Statistical Methods Assessed Homework

Kieran Morris

In this assessed homework we use the `bone_mineral_density` data which can be found in the `spnbmd.csv` file. We have 1003 obervations on 5 variables, the first variable `idnum` refers to the individual the data was taken from, the `ethnic` and `sex` variables are catagorical, and `age` and `spnbmd` variables are continuous. The `spnbmd` variable is a measurement of relative change in spinal bone on four visits. We import the dataset below and call it `bmd`.

```
# Load the data
bmd <- read.csv("spnbmd.csv")

# Display the first few rows of the data
head(bmd,n= 6)
```

```
##   idnum ethnic  age sex spnbmd
## 1     1  White 11.2 mal  0.719
## 2     1  White 12.2 mal  0.732
## 3     1  White 13.2 mal  0.776
## 4     1  White 14.3 mal  0.781
## 5     2  White 12.7 mal  0.620
## 6     2  White 13.8 mal  0.627
```

Looks pretty normal to me, lets first skim the data to get a sense of overall structure, we will use the package `skimr` to do this.

```
library(skimr)

# Define a skim function that does not include histograms,for knit
skim_no_hist <- skim_with(numeric = sfl(hist = NULL),
                          character = sfl(hist = NULL))

# Use the new skimming function on your data
skim_data <- skim_no_hist(bmd)

# Print the skimmed data
print(skim_data)
```

```
## -- Data Summary ------------------------
##                            Values
## Name                       bmd
## Number of rows             1003
## Number of columns          5
## _______________________
## Column type frequency:
##   character                2
```

```
##   numeric                  3
## ________________________
## Group variables            None
##
## -- Variable type: character ---------------------------------------------------
##   skim_variable n_missing complete_rate min max empty n_unique whitespace
## 1 ethnic                0             1   5   8     0        4          0
## 2 sex                   0             1   3   3     0        2          0
##
## -- Variable type: numeric -----------------------------------------------------
##   skim_variable n_missing complete_rate    mean      sd    p0    p25     p50
## 1 idnum                 0             1 190.    121.     1     84.5   181
## 2 age                   0             1  16.3     4.35  8.8   12.8    15.7
## 3 spnbmd                0             1   0.948   0.184 0.536  0.800   0.965
##      p75   p100
## 1 288.   429
## 2  19.4   26.2
## 3   1.07   1.44
```

I actually really hate how `skimr` outputs, but at least we have confirmed our observations, notice that `ethnic` has 4 unique values, and `sex` has 2 unique values. Below we have a function which randomises the data and chooses a training and test set.

```r
# Set the seed
set.seed(123)

# Create a function to split the data into a training and test set
split_data <- function(data, prop = 0.8) {
  # Randomly order the rows of the data
  data <- data[sample(nrow(data)), ]

  # Calculate the number of rows in the training set
  n_train <- round(nrow(data) * prop)

  # Split the data into a training and test set
  train <- data[1:n_train, ]
  test <- data[(n_train + 1):nrow(data), ]

  # Return the training and test set
  list(train = train, test = test)
}

# Use the function to split the data
bmd_split <- split_data(bmd)
train <- bmd_split$train
test <- bmd_split$test
```

Question 1: Simple Linear Regression Model

We will split our (training) data into male and female, and then fit a simple linear regression model on the variables `age` and `spnbmd` for each group. In particular, we assume

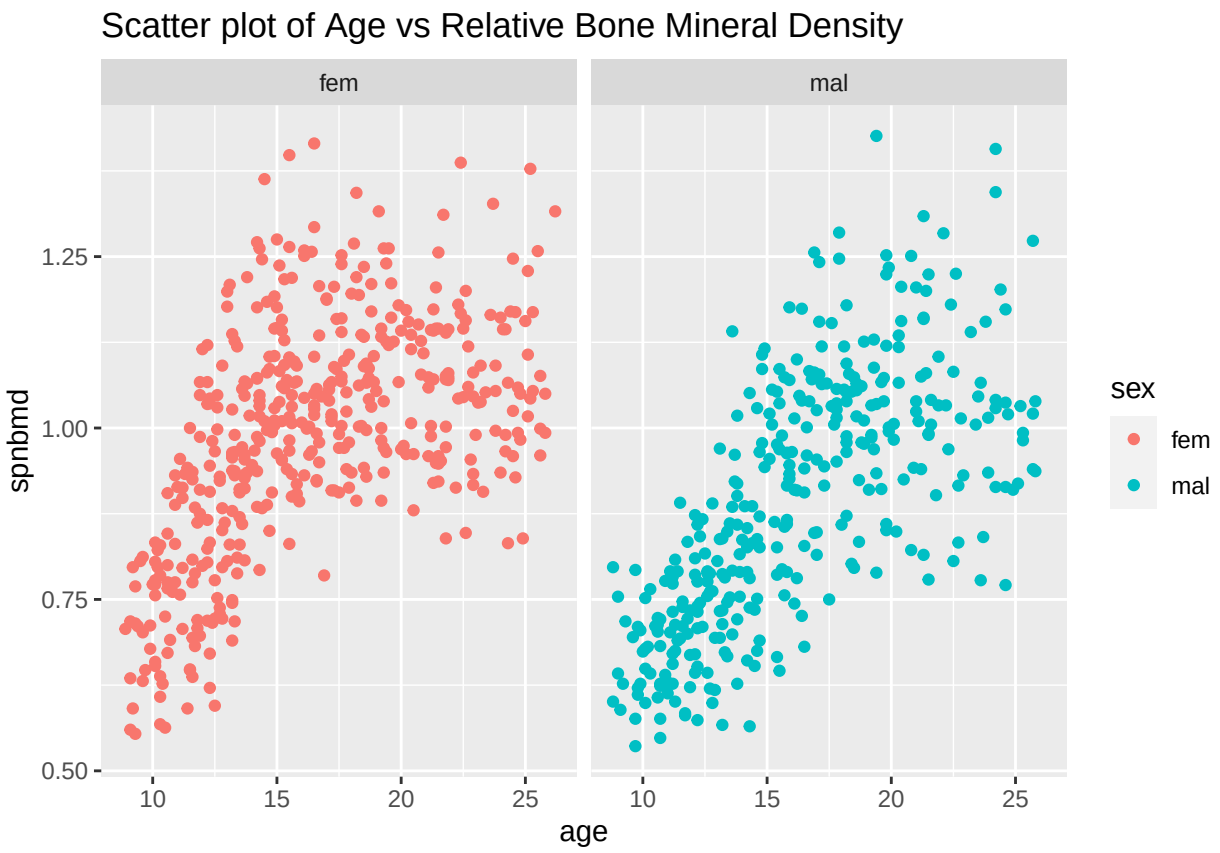
$$\textbf{spnbmd} = \beta_0 + \beta_1 \textbf{age} + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$. Before sticking to a regression model we will plot the data to see if there is any clear relation.

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Create a scatter plot of age and spnbmd
ageVbm <- ggplot(bmd_split$train, aes(x = age, y = spnbmd, color = sex)) +
  geom_point() +
  facet_wrap(~sex) +
  labs(title = "Scatter plot of Age vs Relative Bone Mineral Density")

ageVbm
```



We clearly have correlation between `age` and `spnbmd` for both male and female, however the variance increases as `age` increases. We continue anyway with a linear regression model and plot the estimated regression lines.

```
# Split the train data into male and female

bmd_train_male <- filter(train, train$sex == "mal")
bmd_train_female <- filter(train, train$sex == "fem")

#Fit a regression model on each group

reg_male <- lm(spnbmd ~ age, data = bmd_train_male)
reg_female <- lm(spnbmd ~ age, data = bmd_train_female)

reg_male
```

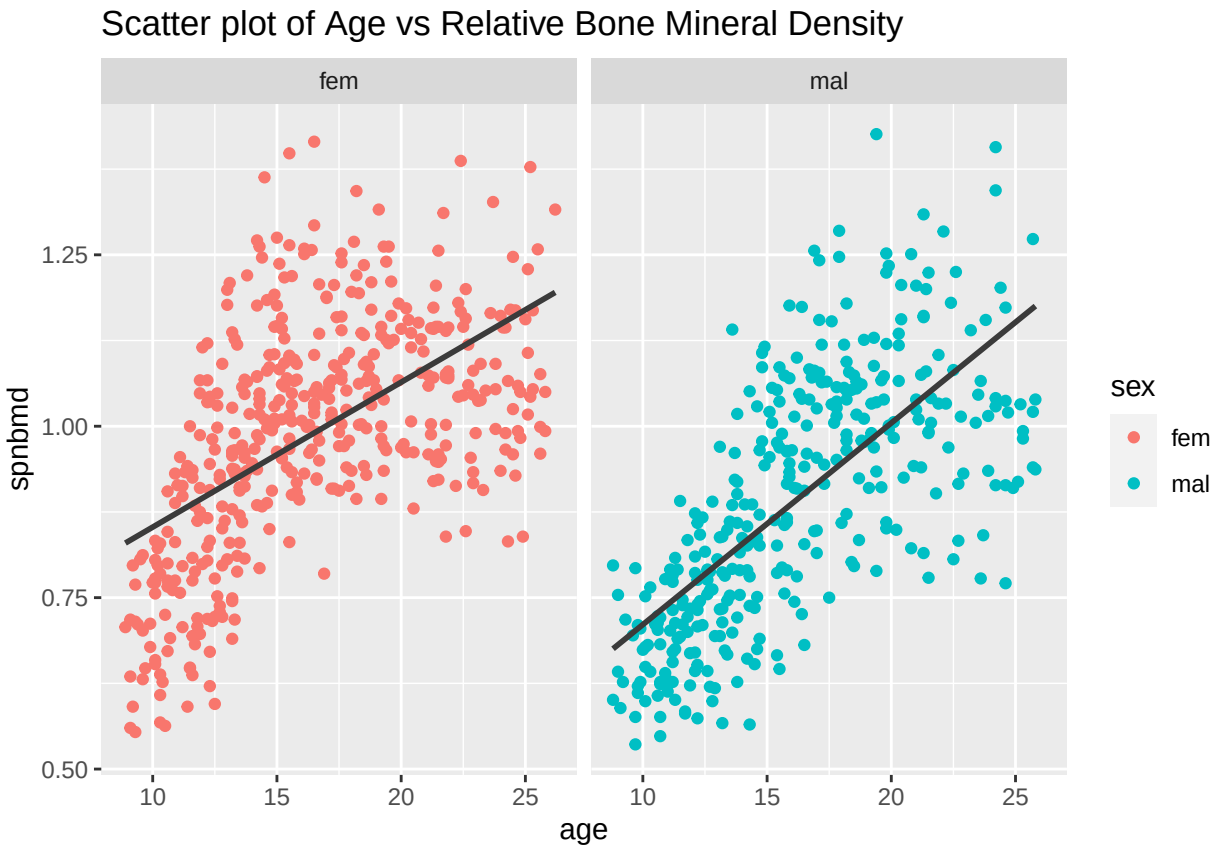
```
##
## Call:
## lm(formula = spnbmd ~ age, data = bmd_train_male)
##
## Coefficients:
## (Intercept)          age
##      0.4167       0.0294
```

```
reg_female
```

```
##
## Call:
## lm(formula = spnbmd ~ age, data = bmd_train_female)
##
## Coefficients:
## (Intercept)          age
##     0.64157      0.02113
```

```
# Plot the estimated regression lines
ageVbm +
  geom_smooth(data = bmd_split$train, method = "lm", se = FALSE,color = "#3b3b3b")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



These regression lines are not a great fit for the data, to verify this we will calculate the mean square error.

```
# Create a function to calculate the mean square error
mse <- function(model, data) {
  # Calculate the predicted values
  pred <- predict(model, newdata = data)

  # Calculate the residuals
  res <- data$spnbmd - pred

  # Calculate the mean square error
  mean(res^2)
}
```

```
#split the test data into male and female
bmd_test_male <- filter(test, sex == "mal")
bmd_test_female <- filter(test, sex == "fem")

mse_male <- mse(reg_male,bmd_test_male)
mse_female <- mse(reg_female,bmd_test_female)

weight_male <- nrow(bmd_test_male) / nrow(test)
weight_female <- nrow(bmd_test_female) / nrow(test)

# Compute the overall test error
mse_all <- weight_male * mse_male + weight_female * mse_female
```

```r
print(paste("MSE for male test set:",mse_male))
```

```
## [1] "MSE for male test set: 0.0224358359310131"
```

```r
print(paste("MSE for female test set:",mse_female))
```

```
## [1] "MSE for female test set: 0.0215337594249837"
```

```r
print(paste("MSE for entire test set:",mse_all))
```

```
## [1] "MSE for entire test set: 0.0219915295922225"
```

In comparison to the range of our data this is not particularly good, although we saw that from the plots.

Question 2: Polynomial Regression Model

This time we attempt to fit a polynomial regression model to the data, so in particular we assume

$$\textbf{spnbmd} = \beta_0 + \beta_1 \textbf{age} + ... + \beta_l \textbf{age}^l + \epsilon$$

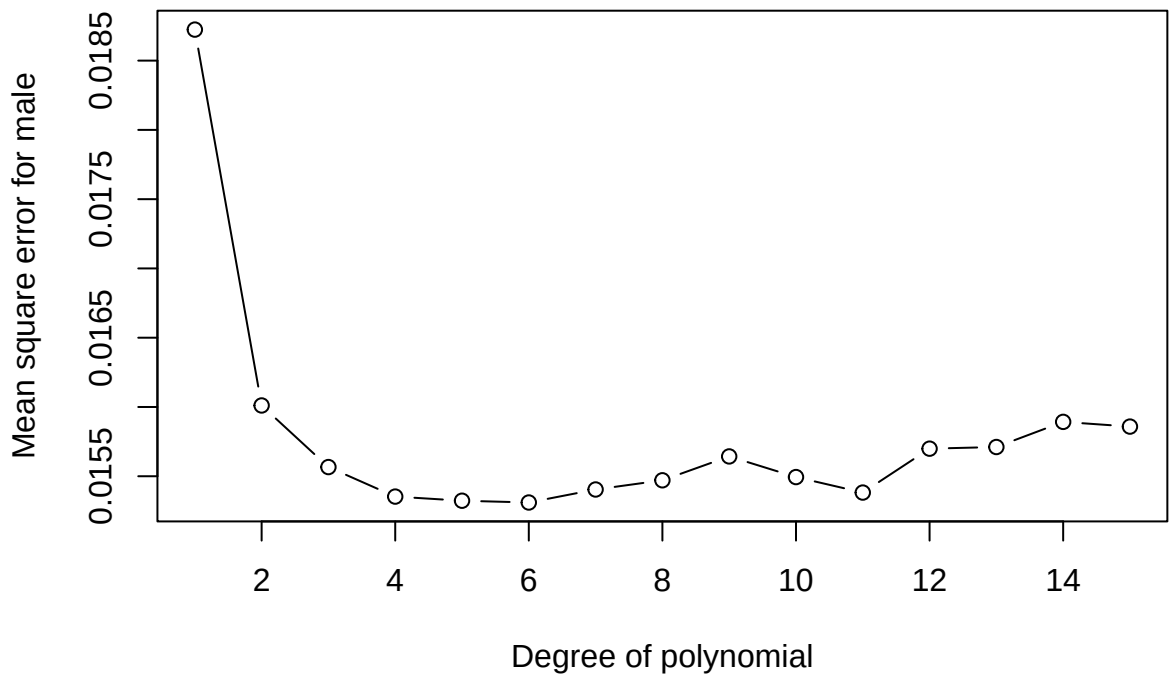
for $\epsilon \sim N(0, \sigma^2)$. As we have a hyperparameter l, we will perform cross-validation to find the best l. Below we create a rudimentary cross validation function.

```r
set.seed(123)
# Create a function to perform k-fold cross-validation
k_fold_cv <- function(data, k, l) {
  # Randomly order the rows of the data
  data <- data[sample(nrow(data)), ]

  # Create a vector to store the mean square errors
  mse <- rep(0, k)

  # Calculate the number of rows in each fold
  n <- nrow(data)
  n_per_fold <- n %/% k

  # Perform k-fold cross-validation
  for (i in 1:k) {
    # Create the training and test set
    test_indices <- ((i - 1) * n_per_fold + 1):(i * n_per_fold)
    test <- data[test_indices, ]
    train <- data[-test_indices, ]

    # Fit the polynomial regression model
    reg <- lm(spnbmd ~ poly(age, l), data = train)

    # Calculate the mean square error
    mse[i] <- mse(reg, test)
```

```
  }

  # Return the mean square errors
  mse
}
```

Now we apply our function to `bmd_male`:

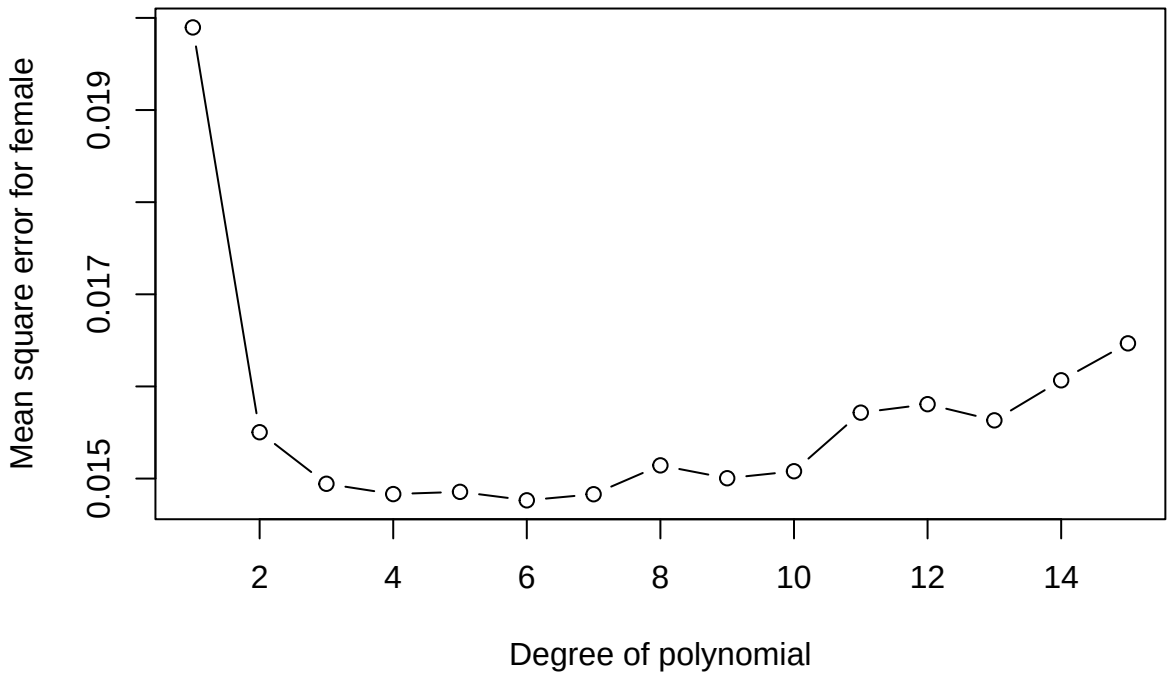
```
bmd_male <- filter(bmd,bmd$sex == "mal")
bmd_female <- filter(bmd,bmd$sex == "fem")

k <- 10
l <- 1:15
# Apply the k-fold cross-validation function to each degree in l
mse_values <- sapply(l, function(degree) k_fold_cv(bmd_male, k, degree))


#plot the mean square errors
plot(l, colMeans(mse_values), type = "b", xlab = "Degree of polynomial", ylab = "Mean square error for m
```



and `bmd_female`:

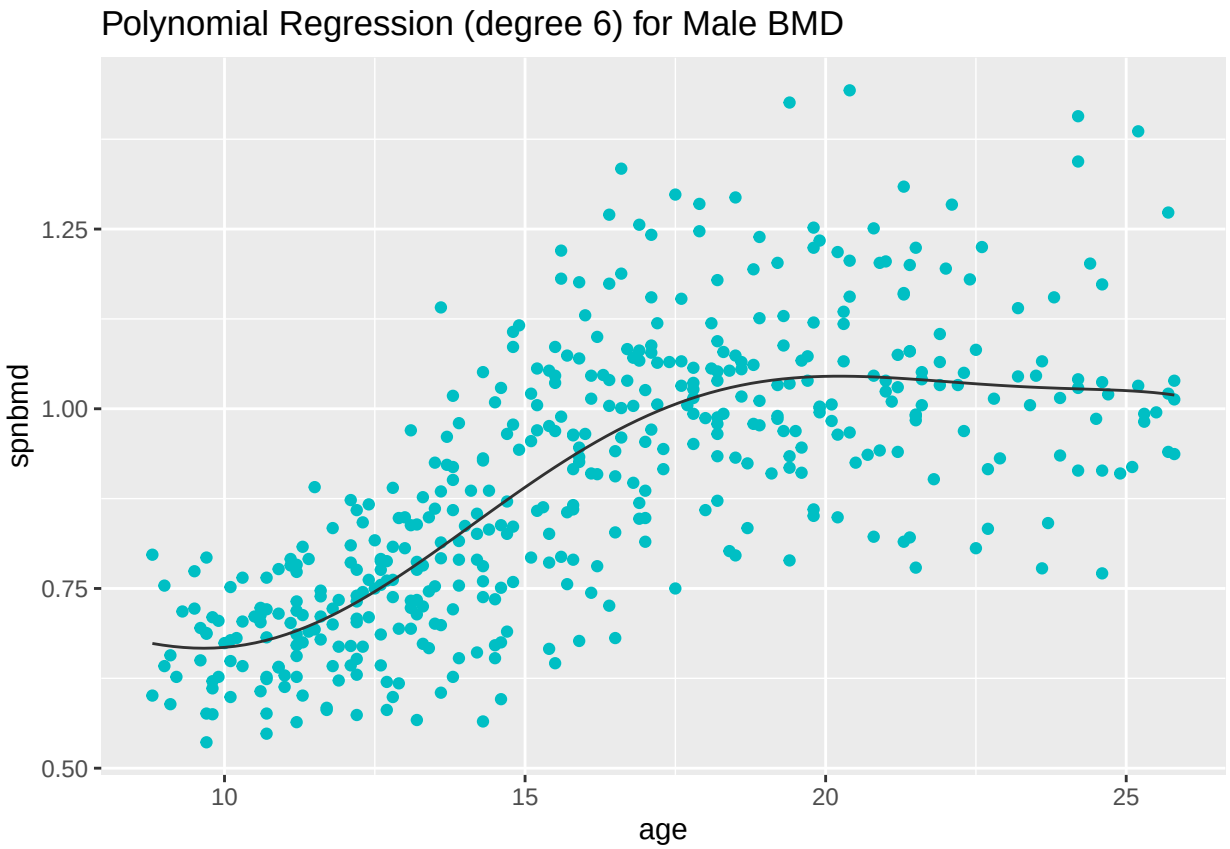
```
set.seed(123)
# Apply the k-fold cross-validation function to each degree in l
mse_values <- sapply(l, function(degree) k_fold_cv(bmd_female, k, degree))
```

```r
#plot the mean square errors
plot(l, colMeans(mse_values), type = "b", xlab = "Degree of polynomial", ylab = "Mean square error for f
```

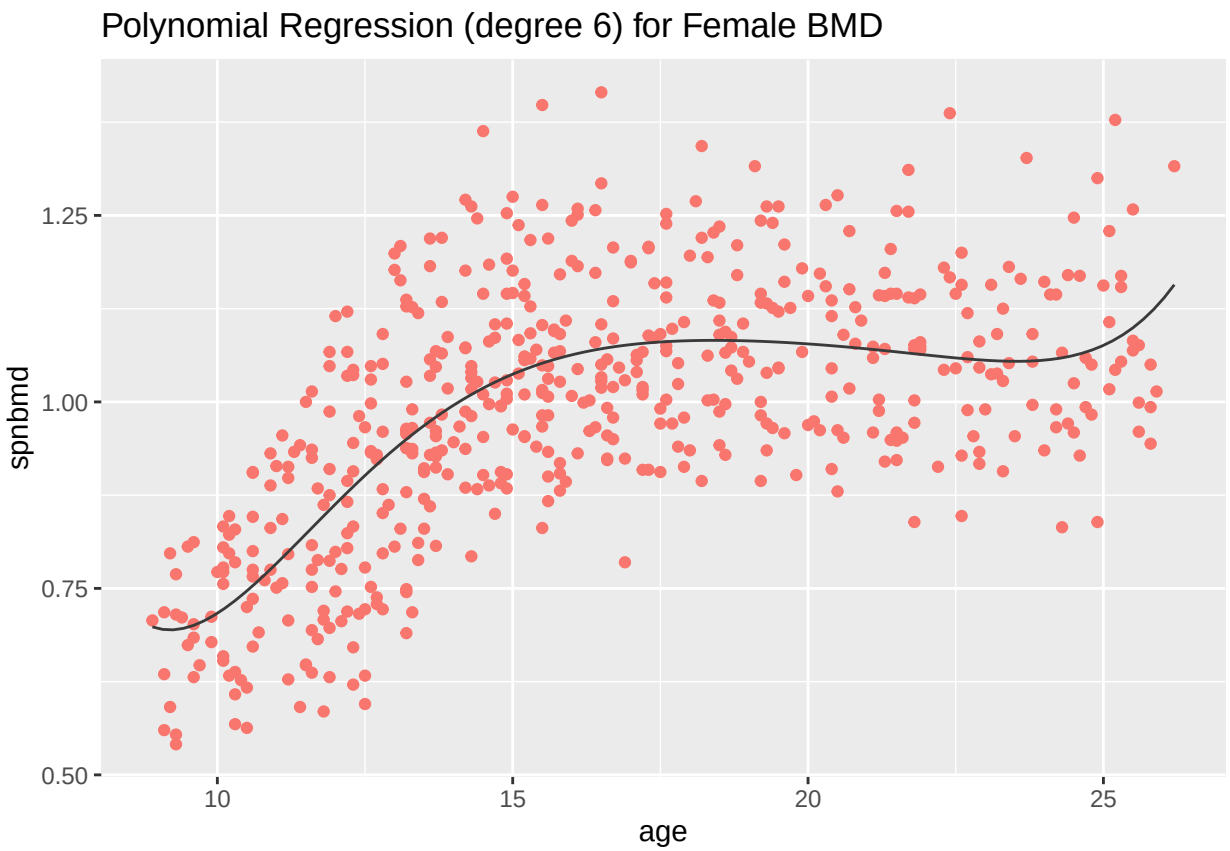


We can see that the mean square error is minimised at $l = 6$ for both. So we will proceed with $l = 6$ for our model. Below we plot the degree 6 polynomial regression against our datasets like before.

```r
# Load the ggplot2 package
library(ggplot2)

# Fit the polynomial regression model for bmd_male
model_male <- lm(spnbmd ~ poly(age, 6, raw = TRUE), data = bmd_train_male)

# Create a new data frame for predictions
newdata_male <- data.frame(age = seq(min(bmd_male$age), max(bmd_male$age), length.out = 100))

# Add the predictions to the new data frame
newdata_male$spnbmd <- predict(model_male, newdata = newdata_male)

# Plot the data and the fitted model
ggplot(bmd_male, aes(x = age, y = spnbmd)) +
  geom_point(color = "#00BFC4") +
  geom_line(data = newdata_male, aes(x = age, y = spnbmd), color = "#313131") +
  ggtitle("Polynomial Regression (degree 6) for Male BMD")
```




```r
# Repeat the process for bmd_female
model_female <- lm(spnbmd ~ poly(age, 6, raw = TRUE), data = bmd_train_female)
newdata_female <- data.frame(age = seq(min(bmd_female$age), max(bmd_female$age), length.out = 100))
newdata_female$spnbmd <- predict(model_female, newdata = newdata_female)

ggplot(bmd_female, aes(x = age, y = spnbmd)) +
  geom_point(color = "#F8766D") +
  geom_line(data = newdata_female, aes(x = age, y = spnbmd), color = "#3a3a3a") +
  ggtitle("Polynomial Regression (degree 6) for Female BMD")
```



This model is certainly superior to the original linear regression model, we will now calculate the mean square error for the test set.

```
# Calculate the mean square error for the test set

mse_male <- mse(model_male,bmd_test_male)
mse_female <- mse(model_female,bmd_test_female)

# Compute the overall test error
mse_all <- weight_male * mse_male + weight_female * mse_female


print(paste("MSE for male test set:",mse_male))
```

```
## [1] "MSE for male test set: 0.0177362190187037"
```

```
print(paste("MSE for female test set:",mse_female))
```

```
## [1] "MSE for female test set: 0.0150015048015356"
```

```
print(paste("MSE for entire test set:",mse_all))
```

```
## [1] "MSE for entire test set: 0.0163892702251731"
```

We have about half as much error, which is great news all round!

Question 3: One-Dimensional Smoothing Methods

Next we use smoothing to fit a model to the data. We will use cubic splines and specifically want to use the following model:

$$\textbf{spnbmd} = a_m + f_m(\textbf{age}) + \epsilon$$

and

$$\textbf{spnbmd} = a_f + f_f(\textbf{age}) + \epsilon$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and we have different parameters for male and female. We will use a GAM model, which automatically performs generalised cross validation on the damping parameter, which saves us a bit of time! When we perform GAM, we search over functions such that $\sum_{i=1}^{n} f(_{ij}^{0}) = 0$ to gaurantee a unqiue solution. We will use the `mgcv` package to fit the model and specify `bs="cr"` to use natural cubic splines.

```
# Load the mgcv package
library(mgcv)
```

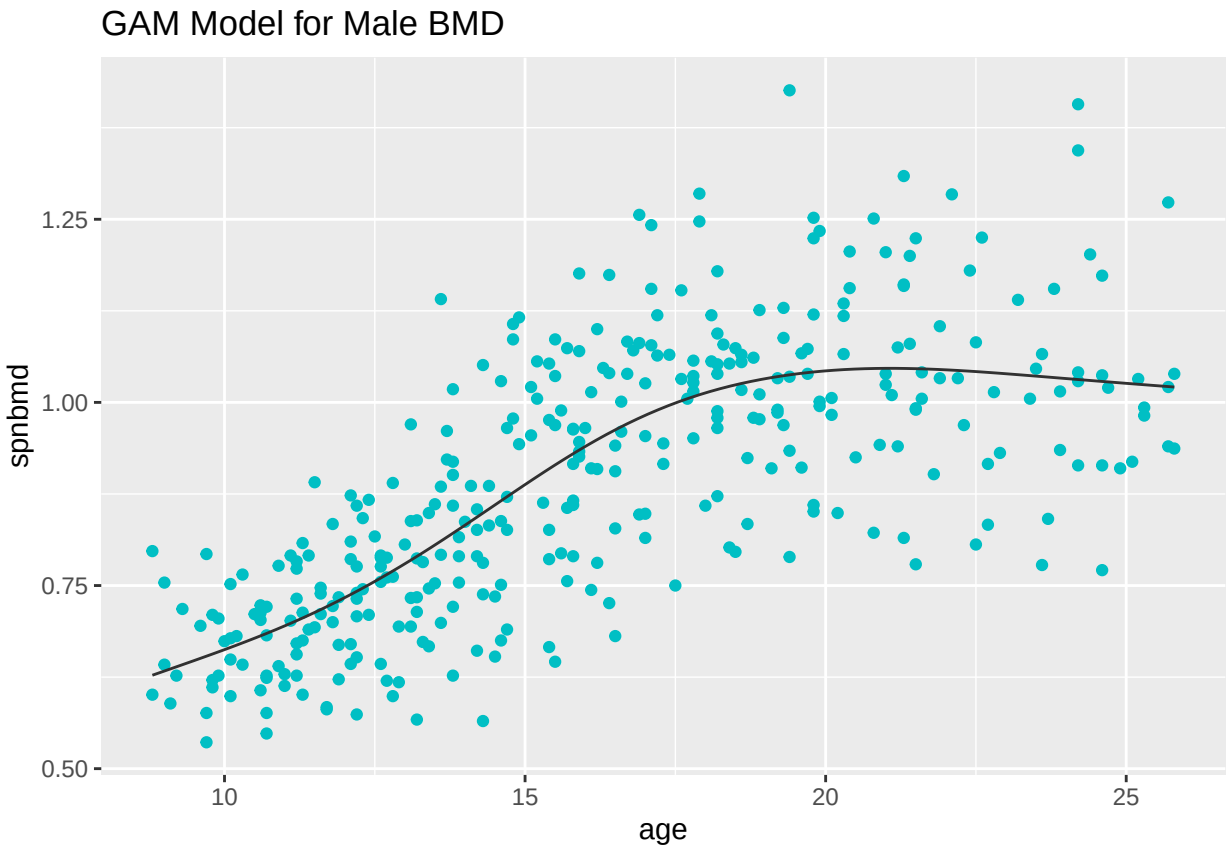
```
## Loading required package: nlme
```

```
##
## Attaching package: 'nlme'
```

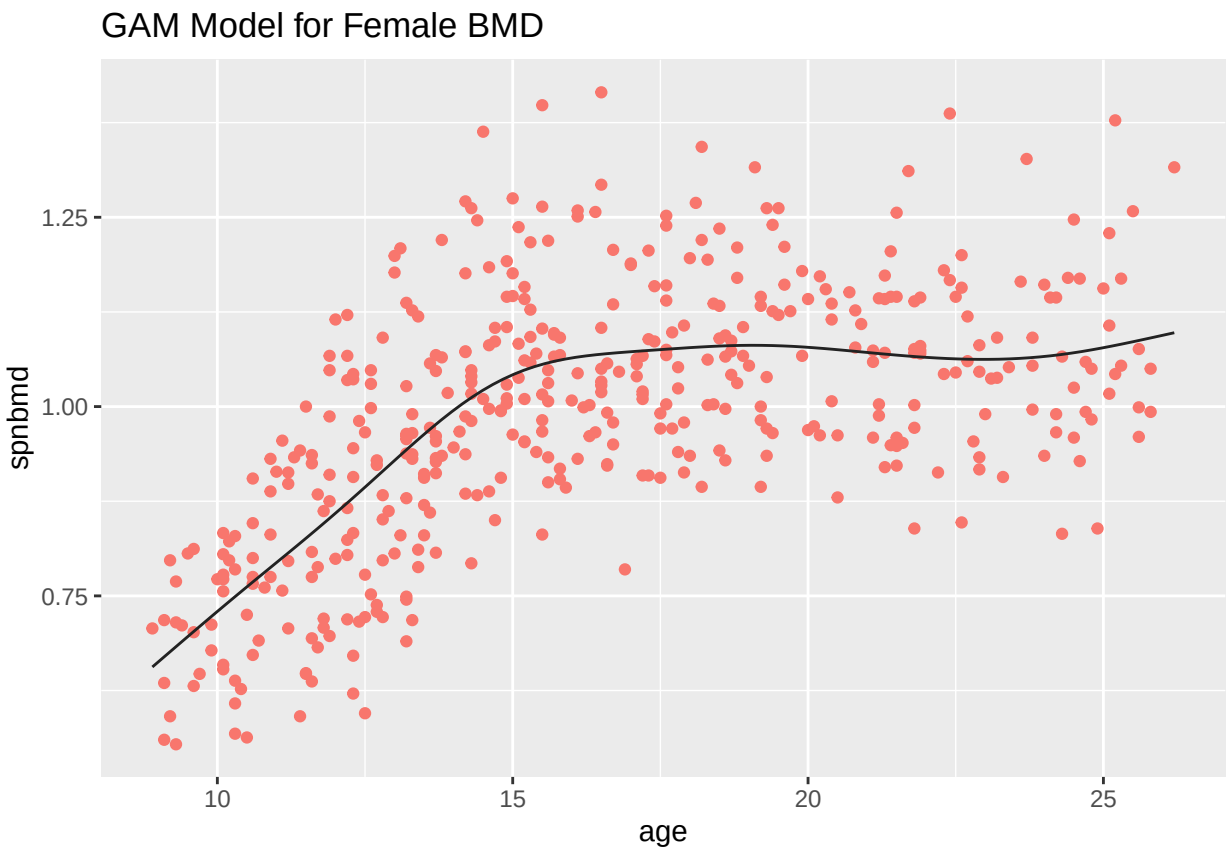
```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
## This is mgcv 1.9-1. For overview type 'help("mgcv-package")'.
```

```
# Fit the GAM
gam_model_male <- gam(spnbmd ~ s(age), ,bs = "cr",data = bmd_train_male)
gam_model_female <- gam(spnbmd ~ s(age), ,bs = "cr",data= bmd_train_female)
#Plot the model in ggplot

# Create a new data frame for predictions
newdata_male <- data.frame(age = seq(min(bmd_train_male$age), max(bmd_train_male$age), length.out = 100)
newdata_female <- data.frame(age = seq(min(bmd_train_female$age), max(bmd_train_female$age), length.out

# Generate predictions
Ydata_Male <- predict(gam_model_male, newdata = newdata_male)
Ydata_Female <- predict(gam_model_female, newdata = newdata_female)

# Add the predictions to the new data frames
newdata_male$spnbmd <- Ydata_Male
newdata_female$spnbmd <- Ydata_Female
# Plot the data and the fitted model for bmd_train_male
ggplot(bmd_train_male, aes(x = age, y = spnbmd)) +
  geom_point(color = "#00BFC4") +
  geom_line(data = newdata_male, aes(x = age, y = spnbmd), color = "#313131") +
  ggtitle("GAM Model for Male BMD")
```




```
# Plot the data and the fitted model for bmd_train_female
ggplot(bmd_train_female, aes(x = age, y = spnbmd)) +
  geom_point(color = "#F8766D") +
  geom_line(data = newdata_female, aes(x = age, y = spnbmd), color = "#222222") +
  ggtitle("GAM Model for Female BMD")
```



This model is very similar to the polynomial regression, lets find the mean squared error with the test set to see if it is any better.

```
mse_male <- mse(gam_model_male,bmd_test_male)
mse_female <- mse(gam_model_female,bmd_test_female)

# Compute the overall test error
mse_all <- weight_male * mse_male + weight_female * mse_female


print(paste("MSE for male test set:",mse_male))
```

```
## [1] "MSE for male test set: 0.0181317754660071"
```

```
print(paste("MSE for female test set:",mse_female))
```

```
## [1] "MSE for female test set: 0.0149056477902181"
```

```
print(paste("MSE for entire test set:",mse_all))
```

```
## [1] "MSE for entire test set: 0.0165427872072077826"
```

As expected these mean square errors are almost indistinguishable from the polynomial regression model, which is not surprising given their plots.

Question 4: Two-Dimensional Smoothing Methods

Fortunately `GAM` models are also very well applied to multi-dimensional data, where we consider the output as a linear combination of smooth functions of the input variables. We will use this to combine our two datas into one, which depends on the parameter `sex`. Our new model is:

$$\mathbf{spnbmd} = a + b\mathbf{sex} + f(\mathbf{age}) + \epsilon$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. One might ask whether this model is suitable, since `sex` is catagorical it could feel strange to add it to a continuous GAM model, however since we do have minor differences between `mal` and `fem` it may be a useful indicator. Chances are that the `b` value will be very small anyway. We will use the `gam` function again, but this time we with another argument. We will have to convert `sex` into a numeric vector first.

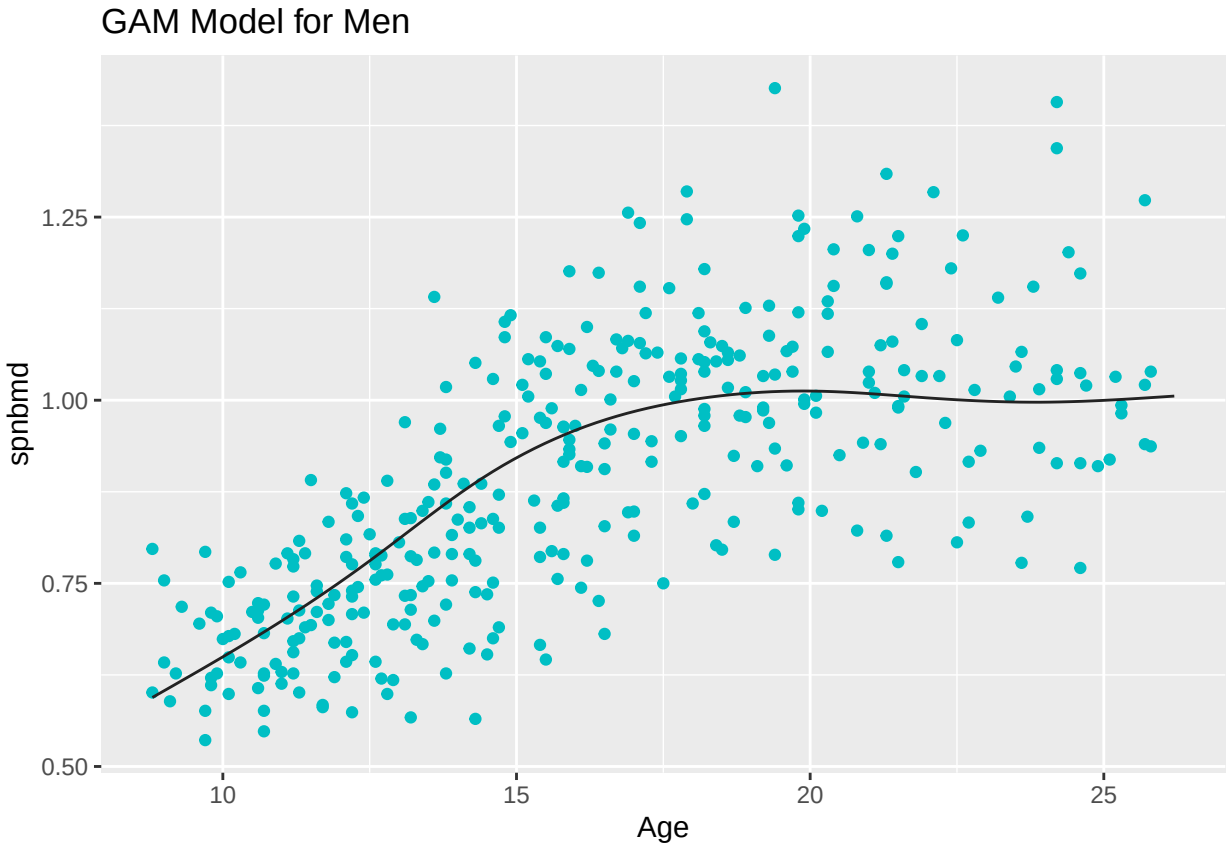
```
train$sex <- recode(train$sex, "mal" = 0, "fem" = 1)
# Fit the GAM
gam_model <- gam(spnbmd ~ s(age, bs = "cr") + sex, data = train)
summary(gam_model)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## spnbmd ~ s(age, bs = "cr") + sex
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.895726   0.006536  137.04   <2e-16 ***
## sex         0.092007   0.008750   10.52   <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(age) 4.678  5.718 143.6  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.539   Deviance explained = 54.2%
## GCV = 0.01523  Scale est. = 0.015104  n = 802
```

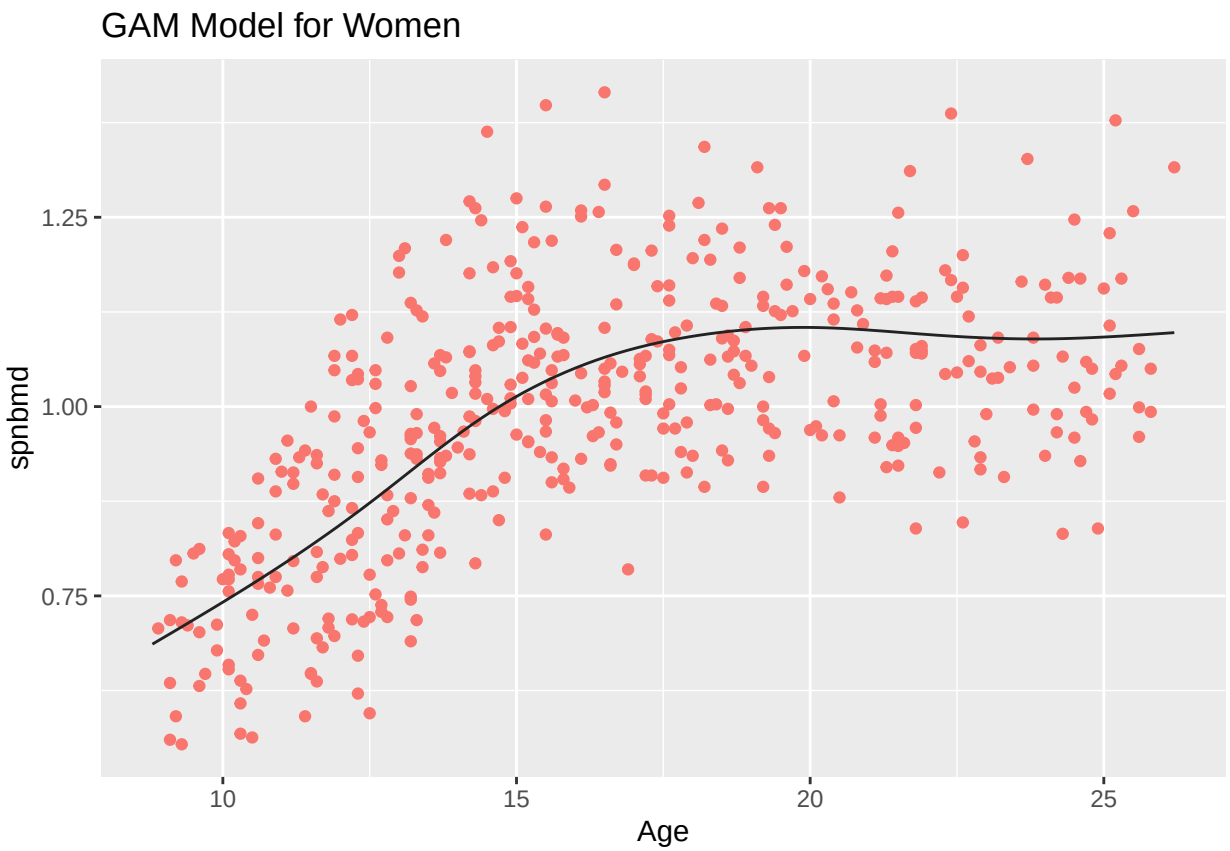
As we said before, `gam()` performs generalised cross validation on the damping parameter by convention. Below we plot the model for both male and female.

```
# Load the ggplot2 package
library(ggplot2)

# Create a new data frame for predictions
newdata <- data.frame(age = seq(min(train$age), max(train$age), length.out = 100))

# Generate predictions for men and women
newdata$sex <- 0  # for men
```

```r
newdata$spnbmd <- predict(gam_model,newdata)

# Create the plot for men
plot_male <- ggplot(bmd_train_male, aes(x = age, y = spnbmd)) +
  geom_point(color = "#00BFC4")+
  geom_line(data = newdata, aes(x = age, y = spnbmd), color = "#222222") +
  labs(title = "GAM Model for Men", x = "Age", y = "spnbmd")

plot_male
```




```r
# Generate predictions for women
newdata$sex <- 1  # for women
newdata$spnbmd <- predict(gam_model, newdata = newdata)

# Create the plot for women
plot_female <- ggplot(bmd_train_female, aes(x = age, y = spnbmd)) +
  geom_point(color = "#F8766D")+
  geom_line(data = newdata, aes(x = age, y = spnbmd), color = "#222222") +
  labs(title = "GAM Model for Women", x = "Age", y = "spnbmd")

# Print the plots

plot_female
```



And finally we find the mse for this final model:

```
mse_male <- mse(gam_model_male,bmd_test_male)
mse_female <- mse(gam_model_female,bmd_test_female)

# Compute the overall test error
mse_all <- weight_male * mse_male + weight_female * mse_female


print(paste("MSE for male test set:",mse_male))
```

```
## [1] "MSE for male test set: 0.0181317754660071"
```

```
print(paste("MSE for female test set:",mse_female))
```

```
## [1] "MSE for female test set: 0.0149056477902181"
```

```
print(paste("MSE for entire test set:",mse_all))
```

```
## [1] "MSE for entire test set: 0.0165427872072077826"
```

Again we see a very low mean square error compared to our initial linear regression, unfortunately it was still beat out by polynomial regression and it was almost indistinguishable from the other GAM model.

Question 5: Weird Model

In this question we consider the strange model:

$$\textbf{spnbmd} = a + f_1(\textbf{age}) + f_2(s_i\textbf{age}) + \epsilon$$

where s is an indictor function on being a woman. One clear obervation about this model is that its not linear, so fitting GAM would not be suitable, and in fact many of our optimisation algorithms would not be suitable. However notice that when s = 0 we have that $\textbf{spnbmd} = a + f_1(\textbf{age}) + \epsilon$ and if s =1 then $\textbf{spnbmd} = a + f_1(\textbf{age}) + f_2(\textbf{age}) + \epsilon = a + f_3(\textbf{age}) + \epsilon$. So in fact our final model is equal to our first one! As we said thought his model is not linear and GAM would not work for it.

Question 6: Choices

I think overall I'd prefer the generalised linear model from question 3, as it keeps the simplicity of a single variable model while getting the accuracy of cubic splines. That being said the mean square error of the GAM is not always better than polynomial regression, and in fact polynomial regression marginally wins in our case.