Portfolio 6

Rachel Wood

2023-03-07

For this portfolio we will use data from the `gss` package, modelling `ret` as a function of `dur`, `gly` and `bmi`.

```
library(gss)
data("wesdr")
head(wesdr)
```

```
##    dur  gly  bmi ret
## 1 10.3 13.7 23.8   0
## 2  9.9 13.5 23.5   0
## 3 15.6 13.8 24.8   0
## 4 26.0 13.0 21.6   1
## 5 13.8 11.1 24.6   1
## 6 31.1 11.3 24.6   1
```

```
dim(wesdr)
```

```
## [1] 669   4
```

We first split the data into training and testing sets:

```
library(dplyr)
train <- sample(1:nrow(wesdr), 500)

data_train <- wesdr[train,] %>%
  as_tibble()
data_test <- wesdr[-train,] %>%
  as_tibble()
```

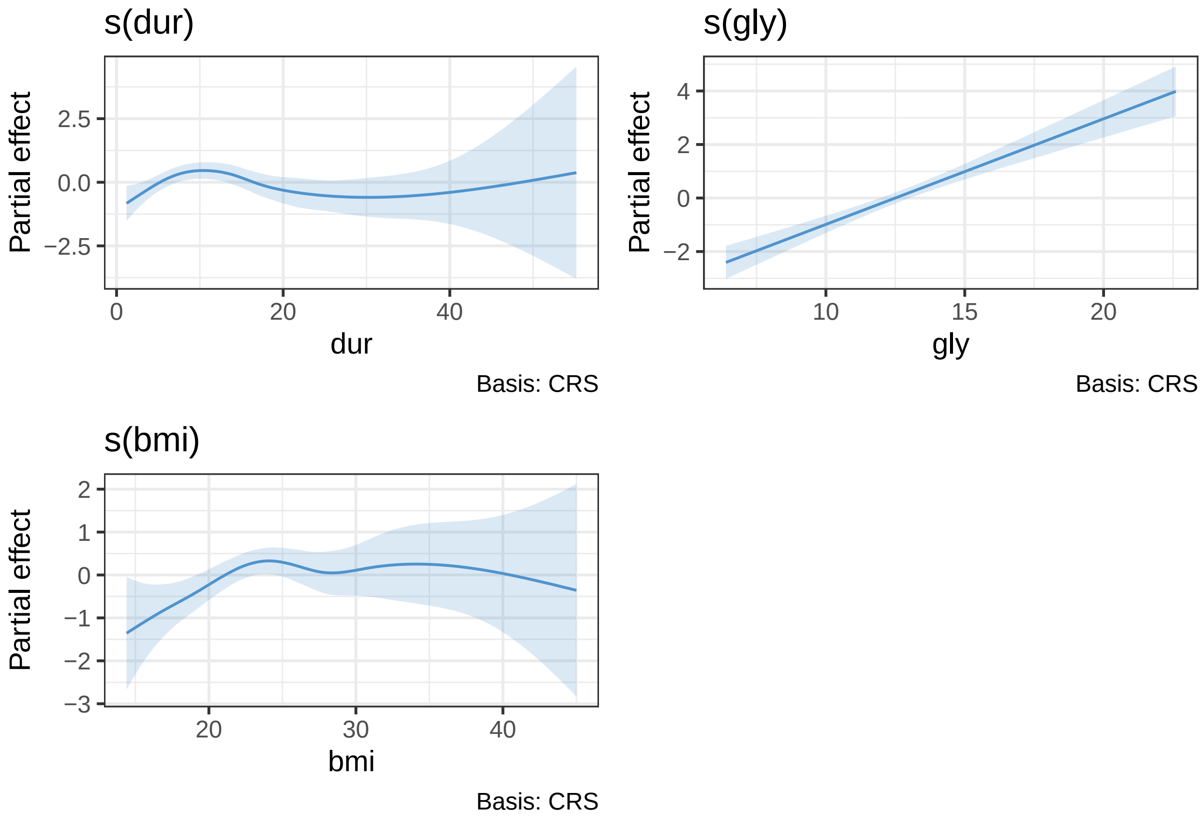
We now fit a GAM model with the `gam()` function, which uses generalised cross-validation to fit our model parameters:

```
library(mgcv)
fit <- gam(ret ~ s(dur, bs = "cr") + s(gly, bs = "cr") + s(bmi, bs = "cr"), data = data_train, family =
```

We now want to visualise the estimated functions to see if they are in fact non-linear:

```
library(gratia)

draw(fit, smooth_col = "steelblue3", ci_col = "steelblue3", rug  = FALSE)
```



We can see that the `gly` variable appears to be linear, but `dur` and `bmi` do not seem to be. From this we can see a GAM would be appropriate. We can confirm this by comparing the out-of-sample error of this model to one which assumes all functions are linear:

```
gam_error <- sum((predict(fit, data_test, type = "response") - data_test$ret)^2)

glm_fit <- glm(ret ~ dur + gly + bmi, data = data_train, family = "binomial")
glm_error <- sum((predict(glm_fit, data_test, type = "response") - data_test$ret)^2)

print(paste("GAM out-of-sample error:", gam_error))
```

```
## [1] "GAM out-of-sample error: 33.8635927690903"
```

```
print(paste("GLM out-of-sample error:", glm_error))
```

```
## [1] "GLM out-of-sample error: 34.6795619994107"
```

From this we see the GAM has slightly superior predictive power, although this is marginal.