PML_6

2023-04-28

Opgave 1 (GAM)

1

Jeg indlæser datasættet `SAhearts`:

2

Jeg bruger `gam()` funktionen til at fitte en additiv model med variable tilsvare figur 5.4 fra bogen dvs.:

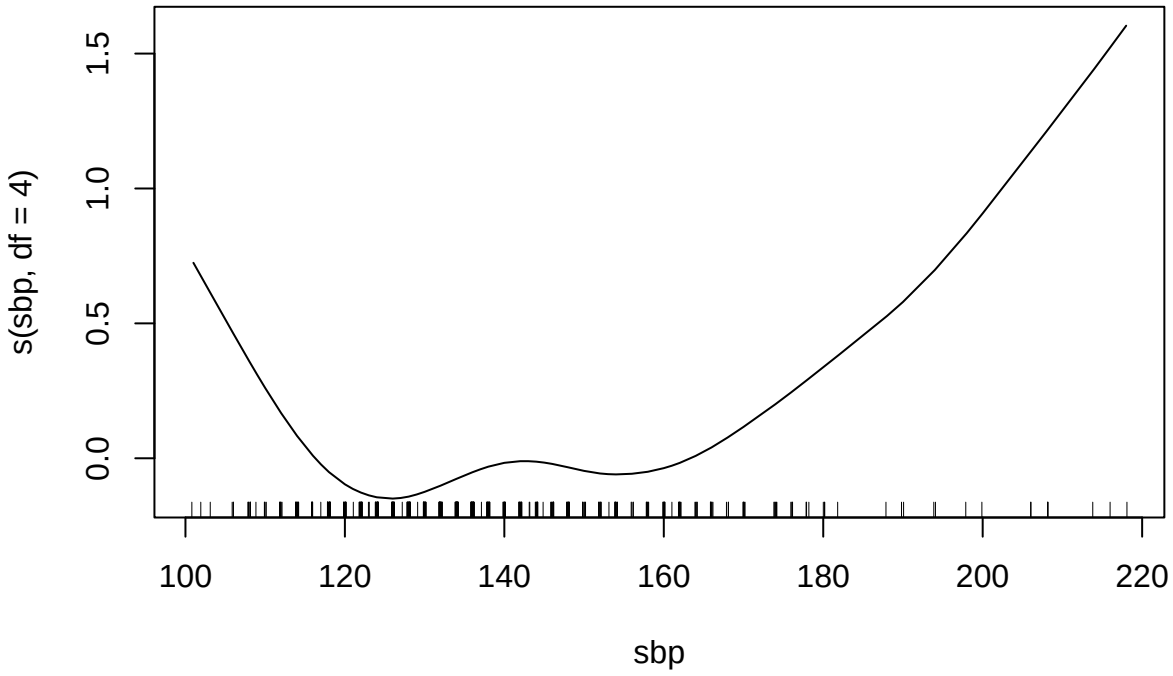
`sbp, tobacco, ldl, famhist, obesity, age`. Jeg sætter `df=4` for alle prædiktorerne (jeg bruger altså regression splines til alle prædiktorerne).

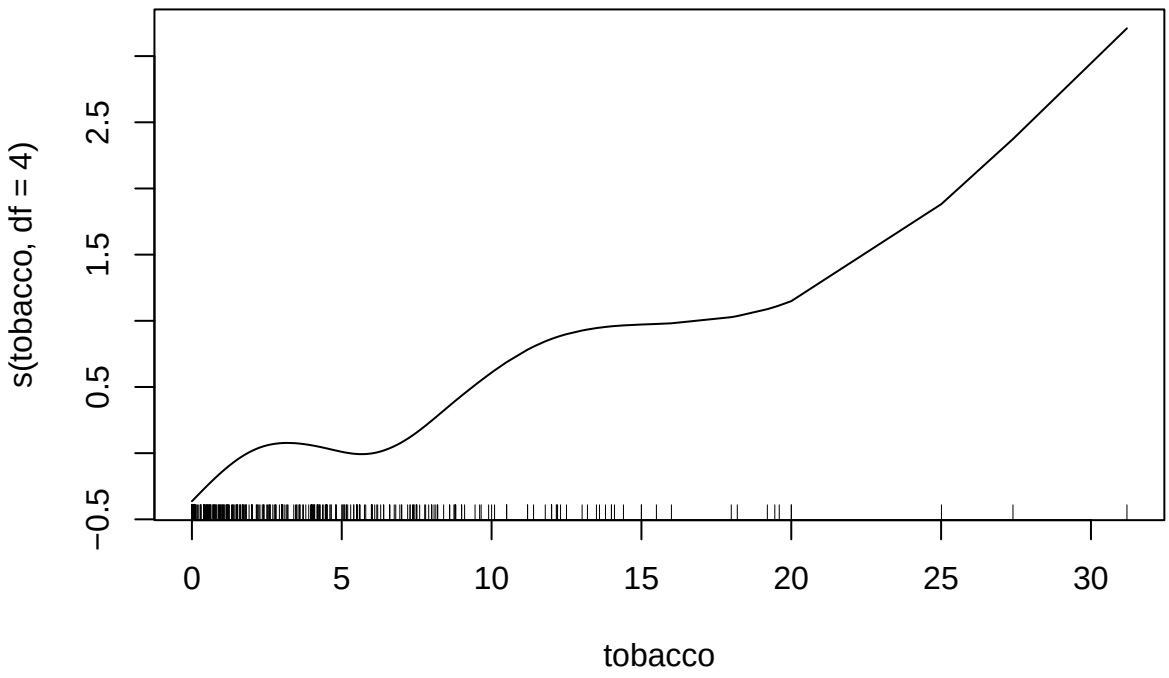
```
##
## Call: gam(formula = chd ~ s(sbp, df = 4) + s(tobacco, df = 4) + s(ldl,
##     df = 4) + famhist + s(obesity, df = 4) + s(age, df = 4),
##     family = binomial, data = SAheart)
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6370 -0.8096 -0.4260  0.8964  2.6546
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 596.1084 on 461 degrees of freedom
## Residual Deviance: 456.409 on 440.0004 degrees of freedom
## AIC: 500.4082
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                     Df Sum Sq Mean Sq F value    Pr(>F)
## s(sbp, df = 4)       1   8.03  8.0257  8.1979  0.004394 **
## s(tobacco, df = 4)   1  18.13 18.1272 18.5161 2.078e-05 ***
## s(ldl, df = 4)       1  15.62 15.6227 15.9579 7.591e-05 ***
## famhist              1  21.02 21.0174 21.4683 4.746e-06 ***
## s(obesity, df = 4)   1   1.52  1.5200  1.5526  0.213412
## s(age, df = 4)       1  15.76 15.7593 16.0974 7.070e-05 ***
## Residuals          440 430.76  0.9790
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                     Npar Df Npar Chisq  P(Chi)
```

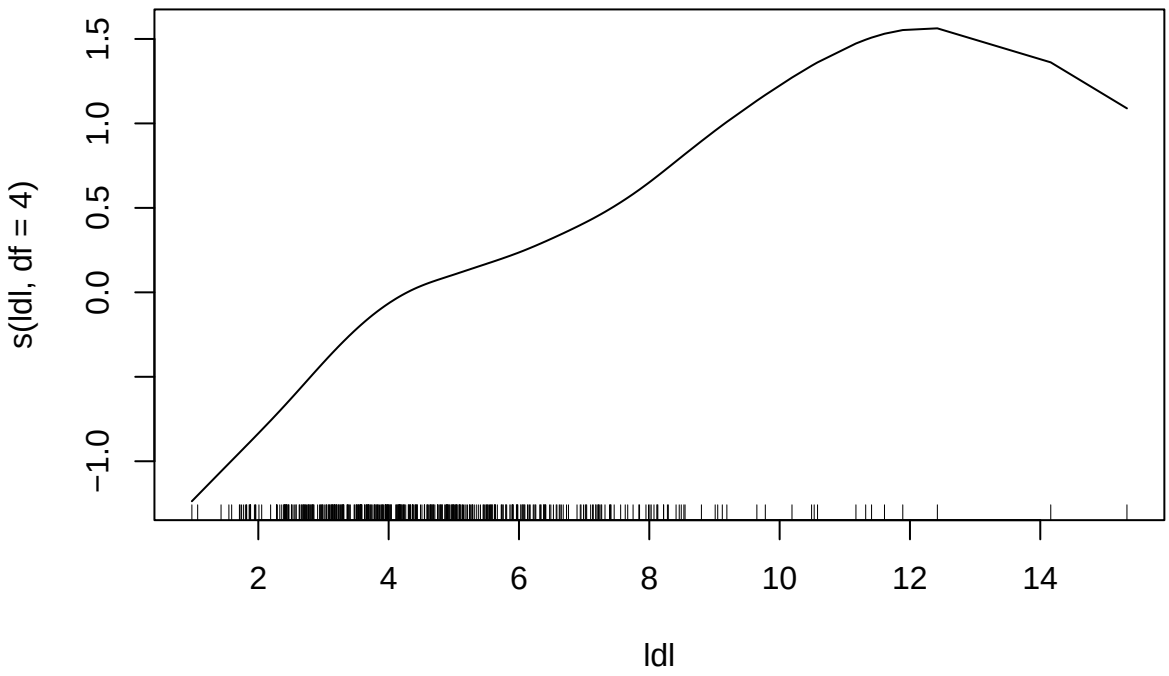
```
## (Intercept)
## s(sbp, df = 4)           3     5.6007 0.13274
## s(tobacco, df = 4)       3     5.8606 0.11857
## s(ldl, df = 4)           3     2.5228 0.47120
## famhist
## s(obesity, df = 4)       3     6.5169 0.08900 .
## s(age, df = 4)           3     7.0330 0.07084 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

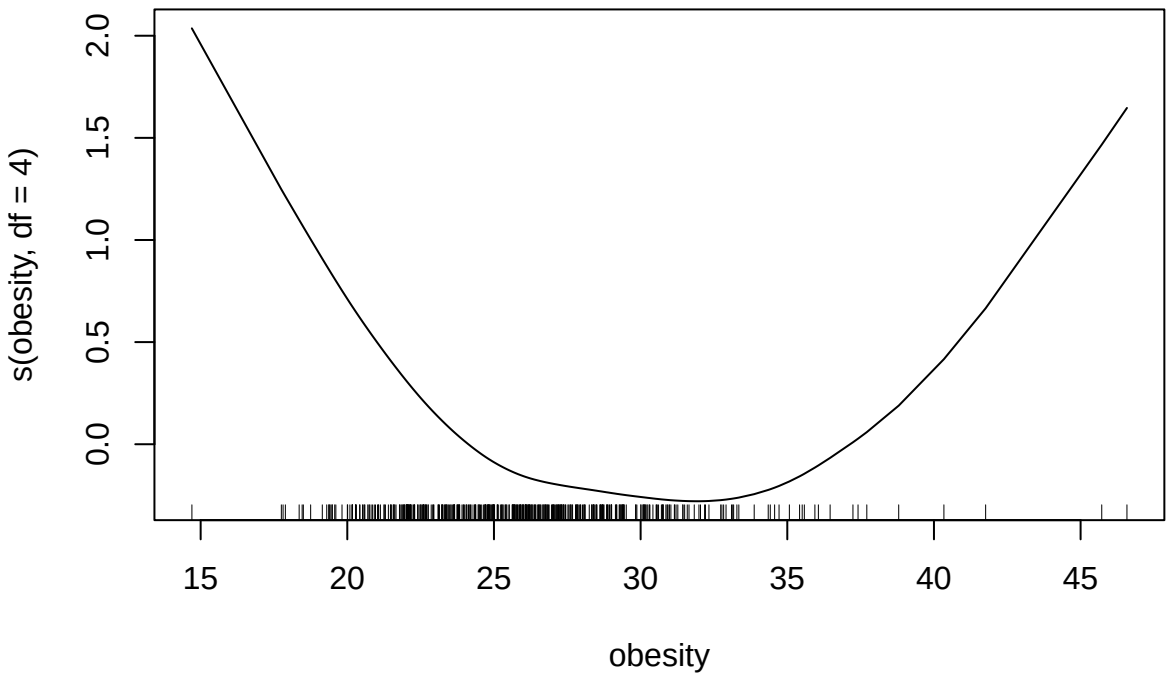
3

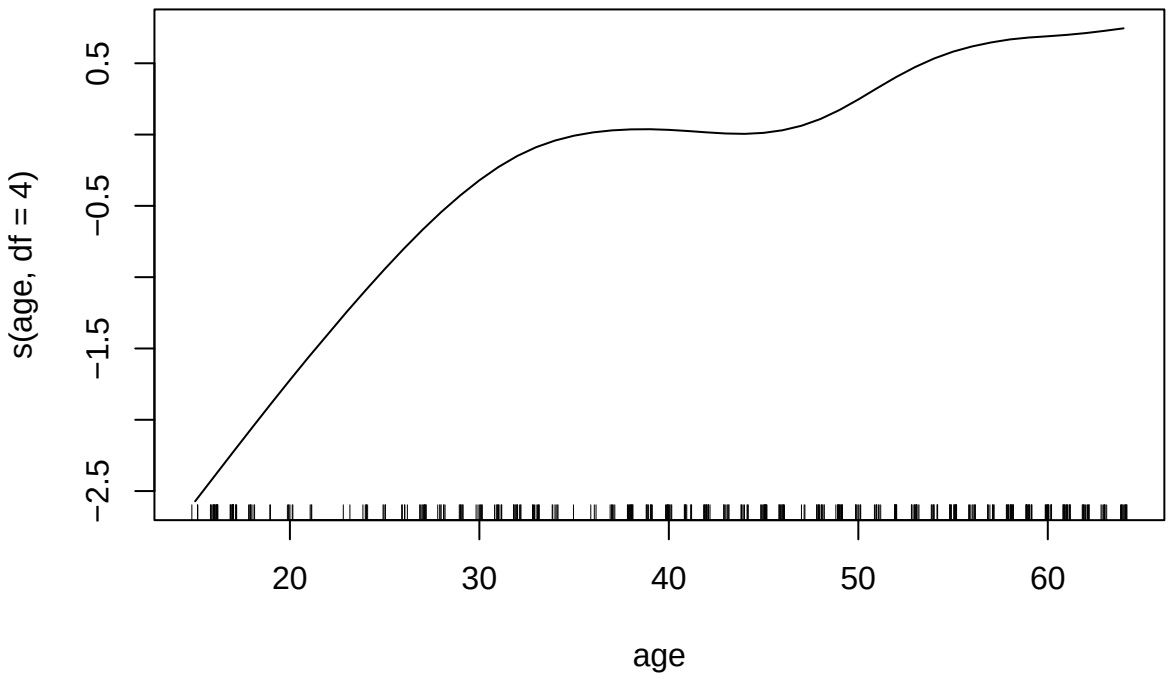
Jeg plotter de partielle effekter af modellens variable.




s(tobacco, df = 4)
−0.5
0.5
1.5
2.5
0
5
10
15
20
25
30
tobacco


s(ldl, df = 4)
−1.0
0.0
0.5
1.0
1.5
2
4
6
8
10
12
14
ldl


s(obesity, df = 4)
0.0
0.5
1.0
1.5
2.0
15
20
25
30
35
40
45
obesity



4

Jeg sammenligner den fulde model med en reduceret model, hvor jeg fjerne `age` variablen og laver en χ^2-test for at se hvilken model der fitter bedre.

```
## Analysis of Deviance Table
##
## Model 1: chd ~ s(sbp, df = 4) + s(tobacco, df = 4) + s(ldl, df = 4) +
##     famhist + s(obesity, df = 4)
## Model 2: chd ~ s(sbp, df = 4) + s(tobacco, df = 4) + s(ldl, df = 4) +
##     famhist + s(obesity, df = 4) + s(age, df = 4)
##   Resid. Df Resid. Dev     Df Deviance  Pr(>Chi)
## 1       444     481.51
## 2       440     456.41 3.9997   25.104 4.792e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Baseret på outputtet ovenfor kan vi se at den fulde model med `age`variablen har en statistik signifikant lavere residual. Hvorfor modellen bør være at fortrække.

5

Jeg udfører tilsvarende analyse men bruger `mgcv::gam` istedet for at diskutere forskelle og ligheder.

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## chd ~ s(age, k = 5) + s(sbp, k = 5) + s(tobacco, k = 5) + s(ldl,
##     k = 5) + s(obesity, k = 5) + famhist
##
## Parametric coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -1.3190     0.1754  -7.521 5.42e-14 ***
## famhistPresent   0.9472     0.2258   4.194 2.74e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df Chi.sq  p-value
## s(age)     3.329  3.756 19.019 0.000857 ***
## s(sbp)     1.658  2.058  2.713 0.270332
## s(tobacco) 1.000  1.000  9.636 0.001909 **
## s(ldl)     1.000  1.000 11.090 0.000868 ***
## s(obesity) 2.091  2.620  5.412 0.115614
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.228   Deviance explained = 20.9%
## UBRE = 0.068907  Scale est. = 1         n = 462
```

Jeg tester `age` variablen for linearitet ved at sammenligne den oprindelige model. Hvor det var på spline form med en hvor det indgår lineært:

```
## Analysis of Deviance Table
##
## Model 1: chd ~ s(sbp, k = 5) + s(tobacco, k = 5) + s(ldl, k = 5) + s(obesity,
##     k = 5) + famhist + age
## Model 2: chd ~ s(age, k = 5) + s(sbp, k = 5) + s(tobacco, k = 5) + s(ldl,
##     k = 5) + s(obesity, k = 5) + famhist
##   Resid. Df Resid. Dev     Df Deviance Pr(>Chi)
## 1    452.77     478.48
## 2    449.57     471.68 3.2087   6.7969  0.09103 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Baseret på p-værdien kan vi ikke inkludere om det ene model er bedre end den anden. Derfor kan vi heller ikke afvise at `age` bør have en lineær form i vores model.

Opgave 2

Vi ser på datasættet `phoneme` og benytter træ-baserede metoder til at prædiktere data.

1

Jeg indlæser datasættet.

2

Jeg gentager analysen for opgavesæt 2 med natural splines til at prædiktere:

```
##
## predtst  aa  ao
##   FALSE 227 103
##   TRUE   58 329
```

Umiddelbart virker det til at modellen er mere villig til at klassificere en lyd som "ao", men dette gør ikke nødvendigvis modellen bedre til at klassificere "ao" (da før eller siden vil modellen jo ramme rigtigt).

3 og 4

Planen er nu at udføre tilsvarende analyse med `rpart, randomForest, ada` og `gbm`.

```
##
## pred_rpart  aa  ao
##         aa 192  92
##         ao  95 338
```

```
##
## pred_rf  aa  ao
##      aa 217  59
##      ao  70 371
```

```
##
## pred_ada  aa  ao
##       ao  77 372
##       aa 210  58
```

```
## [1] "pred_gbm"
```

```
##
##          aa  ao
##   FALSE 212  66
##   TRUE   75 364
```

Baseret på confusion-matricerne er det svært at drage en endelig konklusion om, hvilken model der præsterer bedst. Dog kan der drages nogle observationer:

- Rpart-modellen har klassificeret flere tilfælde forkert end de andre modeller for begge klasser.
- Random Forest-modellen ser ud til at præstere bedre end de andre modeller for "aa"-klassen, men lidt dårligere for "ao"-klassen.
- Ada-modellen ser ud til at præstere bedre end de andre modeller for "ao"-klassen, men lidt dårligere for "aa"-klassen.
- GBM-modellen ser ud til at præstere lignende som de andre modeller for begge klasser.

(Personligt ville jeg måske mene at Random Forest-modellen virker bedst)

Modellerne er kalibreret på følgende parametre:

- rpart: Standardværdierne for funktionen `rpart.control()`, som kaldes af `rpart()` medmindre andet er specificeret, er `maxdepth = 30`, `minsplit = 20`, `cp = 0.01` og `xval = 10`. Disse styrer den maksimale dybde af træet, det minimale antal observationer, der kræves for at lave en splittelse, kompleksitetsparametret til brug for beskæring af træet og antallet af krydsvalideringsfolds.
- randomForest: Standardværdien for antallet af træer er `ntree = 500`. Andre vigtige hyperparametre, der kan specificeres, omfatter antallet af variabler, der tilfældigt udvælges ved hver splittelse (mtry), og dybden af træerne (maxdepth).
- ada: Standardværdien for antallet af træer er `iter = 50`. Andre vigtige hyperparametre inkluderer læringshastigheden (nu) og den maksimale dybde af træerne (maxdepth).
- gbm: Standardværdien for antallet af træer er `n.trees = 100`. Andre vigtige hyperparametre inkluderer læringshastigheden (shrinkage) og interaktionsdybden af træerne (interaction.depth).

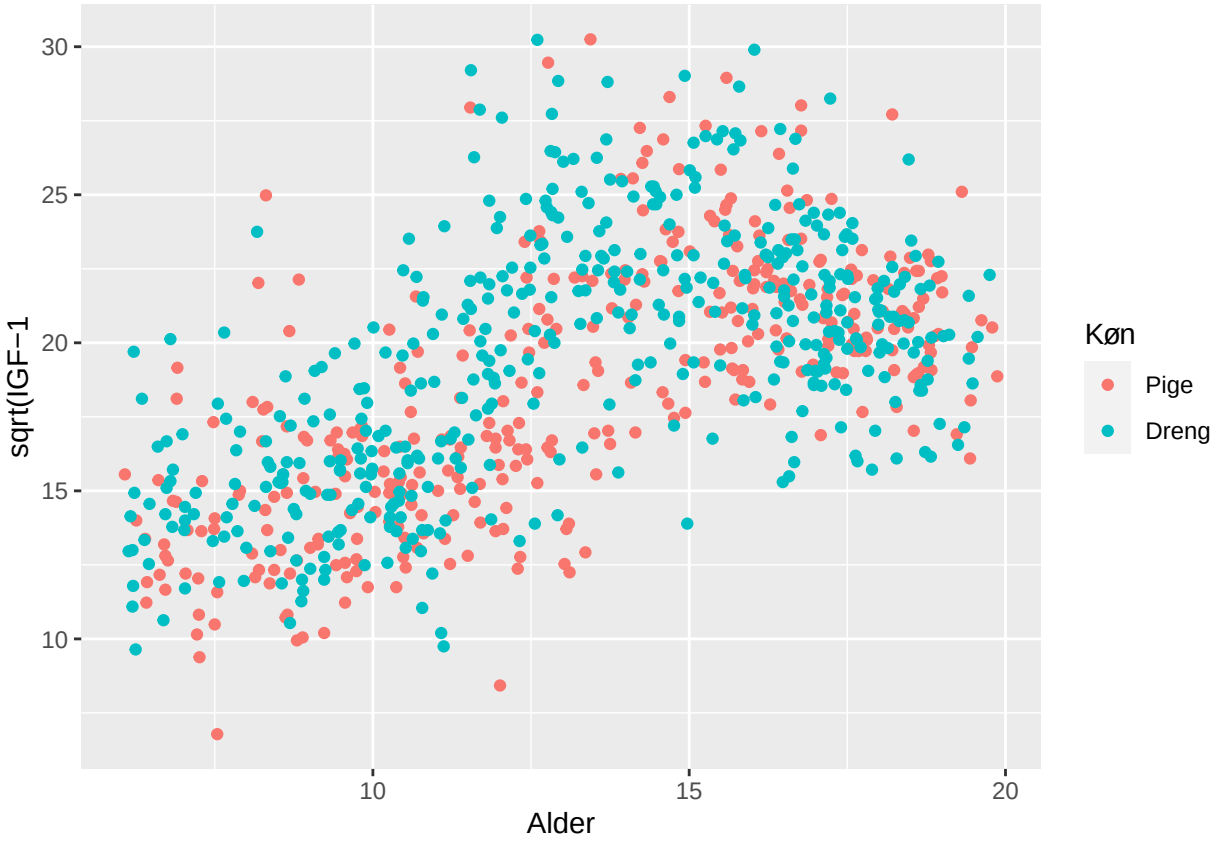
Det er vigtigt at bemærke, at standardværdierne muligvis ikke er optimale for et givet datasæt, og tuning af disse hyperparametre ved hjælp af teknikker som krydsvalidering kan føre til en bedre ydeevne af modellen.

Opgave 3

Vi ser på `juul` datasættet og udtrækker personer mellem 6 og 20 år.

1

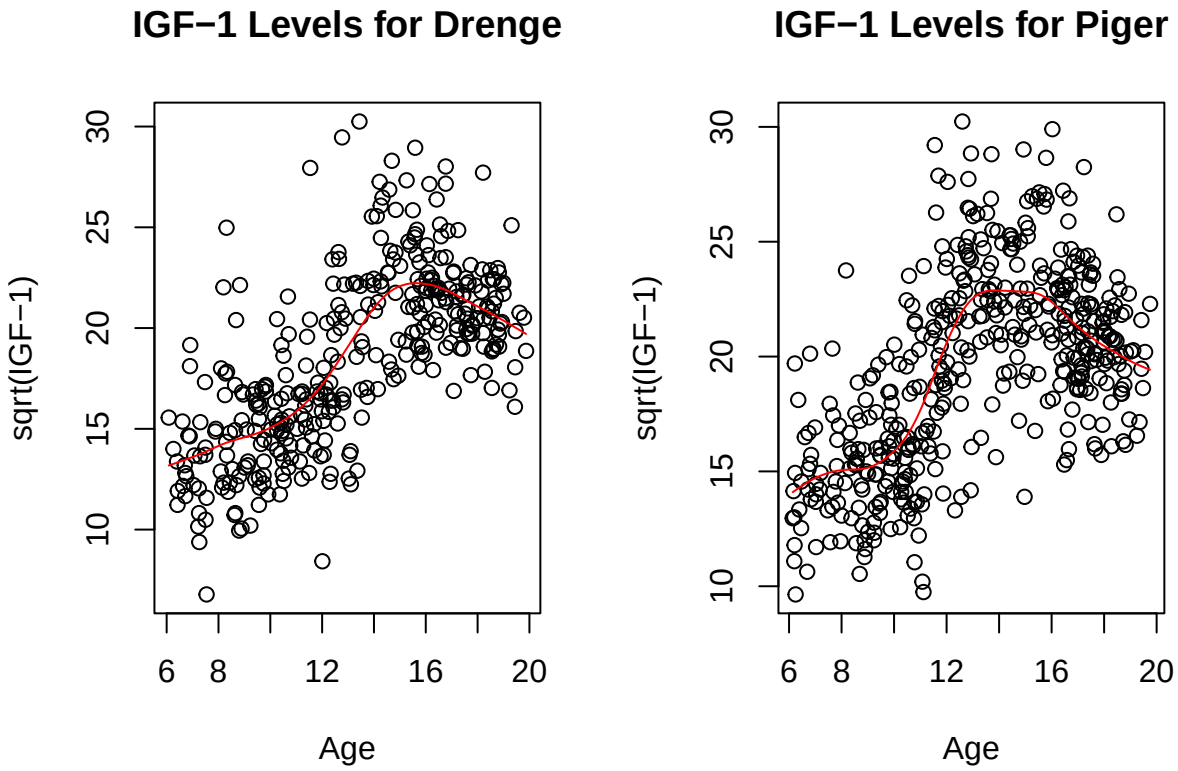
Jeg plotter `sqrt(igf1)` mod `age` med indikation af køn:



Det ser ud til at IGF1 (Insulin-like Growth Factor 1), stiger med alderen for begge køn og falder efter omkring de 15-16 år.

2

Jeg indtegner nu en smoothing spline for hvert køn:

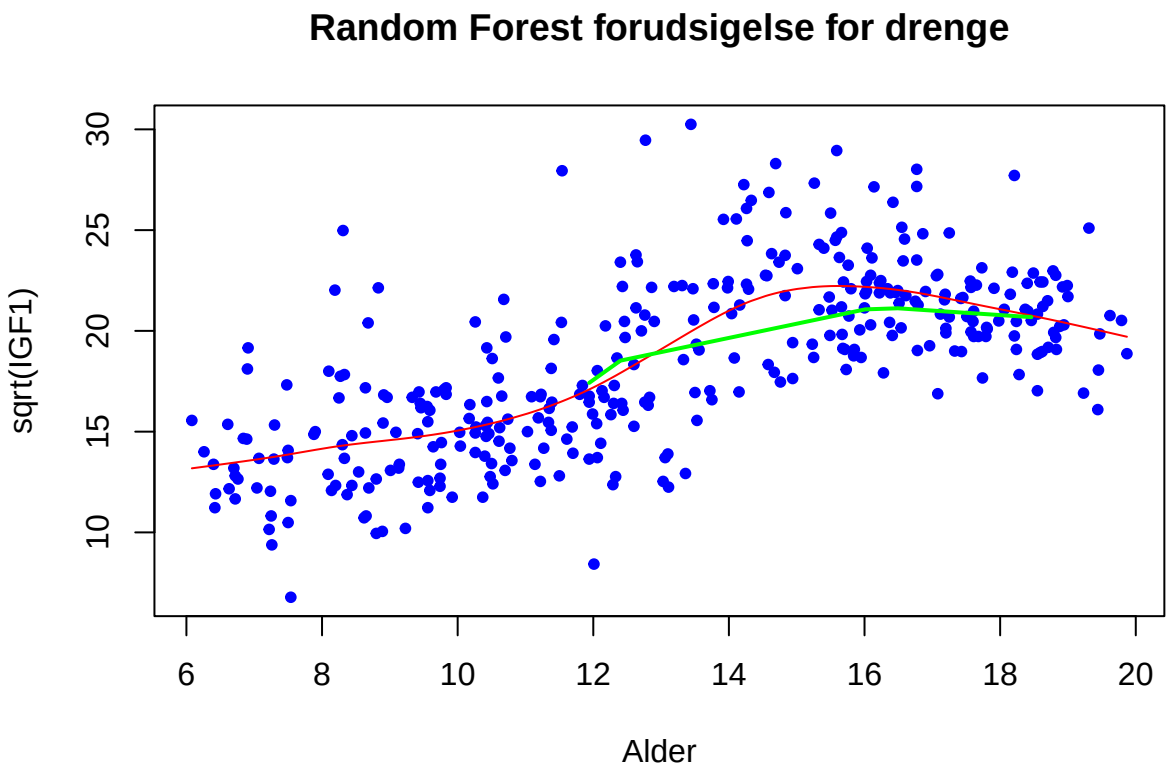


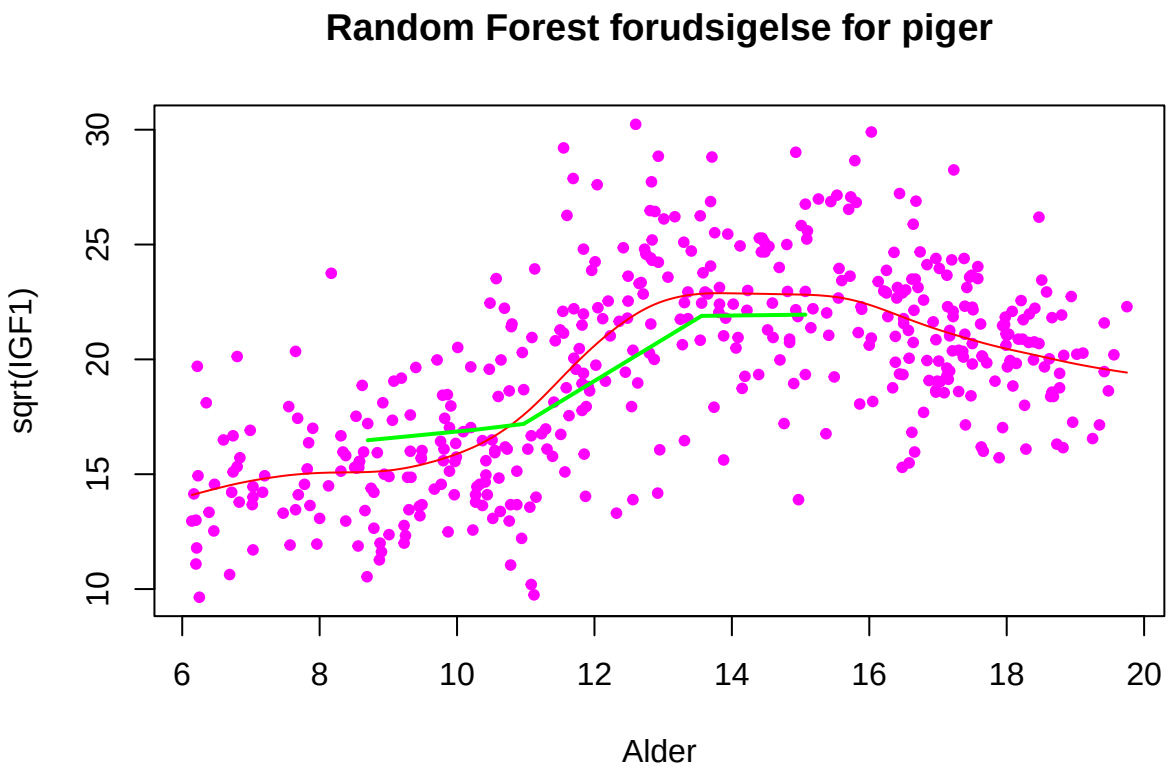
Plotsene viser nogenlune det samme og dermed også splines.

3

Jeg laver nu samme analyse med `randomForest` hvor jeg fitter `age` og `sex` som prædiktorer og tegner de prædikterede værdier:

```
##      |      Out-of-bag   |
## Tree |      MSE  %Var(y) |
##  100 | 1.538e+04    56.24 |
##  200 | 1.513e+04    55.34 |
##  300 | 1.517e+04    55.50 |
##  400 | 1.527e+04    55.86 |
##  500 | 1.517e+04    55.49 |
```


Random Forest forudsigelse for drenge
sqrt(IGF1)
10
15
20
25
30
6
8
10
12
14
16
18
20
Alder

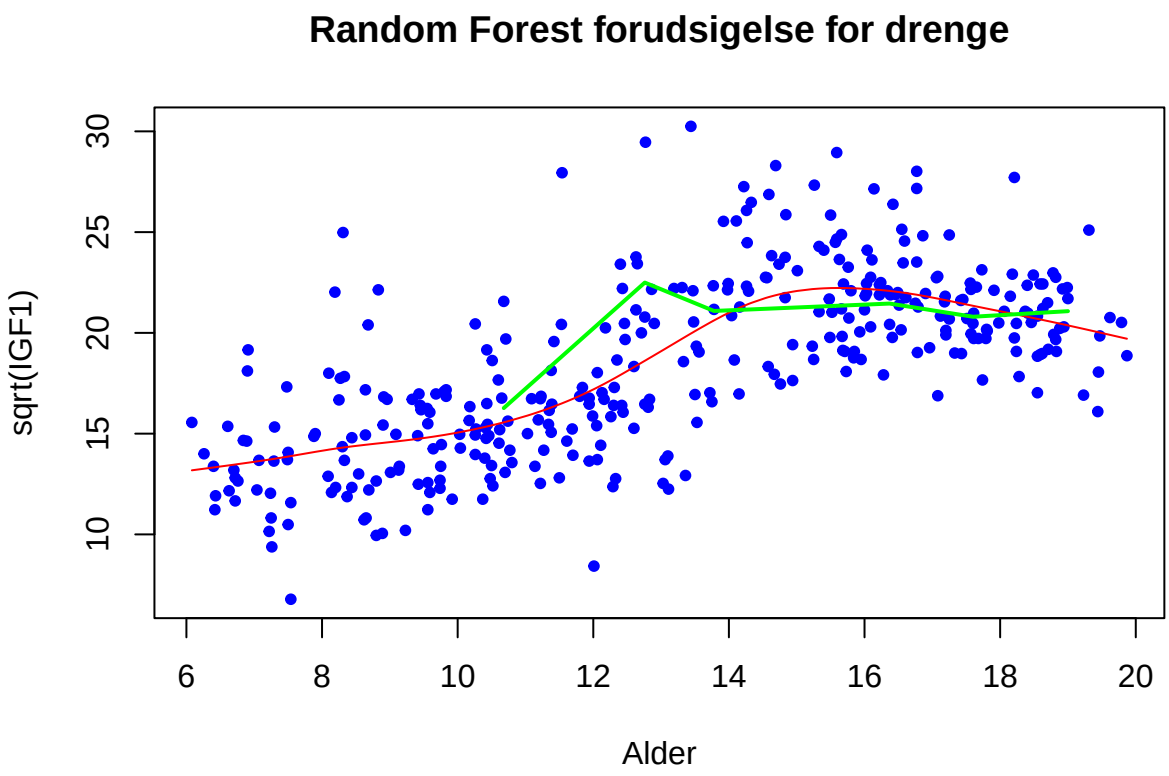


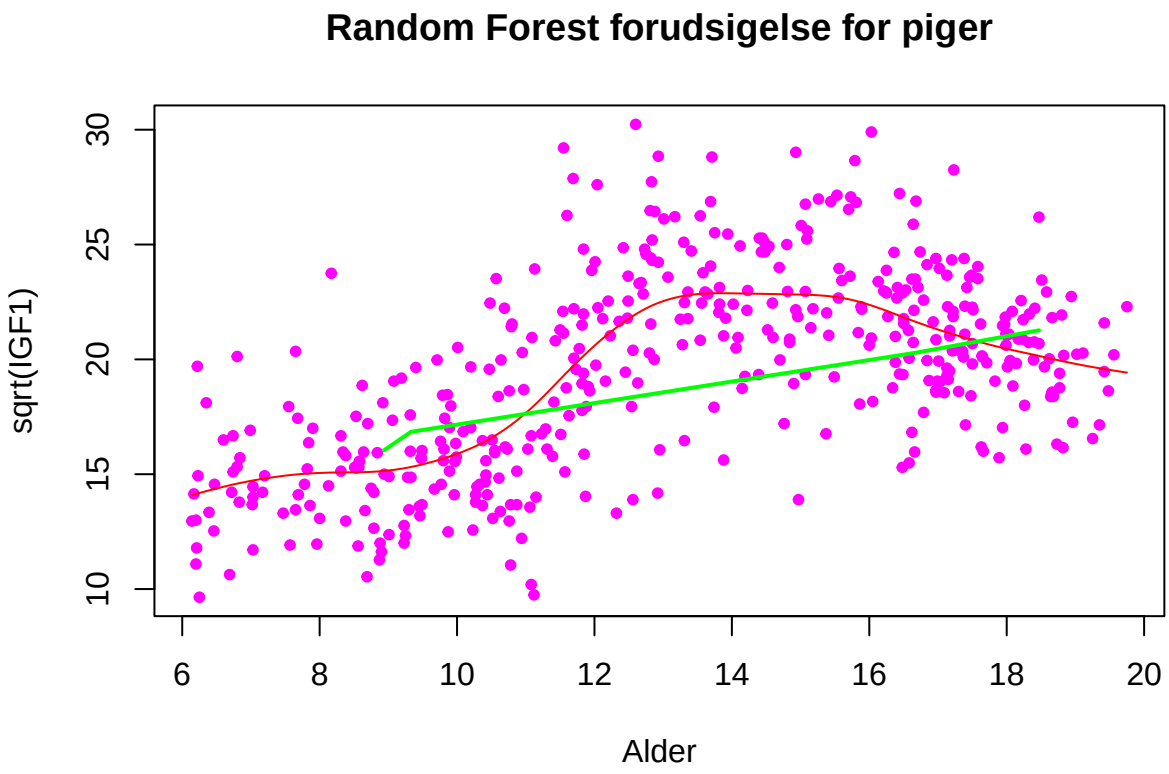
Modellen virker til umiddelbart at prædikterer drengenes niveau af IGF-1 generelt lidt højere end pigernes.

4

Jeg sætter `ntree = 1000` (standard er 500) og `mtry = 3` (standard er $\sqrt{p}$) for at se på om det ændrer prædiktionsevnen.

```
##      |      Out-of-bag   |
## Tree |      MSE  %Var(y) |
##  100 | 1.81e+04    66.45 |
##  200 | 1.803e+04    66.20 |
##  300 | 1.805e+04    66.27 |
##  400 | 1.802e+04    66.15 |
##  500 | 1.796e+04    65.97 |
##  600 | 1.796e+04    65.96 |
##  700 | 1.797e+04    65.97 |
##  800 | 1.795e+04    65.93 |
##  900 | 1.793e+04    65.86 |
## 1000 | 1.792e+04    65.81 |
```


Random Forest forudsigelse for drenge
sqrt(IGF1)
10
15
20
25
30
6
8
10
12
14
16
18
20
Alder



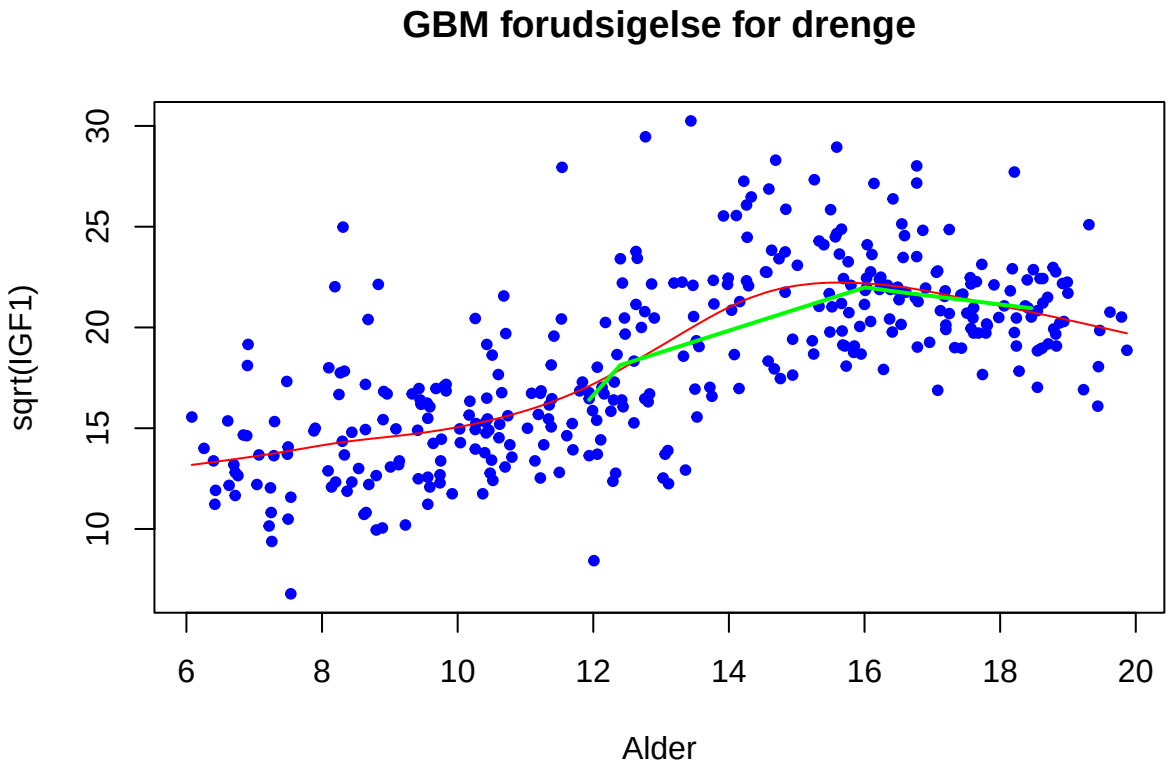
Prædiktionerne virker til at blive mere lineær men for drengene virker det lidt som overfit. Det skyldes at vi har inkluderet flere træer og flere tilfældige udvalgte parametre.

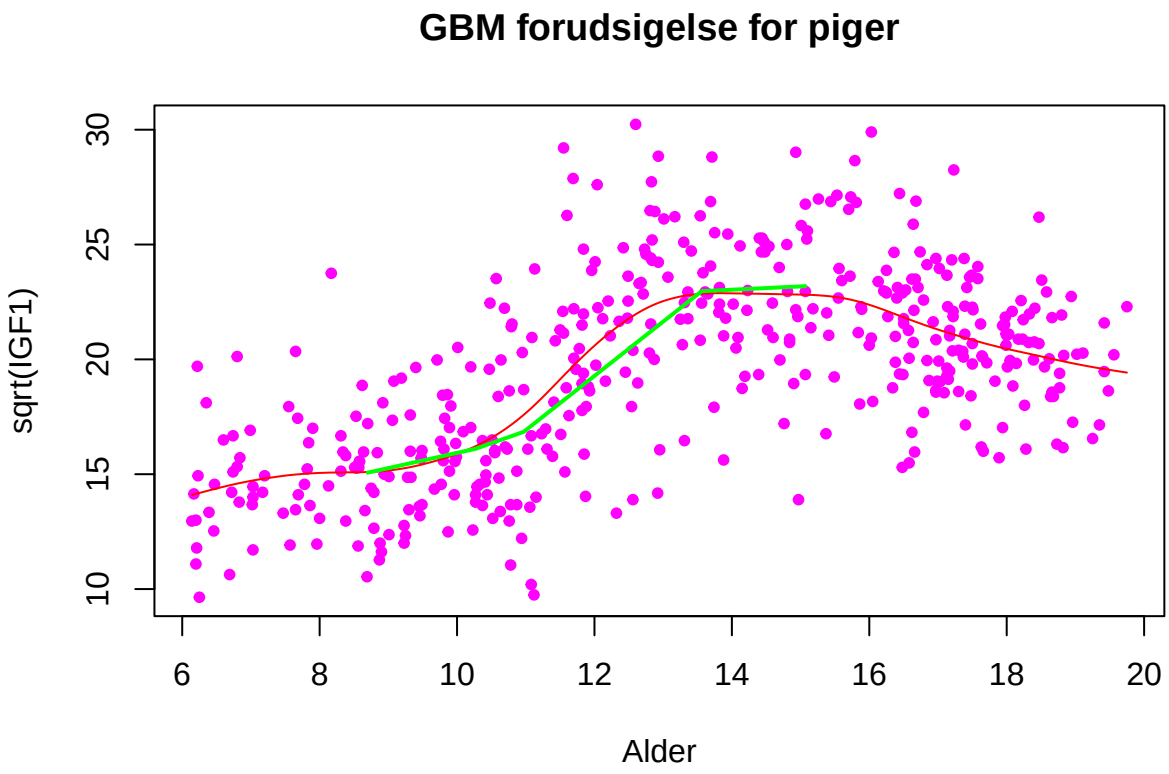
5

Vi laver samme analyse igen men denne gang med et `gbm` fit (gradient boosting). Antallet af træer er sat til 500:

```
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1    27084.0348             nan     0.0100  250.1650
##      2    26831.1615             nan     0.0100  251.4600
##      3    26579.7859             nan     0.0100  233.5126
##      4    26323.0677             nan     0.0100  234.5228
##      5    26090.1834             nan     0.0100  241.0377
##      6    25858.2825             nan     0.0100  235.9886
##      7    25631.3605             nan     0.0100  216.4407
##      8    25410.0139             nan     0.0100  213.7471
##      9    25177.5171             nan     0.0100  212.9641
##     10    24962.8664             nan     0.0100  219.5679
##     20    22995.3322             nan     0.0100  168.8915
##     40    19959.9157             nan     0.0100  113.3995
##     60    17918.8702             nan     0.0100   73.3476
##     80    16499.9344             nan     0.0100   54.0785
##    100    15467.1604             nan     0.0100   20.6405
##    120    14743.1973             nan     0.0100   22.3893
```

```
##     140    14240.4982             nan     0.0100   14.6997
##     160    13871.5836             nan     0.0100   11.8121
##     180    13571.6176             nan     0.0100    5.6833
##     200    13374.1533             nan     0.0100    1.3541
##     220    13218.5619             nan     0.0100    0.9010
##     240    13081.1843             nan     0.0100   -0.6512
##     260    12971.3291             nan     0.0100   -0.5016
##     280    12879.5026             nan     0.0100   -1.9723
##     300    12805.3036             nan     0.0100    0.3845
##     320    12727.5457             nan     0.0100    1.8863
##     340    12666.1320             nan     0.0100   -1.5928
##     360    12609.0863             nan     0.0100   -5.6894
##     380    12557.9776             nan     0.0100   -0.7051
##     400    12510.5384             nan     0.0100   -1.7105
##     420    12463.8181             nan     0.0100   -3.3629
##     440    12418.3815             nan     0.0100   -9.8912
##     460    12377.3937             nan     0.0100   -6.4281
##     480    12342.4109             nan     0.0100   -5.9128
##     500    12307.9405             nan     0.0100   -1.7273
```





Umiddelbart virker det til at Gradient Boosting performer bedre end Random Forest i det her tilfælde (særligt for pigerne)