Bistader

62527 Big data E20

Armandas Rokas(s185144)

11/09 2020

Contents

Introduktion

Dette rapport beskriver databehandling af data fra et bistade. Dataen samles via Raspbarry Pi, som er tilknyttet til forskellige sensorer, som måler vægten, fugtighed, lys osv. (hele listen af variabler kan man se i Variabler afsnit), og gemmes både lokalt på SQLite databasen og sendes til HiveTool.net platformen.

Import af data

Der blev valgt at bruge SQLite til at indlæse dataen, da det er lidt svært at få fat i dataen på HiveTool.net, især hvis man skal bruge en lang tidsperiode. Så der blev skrevet en `import_hive_data` funktion, som indlæser dataen fra SQLite (Koden af funktionen kan findes i Bilag B).

Data beskrivelse

Observationer fra bistadet bliver taget hver 5. minut og der er i alt 105064 observationer (Bilag A). Dataen er dog ikke helt konsistent, da der er nogle huller i datasættet, hvor der mangler målinger, og der er nogle målinger, som har forkerte værdier. Dataen går helt tilabge til 2018-03-07, men den bliver mere og mere inkonsistente jo mere i fortiden man går, så derfor blev der valgt at tage udgangspunkt i et års data, dvs. fra 2019-09-01 til 2020-09-01.

Variabler

Der er i alt 43 variabler i datasættet, men ud fra opsummering af alle variabler i datasættet i Bilag A kan man konstatere at disse variabler kun er i brug og relevente: `hive_observation_time_local`, `hive_weight_kgs`, `hive_temp_c`, `hive_humidity`, `ambient_temp_c`, `ambient_humidity`og `ambient_luminance`. Nedenfor opsummering af disse variabler:

```
des <- Hmisc::describe(hive_data)
des$hive_observation_time_local$extremes <- NULL
print(des)
```

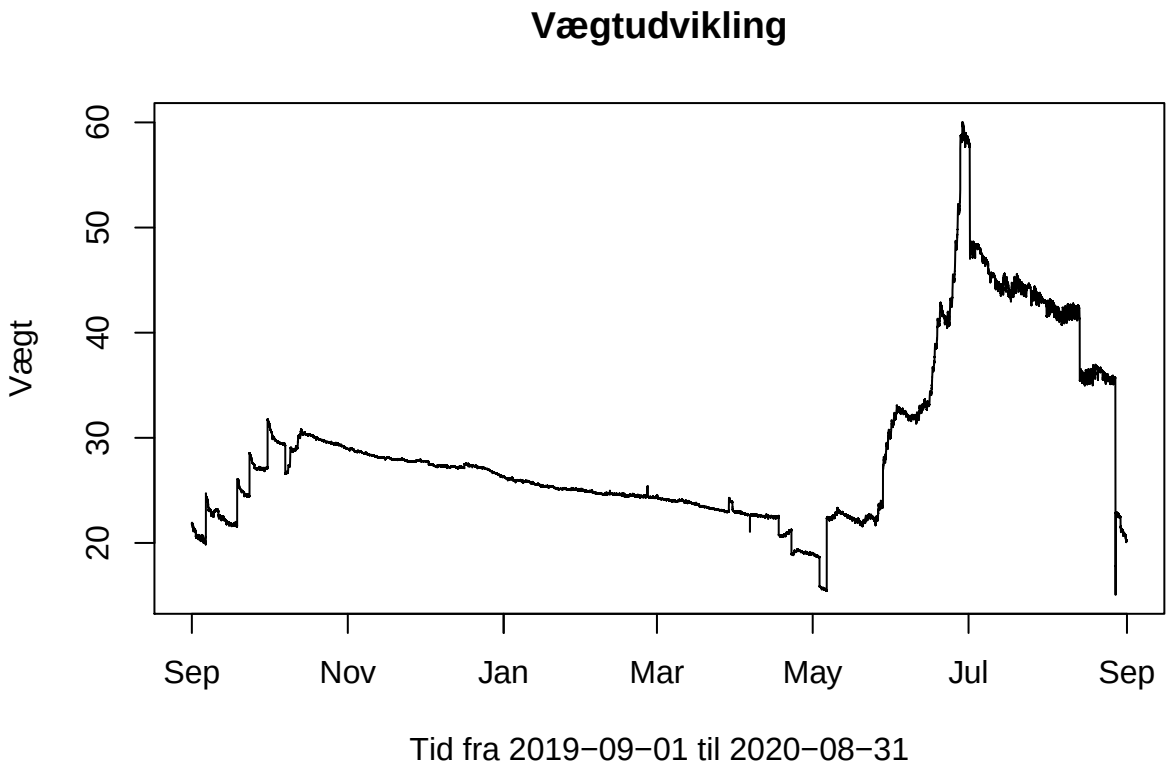
```
## hive_data
##
##  7  Variables      105064  Observations
## ---------------------------------------------------------------------------
## hive_observation_time_local
##        n  missing distinct
##   105064        0   105052
##
## ---------------------------------------------------------------------------
## hive_weight_kgs
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0     3194        1    28.94     8.23    20.68    22.21
##      .25      .50      .75      .90      .95
##    23.65    26.97    30.36    42.99    45.03
##
## lowest : 15.06 15.42 15.43 15.44 15.45, highest: 59.98 60.00 60.01 60.02 60.04
## ---------------------------------------------------------------------------
## hive_temp_c
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      350        1    25.32    9.522      9.0     13.3
```

```
##      .25      .50      .75      .90      .95
##     20.1     25.0     34.2     34.9     35.1
##
## lowest :  2.1  2.2  2.3  2.5  2.6, highest: 36.7 36.8 36.9 37.0 37.1
## ------------------------------------------------------------------------------
## hive_humidity
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      551        1     62.1    11.76    41.42    47.50
##      .25      .50      .75      .90      .95
##    57.10    60.30    67.50    79.00    83.70
##
## lowest : 36.0 36.1 36.2 36.3 36.4, highest: 90.6 90.7 90.8 90.9 91.0
## ------------------------------------------------------------------------------
## ambient_temp_c
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   101787     3277      360        1    9.496    6.932      1.0      2.7
##      .25      .50      .75      .90      .95
##      4.9      8.0     13.6     18.2     20.9
##
## lowest : -4.3 -4.2 -4.1 -4.0 -3.9, highest: 31.2 31.3 31.4 31.5 31.6
## ------------------------------------------------------------------------------
## ambient_humidity
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   101787     3277      433    0.211       99    1.724     94.2     99.9
##      .25      .50      .75      .90      .95
##     99.9     99.9     99.9     99.9     99.9
##
## lowest : 55.1 55.5 55.6 56.0 56.5, highest: 99.5 99.6 99.7 99.8 99.9
## ------------------------------------------------------------------------------
## ambient_luminance
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      669    0.809    14.65    26.32        0        0
##      .25      .50      .75      .90      .95
##        0        0        5       31       71
##
## lowest :    0    1    2    3    4, highest: 1314 1408 1449 1804 2070
## ------------------------------------------------------------------------------
```

Vægten

Vægten er den centrale variable i datasættet. For eksempel nedenfor kan man se en vægtudvikling fra i perioden fra 2019-09-01 til 2020-09-01, hvor man kan se tydeligt to sæsoner. Om vinteren vægten bliver gradvis mindre, fordi biene spiser føder, og om sommer trakker nektar ind i et bistade.

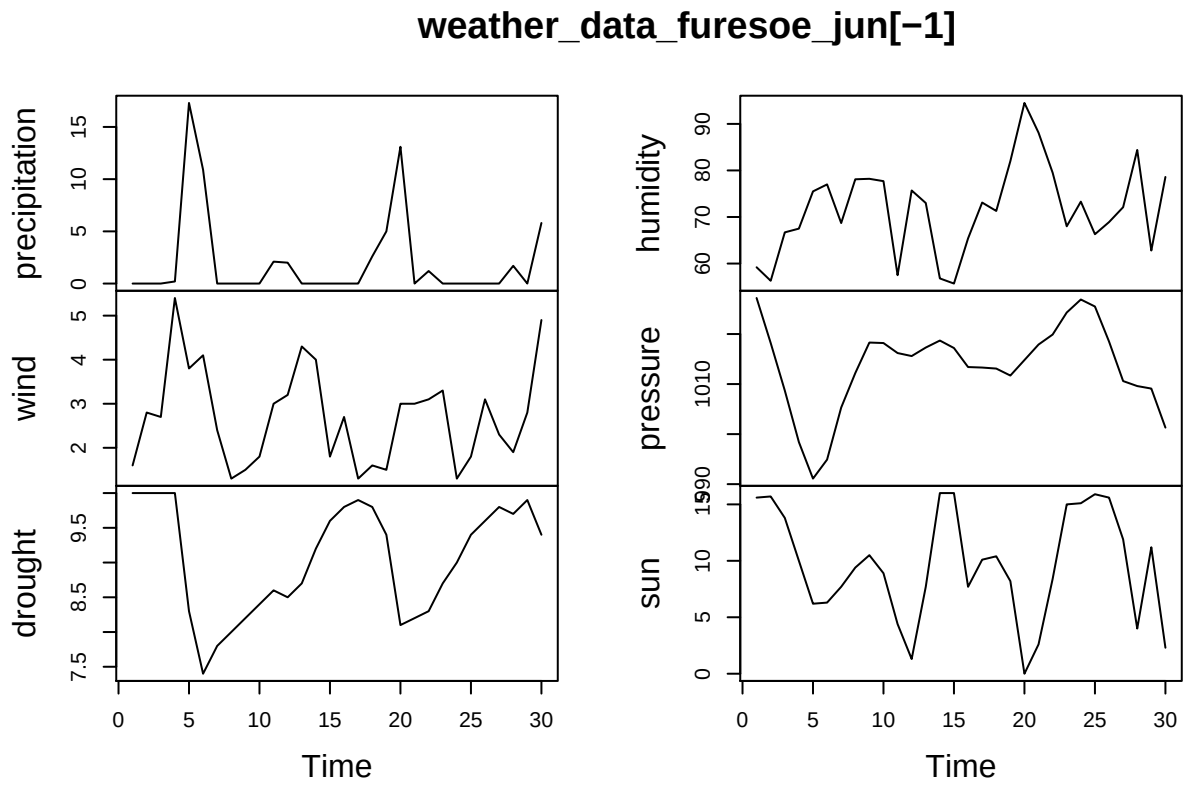
```
## Warning in result_fetch(res@ptr, n = n): Column `wx_wind_degrees`: mixed type,
## first seen values of type integer, coercing other values of type string
```



Eksternal vejr data

til at fylde op datasæt med relavent vejr data blev der benyttet https://www.dmi.dk/vejrarkiv/ hjemsside til at får nedbør, vindhastighed, tørkeindex, luftfugtighed, lufttryk, solskin timer. Nedenfor kan man see opsummering af en eksampel af vejr data i juni måned i 2020:

```
weather_data_furesoe_jun <- read.table(file="data/furesø-kommune-juni-2020.csv", sep="," ,header = TRUE)
weather_data_furesoe_jun$dt <- as.Date(weather_data_furesoe_jun$dt)
#Hmisc::describe(weather_data_furesoe_jun)
plot.ts(weather_data_furesoe_jun[-1])
```

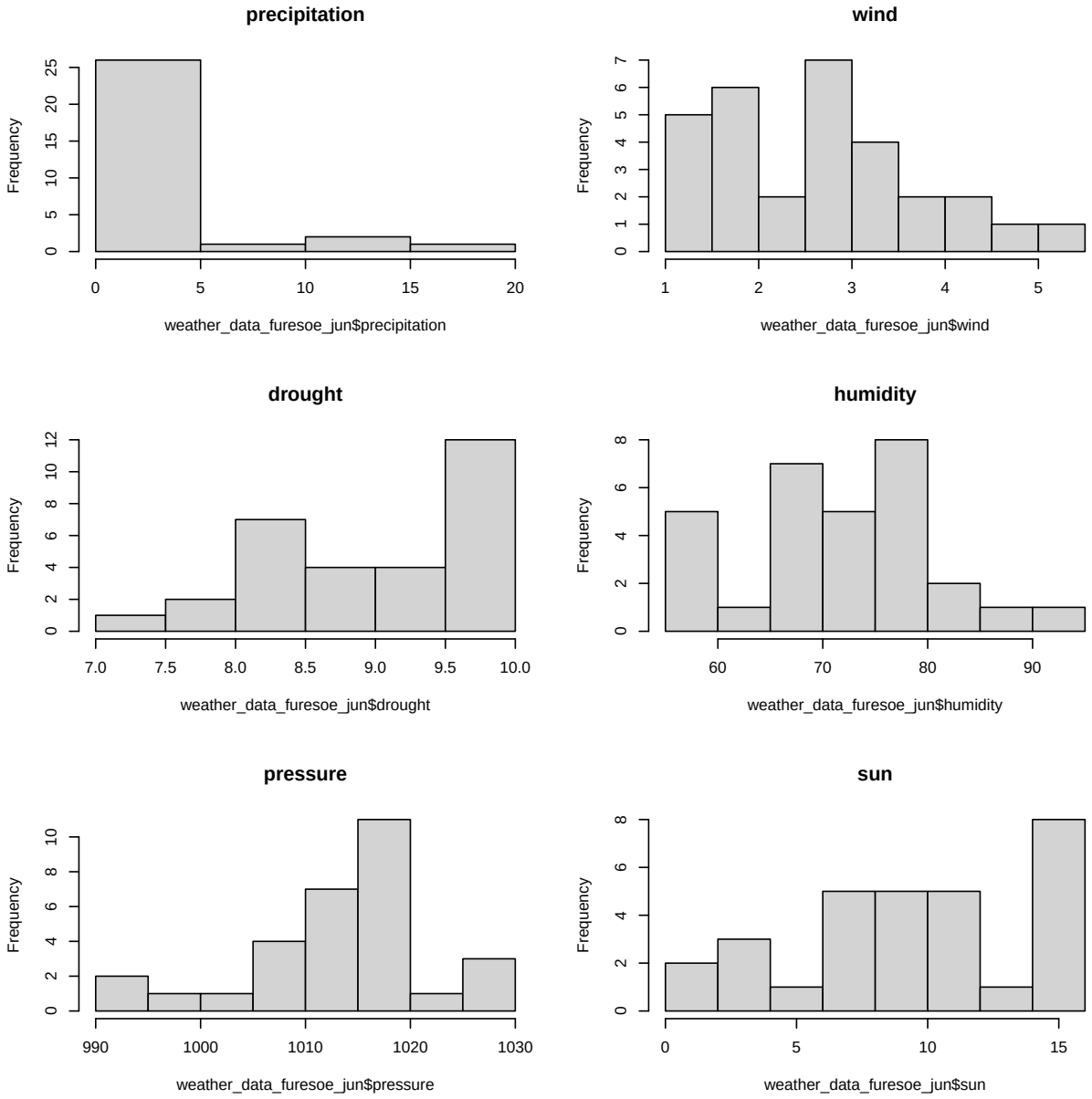


Kan det være, at jeg får en fejl, når jeg laver en modellen, fordi data er meget skewed?

Jeg har også prøvet at normalisere, men jeg får det samme...

Hvad med, hvis jeg definerer grænseværdier selv. Dvs om det regnet eller det var solen hele dag.

```
par(mfrow=c(3,2))
hist(weather_data_furesoe_jun$precipitation, main="precipitation")
hist(weather_data_furesoe_jun$wind, main="wind")
hist(weather_data_furesoe_jun$drought, main="drought")
hist(weather_data_furesoe_jun$humidity, main="humidity")
hist(weather_data_furesoe_jun$pressure, main="pressure")
hist(weather_data_furesoe_jun$sun, main="sun")
```



Dataoprensning

Der er flere årsager til at det er svært at aflæse en eksakt tilvækst af vægten. De mulige årsager kunne være:

- Manuelt indgreb
- Nedbør
- Biernes daglig rutine

Disse er beskrevet videre i afsnittet udtaget nedbør, da dette giver minimalt støj og der er ikke umiddelbart behov til at tage hensyn til det.

Manuelle indgreb

Manuelle indgreb på bistadet medfører de største udsving i vægten, men som ikke forårsagers af bierne, så disse skulle fjernes før man kunne påbegynde noget andet. Eksempler på manuelle indgreb kunne være:

- Påsætte/fjerne magasin
- Andre mindre manipulationer

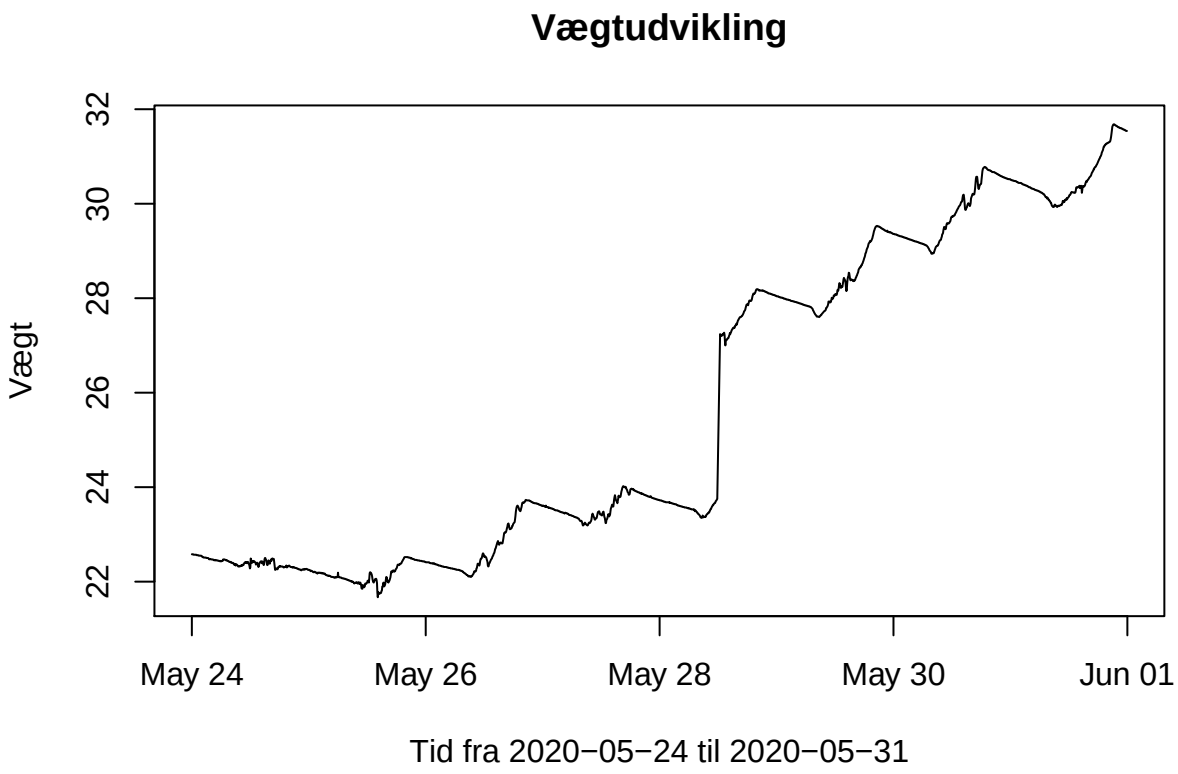
De fleste manual indgreb følge efter afbrydelser i timestamps. Det kan forklares med at en biavler slukker optagelse af målinger, når biavlen laver et manualt indgreb, men når han tænder optagelsen, laver den en stor udsving i vægten i en eller anden retning i for hold hvad han fik lavet. (Bemærk, man behøver ikke at tage hensyn til det manglende timestamps, fordi alt videre analyse er gennemført på middnats værdier, som ikke er manglende).

Så derfor blev der startede med at finde deltaerne mellem vægtene før arbrydelser i timestmaps og efter, dvs. huler i datasættet. En liste over det kan man se nedenfor:

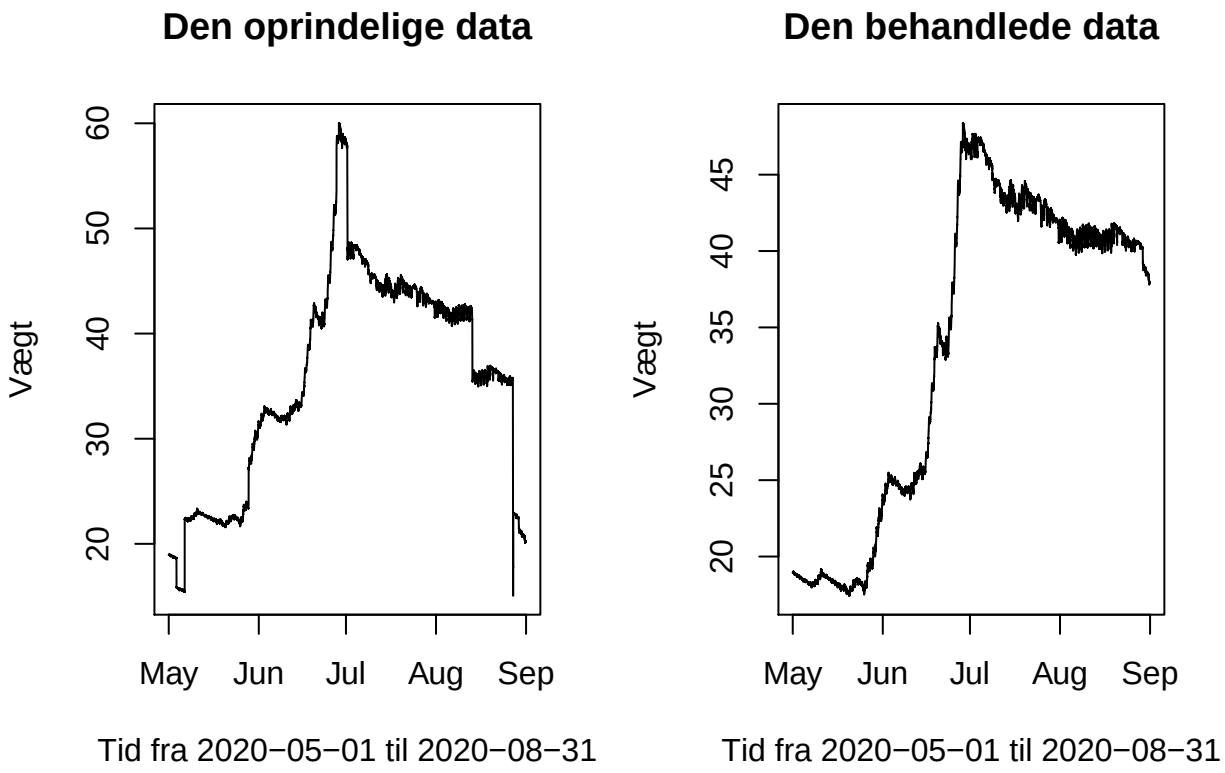
```
##        hive_observation_time_local weight_delta
## 5108           2019-09-18 18:00:01         4.48
## 6497           2019-09-23 13:55:01         4.12
## 8532           2019-09-30 15:35:01         4.60
## 10524          2019-10-07 13:50:01        -2.72
## 67473          2020-04-22 16:25:52        -2.42
## 70626          2020-05-03 16:25:01        -2.78
## 71435          2020-05-06 12:00:41         6.90
## 77757          2020-05-28 12:25:01         3.49
## 86420          2020-06-27 18:15:01         3.89
## 87477          2020-07-01 13:30:01       -10.38
## 99794          2020-08-13 12:50:01        -5.90
## 103794         2020-08-27 13:05:02       -20.61
## 103795         2020-08-27 13:15:56         2.74
## 103805         2020-08-27 14:15:01         5.08
```

- Man kan se at ikke alle huler følger efter en stor udsving i vægten, men det kunne være en mulig løsning til at fjerne alligevel alle, hvis man har lyst til at automatisere databehandling. Den anden mulighed, som er ønsket af projektstilleren at have mulighed at definere selv perioder, som skal ignoreres.

- For at visualisere problemet bedre blev der valgt at plotte vægten videre omkring `2020-05-28 12:25:01`, hvor vægten har øget med `3.49`, fordi der blev sat en nye magasin ind. Nedenfor er grafen over vægten omkring dette tidspunkt.

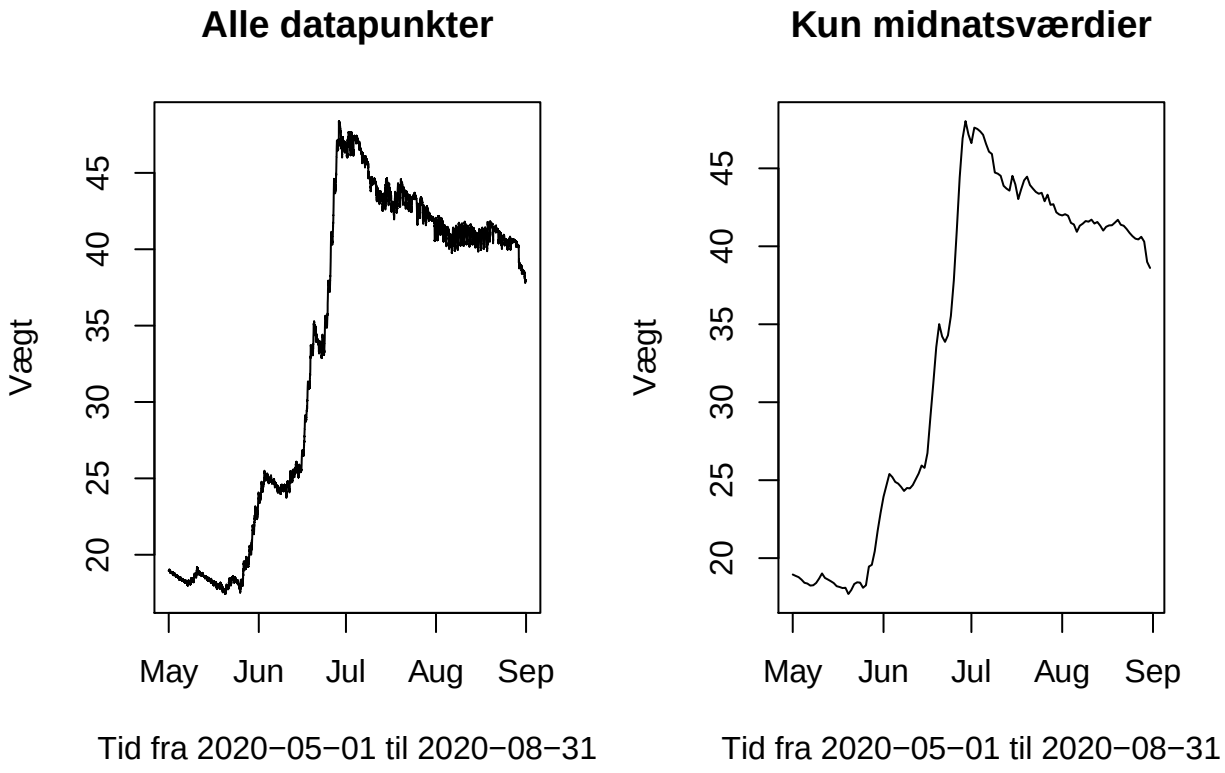

Vægtudvikling
Vægt
22
24
26
28
30
32
May 24
May 26
May 28
May 30
Jun 01
Tid fra 2020−05−24 til 2020−05−31

- Så der blev skrevet en lille funktion til at udligne disse manuelle indgreb (hele koden af funktionen kan findes i Bilag B). Funktionen tager imod bistader data og perioder, som skal udlignes. Den returnere behandlet bistade data. Nedenfor kan man se to grafer, til venstre er der en graf med den oprindelige data og til højre er der en graf med den behandlede data, dvs. udlignede manuelle indgreb.



Midnats vægt

Den anden ting, som gør det svært at aflæse den ægte vægt tilvækst pr. dag, er at der alt for mange datapunkter, hvor der er målinger hvert 5. minut. Bierne følger den daglige rutine. Det flyver ud om morgen, indsamler nektar i løbet af dagen, om natten fordamper den. Det medfører at vægten svinger op og ned i løbet af dagen og for at se den ægte vægt tilvækst pr. dag er det bedst at kigge på midnatsværdier. Så der blev skrevet endnu en funktion, som trækker ud midnatsværdier (hele koden kan findes i Bilag B). Nedenfor er der to grafer. Til venstre er der alle data punkter, dvs. hvert 5. minut. Til højre er der kun midnatsværdier. Man kan se at det er nemmere aflæse grafen kun med midnatsværdier især i august måned, hvor vægten svinger mest i løbet af dagen. - Det var også nødt til at skubbe en dag tilbage. Dvs. den skal vise en f.eks. en vægt måling som tage 2020-05-23 00:00:01 viser faktisk vægt tilvækst i den 2020-05-22, so det vil være ukorrekt at den blev stående i 2020-05-23, dvs. det vil ikke passe med andre målinger som for eksampel maksimale temperatur , da vi har lyst til at finde ud vægt tilvækst i forhold til den forudgående dags maksimale temperatur hvis vi tager f.eks. 2020-05-20 00:00:01 vægt, så vi skal bruge den højeste tempratur dagen før, som var på tidspunt: 2020-05-19 15:55:01 dvs med det tempratur var dette tilvækst.. Defor alle vægt vædier er skubbet bagud med funktionen `lead`



Middags temperatur

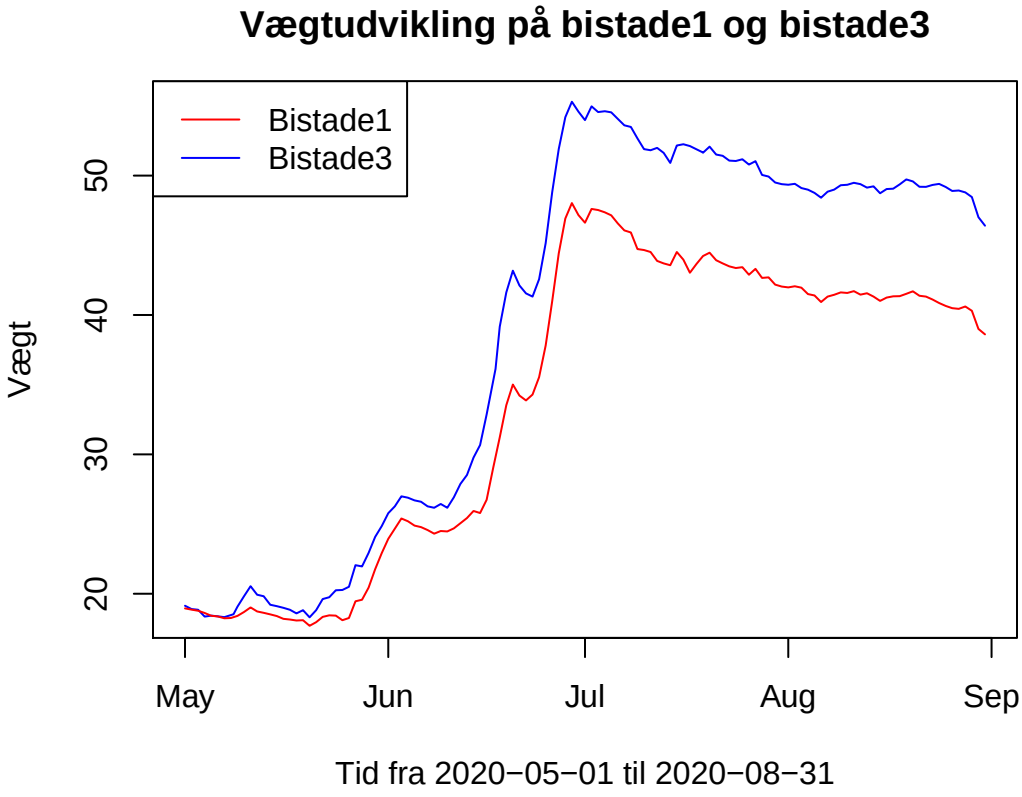
Ligesom kan man få mest indsigt i data ved at kigge på midnatsvægt, så er det bedst at kigge på den højeste temperatur på dagen, da det mest afsåejler bienerne bevægelse. Projektstiller har sagt at det bedste at bruge middags værider for at se udvikling. Men man kan iike bruge temp midnat. Projekt sitllerende forslået at bruge middags værider, men det var lidt svært at trakke dem ud at datasæt, pga. manglende data og ulige tidstempler. Så der blev valgt at bruge den højeste døgn værdi med midnatsvægt.

- Efter der blev fandt den højest tempratur på dagen, blev der nødt til at "lag" tempratur en dag bagud. Det kan forklares, at . Dvs.

Dataanalyze

Sammenligning af bistader

I dette afsnit skal der sammenlignes to bistader fra den samen have (videre i afsnit bliver det kaldt Bistade1, Bistade3). Formålet med sammenligned at vudere om den anden familie præstere bedre end den anden og forudse sygdomme. Nedenfor kan man se to grafer af bistadet, som er behandlede for manuelle indgreb og kun midnatsværdier jf. Oprensning afsnit.



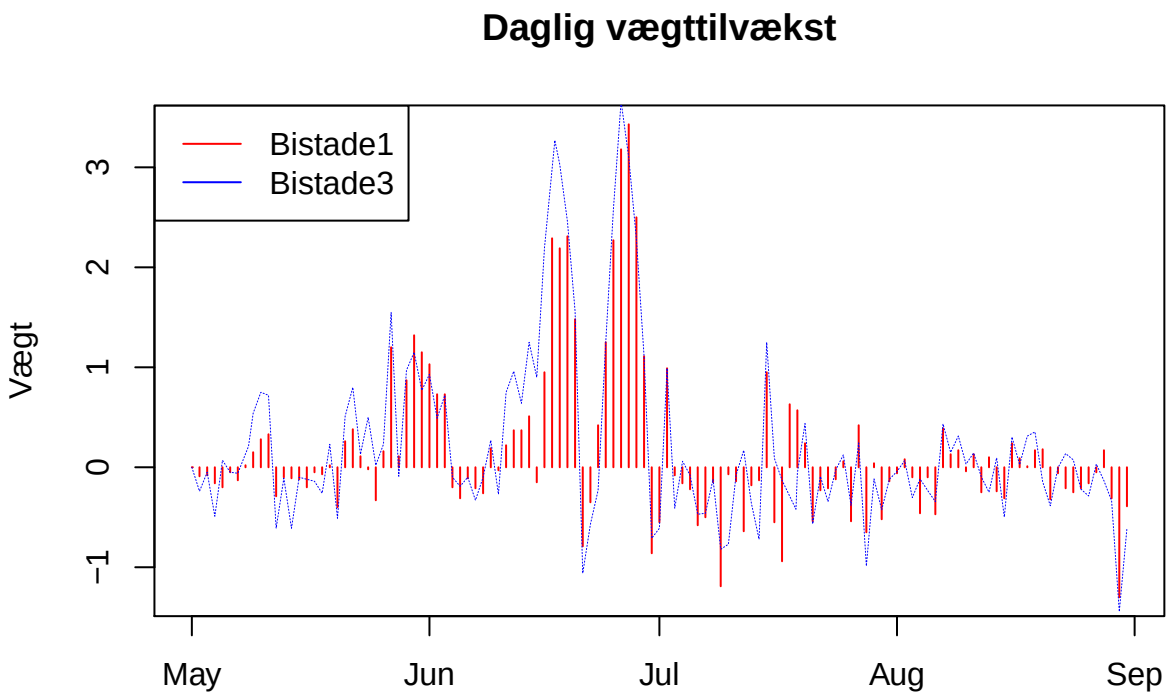
Den daglige vægttilvækst For at gøre det nemmere at sammenligne bistader, blev der taget de daglige vægttvækst af disse bistader. Gennemsnit af de dagligt vægttilvækst for de respektive bistader er $\mu_1 = 0.1598374$(Bistade1) og $\mu_3 = 0.2188468$(Bistade3).

```
mean(hive1$daily_weight_delta)
```

```
## [1] 0.1598374
```

```
mean(hive3$daily_weight_delta)
```

```
## [1] 0.2216829
```



Hypotesetest. Umiddelbart kan man se, at bistade 3 har lidt højere gennemsnit, men er det nok at konstatere forskellen mellem disse stader? For at undersøge det blev der udført et hypotesetest.

$$H_0 \ : \ \mu_1 = \mu_3 \equiv \mu_1 - \mu_3 = 0$$
$$H_1 \ : \ \mu_1 \neq \mu_3 \equiv \mu_1 - \mu_3 \neq 0$$

"Welch Two Sample t-test" Nedenfor **kan der ikke konstateres en signifikant forskel i den daglige vægttilvækst mellem Bistade1 og Bistade3**, fordi p-value (sandsynlighed at vi har fået denne forskel tilfældigt) er ret stor 0.5763 og 95% konfidensinterval inkluderer 0. Dvs. vi ikke kan afvise H_0, sem er, at de daglige vægttilvækst er ens.

```
t.test(hive1$daily_weight_delta, hive3$daily_weight_delta)
```

```
##
##  Welch Two Sample t-test
##
## data:  hive1$daily_weight_delta and hive3$daily_weight_delta
## t = -0.58405, df = 240.5, p-value = 0.5597
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2704368  0.1467457
## sample estimates:
## mean of x mean of y
## 0.1598374 0.2216829
```

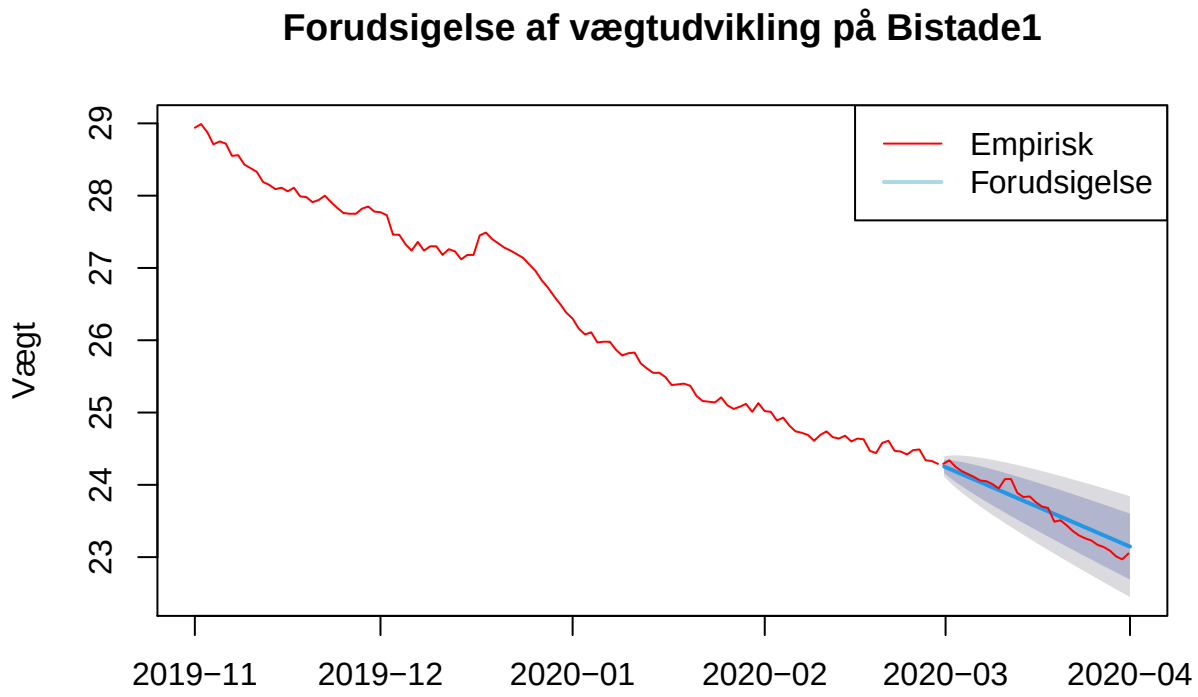
Konklusion på sammenligning Selv om der ikke kunne lade sig gøres at bevise signifikant forskellen på bistader ved at bruge t.test, er den forskellen som man kan sige på grafer er nok til biavlen til at konkludere, at der er en tydeligt eksampel på at bifamilier er forskellige, da stade3 producerer mest honning, og derfor ville han vælge Bistade3, som "mor" til en evt ny familie.

Forudsigelse af vægten i en vinterperiode

Formål Den første formål er, at det er ret vigtigt for biavlen at overvåge om bierne har nok foder i en vinter periode. Det vil sige, at vægten falder gradvis i løbet af vinteren, men den skal ikke falde under en vist niveau, og biavlen skal gribe ind, hvis den gør. Så det kunne være en fordel til biavlen, at han kunne forudsige, hvornår den falder under dette niveau, så biavlen kunne fylde foderdepoter op. Hvis vi f.eks. tager Bistade1, så ligger dens minimumvægt på 15kg, da magasin, tavler og bier vejer omkring 12 kg. Den anden formål kunne være at forudsige om familien har det godt om vinteren. Familien er god stand, hvis vægten falder konstant og som forventet, så det betyder at familien forbruger foder, men hvis vægten stopper med at falde, så det kan være en tegn, at familien er syg eller i det værste tilfælde uddødet.

Modellen Der blev benyttet `ets` funktion i `forecast` pakke til at lave en modellen, hvor der blev givet daglige vægtværdier fra 2019-11-01 til 2020-03-01. Denne modellen blev brugt i `forecast` funktionen til at forudsige vægten i hele marts måned. På figuren nedenfor kan man se de forudsagtede værdier. En blå linje betyder "Point Forecast", den mørke gråzone markerer 80% konfidensintervall, og den lyse gråzone markerer 95% konfidensinterval.

Backtesting Der blev også lavet en backtesting, fordi vi kender jo de empiriske værdier for marts måned, så disse også blev plottet på grafen, og man må sige at modellen blev opsat ret korrekt, da alle empiriske værdier ligger ind i 95% konfidensinterval interval.



Forudsigelse af træk ved at bruge klassificering

Træk er når bierne henter nektar fra blomsterne, hvilket medfører at bistadets vægten stiger. Hvis vægten ikke stiger, så den plejer at falde gradvis, fordi bierne begynder at spise det, som blev indsamlet. Så i denne sektion vil jeg prøve at forudsige om bierne trækker ved at benytte lufttempratur. Formålet ved det kunne være bl.a. at give en "Early Warning" til biavlen, når biene skulle trækker i forhold til verjet, men vægten faktisk faldet, hvilket kan være en tegn på at der er noget galt med bifamilien.

Forberedese af data

Der blev udført disse forberedelse på dataen til at klargøre daten specifikt til klassificering:

- Vægttilvæksten, som oprindeligt var kvantitativ variabel, blev lavet om til kvalitative variablel. Vægttilvæksten blev opdelt i to kategorier, UP og DOWN. UP er når vægtilvæksten er positiv og DOWN er når vægttilvæksten er negativ.
- Der blev valgt at benytte data kun fra juni måned, fordi biene trækker kun, når der er nektar dvs. i slutning af foråret og i starten af sommeren, så derfor vil det ikke give mening at inkludere måneder, hvor der ikke sker noget træk.
- Og sidst såfremt er dataen kun fra 2019 og 2020 , så benyttes der 2020 juni måned til at lave modellen og 2019 juni måned til at validere modellen.

Disse forberedelser kan man se i R kode nedenunder. Valideringsdatasættet blev behandlet på samme måde.

```
# Prepare train data
hive_data_2020_jun.train <- import_hive_data_daily(from = "'2020-06-01 00:00:00'",
                                                   to="'2020-07-01 00:00:00'")
hive_data_2020_jun.train <- hive_data_2020_jun.train %>%  mutate(weight_delta =
                hive_weight_kgs_daily - dplyr::lag(hive_weight_kgs_daily)) %>%
                mutate(weight_delta = ifelse(is.na(weight_delta), 0, weight_delta))
# weightdelta of the first day could not be calculated,
# because the prior day is not available in the current fetch dataset,
# so the weight delta is calculated and inserted manually.
hive_data_2020_jun.train[1,"weight_delta"] <- 0.81
hive_data_2020_jun.train <- select(hive_data_2020_jun.train, !hive_weight_kgs_daily)
hive_data_2020_jun.train <- hive_data_2020_jun.train %>%
  mutate(weight_delta_direction = ifelse(weight_delta<0, "DOWN", "UP"))
hive_data_2020_jun.train <- select(hive_data_2020_jun.train, !weight_delta)
hive_data_2020_jun.train <- select(hive_data_2020_jun.train, !dt)
hive_data_2020_jun.train$weight_delta_direction <- factor(hive_data_2020_jun.train$weight_delta_directi
```

```
## Warning in result_fetch(res@ptr, n = n): Column `wx_wind_degrees`: mixed type,
## first seen values of type integer, coercing other values of type string
```

Modellen

- Der blev benyttet "Logistic regression" til at lave modellen, hvor der bruges `glm` function. Fra en opsummering nedenfor kan vi konkludere, at

```
set.seed(12345)
model <- train(weight_delta_direction~ambient_temp_c_day_max, data = hive_data_2020_jun.train,
                method = "glm",
                trControl = trainControl(method = "cv",number = 5)  )
summary(model)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8036  -0.5358   0.2256   0.5877   1.8753
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -16.1306     6.1559  -2.620  0.00878 **
## ambient_temp_c_day_max   0.8045     0.3008   2.675  0.00748 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 39.429  on 29  degrees of freedom
## Residual deviance: 22.645  on 28  degrees of freedom
## AIC: 26.645
##
## Number of Fisher Scoring iterations: 6
```

Som man kan se fra modelberksivelsen ovenfor. Beskrive parameter, p value

- Hvad betyder Estimate std. in glm???

Jeg kan ikke lige overskue, hvad betyder coefficients her?

Dvs. i linær regression, hvis vi øger med så meget predictor variable, så

So vil det øge med så meget outcome. Men hvad med her?

"coefficients that do differs from zero at the p<.10 level."

Validering kk

```
prob <- predict(model, hive_data_2019_jun.validate)
logit.perf <- table(hive_data_2019_jun.validate$weight_delta_direction, prob, dnn=c("Actual", "Predicte
logit.perf
```

```
##       Predicted
## Actual DOWN UP
##   DOWN    6  4
##   UP      3 17
```

- En fejl, hvis jeg bruger alle variabler
- Er der nok med en variablet med til modellen?
- Hvorfor weight_delta og hive_weight_kgs har så lille correlation?

```
        Predicted
Actual DOWN UP
  DOWN    6  4
  UP      3 17
```

- Dvs når vi forudsiger at vægten skulle gå UP, men det gik faktisk DOWN, så det kan betyde at der er problem med familien.
- det var kun 4 der er farlig i dette. Som var predicted UP men var actual DOWN. Dvs. der kunne man få forket "Early warning", som man ikke skal reagere til.
- "Accuracy and Kappa These are the default metrics used to evaluate algorithms on binary and multi-class classification datasets in caret." https://machinelearningmastery.com/machine-learning-evaluation-metrics-in-r/
- Vi kan se, , at modellen forudsat 23 rigtigt ud af 30 dage. da der bruges validerings data fra 2019 july måned. hvis vi bruger modellen til at forudse vægtændring af dataen fra 2019 July måned
- Der blev brugt bl.a. en "5 fold cross validation" til at estimere, hvor præcist modellen vil estimere i fremtiden. Fra 5-cross validation, vi kan see at modellen har 0.8 nøjagtighed, som er lidt større end resultater fra ved at bruge et års tidlegere data, hvor modellen forudsat 23 rigtigt ud af 30, hivlket giver ca. 0.76 nøjagtighed.

Konklusion

Bilag

Bilag A: Opsummering af alle variabler i datasættet

```
Hmisc::describe(hive_data)
```

```
## hive_data
##
##  43  Variables      105064  Observations
## ---------------------------------------------------------------------------
## rowid
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0   105064        1    85262    35022    37983    43236
##      .25      .50      .75      .90      .95
##    58996    85262   111527   127287   132540
##
## lowest :  32730  32731  32732  32733  32734, highest: 137789 137790 137791 137792 137793
## ---------------------------------------------------------------------------
## hive_id
##        n  missing distinct     Info     Mean      Gmd
##   105064        0        1        0        1        0
##
## Value           1
## Frequency  105064
## Proportion      1
## ---------------------------------------------------------------------------
## timestamp
##        n  missing distinct
##   105064        0   105064
##
## lowest : 2019-08-31 22:01:04 2019-08-31 22:06:04 2019-08-31 22:11:05 2019-08-31 22:16:05 2019-08-31 1
## highest: 2020-08-31 21:35:38 2020-08-31 21:40:38 2020-08-31 21:45:38 2020-08-31 21:50:39 2020-08-31 1
## ---------------------------------------------------------------------------
## hive_observation_time_local
##        n  missing distinct
##   105064        0   105052
##
## lowest : 2019-09-01 00:00:01 2019-09-01 00:05:01 2019-09-01 00:10:01 2019-09-01 00:15:02 2019-09-01 0
## highest: 2020-08-31 23:35:01 2020-08-31 23:40:01 2020-08-31 23:45:01 2020-08-31 23:50:01 2020-08-31 1
## ---------------------------------------------------------------------------
## hive_weight_lbs
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0     3194        1    63.81    18.14    45.59    48.96
##      .25      .50      .75      .90      .95
##    52.14    59.46    66.93    94.78    99.27
##
## lowest :  33.20  33.99  34.02  34.04  34.06, highest: 132.23 132.28 132.30 132.32 132.37
## ---------------------------------------------------------------------------
## hive_weight_kgs
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0     3194        1    28.94     8.23    20.68    22.21
##      .25      .50      .75      .90      .95
##    23.65    26.97    30.36    42.99    45.03
```

```
##
## lowest : 15.06 15.42 15.43 15.44 15.45, highest: 59.98 60.00 60.01 60.02 60.04
## ---------------------------------------------------------------------------
## hive_temp_f
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      350        1    77.57    17.14    48.20    55.94
##      .25      .50      .75      .90      .95
##    68.18    77.00    93.56    94.82    95.18
##
## lowest : 35.78 35.96 36.14 36.50 36.68, highest: 98.06 98.24 98.42 98.60 98.78
## ---------------------------------------------------------------------------
## hive_temp_c
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      350        1    25.32    9.522      9.0     13.3
##      .25      .50      .75      .90      .95
##     20.1     25.0     34.2     34.9     35.1
##
## lowest :  2.1  2.2  2.3  2.5  2.6, highest: 36.7 36.8 36.9 37.0 37.1
## ---------------------------------------------------------------------------
## hive_humidity
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      551        1     62.1    11.76    41.42    47.50
##      .25      .50      .75      .90      .95
##    57.10    60.30    67.50    79.00    83.70
##
## lowest : 36.0 36.1 36.2 36.3 36.4, highest: 90.6 90.7 90.8 90.9 91.0
## ---------------------------------------------------------------------------
## hive_battery_voltage
##        n  missing distinct     Info     Mean      Gmd
##   105064        0        1        0     2.13        0
##
## Value        2.13
## Frequency  105064
## Proportion      1
## ---------------------------------------------------------------------------
## ambient_temp_f
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   101787     3277      360        1    49.09    12.48    33.80    36.86
##      .25      .50      .75      .90      .95
##    40.82    46.40    56.48    64.76    69.62
##
## lowest : 24.26 24.44 24.62 24.80 24.98, highest: 88.16 88.34 88.52 88.70 88.88
## ---------------------------------------------------------------------------
## ambient_temp_c
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   101787     3277      360        1    9.496    6.932      1.0      2.7
##      .25      .50      .75      .90      .95
##      4.9      8.0     13.6     18.2     20.9
##
## lowest : -4.3 -4.2 -4.1 -4.0 -3.9, highest: 31.2 31.3 31.4 31.5 31.6
## ---------------------------------------------------------------------------
## ambient_humidity
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   101787     3277      433    0.211       99    1.724     94.2     99.9
```

```
##      .25      .50      .75      .90      .95
##     99.9     99.9     99.9     99.9     99.9
##
## lowest : 55.1 55.5 55.6 56.0 56.5, highest: 99.5 99.6 99.7 99.8 99.9
## ------------------------------------------------------------------------------
## ambient_luminance
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      669    0.809    14.65    26.32        0        0
##      .25      .50      .75      .90      .95
##        0        0        5       31       71
##
## lowest :    0    1    2    3    4, highest: 1314 1408 1449 1804 2070
## ------------------------------------------------------------------------------
## wx_temp_f
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201      280        1    48.52    13.52     33.1     36.3
##      .25      .50      .75      .90      .95
##     41.0     45.3     53.6     58.3     61.5
##
## lowest :  21.6  21.7  21.9  22.1  22.3, highest:  71.2  72.0  72.1  73.4 999.0
##
## Value         20    30    40    50    60    70  1000
## Frequency    266  2337 14918 11258  6235   779    70
## Proportion 0.007 0.065 0.416 0.314 0.174 0.022 0.002
##
## For the frequency table, variable is rounded to the nearest 10
## ------------------------------------------------------------------------------
## wx_temp_c
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201      223        1    9.414    7.344      1.6      2.7
##      .25      .50      .75      .90      .95
##      5.0      7.4     12.0     14.6     16.4
##
## lowest :   0.0   0.1   0.2   0.3   0.4, highest:  21.8  22.2  22.3  23.0 572.8
##
## Value          0     5    10    15    20    25   575
## Frequency   3350 14763 10214  6256  1203     7    70
## Proportion 0.093 0.412 0.285 0.174 0.034 0.000 0.002
##
## For the frequency table, variable is rounded to the nearest 5
## ------------------------------------------------------------------------------
## wx_relative_humidity
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201       52    0.987    93.14    12.33       73       79
##      .25      .50      .75      .90      .95
##       88       94       98       99       99
##
## lowest :  46  49  50  52  53, highest:  96  97  98  99 999
## ------------------------------------------------------------------------------
## wx_wind_dir
##        n  missing distinct
##   105064        0       17
##
## lowest : E   ENE ESE N   NE , highest: SSW SW  W   WNW WSW
```

```
##
## Value          E   ENE   ESE     N    NE   NNE   NNW  NULL    NW     S    SE
## Frequency   3266  4594  2393  3904  4722  6196  1587 69201   908  1752  2155
## Proportion 0.031 0.044 0.023 0.037 0.045 0.059 0.015 0.659 0.009 0.017 0.021
##
## Value        SSE   SSW    SW     W   WNW   WSW
## Frequency   1778   940   441   383   520   324
## Proportion 0.017 0.009 0.004 0.004 0.005 0.003
## ---------------------------------------------------------------------------
## wx_wind_degrees
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   105064        0      361    0.711    38.59     63.6        0        0
##      .25      .50      .75      .90      .95
##        0        0       37      140      221
##
## lowest :   0   1   2   3   4, highest: 356 357 358 359 999
## ---------------------------------------------------------------------------
## wx_wind_mph
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201       42    0.917    2.844    5.096      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      0.2      1.3      2.7      3.6
##
## lowest :   0.0   0.2   0.4   0.7   0.9, highest:   9.2   9.4  10.3  13.9 999.0
##
## Value          0     2     4     6     8    10    14  1000
## Frequency  24113  9175  2014   448    29    12     2    70
## Proportion 0.672 0.256 0.056 0.012 0.001 0.000 0.000 0.002
##
## For the frequency table, variable is rounded to the nearest 2
## ---------------------------------------------------------------------------
## wx_wind_gust_mph
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201       16      0.9    3.361    5.678      0.0      0.0
##      .25      .50      .75      .90      .95
##      0.0      1.1      2.2      3.4      5.8
##
## lowest :   0.0   1.1   2.2   3.4   4.5, highest:  12.5  13.6  14.8  17.2 999.0
##
## Value        0.0   1.1   2.2   3.4   4.5   5.8   6.9   8.1   9.2  10.3  11.4
## Frequency  15582  8783  5091  2910  1637   822   519   220   120    45    36
## Proportion 0.434 0.245 0.142 0.081 0.046 0.023 0.014 0.006 0.003 0.001 0.001
##
## Value       12.5  13.6  14.8  17.2 999.0
## Frequency     18     4     4     2    70
## Proportion 0.001 0.000 0.000 0.000 0.002
## ---------------------------------------------------------------------------
## wx_pressure_mb
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201      184        1     1002    12.23    981.9    986.7
##      .25      .50      .75      .90      .95
##    995.8   1003.3   1009.7   1014.8   1019.2
##
## lowest :  973.5  973.8  974.2  974.5  974.8, highest: 1034.4 1034.8 1035.1 1035.4 1035.8
```

```
## ------------------------------------------------------------------------------
## wx_dewpoint_f
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201      253        1    45.96    12.71     31.1     34.3
##      .25      .50      .75      .90      .95
##     38.8     43.3     50.7     54.5     56.3
##
## lowest :  20.8  21.0  21.2  21.4  21.6, highest:  66.6  66.9  67.6  69.3 999.0
##
## Value         20    30    40    50    60    70  1000
## Frequency    328  3774 16314 12217  3077    83    70
## Proportion 0.009 0.105 0.455 0.341 0.086 0.002 0.002
##
## For the frequency table, variable is rounded to the nearest 10
## ------------------------------------------------------------------------------
## wx_dewpoint_c
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201      192        1    8.083    6.756      1.1      2.0
##      .25      .50      .75      .90      .95
##      3.9      6.3     10.4     12.5     13.5
##
## lowest :   0.0   0.1   0.2   0.3   0.4, highest:  19.2  19.4  19.8  20.7 572.8
##
## Value          0     5    10    15    20   575
## Frequency   5192 15675 11419  3330   177    70
## Proportion 0.145 0.437 0.318 0.093 0.005 0.002
##
## For the frequency table, variable is rounded to the nearest 5
## ------------------------------------------------------------------------------
## wx_precip_1hr_in
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201       18    0.134    1.954    3.901        0        0
##      .25      .50      .75      .90      .95
##        0        0        0        0        0
##
## lowest :   0.00   0.05   0.07   0.12   0.14, highest:   0.43   0.52   0.61   0.78 999.00
##
## Value        0.00   0.05   0.07   0.12   0.14   0.17   0.19   0.24   0.26
## Frequency   34179    579    629    181     16     42     46     28      4
## Proportion  0.953  0.016  0.018  0.005  0.000  0.001  0.001  0.001  0.000
##
## Value        0.28   0.31   0.35   0.38   0.43   0.52   0.61   0.78 999.00
## Frequency       9     30     15      5     20      4      4      2     70
## Proportion  0.000  0.001  0.000  0.000  0.001  0.000  0.000  0.000  0.002
## ------------------------------------------------------------------------------
## wx_precip_today_in
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##    35863    69201       81    0.809    1.997     3.97     0.00     0.00
##      .25      .50      .75      .90      .95
##     0.00     0.00     0.03     0.15     0.30
##
## lowest :   0.00   0.01   0.02   0.03   0.04, highest:   1.35   1.48   1.50   1.51 999.00
##
## Value          0     2  1000
```

```
## Frequency  35760    33    70
## Proportion 0.997 0.001 0.002
##
## For the frequency table, variable is rounded to the nearest 2
## ----------------------------------------------------------------------------
## quality
##        n  missing distinct     Info     Mean      Gmd
##   105064        0        1        0        5        0
##
## Value           5
## Frequency  105064
## Proportion      1
## ----------------------------------------------------------------------------
##
## Variables with all observations missing:
##
##  [1] hive_observation_time_utc    hive_weight_lbs_delta
##  [3] hive_weight_lbs_delta_daily  hive_weight_lbs_filtered
##  [5] hive_manipulation_change_lbs hive_weight_kgs_delta
##  [7] hive_weight_kgs_delta_daily  hive_weight_kgs_filtered
##  [9] hive_manipulation_change_kgs ambient_precip_in
## [11] wx_station_id                wx_observation_time_rfc822
## [13] wx_pressure_in               wx_solar_radiation
## [15] wx_precip_1hr_metric         wx_precip_today_metric
```

Bilag B: Mine funktioner

```
import_hive_data <- function(from, to){
  library(DBI)
  con <- dbConnect(RSQLite::SQLite(), "data/stade1.db")
  table_name <- dbListTables(con)
  fields <- dbListFields(con, table_name)
  select_period_with_intervention_query <- paste("SELECT * from", table_name, "WHERE hive_observation_t
                        hive_observation_time_local < ", to, sep=" ")

  sendQuery <- dbSendQuery(con, select_period_with_intervention_query )
  #  hive_data <<- dbFetch(sendQuery)
  hive_data <- dbFetch(sendQuery)
  hive_data$hive_observation_time_local <- strptime(hive_data$hive_observation_time_local, format = "%Y
  return (hive_data)
}

import_hive_data_csv <- function(filename){
  hive_data <- read.table(file=filename, sep=",")
  hive_data <- hive_data[ c("V1", "V2")]
  colnames(hive_data) <- c("hive_observation_time_local", "hive_weight_kgs")
  hive_data$hive_observation_time_local <- strptime(hive_data$hive_observation_time_local, format = "%Y
  return (hive_data)
}

extract_rows_given_weightdelta <- function(hive_data, weightdelta){
  hive_data <- hive_data %>%  mutate(weight_delta = hive_weight_kgs -
```

```
                                        dplyr::lag(hive_weight_kgs)) %>%
    mutate(weight_delta = ifelse(is.na(weight_delta), 0, weight_delta))

  hive_data[which(abs(hive_data[,"weight_delta"]) > weightdelta),
            c("hive_observation_time_local", "weight_delta")]
}

extract_rows_given_timedelta <- function(hive_data, timedelta){
  hive_data <- hive_data %>%  mutate(timestamp_delta = hive_observation_time_local -
                                        dplyr::lag(hive_observation_time_local)) %>%
    mutate(timestamp_delta = ifelse(is.na(timestamp_delta), 0, timestamp_delta))
  hive_data[which(abs(hive_data[,"timestamp_delta"]) > timedelta),
            c("hive_observation_time_local", "timestamp_delta")]
}

plot_time_weight <- function(hive_data, title){
  if(missing(title)){
    title <- "Vægtudvikling"
  }
  min <- as.Date(head(hive_data, 1)[,"hive_observation_time_local"])
  max <- as.Date(tail(hive_data, 1)[,"hive_observation_time_local"])
  plot(hive_data$hive_observation_time_local, hive_data$hive_weight_kgs, type = 'l', xlab = paste("Tid
  # , at=seq(as.Date(min),as.Date(max),by=(13*7))
}

plot_time_weight_temp <- function(hive_data){
  min <- as.Date(head(hive_data, 1)[,4])
  max <- as.Date(tail(hive_data, 1)[,4])
  par(mar = c(5, 5, 3, 5))
  plot(hive_data$hive_observation_time_local, hive_data$hive_weight_kgs, type ="l", ylab = "Vægt",
       main ="Sammenhæng mellem vægt og temperatur", xlab = paste("Tid fra", min, "til", max, sep= " ")
       col = "blue")
  par(new = TRUE)
  plot(hive_data$hive_observation_time_local, hive_data$ambient_temp_c, type = "l", xaxt = "n", yaxt =
       ylab = "", xlab = "", col = "red") # , lty = 2
  axis(side = 4)
  mtext("Temperatur", side = 4, line = 3)
  legend("topleft", c("Vægt", "Temperatur"),
         col = c("blue", "red"), lty = c(1, 1))
}


manipulate_weight_deltas <- function(hive_data, periods){
  library(dplyr)
  # Add column weight_delta
  hive_data <- hive_data %>%  mutate(weight_delta = hive_weight_kgs - dplyr::lag(hive_weight_kgs)) %>%
  mutate(weight_delta = ifelse(is.na(weight_delta), 0, weight_delta))

  # Manipulate weight deltas
  for(row in 1:nrow(periods)){
    hive_data$weight_delta[hive_data$hive_observation_time_local> periods[row, "from"]  & hive_data$hiv
  }
```

```
  # Produce cumulative sums of weight deltas
  hive_data <- hive_data %>% mutate(cum_delta=cumsum(weight_delta))

  # Produce new hive_weight_kgs from calculated cumukatiuve sums
  hive_data <- hive_data %>% mutate(hive_weight_kgs = hive_weight_kgs[1]+ cum_delta)


  # Remove the produced columns
  drops <- c("weight_delta", "cum_delta")
  hive_data <- hive_data[ , !(names(hive_data) %in% drops)]
  return(hive_data)
}

extract_midnight_weights <- function(hive_data){

  hive_data <- hive_data %>%
    mutate( dt = as.Date(hive_observation_time_local)) %>%
    group_by(dt) %>%
    filter(hive_observation_time_local == min(hive_observation_time_local)) %>%
    ungroup() %>%
    select(!dt)

  return(data.frame(hive_data))

}

return_period <- function(hive_data, from ,to){
  from <-  strptime(from, format = "%Y-%m-%d %H:%M:%S")
  to <- strptime(to, format = "%Y-%m-%d %H:%M:%S")
  return(hive_data[hive_data$hive_observation_time_local > from & hive_data$hive_observation_time_local
}
```