Statistical learning og programmering

1. Semesterprojekt. Antal ord:

Af Kenneth Gottfredsen, Eva Rauff og Sanne Sørensen

2022-12-21

Indhold

I første omgang 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")
```

Import

I første omgang vil vi importere det datasæt vi har fået udleveret til eksamen:

```
# 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" ...
```

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() funktionen 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 standardindstillingerne 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 variablen '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 da 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 efterspørgselen af koldskål i form af en histogram.

Vi bruger geom_density til at forstå fordelingen og til at forudsige fordelingen af koldskål i en anden undersøgelse.

Man kan se at fordelingen er størst omkring 500 liter koldskål.

```
boxplot.stats(data3$efterspørgsel)$out
```

```
## [1] 47
```

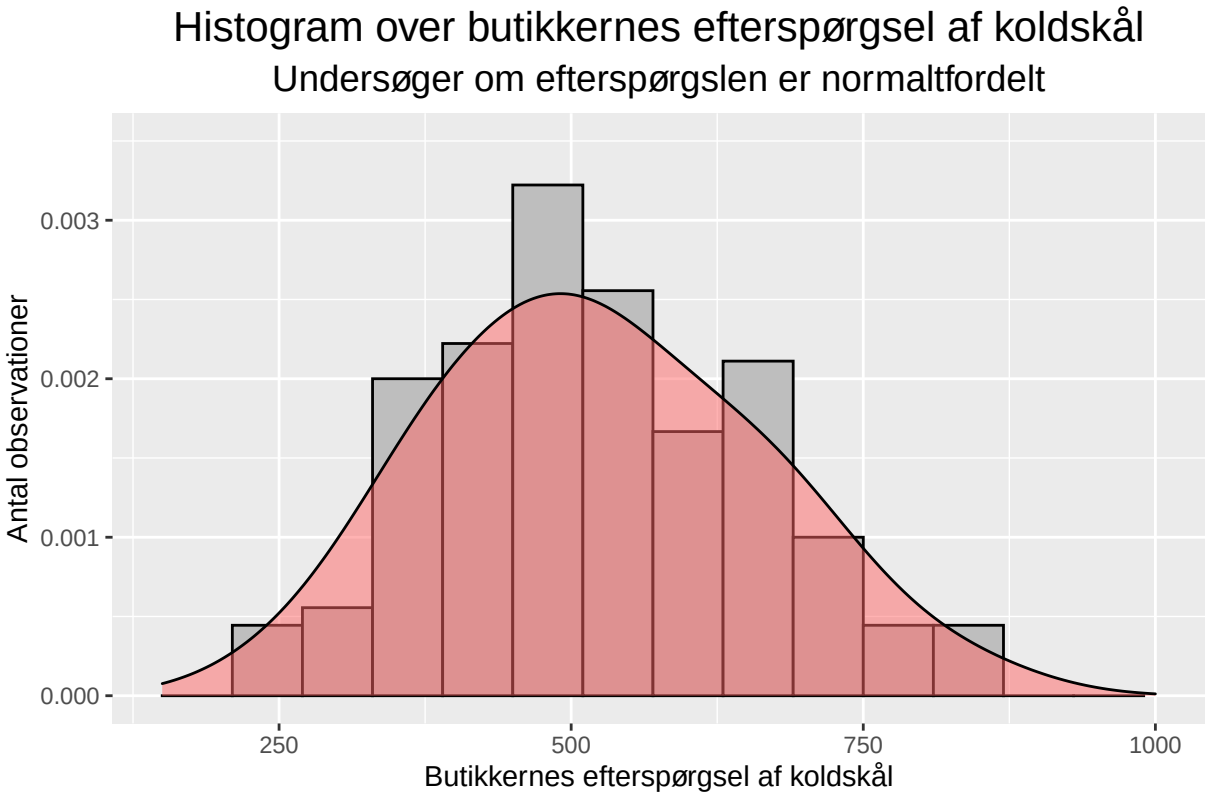
```
data3 <- data3 %>%
  filter(efterspørgsel > 47)


ggplot(data3, aes(x = efterspørgsel)) +
 geom_histogram(aes(y =..density..), colour = "black",
                fill = "gray", binwidth = 60) +
 geom_density(alpha=0.5, fill="#FF6666", adjust=1.5) +
```

```r
  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(150, 1000) + ylim(0, 0.0035)
```

```
## Warning: Removed 1 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 1 rows containing non-finite values (stat_density).
```

```
## Warning: Removed 1 rows containing missing values (geom_bar).
```

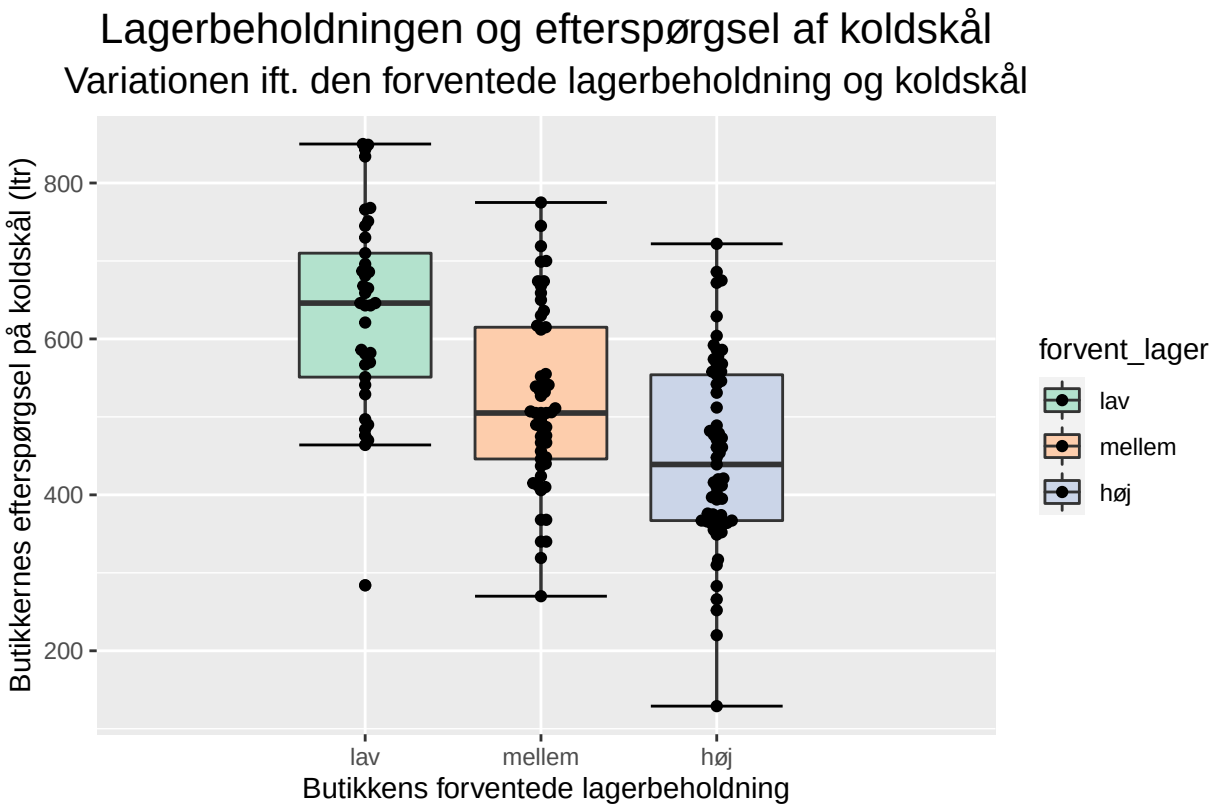


På følgende kode-chunk har vi lavet et boxplot til at vise den statistiske variationen af lagerbeholdningen og efterspørgsel.

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 = 1.3, face = "italic",size = 10 )) +
  scale_fill_brewer(palette = "Pastel2")
```

```
## Warning: position_dodge requires non-overlapping x intervals
```

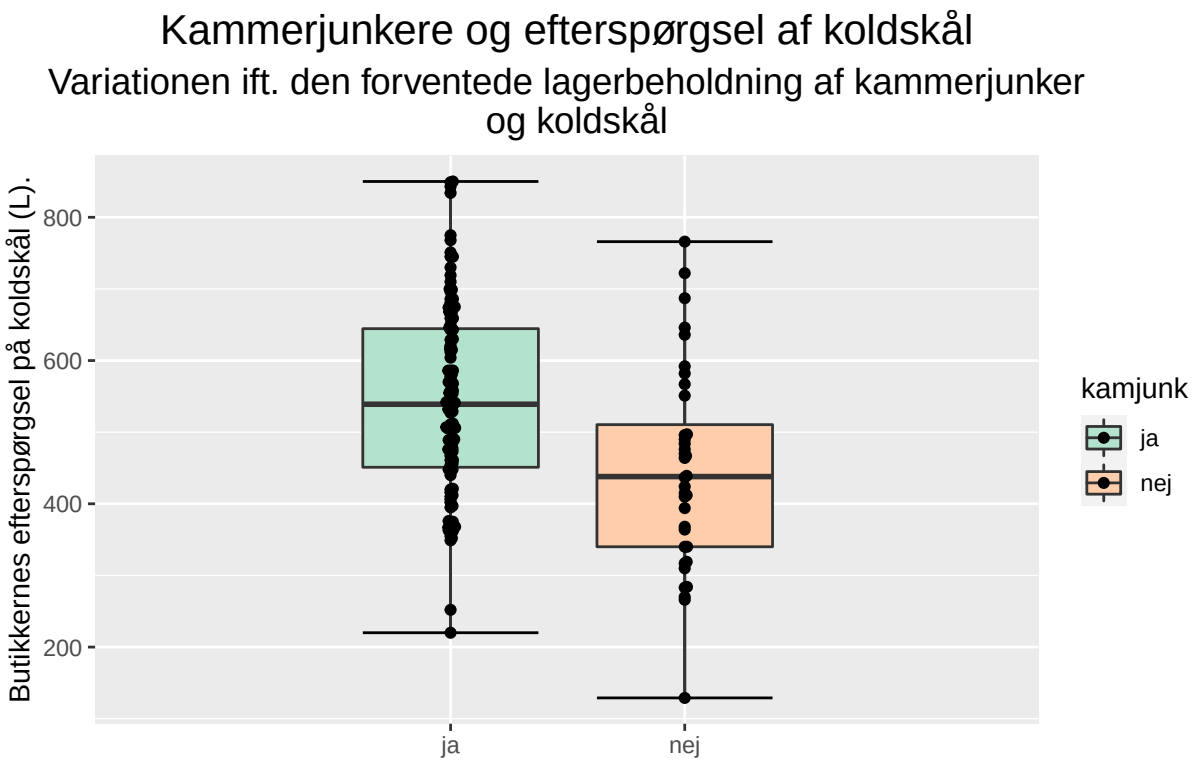


I næste kode-chunk har vi lavet et boxplot som viser fordelingen af efterspørgslen i forhold til om 25% af butikkerne er løbet tør for kammerjunkere eller ej. På baggrund af plottet kan vi se at hvis butikkerne ikke har kammerjunkere på lageret så falder efterspørgslen. Konklusionen er at efterspørgslen 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 kammerjunker
  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") +
  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 = 1.5, face = "italic",
                                     size = 10 )) +
  scale_fill_brewer(palette = "Pastel2")
```

```
## Warning: position_dodge requires non-overlapping x intervals
```

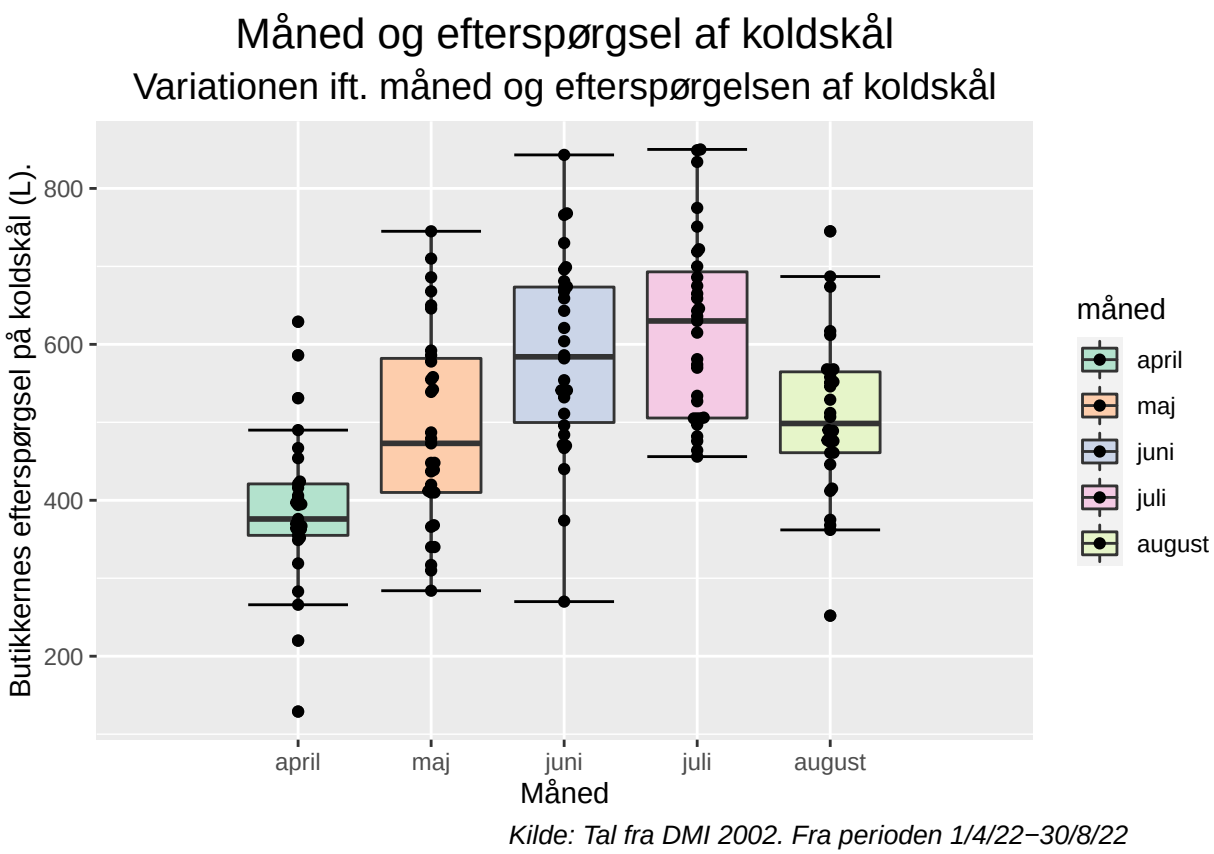


Forneden har vi lavet et boxplot som viser sammenhængen mellem måned og efterspørgslen af koldskål. Det er tydeligt at se at efterspørgslen stiger fra april-juli og derefter falder efterspørgslen i august. Hvilket kan tyde på at efterspørgselen af koldskål hænger sammen med sommerperioden.

```
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")
```

```
## Warning: position_dodge requires non-overlapping x intervals
```



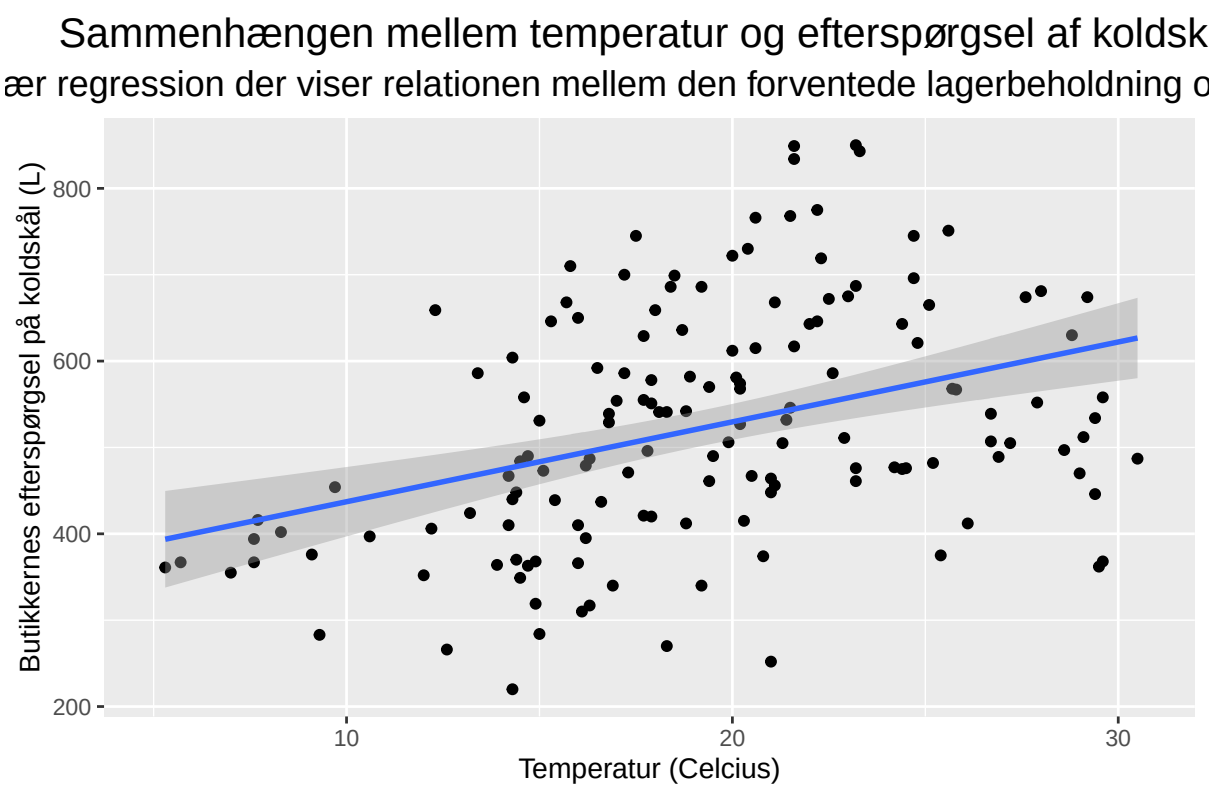

```
# Basic scatter plot.
ggplot(data3, aes(x = temp_max_past1h, y = efterspørgsel)) +
geom_point() +
```

```
geom_smooth(method = lm, se = TRUE) +
labs(title = "Sammenhængen mellem temperatur og efterspørgsel af koldskål",
subtitle = "Linelær regression der viser relationen mellem den forventede lagerbehold
caption = "Kilde: tal fra DMI 2002 fra perioden 1/4/22-30/8/22",
y = "Butikkernes efterspørgsel på koldskål (L)",
x = "Temperatur (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, 30.70) + ylim(220, 850) +
  scale_fill_brewer(palette = "Pastel2")
```

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

```
## Warning: Removed 2 rows containing non-finite values (stat_smooth).
```

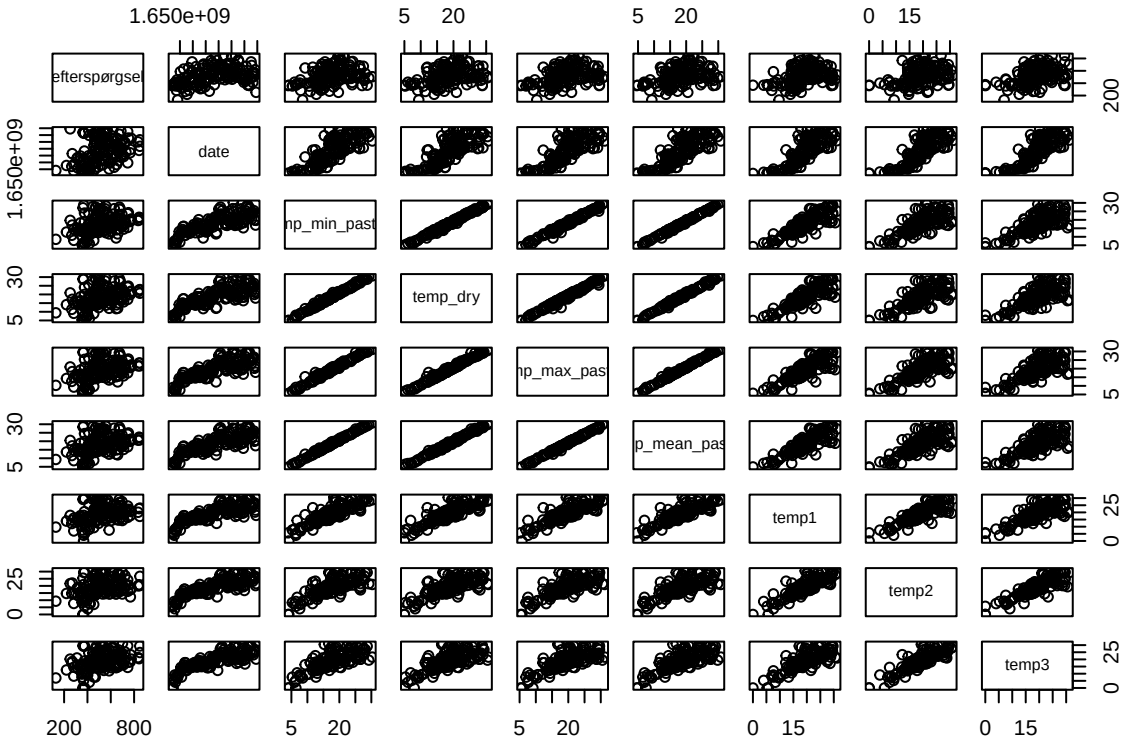
```
## Warning: Removed 2 rows containing missing values (geom_point).
```



I dette afsnit vil vi gå i gang med analysen.

```
# LOOCV metoden er mindre biased end validation set metoden; hver gang vi fit'er en
```

```
cor_matrice <- data3 |>  dplyr::select(efterspørgsel, date, temp_min_past1h, temp_dry
plot(cor_matrice)
```




```
# Vi finder den model med den mindste MSE.
# n-fold=LOOVC
```

```
attach(data3)
lm.fit1 = lm(efterspørgsel ~ forvent_lager + weekend_helligdag + kamjunk + temp_gt25_
summary(lm.fit1)
```

```
##
```

```
## Call:
## lm(formula = efterspørgsel ~ forvent_lager + weekend_helligdag +
##     kamjunk + temp_gt25_3_dage + dag + temp_dry + temp_mean_past1h +
##     humidity_past1h + temp_max_past1h)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -222.897  -56.155   -1.378   63.201  203.761
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          336.4846    69.5848   4.836 3.56e-06 ***
## forvent_lagermellem -107.9360    21.3912  -5.046 1.43e-06 ***
## forvent_lagerhøj    -142.7371    25.6662  -5.561 1.38e-07 ***
## weekend_helligdag1    98.3544    49.2416   1.997  0.04779 *
## kamjunknej           -70.0384    26.4864  -2.644  0.00916 **
## temp_gt25_3_dage    -114.1995    37.2869  -3.063  0.00265 **
## dagtirsdag            49.6271    37.5925   1.320  0.18903
## dagonsdag             38.7076    38.6969   1.000  0.31897
## dagtorsdag            12.9434    40.7340   0.318  0.75116
## dagfredag             69.2919    56.9452   1.217  0.22580
## daglørdag             33.6285    57.8889   0.581  0.56227
## dagsøndag             16.4749    57.2125   0.288  0.77382
## temp_dry             -14.8037    13.7626  -1.076  0.28400
## temp_mean_past1h       4.1220    25.0537   0.165  0.86956
## humidity_past1h        0.6304     0.5824   1.083  0.28095
## temp_max_past1h       20.1571    17.9298   1.124  0.26291
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 91.79 on 135 degrees of freedom
## Multiple R-squared:  0.6112, Adjusted R-squared:  0.568
```

```
## F-statistic: 14.15 on 15 and 135 DF,  p-value: < 2.2e-16
```

```
lm.fit2 = lm(efterspørgsel ~ 1) # simpel model
summary(lm.fit2)
```

```
##
## 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.fit3 = lm(efterspørgsel ~ temp_max_past1h^2) # melllem model.
summary(lm.fit3)
```

```
##
## Call:
## lm(formula = efterspørgsel ~ temp_max_past1h^2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -305.64  -92.07  -13.14   83.77  306.86
##
## Coefficients:
```

```
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      336.657     37.547   8.966 1.19e-15 ***
## temp_max_past1h    9.513      1.868   5.093 1.05e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 129.3 on 149 degrees of freedom
## Multiple R-squared:  0.1483, Adjusted R-squared:  0.1426
## F-statistic: 25.94 on 1 and 149 DF,  p-value: 1.049e-06
```

```r
lm.fit4 = lm(efterspørgsel ~ temp_max_past1h + temp_max_past1h^5) # ekstrem model
summary(lm.fit4)
```

```
##
## Call:
## lm(formula = efterspørgsel ~ temp_max_past1h + temp_max_past1h^5)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -305.64  -92.07  -13.14   83.77  306.86
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      336.657     37.547   8.966 1.19e-15 ***
## temp_max_past1h    9.513      1.868   5.093 1.05e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 129.3 on 149 degrees of freedom
## Multiple R-squared:  0.1483, Adjusted R-squared:  0.1426
## F-statistic: 25.94 on 1 and 149 DF,  p-value: 1.049e-06
```

```r
x <- data3$temp_max_past1h
y <- data3$efterspørgsel
data <- data.frame(y, x)
data
```

```r
set.seed(1)
cv.error <- rep(0, 5)
for (i in 1:5) {
  glm.fit <- glm(y~poly(x , i), data = )
  cv.error[i] <- cv.glm(data , glm.fit)$delta [1]
}
cv.error
```

```
## [1] 16916.63 15619.05 14636.34 14742.51 14743.11
```

```r
cv.error1 <- rep(0, 10)
for (i in 1:10) {
  glm.fit1 <- glm(efterspørgsel ~ poly(temp_max_past1h + temp_min_past1h + humidity +
  cv.error1[i] <- cv.glm(data3, glm.fit1)$delta[1]
}
cv.error1
```

```
##  [1]   18333.77   17666.40   17948.82   17372.00   17712.58   49080.10
##  [7]  147694.46 1440160.66 5137290.16  646329.61
```

```r
summary(glm.fit1)
```

```
##
## Call:
## glm(formula = efterspørgsel ~ poly(temp_max_past1h + temp_min_past1h +
##     humidity + temp_dry + temp_dew + humidity_past1h + temp_mean_past1h,
##     i), data = data3)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -310.33   -79.00    -6.95    74.26   332.51
##
## Coefficients:
##
## (Intercept)
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
##
## (Intercept)
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
##
## (Intercept)
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
```

```
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
##
## (Intercept)
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
##
## (Intercept)
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
```

```
## poly(temp_max_past1h + temp_min_past1h + humidity + temp_dry + temp_dew + humidity
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 16444.84)
##
##     Null deviance: 2925858  on 150  degrees of freedom
## Residual deviance: 2302278  on 140  degrees of freedom
## AIC: 1907
##
## Number of Fisher Scoring iterations: 2
```

```
x <- data3$temp_max_past1h
y <- data3$efterspørgsel
data <- data.frame(y, x)
data
```

```
set.seed(1)
cv.error <- rep(0, 5)
for (i in 1:5) {
  glm.fit <- glm(y~poly(x , i), data = data)
  cv.error[i] <- cv.glm(data , glm.fit)$delta [1]
}
cv.error
```

```
## [1] 16916.63 15619.05 14636.34 14742.51 14743.11
```

```
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="blu
  geom_smooth(method="glm", formula = y ~ poly(x, 3, raw=TRUE), se=FALSE, colour="gre
  geom_smooth(method="glm", formula = y ~ poly(x, 22, raw=TRUE), se=FALSE, colour="re
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, 30.70) + ylim(220, 850) +
  scale_fill_brewer(palette = "Pastel2")
```

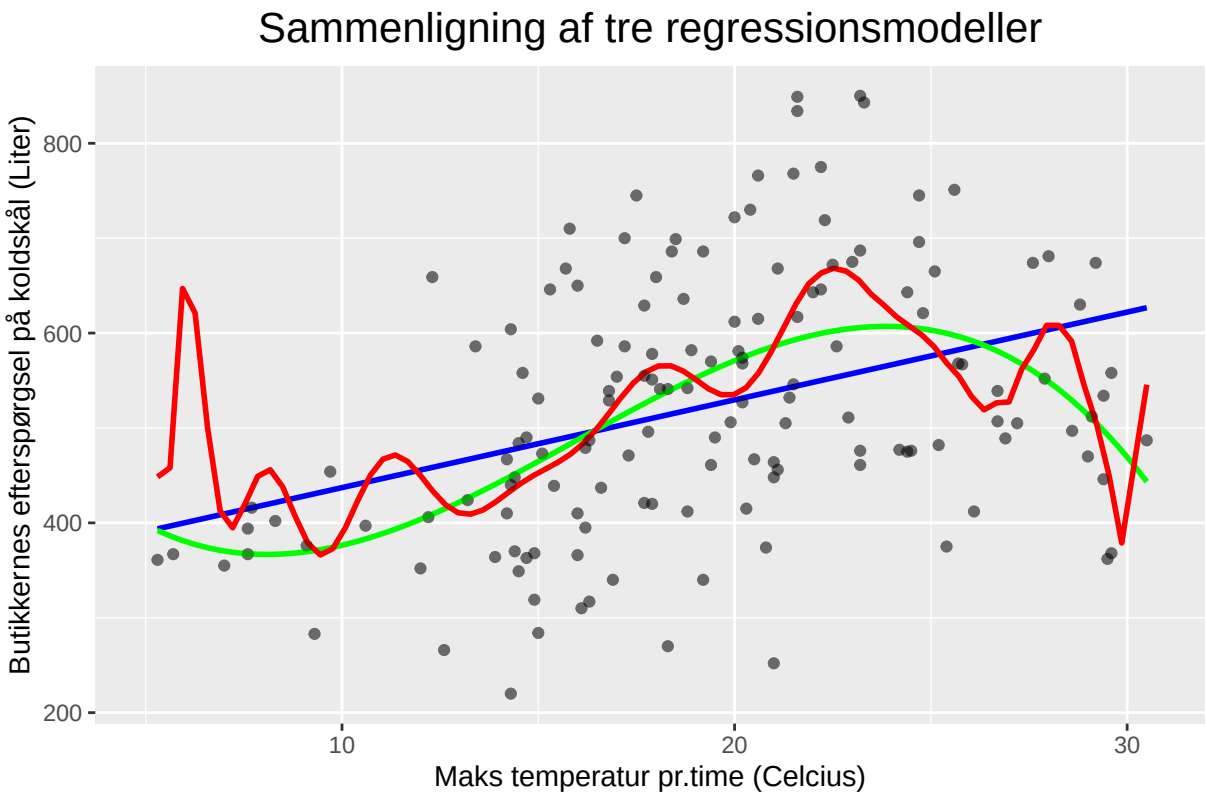
```
## Warning: Removed 2 rows containing non-finite values (stat_smooth).
## Removed 2 rows containing non-finite values (stat_smooth).
## Removed 2 rows containing non-finite values (stat_smooth).
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing missing values (geom_smooth).
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



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

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