concrete_data

Shiva Sankar Modala

2023-03-01

```r
#install.packages("tidyverse")

# readxl packages to load Excel data
#install.packages("readxl")
#install.packages("magrittr")
#install.packages("corrplot")

# Use the mgcv package to create a generalized additive model
#install.packages("mgcv")

# Visualize the regression using the visreg package,
#install.packages("visreg")

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.0     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.1     ✓ tibble    3.1.8
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force
all conflicts to become errors

library(readxl)
library(magrittr)

##
## Attaching package: 'magrittr'
##
## The following object is masked from 'package:purrr':
##
##     set_names
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

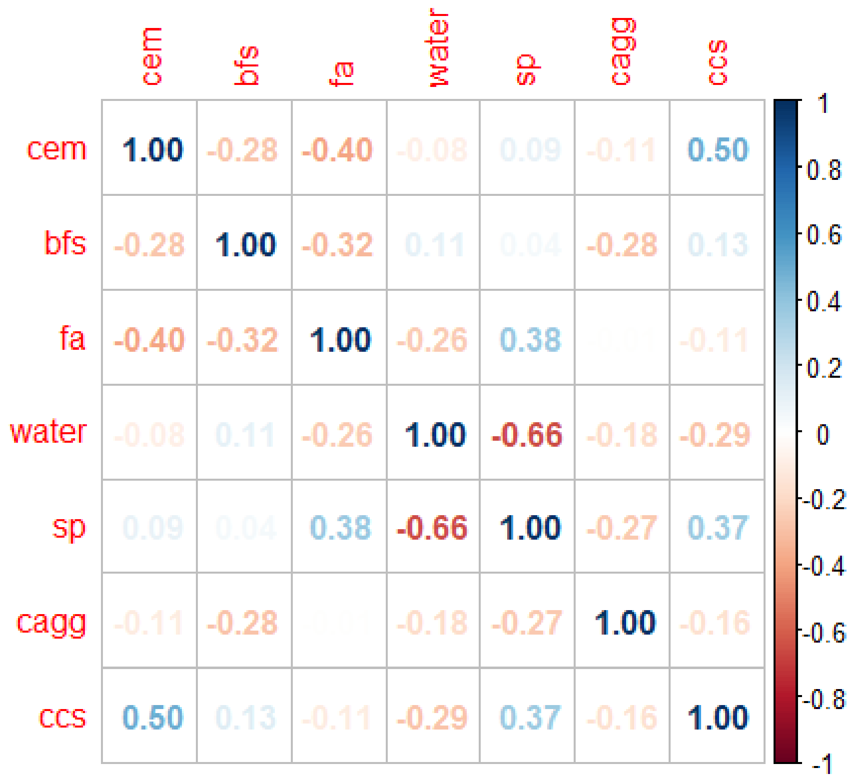
```r
library(corrplot)

## corrplot 0.92 loaded

library(mgcv)

## Loading required package: nlme
##
## Attaching package: 'nlme'
##
## The following object is masked from 'package:dplyr':
##
##     collapse
##
## This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.

library(visreg)

# Load the Concrete Compressive Strength sample dataset
concrete_data <- read_excel("C:/Users/shiva/OneDrive/Desktop/dpa
Assignments/Assignment3/Concrete_Data.xls")
summary(concrete_data)

##  Cement (component 1)(kg in a m^3 mixture)
##  Min.   :102.0
##  1st Qu.:192.4
##  Median :272.9
##  Mean   :281.2
##  3rd Qu.:350.0
##  Max.   :540.0
##  Blast Furnace Slag (component 2)(kg in a m^3 mixture)
##  Min.   :  0.0
##  1st Qu.:  0.0
##  Median : 22.0
##  Mean   : 73.9
##  3rd Qu.:142.9
##  Max.   :359.4
##  Fly Ash (component 3)(kg in a m^3 mixture)
##  Min.   :  0.00
##  1st Qu.:  0.00
##  Median :  0.00
##  Mean   : 54.19
##  3rd Qu.:118.27
##  Max.   :200.10
##  Water  (component 4)(kg in a m^3 mixture)
##  Min.   :121.8
##  1st Qu.:164.9
##  Median :185.0
##  Mean   :181.6
##  3rd Qu.:192.0
##  Max.   :247.0
```

```
##  Superplasticizer (component 5)(kg in a m^3 mixture)
##  Min.   : 0.000
##  1st Qu.: 0.000
##  Median : 6.350
##  Mean   : 6.203
##  3rd Qu.:10.160
##  Max.   :32.200
##  Coarse Aggregate  (component 6)(kg in a m^3 mixture)
##  Min.   : 801.0
##  1st Qu.: 932.0
##  Median : 968.0
##  Mean   : 972.9
##  3rd Qu.:1029.4
##  Max.   :1145.0
##  Fine Aggregate (component 7)(kg in a m^3 mixture)   Age (day)
##  Min.   :594.0                                      Min.   :  1.00
##  1st Qu.:731.0                                      1st Qu.:  7.00
##  Median :779.5                                      Median : 28.00
##  Mean   :773.6                                      Mean   : 45.66
##  3rd Qu.:824.0                                      3rd Qu.: 56.00
##  Max.   :992.6                                      Max.   :365.00
##  Concrete compressive strength(MPa, megapascals)
##  Min.   : 2.332
##  1st Qu.:23.707
##  Median :34.443
##  Mean   :35.818
##  3rd Qu.:46.136
##  Max.   :82.599
```

```
colnames(concrete_data) = c("cem", "bfs", "fa", "water", "sp", "cagg",
"fagg", "age", "ccs")
column_names = c("cem", "bfs", "fa", "water", "sp", "cagg", "ccs")
concrete_data = concrete_data[column_names]
summary(concrete_data)
```

```
##       cem             bfs              fa             water
##  Min.   :102.0   Min.   :  0.0   Min.   :  0.00   Min.   :121.8
##  1st Qu.:192.4   1st Qu.:  0.0   1st Qu.:  0.00   1st Qu.:164.9
##  Median :272.9   Median : 22.0   Median :  0.00   Median :185.0
##  Mean   :281.2   Mean   : 73.9   Mean   : 54.19   Mean   :181.6
##  3rd Qu.:350.0   3rd Qu.:142.9   3rd Qu.:118.27   3rd Qu.:192.0
##  Max.   :540.0   Max.   :359.4   Max.   :200.10   Max.   :247.0
##        sp              cagg             ccs
##  Min.   : 0.000   Min.   : 801.0   Min.   : 2.332
##  1st Qu.: 0.000   1st Qu.: 932.0   1st Qu.:23.707
##  Median : 6.350   Median : 968.0   Median :34.443
##  Mean   : 6.203   Mean   : 972.9   Mean   :35.818
##  3rd Qu.:10.160   3rd Qu.:1029.4   3rd Qu.:46.136
##  Max.   :32.200   Max.   :1145.0   Max.   :82.599
```

```
corrplot(cor(concrete_data), method = "number")
```




```
# gam function) to predict the Concrete Compressive Strength
dataModel1 <- gam(ccs ~ cem + bfs + fa + water + sp + cagg ,
data=concrete_data)
summary(dataModel1)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## ccs ~ cem + bfs + fa + water + sp + cagg
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.326997  10.510518   0.507 0.612387
## cem           0.108256   0.005214  20.761  < 2e-16 ***
## bfs           0.079357   0.006193  12.814  < 2e-16 ***
## fa            0.055928   0.009287   6.022  2.4e-09 ***
## water        -0.103871   0.027796  -3.737 0.000197 ***
## sp            0.356016   0.110251   3.229 0.001281 **
## cagg          0.008027   0.006272   1.280 0.200940
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
```

```
## R-sq.(adj) =  0.445   Deviance explained = 44.9%
## GCV = 155.83  Scale est. = 154.77    n = 1030
```

```r
# compare the R2 value for a GAM with linear terms as well as smoothed terms
cat("The corrected R-squared + shows that a sizable portion of the variation
is present, and it appears that we have statistical effects for CEM and BFS
but not for CAGG.")
```

```
## The corrected R-squared + shows that a sizable portion of the variation is
present, and it appears that we have statistical effects for CEM and BFS but
not for CAGG.
```

```r
# Use the s() function to apply smoothing using the default bs of tp).
dataModel2 <- gam(ccs ~ s(cem) + s(bfs) + s(fa) + s(water) + s(sp) + s(cagg)
, data=concrete_data)
summary(dataModel2)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## ccs ~ s(cem) + s(bfs) + s(fa) + s(water) + s(sp) + s(cagg)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  35.8178     0.3566   100.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##            edf Ref.df      F  p-value
## s(cem)   4.464  5.513 69.530  < 2e-16 ***
## s(bfs)   2.088  2.578 48.091  < 2e-16 ***
## s(fa)    5.332  6.404  1.784    0.101
## s(water) 8.567  8.936 13.504  < 2e-16 ***
## s(sp)    7.133  8.143  5.498 1.22e-06 ***
## s(cagg)  1.000  1.000  0.018    0.892
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.531   Deviance explained = 54.4%
## GCV = 134.84  Scale est. = 130.96    n = 1030
```

```r
cat("We should also remark that this model, with an adjusted R-squared
of.531, explains a large portion of the variance in CCS. In summary, it
appears that the CEM and CCS are connected.")
```

```
## We should also remark that this model, with an adjusted R-squared of.531,
explains a large portion of the variance in CCS. In summary, it appears that
the CEM and CCS are connected.
```

```r
#  showing the fit as a function of each predictor
dataModel1.sse <- sum(fitted(dataModel1)-concrete_data$ccs)^2
dataModel1.ssr <- sum(fitted(dataModel1) -mean(concrete_data$ccs))^2
dataModel1.sst = dataModel1.sse + dataModel1.ssr
Rsquared=1-(dataModel1.sse/dataModel1.sst)
cat(Rsquared)
```

```
## 0.4967177
```

```r
dataModel2.sse <- sum(fitted(dataModel2)-concrete_data$ccs)^2
dataModel2.ssr <- sum(fitted(dataModel2) -mean(concrete_data$ccs))^2
dataModel2.sst = dataModel2.sse + dataModel2.ssr
Rsquared_sm=1-(dataModel2.sse/dataModel2.sst)
cat(Rsquared_sm)
```

```
## 0.5000744
```

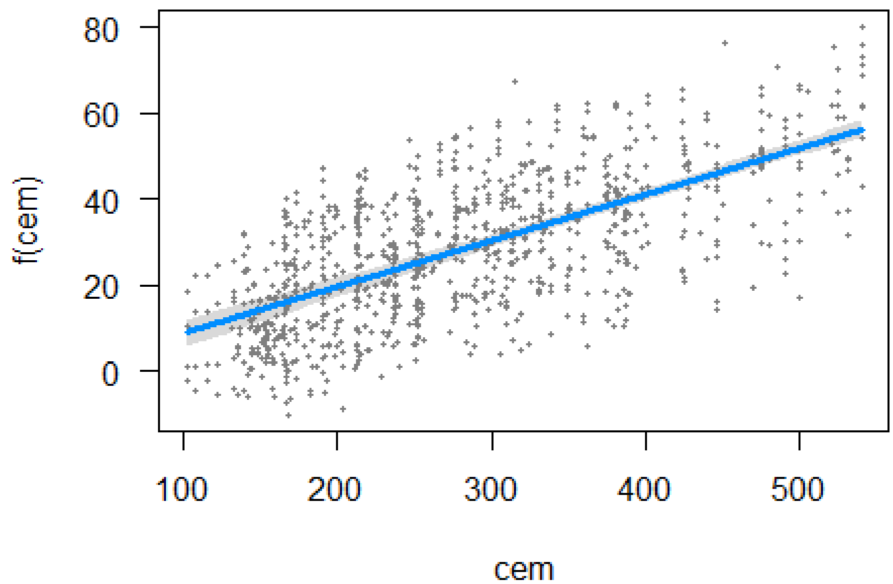
```r
anova(dataModel1, dataModel2, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: ccs ~ cem + bfs + fa + water + sp + cagg
## Model 2: ccs ~ s(cem) + s(bfs) + s(fa) + s(water) + s(sp) + s(cagg)
##   Resid. Df Resid. Dev     Df Deviance  Pr(>Chi)
## 1   1023.00     158334
## 2    996.43     131019 26.574    27315 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

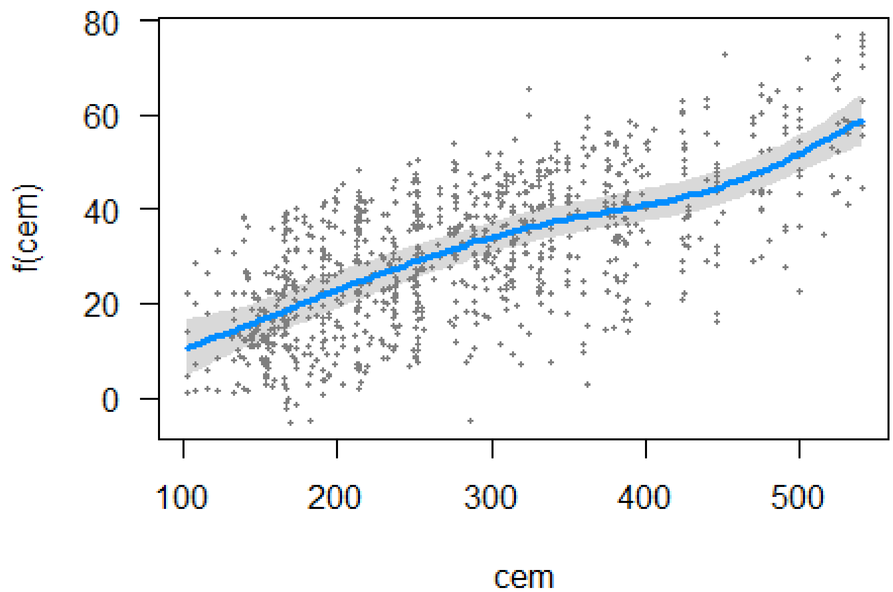
```r
cat("Although we couldn't have known as much already, new statistical
evidence suggests that adding in the variables' nonlinear correlations
enhances the model.")
```

```
## Although we couldn't have known as much already, new statistical evidence
suggests that adding in the variables' nonlinear correlations enhances the
model.
```

```r
visreg(dataModel1,'cem')
```



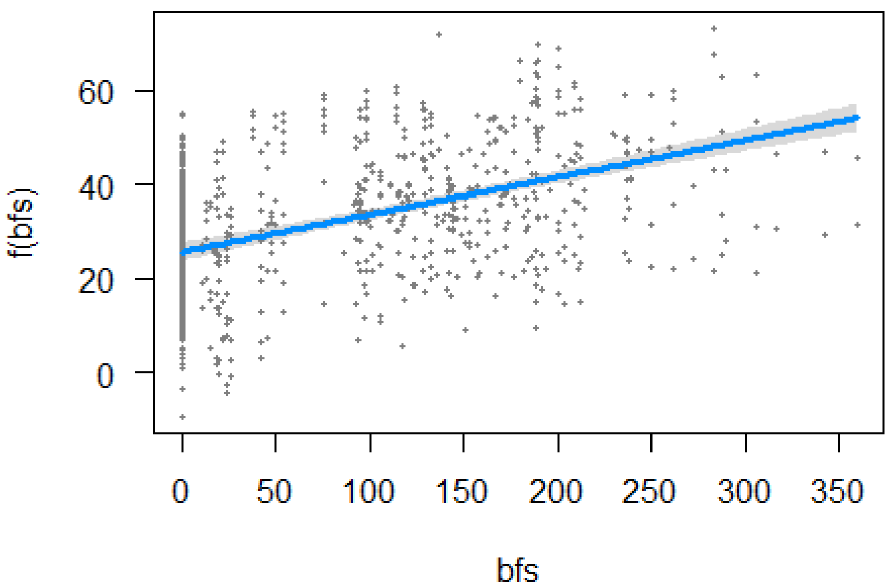

```
visreg(dataModel2,'cem')
```



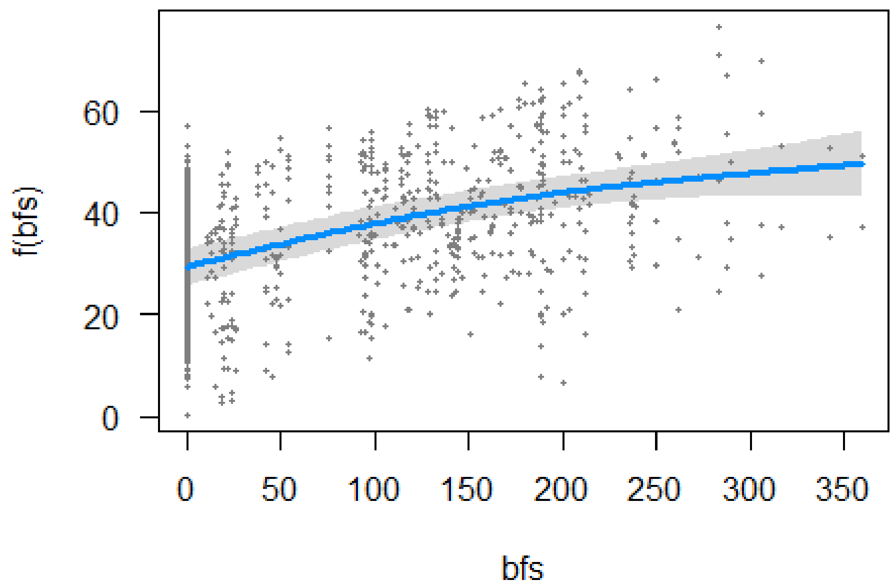
```
cat("The end result, with all other model variables maintained constant,
    is a plot showing how the expected value of the CCS changes as a function
of x (CEM).
    It contains the following information: (1) the expected value (blue
line),
    (2) a confidence interval for the expected value (gray band), and
    (3) partial residuals (dark gray dots).")
```

```
## The end result, with all other model variables maintained constant,
##     is a plot showing how the expected value of the CCS changes as a
function of x (CEM).
##     It contains the following information: (1) the expected value (blue
line),
##     (2) a confidence interval for the expected value (gray band), and
##     (3) partial residuals (dark gray dots).
```

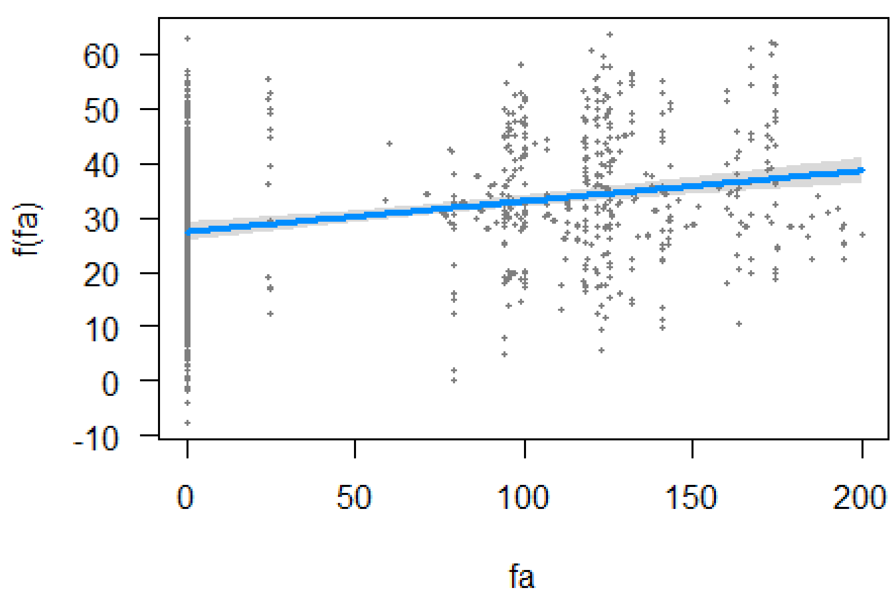
```
# Visualizing the feature with the function of their feature
visreg(dataModel1,'bfs')
```



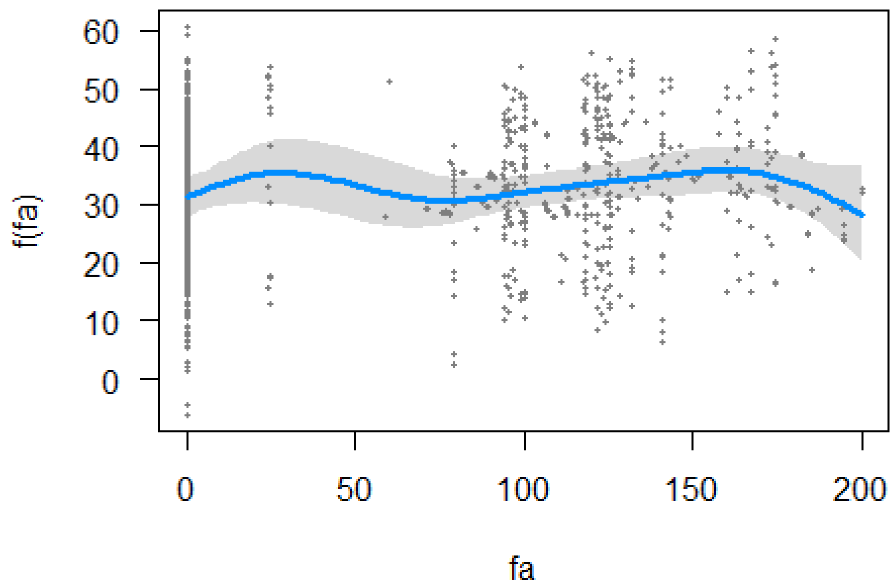

```
visreg(dataModel2,'bfs')
```



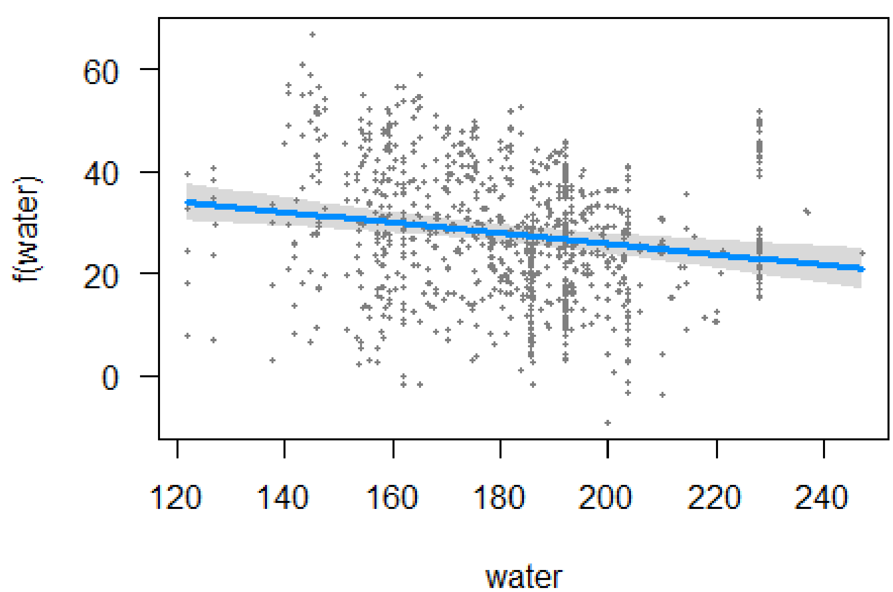

```
visreg(dataModel1,'fa')
```



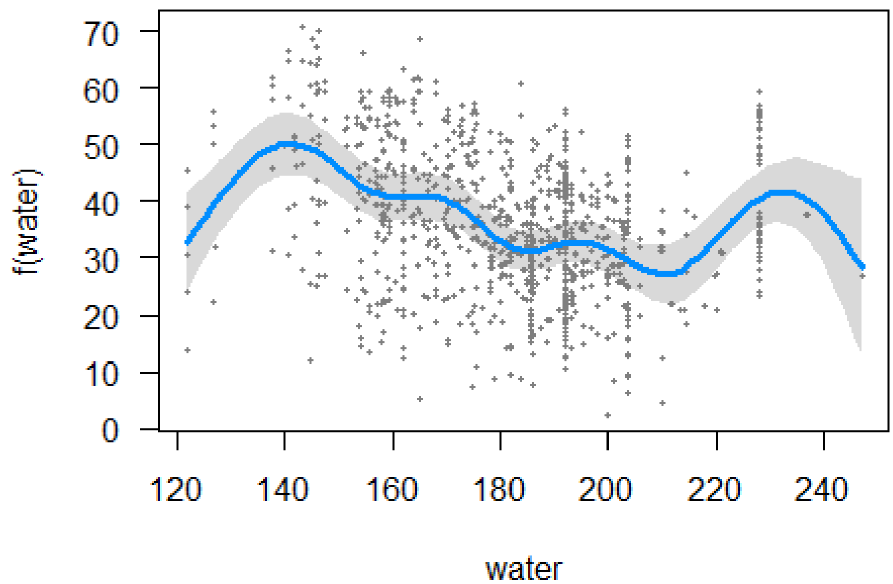

```
visreg(dataModel2,'fa')
```



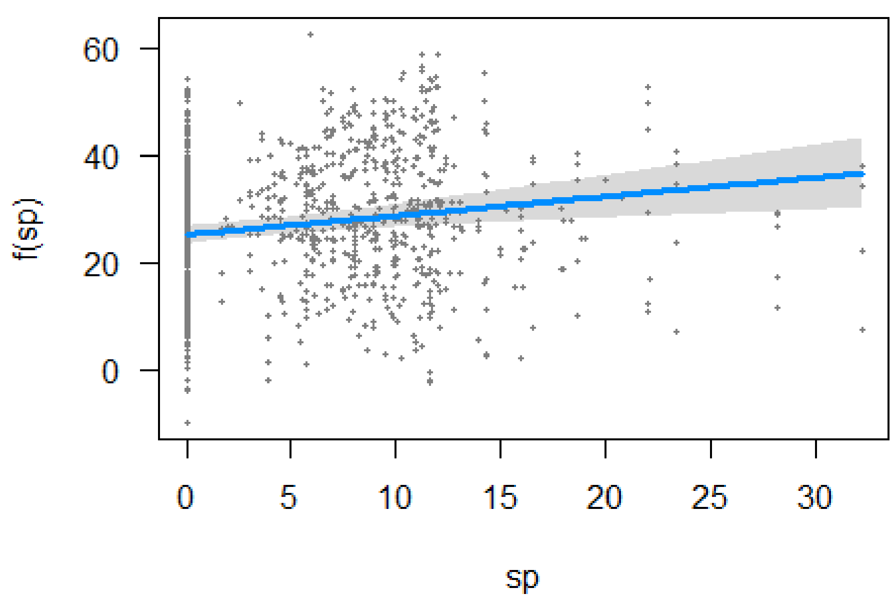

```
visreg(dataModel1,'water')
```



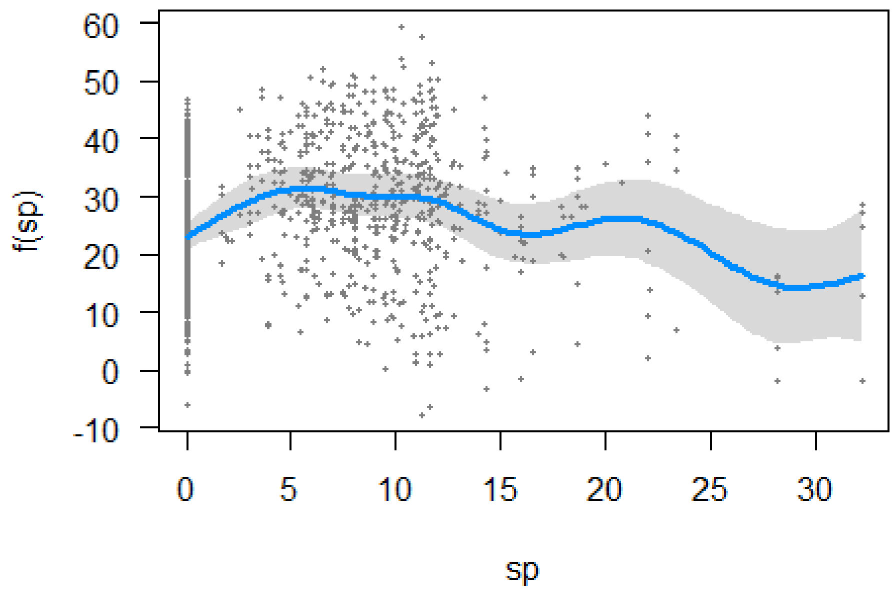

```
visreg(dataModel2,'water')
```



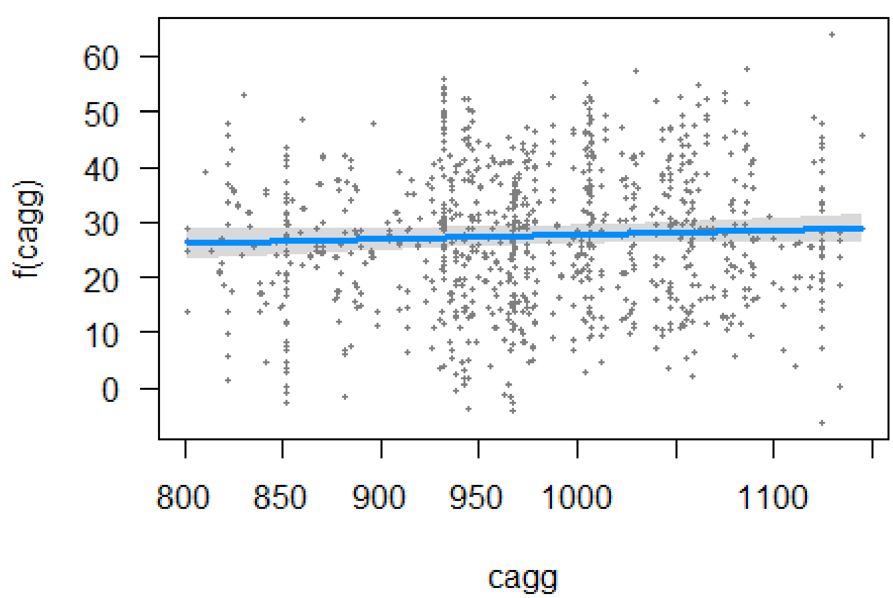

```
visreg(dataModel1,'sp')
```



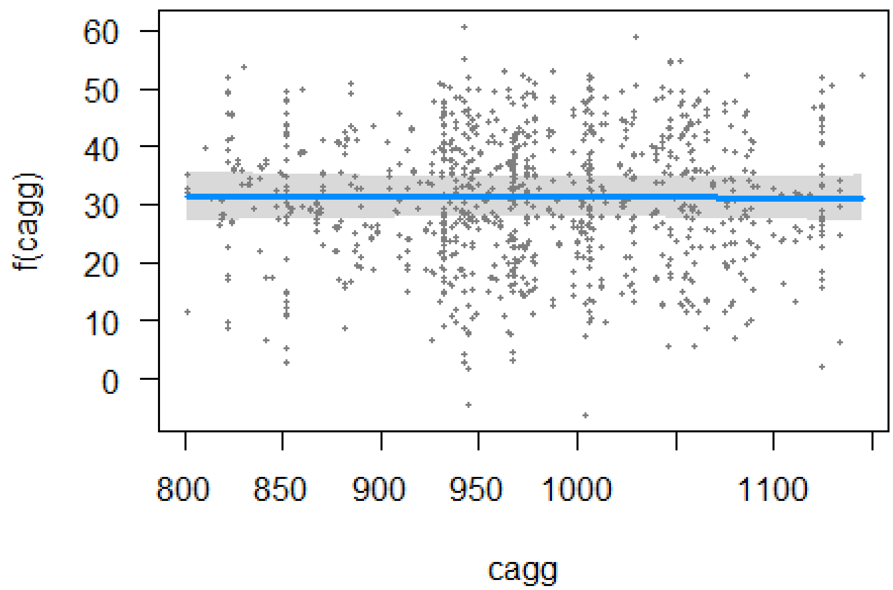

```
visreg(dataModel2,'sp')
```




```
visreg(dataModel1,'cagg')
```




```
visreg(dataModel2,'cagg')
```




```
cat("We can see from the CEM graph that the confidence interval has a higher value after adding the smoothing function than the model had without it. Using the smoothing function improves the confidence interval.")

## We can see from the CEM graph that the confidence interval has a higher value after adding the smoothing function than the model had without it. Using the smoothing function improves the confidence interval.
```