Nonlinear

JunLu

4/1/2019

Import the train dataset

```
train = read_csv("./train.csv")

y = train$transformed_value
options(na.action = 'na.pass')
x = model.matrix(transformed_value ~ ., train)[,-1]
```

Generallized additive model (GAM)

Use caret package

```
set.seed(2)
ctrl1 <- trainControl(method = "cv", number = 10)
gam.fit = train(x, y,
                preProcess = "medianImpute",
                method = "gam",
                tuneGrid = data.frame(method = "GCV.Cp",
                                      select = c(TRUE,FALSE)),
                trControl = ctrl1)

save(gam.fit, file = "./gam_fit.rda")
```

```
load(file = "./gam_fit.rda")
gam.fit$bestTune
```

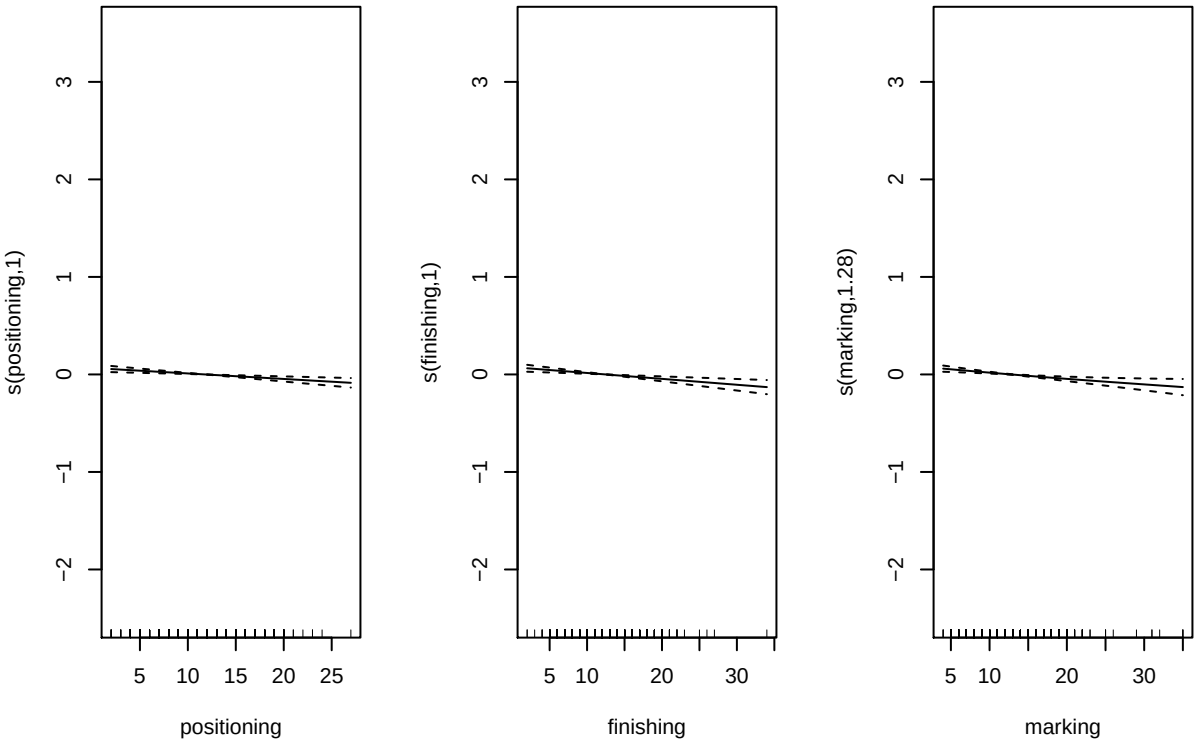
```
##   select method
## 1  FALSE GCV.Cp
```

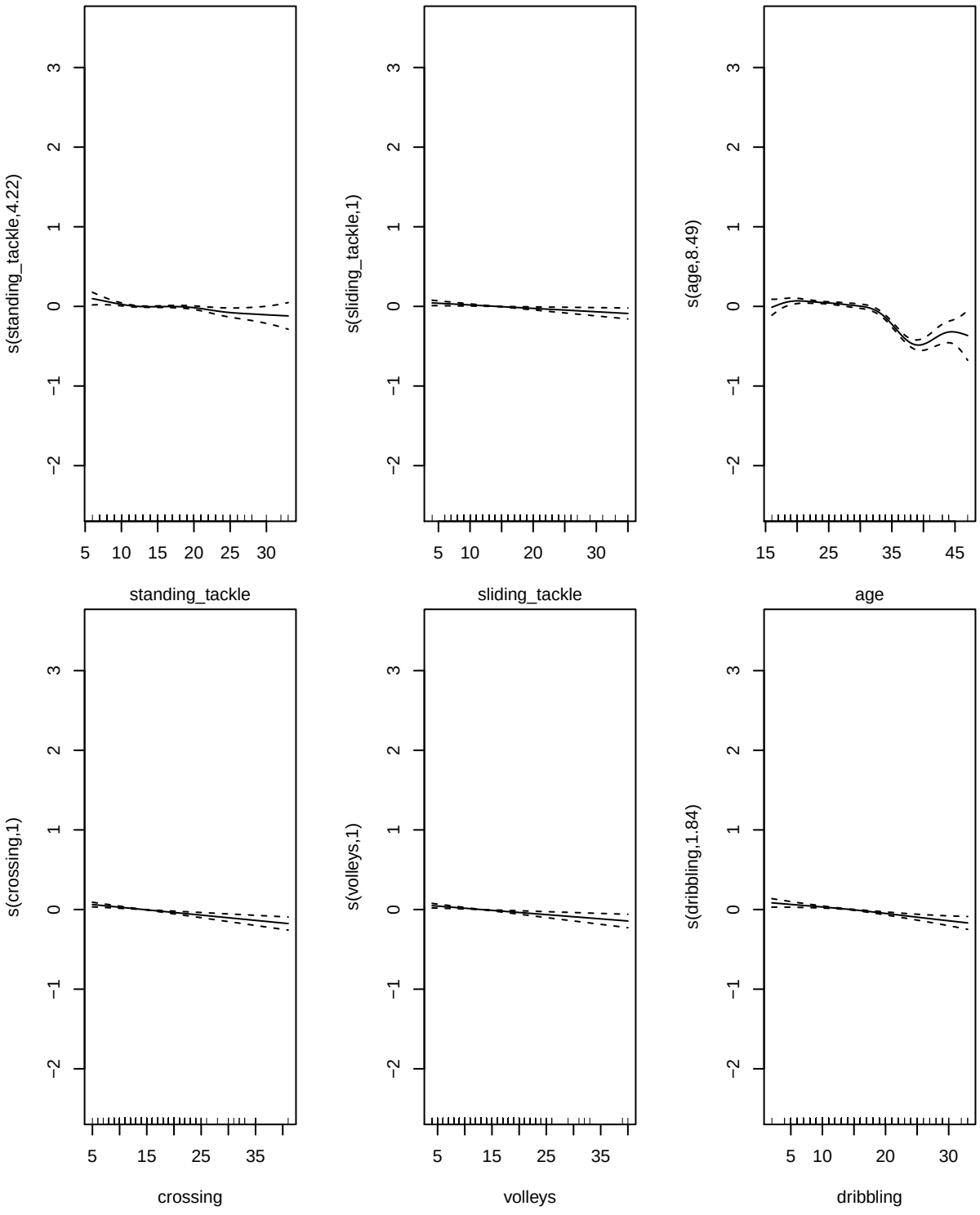
```
gam.fit$finalModel
```

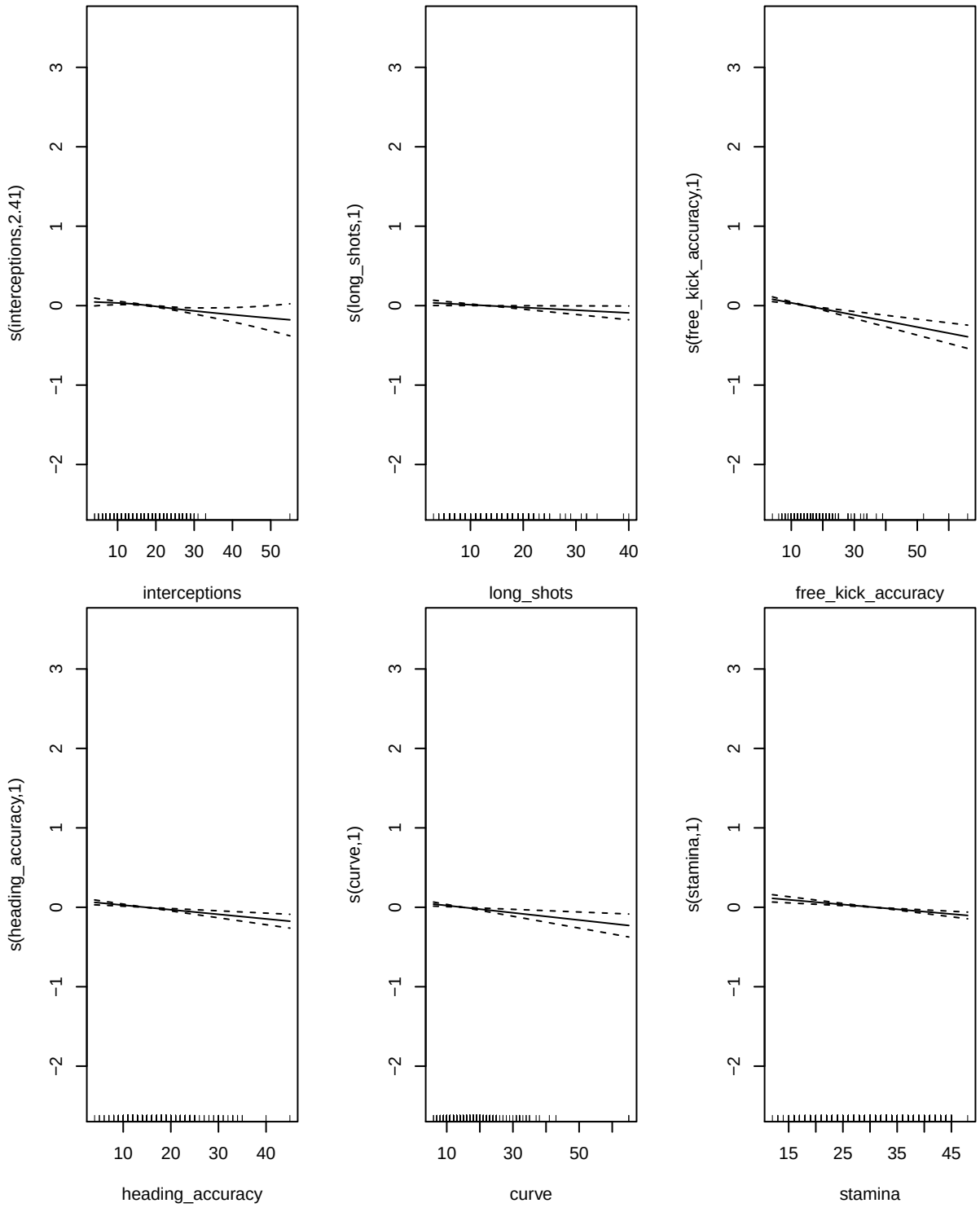
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## .outcome ~ nationalityas + nationalityeu + nationalitysa + s(positioning) +
##     s(finishing) + s(marking) + s(standing_tackle) + s(sliding_tackle) +
##     s(age) + s(crossing) + s(volleys) + s(dribbling) + s(interceptions) +
##     s(long_shots) + s(free_kick_accuracy) + s(heading_accuracy) +
##     s(curve) + s(stamina) + s(aggression) + s(ball_control) +
##     s(potential) + s(gk_handling) + s(short_passing) + s(gk_diving) +
##     s(gk_reflexes) + s(penalties) + s(gk_kicking) + s(gk_positioning) +
##     s(long_passing) + s(shot_power) + s(acceleration) + s(sprint_speed) +
##     s(balance) + s(reactions) + s(agility) + s(composure) + s(strength) +
##     s(vision) + s(jumping) + s(special)
```

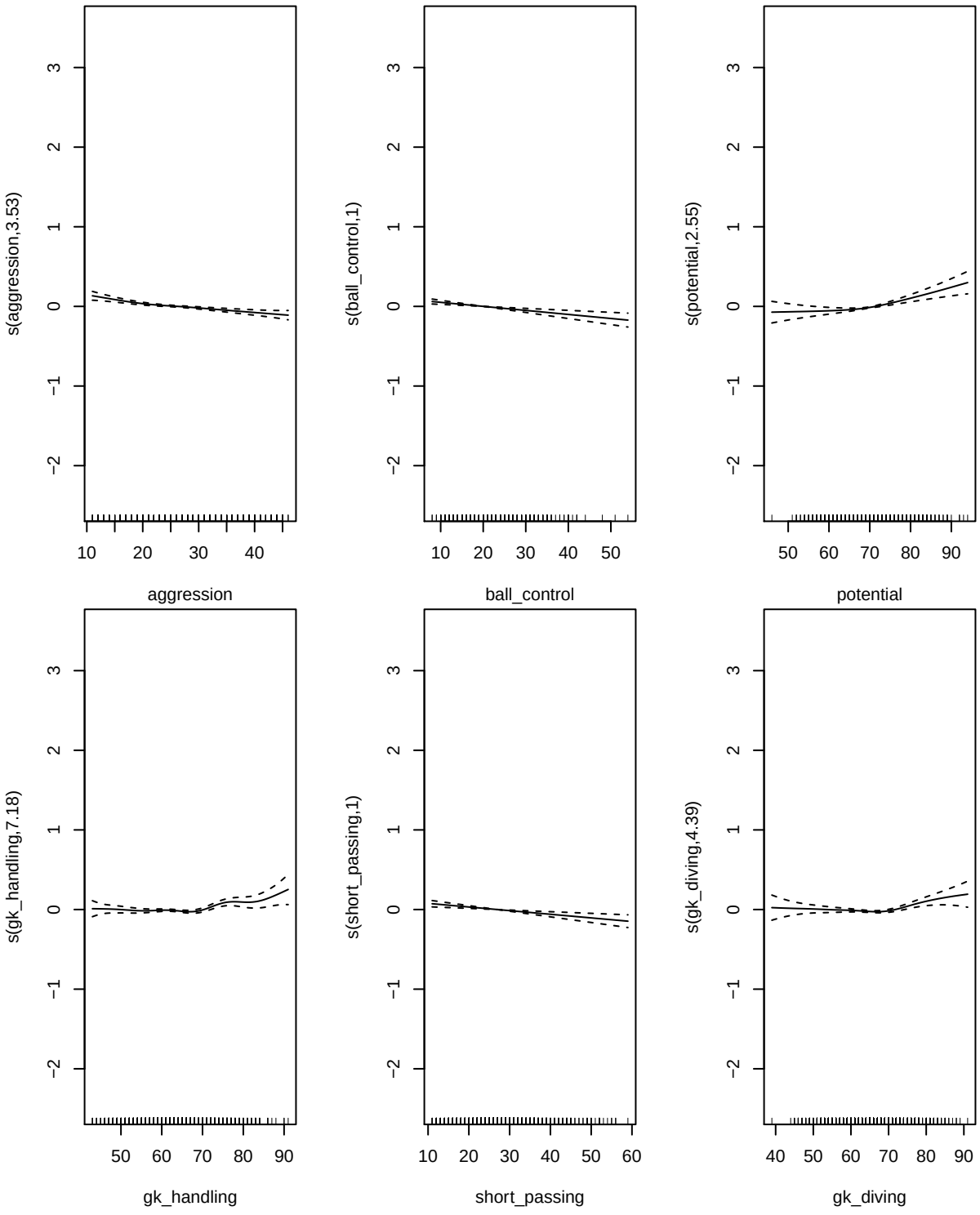
```
##
## Estimated degrees of freedom:
## 1.00 1.00 1.28 4.22 1.00 8.49 1.00
## 1.00 1.84 2.41 1.00 1.00 1.00 1.00
## 1.00 3.53 1.00 2.55 7.18 1.00 4.39
## 2.97 8.90 6.35 1.00 1.00 2.68 1.00
## 1.00 2.67 2.89 1.40 1.00 3.88 8.78
## 1.00 2.88  total = 101.29
##
## GCV score: 0.01923898
```

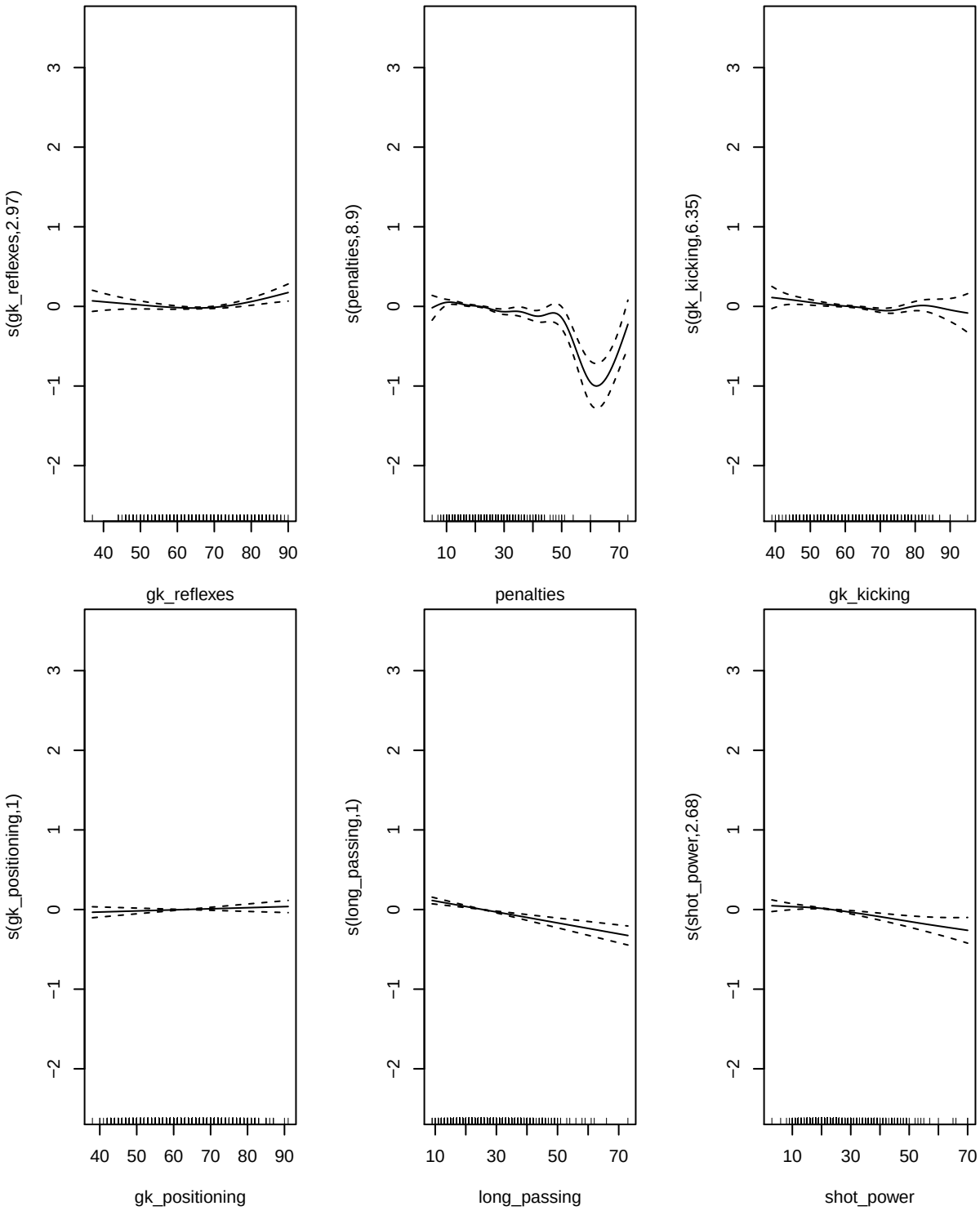
```r
par(mfrow = c(1,3))
plot(gam.fit$finalModel)
```

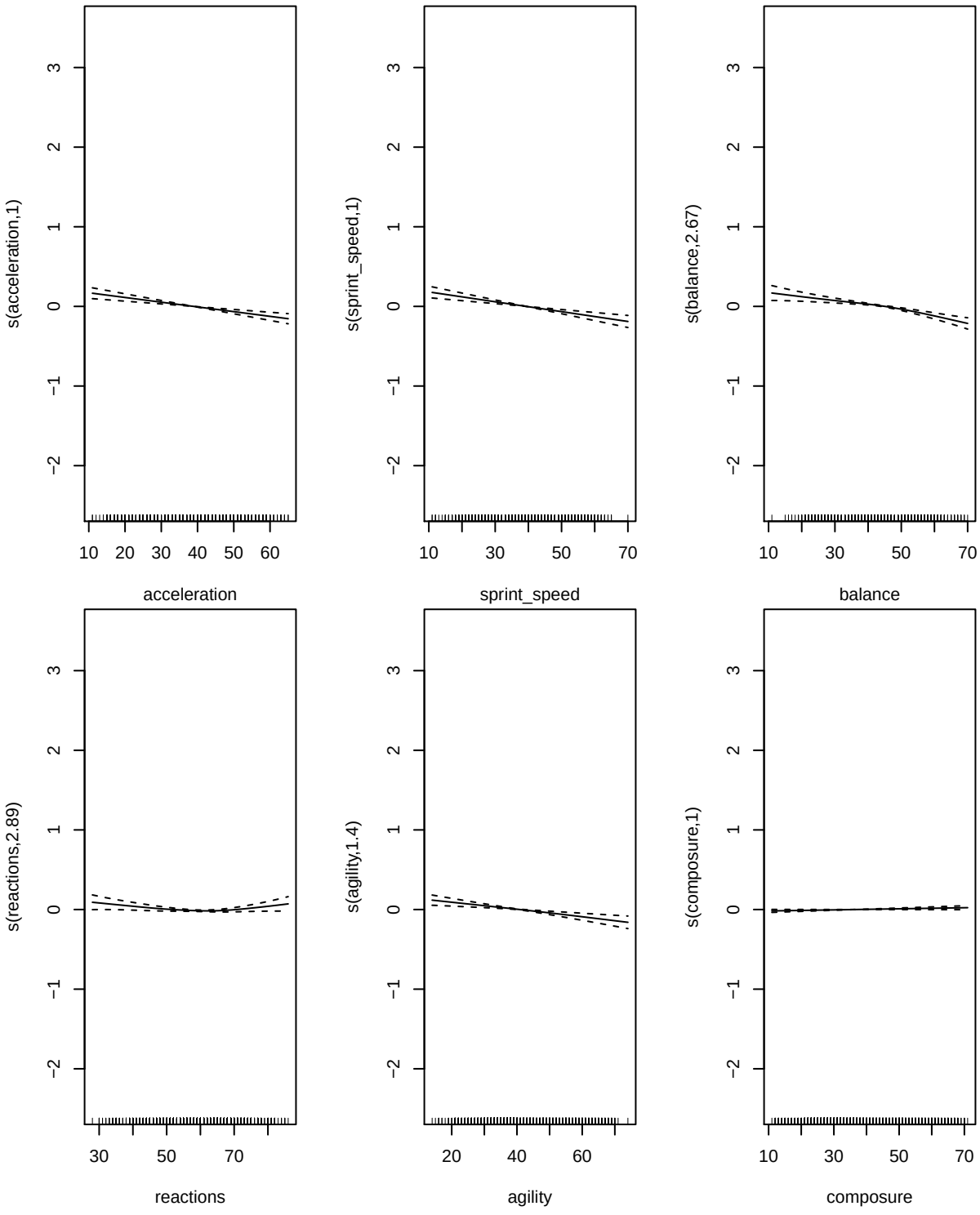


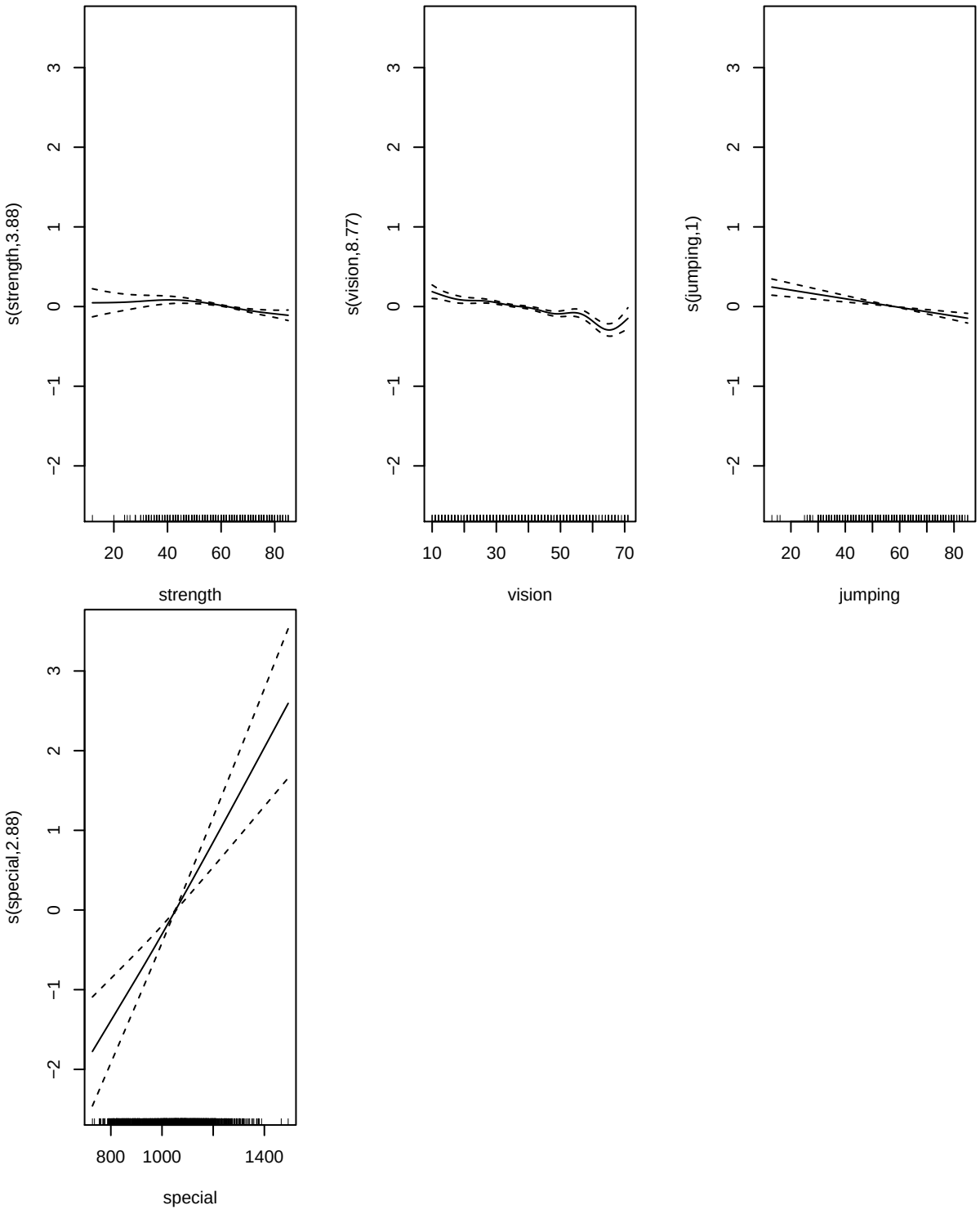

s(standing_tackle,4.22)
standing_tackle
s(sliding_tackle,1)
sliding_tackle
s(age,8.49)
age
s(crossing,1)
crossing
s(volleys,1)
volleys
s(dribbling,1.84)
dribbling


s(interceptions,2.41)
interceptions
s(long_shots,1)
long_shots
s(free_kick_accuracy,1)
free_kick_accuracy
s(heading_accuracy,1)
heading_accuracy
s(curve,1)
curve
s(stamina,1)
stamina


s(aggression,3.53)
aggression
s(ball_control,1)
ball_control
s(potential,2.55)
potential
s(gk_handling,7.18)
gk_handling
s(short_passing,1)
short_passing
s(gk_diving,4.39)
gk_diving


s(gk_reflexes,2.97)
gk_reflexes
s(penalties,8.9)
penalties
s(gk_kicking,6.35)
gk_kicking
s(gk_positioning,1)
gk_positioning
s(long_passing,1)
long_passing
s(shot_power,2.68)
shot_power


s(acceleration,1)
acceleration
s(sprint_speed,1)
sprint_speed
s(balance,2.67)
balance
s(reactions,2.89)
reactions
s(agility,1.4)
agility
s(composure,1)
composure



Multivariate Adaptive Regression Splines (MARS)

```
set.seed(2)
ctrl1 <- trainControl(method = "cv", number = 10)
mars_grid = expand.grid(degree = 1:2,
```

```
                        nprune = 2:38)
mars.fit = train(x, y,
                 method = "earth",
                 preProcess = "medianImpute",
                 tuneGrid = mars_grid,
                 trControl = ctrl1
                 )
save(mars.fit, file = "./earth.rda")
```

```
load(file = "./earth.rda")
summary(mars.fit)
```

```
## Call: earth(x=matrix[1523,42], y=c(2.795,2.772,1...), keepxy=TRUE,
##             degree=1, nprune=18)
##
##                       coefficients
## (Intercept)             0.70326337
## nationalityeu           0.04077075
## h(age-29)              -0.02402008
## h(age-33)              -0.05729609
## h(age-39)               0.12757144
## h(71-potential)        -0.00539790
## h(potential-71)         0.01690058
## h(61-agility)          -0.00171512
## h(balance-47)          -0.00319876
## h(68-gk_diving)        -0.00717104
## h(gk_diving-68)         0.01607073
## h(67-gk_handling)      -0.00482717
## h(gk_handling-67)       0.01440001
## h(gk_kicking-72)        0.00936914
## h(gk_positioning-43)    0.00790612
## h(gk_reflexes-71)       0.01509622
## h(reactions-63)         0.01070404
## h(41-strength)         -0.01290410
##
## Selected 18 of 28 terms, and 12 of 42 predictors
## Termination condition: RSq changed by less than 0.001 at 28 terms
## Importance: gk_diving, age, gk_positioning, potential, reactions, ...
## Number of terms at each degree of interaction: 1 17 (additive model)
## GCV 0.02047396    RSS 29.76514    GRSq 0.8598615    RSq 0.8660527
```

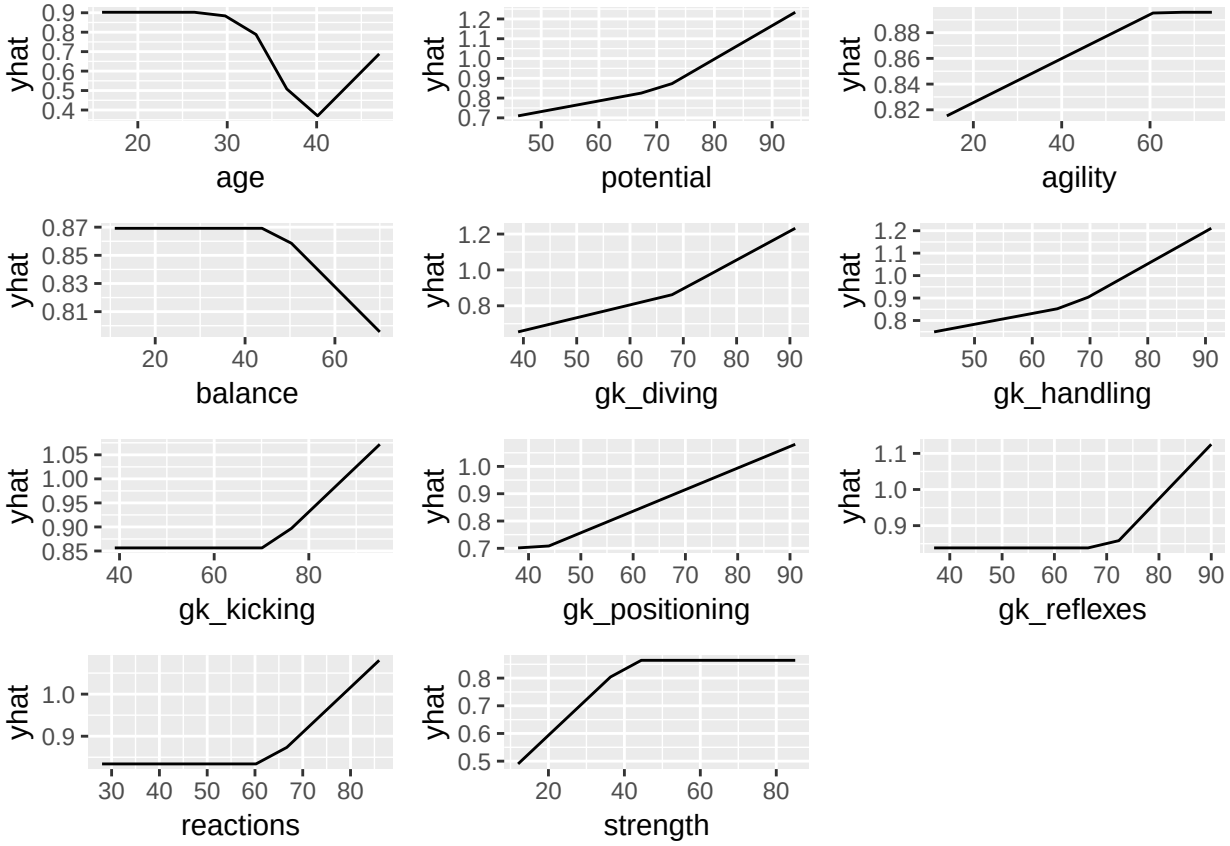
```
mars.fit$bestTune
```

```
##    nprune degree
## 17     18      1
```

```
p1 = partial(mars.fit, pred.var = c("age"), grid.resolution = 10) %>% autoplot()
p2 = partial(mars.fit, pred.var = c("potential"), grid.resolution = 10) %>% autoplot()
p3 = partial(mars.fit, pred.var = c("agility"), grid.resolution = 10) %>% autoplot()
p4 = partial(mars.fit, pred.var = c("balance"), grid.resolution = 10) %>% autoplot()
p5 = partial(mars.fit, pred.var = c("gk_diving"), grid.resolution = 10) %>% autoplot()
p6 = partial(mars.fit, pred.var = c("gk_handling"), grid.resolution = 10) %>% autoplot()
p7 = partial(mars.fit, pred.var = c("gk_kicking"), grid.resolution = 10) %>% autoplot()
p8 = partial(mars.fit, pred.var = c("gk_positioning"), grid.resolution = 10) %>% autoplot()
p9 = partial(mars.fit, pred.var = c("gk_reflexes"), grid.resolution = 10) %>% autoplot()
```

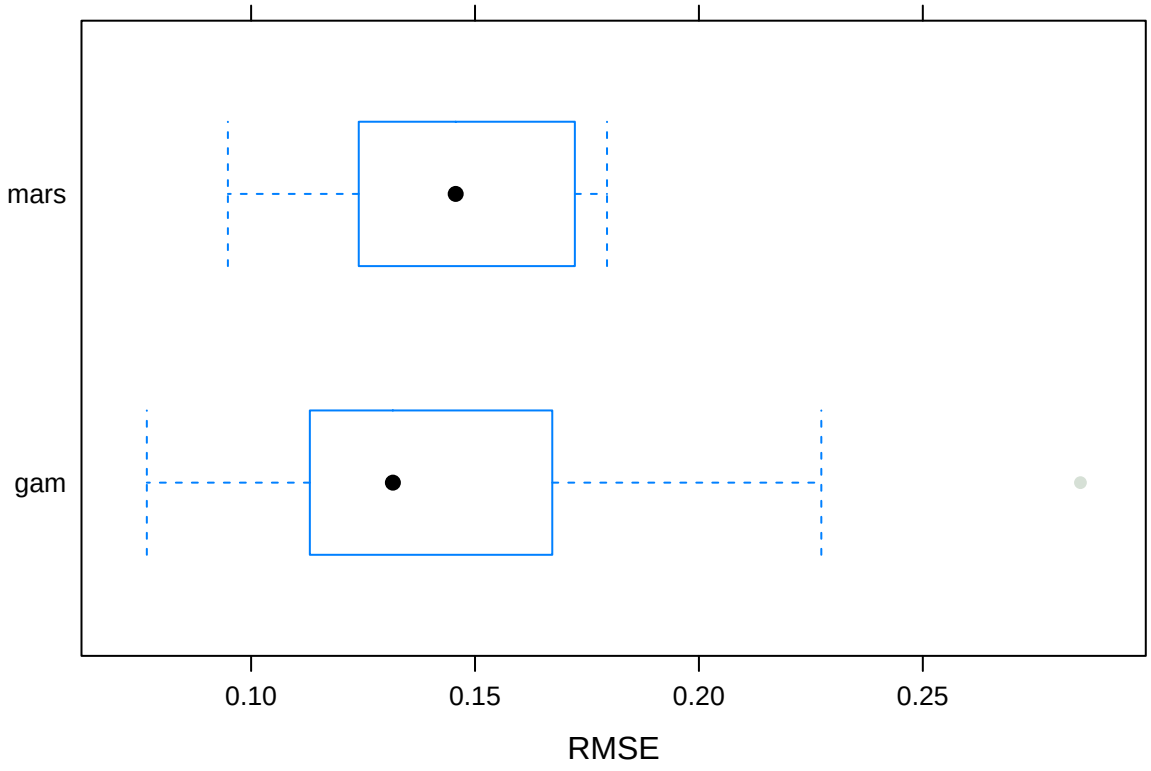
```
p10 = partial(mars.fit, pred.var = c("reactions"), grid.resolution = 10) %>% autoplot()
p11 = partial(mars.fit, pred.var = c("strength"), grid.resolution = 10) %>% autoplot()

grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, ncol = 3, nrow = 4)
```



Compare those models

```
bwplot(resamples(list(mars = mars.fit,gam = gam.fit)),
       metric = "RMSE")
```



Importance

```
varImp(mars.fit)
```

```
## earth variable importance
##
##   only 20 most important variables shown (out of 42)
##
##                     Overall
## gk_diving          100.000
## age                 44.619
## gk_positioning      44.619
## gk_reflexes         32.682
## potential           30.563
## reactions           24.640
## gk_handling         18.626
## strength            14.856
## nationalityeu       10.442
## gk_kicking           7.197
## agility              3.178
## balance              3.178
## free_kick_accuracy   0.000
## special              0.000
## long_shots           0.000
## aggression           0.000
## acceleration         0.000
## ball_control         0.000
## long_passing         0.000
```

```
## stamina              0.000
```

```
varImp(gam.fit)
```

```
## gam variable importance
##
##   only 20 most important variables shown (out of 40)
##
##                    Overall
## age                100.000
## vision               9.209
## penalties            7.638
## balance              7.021
## long_passing         6.349
## special              6.112
## free_kick_accuracy   6.105
## sprint_speed         5.358
## potential            5.107
## stamina              5.073
## acceleration         5.070
## jumping              4.968
## dribbling            4.891
## strength             4.882
## gk_handling          4.527
## aggression           4.331
## crossing             3.954
## gk_diving            3.585
## heading_accuracy     3.500
## ball_control         3.362
```

Questions

1. As we can't use the test dataset to choose our final model, do we need to calculate test error for each model or just our final model.

Do we write (linear -> nonlinear, if we know it is nonlinear why we try linear firstly)

depend on oursevles

2. How to know our model is good or not, just with a train error and a test error. (close and both small -> good, train << test overfitted, both large underfitted) However, how do we know it is small or large (MSE also depends on y)

can't

3. Which variables are important? Different models have different important variables. Do we only use the result of our final model?
4. how do we present our model in the report (coefficients? tunning parameter? plot?)
5. Model assumptions and limitations

linear Multi linear: model linear in parameter (error term mean 0 constant variance uncorrelated)

lasso and ridge: linear (Anything else? Multicollinearity?)

pcr: linear (Anything else? Multicollinearity?)

non-linear gam: Nonlinear relationship (Anything else?) mars: Nonlinear relationship (Anything else)

6. How to choose tunning parameters? like we try lambda from (eˆ-10, eˆ10), then we choose the best lambda by 10-fold cross vaildation.

7.How do we know model is enough flexible? How do we make prediction?

$$Y_{transformed} = 0.703 + 0.041X_{Europe} - 0.024h(X_{age} - 29) - 0.057h(X_{age} - 33) + 0.128h(X_{age} - 39)$$

$$-0.005h(71-X_{potential})+0.017h(X_{potential}-71)-0.002h(61-X_{agility})-0.003h(X_{balance}-47)-0.007h(68-X_{gk\_diving})$$

$$+0.016h(X_{gk\_diving}-68)-0.005h(67-X_{gk\_handling})+0.014h(X_{gk\_handling}-67)+0.009h(X_{gk\_kicking}-72)$$

$$+0.008h(X_{gk\_positioning}-43)+0.015h(X_{gk\_reflexes}-71)+0.011h(X_{reactions}-63)-0.013h(41-X_{strength})$$

$$h(x) = x_+$$