Q28

Rahul Atre

2023-11-8

Q28 [Nonlinear Modeling]

In this exercise, you will further analyze the **Wage** data set considered throughout this chapter.

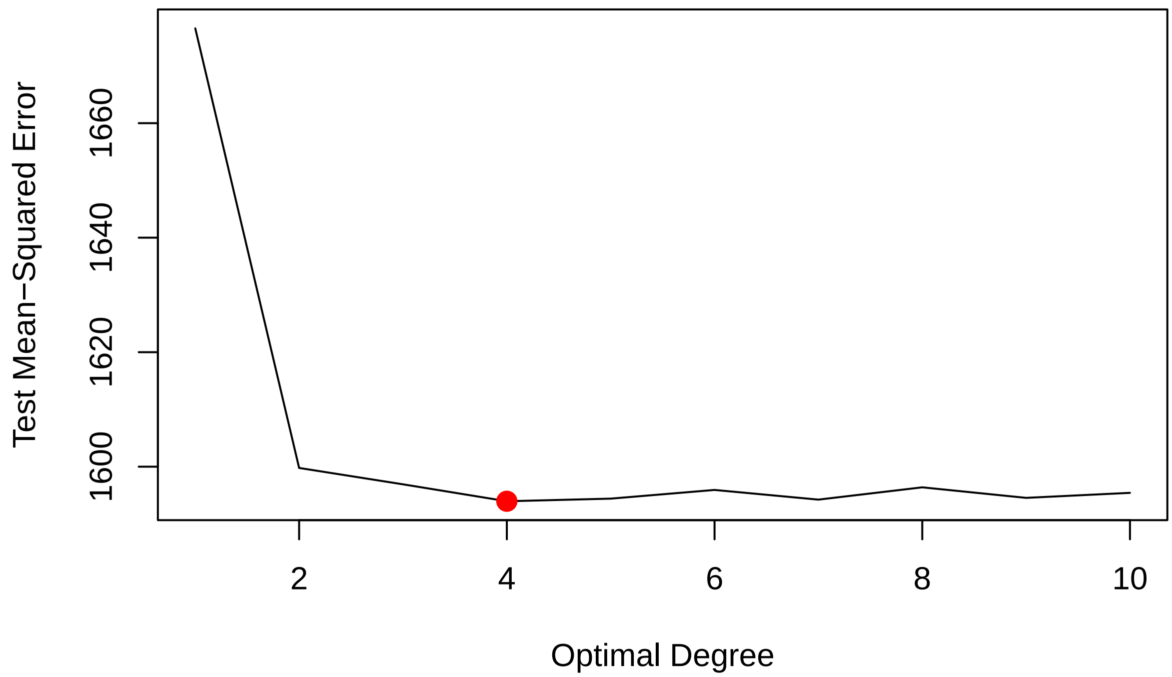
1. Perform polynomial regression to predict **wage** using **age**. Use cross-validation to select the optimal degree d for the polynomial. What degree was chosen, and how does this compare to the results of hypothesis testing using ANOVA? Make a plot of the resulting polynomial fit to the data.

Ans: First, we can load the data using the following command:

```
library(boot)
wageData = read.csv("Wage.csv")
```

Now, we need to perform the polynomial regression method on wage and age.

```
set.seed(2023) #Setting the seed for replicability
degree <- 10
cv.errors <- rep(NA, degree)

for (i in 1:degree) { #Cross-validation process
  polynomial_model = glm(wage ~ poly(age, i), data = wageData)
  cv.errors[i] = cv.glm(wageData, polynomial_model, K = 10)$delta[1]
}

#Above, K refers to number of folds in the cross-validation process, i.e. splitting the data into multi

degree.min <- which.min(cv.errors) #Minimum degree selected from cross validation
plot(1:degree, cv.errors, xlab = 'Optimal Degree', ylab = 'Test Mean-Squared Error', type = 'l')
points(degree.min, cv.errors[degree.min], col = 'red', cex = 2, pch = 20)
```



From the cross-validation process, the optimal degree for the polynomial is 4. Let's perform the ANOVA hypothesis test for each polynomial to see if the result is valid:

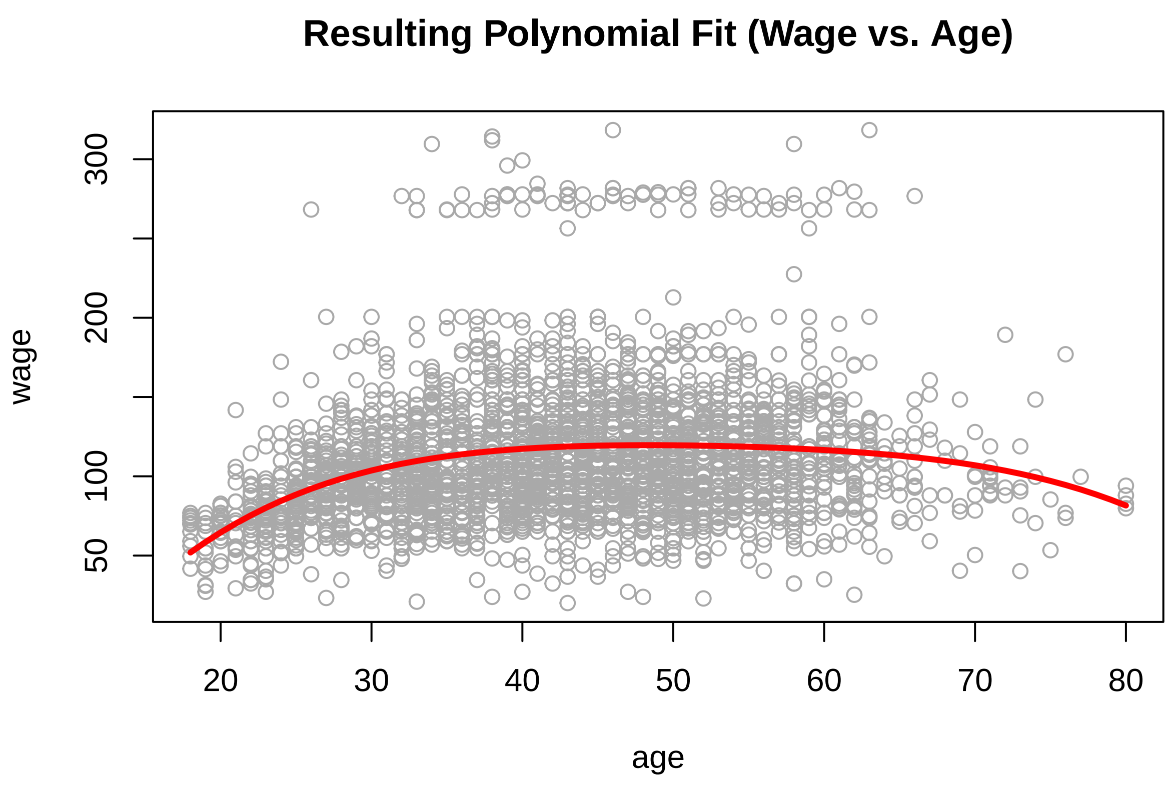
```r
fit1 <- lm(wage ~ age, data=wageData)
fit2 <- lm(wage ~ poly(age, 2), data=wageData)
fit3 <- lm(wage ~ poly(age, 3), data=wageData)
fit4 <- lm(wage ~ poly(age, 4), data=wageData)
fit5 <- lm(wage ~ poly(age, 5), data=wageData)
anova(fit1, fit2, fit3, fit4, fit5)
```

```
## Analysis of Variance Table
##
## Model 1: wage ~ age
## Model 2: wage ~ poly(age, 2)
## Model 3: wage ~ poly(age, 3)
## Model 4: wage ~ poly(age, 4)
## Model 5: wage ~ poly(age, 5)
##   Res.Df     RSS Df Sum of Sq        F    Pr(>F)
## 1   2998 5022216
## 2   2997 4793430  1    228786 143.5931 < 2.2e-16 ***
## 3   2996 4777674  1     15756   9.8888  0.001679 **
## 4   2995 4771604  1      6070   3.8098  0.051046 .
## 5   2994 4770322  1      1283   0.8050  0.369682
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the above test, we can see that the p-value for the 4th degree is 0.051046, which is very close to 0.05, therefore it is somewhat statistically significant. We can also see that the polynomial's with degree 2 and 3 have a very low p-value, so those are statistically significant as well. This analysis shows that 4 is indeed the optimal value.

Constructing the plot of the resulting polynomial fit to the data:

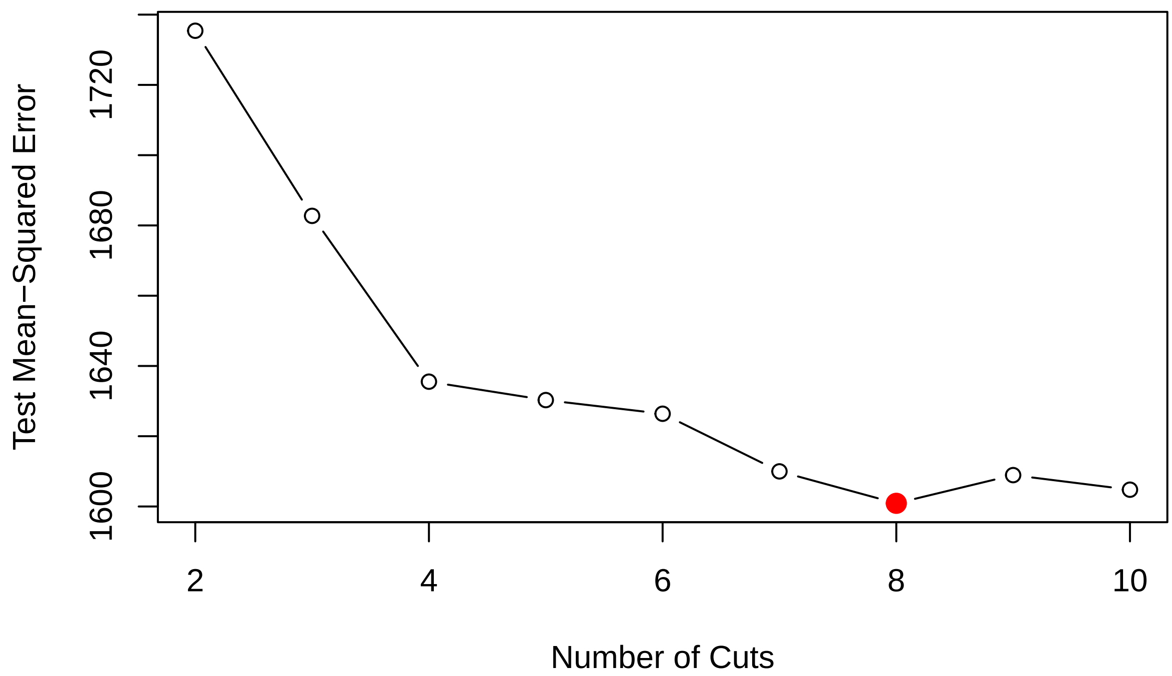
```
age.range <- range(wageData$age)
age.grid <- seq(from = age.range[1], to = age.range[2])

plot(wage ~ age, data = wageData, col = "darkgrey")
prediction <- predict(fit4, newdata = list(age = age.grid))
lines(age.grid, prediction, col = "red", lwd = 3)
title(main = "Resulting Polynomial Fit (Wage vs. Age)")
```



2. Fit a step function to predict **wage** using **age**, and perform cross-validation to choose the optimal number of cuts. Make a plot of the fit obtained.

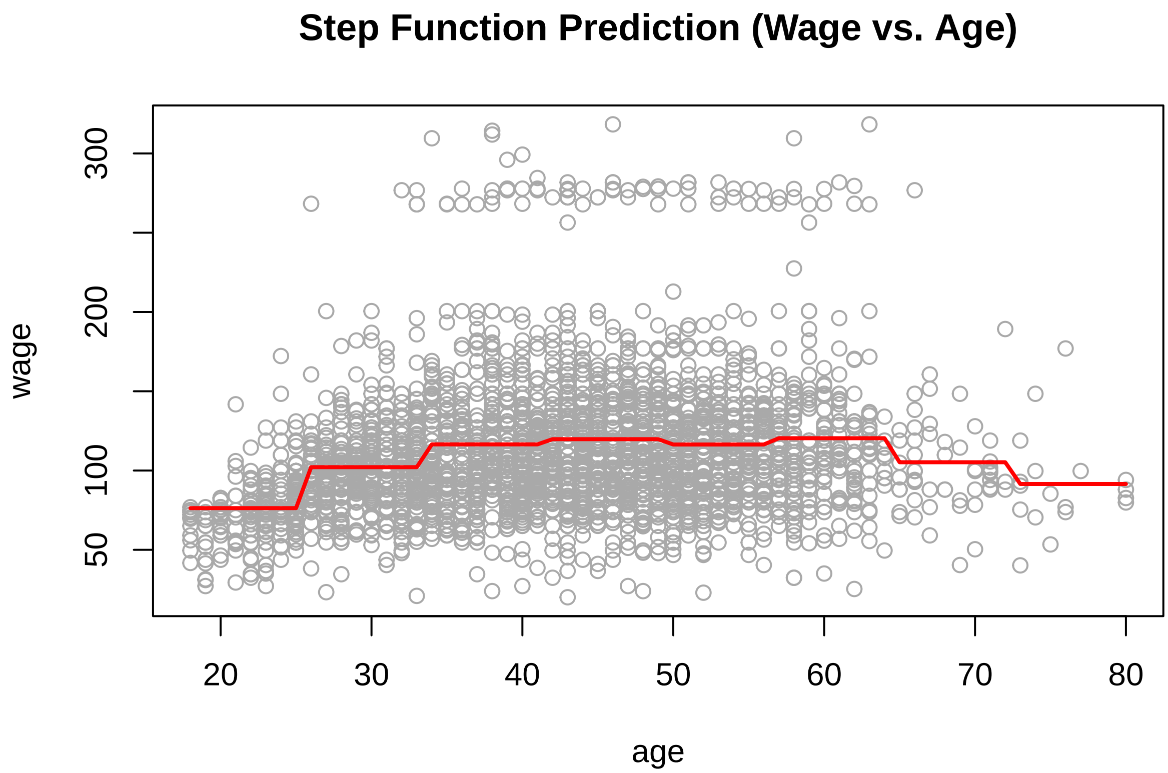
```
#First, find the optimal number of cuts using cross-validation

degree <- 10
cv.errors <- rep(NA, degree) #Reset the degree and cv.errors variables


for(i in 2:degree){
  wageData$age.cuts <- cut(wageData$age, i)
```

```r
  general_lin_model <- glm(wage ~ age.cuts, data=wageData)
  cv.errors[i] <- cv.glm(wageData, general_lin_model, K=10)$delta[1]
}

degree.min <- which.min(cv.errors) #Minimum degree selected from cross validation

plot(2:degree, cv.errors[-1], xlab="Number of Cuts", ylab="Test Mean-Squared Error", type="b")
points(degree.min, cv.errors[degree.min], col="red", cex=2, pch=20)
```



From the above graph, we can see that the optimal number of cuts is 8. We can now plot the fit obtained as follows:

```r
plot(wage ~ age, data = wageData, col = "darkgrey")
step_fit <- glm(wage ~ cut(age, 8), data = wageData)
step_prediction <- predict(step_fit, list(age = age.grid))
lines(age.grid, step_prediction, col = "red", lwd = 2)
title(main = "Step Function Prediction (Wage vs. Age)")
```


Step Function Prediction (Wage vs. Age)
wage
300
200
100
50
20
30
40
50
60
70
80
age