DATA 558 Homework 4

Shrusti Ghela

5/24/2022

1. Suppose that a curve $\hat{g}$ is computed to smoothly fit a set of n points using the following formula:

$$\hat{g} = \underset{g}{\mathbf{argmin}}\left(\sum_{i=1}^{n}\left(y_i - g(x_i)\right)^2 + \lambda \int \left[g^{(m)}(x)\right]^2 dx\right)$$

where $g^{(m)}$ represents the m^{th} derivative of g (and $g^{(0)} = g$). Provide example sketches of $\hat{g}$ in each of the following scenarios.

Before I begin, let me generate a set of datapoints on which we sketch the changes in $\hat{g}$ under the given conditions.

The example I am going to use is:

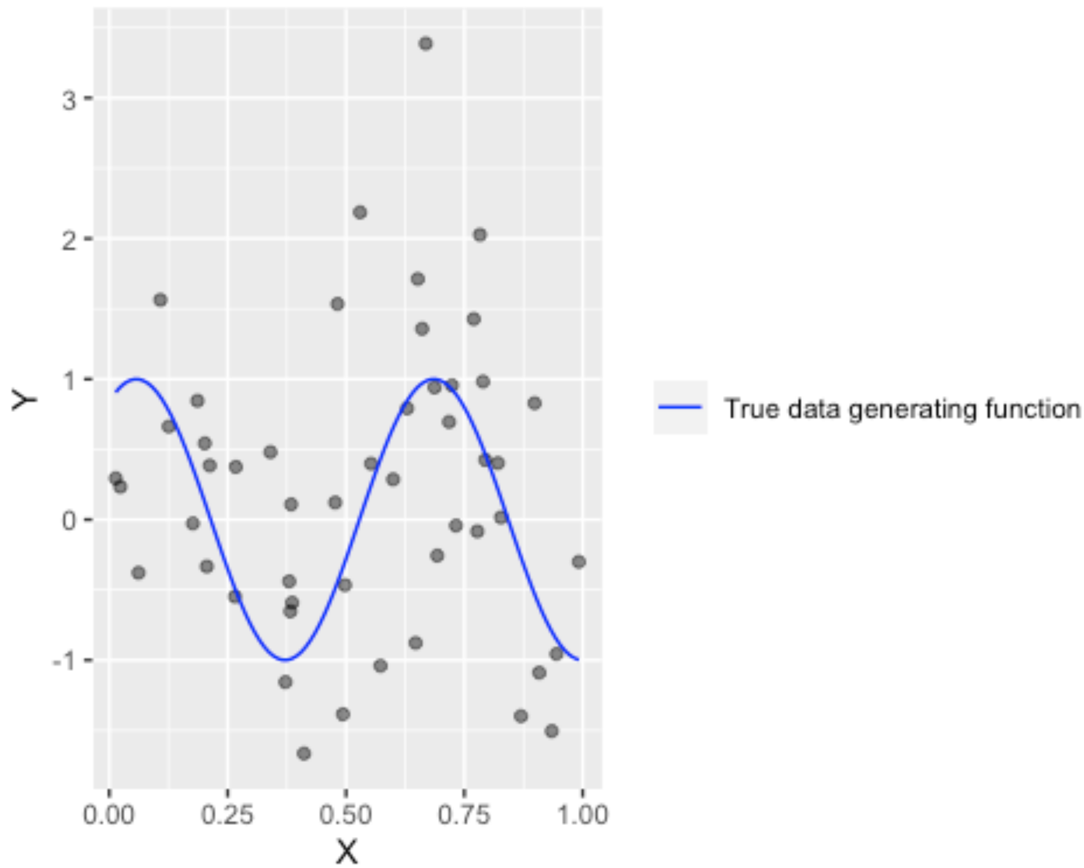
$$Y = g(X) + \epsilon$$

Here $g(X)$ is the true data-generating function.

Let us take $g(X) = \sin\left(10(X + 0.1)\right)$

Here, $X \sim \mathcal{U}(50)$ and $\epsilon \sim \mathcal{N}(0,1)$

```
library(ggplot2)
set.seed(1)

X <- runif(50)
eps <- rnorm(50)
Y <- sin(10*(X + 0.1)) + eps
generating_fn <- function(X) {sin(10*(X + 0.1))}
df <- data.frame(X, Y)

ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.5) +
  stat_function(fun = generating_fn, aes(col = "True data generating
function")) +
  scale_color_manual(values = "blue") +
  theme(legend.position = "right", legend.title = element_blank())
```



a. $\lambda = \infty, m = 0$

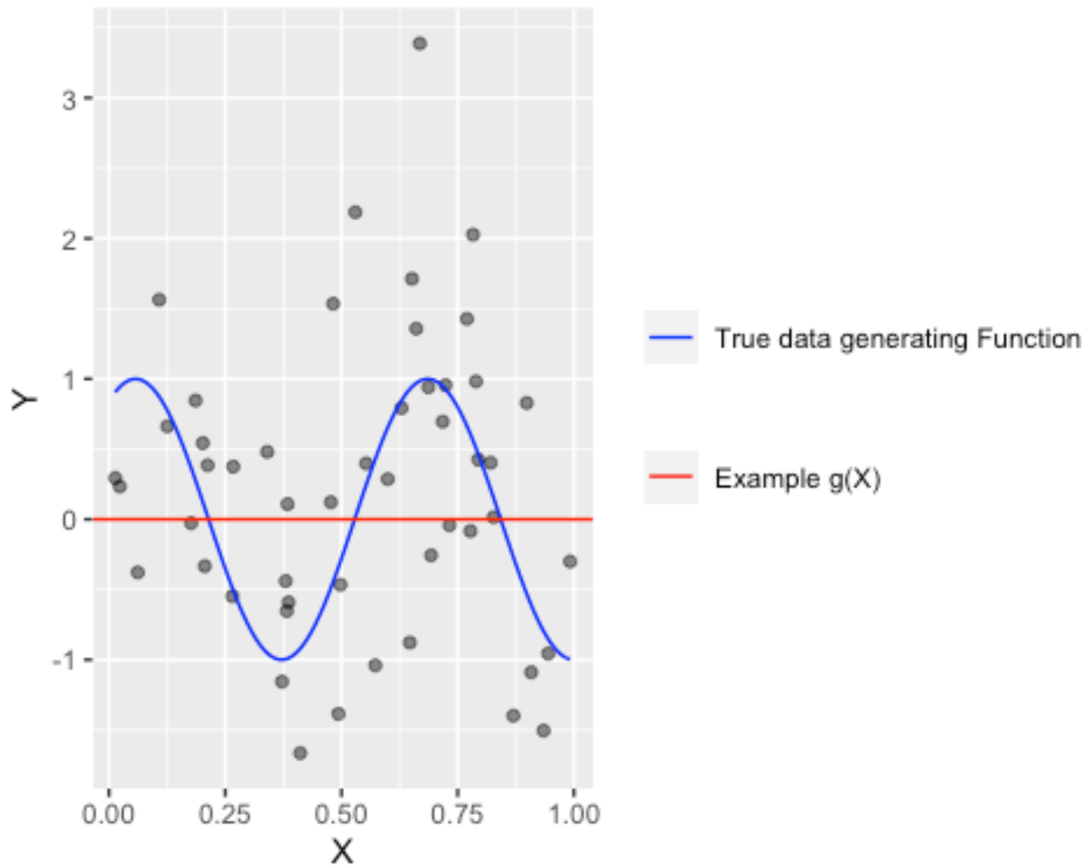
For $m = 0$, we get

$$\hat{g} = \underset{g}{\operatorname{argmin}}\left(\sum_{i=1}^{n} \left(y_i - g(x_i)\right)^2 + \lambda \int [g(x)]^2 \, dx\right)$$

So as λ increases, the penalty term gets more and more dominant.As $\lambda \to \infty$, this forces $g(x) \to 0$.

Hence, $\hat{g}(x) = 0$

```
ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.5) +
  stat_function(fun = generating_fn, aes(col = "True data generating
Function")) +
  geom_hline(aes(yintercept = 0, linetype = "Example g(X)"), col = "red") +
  scale_color_manual(values = "blue") +
  theme(legend.position = "right", legend.title = element_blank())
```



b. $\lambda = \infty, m = 1$

For m = 1, we get

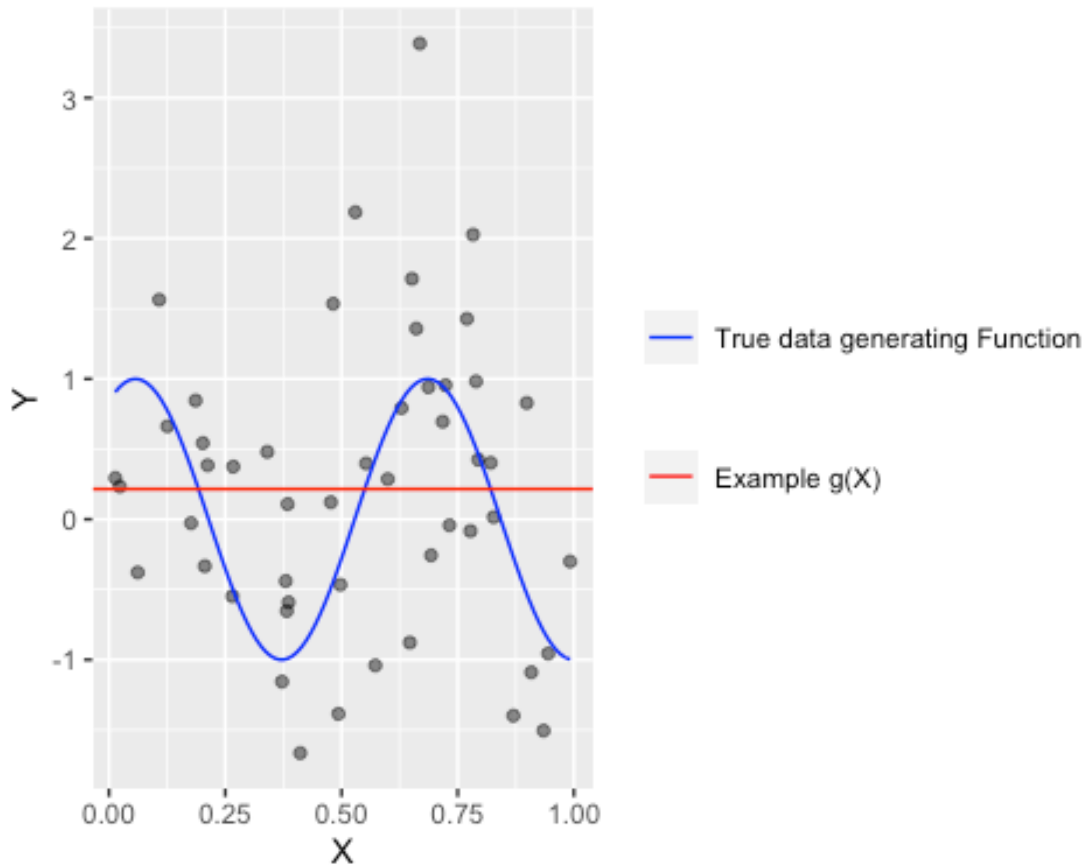
$$\hat{g} = \underset{g}{\operatorname{argmin}}(\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int [g'(x)]^2\, dx)$$

As $\lambda \to \infty$, this forces $g'(x) \to 0$.

This means we would get $\hat{g}(x) = (\text{some constant})\ c$

Now, we can take $\hat{g}(x) = c = \sum_{i=1}^{n} y_i$, because all other constant function will have a first derivative of zero but $\hat{g}(x) = c = \sum_{i=1}^{n} y_i$ will also minimize the RSS.

```
ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.5) +
  stat_function(fun = generating_fn, aes(col = "True data generating 
Function")) +
  geom_hline(aes(yintercept = mean(Y), linetype = "Example g(X)"), col = 
"red") +
  scale_color_manual(values = "blue") +
  theme(legend.position = "right", legend.title = element_blank())
```



c. $\lambda = \infty, m = 2$

For m = 2, we get

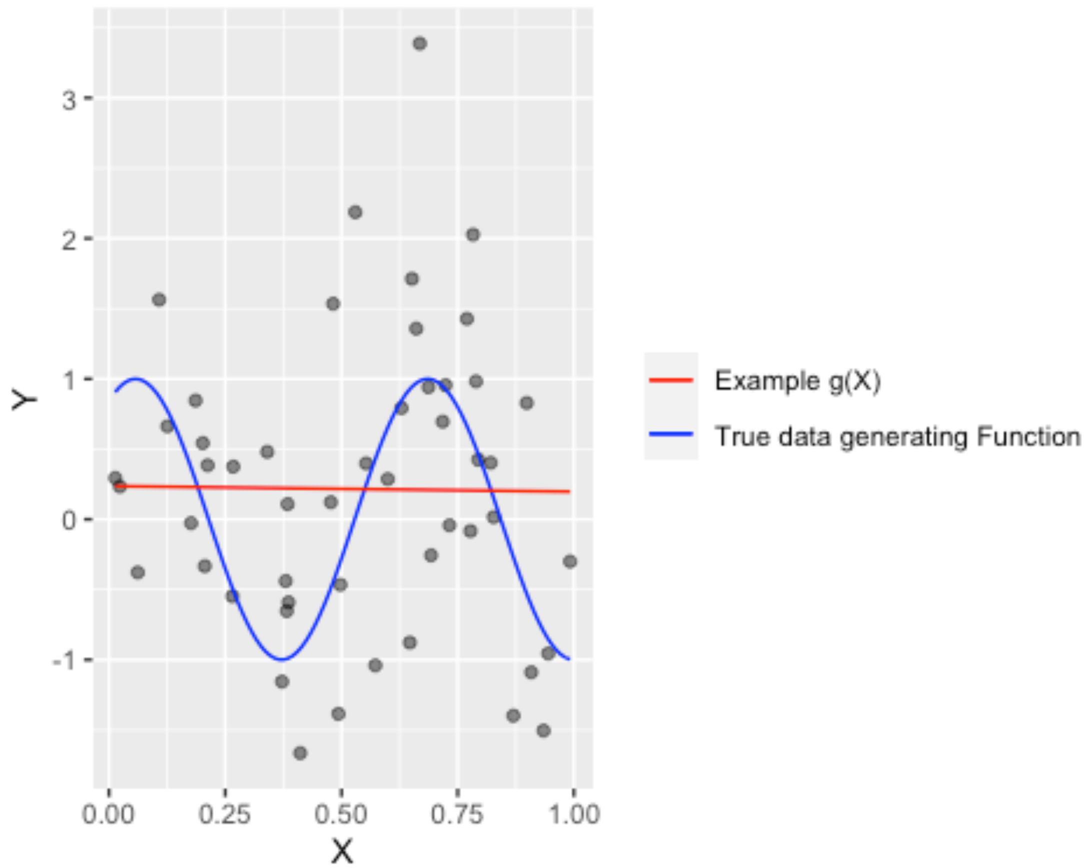
$$\hat{g} = \underset{g}{\operatorname{argmin}}(\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int [g''(x)]^2 \, dx)$$

As $\lambda \to \infty$, this forces $g''(x) \to 0$.

This means we would get $\hat{g}(x) = wx + b$

Now, we can take $\hat{g}(x) = wx + b$ to be the linear least squares line, because all other linear function will have a second derivative of zero but LLS will also minimize the RSS.

```
ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.5) +
  stat_function(fun = generating_fn, aes(col = "True data generating
Function")) +
  geom_smooth(method = "lm", formula = "y ~ x", se = F, aes(col = "Example
g(X)"), size=0.5) +
  scale_color_manual(values = c("red", "blue")) +
  theme(legend.position = "right", legend.title = element_blank())
```



d. $\lambda = \infty, m = 3$

For m = 3, we get

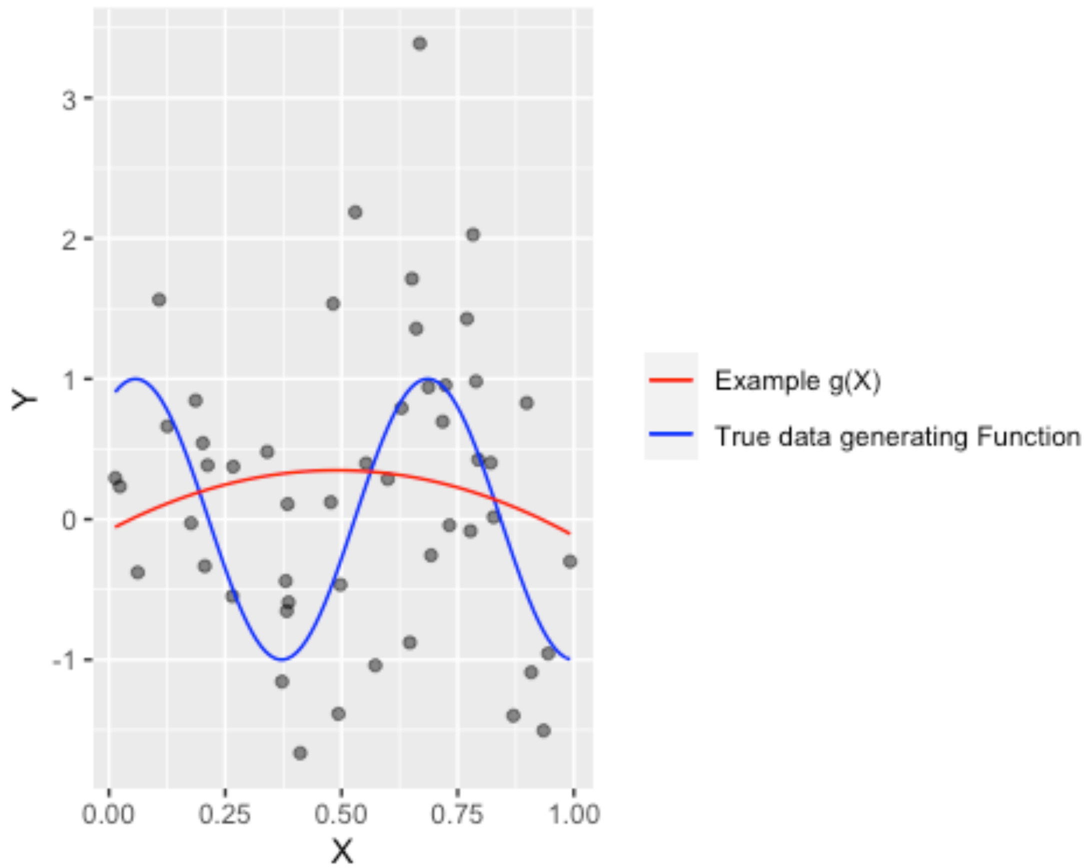
$$\hat{g} = \underset{g}{\operatorname{argmin}}\left(\sum_{i=1}^{n}\left(y_i - g(x_i)\right)^2 + \lambda \int \left[g^{(3)}(x)\right]^2 dx\right)$$

As $\lambda \to \infty$, this forces $g^{(3)}(x) \to 0$.

This means we would get $\hat{g}(x) = ux^2 + wx + b$

Now, we can take $\hat{g}(x) = ux^2 + wx + b$ to be the quadratic least squares line, because all other linear function will have a third derivative of zero but this will also minimize the RSS.

```
ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.5) +
  stat_function(fun = generating_fn, aes(col = "True data generating
Function")) +
  geom_smooth(method = "lm", formula = "y ~ x + I(x^2)", se = F, aes(col =
"Example g(X)"), size=0.5) +
  scale_color_manual(values = c("red", "blue")) +
  theme(legend.position = "right", legend.title = element_blank())
```



e. $\lambda = 0, m = 3$

For m = 3, we get

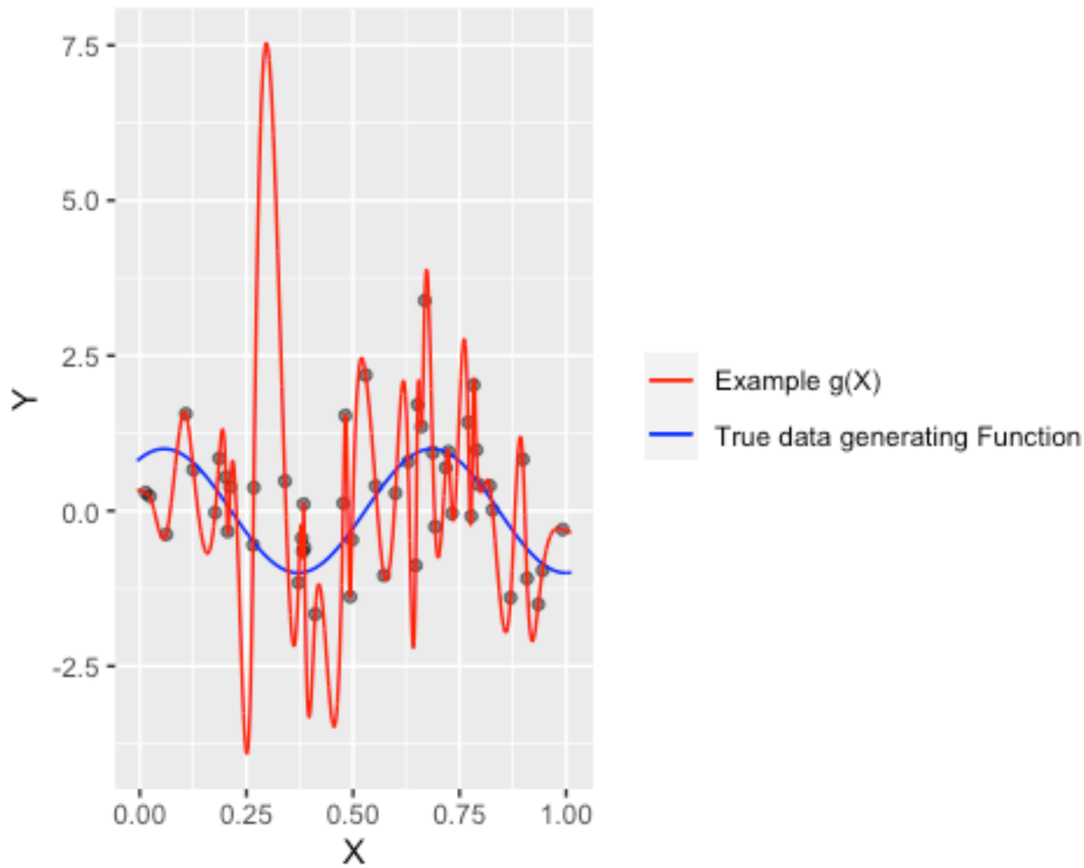
$$\hat{g} = \underset{g}{\operatorname{argmin}}\left(\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int \left[g^{(3)}(x)\right]^2 dx\right)$$

But now that $\lambda = 0$, the penalty term is no longer considered in the selection of $\hat{g}(x)$. Becasue of this we can achieve RSS=0 by simply fitting "connect the dots" model.

Taking a cubic smoothing spline (with no smoothing) as an example:

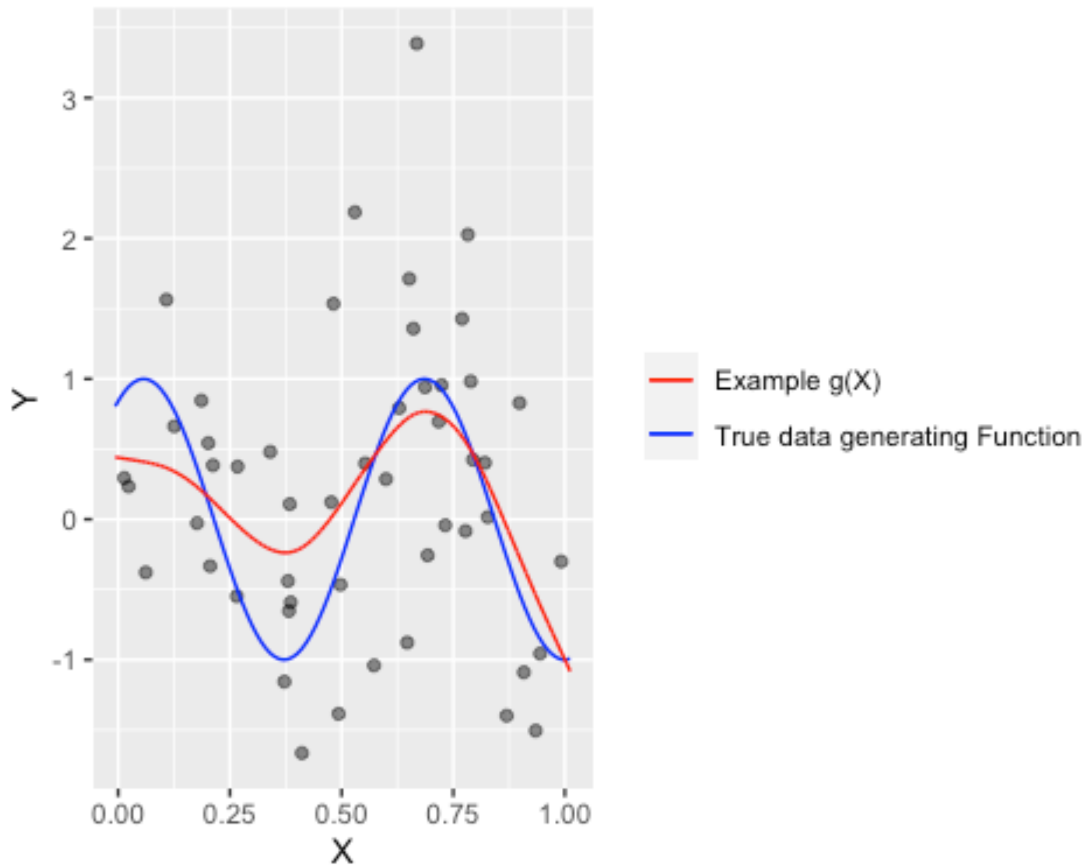
```
interp_spline <- smooth.spline(x = df$X, y = df$Y, all.knots = T, lambda =
0.0000000000001)
fitted <- predict(interp_spline, x = seq(min(X) - 0.02, max(X) + 0.02, by =
0.0001))
fitted <- data.frame(x = fitted$x, fitted_y = fitted$y)

ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.5) +
  stat_function(fun = generating_fn, aes(col = "True data generating
Function")) +
  geom_line(data = fitted,
```

```
            aes(x = x, y = fitted_y, col = "Example g(X)")) +
  scale_color_manual(values = c("red", "blue")) +
  theme(legend.position = "right", legend.title = element_blank())
```



Using Cross Validation to select the value of λ in the above approach

```
smooth_spline <- smooth.spline(x = df$X, y = df$Y, all.knots = T)
fitted <- predict(smooth_spline, x = seq(min(X) - 0.02, max(X) + 0.02, by =
0.0001))
fitted <- data.frame(x = fitted$x, fitted_y = fitted$y)

ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.5) +
  stat_function(fun = generating_fn, aes(col = "True data generating
Function")) +
  geom_line(data = fitted,
            aes(x = x, y = fitted_y, col = "Example g(X)")) +
  scale_color_manual(values = c("red", "blue")) +
  theme(legend.position = "right", legend.title = element_blank())
```



The above fit is much closer to the true generating function than anything so far!

2. Suppose we fit a curve with basis functions $b_1(X) = I(0 \leq X \leq 2) - (X+1)I(1 \leq X \leq 2), b_2(X) = (2X-2)I(3 \leq X \leq 4) - I(4 < X \leq 5)$. We fit the linear regression model

$$Y = \beta_0 + \beta_1 b_1(X) + \beta_2 b_2(X) + \epsilon$$

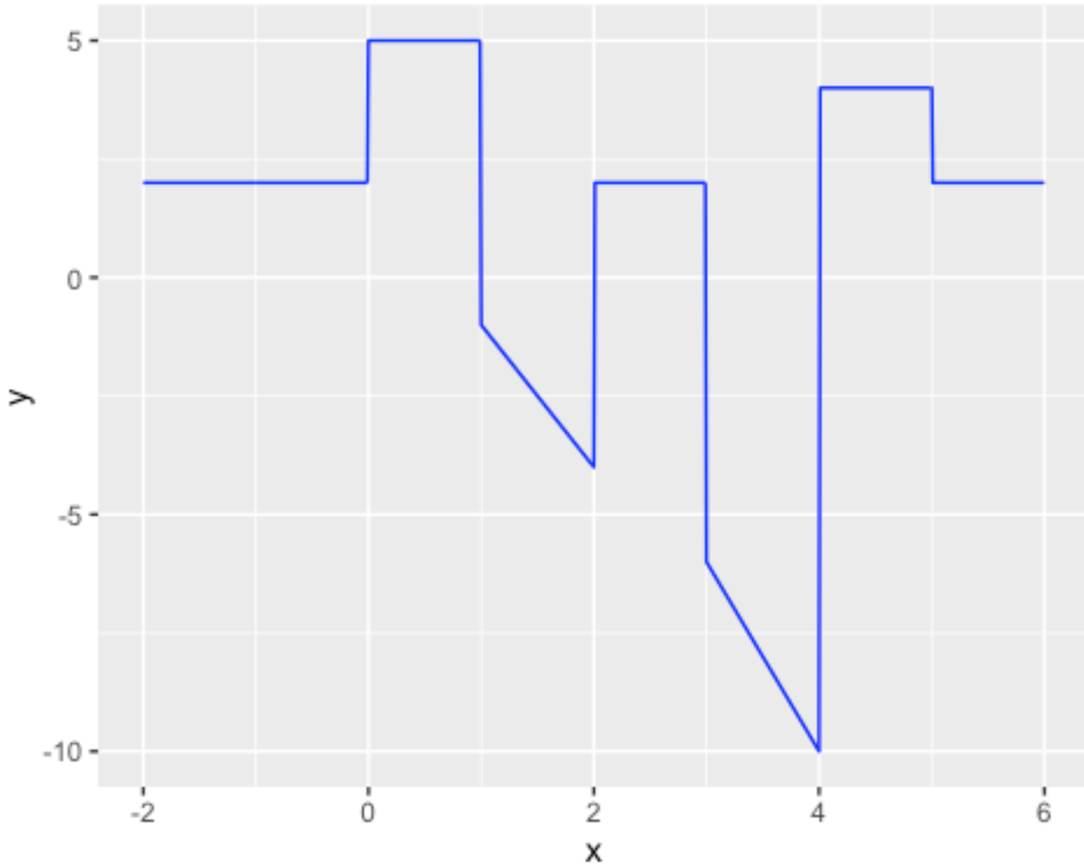
and obtain coefficient estimates $\hat{\beta}_0 = 2, \hat{\beta}_1 = 3, \hat{\beta}_2 = -2$. Sketch the estimated curve between $X = -2$ and $X = 6$. Note the intercepts, slopes, and other relevant information.

Using the information from the question, we can simplify the linear regression model to the following

$$Y = \begin{cases} 2 & -2 \leq x < 0 \\ 5 & 0 \leq x < 1 \\ 2 - 3x & 1 \leq x \leq 2 \\ 2 & 2 < x < 3 \\ 6 - 4x & 3 \leq x \leq 4 \\ 4 & 4 < x \leq 5 \\ 2 & 5 < x \leq 6 \end{cases}$$

```
x <- seq(-2, 6, 0.01)
y <- (x >= -2 & x < 0) * 2 +
```

```
    (x >= 0 & x <1) * 5 +
    (x >= 1 & x <= 2) * (2 - 3*x) +
    (x > 2 & x < 3) * 2 +
    (x >= 3 & x <= 4) * (6 - 4*x) +
    (x >4 & x <= 5) * 4 +
    (x >5 & x <= 6) * 2

ggplot()+
  geom_line(aes(x,y), col="blue")
```



3. Prove that any function of the form

$$f(X) = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 (X - \psi)_+^3$$

is a cubic spline with a knot at ψ

In the chapter we discussed that a cubic regression spline with one knot at ψ can be obtained using a basis of the form $x, x^2, x^3, (x-\psi)_+^3$, where $(x-\psi)_+^3 = (x-\psi)^3$ if $x > \psi$ and equals 0 otherwise.

Let us now understand why a function of the form

$$f(X) = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 (X - \psi)_+^3$$

is a cubic spline regardless of the values of $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4$

Let us first find a cubic polynomial

$$f_1(x) = a_1 + b_1 x + c_1 x^2 + d_1 x^3$$

such that $f(x) = f_1(x)$ for all $x \leq \psi$

For $x \leq \psi$,

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

Because $f(x) = f_1(x)$,

$$a_1 + b_1 x + c_1 x^2 + d_1 x^3 = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

So we take, $a_1 = \beta_0, b_1 = \beta_1, c_1 = \beta_2, d_1 = \beta_3$

Now let us find a cubic polynomial

$$f_2(x) = a_2 + b_2 x + c_2 x^2 + d_2 x^3$$

such that $f(x) = f_2(x)$ for all $x > \psi$

For $x > \psi$,

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - \psi)^3$$

Expanding $ _4(x-)^3$

$$\beta_4 (x - \psi)^3 = \beta_4 (x^3 - \psi^3 + 3\psi^2 x - 3\psi x^2)$$

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^3 - \beta_4 \psi^3 + 3\beta_4 \psi^2 x - 3\beta_4 \psi x^2$$

$$f(x) = (\beta_0 - \beta_4 \psi^3) + (\beta_1 + 3\beta_4 \psi^2) x + (\beta_2 - 3\beta_4 \psi) x^2 + (\beta_3 + \beta_4) x^3$$

So, $a_2 = \beta_0 - \beta_4 \psi^3, b_2 = \beta_1 + 3\beta_4 \psi^2, c_2 = \beta_2 - 3\beta_4 \psi, d_2 = \beta_3 + \beta_4$

Now,

$$f_1(\psi) = \beta_0 + \beta_1 \psi + \beta_2 \psi^2 + \beta_3 \psi^3$$

and

$$\begin{aligned} f_2(\psi) = (\beta_0 - \beta_4 \psi^3) + (\beta_1 + 3\beta_4 \psi^2)\psi + (\beta_2 - 3\beta_4 \psi)\psi^2 + (\beta_3 + \beta_4)\psi^3 \\ = \beta_0 + \beta_1 \psi + \beta_2 \psi^2 + \beta_3 \psi^3 = f_1(\psi) \end{aligned}$$

$f_1(\psi) = f_2(\psi) \Longrightarrow f(x)$ is continuous at ψ

Also,

$$f_1'(\psi) = \beta_1 + 2\beta_2 \psi + 3\beta_3 \psi^2$$

and

$$f_2'(\psi) = (\beta_1 + 3\beta_4\psi^2) + 2(\beta_2 - 3\beta_4\psi)\psi + 3(\beta_3 + \beta_4)\psi^2 = \beta_1 + 2\beta_2\psi + 3\beta_3\psi^2$$

$f_1'(\psi) = f_2'(\psi) \Longrightarrow f'(x)$ is continuous at ψ

Finally,

$$f_1''(\psi) = 2\beta_2 + 6\beta_3\psi$$

and

$$f_2''(\psi) = 2(\beta_2 - 3\beta_4\psi) + 6(\beta_3 + \beta_4)\psi = 2\beta_2 + 6\beta_3\psi$$

$f_1''(\psi) = f_2''(\psi) \Longrightarrow f''(x)$ is continuous at ψ

Therefore, $f(x)$ is indeed a cubic spline.

4. For this problem, we will use the Wage data set that is part of the ISLR package. Split the data into a training set and a test set, and then fit the models to predict Wage using Age on the training set. Make some plots, and comment on your results. Which approach yields the best results on the test set?

```
library(ISLR2)

set.seed(7)
dt <- sample(1:nrow(Wage), nrow(Wage) / 2)
train <- Wage[dt,]
test <- Wage[-dt,]
testVar <- Wage$wage[-dt]
```

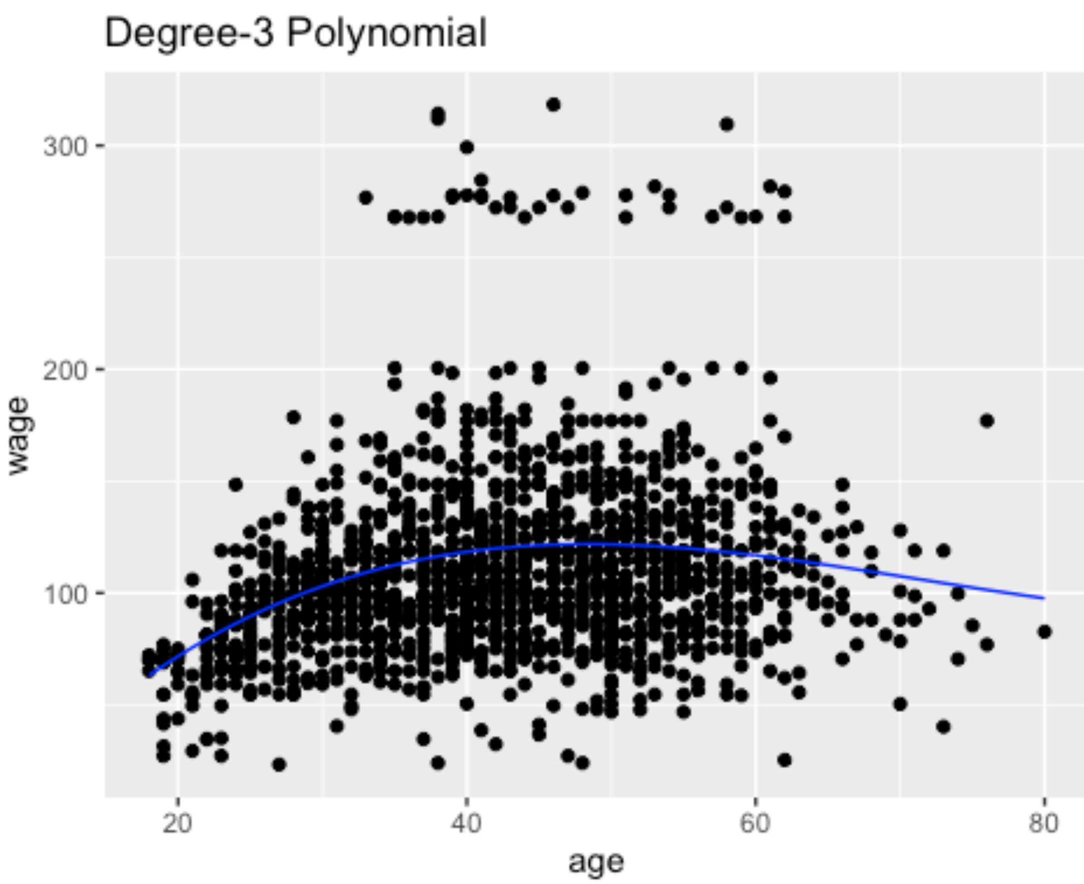
a. Polynomial

```
lm.Wage1 <- lm(wage ~ poly(age, 1), data = train)
lm.Wage2 <- lm(wage ~ poly(age, 2), data = train)
lm.Wage3 <- lm(wage ~ poly(age, 3), data = train)
lm.Wage4 <- lm(wage ~ poly(age, 4), data = train)
lm.Wage5 <- lm(wage ~ poly(age, 5), data = train)
lm.Wage6 <- lm(wage ~ poly(age, 6), data = train)

lm.Wagepred <- predict(lm.Wage1, test)
mean((lm.Wagepred - testVar)^2)

## [1] 1634.738

lm.Wagepred <- predict(lm.Wage2, test)
mean((lm.Wagepred - testVar)^2)

## [1] 1555.61

lm.Wagepred <- predict(lm.Wage3, test)
mean((lm.Wagepred - testVar)^2)

## [1] 1548.802

lm.Wagepred <- predict(lm.Wage4, test)
mean((lm.Wagepred - testVar)^2)
```

```
## [1] 1547.226

lm.Wagepred <- predict(lm.Wage5, test)
mean((lm.Wagepred - testVar)^2)

## [1] 1546.823

lm.Wagepred <- predict(lm.Wage6, test)
mean((lm.Wagepred - testVar)^2)

## [1] 1545.156
```

Usually, it is better to use a simple model that explains relatively the most variation. There is a significant drop in test MSE till polynomial of order 3. After that, there is only a little difference in the test MSE. So, Model 3 appears to be the best model for this dataset.

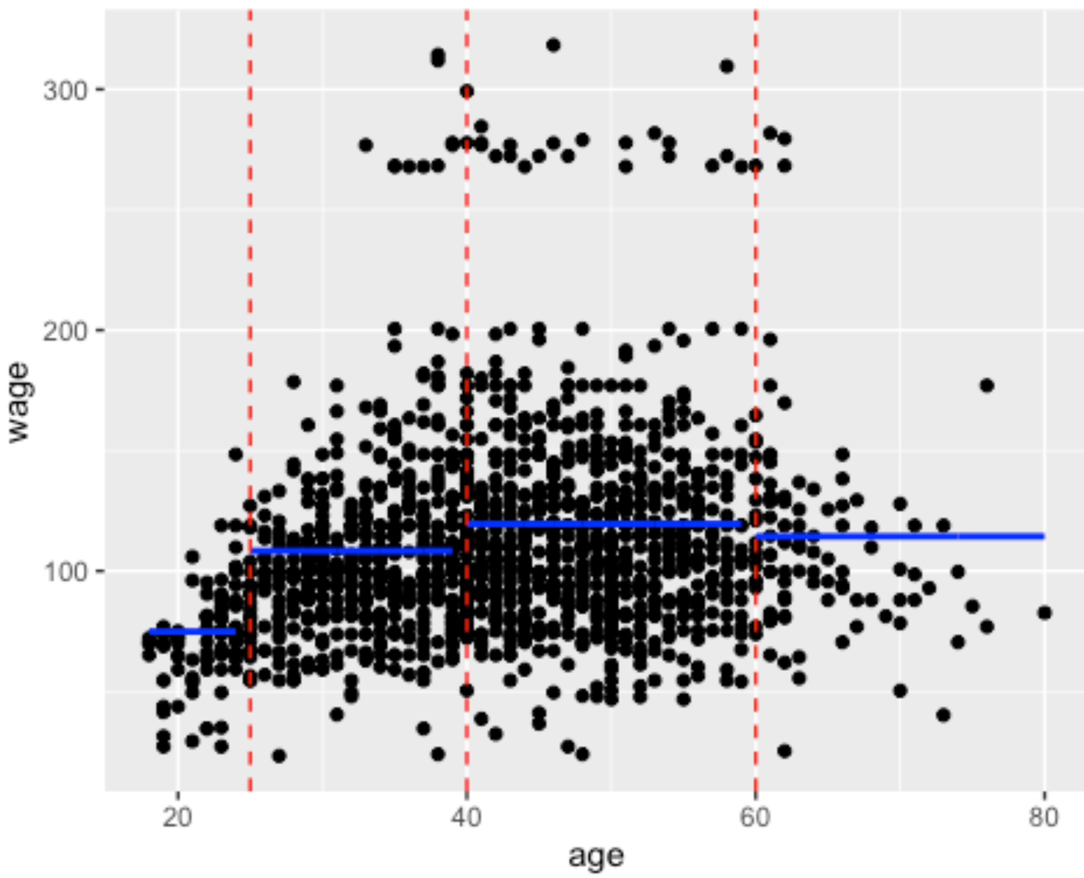
```
lm.Wagepred <- predict(lm.Wage3, test)
ggplot() +
  geom_point(data =test, aes(x = age, y = wage)) +
  geom_line(data = test, aes(x = age, y = lm.Wagepred), color = "blue")  +
  labs(title = "Degree-3 Polynomial")
```




```
mean((lm.Wagepred - testVar)^2)

## [1] 1548.802
```

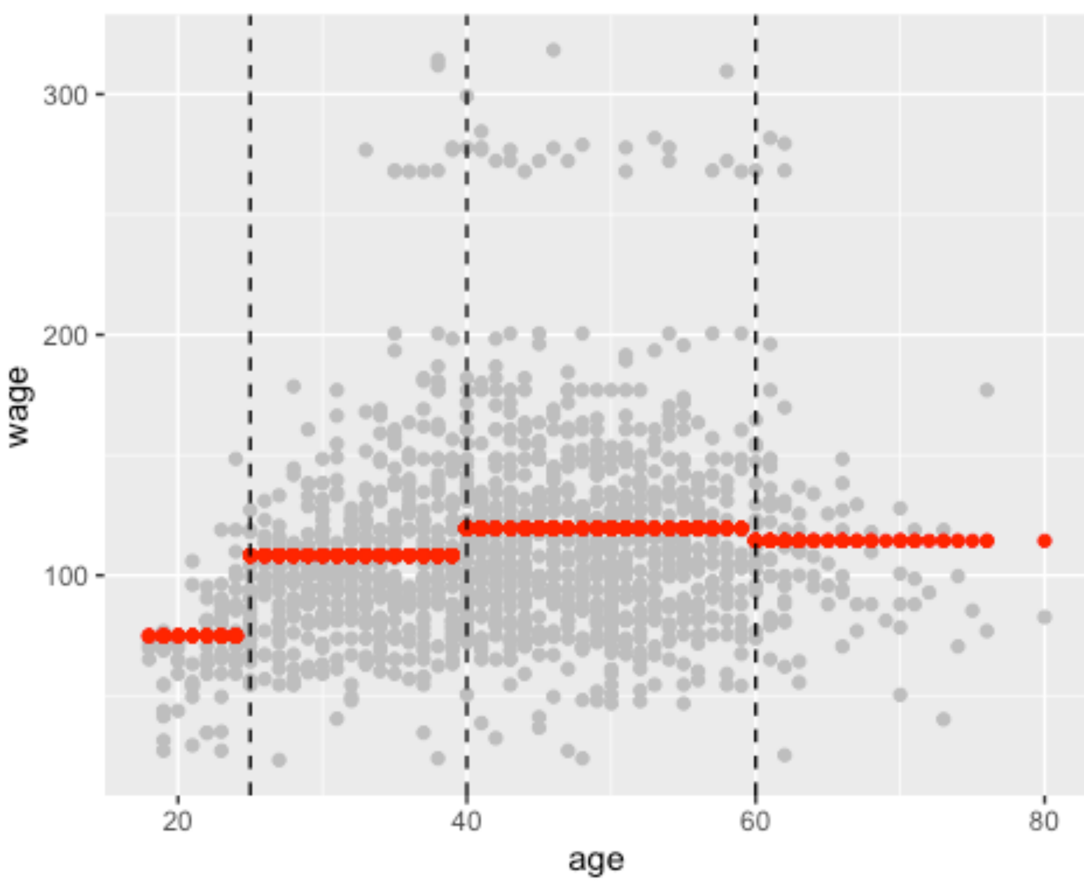
b. Step Function

```
pred1 <- mean(train[train$age<25,]$wage)
pred2 <- mean(train[train$age >=25 & train$age<40,]$wage)
pred3 <- mean(train[train$age >=40 & train$age<60,]$wage)
pred4 <-  mean(train[train$age>=60,]$wage)

ggplot()+
  geom_point(data=test, aes(x=age, y=wage))+
  geom_line(data=test[test$age<25,],
            aes(y = pred1, x=age), size = 1, col="blue") +
  geom_line(data=test[test$age >=25 & test$age<40,],
            aes(y = pred2, x=age), size = 1, col="blue") +
  geom_line(data=test[test$age>=40 & test$age<60,],
            aes(y = pred3, x=age), size = 1, col="blue") +
  geom_line(data=test[test$age>=60,],
            aes(y = pred4, x=age), size = 1, col="blue") +
  geom_vline(xintercept = 25, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="red", size=0.5)
```



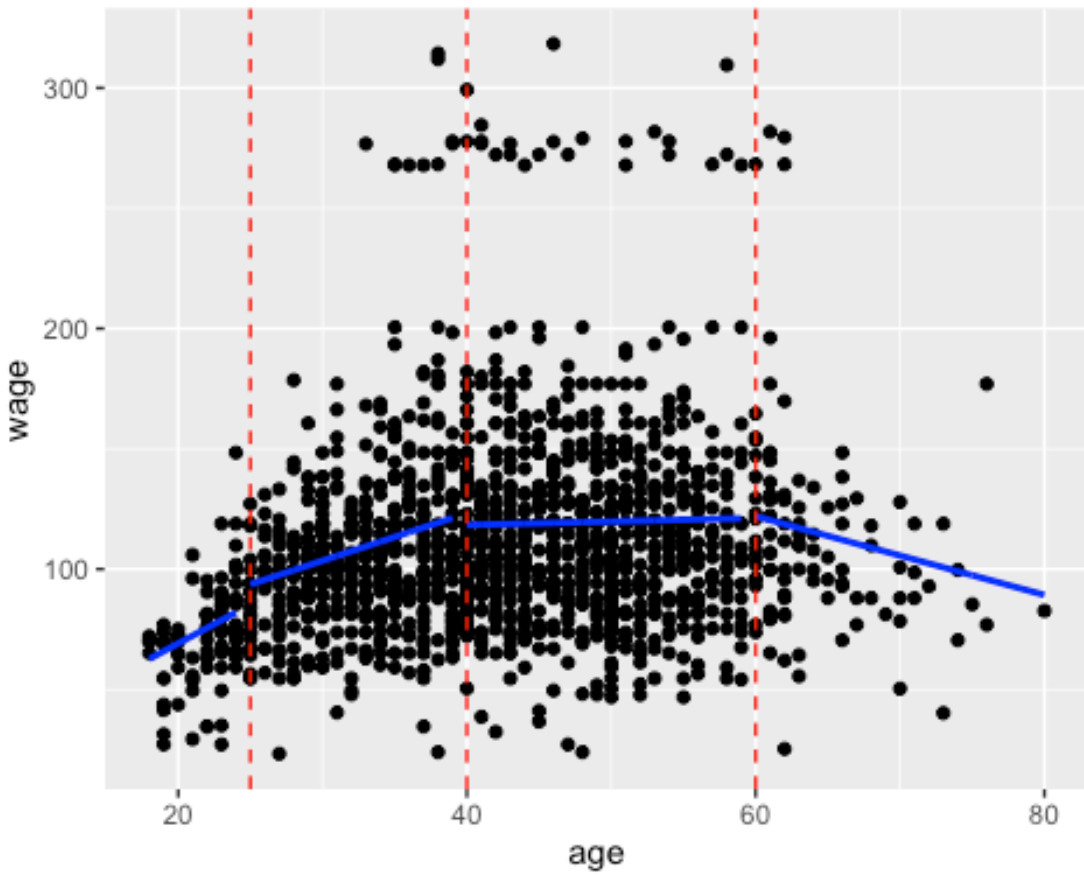

```
for (i in 1:length(test$age)){
  if (test$age[i]<25){
    test$pred_step[i] <- pred1
```

```r
  }
  else if (test$age[i]>=25 & test$age[i]<40){
    test$pred_step[i] <- pred2
  }
  else if (test$age[i]>=40 & test$age[i]<60){
    test$pred_step[i] <- pred3
  }
  else if (test$age[i]>=60){
    test$pred_step[i] <- pred4
  }
}

mean((test$pred_step- test$wage)^2)

## [1] 1585.837

ggplot()+
  geom_point(data=test, aes(x=age, y=wage), col="gray")+
  geom_point(data=test, aes(x=age, y=pred_step), col="red")+
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```



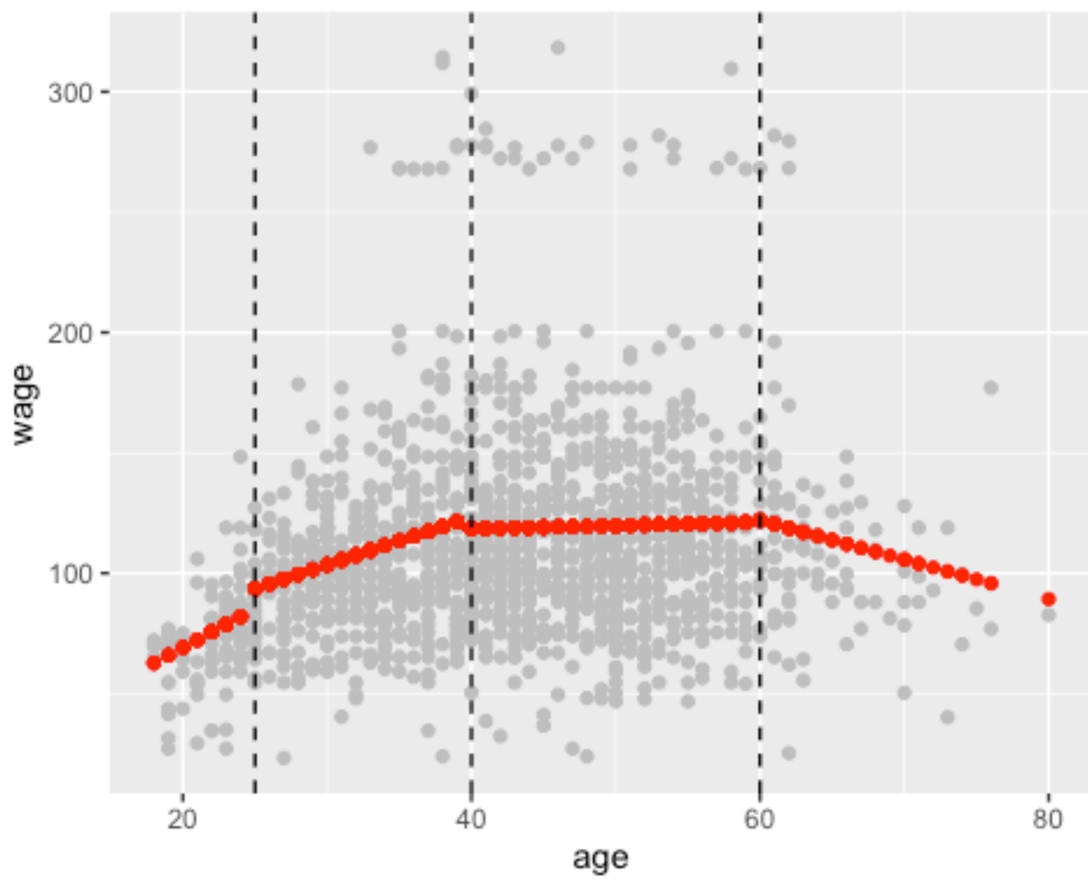
c. piecewise polynomial

```
lm.1 <- lm(wage~age, data = train[train$age<25,])
lm.2 <- lm(wage~age, data = train[train$age >=25 & train$age<40,])
lm.3 <- lm(wage~age, data = train[train$age>=40 & train$age<60,])
lm.4 <- lm(wage~age, data = train[train$age>=60,])

pred1 <- predict(lm(wage~age,
                    data = train[train$age<25,]))
pred2 <- predict(lm(wage~age,
                    data = train[train$age >=25 & train$age<40,]))
pred3 <- predict(lm(wage~age,
                    data = train[train$age>=40 & train$age<60,]))
pred4 <-  predict(lm(wage~age,
                    data = train[train$age>=60,]))

ggplot() +
  geom_point(data = test, aes(x = age, y = wage)) +
  geom_line(data=train[train$age<25,],
            aes(y = pred1, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age >=25 & train$age<40,],
            aes(y = pred2, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age>=40 & train$age<60,],
            aes(y = pred3, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age>=60,],
            aes(y = pred4, x=age), size = 1, col="blue") +
  geom_vline(xintercept = 25, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="red", size=0.5)
```



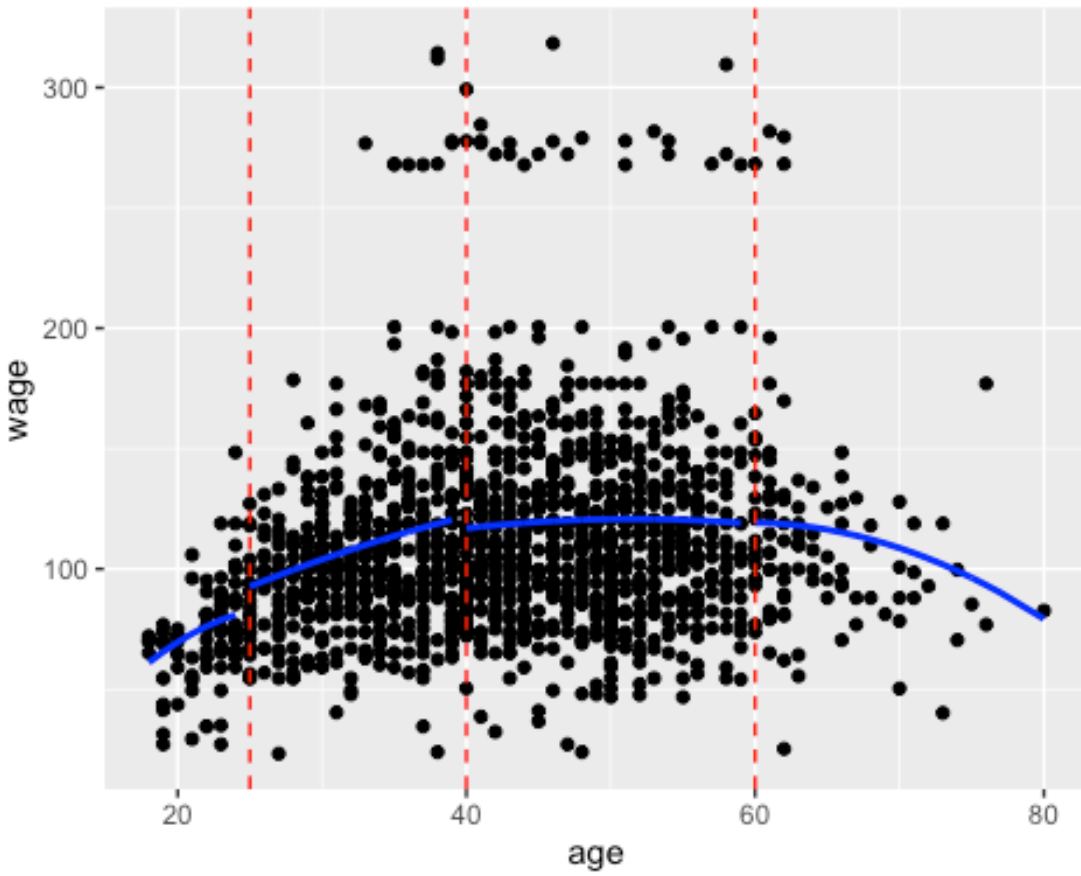

```
for (i in 1:length(test$age)){
  if (test$age[i]<25){
    test$pred_pLR[i] <- lm.1$coefficients[1] +
lm.1$coefficients[2]*test$age[i]
  }else if (test$age[i]>=25 & test$age[i]<40){
    test$pred_pLR[i] <- lm.2$coefficients[1] +
lm.2$coefficients[2]*test$age[i]
  }else if (test$age[i]>=40 & test$age[i]<60){
    test$pred_pLR[i] <- lm.3$coefficients[1] +
lm.3$coefficients[2]*test$age[i]
  }else if (test$age[i]>=60){
    test$pred_pLR[i] <- lm.4$coefficients[1] +
lm.4$coefficients[2]*test$age[i]
  }
}

mean((test$pred_pLR- test$wage)^2)

## [1] 1541.278

ggplot()+
  geom_point(data=test, aes(x=age, y=wage), col="gray")+
  geom_point(data=test, aes(x=age, y=pred_pLR), col="red")+
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
```

```
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```



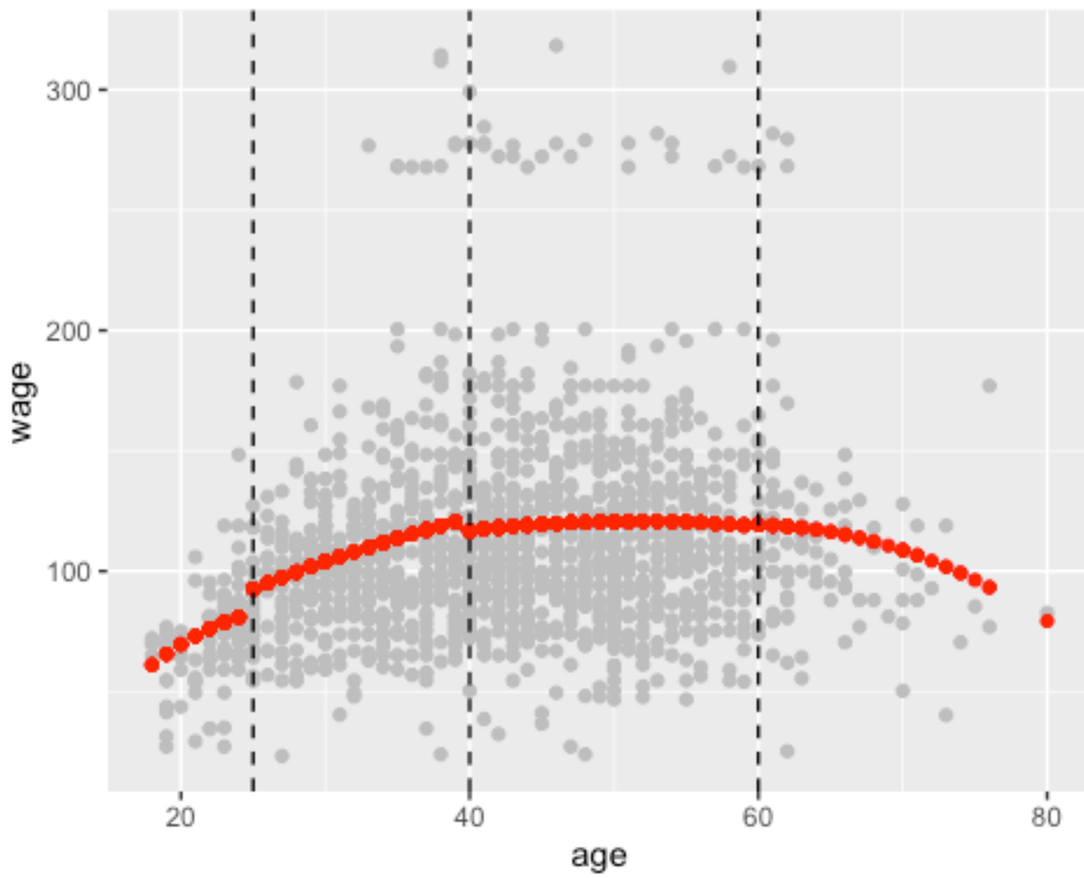

```
lm.1 <- lm(wage~poly(age,2, raw=TRUE), data = train[train$age<25,])
lm.2 <- lm(wage~poly(age,2, raw=TRUE), data = train[train$age >=25 &
train$age<40,])
lm.3 <- lm(wage~poly(age,2, raw=TRUE), data = train[train$age>=40 &
train$age<60,])
lm.4 <- lm(wage~poly(age,2, raw=TRUE), data = train[train$age>=60,])

pred1 <- predict(lm(wage~poly(age,2, raw=TRUE),
                    data = train[train$age<25,]))
pred2 <- predict(lm(wage~poly(age,2, raw=TRUE),
                    data = train[train$age >=25 & train$age<40,]))
pred3 <- predict(lm(wage~poly(age,2, raw=TRUE),
                    data = train[train$age>=40 & train$age<60,]))
pred4 <-  predict(lm(wage~poly(age,2, raw=TRUE),
                    data = train[train$age>=60,]))

ggplot() +
  geom_point(data = test, aes(x = age, y = wage)) +
  geom_line(data=train[train$age<25,],
            aes(y = pred1, x=age), size = 1, col="blue") +
```

```
  geom_line(data=train[train$age >=25 & train$age<40,],
            aes(y = pred2, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age>=40 & train$age<60,],
            aes(y = pred3, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age>=60,],
            aes(y = pred4, x=age), size = 1, col="blue") +
  geom_vline(xintercept = 25, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="red", size=0.5)
```



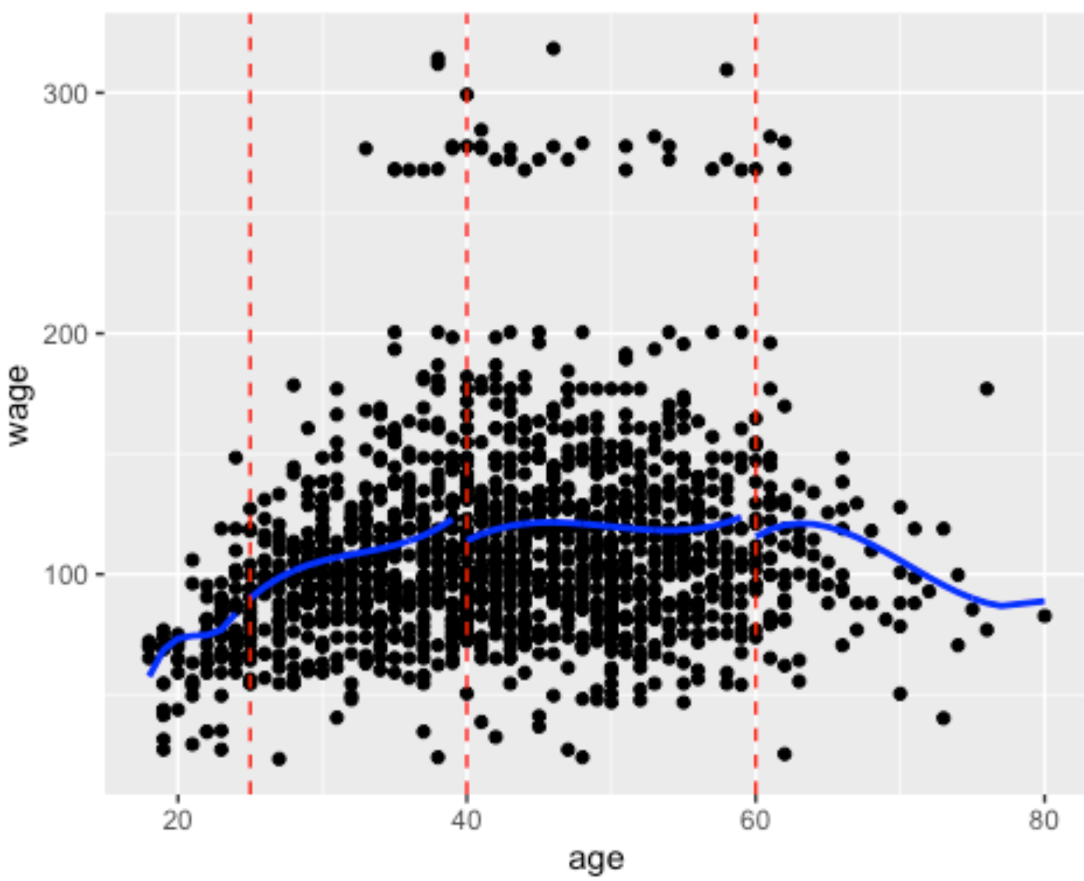

```
for (i in 1:length(test$age)){
  if (test$age[i]<25){
    test$pred_pq[i] <- lm.1$coefficients[1] +
lm.1$coefficients[2]*test$age[i] + lm.1$coefficients[3]*((test$age[i])^2)
  }else if (test$age[i]>=25 & test$age[i]<40){
    test$pred_pq[i] <- lm.2$coefficients[1] +
lm.2$coefficients[2]*test$age[i] + lm.2$coefficients[3]*((test$age[i])^2)
  }else if (test$age[i]>=40 & test$age[i]<60){
    test$pred_pq[i] <- lm.3$coefficients[1] +
lm.3$coefficients[2]*test$age[i] + lm.3$coefficients[3]*((test$age[i])^2)
  }else if (test$age[i]>=60){
    test$pred_pq[i] <- lm.4$coefficients[1] +
lm.4$coefficients[2]*test$age[i] + lm.4$coefficients[3]*((test$age[i])^2)
```

```
  }
}

ggplot()+
  geom_point(data=test, aes(x=age, y=wage), col="gray")+
  geom_point(data=test, aes(x=age, y=pred_pq), col="red")+
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```



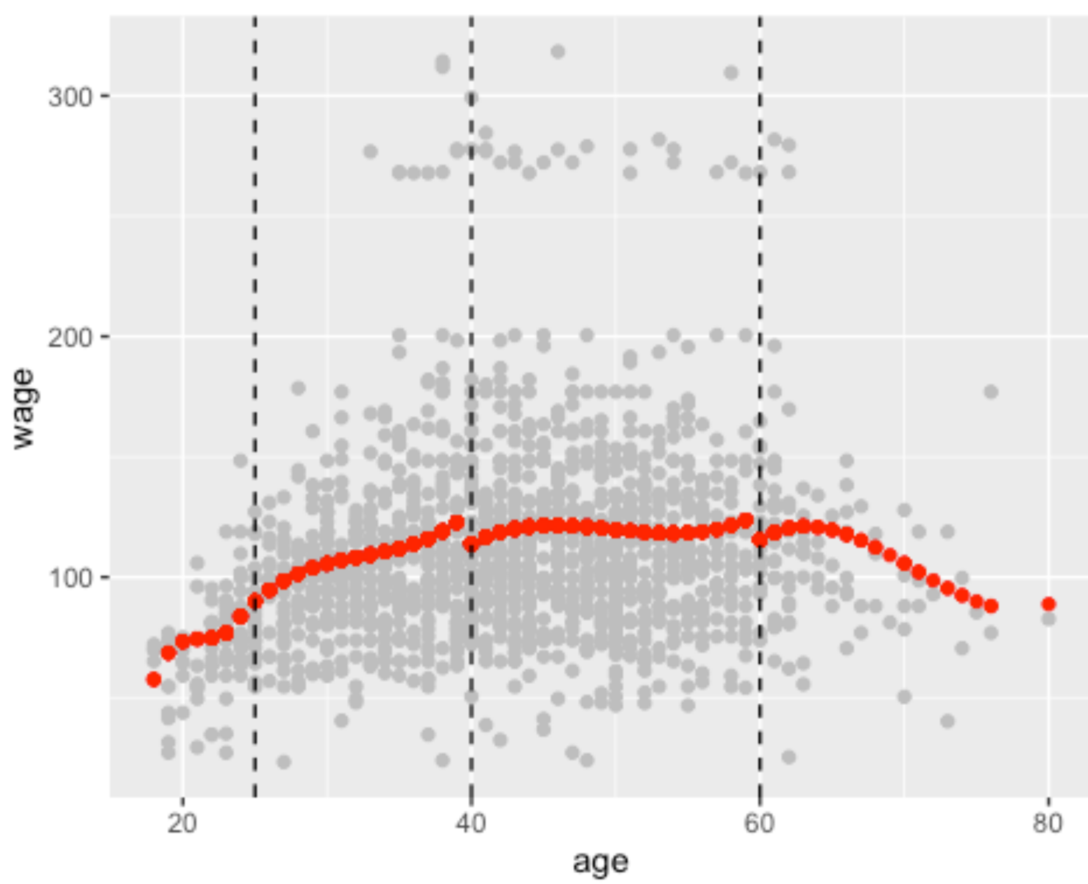

```
mean((test$pred_pq- test$wage)^2)

## [1] 1545.716

lm.1 <- lm(wage~poly(age,3, raw=TRUE), data = train[train$age<25,])
lm.2 <- lm(wage~poly(age,3, raw=TRUE), data = train[train$age >=25 &
train$age<40,])
lm.3 <- lm(wage~poly(age,3, raw=TRUE), data = train[train$age>=40 &
train$age<60,])
lm.4 <- lm(wage~poly(age,3, raw=TRUE), data = train[train$age>=60,])

pred1 <- predict(lm(wage~poly(age,3, raw=TRUE),
                    data = train[train$age<25,]))
pred2 <- predict(lm(wage~poly(age,3, raw=TRUE),
```

```
                    data = train[train$age >=25 & train$age<40,]))
pred3 <- predict(lm(wage~poly(age,3, raw=TRUE),
                    data = train[train$age>=40 & train$age<60,]))
pred4 <-  predict(lm(wage~poly(age,3, raw=TRUE),
                    data = train[train$age>=60,]))

ggplot() +
  geom_point(data = test, aes(x = age, y = wage)) +
  geom_line(data=train[train$age<25,],
            aes(y = pred1, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age >=25 & train$age<40,],
            aes(y = pred2, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age>=40 & train$age<60,],
            aes(y = pred3, x=age), size = 1, col="blue") +
  geom_line(data=train[train$age>=60,],
            aes(y = pred4, x=age), size = 1, col="blue") +
  geom_vline(xintercept = 25, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="red", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="red", size=0.5)
```




```
for (i in 1:length(test$age)){
  if (test$age[i]<25){
    test$pred_pc[i] <- lm.1$coefficients[1] +
```

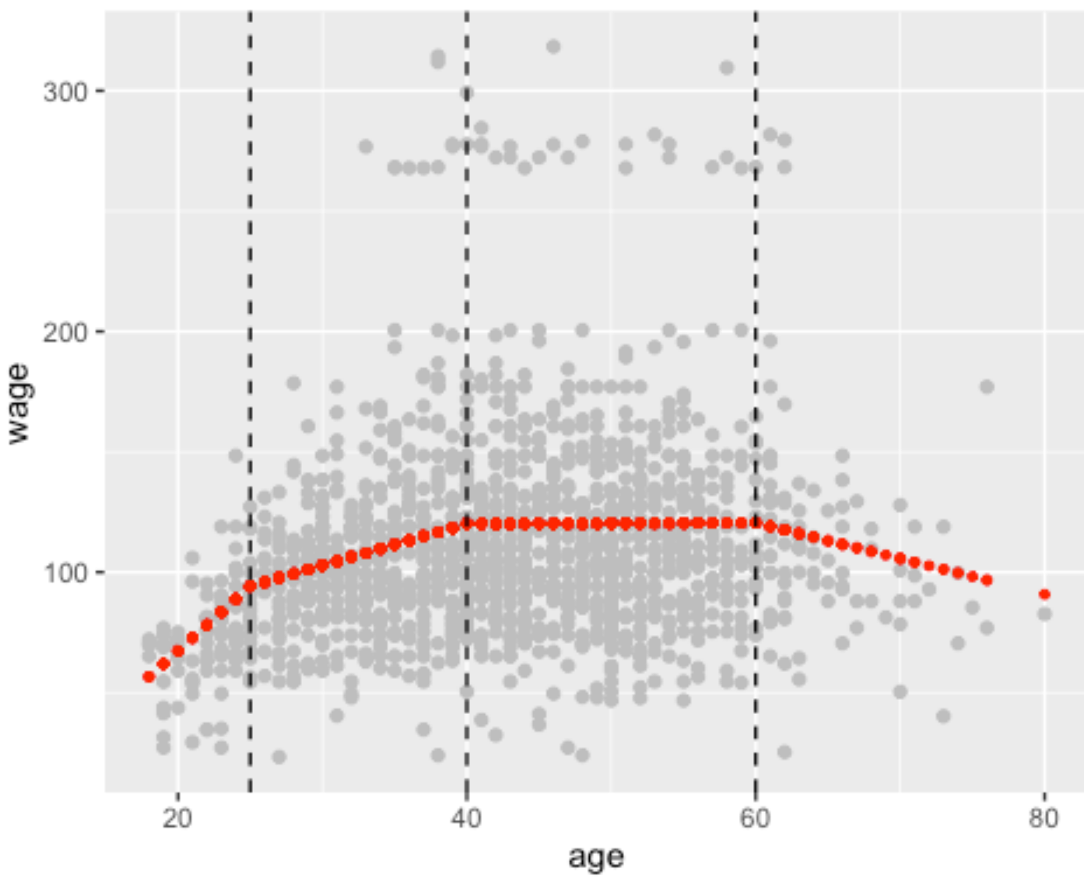
```
lm.1$coefficients[2]*test$age[i] + lm.1$coefficients[3]*((test$age[i])^2) +
lm.1$coefficients[4]*((test$age[i])^3)
  }else if (test$age[i]>=25 & test$age[i]<40){
    test$pred_pc[i] <- lm.2$coefficients[1] +
lm.2$coefficients[2]*test$age[i] + lm.2$coefficients[3]*((test$age[i])^2) +
lm.2$coefficients[4]*((test$age[i])^3)
  }else if (test$age[i]>=40 & test$age[i]<60){
    test$pred_pc[i] <- lm.3$coefficients[1] +
lm.3$coefficients[2]*test$age[i] + lm.3$coefficients[3]*((test$age[i])^2) +
lm.3$coefficients[4]*((test$age[i])^3)
  }else if (test$age[i]>=60){
    test$pred_pc[i] <- lm.4$coefficients[1] +
lm.4$coefficients[2]*test$age[i] + lm.4$coefficients[3]*((test$age[i])^2) +
lm.4$coefficients[4]*((test$age[i])^3)
  }
}

ggplot()+
  geom_point(data=test, aes(x=age, y=wage), col="gray")+
  geom_point(data=test, aes(x=age, y=pred_pc), col="red")+
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```



```
mean((test$pred_pc- test$wage)^2)

## [1] 1547.622
```

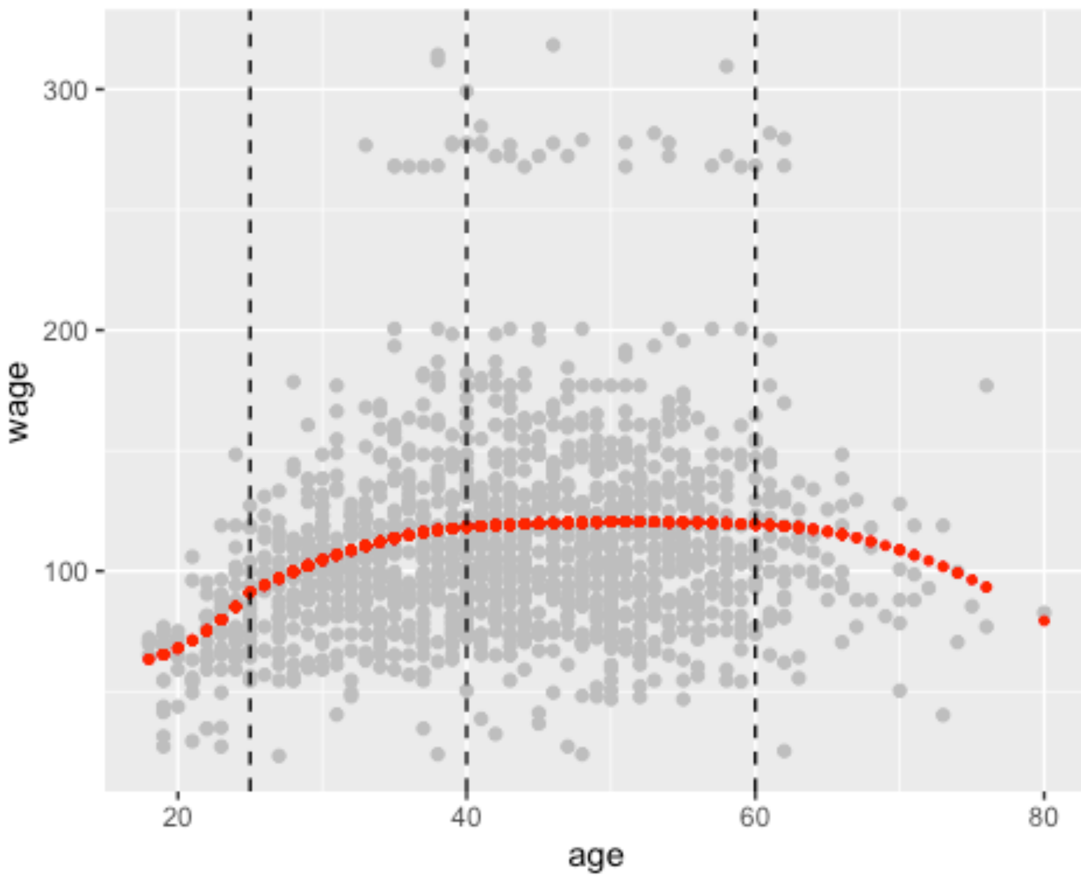
extra: Continuous piecewise

```
pred.lm <- lm(wage~ age + I((age-25)*(age>=25)) +
                                   I((age-40)*(age >= 40)) +
                                   I((age-60)*(age >= 60)),
                    data = train)

test$cpLR <- predict(pred.lm, test)

ggplot() +
  geom_point(data = test, aes(x = age, y = wage), col="gray") +
  geom_point(data=test, aes(y = cpLR , x=age), size = 1, col="red") +
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```



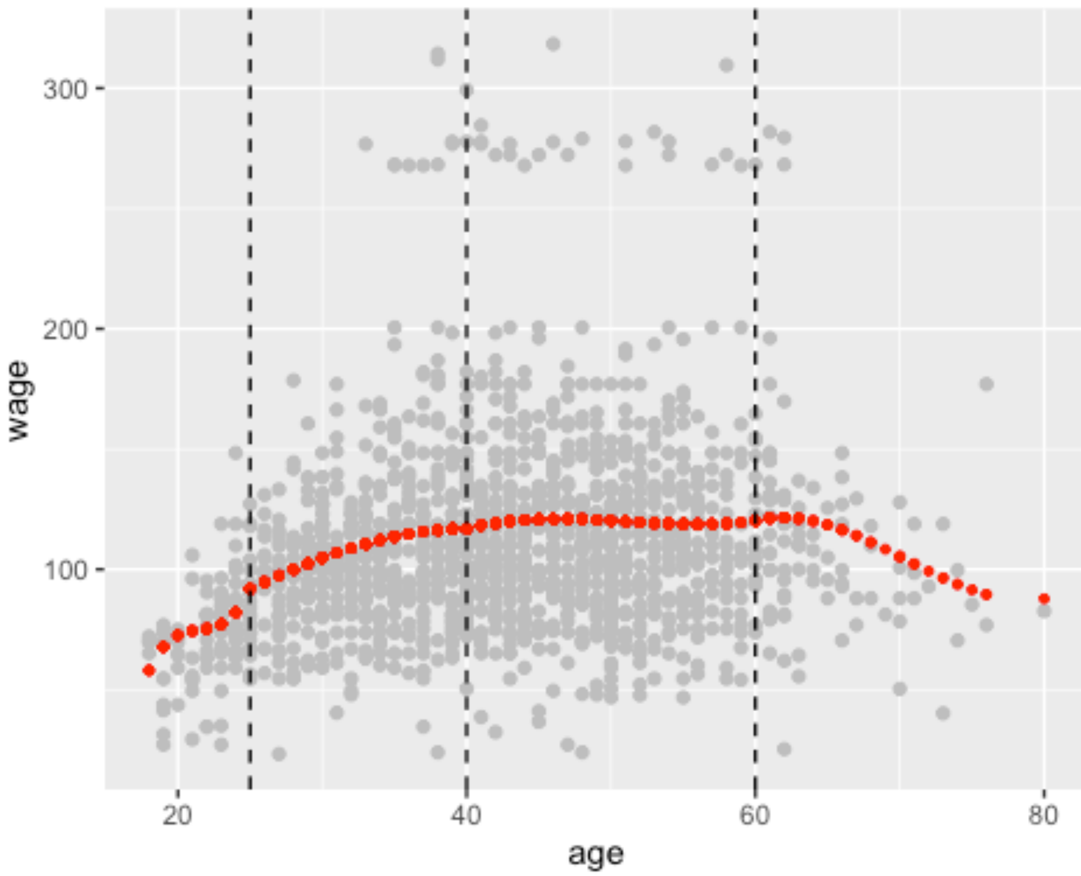

```
mean((test$cpLR - test$wage)^2)

## [1] 1543.944
```

```r
pred.quad <- lm(wage~ age + I(age^2) +
                    I((age-25)*(age>=25)) + I((age-25)^2*(age>=25)) +
                    I((age-40)*(age >= 40)) + I((age-40)^2*(age>=40)) +
                    I((age-60)*(age >= 60)) + I((age-60)^2*(age>=60)),
                    data = train)

test$cpq <- predict(pred.quad, test)

ggplot() +
  geom_point(data = test, aes(x = age, y = wage), col="gray") +
  geom_point(data=test,
            aes(y = cpq, x=age), size = 1, col="red") +
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```




```r
mean((test$cpq - test$wage)^2)

## [1] 1545.322

pred.cubic <- lm(wage~ age + I(age^2) + I(age^3) +
                    I((age-25)*(age>=25)) + I((age-25)^2*(age>=25)) +
                      I((age-25)^3*(age>=25)) +
                    I((age-40)*(age >= 40)) + I((age-40)^2*(age>=40)) +
```

```
                      I((age-40)^3*(age>=40)) +
                    I((age-60)*(age >= 60)) + I((age-60)^2*(age>=60)) +
                      I((age-60)^3*(age>=60)),
                    data = train)

test$cpc <- predict(pred.cubic, test)

ggplot() +
  geom_point(data = test, aes(x = age, y = wage), col="gray") +
  geom_point(data=test,
            aes(y = cpc, x=age), size = 1, col="red") +
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```

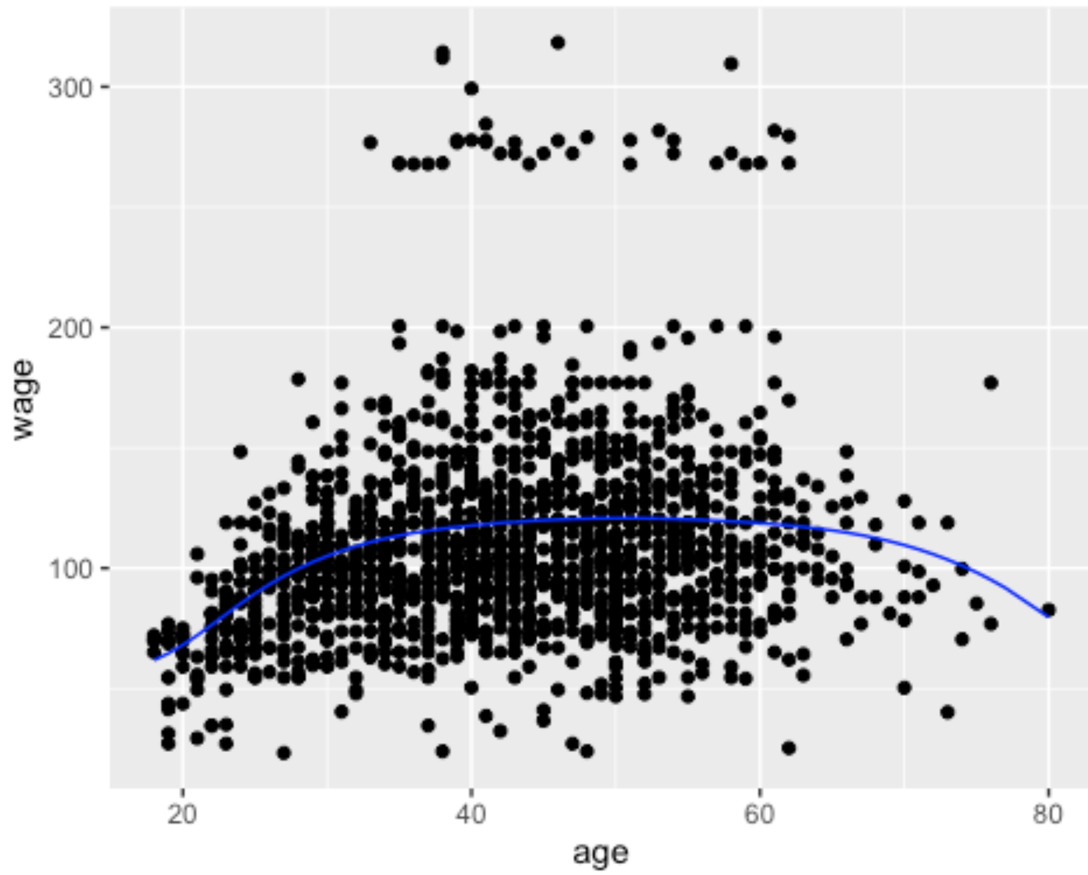



```
mean((test$cpc - test$wage)^2)

## [1] 1545.937
```

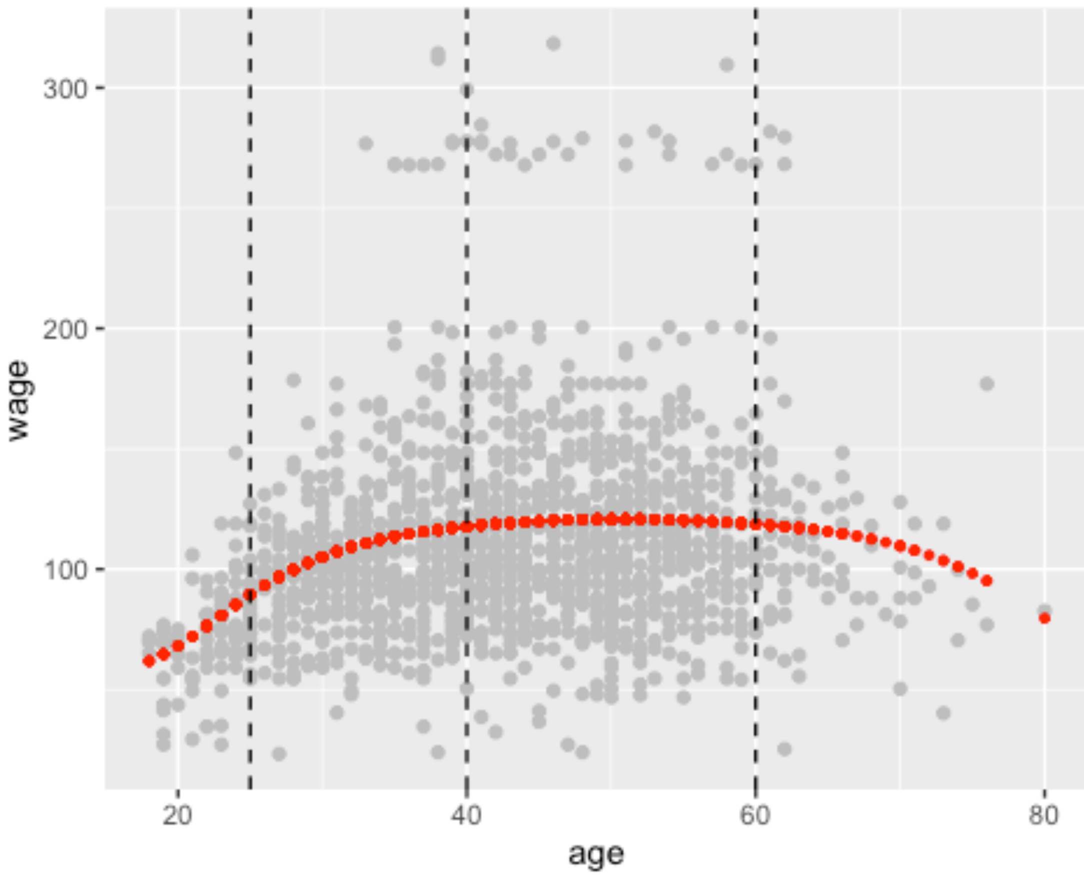
d. cubic spline

```
library(splines)

fit = lm(wage~bs(age, knots = c(25,40,60)), data = train)
```

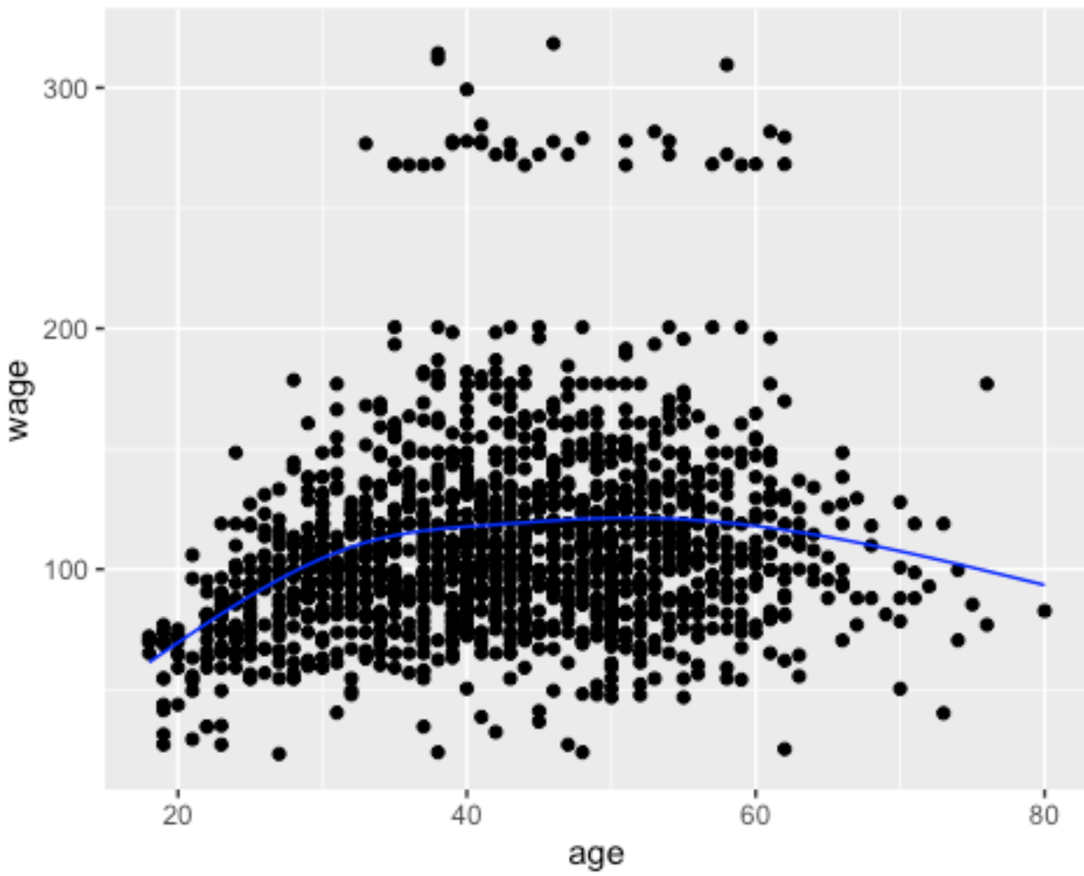
```
test$pred_lsp <- predict(fit, newdata=test)


ggplot() +
  geom_point(data = test, aes(x = age, y = wage)) +
  geom_line(data = train, aes(x = age, y = predict(fit,train)) , color =
"blue")
```



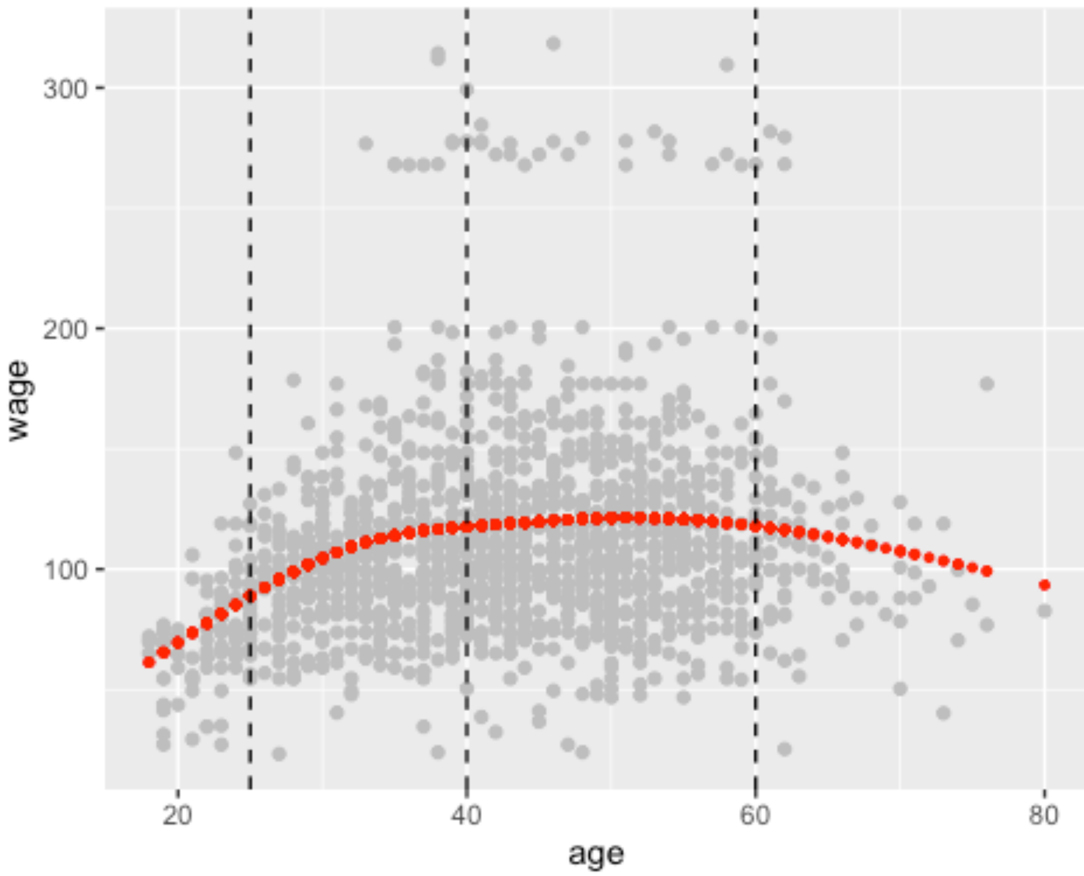

```
ggplot() +
  geom_point(data = test, aes(x = age, y = wage), col="gray") +
  geom_point(data = test, aes(y = pred_lsp, x=age), size = 1, col="red") +
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```




```
mean((test$pred_lsp- test$wage)^2)

## [1] 1547.277

fit2 = lm(wage~ns(age, df = 4), data = train)

test$pred_ns <- predict(fit2, newdata = test)

ggplot() +
  geom_point(data = test, aes(x = age, y = wage)) +
  geom_line(data = train, aes(x = age, y = predict(fit2,train)) , color =
"blue")
```



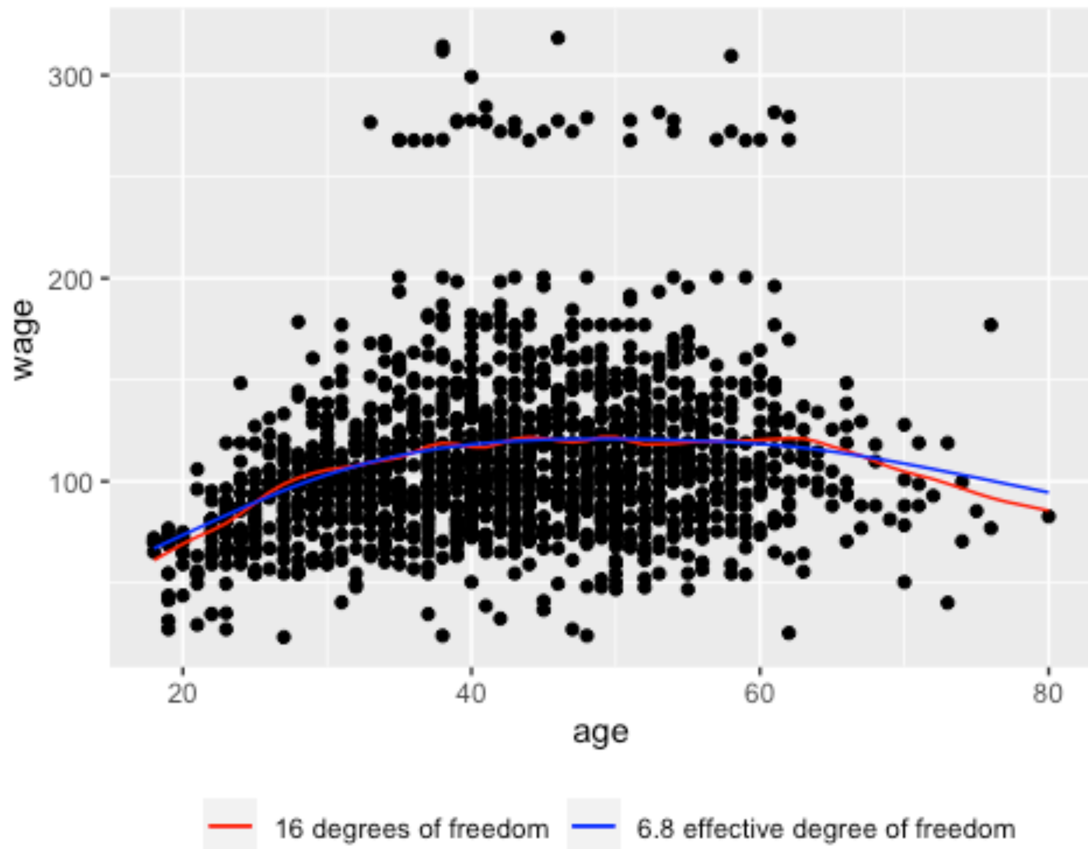

```
ggplot() +
  geom_point(data = test, aes(x = age, y = wage), col="gray") +
  geom_point(data = test, aes(y = pred_ns, x=age), size = 1, col="red")+
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```




```
mean((test$pred_ns- test$wage)^2)

## [1] 1548.026
```

e. Smoothing spline

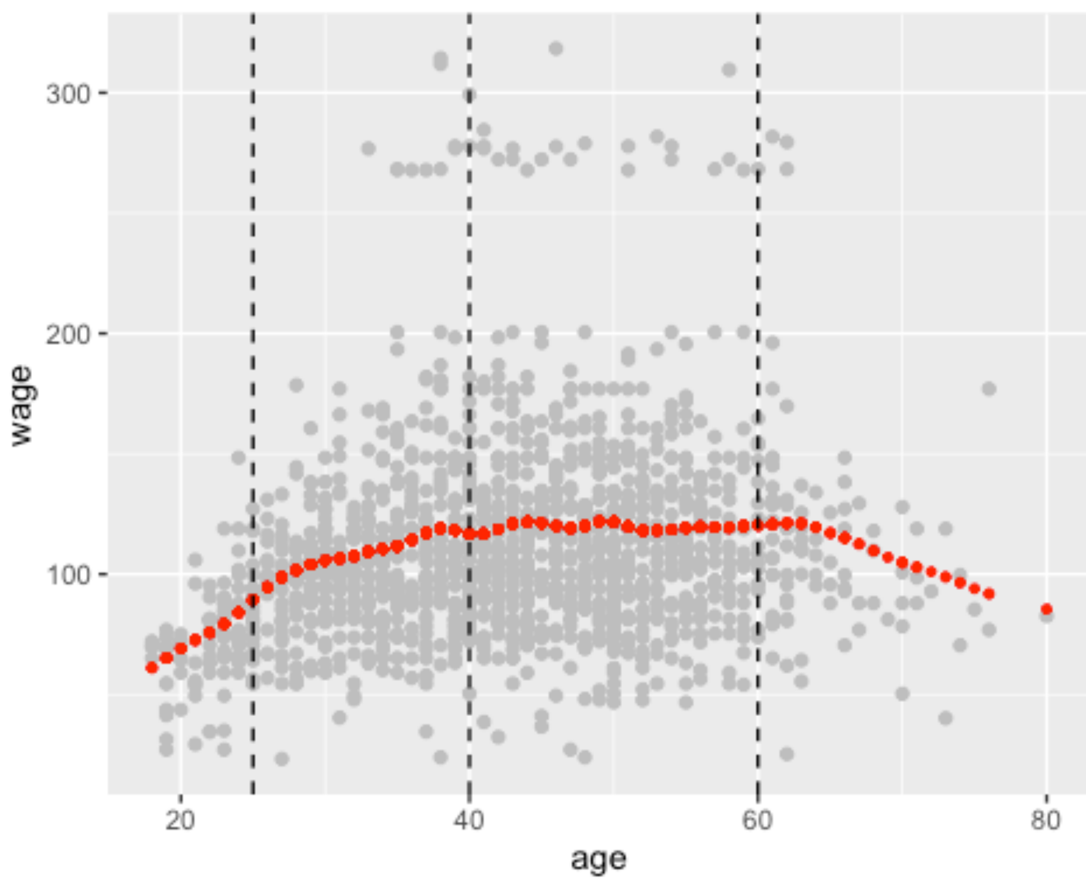
```
fit_smooth = with(train, smooth.spline(age, wage, df = 16))

fit_smooth_cv = with(train, smooth.spline(age, wage, cv = TRUE))

## Warning in smooth.spline(age, wage, cv = TRUE): cross-validation with non-
unique
## 'x' values seems doubtful

ggplot() +
  geom_point(data = test, aes(x = age, y = wage)) +
  geom_line(aes(x = fit_smooth$x, y = fit_smooth$y, col="16 degrees of
freedom"))  +
  geom_line(aes(x = fit_smooth_cv$x, y = fit_smooth_cv$y, col="6.8 effective
degree of freedom"))+
  scale_color_manual(values = c("red", "blue")) +
  theme(legend.position = 'bottom', legend.title = element_blank())
```




```
smooth <- as.data.frame(predict(fit_smooth, newdata=test))

test <-merge (test, smooth, by.x = 'age', by.y ='x', all.x=TRUE)

names(test)[names(test) == "y"] <- "pred_smc"
head(test)

##   age year           maritl     race       education              region
## 1  18 2004 1. Never Married 1. White      2. HS Grad 2. Middle Atlantic
## 2  18 2006 1. Never Married 1. White    1. < HS Grad 2. Middle Atlantic
## 3  18 2004 1. Never Married 1. White    1. < HS Grad 2. Middle Atlantic
## 4  18 2003 1. Never Married 2. Black      2. HS Grad 2. Middle Atlantic
## 5  19 2006 1. Never Married 1. White      2. HS Grad 2. Middle Atlantic
## 6  19 2007 1. Never Married 1. White 3. Some College 2. Middle Atlantic
##         jobclass         health health_ins  logwage     wage pred_step
pred_pLR
## 1 1. Industrial 2. >=Very Good     1. Yes 4.278754 72.15046  74.86721
62.65880
## 2 1. Industrial 2. >=Very Good      2. No 4.176091 65.11085  74.86721
62.65880
## 3 1. Industrial      1. <=Good     1. Yes 4.243038 69.61904  74.86721
62.65880
## 4 1. Industrial 2. >=Very Good     1. Yes 4.255273 70.47602  74.86721
```

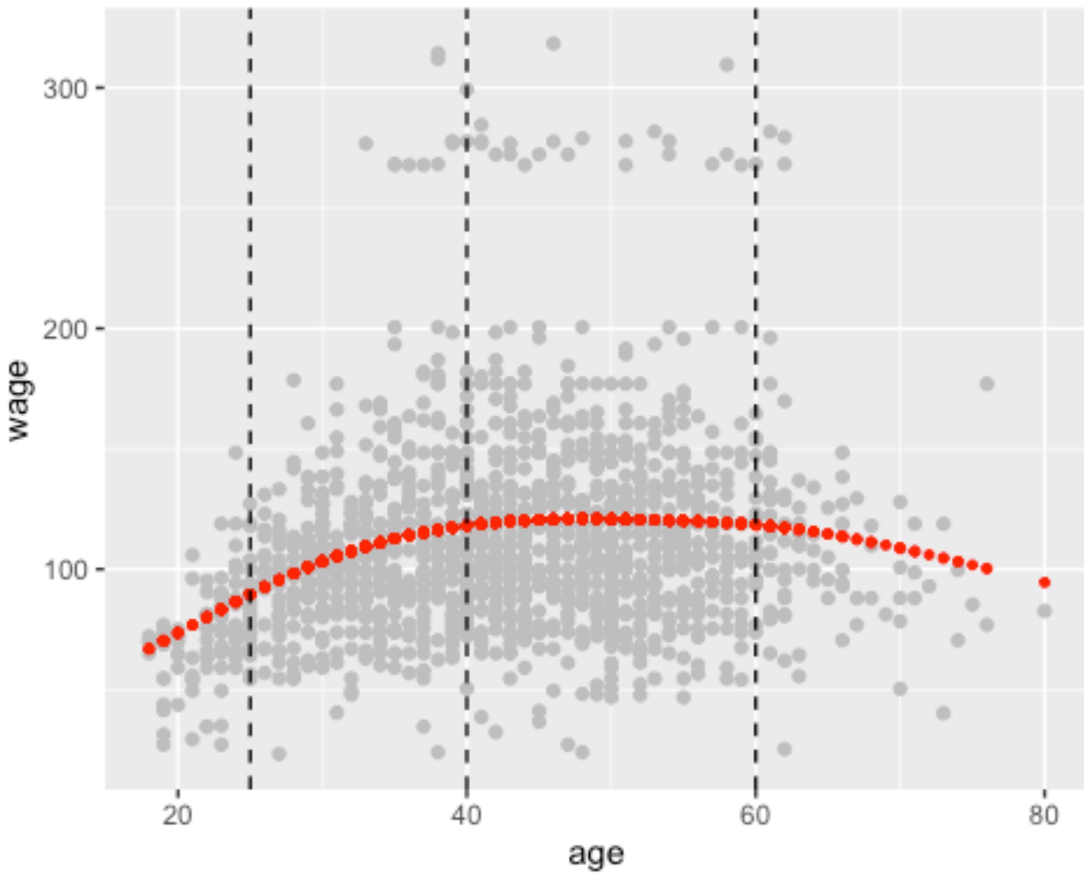
```
62.65880
## 5 1. Industrial 2. >=Very Good      2. No 4.342423 76.89360  74.86721
65.88096
## 6 1. Industrial      1. <=Good      2. No 3.724276 41.44121  74.86721
65.88096
##    pred_pq  pred_pc     cpLR      cpq      cpc pred_lsp  pred_ns pred_smc
## 1 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 2 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 3 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 4 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 5 65.59969 68.58183 61.96656 65.33433 67.81020 64.64774 65.44212 65.19237
## 6 65.59969 68.58183 61.96656 65.33433 67.81020 64.64774 65.44212 65.19237
```

```
ggplot() +
  geom_point(data = test, aes(x = age, y = wage), col="gray") +
  geom_point(data = test, aes(y = pred_smc, x=age), size = 1, col="red") +
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```




```
mean((test$pred_smc-test$wage)^2)

## [1] 1546.966
```

```
smooth_cv <- as.data.frame(predict(fit_smooth_cv, newdata=test))

test <-merge (test, smooth_cv, by.x = 'age', by.y ='x', all.x=TRUE)
names(test)[names(test) == "y"] <- "pred_smcv"
head(test)

##   age year            maritl     race       education             region
## 1  18 2004 1. Never Married 1. White      2. HS Grad 2. Middle Atlantic
## 2  18 2006 1. Never Married 1. White    1. < HS Grad 2. Middle Atlantic
## 3  18 2004 1. Never Married 1. White    1. < HS Grad 2. Middle Atlantic
## 4  18 2003 1. Never Married 2. Black      2. HS Grad 2. Middle Atlantic
## 5  19 2006 1. Never Married 1. White      2. HS Grad 2. Middle Atlantic
## 6  19 2007 1. Never Married 1. White 3. Some College 2. Middle Atlantic
##         jobclass         health health_ins  logwage     wage pred_step
pred_pLR
## 1 1. Industrial 2. >=Very Good     1. Yes 4.278754 72.15046  74.86721
62.65880
## 2 1. Industrial 2. >=Very Good      2. No 4.176091 65.11085  74.86721
62.65880
## 3 1. Industrial      1. <=Good     1. Yes 4.243038 69.61904  74.86721
62.65880
## 4 1. Industrial 2. >=Very Good     1. Yes 4.255273 70.47602  74.86721
62.65880
## 5 1. Industrial 2. >=Very Good      2. No 4.342423 76.89360  74.86721
65.88096
## 6 1. Industrial      1. <=Good      2. No 3.724276 41.44121  74.86721
65.88096
##    pred_pq  pred_pc     cpLR      cpq      cpc pred_lsp  pred_ns pred_smc
## 1 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 2 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 3 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 4 61.15119 57.53980 56.59779 63.36969 58.05987 61.83903 61.33527 61.18052
## 5 65.59969 68.58183 61.96656 65.33433 67.81020 64.64774 65.44212 65.19237
## 6 65.59969 68.58183 61.96656 65.33433 67.81020 64.64774 65.44212 65.19237
##   pred_smcv
## 1  66.96963
## 2  66.96963
## 3  66.96963
## 4  66.96963
## 5  70.25202
## 6  70.25202

ggplot() +
  geom_point(data = test, aes(x = age, y = wage), col="gray") +
  geom_point(data = test, aes(y = pred_smcv, x=age), size = 1, col="red") +
  geom_vline(xintercept = 25, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 40, linetype="dashed", color="black", size=0.5) +
  geom_vline(xintercept = 60, linetype="dashed", color="black", size=0.5)
```




```
mean((test$pred_smcv-test$wage)^2)

## [1] 1547.498
```

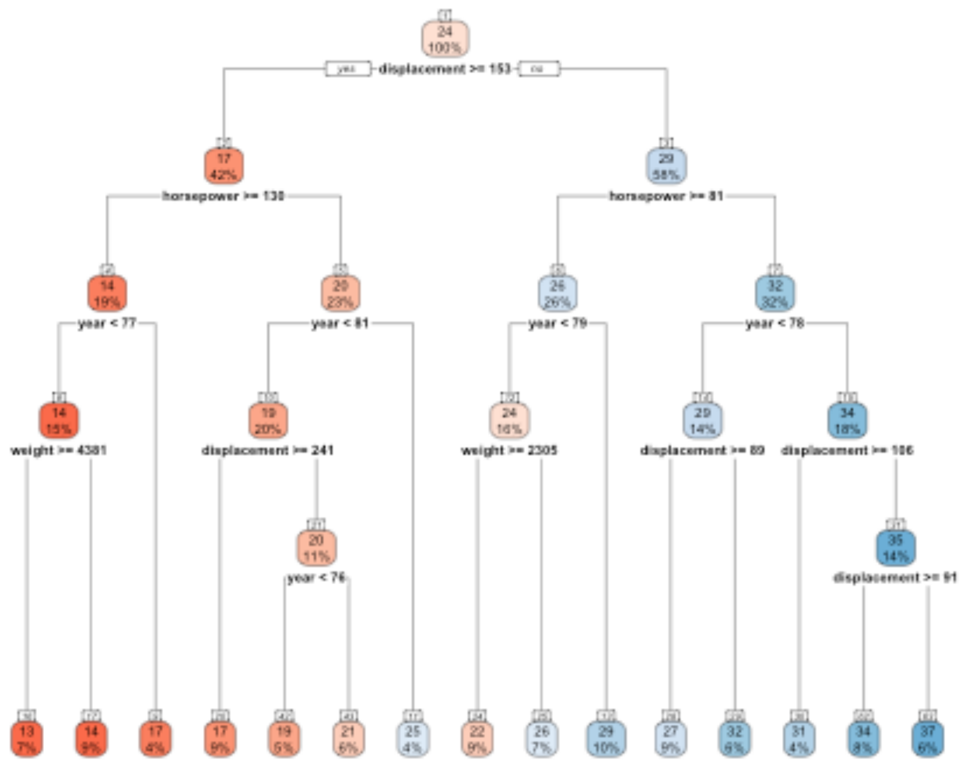
For this question, I randomly selected knots at 25, 40, 60 and fit the models on these knots. The number of knots could be further changed but I haven't done that here. For this question that was not required. What we are trying to see here is that for which approach yields the best results for our dataset. What I did here is that I first split the dataset into train and test set. Then, I fit all the required models for the question and some extra ones to compare all these approaches. What I observed from this particular question is that all the approaches have almost similar test error. This might be because of our datapoints are spread in such a way, that what all these approaches try to do is stay around the center so as to reduce the mean squared error. And looking at the graphs, we see a very strong similar predicted values in all these approaches. So, for this data, all of these approaches behave very similarly.

5. Use the Auto data set to predict a car's mpg. (You should remove the name variable before you begin.)

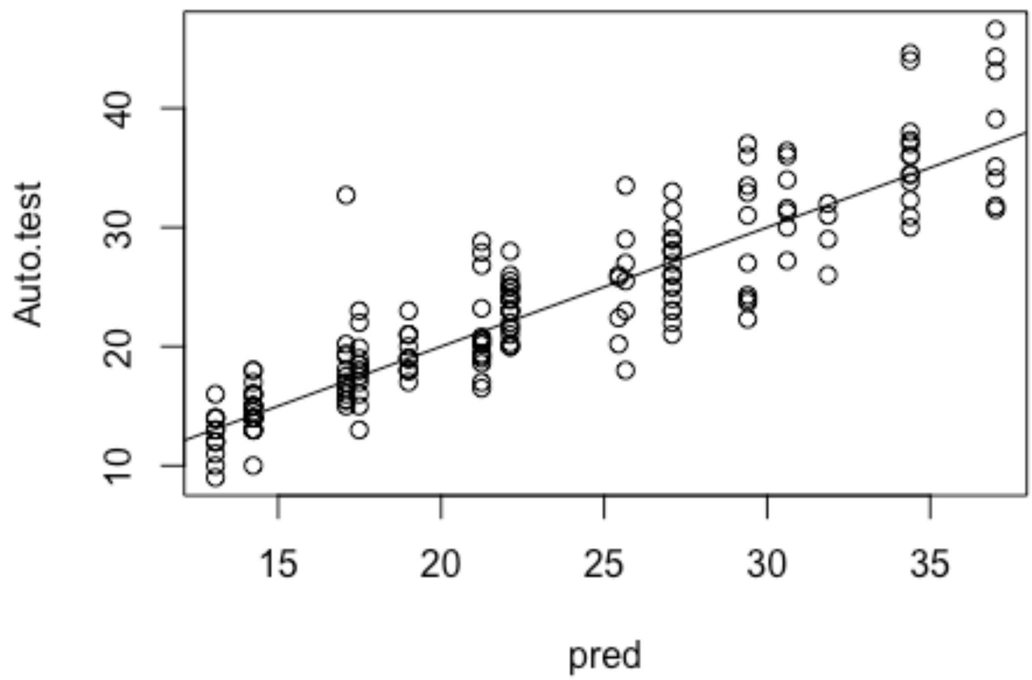
```
library(ISLR2)


Auto <- subset(Auto, select = - c(name))
```

a. First, try using a regression tree. You should grow a big tree, and then consider pruning the tree. How accurately does your regression tree predict a car's gas mileage? Make some figures, and comment on your results.

```
set.seed(1023)
train <- sample(1:nrow(Auto), nrow(Auto)/2)

library(rpart)
Auto.rpart <- rpart(mpg ~ . , data=Auto, subset=train,
control=rpart.control(cp=0))

library(rpart.plot)
rpart.plot(Auto.rpart, box.palette = "RdBu", nn=TRUE)
```



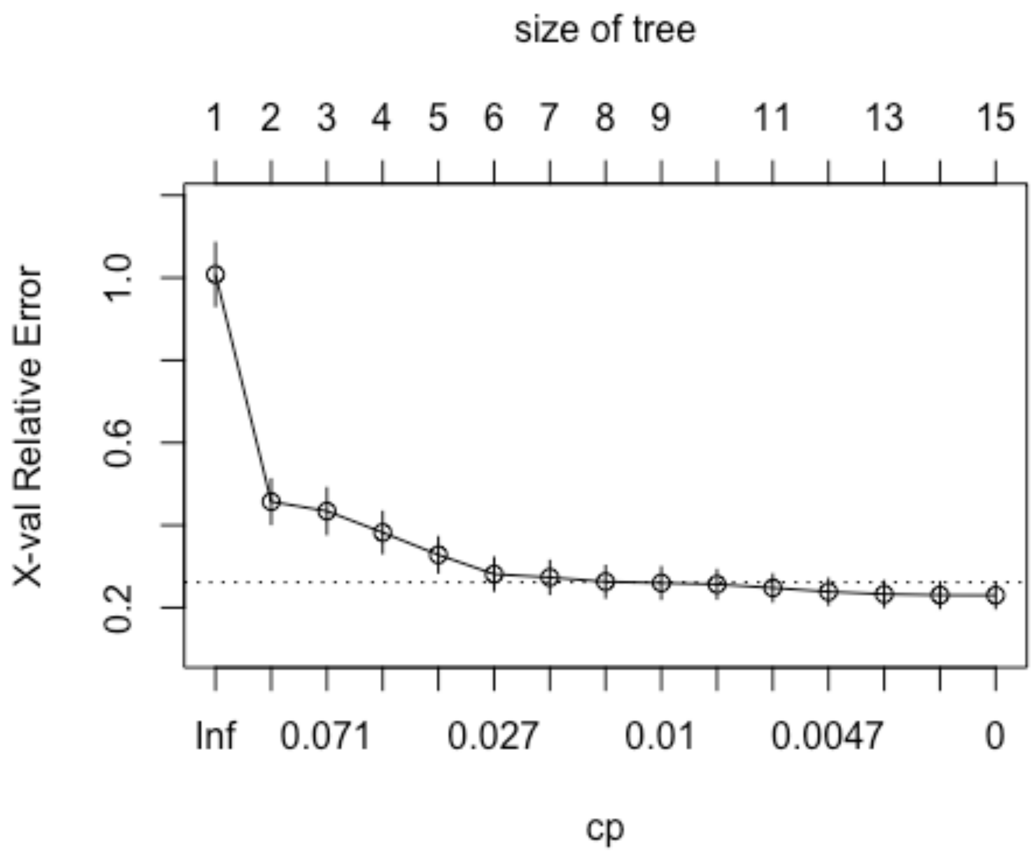

```
pred <- predict(Auto.rpart, newdata=Auto[-train,])
Auto.test <- Auto[-train, "mpg"]
plot(pred, Auto.test)
abline(0,1)
```



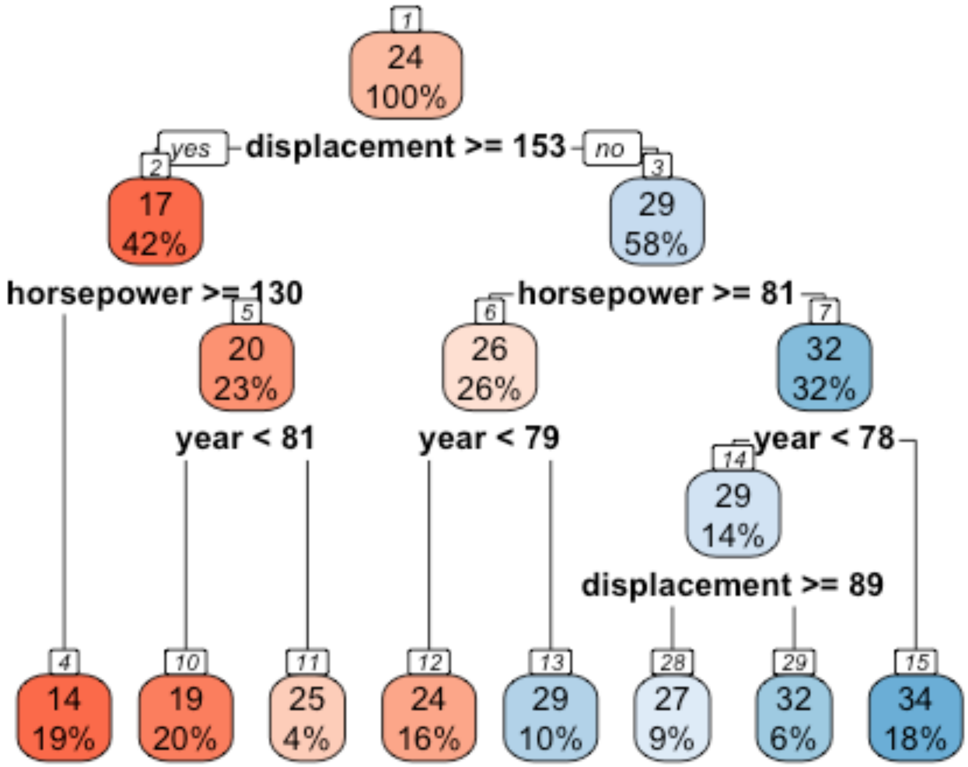

```
mean((pred-Auto.test)^2)

## [1] 11.01688

printcp(Auto.rpart)

##
## Regression tree:
## rpart(formula = mpg ~ ., data = Auto, subset = train, control =
rpart.control(cp = 0))
##
## Variables actually used in tree construction:
## [1] displacement horsepower   weight       year
##
## Root node error: 11404/196 = 58.183
##
## n= 196
##
##             CP nsplit rel error  xerror     xstd
## 1  0.58335211      0   1.00000 1.00814 0.076709
## 2  0.09034839      1   0.41665 0.45711 0.054381
## 3  0.05651788      2   0.32630 0.43413 0.055889
## 4  0.03954453      3   0.26978 0.38224 0.050339
## 5  0.03337170      4   0.23024 0.32810 0.043193
```

```
## 6  0.02165424      5   0.19687 0.28122 0.040447
## 7  0.01335512      6   0.17521 0.27333 0.040337
## 8  0.01269315      7   0.16186 0.26288 0.038530
## 9  0.00843418      8   0.14916 0.25984 0.038627
## 10 0.00662556      9   0.14073 0.25661 0.034965
## 11 0.00559331     10   0.13410 0.24786 0.032999
## 12 0.00392847     11   0.12851 0.23859 0.031736
## 13 0.00226616     12   0.12458 0.23272 0.031425
## 14 0.00086678     13   0.12232 0.23040 0.031667
## 15 0.00000000     14   0.12145 0.22995 0.031624
```

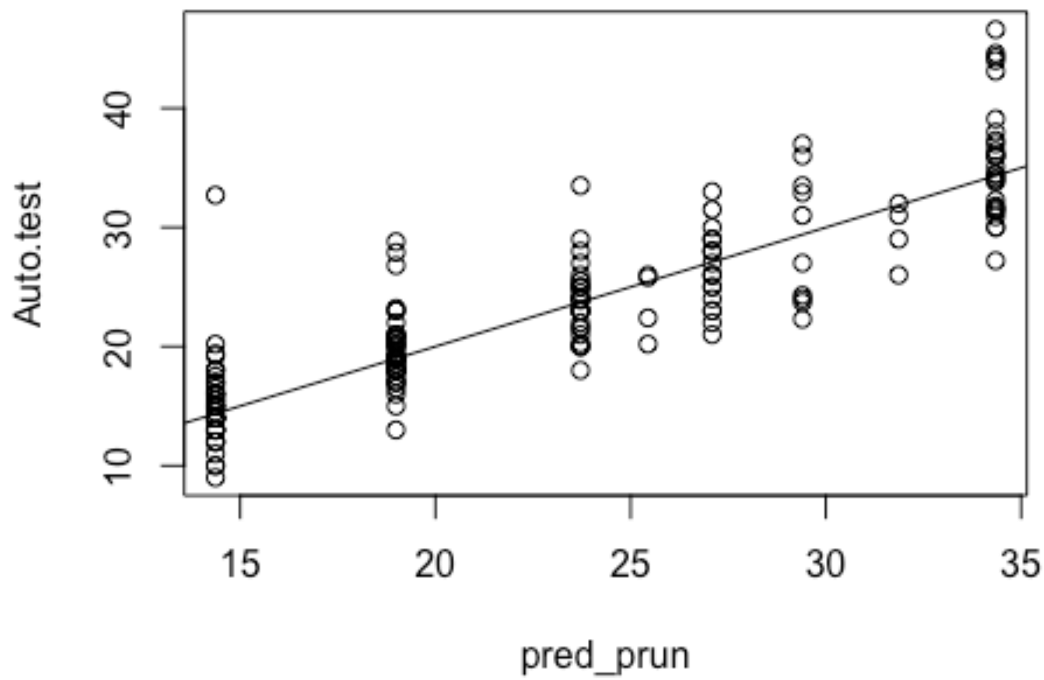
```r
plotcp(Auto.rpart)
```




```r
Auto.rpart.pruned <- prune(Auto.rpart, cp=0.01335512)
rpart.plot(Auto.rpart.pruned, box.palette = "RdBu", nn=TRUE)
```




```
pred_prun <- predict(Auto.rpart.pruned, newdata=Auto[-train,])
Auto.test <- Auto[-train, "mpg"]
plot(pred_prun, Auto.test)
abline(0,1)
```




```
mean((pred_prun-Auto.test)^2)

## [1] 13.25431
```

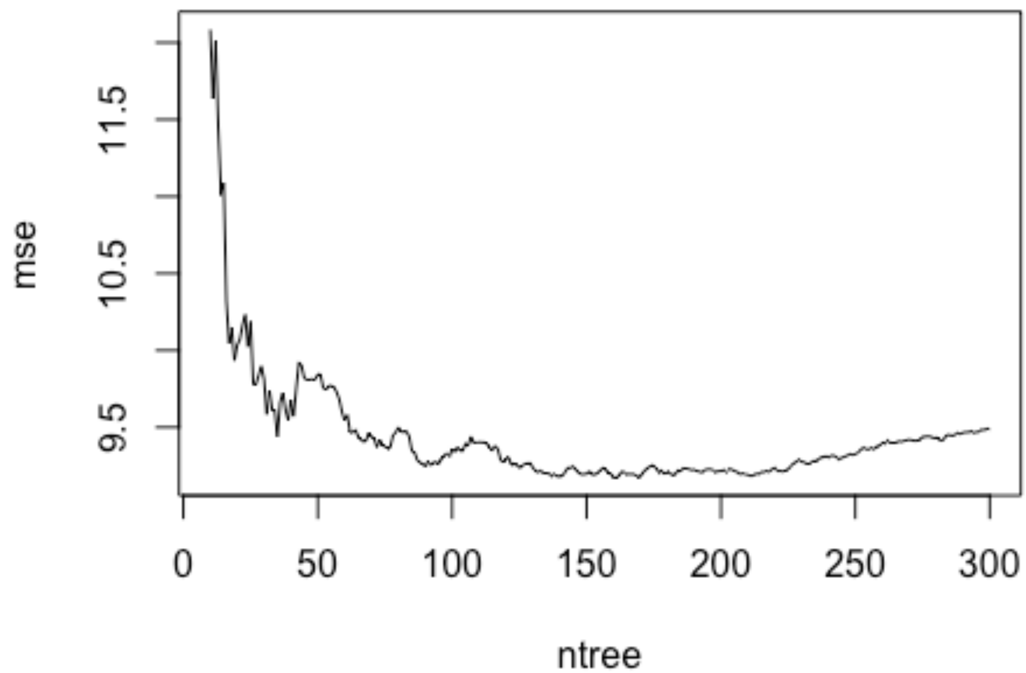
Initially, I “grew” the decision tree fully. Then, for pruning I considered “cp” and factored that in growing a smaller tree. The pruned tree has slightly higher error than a fully-grown tree.

b. Fit a bagged regression tree model to predict a car’s mpg. How accurately does this model predict gas mileage? What tuning parameter value(s) did you use in fitting this model?

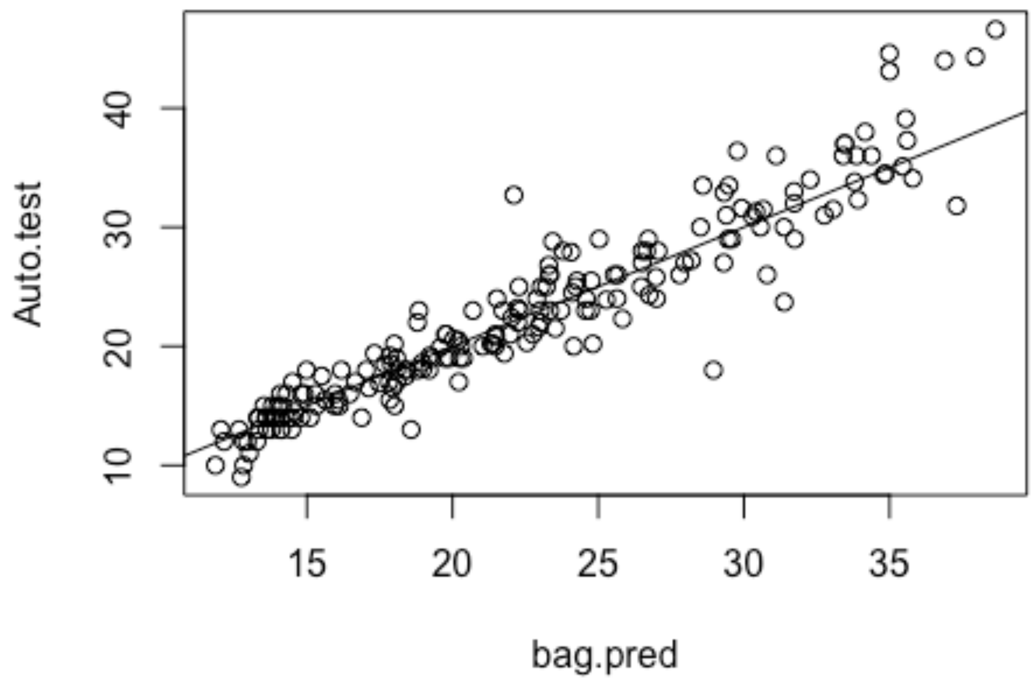
```
library(caret)

## Loading required package: lattice

set.seed(1729)
cntrl <- trainControl(method = "cv", number = 10)
bagg.Auto <- train(mpg ~ ., data = Auto, subset=train, method = "treebag",
        trControl = cntrl)

summary(bagg.Auto)

##              Length Class      Mode
## y            196    -none-     numeric
## X              0    -none-     NULL
```

```
## mtrees       25    -none-     list
## OOB           1    -none-     logical
## comb          1    -none-     logical
## xNames        7    -none-     character
## problemType   1    -none-     character
## tuneValue     1    data.frame list
## obsLevels     1    -none-     logical
## param         0    -none-     list
```

```
library(ipred)

ntree <-10:300
mse <- vector(mode="numeric", length=length(ntree))


for (i in seq_along(ntree)){
  set.seed(123)
  model <- bagging(formula=mpg~.,
                   data=Auto[train,],
                   coob=TRUE,
                   nbagg=ntree[i],
                   control = rpart.control(minsplit = 2, cp = 0))
  mse[i] <- ((model$err)^2)

}

plot(ntree, mse, type='l')
```




```
bag.Auto <- bagging(
  formula = mpg ~ .,
  data = Auto[train, ],
  nbagg = 150,
  coob = TRUE,
  control = rpart.control(minsplit = 2, cp = 0)
)

bag.pred <- predict(bag.Auto, newdata=Auto[-train,])
plot(bag.pred, Auto.test)
abline(0,1)
```




```
mean((bag.pred- Auto.test)^2)

## [1] 6.919
```

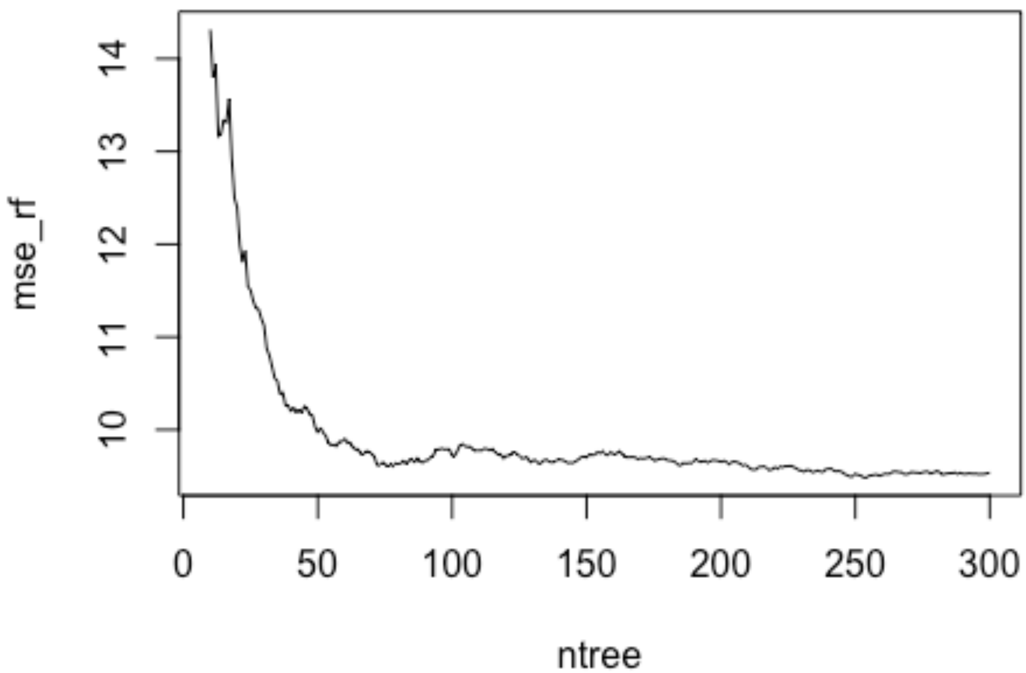
Here, instead of just growing one decision tree, I used the bagging approach to create multiple decision trees and then use all those trees aggregately to predict. This is a kind of ensemble approach to deal with this data.

For bagging, we use all the features. So, we don't need to tune the number of features. However, we can tune the number of trees. Number of trees is our hyper-parameter. I tried to do this by bagging for 10 trees to 300 trees and then calculated the error for the OOB. I plotted this error with the number of trees and I observed that initially for the increase in number of trees, there is a decrease in the OOB error, however, after a point, the increase in the number of trees does not significantly affect the OOB error. So, I randomly selected 15 trees for the final bagging model and predicted the mpg values for the test data, and calculated the test error. Here, the test error is much better as compared to a single-decision tree.

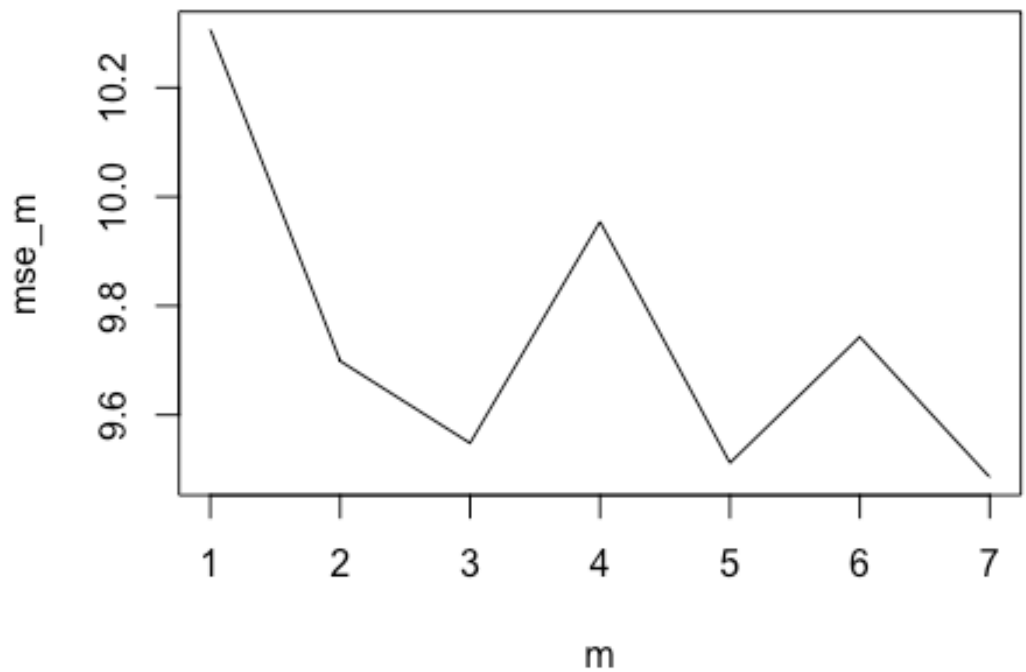
We verify the fact that we learnt in the class that Bagging outperforms single decision tree because of lower-variance in Bagging.

c. Fit a random forest model to predict a car's mpg. How accurately does this model predict gas mileage? What tuning parameter value(s) did you use in fitting this model?

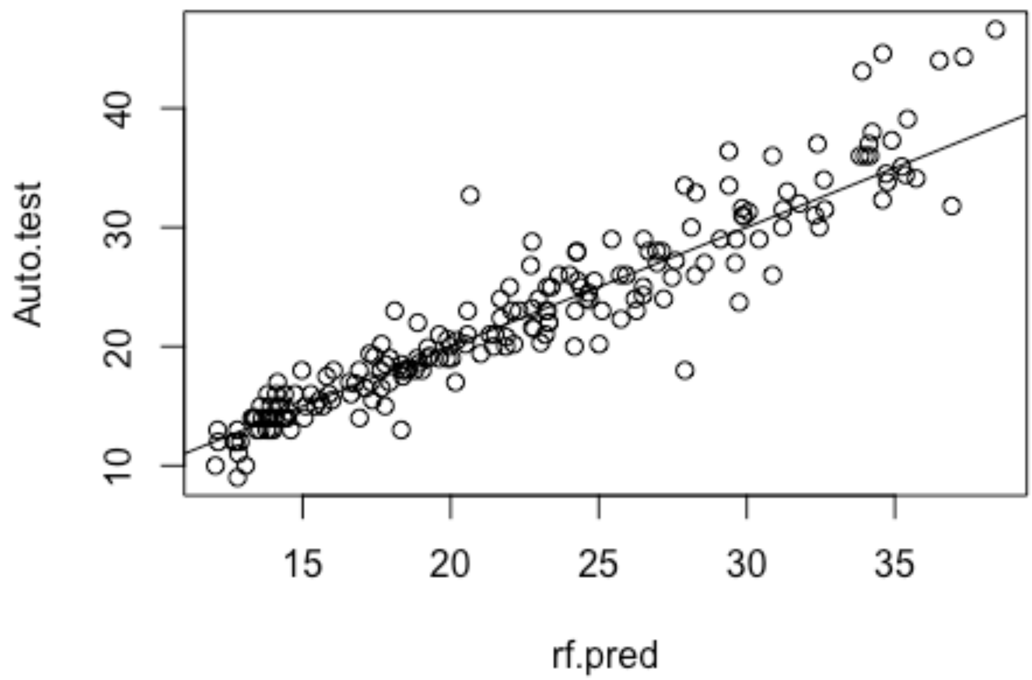
```
library(randomForest)

## randomForest 4.7-1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

ntree <-10:300
mse_rf <- vector(mode="numeric", length=length(ntree))
for (i in seq_along(ntree)){
  set.seed(123)
  model <- randomForest(mpg~.,
                    data=Auto[train,],
                    ntree=ntree[i])
  mse_rf[i] <- (tail (model$mse, 1))

}

plot(ntree, mse_rf, type='l')
```



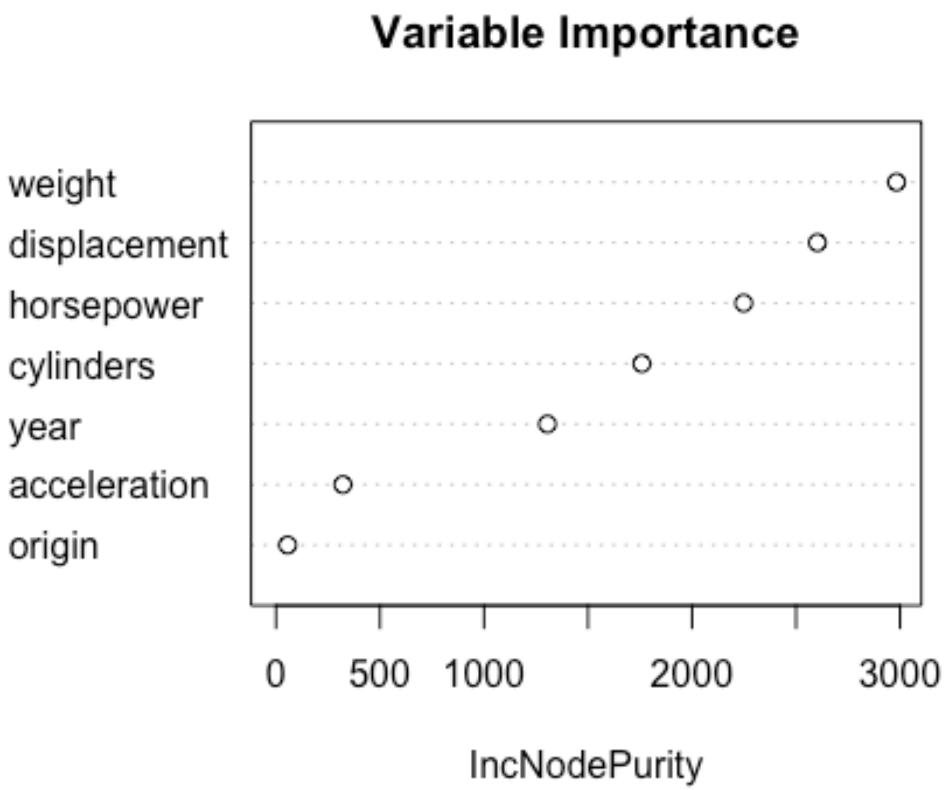

```
m <-1:7
mse_m <- vector(mode="numeric", length=length(m))
for (i in seq_along(m)){
  set.seed(123)
  model <- randomForest(mpg~.,
                    data=Auto,
                    subset = train,
                    ntree=150,
                    mtry=m[i])
  mse_m[i] <- (tail (model$mse, 1))

}

plot(m, mse_m, type='l')
```




```
library(randomForest)
rf.Auto <- randomForest(mpg~.,
                   data = Auto[train,],
                   ntree = 150,
                   mtry = 5)

rf.pred <- predict(rf.Auto, newdata=Auto[-train,])
plot(rf.pred, Auto.test)
abline(0,1)
```




```
mean((rf.pred- Auto.test)^2)

## [1] 7.330786

varImpPlot(rf.Auto,
           sort = T,
           n.var = 7,
           main = "Variable Importance")
```



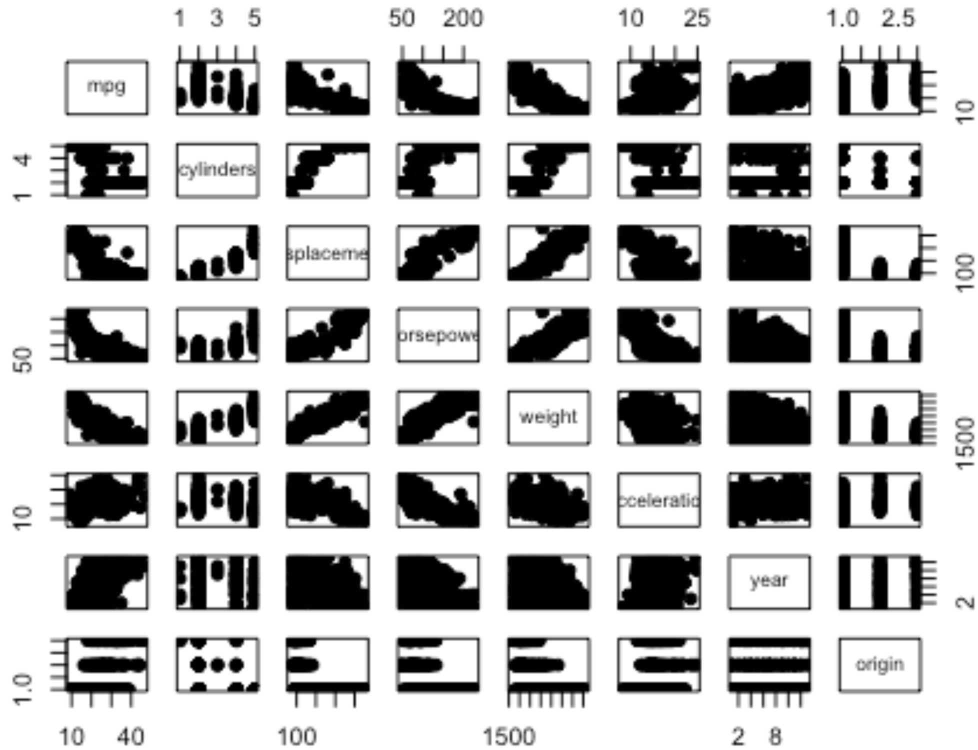

```
importance(rf.Auto)

##              IncNodePurity
## cylinders       1758.92654
## displacement    2602.78097
## horsepower      2248.26974
## weight          2983.23413
## acceleration     321.58805
## year            1304.97516
## origin            56.23384
```

For randomforest, we can tune the number of features as well as the number of trees. I did the same thing as in the Bagging, I calculated the OOB error for all RandomForest with 10 trees to 300 trees and arrived at a similar result that after a certain point, the increase in number of trees does not affect the OOB error significantly.

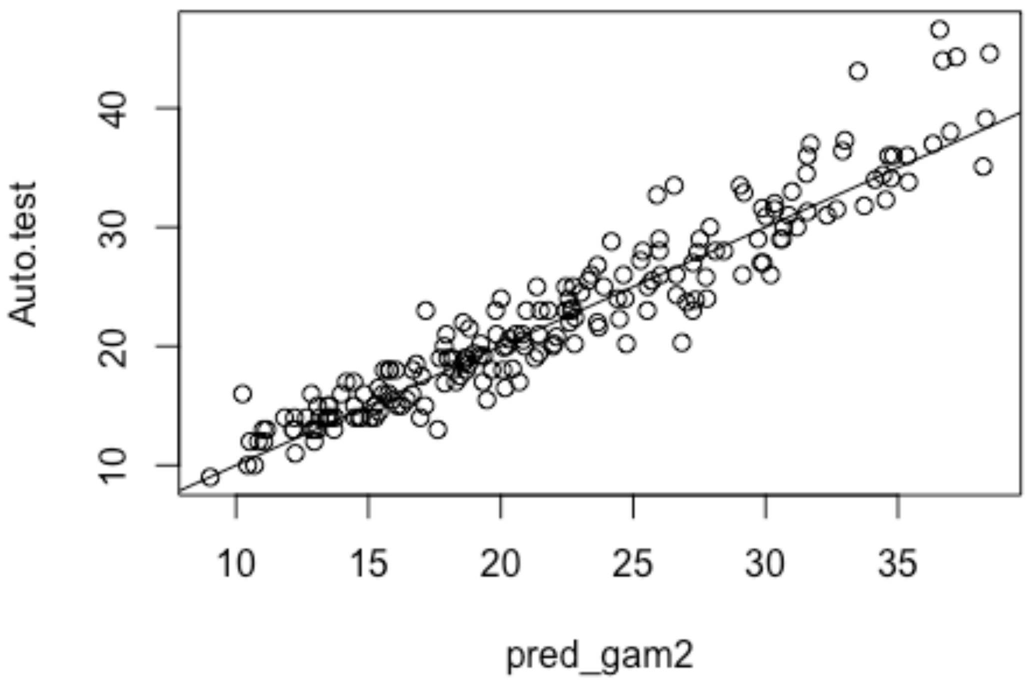
So, I chose 150 trees at random and tried to see how many features do we need? I checked out randomforest with 150 trees for 1 to 7 features. I selected 5 features and predicted the mpg for test data and calculated the test error. The test error as compared to the single decsion tree is much better for Random Forest.

d. Fit a generalized additive model (GAM) model to predict a car's mpg. How accurately does your GAM model predict a car's gas mileage? Make some figures to help visualize the fitted functions in your GAM model, and comment on your results.

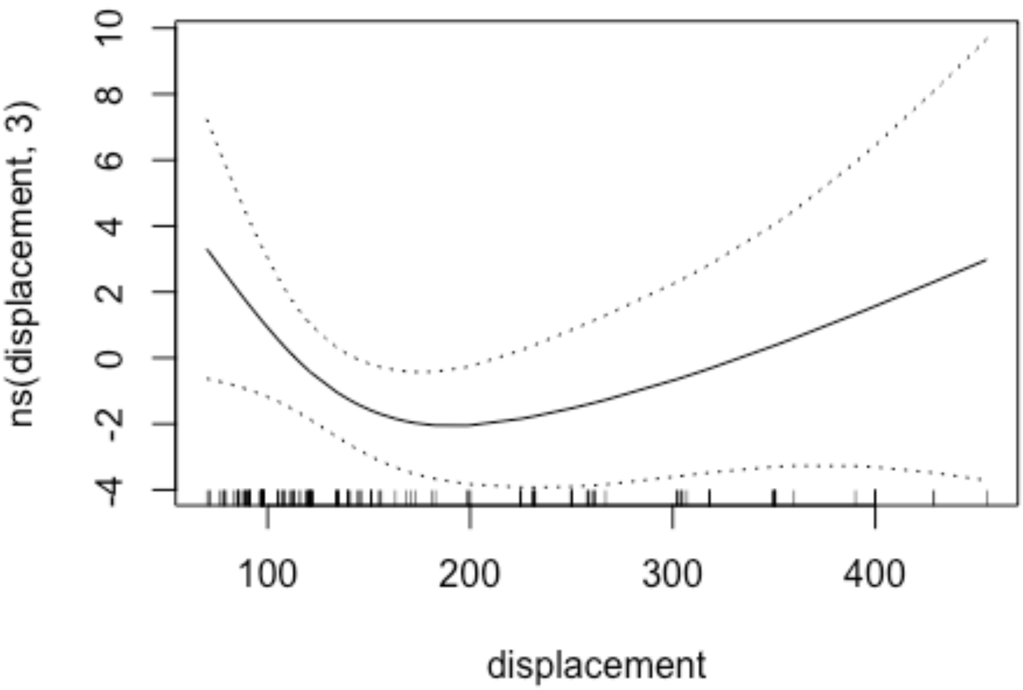
```
head(Auto)

##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1

Auto$cylinders <- as.factor(Auto$cylinders)
Auto$year <- as.factor(Auto$year)
Auto$origin <- as.factor(Auto$origin)

head(Auto)

##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1

pairs(Auto[,], pch = 19)
```



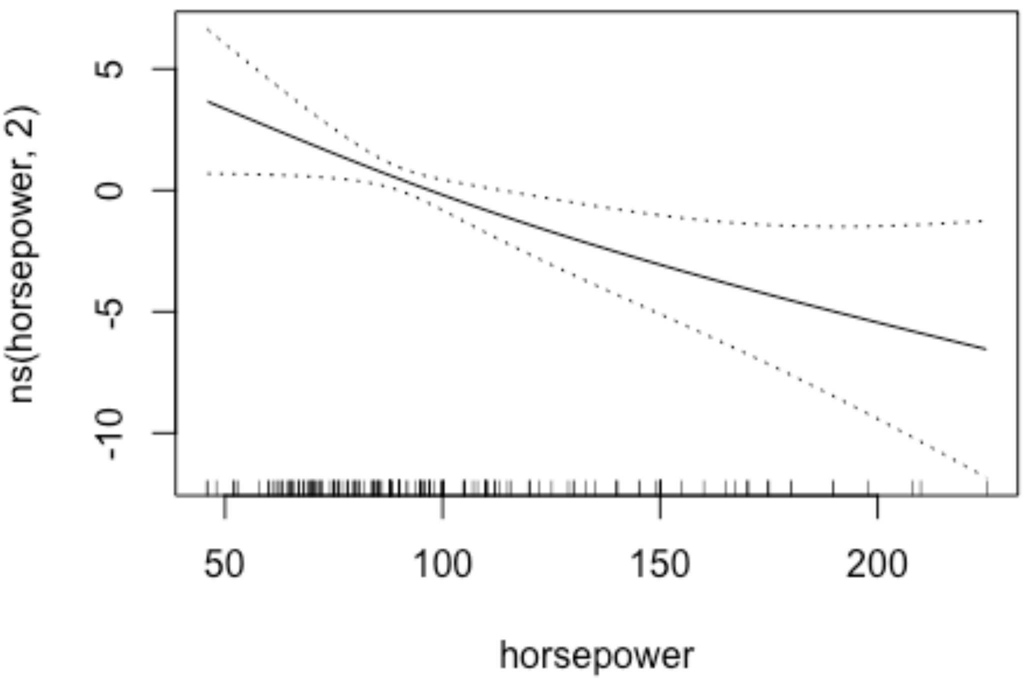

```
library(gam)

## Loading required package: foreach

## Loaded gam 1.20.1

gam2 <- gam(mpg ~ ns(displacement, 3) + ns(horsepower, 2)  + ns(weight, 2) +
ns(acceleration, 3) + year + origin + cylinders, data = Auto, subset = train)
pred_gam2 <- predict(gam2, newdata=Auto[-train,])
plot(pred_gam2, Auto.test)
abline(0,1)
```



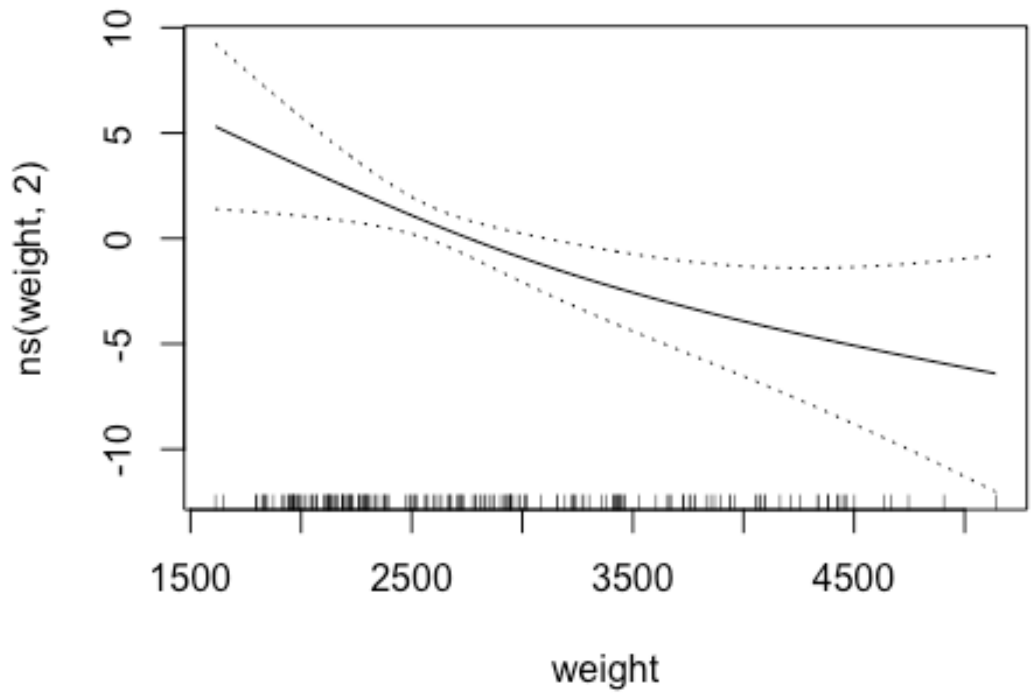

```
mean((pred_gam2- Auto.test)^2)

## [1] 6.455422

plot.Gam(gam2, se = TRUE)
```



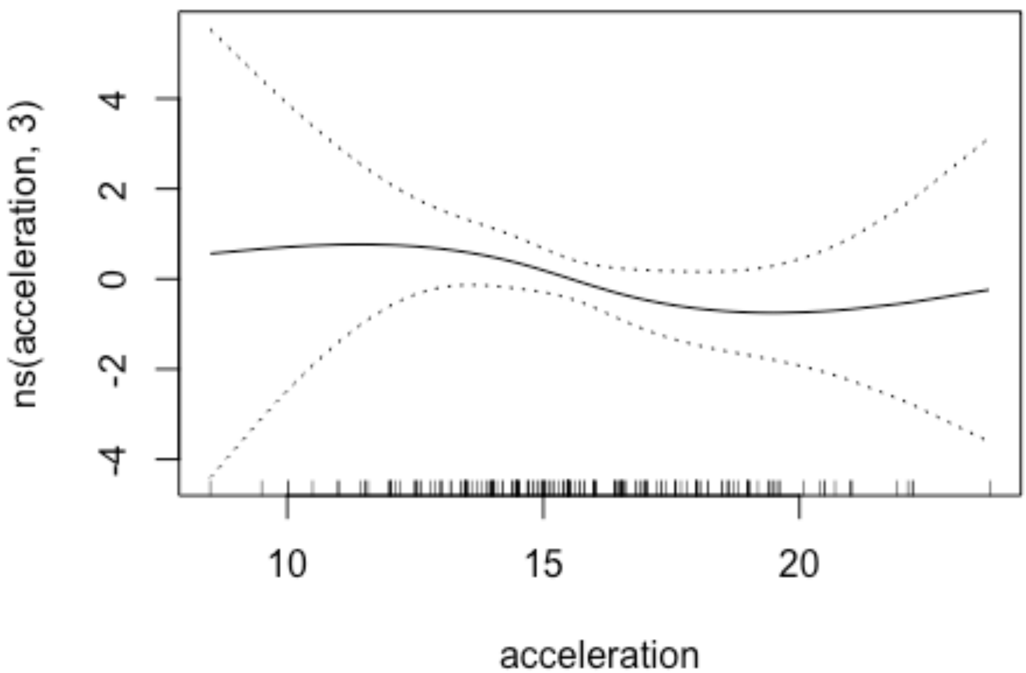
ns(displacement, 3)
10
8
6
4
2
0
-2
-4
100
200
300
400
displacement

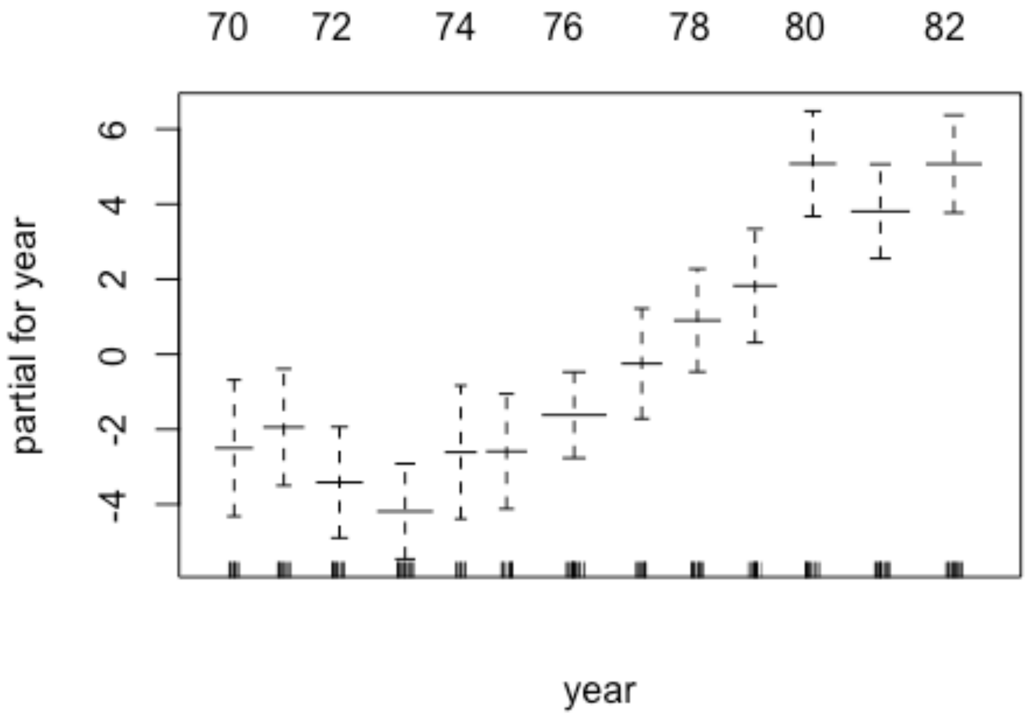


ns(horsepower, 2)
5
0
-5
-10
50
100
150
200
horsepower

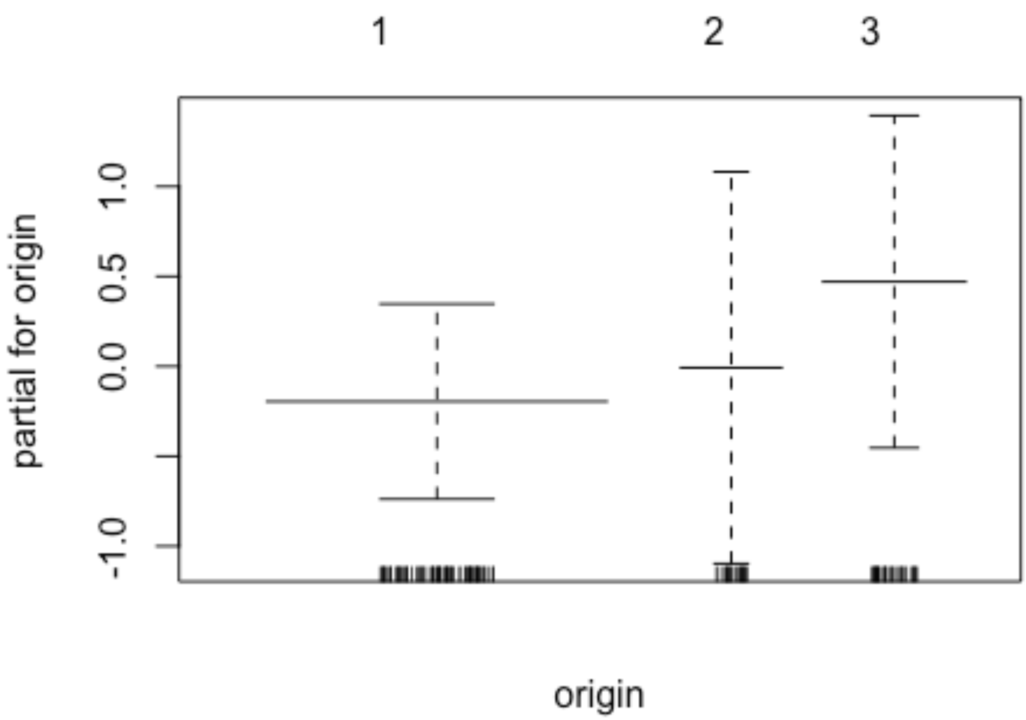


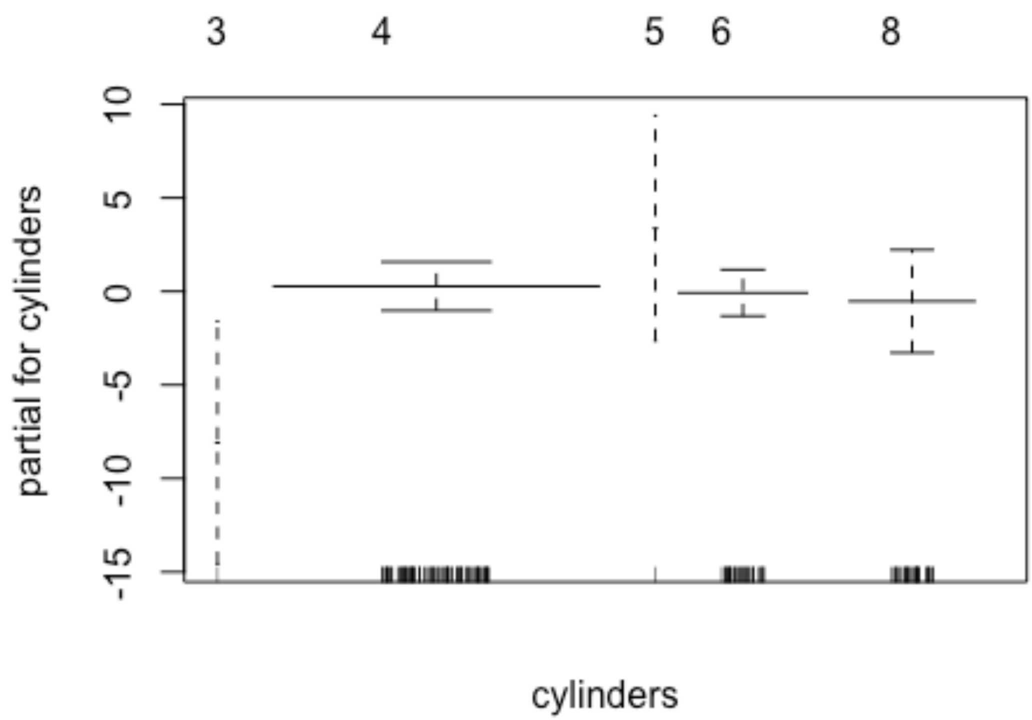
ns(weight, 2)
10
5
0
-5
-10
1500
2500
3500
4500
weight


ns(acceleration, 3)
4
2
0
-2
-4
10
15
20
acceleration

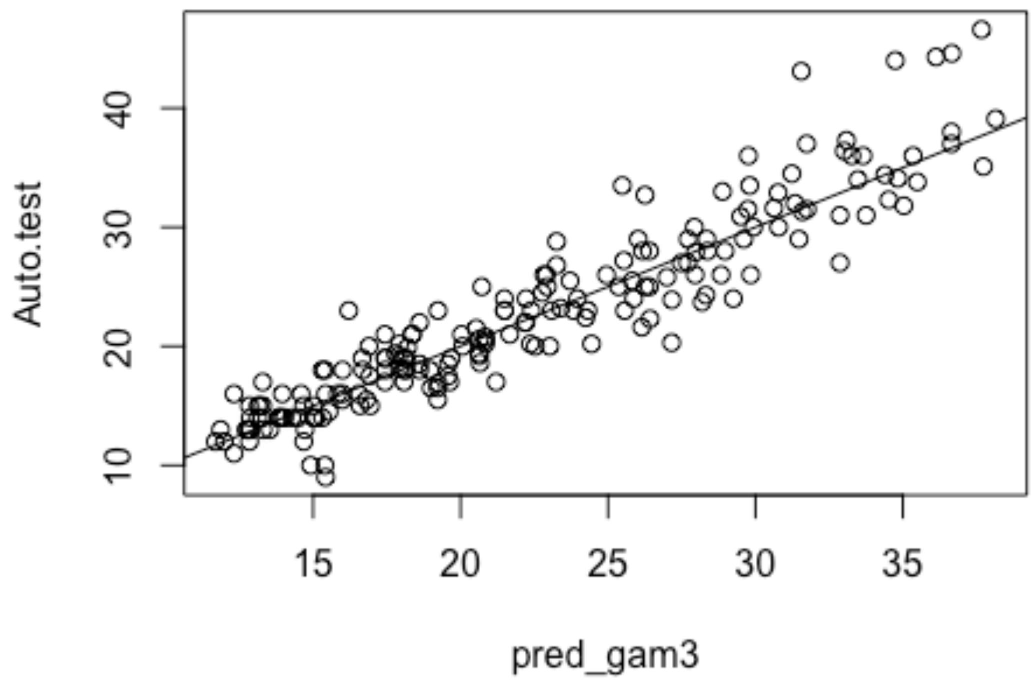


70
72
74
76
78
80
82
partial for year
6
4
2
0
-2
-4
year

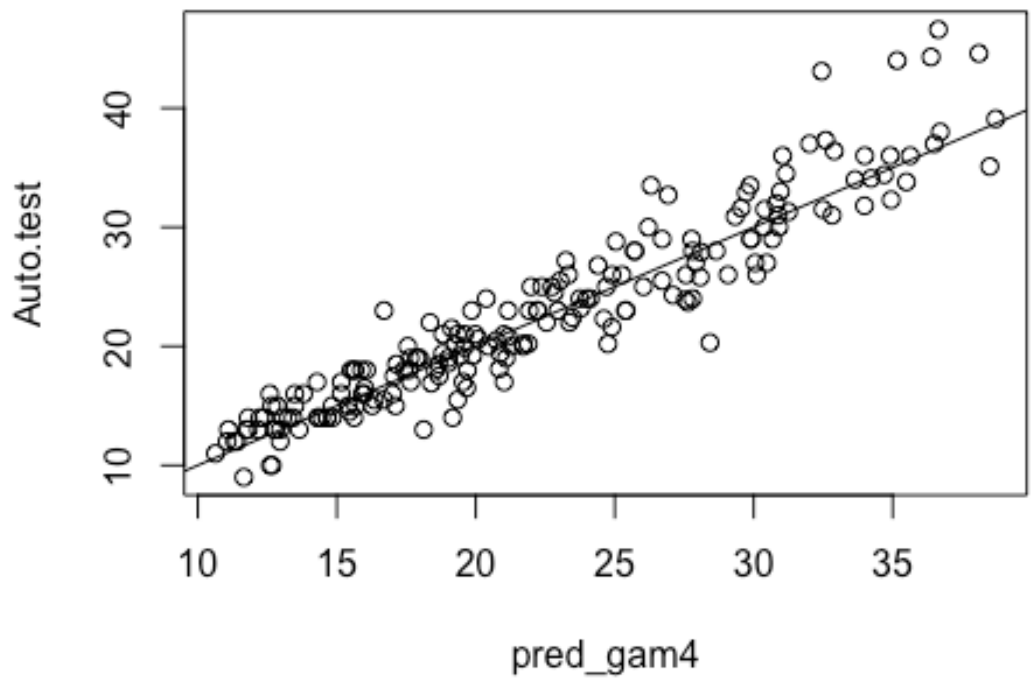

1
2
3
partial for origin
1.0
0.5
0.0
-1.0
origin



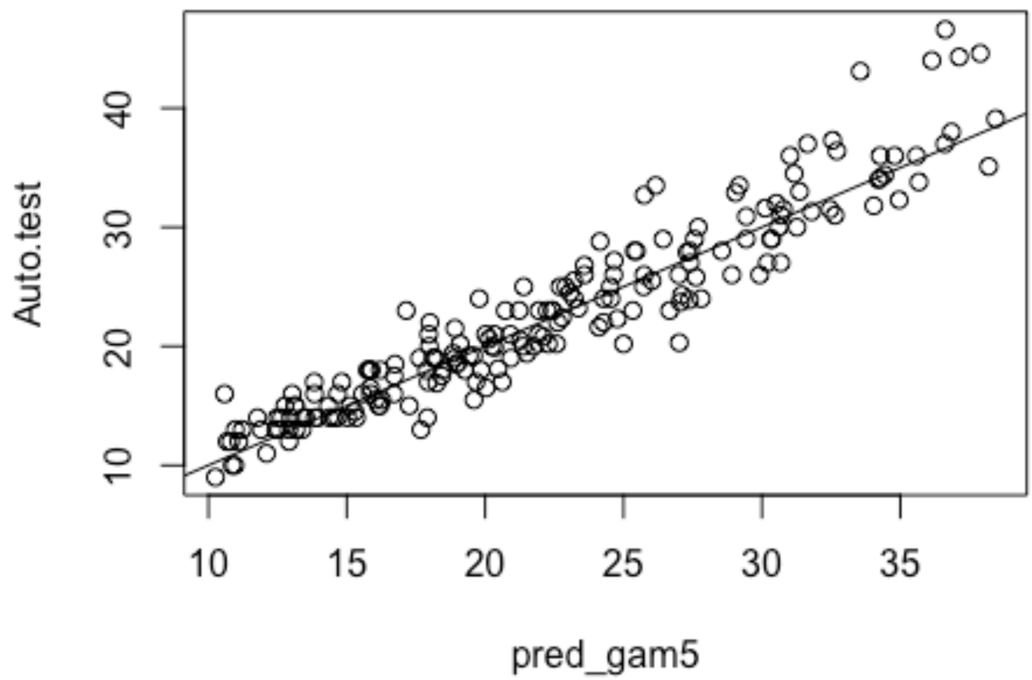

```
gam3 <- gam(mpg ~ ns(displacement, 3) + year + origin + cylinders, data =
Auto, subset = train)
pred_gam3 <- predict(gam3, newdata=Auto[-train,])
plot(pred_gam3, Auto.test)
abline(0,1)
```




```
mean((pred_gam3- Auto.test)^2)

## [1] 7.961377

gam4 <- gam(mpg ~ ns(weight,2) +ns(displacement, 3) + year + origin +
cylinders, data = Auto, subset = train)
pred_gam4 <- predict(gam4, newdata=Auto[-train,])
plot(pred_gam4, Auto.test)
abline(0,1)
```




```
mean((pred_gam4- Auto.test)^2)

## [1] 6.983401

gam5 <- gam(mpg ~ ns(horsepower, 2) + ns(weight,2) +ns(displacement, 3) +
year + origin + cylinders, data = Auto, subset = train)
pred_gam5 <- predict(gam5, newdata=Auto[-train,])
plot(pred_gam5, Auto.test)
abline(0,1)
```




```
mean((pred_gam5- Auto.test)^2)

## [1] 6.740045

anova(gam2, gam3, gam4, gam5, test = "F")

## Analysis of Deviance Table
##
## Model 1: mpg ~ ns(displacement, 3) + ns(horsepower, 2) + ns(weight, 2) +
##     ns(acceleration, 3) + year + origin + cylinders
## Model 2: mpg ~ ns(displacement, 3) + year + origin + cylinders
## Model 3: mpg ~ ns(weight, 2) + ns(displacement, 3) + year + origin +
cylinders
## Model 4: mpg ~ ns(horsepower, 2) + ns(weight, 2) + ns(displacement, 3) +
##     year + origin + cylinders
##   Resid. Df Resid. Dev Df Deviance       F    Pr(>F)
## 1       167     1185.2
## 2       174     1486.6 -7 -301.403  6.0670 2.551e-06 ***
## 3       172     1260.3  2  226.300 15.9434 4.605e-07 ***
## 4       170     1204.2  2   56.143  3.9554   0.02097 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here, I considered multiple GAMs on the basis of EDA. Firstly, I considered a model with all the features. From EDA it was evident that there is some sort of relation between displacement, weight, acceleration, and horsepower. This means that if I model using some of these and not all these related features, there should not be any significant change in the prediction capacity of the model. I modeled total of 4 GAM based on this hypothesis and found that as compared to the full model. From Anova testing, we see that this hypothesis is true. I would probably go with the model that has displacement, year, origin and cylinders as the features.

e. Considering both accuracy and interpretability of the fitted model, which of the models in (a)-(d) do you prefer? Justify your answer.

Just considering the interpretability, Single Decision tree is the most interpretable out of the lot. However, in terms of accuracy, the ensemble methods like Bagging and RandomForest perform better due to low variance.

Both these things aside, if I were to consider both accuracy and interpretability of the fitted model, I would definitely go with GAMs, because it is easy to understand the effect of each feature in the prediction. Also, comparing the accuracy, it is much better than Decision Tree and close to Bagging and RandomForest. It’s the best of both worlds!