Statistical learning og programmering

1. Semesterprojekt. Antal ord:

Af Kenneth Gottfredsen, Eva Rauff og Sanne Sørensen

2022-12-29

1 Problemfelt

Det har været et godt koldskålsår hos Thise Mejeri (Kjer 2022). Salget af koldskål er nemlig steget med 5%, sammenlignet med det forrige år (ibid). Sommervejret påvirker i stor grad kundernes efterspørgsel på koldskål (Kjer 2022, Jensen 2022, Holland 2022). Men det er vanskeligt at forudsige præcist, hvor meget koldskål Thise skal producere til deres kunder. Hos Thise bruger medarbejderne vejrudsigten og mange års erfaring, når de skal vurdere, hvor mange liter koldskål, der skal produceres fremadrettet (Jensen 2022). I en artikel fra TV Midtvest fortæller adm. direktør Poul Pedersen fra Thise Mejeri at: *"Vi kigger på tal og vejrudsigter som aldrig før. Så beslutter vi os for et tal, og vi plejer at være gode til at gætte." - (Jensen 2022).* Det er imidlertid et erhvervsøkonomisk problem, hvis man udelukkende bruger vejrudsigten og gætteri til at forudsige, hvor meget koldskål produktionsafdelingen skal producere. Gætteriet indebærer en betydelig risiko, da man ikke kan garantere, at alt koldskålen bliver solgt - selv når vejret er godt! (Jensen 2022).

Konsekvensen kan føre til et stort økonomisk tab, fordi den mængde koldskål som ikke bliver solgt i butikkerne, leveres tilbage til Thises eget lager igen (Holland 2022). Afhænger COOPs efterspørgsel på koldskål kun af vejret? Eller er der andre mekanismer end vejret, som forårsager en stigende eller faldende efterspørgsel på koldskål?

For at løse dette problem bliver undersøgelsens formål derfor at bidrage med en multibel lineær regressionsmodel, som kan give en præcis forudsigelse af, hvor mange liter koldskål COOP vil efterspørge. Den faglige vurdering er, at ovenstående problemstillinger sandsynligvis kan hænge sammen med Thises datamodenshedsproces. Det vil sige deres evne at bruge deres egne data til og skabe større økonomisk vækst i forbindelse med deres produktionsplanlægning.

Produktionsafdelingen hos Thise Mejeri mangler klarhed over, hvordan deres deres egen datamodenhedsproces hænger sammen med COOPs efterspørgsel af koldskål, samt hvilke

andre mekanismer der kan have en positiv eller negativ indflydelse. Der findes forskellige teorier på området, som kan forklare sammenhængen. Det er disse teorier og andre vejr-variable, som er emnet for denne undersøgelse, da vi vil forsøge at forstå, hvordan disse hænger sammen med COOPs efterspørgsmål på koldskål, Thises datamodenshedsproces og andre faktorer.

1.2 Problemformulering

På baggrund af ovenstående problemfeltet bliver der i næste afsnit formuleret et hovedspørgsmål og nogle underspørgsmål, de skal bruges til at besvare den erhvervsøkonomiske problemstillingen.

1.3 Hovedspørgsmål

"Hvordan kan Thises produktionsafdeling forbedre deres datamodenhed og dermed forbedre udnyttelsen af deres egne data til produktionsplanlægning?"

Hovedspørgsmålet er løsningsorienteret. Fordi spørgsmålet skal bidrage med datainitiativer som Thise selv kan benytte i sig af under produktionsplanlægningen. Fordi datainitiativerne vil forbedre deres evne til at bruge data, den selv producerer, til og skabe større økonomisk værdi.

1.4 Underspørgsmål 1

"Hvor befinder Thises produktionsafdeling sig på nuværende tidspunkt datamodenhedsmæssigt?" Underspørgsmålet skal skabe større forståelse for, hvor langt i datamodenhedsprocessen Thises produktionsafdeling befinder sig på nuværende tidspunkt.

1.5 Underspørgsmål 2

"Hvilke faktorer påvirker COOPs efterspørgsel af koldskål?" Underspørgsmålet skal beskrive hvilke andre variable der hænger sammen med efterspørgslen af koldskål.

1.6 Underspørgsmål 3

"Hvilke faktorer kan Thises produktionsafdeling bruge til at forudsige COOPs efterspørgsel af koldskål?" Underspørgsmålet skal analysere de bedste variable som produktionsafdeling skal have med i den endelige model, når de fremadrettet skal forsøge at forudsige Coops efterspørgsel af koldskål.

1.7 Underspørgsmål 4

"Kan undersøgelsens resultater anvendes til at styrke produktionsplanlægningen?" Underspørgsmålet skal perspektivere undersøgelsens statistiske analyse til en anden produktionskontekst.

2 Videnskabsteori

I dette afsnit vil vi i første omgang argumentere for det videnskabsteoretiske paradigme, vi har valgt. Dernæst vil vi redegøre og bringe begreberne ontologi og epistemologi i spil.

Et paradigme er et tankesystem (Bergfors 2021). Vores forståelse af sammenhængen mellem Thises datamodenhedsproces, vejrforholdene og efterspørgslen af koldskål er yderst kompleks. Der er behov for to forskellige former for viden om denne problemstilling. Vi har derfor valgt det realistiske paradigme, fordi vi gerne vil skabe større forståelse for, hvad der ligger bag Thises datamodenshedsproces. Vi vil også gerne kunne forklare med tal og dermed beskrive, hvorfor tallene ser ud, som de gør. Dette er, fordi vi antager, at Thises datamodenshedsproces hænger sammen med Coops efterspørgsmål på koldskål samt andre vejr-variables effekter herpå.

Med en realistisk vinkel kan man derfor anvende både kvalitative og kvantitative dataformer. Hvor datapunkter om efterspørgslen af koldskål fx består af tal og datapunkter om datamodenhedsprocessen fx omhandler det menneskelige sprog, holdninger, opfattelser. Vurderingen er, at de to former for data med fordel kan supplere hinanden i projektets analysedel. Ontologi er en antagelse om, hvordan man anskuer den verden, problemstillingen befinder sig i (Bergfors 2021). Den ontologiske opfattelse er, at virkeligheden hos Thise Mejeri eksisterer udenfor eller inden i medarbejdere som arbejder på mejeriet. Men denne virkelighed opfattes som værende uafhængig af, hvordan en medarbejder opfatter verden, hvor sand viden om koldskål og datamodenhed i større grad er objektivt. (Bergfors 2021). Dette paradigme har derfor et større fokus på helheden i form af statistiske lovmæssigheder og repræsentativitet, men konteksten og enkeltdele spiller også en rolle. Havde vi kun fokus på tal, havde vi udelukkende valgt det positivistiske paradigme. Havde vi kun haft fokus på det enkelte menneskes sprog og

opfattelser, havde vi valgt et socialkonstruktivistisk paradigme. Hvor verden udelukkende opfattes som værende en subjektiv social konstruktion, og hvor alt er under konstant forandring (ibid).

Epistemologi handler om, hvordan man anskaffer viden om en problemstilling (Bergfors 2021). Som nævnt tidligere vil vi benytte os af kvalitative og kvantitative data i samspil med hinanden, men at statistisk repræsentativitet spiller en større rolle i den her undersøgelse. Vores fokus bliver derfor på at forstå og fortolke, hvorfor tallene fremstår, som de gør, samt hvordan efterspørgslen af koldskål hænger sammen med Thises datamodenshedsproces set ud fra konteksten af produktionsafdelingen.

3 Undersøgelsesdesign

Først vil vi argumentere for, hvorfor vi har valgt et casestudie som undersøgelsesdesign. Dernæst vil vi argumentere for metodevalget. Nedenstående figur beskriver vores undersøgelsesdesign.

Et undersøgelsesdesign er en strategisk plan som bruges til at besvare en problemstilling vha. empiriske data (DeVaus 2001, Bergfors 2021). Vi anvender et statisk undersøgelsesdesign til, at besvare vores problemstilling, fordi designets natur i sig selv giver et her - og nu billede. Dertil har vi valgt et casestudie, fordi vores problemstilling tager afsæt i en nutidig kontekst (ibid). Men også fordi vi gerne vil beskrive, forklare og forstå det komplekse samspil, der er mellem Thises datamodenhedsproces, det danske sommervejr og COOPs efterspørgsel påkoldskål (ibid).

3.1 Metodologi

Metodologi henviser til de forskellige metoder, man kan bruge til at indsamle data med (ibid). Vi har valgt at anvende eksisterende kvantitative data. Formålet med den kvantitative data er, at den skal bruges i forbindelse med en regressionsanalyse. Fordi vi gerne vil kunne forklare, hvorfor forskellige typer af vejr-variable - samt andre relevante variable påvirker Coops efterspørgsel af koldskål. Vi bruger eksisterende sekundære vejrdata hentet fra Danmarks meteorologiske instituttet (DMI), og produktionsdata

om koldskål hentet fra Thises VMI-system. Regressionsanalysen bruges til at forudsige (prædikte) Coops efterspørgsel af koldskål samt andre variables effekt på denne (Se evt. variabelbeskrivelsen under bilag).

Som supplement til de kvantitative data kombineres der med nogle kvalitative data i form af to semistrukturerede interviews med to informanter. Den ene informant arbejder i produktionsafdelingen, og den anden informant arbejder i salgsafdelingen. Formålet med den kvalitative data er, at den kan bruges til og forstå, hvorfor Thises datamodenhedsproces ser ud, som den gør på nuværende tidspunkt. Med den kvalitative data kan vi derfor bevæge os dybere ned i vores problemstilling. Fremgangsmåde kaldes for metodetriangulering, idet vi vil kombinere tre forskellige datakilder til at besvare vores problemstilling med (ibid).

Den metodologiske fremgangsmåde betyder, at vi i større grad har arbejdet induktivt - dvs. hvor vi gik fra det konkrete til det generelle (ibid). Vi startede fx først ud med at lave en interviewguide og nogle spørgsmålsformuleringer, som vi operationaliserede ud fra vores hovedspørgsmål og vores underspørgsmål (se interviewguiden under bilag). Spørgsmålene blev stillet til vores informanter i to semistrukturerede interviews under vores besøg hos Thise Mejeri. Der har også været perioder, hvor vi har arbejdet deduktivt - dvs. hvor vi fik fra det generelle til det konkrete. Fx fandt vi på forhånd ud af nogle teorier om, hvorfor COOPs efterspørgsel på koldskål faldt eller steg. Det viste sig, at fx. at vejret og konkurrenternes pris på koldskål påvirkede efterspørgslen.

Den statistiske metode som anvendes i denne undersøgelse, kaldes for superviseret metode, fordi den tager udgangspunkt i én afhængig variabel. Den konkrete metode er en multibel lineær regression. Den vil vi anvende til og forudsige efterspørgslen på koldskål i liter fremadrettet på baggrund af effekten af nogle uafhængige vejr-variabler. Når vi først har trænet vores model på træningsdata og fået beregnet de nødvendige koefficienter, får alle variablerne i ligningen smidt en hat på toppen - dette indikerer prædiktion (Hastie et.al 2021). Den generelle formel bliver beskrevet nedenunder:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 ... + \hat{\beta}_p x_p + \epsilon$$

$\hat{y}$ er den forudsagte værdi af Y og $\hat{f}$ er et estimat for f. $\hat{Y}$ er desuden den afhængige variabel efterspørgsel i liter. x_i er udvalgte uafhængige variabler. $\hat{y} = f(X)$ indeholder variation som vi kan reducere ved, at bruge den korrekte SL-metode til, at beregne f med.

Men det vil aldrig være en fejlfri model vi ender ud med. Fordi estimatet ϵ er tilfældige fejl eller støj, man ikke kan gøre noget ved og den type fejl vil altid være til stede (Hastie et.al 2021).

4 Introduktion til dataanalysen og baggrund

Thise 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 køldiske 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.

Baggrund

Thise 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 køldiske 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.

Formål

Formålet med analysen er derfor, at udregne 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 efterspørgspørgslen af koldskål.

Først indlæses `pacman::load()`:

```
# Vi bruger 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", "pscl", "caret")
```

Import

I første omgang importeres 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 og transformering af datasæt

Nu har vi fået indlæst datasættet. Det næste skridt er gøre strukturen i vores dataframe nemmere at arbejde med og mere læsevenlig. Denne proces kaldes for tidy data, det betyder at hver variabel har en kolonne, hver observation en række, samt hver observationsenhed er i en tabel (Wickham 2022). Det gør analysearbejdet nemmere. Først rekode nogle af variablerne, så de stemmer overens med hvad der står i opgavebesvarelsen.

I nedestående kode-chunk rekodes og transformeres 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 kaldes data1.

Derefter bruges `mutate()` til at lave en ny kolonne ud fra data1. Først laves der en date-variabel, som bliver kodet om til et date objekt med `ymd()` fra lubridate pakken.

I den næste del anvendes mutate igen 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, navngives disse med `fct_recode()` funktionen.

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

```
## Rows: 152
## Columns: 7
## $ date              <dttm> 2022-04-01, 2022-04-02, 2022-04-03, 2022-04-04, 202~
## $ måned             <fct> april, april, april, april, april, april, april, apr~
## $ dag               <fct> fredag, lørdag, søndag, mandag, tirsdag, onsdag, tor~
## $ efterspørgsel     <dbl> 367, 361, 376, 47, 367, 402, 416, 355, 283, 454, 129~
## $ kamjunk           <fct> ja, ja, ja, nej, ja, ja, ja, ja, nej, ja, nej, ja, j~
## $ forvent_lager     <fct> høj, høj, høj, høj, høj, høj, høj, høj, høj, høj, hø~
## $ weekend_helligdag <fct> 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1~
```

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 bruges `pivot_wider()` til at fordele variablerne ud i deres egne separate kolonner.

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

`Filter(str_sub)`funktionen udvælger rækker, der indeholder de første fire karakterer: "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 bruges `left_join()` til at returnere alle rækkerne fra x, samt alle kolonner langs x og y (Wickham 2022). Udvalgte dataframes bliver sammenkoblet fra data1 og data2 til data3. Da alle kolonnerne er lige lange, optræder der ingen missing værdier.

Dernæst anvendes mutate 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 fordi der bruges `if_else`. Da vi har et begrænset antal ord og tegn med mellemrum til rådighed, vil vi ikke lave en udtømmende variabelbeskrivelse. Relevante variabler bliver beskrevet når der tolkes på modelparametre i regressionsanalysen.

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

```
## # A tibble: 152 x 15
##    date                time  weeke~1 måned dag   efter~2 kamjunk forve~3 temp_~4
##    <dttm>              <tim> <fct>   <fct> <fct>   <dbl> <fct>   <fct>     <dbl>
##  1 2022-04-01 00:00:00 12:00 1       april fred~     367 ja      høj         4.2
##  2 2022-04-02 00:00:00 12:00 1       april lørd~     361 ja      høj         4
##  3 2022-04-03 00:00:00 12:00 1       april sønd~     376 ja      høj         6.4
##  4 2022-04-04 00:00:00 12:00 0       april mand~      47 nej     høj         3.4
##  5 2022-04-05 00:00:00 12:00 0       april tirs~     367 ja      høj         6.4
##  6 2022-04-06 00:00:00 12:00 0       april onsd~     402 ja      høj         7.6
##  7 2022-04-07 00:00:00 12:00 0       april tors~     416 ja      høj         6
##  8 2022-04-08 00:00:00 12:00 1       april fred~     355 ja      høj         5.3
##  9 2022-04-09 00:00:00 12:00 1       april lørd~     283 nej     høj         7.9
## 10 2022-04-10 00:00:00 12:00 1       april sønd~     454 ja      høj         7.8
## # ... with 142 more rows, 6 more variables: humidity <dbl>, temp_dry <dbl>,
## #   temp_dew <dbl>, temp_max_past1h <dbl>, humidity_past1h <dbl>,
## #   temp_mean_past1h <dbl>, and abbreviated variable names
## #   1: weekend_helligdag, 2: efterspørgsel, 3: forvent_lager,
## #   4: temp_min_past1h
```

```r
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))
```

Datavisualisering og eksplorativ analyse

Nu er data blevet gjort tidy. Næste skridt er at undersøge hvilke vejr-mønstre der hænger sammen med butikkernes efterspørgslen af koldskål i det sammenkoblede datasæt, som vi har kaldt data3. I næste afsnit starter den eksplorative analyse.

Først identificeres potentielle outliers i vores dataset. Der fjernes 1 outlier som er 47, fordi den skiller sig væsentligt ud i forhold til de andre observationer. Måske denne er en tastefejl. Umiddelbart vurderes det ikke, at der er mange outliers i data som kan have indflydelse på den samlede varians, hvorfor der kun er fjernet den ene. Tilbage er der n = 151 i data3.

Derefter laves der et ggplot for at se fordelingen af efterspørgselen af koldskål i form af simpelt histogram, fordi efterspørgslen er en kontinuert variabel. Det er derfor muligt, at beregne spredningen mellem observationerne.

`geom_density()` funktionen bruges 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 520 liter. Endvidere kan det ses, at efterspørgslen af koldskål er tilnærmelsesvis normalfordelt, og at sandsynlighedskurven er symmetrisk klokkeformet.

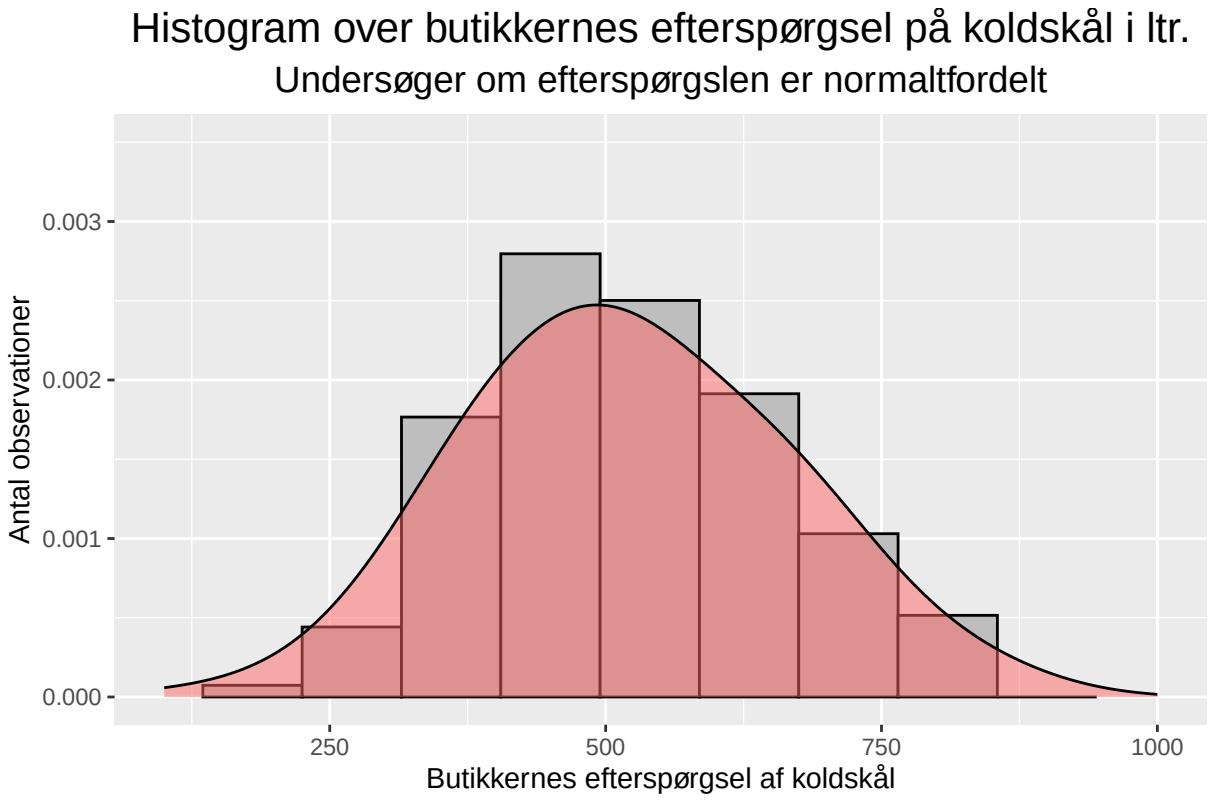
Dog kan man også se, at nogle af observationerne falder udenfor, hvilket kan skyldes tilfældig variation eller systematiske fejl. Man ved desuden, at ca. 50% af observationerne

befinder sig til venstre og højre af midtpunktet, dette er middelværdien.

At data er normalfordelt er en fordel, fordi den lineære regressionsmodel er en parametrisk test, hvor ét af kravene er at data er skal være normaltfordelt. At dette krav er opfyldt gør, at regressionsmodellens parametre er mere pålidelige.

```
data3 <- data3 %>%
  filter(efterspørgsel > 47) # Fjerner obs. 47 fra datasættet.


ggplot(data3, aes(x = efterspørgsel)) +
 geom_histogram(aes(y = ..density..), color = "black",
                fill = "grey", binwidth = 90) +
 geom_density(alpha = 0.5, fill="#FF6666", adjust = 1.6) +
  labs(title = "Histogram over butikkernes efterspørgsel på koldskål i ltr.",
  subtitle = "Undersøger om efterspørgslen er normaltfordelt",
  y = "Antal observationer",
  x = "Butikkernes efterspørgsel af koldskål",
  caption = "Kilde: Tal fra DMI 2002. Fra perioden 1/4/22-30/8/22") +
  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)
```

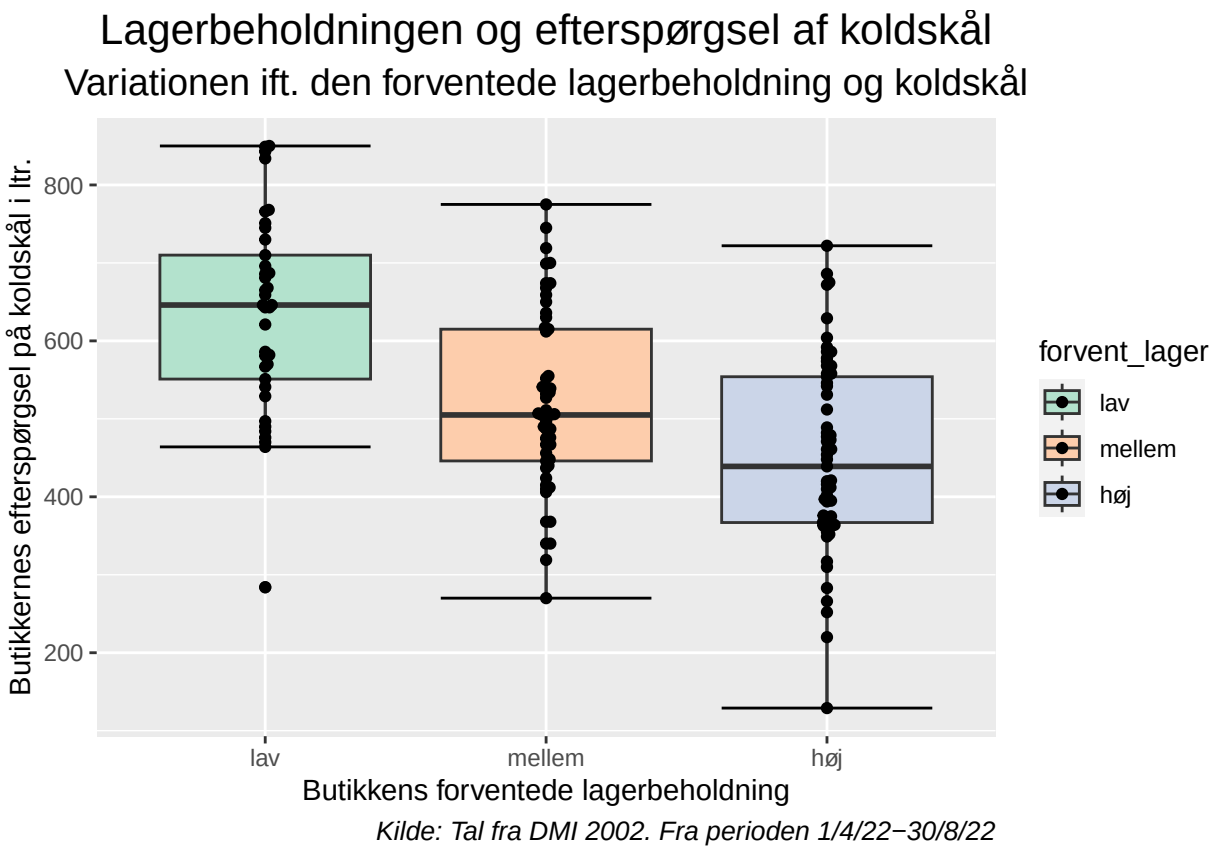


I følgende kode-chunk har er der lavet et boxplot. Det skal vise den statistiske variationen ift. butikkens forventede lagerbeholdning og efterspørgslen på koldskål.

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 signifikant sammenhæng mellem de 2 variabler. Man kan også vha. `geom = 'errorbar'` se, at der er fx. ved en høj forventet lagerbeholdning er en relativ stor usikkerhed i forhold til mellem og lav lagerbeholdning, dette indikerer at datapunkterne er forholdsvis meget spredt ud.

```
ggplot(data = data3, mapping = aes(x = forvent_lager,
                                   y = efterspørgsel,
                                   fill = forvent_lager)) +
  stat_boxplot(geom = 'errorbar') + # Viser usikkerheden.
  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 i ltr.",
```

```
x = "Butikkens forventede lagerbeholdning") +
  geom_beeswarm(dodge.width=0, cex = 0.5, color = "black") +
  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 )) +
scale_fill_brewer(palette = "Pastel2")
```

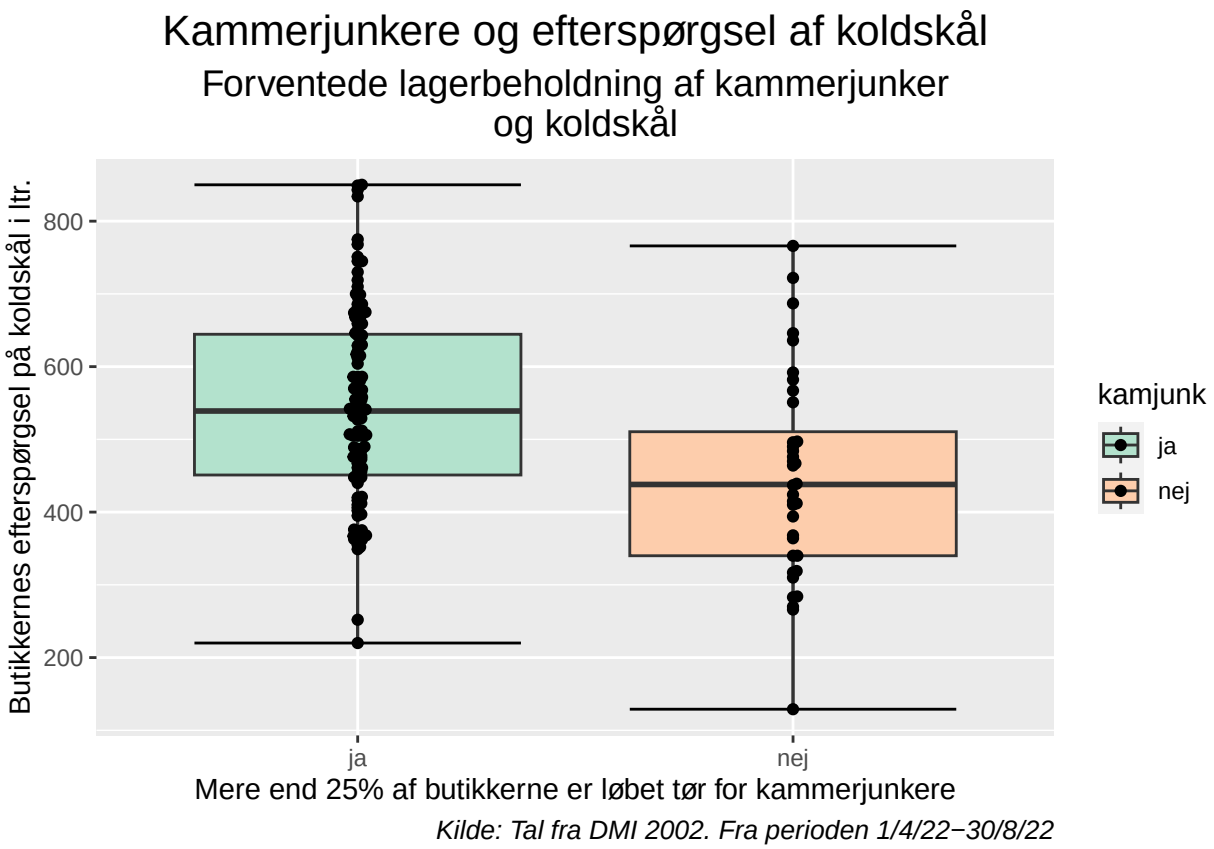


I næste kode-chunk er der lavet et boxplot som viser fordelingen af efterspørgslen i forhold til om 25% af butikkerne er løbet tør for kammerjunkere eller ej.

På baggrund af plottet kan man se at hvis butikkerne ikke har kammerjunkere på lageret så falder efterspørgslen. Det betyder at Efterspørgslen på koldskål stiger hvis butikkerne er løbet tør for kammerjunkere.

```
ggplot(data = data3, mapping = aes(x = kamjunk, y = efterspørgsel, fill =
                                     kamjunk)) +
  stat_boxplot(geom = 'errorbar') +
```

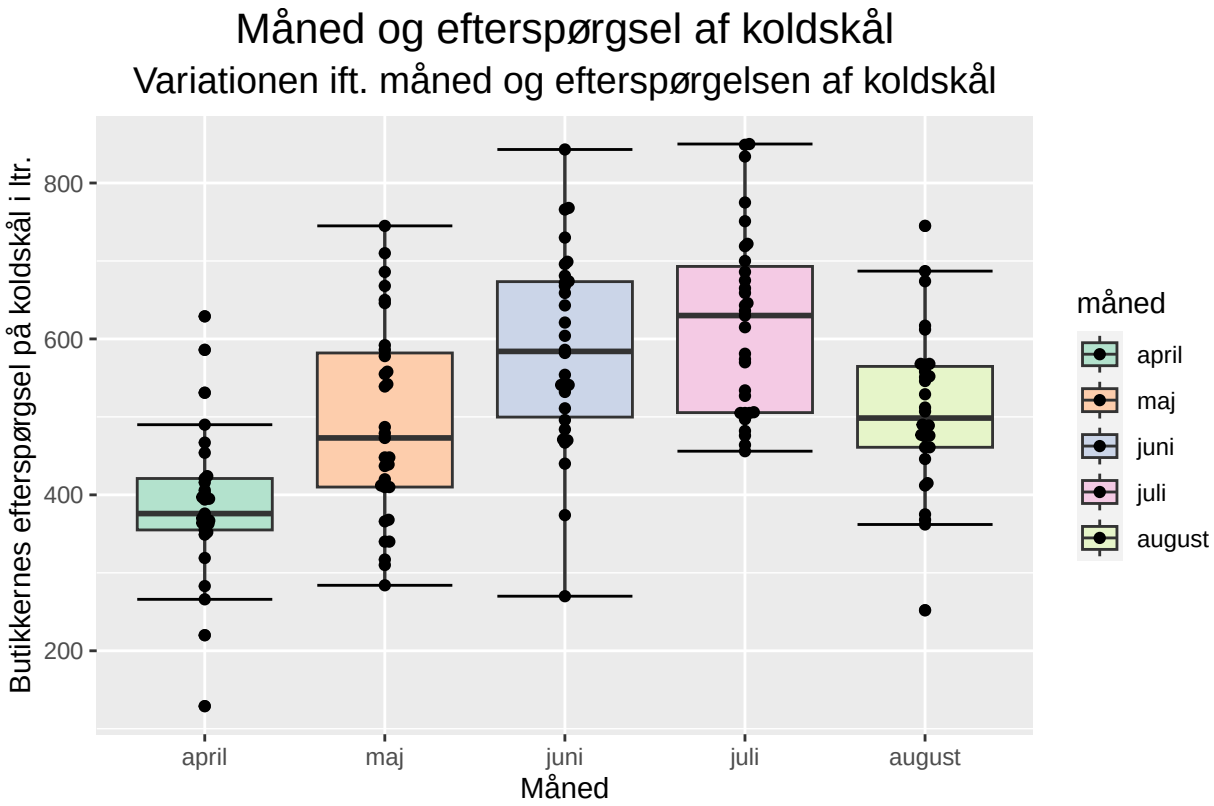
```
  geom_boxplot() +
  labs(title = "Kammerjunkere og efterspørgsel af koldskål",
  subtitle = "Forventede lagerbeholdning af kammerjunker
  og koldskål",
  caption = "Kilde: Tal fra DMI 2002. Fra perioden 1/4/22-30/8/22",
  y = "Butikkernes efterspørgsel på koldskål i ltr.",
  x = "Mere end 25% af butikkerne er løbet tør for kammerjunkere") +
  geom_beeswarm(dodge.width = 0,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, face = "italic",
                                     size = 10 )) +
scale_fill_brewer(palette = "Pastel2")
```



Forneden er et boxplot som viser sammenhængen mellem måned og efterspørgslen af koldskål. Det er tydeligt at se at median-efterspørgslen stiger fra april-juli hvorefter den efterspørgslen igen falder i august.

Den overordnede observation stemmer også overens med påstande fra vores kilder i problemfeltet, og udsagn fra vores interviews med medarbejderne hos Thise Mejeri. Dette indikerer at efterspørgselen på koldskål hænger moderat sammen med årstiden, dvs. selve sommerperioden, da hver af de forskellige boxplots ikke overlapper hinanden. I næste afsnit undersøges sammenhængen mere dybdegående.

```
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 i ltr.",
  x = "Måned") +
  ggeasy::easy_center_title() + # Centrerer titlen.
  geom_beeswarm(dodge.width = 0,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, face =
                                      "italic", size = 10 )) +
scale_fill_brewer(palette = "Pastel2")
```



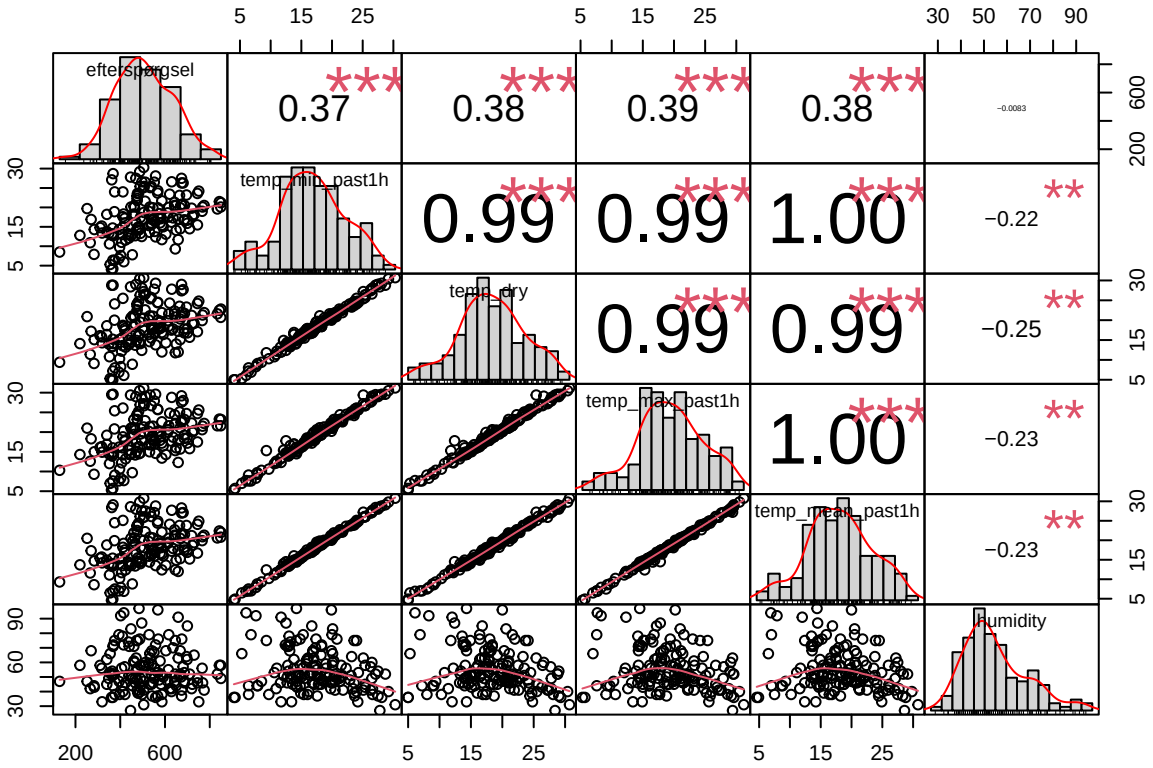
Kilde: Tal fra DMI 2002. Fra perioden 1/4/22−30/8/22

I nedestående kodechunk er der udvalgt 6 kontinuerte variabler, fordi ønskes er, at undersøge om disse korrelerer med hinanden, og om deres indbyrdes korrelation er statistisk signifikant. Der anvendes en `chart.Correlation()` til at foretage en korrelationsanalyse.

Efterspørgel og humidity er ikke korreleret med hinanden, og dermed ikke statistisk signifikant. Beslutningen er derfor, at humidity ikke vil blive inkluderet i analysen. Efterspørgsel og den gennemsnitlige temperatur per time har en korrelations koefficient på 0.38. P-værdien er meget lav med tre stjerner, det betyder at sammenhængen er signifikant, og det er derfor usandsynligt at opnå et mere ekstremt resultat, hvis man indsamlede en ny stikprøve igen og igen.

Alle temperatur-variablerne er tæt på 1, hvilket betyder at de har stærk samvariation. Dette kaldes for multikolinearitet. Det vil sige, hvis de blev brugt i den endelige model ville det være vanskeligt, at fortolke på koefficienterne. Fordi en Model med høj multikolinearitet bliver mindre præcis og mindre pålidelig. For at reducere multikolineariteten fjernes de øvrige temperatur-variable fra modellen.

```
cor_matrice <- data3 %>%
  dplyr::select(efterspørgsel,
                temp_min_past1h,
                temp_dry,
                temp_max_past1h,
                humidity)
chart.Correlation(cor_matrice, histogram = TRUE, method = "pearson")
```



Som førnævnt var der en moderat signifikant sammenhæng mellem efterspørgslen og gennemsnits temperaturen. Derfor bruges temp_mean_past1h som den uafhængige effekt i næste kodechunk. Der anvendes `predict()` til at konstruere et 95 prædiktionsinterval, efterfulgt af `geom_smooth()` til og visualisere sammenhængen med et scatterplot.

Ud fra scatterplottet kan man se, at forholdet mellem den gennemsnitlige temperatur og butikkernes efterspørgsel på koldskål er moderat lineært. Fordi hældningen på tendenslinjen er positiv. Vi antager at når gennemsnits temperaturen stiger én enhed, vil efterspørgslen stige tilsvarende, da mange af datapunkterne er tæt på linjen. Der anvendes lineær regression, fordi det er en simpel metode, hvor det er nemt og tolke på parametrene.

Men mange af datapunkterne ligger også langt væk fra tendenslinjes som udtrykker en

stigende gennemsnitlig efterspørgsel på koldskål i liter. Dette indikerer at der er stor varians og bias. En mere kompleks model skal derfor anvendes til, at forklare sammenhængen.

Efterspørgslen af koldskål bliver prædiktet ud fra den gennemsnitlige temperatur hver time i °C. Først ved 10, 20 og 30 grader. Den grå linje omkring tendenslinjen referer til konfidensintervallet af den gennemsnitlige efterspørgsel på koldskål ved en given temperatur.

Mange af observationerne er placeret udenfor dette bånd, hvorfor det er besluttet at anvende et prædiktionsinterval i stedet - der er den røde stiplede linje. Formålet er med andre ord, at indfange usikkerheden omkring de individuelle værdier og ikke usikkerheden omkring gennemsnittet.

Når den gennemsnitlige temperatur hver time er 10 °C, forudsiger vi at efterspørgslen på koldskål for én ny observation vil være 440.64 liter. Ved samme temperatur vil efterspørgslen med 95% sikkerhed være [181.78:699.51].

Når den gennemsnitlige temperatur hver time er 20 °C, forudsiger vi at efterspørgslen på koldskål for én ny observation vil være 535.25 liter. Ved samme temperatur vil butikkernes efterspørgslen af koldskål med 95% sikkerhed være [278.23:792.28].

Når den gennemsnitlige temperatur hver time er 30 °C, forudsiger vi at efterspørgslen på koldskål for én ny observation vil være 629.87 liter. Ved samme temperatur vil butikkernes efterspørgslen af koldskål med 95% sikkerhed være [369.30:890.43].

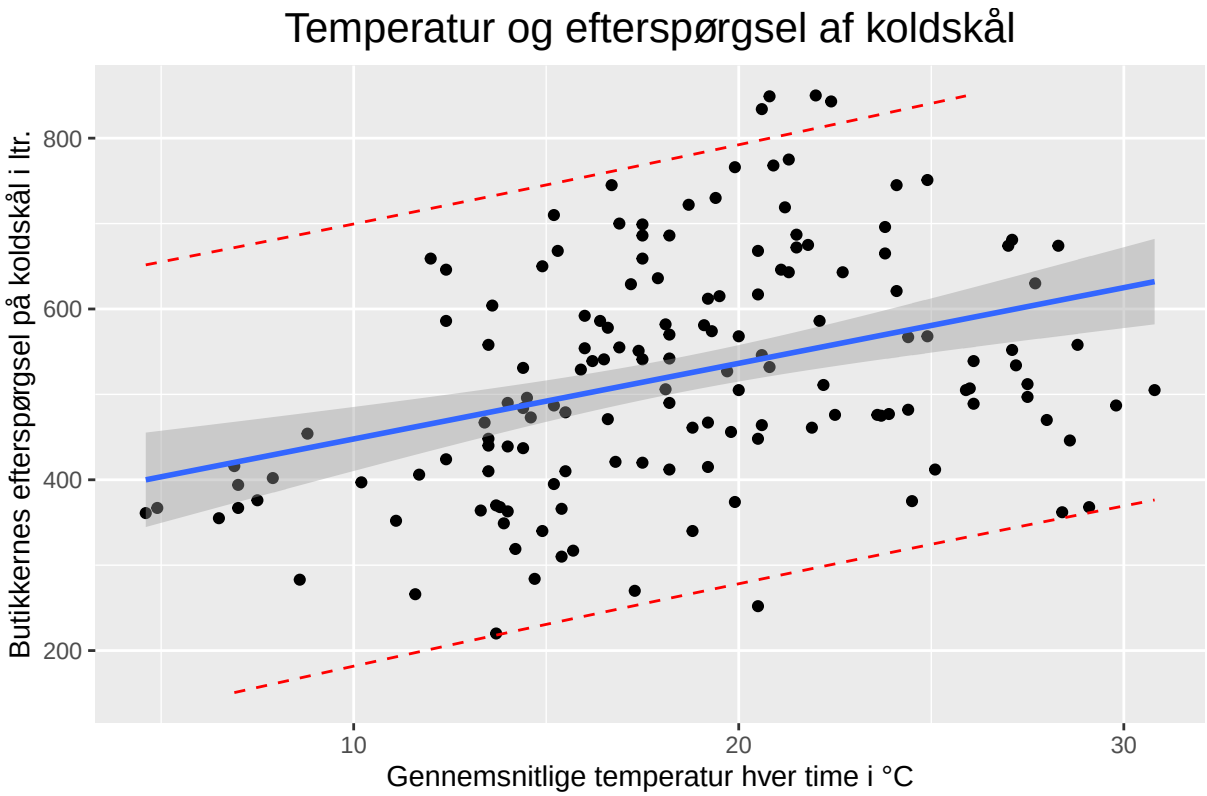
Man kan på baggrund af ovenstående tydeligt se, at hvis den gennemsnitlige temperatur i °C stiger, så stiger butikkernes efterspørgsel på koldskål tilsvarende.

```
model1 <- lm(efterspørgsel ~ temp_mean_past1h, data = data3)
predict(model1,data.frame(temp_mean_past1h = (c(10,20,30))),
        interval = "prediction", level = 0.95)
```

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

```r
prædiktion <- predict(model1, 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 = "Temperatur og efterspørgsel af koldskål",
    caption = "Kilde: tal fra DMI 2002 fra perioden 1/4/22-30/8/22",
    y = "Butikkernes efterspørgsel på koldskål i ltr.",
    x = "Gennemsnitlige temperatur hver time i °C") +
ggeasy::easy_center_title() + # Centrerer titlen.
theme( plot.title = element_text(hjust = 0.5, size = 16),
plot.subtitle = element_text(hjust = 0.5, size = 10),
plot.caption = element_text(hjust = 1, face = "italic", size = 10 )) +
xlim(4.6, 30.8) + ylim(150, 850)
```

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



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

Træning på træningsdata

I første omgang trændes modellen på træningsdata, fordi vi gerne vil tilpasse modelparametrene. Vi bruger træningsdata til, at fintune vores regressionsmodel. Når modellen er blevet trænet godt igennem, bliver den afprøvet på testdata, da vi gerne vil undersøge hvor god modellen er til, at forudsige en så præcis efterspørgsel på koldskål som mulig. Vurderingen af modelpræcisionen bestemmes ud fra den laveste MSE værdi. MSE måler hvor langt den forudsagte værdi for en observation er fra den faktiske værdi for en observation.

Er MSE lille er der den forudsagte værdi tæt på den faktiske værdi, er MSE stor er den forudsagte værdi langt fra den faktiske værdi. MSE er således et udtryk for, hvor præcis den udvalgte model er til, at forudsige efterspørgslen af koldskål (Hastie et.al 2021).

Da antallet af variabler i det samlede datasæt er mindre end antallet af observationer bruges backward-selection til, at udvælge de uafhængige variabler som fremadrettet skal indgå i modellerne. Det vil sige, at vi tilføjer alle variable ind på højre side af ligningen, og fjerner dem med den højeste p-værdi indtil der kun er signifikante uafhængige variable

tilbage (Hastie et.al 2021).

Denne teknik kan hjælpe med at reducere unødvendig varians i den udvalgte model, men på samme tid være effektiv nok til at identificere vigtige relationer i datasættet (ibid).

```
# Baseline model

lm.fit_træning <- lm(efterspørgsel ~ 1, data = data3)
lm_fit_summary <- summary(lm.fit_træning)
#mean(lm_fit_summary$residuals^2) # MSE 19376.55
rmse(lm.fit_træning, data = data3)  # RMSE = 139.19
```

```
## [1] 139.1997
```

```
#lm_fit1.1_summary

# Simpel model

lm.fit2_træning <- lm(efterspørgsel ~ temp_mean_past1h, data = data3)
lm_fit2_summary <- summary(lm.fit2_træning)
#mean(lm_fit2_summary$residuals^2) #  MSE = 16576.45
rmse(lm.fit2_træning, data = data3)  # RMSE = 128.74
```

```
## [1] 128.7495
```

```
#lm_fit2_summary

# Mellem model

lm.fit3_træning = lm(efterspørgsel ~ forvent_lager +
                       weekend_helligdag + kamjunk +
                       temp_gt25_3_dage + måned +
                       I(temp_mean_past1h), data = data3)
lm_fit3_summary <- summary(lm.fit3_træning)
```

```
#mean(lm_fit3_summary$residuals^2) # MSE = 6740.342
rmse(lm.fit3_træning, data = data3) # RMSE = 82.09959
```

```
## [1] 82.09959
```

```
#lm_fit3_summary

# Kompleks model

lm.fit4_træning = lm(efterspørgsel ~ forvent_lager +
                       weekend_helligdag +
                       kamjunk + temp_gt25_3_dage +
                       måned +
                       I(temp_mean_past1h^22), data = data3)
lm_fit4_summary <- summary(lm.fit4_træning)
#mean(lm_fit4_summary$residuals^2) # MSE = 6923.087
rmse(lm.fit4_træning, data = data3) # RMSE = 83.20509
```

```
## [1] 83.20509
```

```
#lm_fit4_summary
```

Test på testdata

I fornævnte afsnit blev den gennemsnitlige MSE beregnet for hver af de fire modeller på træningsdata. Vi er egentlig ligeglade med disse MSE værdier, det er mere interessant at se hvor præcise forudsigelserne er på testdata. Træningsdata anvendes som førnævnt til, at udvælge signifikante uafhængige variable og tilpasse modelparametrene.

Dog er det værd at nævne, at den data undersøgelsen er baseret på simulerede data. Dvs. at f er kendt allerede. Den virkelige og komplekse sandhed om efterspørgslen af koldskål vides dog ikke. Men hvis der bliver udtrukket nogle testdata ud fra data3, kan man validere hvor godt en given model performer på disse testdata, hvis modelkompleksiteten

øges. Kompleksiteten kan øges ved, at de kontinuerte variable opløftes i flere potenser, eller ved og inkludere flere uafhængige variabler i modellerne.

Valg af metode til at teste model performance

Man kan producere testdata på flere måder. Der anvendes i denne forbindelse en LOOCV metode, fordi data3 kun indeholder 151 observationer i alt. Det gode ved denne fremgangsmåde er, at den træner på alle observationerne, undtagen ét datapunkt. Processen gentages i dette tilfælde 150 gange. Derefter beregnes en gennemsnitlig MSE score, som udtrykker hvor god modelpræcisionen er (Hastie et.al 2021).

Problemet med metoden er, at det kræver stor computerkraft. Det tog denne bærbare computer ca. 2 minutter hver gang koden stumpen blev kørt. Det skyldes at modellen bliver trænet k gange (ibid).

Resultater og tolkning af den multible lineær regression

Med udgangspunkt i tabel 1 tolkes der på modelparametrene.

Baseline model

Simpel model

Mellem model

Kompleks model

	Baseline	Simpel	Mellem	Kompleks
Forvent_lager(mellem)			-76.57^{***}	-74.55^{***}
			{19.82}	{19.72}
Forvent_lager(høj)			-82.67^{***}	-87.00^{***}
			{22.58}	{22.81}

	Baseline	Simpel	Mellem	Kompleks
Weekend_helligdag(ja)			113.01***	107.33***
			{15.00}	{15.16}
Kamjunk(nej)			−71.89***	−76.52***
			{17.50}	{19.82}
Temp_gt25_3_dage			−82.00*	−66.74*
			{34.23}	{34.90}
Måned(maj)			84.37***	101.69***
			{24.54}	{23.36}
Måned(juni)			129.51***	162.71***
			{32.79}	{27.87}
Måned(juli)			155.95***	155.947***
			{32.79}	{29.12}
Måned(august)			85.10*	197.44*
			{34.76}	{30.96}
Temp_mean_past1h		9.46***	4.249*	0.0
		{1.89}	{2.07}	
Uafhængige variable	0	1	5	6
Skæring $\hat{\beta}_0$	520.23***	346.04***	379.93***	434.78***
Model P-værdi	* * *	* * *	* * *	* * *
MSE_træning	139.20	128.74	82.10	83.21
MSE_test	139.20	130.36	88.55	89.37
R^2		0.15	0.65	0.64
Obs.	150	150	150	150

Tabel 1. Summeret modelreferat fra testdata. Referencegrupper () for faktorerne er: kamjunkja, forvent_lagerlav, månedapril. {} referer til standardfejlen. Note:$* = P < 0.1; ** = P < 0.05; * * * = P < 0.01$

```
ctrl <- trainControl(method = "LOOCV") # Udvælger cross-validation metode
# Baseline model
```

```r
lm.fit_træning <- glm(efterspørgsel ~ 1, data = data3)
cv.err1 <- cv.glm(data3, lm.fit_træning)
cv.err1$delta[[1]]
```

```
## [1] 19635.76
```

```r
rmse(lm.fit_træning, data = data3) # 139.1997
```

```
## [1] 139.1997
```

```r
#lm.fit_træning <- lm(efterspørgsel ~ 1, data = data3) # Baseline model
#lm_fit1.1_summary <- summary(lm.fit_træning) # RMSE = 139.19
#summary(lm.fit_træning)

# Simpel model

model1_test <- train(efterspørgsel ~ temp_mean_past1h, data = data3,
                     method = "lm", trControl = ctrl)
model1_test # RMSE 130.36
```

```
## Linear Regression
##
## 151 samples
##   1 predictor
##
## No pre-processing
## Resampling: Leave-One-Out Cross-Validation
## Summary of sample sizes: 150, 150, 150, 150, 150, 150, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   130.3607  0.1238588  105.8906
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```r
#summary(model1_test)

# Mellem model

model2_test <- train(efterspørgsel ~ forvent_lager +
                       weekend_helligdag + måned +
                       kamjunk +
                       temp_gt25_3_dage +
                       I(temp_mean_past1h^1), data = data3,
                       method = "lm", trControl = ctrl)
model2_test # RMSE 88.55
```

```
## Linear Regression
##
## 151 samples
##   6 predictor
##
## No pre-processing
## Resampling: Leave-One-Out Cross-Validation
## Summary of sample sizes: 150, 150, 150, 150, 150, 150, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   88.55084  0.5967607  72.56083
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```r
summary(model2_test)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
```

```
##
## 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 0.000000000000000235 ***
## 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 0.000000000005658725 ***
## månedmaj                   84.369     24.541   3.438             0.000773 ***
## månedjuni                 129.512     30.656   4.225 0.000042884648209182 ***
## månedjuli                 155.947     32.790   4.756 0.000004853815232615 ***
## månedaugust                85.101     34.768   2.448             0.015616 *
## kamjunknej                -71.890     17.507  -4.106 0.000068019491738631 ***
## temp_gt25_3_dage          -81.992     34.237  -2.395             0.017951 *
## `I(temp_mean_past1h^1)`     4.249      2.096   2.028             0.044475 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 85.26 on 140 degrees of freedom
## Multiple R-squared:  0.6521, Adjusted R-squared:  0.6273
## F-statistic: 26.25 on 10 and 140 DF,  p-value: < 0.00000000000000022
```

```
# Kompleks model

model3_test <- train(efterspørgsel ~ forvent_lager +
                       weekend_helligdag +
                       kamjunk +
                       temp_gt25_3_dage +
                       måned +
                       I(temp_mean_past1h^22), data = data3,
```

```
                        method = "lm", trControl = ctrl)
model3_test # RMSE 89.37393
```

```
## Linear Regression
##
## 151 samples
##   6 predictor
##
## No pre-processing
## Resampling: Leave-One-Out Cross-Validation
## Summary of sample sizes: 150, 150, 150, 150, 150, 150, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   89.37393  0.5890984  72.52832
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
summary(model3_test)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -228.715  -55.834    9.064   60.411  195.816
##
## Coefficients:
##                                                    Estimate
## (Intercept)              434.78508352052710961288539692759513855
## forvent_lagermellem      -74.55965162064809703679202357307076454
```

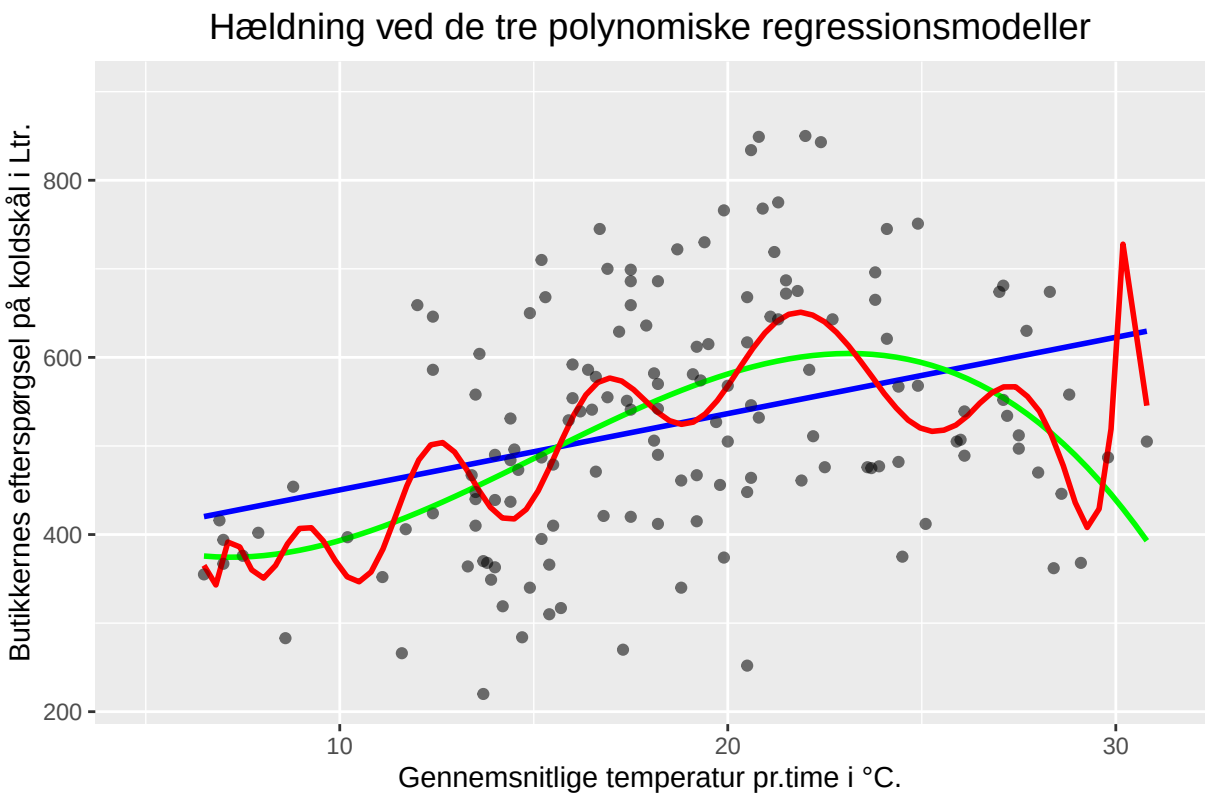
```
## forvent_lagerhøj         -87.00324055382570520578155992552638054
## weekend_helligdag1       107.33113217623082391583011485636234283
## kamjunknej               -76.52366312070434162251331144943833351
## temp_gt25_3_dage         -66.74127366335562783206114545464515686
## månedmaj                 101.69276592064437636508955620229244232
## månedjuni                162.71820081638725241646170616149902344
## månedjuli                197.44866771887388904360705055296421105
## månedaugust              132.93820936952900524374854285269975662
## `I(temp_mean_past1h^22)` -0.00000000000000000000000000000007542
##                                                      Std. Error t value
## (Intercept)               31.17559677473798984692621161229908466  13.946
## forvent_lagermellem       19.87923255498784769201847666408866644  -3.751
## forvent_lagerhøj          22.78959044924258137143624480813741684  -3.818
## weekend_helligdag1        15.11099148544141712591226678341627121   7.103
## kamjunknej                17.63033571287086331835780583787709475  -4.340
## temp_gt25_3_dage          34.29448923826949169324507238343358040  -1.946
## månedmaj                  23.25704954450009154243161901831626892   4.373
## månedjuni                 26.24779480159469713385078648570924997   6.199
## månedjuli                 26.47990487424752004130823479499667883   7.457
## månedaugust               26.59059049046405931449044146575033665   4.999
## `I(temp_mean_past1h^22)`   0.00000000000000000000000000000013588  -0.555
##                                        Pr(>|t|)
## (Intercept)              < 0.0000000000000002 ***
## forvent_lagermellem                 0.000257 ***
## forvent_lagerhøj                    0.000202 ***
## weekend_helligdag1         0.00000000005658 ***
## kamjunknej                 0.00002707391607 ***
## temp_gt25_3_dage                    0.053643 .
## månedmaj                   0.00002379661842 ***
## månedjuni                  0.00000000598338 ***
## månedjuli                  0.00000000000845 ***
## månedaugust                0.00000169161621 ***
```

```
## `I(temp_mean_past1h^22)`              0.579766
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 86.41 on 140 degrees of freedom
## Multiple R-squared:  0.6427, Adjusted R-squared:  0.6172
## F-statistic: 25.18 on 10 and 140 DF,  p-value: < 0.00000000000000022
```

```
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 = "Hældning ved de tre polynomiske regressionsmodeller",
  caption = "Kilde: Tal fra DMI 2002 fra perioden 1/4/22-30/8/22",
y = "Butikkernes efterspørgsel på koldskål i Ltr.",
x = "Gennemsnitlige temperatur pr.time i °C.") +
  ggeasy::easy_center_title() + # Centrerer titlen.
```

```
  theme( plot.title = element_text(hjust = 0.5, size = 14),
  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)
```



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

Vurdering af modellens performance

På baggrund af vores MSE værdier, har vi som sammenlignings grundlag opstillet et histogram der visuelt viser, hvordan MSE værdierne bliver mindre i takt med at modelkompleksiteten stiger.

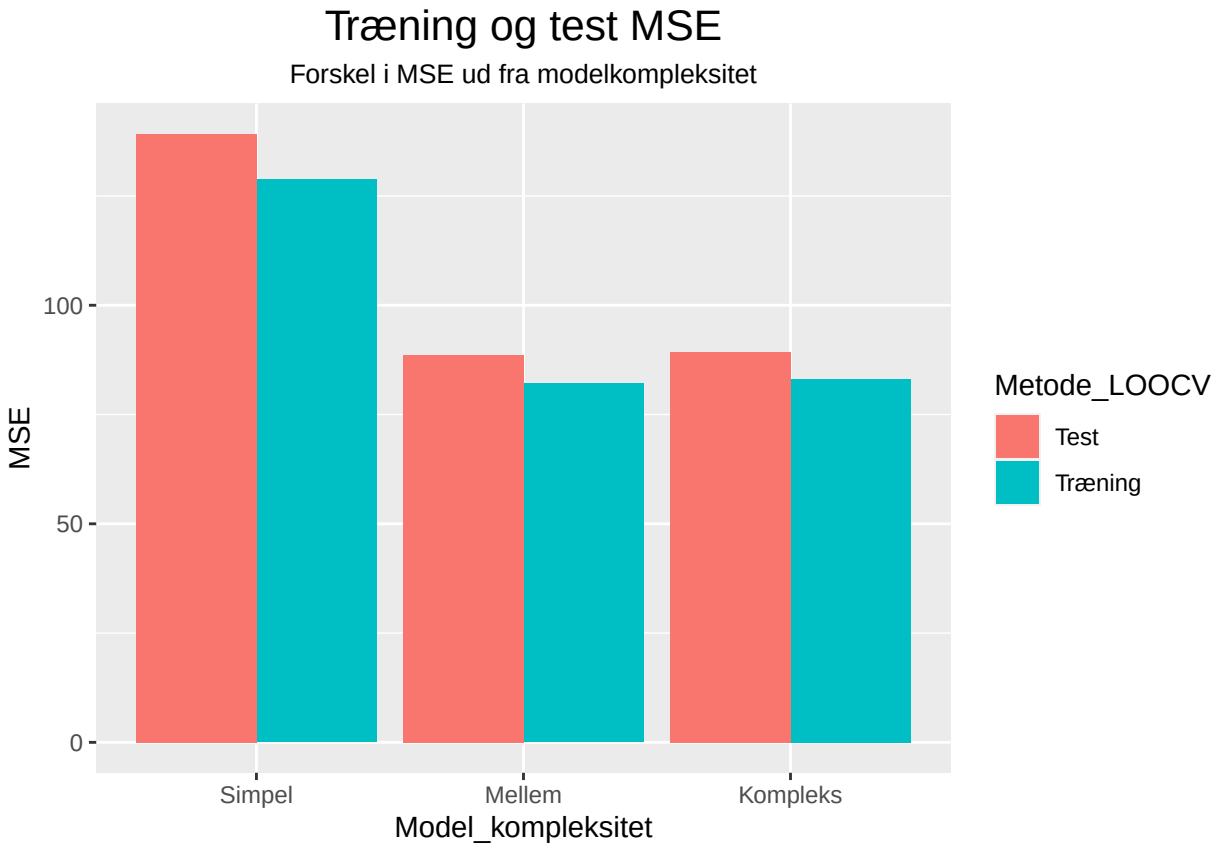
```
Metode_LOOCV = c("Træning", "Træning", "Træning",
                 "Test", "Test", "Test")
Model_kompleksitet = c("Simpel", "Mellem", "Kompleks",
                       "Simpel", "Mellem", "Kompleks")
MSE = c(128.7495, 82.09959, 83.20509, 139.1997, 88.55084, 89.37393)
test_frame = data.frame(Metode_LOOCV, Model_kompleksitet, MSE,
```

```
                              stringsAsFactors = TRUE)
test_frame$Model_kompleksitet <- factor(test_frame$Model_kompleksitet,
                        levels = c("Simpel", "Mellem", "Kompleks"))


test_frame
```

```
##   Metode_LOOCV Model_kompleksitet       MSE
## 1      Træning              Simpel 128.74950
## 2      Træning              Mellem  82.09959
## 3      Træning            Kompleks  83.20509
## 4         Test              Simpel 139.19970
## 5         Test              Mellem  88.55084
## 6         Test            Kompleks  89.37393
```

```
MSE_plot <- ggplot(test_frame) +
 geom_bar(aes(x = Model_kompleksitet, y = MSE, fill=Metode_LOOCV),
 stat = "identity", # Ikke transformere data.
 position = "dodge") +
 labs(title = "Træning og test MSE",
 subtitle = "Forskel i MSE ud fra modelkompleksitet") +
 ggeasy::easy_center_title() +
 theme( plot.title = element_text(hjust = 0.5, size = 16),
 plot.subtitle = element_text(hjust = 0.5, size = 10),
 plot.caption = element_text(hjust = 1, face = "italic", size = 10 ))
MSE_plot
```




```
#test_frame
#str(test_frame)
#levels(test_frame$Kompleksitet)
#attributes(test_frame)
#glimpse(test_frame)
```

Resultater

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Konkurrerende modeller

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Modeludvælgelse

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Endelig model

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Overordnet præcision i forudsigelse

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Observeret og prædiktet værdier af målvariablen

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of

the document itself, and is automatically re-run whenever the document is rendered.

Forbedring i forhold til en baseline model (ingen model)

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Prædiktion med selv-konstruerede forklarende variabler

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Diskussion

Vurdering af modellens performance

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Vurdering af bidraget til løsningen af forretningsproblemet

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Udrulning af anbefalingerne

Konklusion

Quarto supports executable code blocks within markdown allowing you to create fully reproducible documents and reports. The code required to produce your output is part of the document itself, and is automatically re-run whenever the document is rendered.

Sessioninformation

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

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

```
## R version 4.2.1 (2022-06-23)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur ... 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRblas.0.dyli
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRlapack.dyli
##
## locale:
## [1] da_DK.UTF-8/da_DK.UTF-8/da_DK.UTF-8/C/da_DK.UTF-8/da_DK.UTF-8
##
## attached base packages:
## [1] splines   stats     graphics  grDevices utils     datasets  methods
## [8] base
##
```

```
## other attached packages:
##  [1] pscl_1.5.5                car_3.1-0
##  [3] carData_3.0-5             PerformanceAnalytics_2.0.4
##  [5] xts_0.12.2                zoo_1.8-11
##  [7] writexl_1.4.1             openxlsx_4.2.5
##  [9] boot_1.3-28.1             RColorBrewer_1.1-3
## [11] palmerpenguins_0.1.1      ggbeeswarm_0.6.0
## [13] tinytex_0.41              timeDate_4021.104
## [15] readxl_1.4.1              viridis_0.6.2
## [17] viridisLite_0.4.0         fueleconomy_1.0.0
## [19] nasaweather_0.1           babynames_1.0.1
## [21] class_7.3-20              RSQLite_2.2.16
## [23] caret_6.0-93              lattice_0.20-45
## [25] leaps_3.1                 testthat_3.1.5
## [27] MASS_7.3-58.1             ISLR2_1.3-1
## [29] microbenchmark_1.4.9      jsonlite_1.8.0
## [31] httr_1.4.4                XML_3.99-0.10
## [33] modelr_0.1.9              pander_0.6.5
## [35] broom_1.0.1               htmlwidgets_1.5.4
## [37] feather_0.3.5             hexbin_1.28.2
## [39] hms_1.1.2                 pryr_0.1.5
## [41] lubridate_1.8.0           maps_3.4.1
## [43] Lahman_10.0-1             gapminder_0.3.0
## [45] nycflights13_1.0.2        magrittr_2.0.3
## [47] forcats_0.5.2             stringr_1.4.1
## [49] dplyr_1.0.10              purrr_0.3.4
## [51] readr_2.1.2               tidyr_1.2.0
## [53] tibble_3.1.8              ggplot2_3.4.0
## [55] tidyverse_1.3.2
##
## loaded via a namespace (and not attached):
##  [1] backports_1.4.1      plyr_1.8.7           listenv_0.8.0
```

```
##  [4] digest_0.6.29        foreach_1.5.2        htmltools_0.5.3
##  [7] fansi_1.0.3          memoise_2.0.1        googlesheets4_1.0.1
## [10] tzdb_0.3.0           recipes_1.0.1        globals_0.16.1
## [13] gower_1.0.0          hardhat_1.2.0        colorspace_2.0-3
## [16] blob_1.2.3           rvest_1.0.3          haven_2.5.0
## [19] xfun_0.32            crayon_1.5.1         survival_3.3-1
## [22] iterators_1.0.14     glue_1.6.2           gtable_0.3.0
## [25] gargle_1.2.0         ipred_0.9-13         future.apply_1.9.1
## [28] abind_1.4-5          scales_1.2.1         DBI_1.1.3
## [31] Rcpp_1.0.9           bit_4.0.4            stats4_4.2.1
## [34] lava_1.6.10          prodlim_2019.11.13   ellipsis_0.3.2
## [37] farver_2.1.1         pkgconfig_2.0.3      nnet_7.3-17
## [40] dbplyr_2.2.1         utf8_1.2.2           labeling_0.4.2
## [43] tidyselect_1.1.2     rlang_1.0.6          reshape2_1.4.4
## [46] munsell_0.5.0        cellranger_1.1.0     tools_4.2.1
## [49] cachem_1.0.6         cli_3.4.1            generics_0.1.3
## [52] pacman_0.5.1         evaluate_0.16        fastmap_1.1.0
## [55] yaml_2.3.5           ModelMetrics_1.2.2.2 knitr_1.40
## [58] bit64_4.0.5          fs_1.5.2             zip_2.2.0
## [61] future_1.28.0        nlme_3.1-157         xml2_1.3.3
## [64] brio_1.1.3           compiler_4.2.1       rstudioapi_0.13
## [67] curl_4.3.2           beeswarm_0.4.0       reprex_2.0.2
## [70] stringi_1.7.8        highr_0.9            Matrix_1.4-1
## [73] vctrs_0.5.1          pillar_1.8.1         lifecycle_1.0.3
## [76] data.table_1.14.2    R6_2.5.1             gridExtra_2.3
## [79] vipor_0.4.5          parallelly_1.32.1    codetools_0.2-18
## [82] assertthat_0.2.1     withr_2.5.0          mgcv_1.8-40
## [85] parallel_4.2.1       quadprog_1.5-8       grid_4.2.1
## [88] rpart_4.1.16         rmarkdown_2.16       googledrive_2.0.0
## [91] pROC_1.18.0          ggeasy_0.1.3
```

Litteratur

Bilag