunk = fct_recode(kammerjunkere, "ja" = "0",
                                     "nej" = "1")) %>%
  mutate(data1, forvent_lager = fct_recode(forventet_l_lager, "lav" = "1",
                                             "mellem" = "2", "høj" = "3")) %>%
  mutate(data1, måned = fct_recode(måned, "april" = "4", "maj" = "5",
```

```

        "juni" = "6", "juli" = "7",
        "august" = "8")) %>%
mutate(data1, dag = fct_recode(dag, "mandag" = "1", "tirsdag" = "2",
        "onsdag" = "3", "torsdag" = "4",
        "fredag" = "5", "lørdag" = "6",
        "søndag" = "7")) %>%
dplyr::select(date, måned, dag, efterspørgsel, kamjunk, forvent_lager,
        weekend_helligdag = weekend)

```

I denne kodechunk vil vi lave en HTTP GET-anmodning til en API fra DMI. Vi skal bruge adgangen til at få de relevante vejr-variable som vi senere skal bruge i vores analyse. API'en leverer til slut et objekt i JSON format som bliver transformeret om til en dataframe i stedet for en liste.

Først bruger vi `base_url` og `info_url` til at anmode om vejrdata fra DMI's API. `req_url` bruges til at udvælge specifikke parametre fra API'en.

I denne kodechunk vil vi transformere den data vi har hentet fra vores API-kald til nogle mere brugbare data.

Først bruger vi `base_url` og `info_url` til at anmode om vejrdata fra DMI's API. `req_url` bruges til at udvælge specifikke parametre fra API'en.

Derefter bruger vi `pivot_wider`-funktionen til at sprede variablerne ud i separate kolonner.

Vi bruger derefter `Mutate`-funktionen til at konvertere kolonnen 'målingstidspunkt' til en datoformat. `Separate`-funktionen bruges til at opdele kolonnen 'målingstidspunkt' i to separate kolonner som vi navngiver 'date' og 'time'.

`Filter`-funktionen udvælger rækker, der indeholder de første fire characters: "12:0".

```

data2 <- as.data.frame(do.call(cbind, list_dmi))
data2 <- dplyr::select(data2, features.properties.observed,
        features.properties.value,
        features.properties.parameterId) %>%
rename(værdi = features.properties.value, parameter =
        features.properties.parameterId,

```

```

    målingstidspunkt = features.properties.observed) %>%
pivot_wider(names_from = parameter, values_from = værdi) %>%
mutate(målingstidspunkt = as_datetime(målingstidspunkt)) %>%
separate(målingstidspunkt, into = c('date', 'time'), sep = " ") %>%
filter(str_sub(time, 1, 4) == "12:0") %>%
mutate(date = as_date(date)) %>%
mutate(time = as_hms(time)) %>%
dplyr::select(-(temp_max_past12h:temp_min_past12h))

```

I nedestående kode-chunk merger vi data1 og data2 til data3 for at beholde alle observationer i x.

Vi bruger derefter mutate-funktionen til at oprette fire nye variabler i data3 kaldet temp_gt25_3_dage. Lag-funktionen er brugt til at lave variablerne, som har opfanget forsinkede værdier fra temp1, temp2 og temp3. Afslutningsvis dannes variabelen 'temp_gt25_3_dage', som måler de dage hvor der har været mere end 3 dage i træk med >= 25 grader. Det er en dummyvariabel fordi vi bruger if_else.

```

data3 <- data1 %>%
left_join(data2, data1, by = c("date" = "date"))
dplyr::select(data3, date, time, weekend_helligdag, everything())

```

```

data3 <- data3 %>%
  mutate(temp1 = lag(temp_max_past1h, 1),
         temp2 = lag(temp_max_past1h, 2),
         temp3 = lag(temp_max_past1h, 3),
         temp1 = if_else(is.na(temp1), 0, temp1),
         temp2 = if_else(is.na(temp2), 0, temp2),
         temp3 = if_else(is.na(temp3), 0, temp3),
         temp_gt25_3_dage = if_else(temp1 >= 25 & temp2 >= 25 & temp3 >= 25,
                                   1, 0))

```

Først identificerer vi outliers i vores dataset data3. Derefter fjerner vi 1 outlier som er 47.

Derefter laver vi en ggplot for at se fordelingen af efterspørgslen af koldskål i form af et histogram.

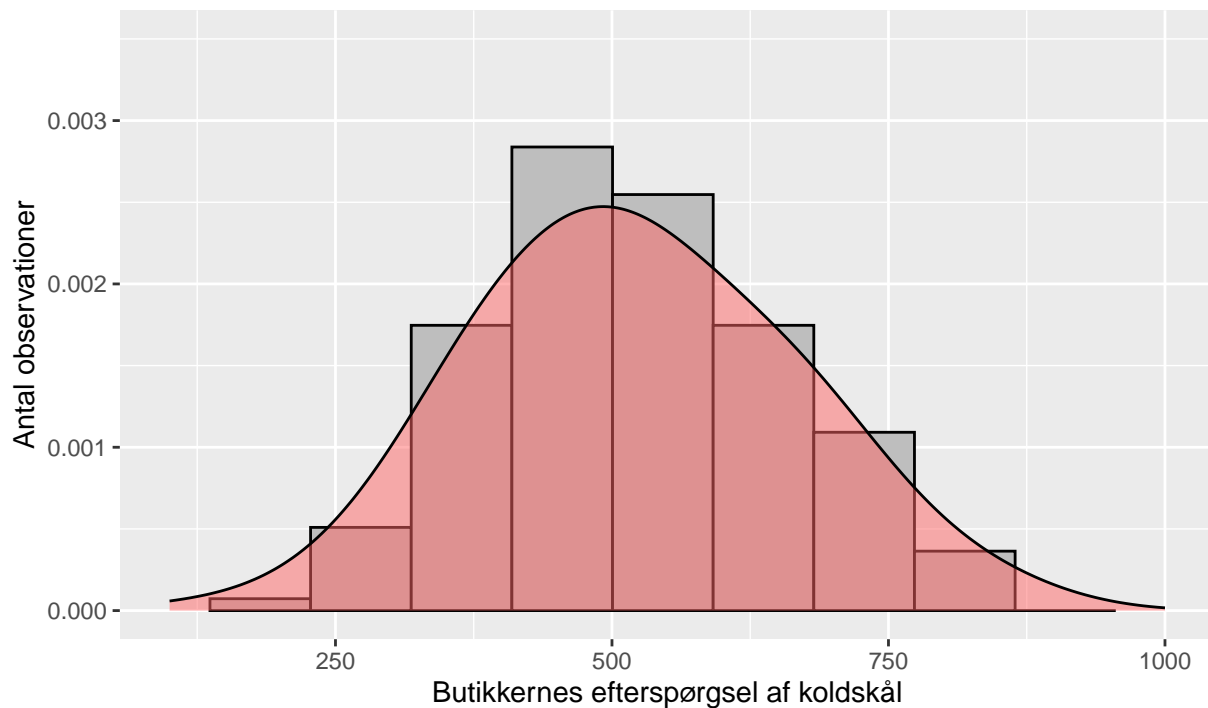
Vi bruger `geom_density()` funktionen til, at forstå fordelingen og til at forudsige den forventede fordeling af efterspørgslen på koldskål. Man kan se at at spredningen af observationerne er størst omkring 500. Endvidere kan det ses, at efterspørgslen af koldskål er tilnærmelsesvis normalfordelt, og at sandsynlighedskurven er symmetrisk klokkeformet. Dog kan vi også se, at nogle af observationerne falder udenfor, hvilket kan skyldes tilfældig variation. Vi ved derfor, at ca. 50% af observationerne befinder sig til venstre og højre af midten dvs. middelværdien. At vores data er normalfordelt er godt, fordi den lineære regressionsmodel som vi senere vil udføre er en parametrisk test, som kræver at data er normalfordelt.

```
data3 <- data3 %>%
  filter(efterspørgsel > 47)

ggplot(data3, aes(x = efterspørgsel)) +
  geom_histogram(aes(y = ..density..), colour = "black",
                 fill = "gray", binwidth = 91) +
  geom_density(alpha=0.5, fill="#FF6666", adjust=1.6) +
  labs(title = "Histogram over butikkernes efterspørgsel af koldskål",
       subtitle = "Undersøger om efterspørgslen er normaltfordelt",
       y = "Antal observationer",
       x = "Butikkernes efterspørgsel af koldskål",
       caption = "Kilde: Thise Mejeri 2022") +
  ggeasy::easy_center_title() + # Centrerer titlen.
  theme(plot.title = element_text(hjust = 0.5, size = 16),
        plot.subtitle = element_text(hjust = 0.5, size = 14),
        plot.caption = element_text(hjust = 1, face = "italic", size = 10)) +
  xlim(100, 1000) + ylim(0, 0.0035) +
  theme_gray()
```


Histogram over butikkernes efterspørgsel af koldskål

Undersøger om efterspørgslen er normalfordelt



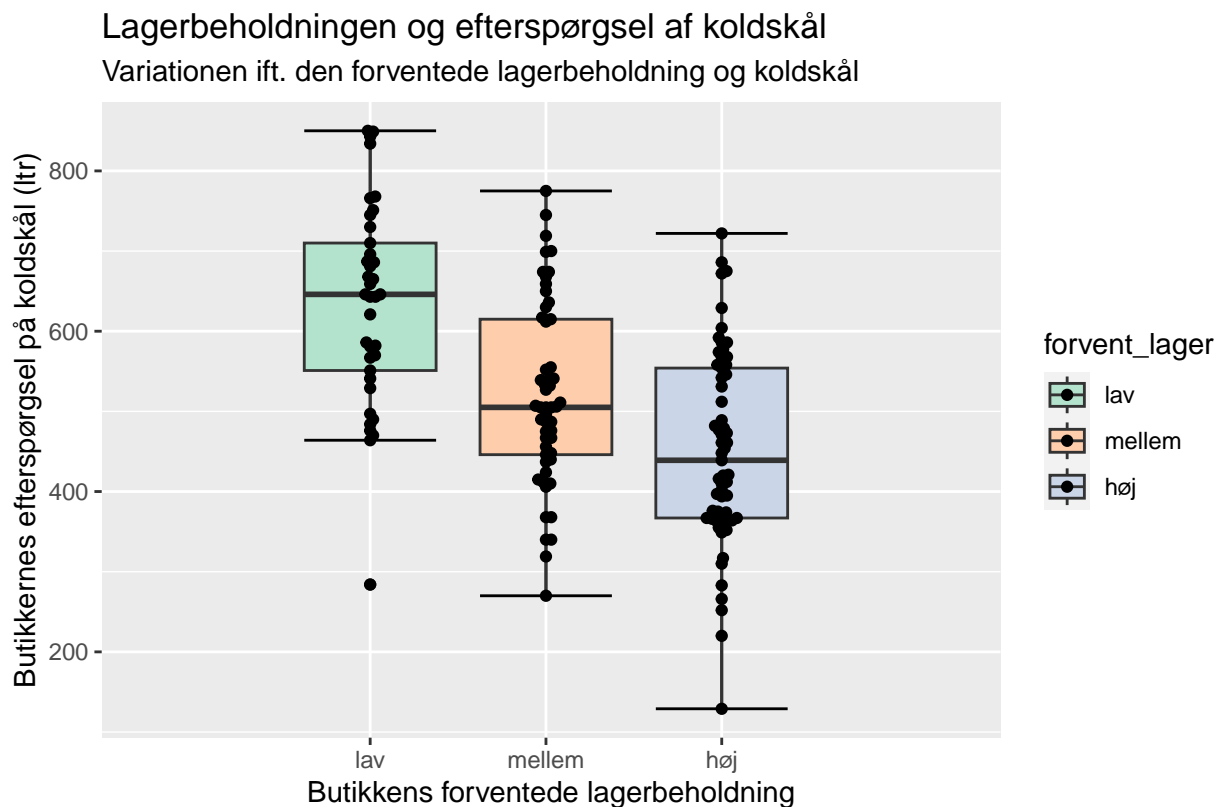
Kilde: Thise Mejeri 2022

På følgende kode-chunk har vi lavet et boxplot til at vise den statistiske variationen ift. butikkens forventede lagerbeholdning og efterspørgslen på koldskål.

Her kan man se at median-efterspørgslen stiger når man går fra høj til lav forventet lagerbeholdning af koldskål. Dette tyder også på at der er en sammenhæng mellem de 2 variabler.

```
ggplot(data = data3, mapping = aes(x = forvent_lager, y = efterspørgsel, fill =  
                                     forvent_lager)) +  
  stat_boxplot(geom = 'errorbar') + # whiskers.  
  geom_boxplot() +  
  labs(title = "Lagerbeholdningen og efterspørgsel af koldskål",  
        subtitle = "Variationen ift. den forventede lagerbeholdning og koldskål",  
        caption = "Kilde: Tal fra DMI 2002. Fra perioden 1/4/22-30/8/22",  
        y = "Butikkernes efterspørgsel på koldskål (ltr)",  
        x = "Butikkens forventede lagerbeholdning") + # Undgår overplotting  
  geom_beeswarm(dodge.width=3, cex =1, color = "black") + # Justerer boksbredden  
  ggeasy::easy_center_title() + # Centrerer titlen.
```

```
theme( plot.title = element_text(hjust = 0.5, size = 16),
       plot.subtitle = element_text(hjust = 0.5, size = 14),
       plot.caption = element_text(hjust = 2.3, face = "italic",size = 10 )) +
scale_fill_brewer(palette = "Pastel2") +
theme_gray()
```



Kilde: Tal fra DMI 2002. Fra perioden 1/4/22–30/8/22

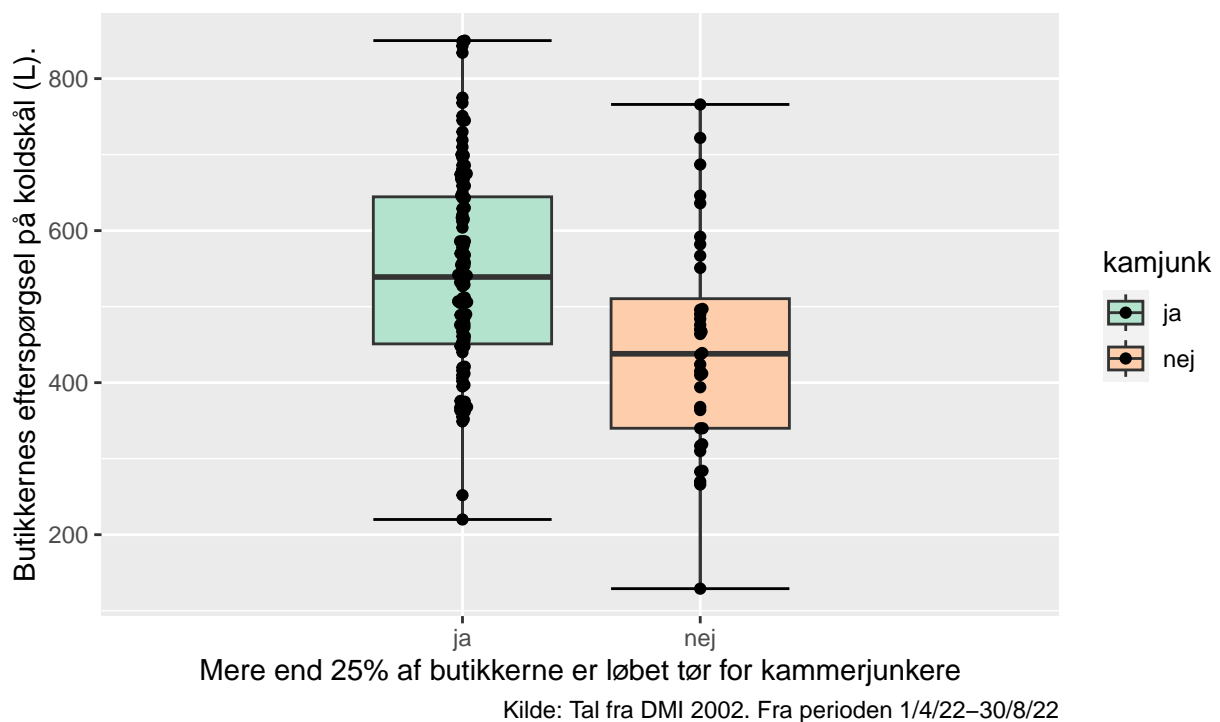
I næste kode-chunk har vi lavet et boxplot som viser fordelingen af efterspørgslen i forhold til om 25% af butikkerne er løbet tør for kammerjunkere eller ej. På baggrund af plottet kan vi se at hvis butikkerne ikke har kammerjunkere på lageret så falder efterspørgslen. Konklusionen er at efterspørgslen på koldskål stiger når de er løbet tør for kammerjunkere.

```
ggplot(data = data3, mapping = aes(x = kamjunk, y = efterspørgsel, fill =
                                   kamjunk)) +
  stat_boxplot(geom = 'errorbar') +
  geom_boxplot() +
  labs(title = "Kammerjunkere og efterspørgsel af koldskål",
       subtitle = "Variationen ift. den forventede lagerbeholdning af kammerjunkere
       og koldskål",
```

```
caption = "Kilde: Tal fra DMI 2002. Fra perioden 1/4/22-30/8/22",
y = "Butikkernes efterspørgsel på koldskål (L).",
x = "Mere end 25% af butikkerne er løbet tør for kammerjunkere") +
ggplot::easy_center_title() + # Centrerer titlen.
geom_beeswarm(dodge.width=3,cex=0.5, color = "black") + # Justerer boksbredden.
theme( plot.title = element_text(hjust = 0.5, size = 16),
        plot.subtitle = element_text(hjust = 0.5, size = 14),
        plot.caption = element_text(hjust = 1.8, face = "italic",
                                     size = 10 )) +
scale_fill_brewer(palette = "Pastel2") +
theme_gray()
```

Kammerjunkere og efterspørgsel af koldskål

Variationen ift. den forventede lagerbeholdning af kammerjunker og koldskål



Forneden har vi lavet et boxplot som viser sammenhængen mellem måned og efterspørgslen af koldskål. Det er tydeligt at se at efterspørgslen stiger fra april-juli hvorefter den falder efterspørgslen i august. Hvilket kan tyde på at efterspørgslen af koldskål hænger moderat sammen med sommerperioden. Bemærk også at kattehårene på boksplottene er forholdsvis brede, dette indikerer fx. at observationerne ligger langt væk fra medianen hvilket øger den statistiske usikkerhed.

```

ggplot(data = data3, mapping = aes(x = måned, y = efterspørgsel,
                                   fill = måned)) +

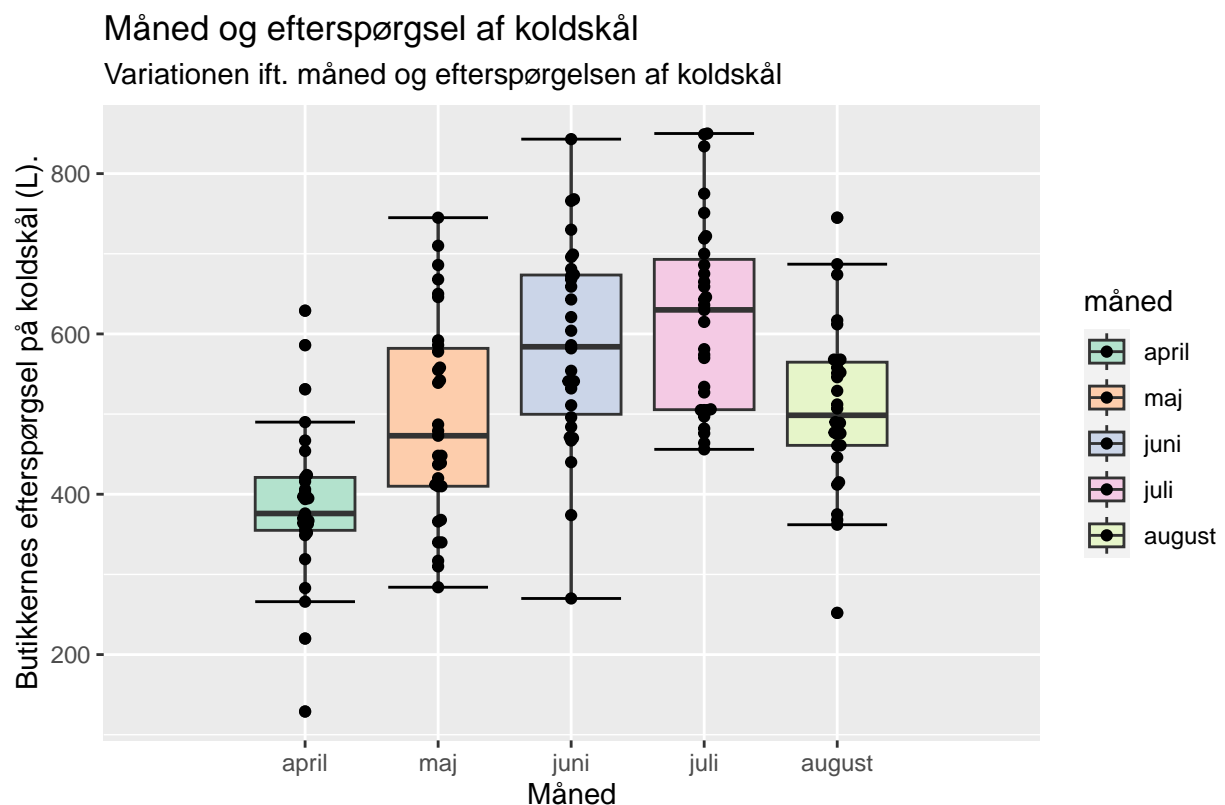
  stat_boxplot(geom = 'errorbar') +
  geom_boxplot() +

  labs(title = "Måned og efterspørgsel af koldskål",
       subtitle = "Variationen ift. måned og efterspørgelsen af koldskål",
       caption = "Kilde: Tal fra DMI 2002. Fra perioden 1/4/22-30/8/22",
       y = "Butikkernes efterspørgsel på koldskål (L).",
       x = "Måned") +

  ggeasy::easy_center_title() + # Centrerer titlen.
  geom_beeswarm(dodge.width=3,cex=0.5, color = "black") + # Justerer boksbredden.
  theme( plot.title = element_text(hjust = 0.5, size = 16),
        plot.subtitle = element_text(hjust = 0.5, size = 14),
        plot.caption = element_text(hjust = 1.3, face =
                                   "italic", size = 10 )) +

  scale_fill_brewer(palette = "Pastel2") +
  theme_gray()

```



```

attach(data3)
modell1 <- lm(efterspørgsel ~ temp_mean_past1h, data = data3)
predict(modell1, data.frame(temp_mean_past1h = (c(10,20,30))), interval = "prediction",

##          fit          lwr          upr
## 1 440.6455 181.7817 699.5094
## 2 535.2564 278.2290 792.2838
## 3 629.8672 369.3044 890.4301

prædiktion <- predict(modell1, interval = "prediction", level = 0.95)
ny_df <- cbind(data3, prædiktion)

ggplot(ny_df, aes(temp_mean_past1h, efterspørgsel)) +
  geom_point() +
  geom_line(aes(y=lwr), color = "red", linetype = "dashed") +
  geom_line(aes(y=upr), color = "red", linetype = "dashed") +
  geom_smooth(method=lm, se = TRUE) +
  labs(title = "Sammenhængen mellem temperatur og efterspørgsel af koldskål",
        subtitle = "Linelær regression der viser relationen mellem den forventede
        caption = "Kilde: tal fra DMI 2002 fra perioden 1/4/22-30/8/22",
        y = "Butikkernes efterspørgsel på koldskål (ltr.)",
        x = "Maks temperatur hver time (celcius)") +
  ggeasy::easy_center_title() + # Centrerer titlen.
  theme(plot.title = element_text(hjust = 0.5, size = 16),
        plot.subtitle = element_text(hjust = 0.5, size = 14),
        plot.caption = element_text(hjust = 1, face = "italic", size = 10 ))+
  xlim(4.6, 30.8) + ylim(129, 850) +
  theme_gray()

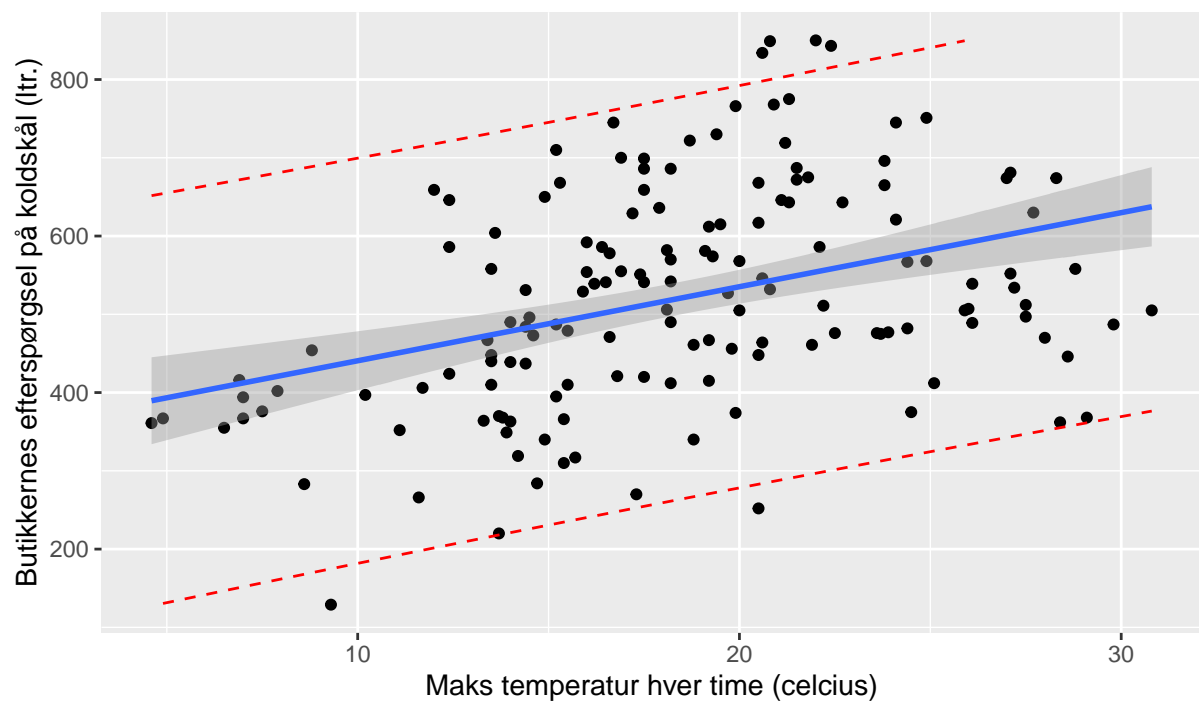
## `geom_smooth()` using formula = 'y ~ x'

```

Sammenhængen mellem temperatur og efterspørgsel af koldskål

Lineær regression der viser relationen mellem den forventede

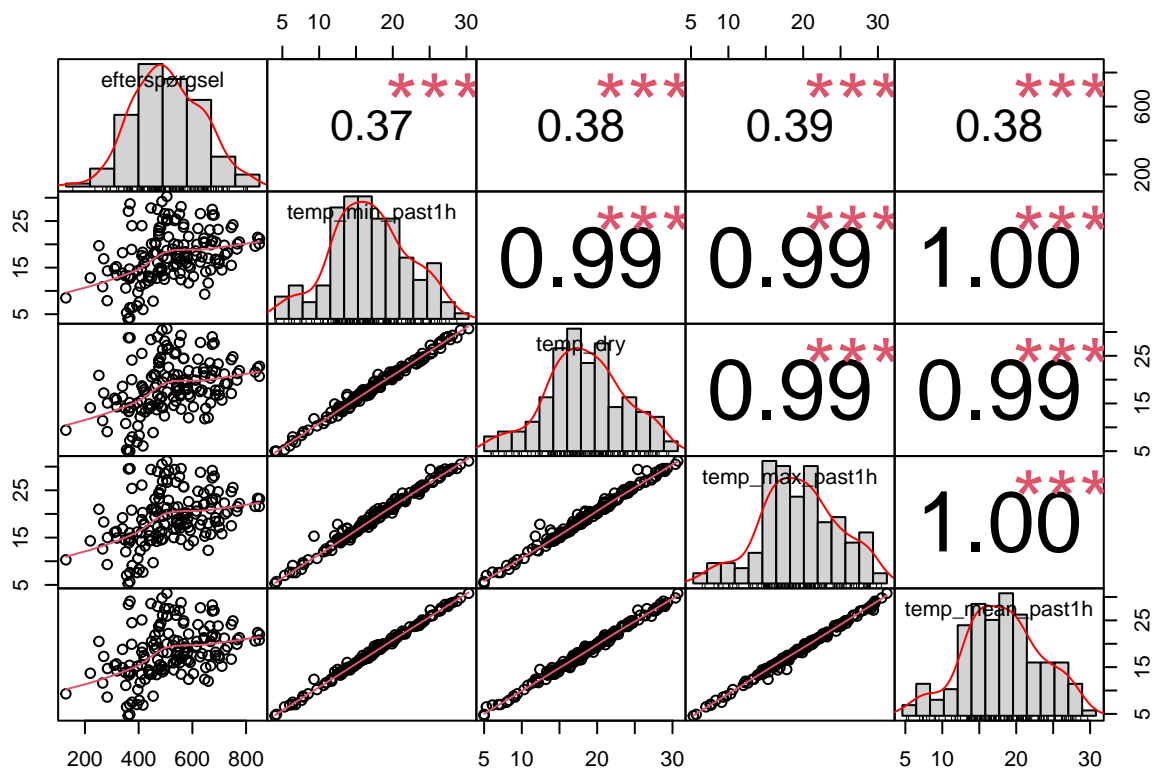
lagerbeholdning



Kilde: tal fra DMI 2002 fra perioden 1/4/22–30/8/22

I dette afsnit vil vi gå i gang med analysen.

```
# Tester for samvariation og multikolinearitet på de kontinuerte variabler.  
cor_matrice <- data3 |>  
  dplyr::select(efterspørgsel,  
                temp_min_past1h,  
                temp_dry,  
                temp_max_past1h,  
chart.Correlation(cor_matrice, histogram = TRUE, method = "pearson")
```



Der er stærk multikolinearitet, det kan være et problem ift. tolkningen af
vores multiple regressionsmodel. Dette har også en negativ indvirkning på
modellens pålidelighed.

```
lm.fit1 = lm(efterspørgsel ~ forvent_lager + weekend_helligdag + kamjunk + temp_gt25_3_dage)
vif(lm.fit1) # VIF > 1 indikerer at der er inflation i variansen på alle variabler.
```

```
##              GVIF Df GVIF^(1/(2*Df))
## forvent_lager    1.403145  2      1.088368
## weekend_helligdag  1.153516  1      1.074018
## kamjunk          1.131247  1      1.063601
## temp_gt25_3_dage  1.243248  1      1.115010
## I(temp_mean_past1h^1) 1.486803  1      1.219345
```

```
#summary(lm.fit1)
```

```
#plot(lm.fit1)
```

R² indikerer at de uafhængige variable forklarer 58% af variansen i data.

```
lm.fit1 = lm(efterspørgsel ~ 1) # simpel model
summary(lm.fit1)
```

```
##
## Call:
## lm(formula = efterspørgsel ~ 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -391.23 -104.73  -14.23   104.77   329.77
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    520.23      11.37   45.77  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 139.7 on 150 degrees of freedom
```

```
lm.fit2 <- lm(efterspørgsel ~ poly(temp_mean_past1h, degree = 3), data = data3)
summary(lm.fit2)
```

```
##
## Call:
## lm(formula = efterspørgsel ~ poly(temp_mean_past1h, degree = 3),
##     data = data3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -336.71  -86.84   -0.81   76.66  256.84
##
## Coefficients:
```



```
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   520.225      9.827  52.938 < 2e-16 ***
## poly(temp_mean_past1h, degree = 3)1  650.242    120.756   5.385 2.81e-07 ***
## poly(temp_mean_past1h, degree = 3)2 -469.047    120.756  -3.884 0.000155 ***
## poly(temp_mean_past1h, degree = 3)3 -373.456    120.756  -3.093 0.002375 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 120.8 on 147 degrees of freedom
## Multiple R-squared:  0.2674, Adjusted R-squared:  0.2524
## F-statistic: 17.88 on 3 and 147 DF,  p-value: 6e-10
```

```
lm.fit3 = lm(efterspørgsel ~ temp_mean_past1h + temp_mean_past1h^5) # ekstrem model
summary(lm.fit3)
```

```
##
## Call:
## lm(formula = efterspørgsel ~ temp_mean_past1h + temp_mean_past1h^5)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -305.02  -92.77  -11.28   84.39  306.18
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    346.035     36.288   9.536 < 2e-16 ***
## temp_mean_past1h  9.461       1.886   5.017 1.48e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 129.6 on 149 degrees of freedom
## Multiple R-squared:  0.1445, Adjusted R-squared:  0.1388
## F-statistic: 25.17 on 1 and 149 DF,  p-value: 1.476e-06
```

```
#predict(model1,data.frame(temp_mean_past1h), interval = "prediction", level = 0.95)
```

```
glm.fit1 = glm(efterspørgsel ~ 1) # simpel model  
summary(lm.fit1)
```

```
##
```

```
## Call:
```

```
## lm(formula = efterspørgsel ~ 1)
```

```
##
```

```
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max  
## -391.23 -104.73  -14.23   104.77   329.77
```

```
##
```

```
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)  
## (Intercept)    520.23      11.37   45.77  <2e-16 ***  
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
```

```
## Residual standard error: 139.7 on 150 degrees of freedom
```

```
glimpse(data3)
```

```
## Rows: 151
```

```
## Columns: 19
```

```
## $ date          <dtm> 2022-04-01, 2022-04-02, 2022-04-03, 2022-04-05, 202~  
## $ måned         <fct> april, april, april, april, april, april, april, apr~  
## $ dag           <fct> fredag, lørdag, søndag, tirsdag, onsdag, torsdag, fr~  
## $ efterspørgsel  <dbl> 367, 361, 376, 367, 402, 416, 355, 283, 454, 129, 39~  
## $ kamjunk       <fct> ja, ja, ja, ja, ja, ja, ja, nej, ja, nej, ja, ja, ja~  
## $ forvent_lager  <fct> høj, høj, høj, høj, høj, høj, høj, høj, høj, høj, høj, hø~  
## $ weekend_helligdag <fct> 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0~
```

```
## $ time          <time> 12:00:00, 12:00:00, 12:00:00, 12:00:00, 12:00:00, 1~
## $ temp_min_past1h <dbl> 4.2, 4.0, 6.4, 6.4, 7.6, 6.0, 5.3, 7.9, 7.8, 8.5, 9.~
## $ humidity       <dbl> 36, 36, 38, 44, 92, 94, 66, 44, 44, 47, 38, 53, 72, ~
## $ temp_dry       <dbl> 5.0, 5.0, 8.0, 7.2, 8.3, 6.1, 5.3, 9.1, 8.9, 9.4, 10~
## $ temp_dew       <dbl> -8.8, -8.9, -5.4, -4.2, 7.2, 5.2, -0.6, -2.4, -2.6, ~
## $ temp_max_past1h <dbl> 5.7, 5.3, 9.1, 7.6, 8.3, 7.7, 7.0, 9.3, 9.7, 10.3, 1~
## $ humidity_past1h <dbl> 38, 37, 41, 45, 94, 91, 59, 45, 47, 49, 37, 52, 74, ~
## $ temp_mean_past1h <dbl> 4.9, 4.6, 7.5, 7.0, 7.9, 6.9, 6.5, 8.6, 8.8, 9.3, 10~
## $ temp1          <dbl> 0.0, 5.7, 5.3, 4.0, 7.6, 8.3, 7.7, 7.0, 9.3, 9.7, 10~
## $ temp2          <dbl> 0.0, 0.0, 5.7, 9.1, 4.0, 7.6, 8.3, 7.7, 7.0, 9.3, 9.~
## $ temp3          <dbl> 0.0, 0.0, 0.0, 5.3, 9.1, 4.0, 7.6, 8.3, 7.7, 7.0, 9.~
## $ temp_gt25_3_dage <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

```
glm.fit2 <- glm(efterspørgsel ~ + forvent_lager + weekend_helligdag + kamjunk + temp_
cv.err3 <- cv.glm(data3, glm.fit2) # Mellem.
cv.err3$delta[[1]]
```

```
## [1] 7841.251
```

```
summary(glm.fit2)
```

```
##
```

```
## Call:
```

```
## glm(formula = efterspørgsel ~ +forvent_lager + weekend_helligdag +
```

```
##      kamjunk + temp_gt25_3_dage + måned + temp_mean_past1h, data = data3)
```

```
##
```

```
## Deviance Residuals:
```

```
##      Min      1Q   Median      3Q      Max
## -217.464  -57.822    6.436   62.564   202.505
```

```
##
```

```
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      379.923     40.788   9.314 2.35e-16 ***
```

```
## forvent_lagermellem -76.565      19.482   -3.930 0.000133 ***
## forvent_lagerhøj    -82.672      22.579   -3.661 0.000355 ***
## weekend_helligdag1   113.010      15.007    7.530 5.66e-12 ***
## kamjunknej         -71.890      17.507   -4.106 6.80e-05 ***
## temp_gt25_3_dage    -81.992      34.237   -2.395 0.017951 *
## månedmaj           84.369       24.541    3.438 0.000773 ***
## månedjuni          129.512      30.656    4.225 4.29e-05 ***
## månedjuli          155.947      32.790    4.756 4.85e-06 ***
## månedaugust         85.101       34.768    2.448 0.015616 *
## temp_mean_past1h     4.249        2.096    2.028 0.044475 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 7269.941)
##
##      Null deviance: 2925858  on 150  degrees of freedom
## Residual deviance: 1017792  on 140  degrees of freedom
## AIC: 1783.7
##
## Number of Fisher Scoring iterations: 2
```

```
glm.fit3 <- glm(efterspørgsel ~ + forvent_lager + weekend_helligdag + kamjunk + temp_
cv.err3 <- cv.glm(data3, glm.fit3) # Kompleks model overfitter, da MSE stiger.
cv.err3$delta[[1]]
```

```
## [1] 8021.179
```

```
summary(glm.fit3)
```

```
##
## Call:
## glm(formula = efterspørgsel ~ +forvent_lager + weekend_helligdag +
##      kamjunk + temp_gt25_3_dage + måned + I(temp_mean_past1h^3),
```

```
##      data = data3)
##
## Deviance Residuals:
##      Min        1Q      Median        3Q        Max
## -221.256   -56.598     6.328    62.528   200.854
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      429.119609   31.672090   13.549 < 2e-16 ***
## forvent_lagermellem -76.537449   19.725445   -3.880 0.000160 ***
## forvent_lagerhøj    -85.329967   22.818829   -3.739 0.000268 ***
## weekend_helligdag1   110.172053   15.161385    7.267 2.36e-11 ***
## kamjunknej         -74.260703   17.719015   -4.191 4.89e-05 ***
## temp_gt25_3_dage    -75.688080   34.908239   -2.168 0.031833 *
## månedmaj           99.704921   23.368071    4.267 3.63e-05 ***
## månedjuni          154.527876   27.871703    5.544 1.42e-07 ***
## månedjuli           185.559489   29.127768    6.371 2.53e-09 ***
## månedaugust         117.611689   30.962316    3.799 0.000216 ***
## I(temp_mean_past1h^3) 0.001376   0.001584    0.869 0.386471
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 7443.348)
##
##      Null deviance: 2925858  on 150  degrees of freedom
## Residual deviance: 1042069  on 140  degrees of freedom
## AIC: 1787.3
##
## Number of Fisher Scoring iterations: 2
```

```
x <- data3$temp_mean_past1h
y <- data3$efterspørgsel
```

```

data <- data.frame(y, x)

ggplot(data3, mapping = aes(x=x, y=y)) +
  geom_point(alpha=1/3) +
  geom_smooth(method="glm", formula = y ~ poly(x, 1, raw=TRUE), se=FALSE, colour="blue",
  geom_smooth(method="glm", formula = y ~ poly(x, 3, raw=TRUE), se=FALSE, colour="green",
  geom_smooth(method="glm", formula = y ~ poly(x, 22, raw=TRUE), se=FALSE, colour="red",
  geom_point(data=data3, mapping = aes(x=x, y=y), alpha=1/3) +
  labs(title = "Sammenligning af tre regressionsmodeller",
  caption = "Kilde: Tal fra DMI 2002 fra perioden 1/4/22-30/8/22",
  y = "Butikkernes efterspørgsel på koldskål (Liter)",
  x = "Maks temperatur pr.time (Celcius)") +
  ggeasy::easy_center_title() + # Centrerer titlen.
  theme( plot.title = element_text(hjust = 0.5, size = 16),
  plot.subtitle = element_text(hjust = 0.5, size = 14),
  plot.caption = element_text(hjust = 1, face = "italic", size = 10 ))+
  xlim(5, 31) + ylim(220, 900) +
  theme_gray()

```

Tidy

Transformer

Visualiser

Model

Kommunikér/analyse

Sessioninformation

For at højne reproducerbarheden printes der en udskrift om den nuværende R session:

```
SI <- sessionInfo(package = NULL) # Udskriver en liste om denne R session.
```

Litteratur

Bilag