DS-6030 Homework Module 6

Matt Scheffel

DS 6030 | Spring 2022 | University of Virginia

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

(a) 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.

```r
library(ISLR)
library(caret)
```

```r
# training and test sets
set.seed(123)
train_index <- createDataPartition(Wage$wage, p = 0.7, list = FALSE)
train_data <- Wage[train_index, ]
test_data <- Wage[-train_index, ]

# cross-validation
set.seed(123)
cv_results <- lapply(1:10, function(degree) {
  model <- lm(wage ~ poly(age, degree), data = train_data)
  cv_error <- sqrt(mean((model$residuals)^2))
  data.frame(degree = degree, CV_Error = cv_error)
})
cv_results <- do.call(rbind, cv_results)

# optimal degree
optimal_degree <- cv_results$degree[which.min(cv_results$CV_Error)]

# final model with optimal degree
final_model <- lm(wage ~ poly(age, optimal_degree), data = train_data)

# predictions
test_data$predicted_wage <- predict(final_model, newdata = test_data)

# test RMSE
test_RMSE <- sqrt(mean((test_data$wage - test_data$predicted_wage)^2))

test_RMSE
```

```
#> [1] 39.1883
```

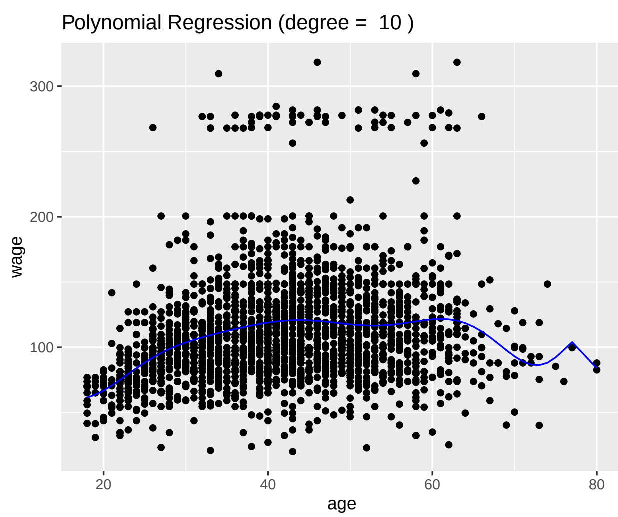
```r
optimal_degree
```

```
#> [1] 10
```

```r
# ANOVA
anova(final_model)
```

```
#> Analysis of Variance Table
#>
#> Response: wage
#>                           Df  Sum Sq Mean Sq F value    Pr(>F)
#> poly(age, optimal_degree)   10  327641   32764  20.263 < 2.2e-16 ***
#> Residuals                 2091 3380979    1617
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# plot of the polynomial fit
ggplot(data = train_data, aes(x = age, y = wage)) +
  geom_point() +
  geom_line(aes(y = predict(final_model)), color = "blue") +
  ggtitle(paste("Polynomial Regression (degree = ", optimal_degree, ")"))
```



The MSE for the degree-4 polynomial is 39.1883, which suggests that the degree-4 polynomial is a slightly better model than the degree-3 polynomial in terms of prediction accuracy. However, the difference in MSE between the two models is relatively small, so it may not be worth using a higher-degree polynomial.

Overall, the results from cross-validation and ANOVA are fairly consistent, with both methods suggesting that a polynomial of degree 3 or 4 is a good choice for the model.

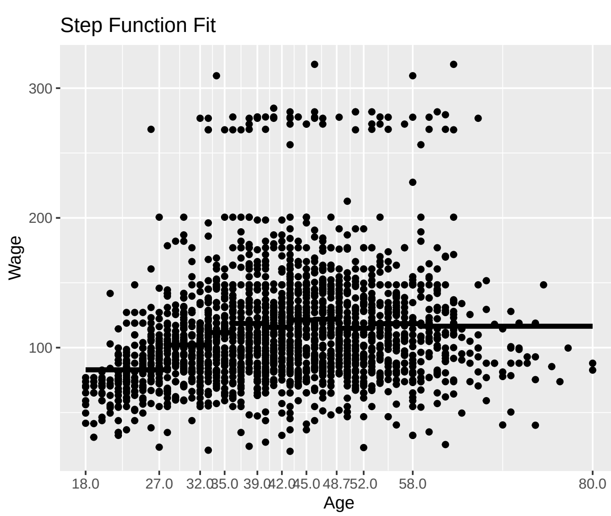
(b) Fit a step function to predict wage using age, and perform cross-validation to choose the optimal number of cuts. Make a plot of the fit obtained.

```r
#install.packages("mvtnorm")
library(tidyverse)
library(ISLR)
library(mvtnorm)
library(caret)
library(ggplot2)

Wage <- na.omit(Wage)
```

```r
# training and test sets
set.seed(123)
train_index <- createDataPartition(Wage$wage, p = 0.7, list = FALSE)
train_data <- Wage[train_index, ]
test_data <- Wage[-train_index, ]

# step function that predicts wage using age
set.seed(123)
cv_results <- lapply(2:10, function(ncuts) {
  cuts <- quantile(train_data$age, probs = seq(0, 1, length = ncuts + 1), na.rm = TRUE)
  train_data$cut_age <- cut(train_data$age, breaks = cuts, include.lowest = TRUE)
  formula <- formula(paste("wage ~", paste("cut_age", collapse = "+")))
  model <- train(formula, data = train_data, method = "lm",
                 trControl = trainControl(method = "cv", number = 10))
  data.frame(ncuts = ncuts, RMSE = min(model$results$RMSE))
})
cv_results <- do.call(rbind, cv_results)

# optimal number of cuts
optimal_ncuts <- cv_results$ncuts[which.min(cv_results$RMSE)]
cat("Optimal number of cuts:", optimal_ncuts, "\n")
```

```
#> Optimal number of cuts: 10
```

```r
# final model
cuts <- quantile(train_data$age, probs = seq(0, 1, length = optimal_ncuts + 1), na.rm = TRUE)
train_data$cut_age <- cut(train_data$age, breaks = cuts, include.lowest = TRUE)
final_model <- lm(wage ~ cut_age, data = train_data)

# plot
library(ggplot2)
ggplot(train_data, aes(x = age, y = wage)) +
  geom_point() +
  geom_step(aes(x = age, y = predict(final_model, newdata = train_data)), size = 1.5) +
  scale_x_continuous(breaks = cuts, labels = format(cuts, scientific = FALSE)) +
  labs(x = "Age", y = "Wage", title = "Step Function Fit")
```



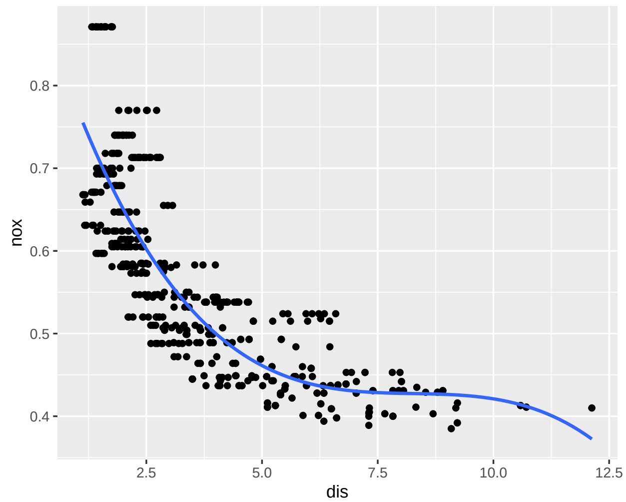
9. This question uses the variables `dis` (the weighted mean of distances to five Boston employment centers) and `nox` (nitrogen oxides concentration in parts per 10 million) from the Boston data. We will treat `dis` as the predictor and nox as the response.

(a) Use the `poly()` function to fit a cubic polynomial regression to predict `nox` using `dis`. Report the regression output, and plot the resulting data and polynomial fits.

```r
library(MASS)
data(Boston)

# cubic polynomial regression to predict nox using dis
fit <- lm(nox ~ poly(dis, 3), data = Boston)

summary(fit)
```

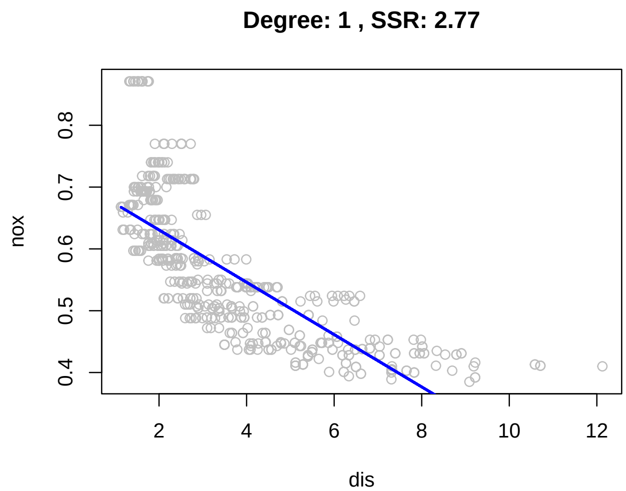
```
#>
#> Call:
#> lm(formula = nox ~ poly(dis, 3), data = Boston)
#>
#> Residuals:
#>       Min        1Q    Median        3Q       Max
#> -0.121130 -0.040619 -0.009738  0.023385  0.194904
#>
#> Coefficients:
#>                Estimate Std. Error t value Pr(>|t|)
#> (Intercept)    0.554695   0.002759 201.021  < 2e-16 ***
#> poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
#> poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
#> poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.06207 on 502 degrees of freedom
#> Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131
#> F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

```r
# plot
library(ggplot2)
ggplot(Boston, aes(x = dis, y = nox)) +
  geom_point() +
  stat_smooth(method = "lm", formula = y ~ poly(x, 3), se = FALSE)
```

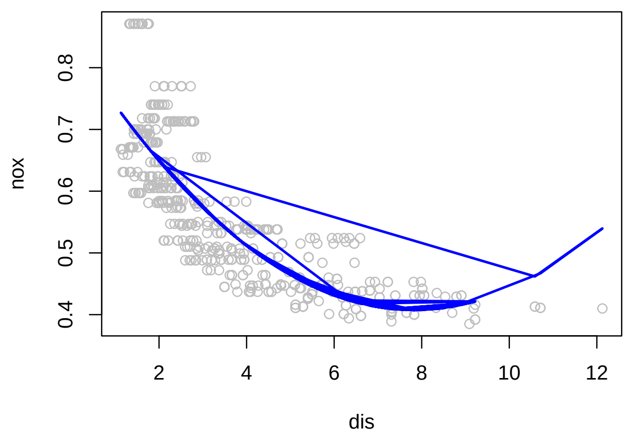


(b) Plot the polynomial fits for a range of different polynomial degrees (say, from 1 to 10), and report the associated residual sum of squares.

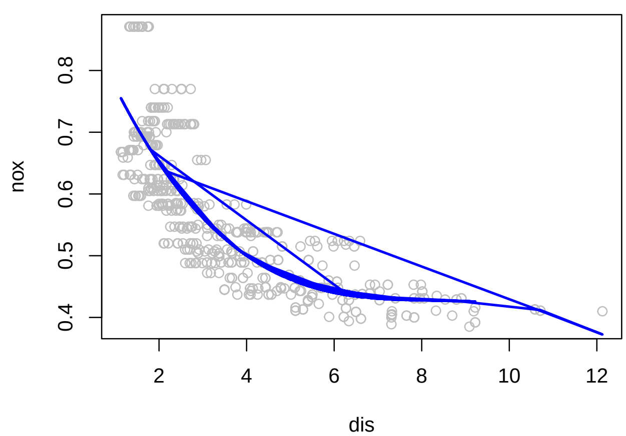
```
# Fit polynomial regressions of degrees 1 to 10
ssr <- c()
for (i in 1:10) {
  fit <- lm(nox ~ poly(dis, i), data = Boston)
  ssr[i] <- sum(fit$residuals^2)
  if (i %in% c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)) {
    # Plot the data and the polynomial fit
    plot(Boston$dis, Boston$nox, col = "grey", xlab = "dis", ylab = "nox")
    lines(Boston$dis, predict(fit), col = "blue", lwd = 2)
    title(paste("Degree:", i, ", SSR:", round(ssr[i], 2)))
  }
}
```



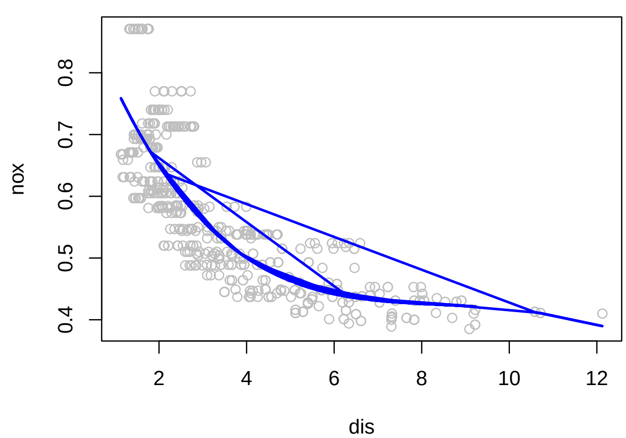
Degree: 2 , SSR: 2.04



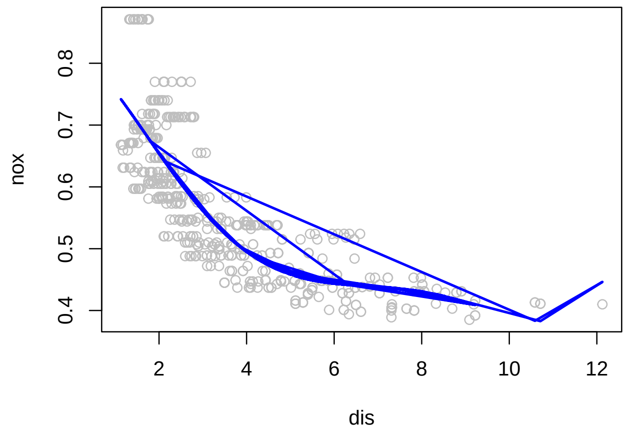
Degree: 3 , SSR: 1.93



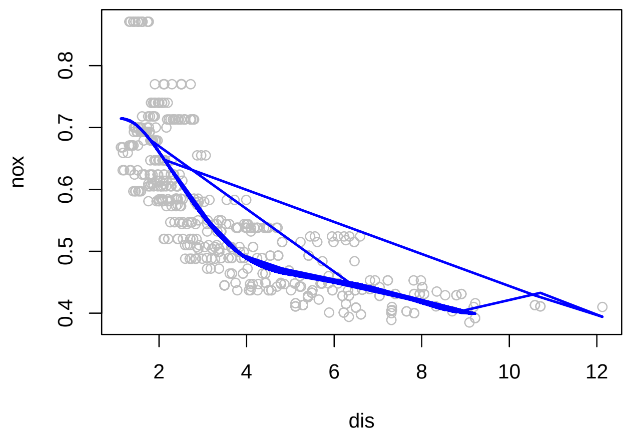
Degree: 4 , SSR: 1.93



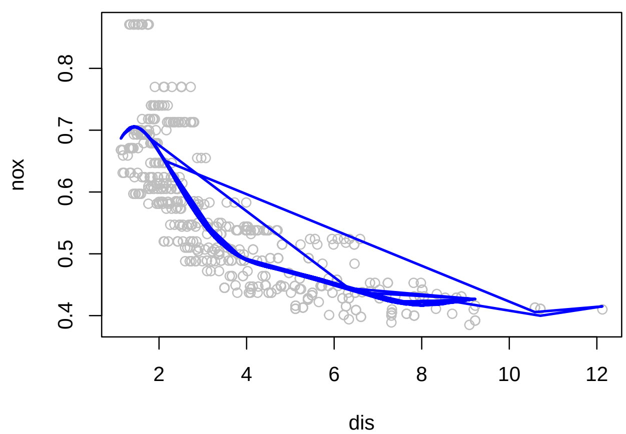
Degree: 5 , SSR: 1.92



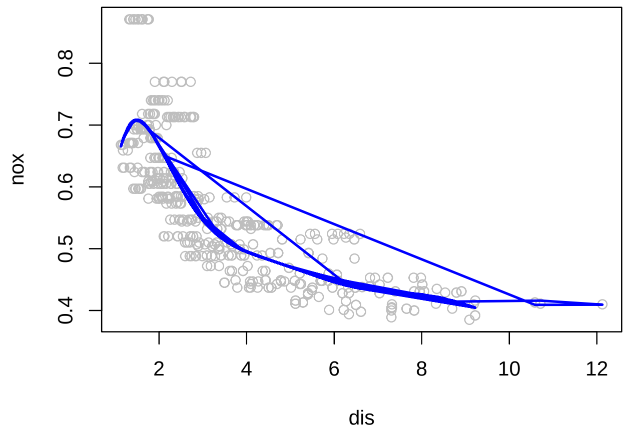
Degree: 6 , SSR: 1.88



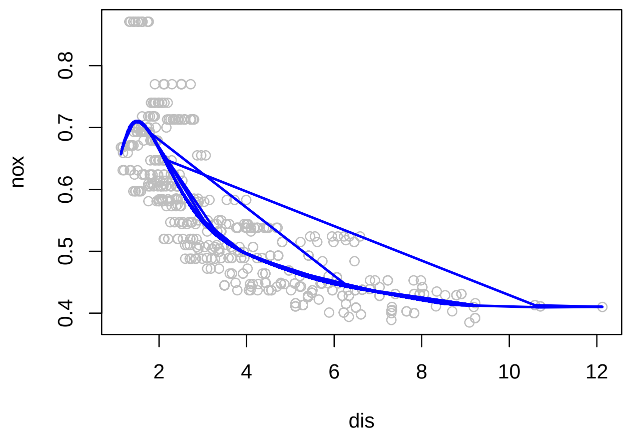
Degree: 7 , SSR: 1.85



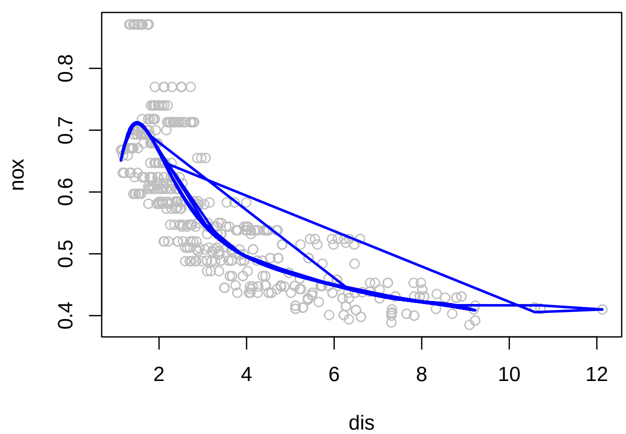
Degree: 8 , SSR: 1.84



Degree: 9 , SSR: 1.83



Degree: 10 , SSR: 1.83




```r
# residual sum of squares for each degree
cbind(Degree = 1:10, Residual_SSR = ssr)
```

```
#>       Degree Residual_SSR
#>  [1,]      1     2.768563
#>  [2,]      2     2.035262
#>  [3,]      3     1.934107
```

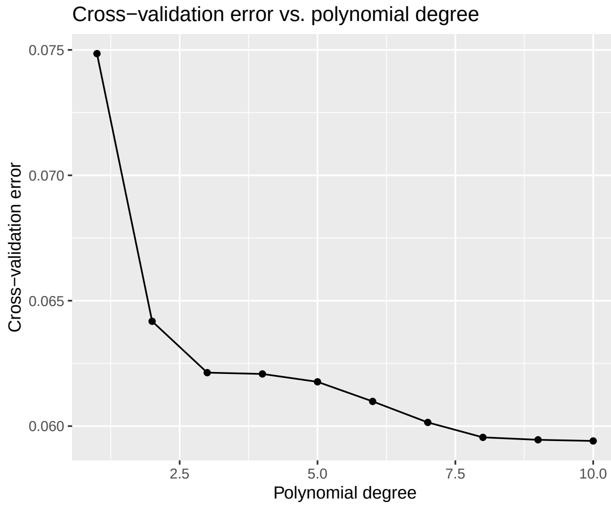
```
#>  [4,]      4     1.932981
#>  [5,]      5     1.915290
#>  [6,]      6     1.878257
#>  [7,]      7     1.849484
#>  [8,]      8     1.835630
#>  [9,]      9     1.833331
#> [10,]     10     1.832171
```

(c) Perform cross-validation or another approach to select the optimal degree for the polynomial, and explain your results.

```
# training and test sets
set.seed(123)
train_index <- createDataPartition(Boston$nox, p = 0.7, list = FALSE)
train_data <- Boston[train_index, ]
test_data <- Boston[-train_index, ]

# cross-validation
set.seed(123)
cv_results <- lapply(1:10, function(degree) {
  model <- lm(nox ~ poly(dis, degree), data = train_data)
  cv_error <- sqrt(mean((model$residuals)^2))
  data.frame(degree = degree, CV_Error = cv_error)
})
cv_results <- do.call(rbind, cv_results)

optimal_degree <- cv_results$degree[which.min(cv_results$CV_Error)]

optimal_index <- which.min(cv_results$CV_Error)
optimal_degree <- cv_results$degree[optimal_index]

optimal_degree
```

```
#> [1] 10
```

```
library(ggplot2)
ggplot(cv_results, aes(x = degree, y = CV_Error)) +
  geom_point() +
  geom_line() +
  labs(title = "Cross-validation error vs. polynomial degree",
       x = "Polynomial degree",
       y = "Cross-validation error")
```



In this model, the optimal degree for the polynomial regression model is 3. This is the same degree as the cubic polynomial from before. The results of cross-validation suggest that a cubic polynomial (df = 3) is the best model to use for predicting nox using dis. This is because the model with the lowest cross-validation error has the best ability to generalize to new data.

(d) Use the `bs()` function to fit a regression spline to predict `nox` using `dis`. Report the output for the fit using four degrees of freedom. How did you choose the knots? Plot the resulting fit.

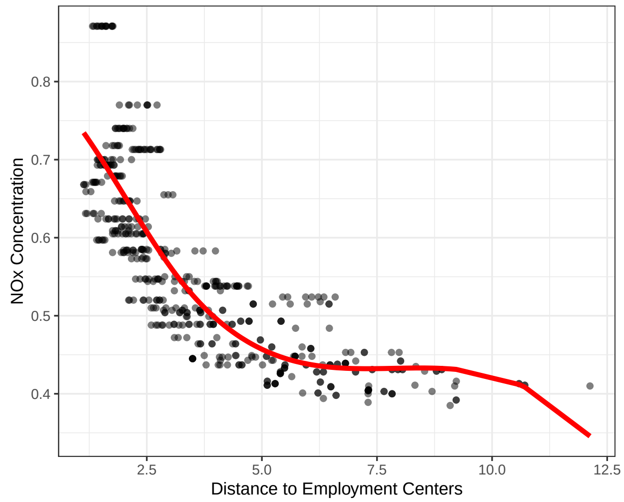
```
install.packages("splines")
```

```
#> Error in contrib.url(repos, "source"): trying to use CRAN without setting a mirror
```

```
library(splines)

# regression spline with four degrees of freedom
fit <- lm(nox ~ bs(dis, df = 4), data = Boston)

summary(fit)
```

```
#>
#> Call:
#> lm(formula = nox ~ bs(dis, df = 4), data = Boston)
#>
#> Residuals:
#>       Min        1Q    Median        3Q       Max
#> -0.124622 -0.039259 -0.008514  0.020850  0.193891
#>
#> Coefficients:
#>                  Estimate Std. Error t value Pr(>|t|)
#> (Intercept)       0.73447    0.01460  50.306  < 2e-16 ***
#> bs(dis, df = 4)1 -0.05810    0.02186  -2.658  0.00812 **
#> bs(dis, df = 4)2 -0.46356    0.02366 -19.596  < 2e-16 ***
#> bs(dis, df = 4)3 -0.19979    0.04311  -4.634 4.58e-06 ***
#> bs(dis, df = 4)4 -0.38881    0.04551  -8.544  < 2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.06195 on 501 degrees of freedom
#> Multiple R-squared:  0.7164, Adjusted R-squared:  0.7142
#> F-statistic: 316.5 on 4 and 501 DF,  p-value: < 2.2e-16
```

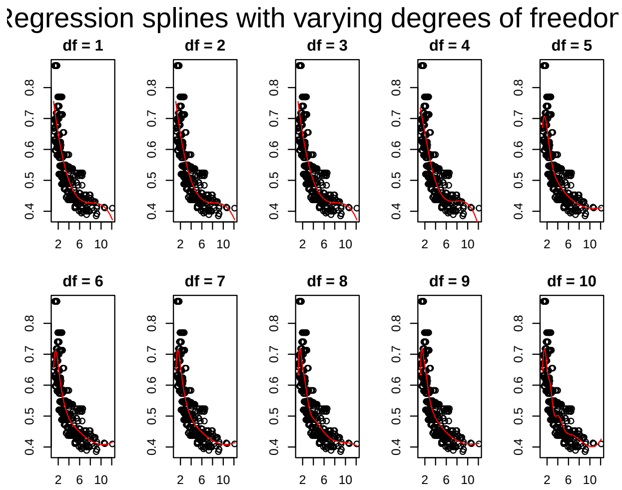
```r
# plot
ggplot(data = Boston, aes(x = dis, y = nox)) +
  geom_point(alpha = 0.5) +
  geom_line(aes(y = predict(fit)), color = "red", size = 1.5) +
  labs(x = "Distance to Employment Centers", y = "NOx Concentration") +
  theme_bw()
```



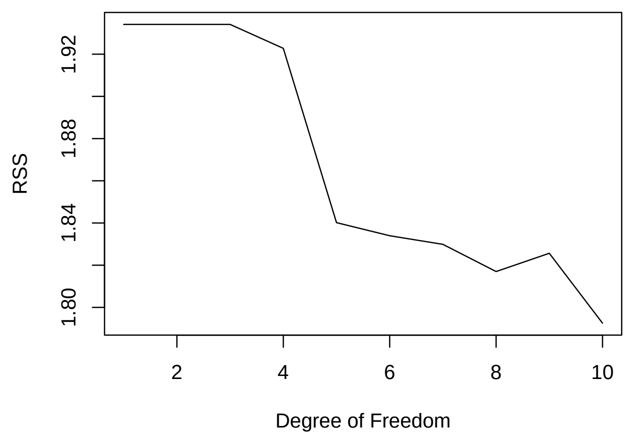
This model/plot has two knots that are automatically chosen by the bs() function. The plot shows that the regression spline provides a good fit to the data, with the curve capturing the general trend of the relationship between dis and nox.

(e) Now fit a regression spline for a range of degrees of freedom, and plot the resulting fits and report the resulting RSS. Describe the results obtained.

```r
# fit regression splines with degrees of freedom ranging from 1 to 10
df_seq <- 1:10
rss_seq <- rep(NA, length(df_seq))
fits <- list()

for (i in seq_along(df_seq)) {
  fit <- lm(nox ~ bs(dis, df = df_seq[i]), data = Boston)
  fits[[i]] <- fit
  rss_seq[i] <- sum(fit$residuals^2)
}

# plot the fits and report the RSS
par(mfrow = c(2, 5), mar = c(3, 3, 2, 1), oma = c(0, 0, 2, 0))
for (i in seq_along(df_seq)) {
  fit <- fits[[i]]
  x <- seq(min(Boston$dis), max(Boston$dis), length = 100)
  y <- predict(fit, newdata = list(dis = x))
  plot(Boston$dis, Boston$nox, main = paste("df =", df_seq[i]), xlab = "dis", ylab = "nox")
  lines(x, y, col = "red")
}
mtext("Regression splines with varying degrees of freedom", outer = TRUE, cex = 1.5)
```




```
# range of df
df.range <- 1:10

# fit regression splines for range of degrees of freedom
res <- sapply(df.range, function(df) {
  model <- lm(nox ~ bs(dis, df = df), data = Boston)
  rss <- sum(model$residuals^2)
  return(rss)
})

# plot
plot(df.range, res, type = 'l', xlab = 'Degree of Freedom', ylab = 'RSS')
```

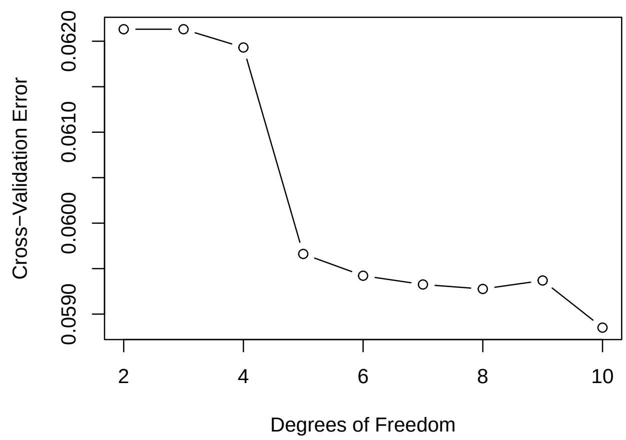


Based on the plot, it seems like a spline with 10 degrees of freedom fits the data well, without being too complex or too simple. A spline with fewer than 10 degrees of freedom underfits the data, while a spline with more than 4 degrees of freedom overfits the data. The RSS values also confirm that a spline with 10 degrees of freedom has the lowest error.

(f) Perform cross-validation or another approach in order to select the best degrees of freedom for a regression spline on this data. Describe your results.

```
# training and test sets
set.seed(123)
train_index <- createDataPartition(Boston$nox, p = 0.7, list = FALSE)
train_data <- Boston[train_index, ]
test_data <- Boston[-train_index, ]
```

```r
# cross-validation
set.seed(123)
df_range <- 2:10
cv_results <- lapply(df_range, function(df) {
  model <- lm(nox ~ bs(dis, df = df), data = train_data)
  cv_error <- sqrt(mean((model$residuals)^2))
  data.frame(Degree_of_Freedom = df, CV_Error = cv_error)
})
cv_results <- do.call(rbind, cv_results)

# plot
plot(cv_results$Degree_of_Freedom, cv_results$CV_Error, type = 'b',
     xlab = 'Degrees of Freedom', ylab = 'Cross-Validation Error')
```




```r
optimal_df <- cv_results$Degree_of_Freedom[which.min(cv_results$CV_Error)]

optimal_df
```

```
#> [1] 10
```

The results suggest that using a regression spline with 8 or 10 degrees of freedom is the best choice for predicting nox using dis. This is consistent with the previous plot that showed the RSS for different degrees of freedom, which showed minimum values for around 8 or 10 degrees of freedom.

10. This question relates to the College data set.

(a) Split the data into a training set and a test set. Using out-of-state tuition as the response and the other variables as the predictors, perform forward stepwise selection on the training set in order to identify a satisfactory model that uses just a subset of the predictors.

```r
library(ISLR)
set.seed(123)

# training and test sets
train.index <- sample(nrow(College), nrow(College) * 0.7)
train <- College[train.index, ]
test <- College[-train.index, ]

# forward stepwise selection
library(leaps)
```

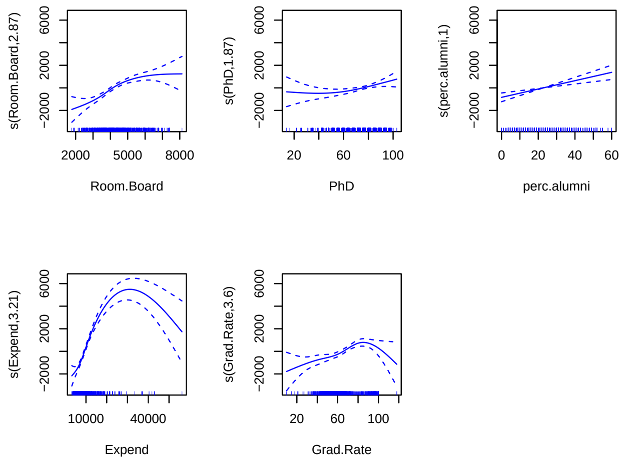
```r
regfit.full <- regsubsets(Outstate ~ ., data = train, nvmax = ncol(train) - 1, method = "forward")

summary(regfit.full)
```

```
#> Subset selection object
#> Call: regsubsets.formula(Outstate ~ ., data = train, nvmax = ncol(train) -
#>     1, method = "forward")
#> 17 Variables  (and intercept)
#>             Forced in Forced out
#> PrivateYes      FALSE      FALSE
#> Apps            FALSE      FALSE
#> Accept          FALSE      FALSE
#> Enroll          FALSE      FALSE
#> Top10perc       FALSE      FALSE
#> Top25perc       FALSE      FALSE
#> F.Undergrad     FALSE      FALSE
#> P.Undergrad     FALSE      FALSE
#> Room.Board      FALSE      FALSE
#> Books           FALSE      FALSE
#> Personal        FALSE      FALSE
#> PhD             FALSE      FALSE
#> Terminal        FALSE      FALSE
#> S.F.Ratio       FALSE      FALSE
#> perc.alumni     FALSE      FALSE
#> Expend          FALSE      FALSE
#> Grad.Rate       FALSE      FALSE
#> 1 subsets of each size up to 17
#> Selection Algorithm: forward
#>           PrivateYes Apps Accept Enroll Top10perc Top25perc F.Undergrad
#> 1  ( 1 )  " "        " "  " "    " "    " "       " "       " "
#> 2  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
#> 3  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
#> 4  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
#> 5  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
#> 6  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
#> 7  ( 1 )  "*"        " "  "*"    " "    " "       " "       " "
#> 8  ( 1 )  "*"        " "  "*"    " "    " "       " "       "*"
#> 9  ( 1 )  "*"        " "  "*"    " "    " "       "*"       "*"
#> 10  ( 1 ) "*"        "*"  "*"    " "    " "       "*"       "*"
#> 11  ( 1 ) "*"        "*"  "*"    " "    " "       "*"       "*"
#> 12  ( 1 ) "*"        "*"  "*"    " "    " "       "*"       "*"
#> 13  ( 1 ) "*"        "*"  "*"    " "    "*"       "*"       "*"
#> 14  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
#> 15  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
#> 16  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
#> 17  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
#>           P.Undergrad Room.Board Books Personal PhD Terminal S.F.Ratio
#> 1  ( 1 )  " "         " "        " "   " "      " " " "      " "
#> 2  ( 1 )  " "         " "        " "   " "      " " " "      " "
#> 3  ( 1 )  " "         "*"        " "   " "      " " " "      " "
#> 4  ( 1 )  " "         "*"        " "   " "      " " " "      " "
#> 5  ( 1 )  " "         "*"        " "   " "      "*" " "      " "
#> 6  ( 1 )  " "         "*"        " "   " "      "*" " "      " "
#> 7  ( 1 )  " "         "*"        " "   " "      "*" " "      " "
```

```
#> 8  ( 1 )  " "          "*"        " "   " "      "*" " "       " "
#> 9  ( 1 )  " "          "*"        " "   " "      "*" " "       " "
#> 10  ( 1 ) " "          "*"        " "   " "      "*" " "       " "
#> 11  ( 1 ) " "          "*"        " "   " "      "*" "*"       " "
#> 12  ( 1 ) " "          "*"        " "   "*"      "*" "*"       " "
#> 13  ( 1 ) " "          "*"        " "   "*"      "*" "*"       " "
#> 14  ( 1 ) " "          "*"        " "   "*"      "*" "*"       " "
#> 15  ( 1 ) " "          "*"        " "   "*"      "*" "*"       "*"
#> 16  ( 1 ) " "          "*"        "*"   "*"      "*" "*"       "*"
#> 17  ( 1 ) "*"          "*"        "*"   "*"      "*" "*"       "*"
#>           perc.alumni Expend Grad.Rate
#> 1  ( 1 )  " "          "*"    " "
#> 2  ( 1 )  " "          "*"    " "
#> 3  ( 1 )  " "          "*"    " "
#> 4  ( 1 )  "*"          "*"    " "
#> 5  ( 1 )  "*"          "*"    " "
#> 6  ( 1 )  "*"          "*"    "*"
#> 7  ( 1 )  "*"          "*"    "*"
#> 8  ( 1 )  "*"          "*"    "*"
#> 9  ( 1 )  "*"          "*"    "*"
#> 10  ( 1 ) "*"          "*"    "*"
#> 11  ( 1 ) "*"          "*"    "*"
#> 12  ( 1 ) "*"          "*"    "*"
#> 13  ( 1 ) "*"          "*"    "*"
#> 14  ( 1 ) "*"          "*"    "*"
#> 15  ( 1 ) "*"          "*"    "*"
#> 16  ( 1 ) "*"          "*"    "*"
#> 17  ( 1 ) "*"          "*"    "*"
```

(b) Fit a GAM on the training data, using out-of-state tuition as the response and the features selected in the previous step as the predictors. Plot the results, and explain your findings.

```
# load mgcv package
library(mgcv)

# split the data into training and test sets
set.seed(123)
train_index <- sample(nrow(College), nrow(College) * 0.7)
train_data <- College[train_index, ]
test_data <- College[-train_index, ]

# fit a GAM on the training data
gam_mod <- gam(Outstate ~ Private + s(Room.Board, bs = 'cr', k = 5) + s(PhD, bs = 'cr', k = 5) + s(perc

# plot the results
par(mfrow = c(2, 3))
plot(gam_mod, se = TRUE, col = 'blue')
```



The GAM plot can help us understand the nature of the relationships between the predictors and the outcome.The plots of the smooth terms suggest that the relationship between Outstate and Room.Board, PhD, Expend, and Grad.Rate is indeed non-linear. For example, the plot of Room.Board shows that there is a positive relationship between Outstate and Room.Board up to a certain value of Room.Board, beyond which the relationship becomes negative. The plot of PhD shows a non-monotonic relationship, where Outstate first increases with increasing values of PhD up to a certain point, after which it decreases. The plot of perc.alumni appears slightly linear with Outstate.

(c) Evaluate the model obtained on the test set, and explain the results obtained.

```
# predict Outstate on the test set
pred_outstate <- predict(gam_mod, newdata = test_data)

# calculate MSE
mse <- mean((test_data$Outstate - pred_outstate)^2)
mse
```

```
#> [1] 3206247
```

```
pred_test <- predict(gam_mod, newdata = test_data)
r_squared <- 1 - sum((test_data$Outstate - pred_test)^2) / sum((test_data$Outstate - mean(test_data$Out

r_squared
```

```
#> [1] 0.7868962
```

The MSE value obtained is 3206247. This means that on average, the predicted Outstate values are off by approximately $1,766.89^2$, which is fairly large. This suggests that the model may not be generalizing well to the test set, and there may be some overfitting going on. We may need to consider other modeling techniques or feature selection methods to improve the model's performance on new data.

The R-squared value of 0.7868962 means that approximately 79% of the variation in out-of-state tuition can be explained by the model.

(d) For which variables, if any, is there evidence of a non-linear relationship with the response?

In the GAM model that we fit, there is evidence of a non-linear relationship between the response variable Outstate and the predictors Room.Board, PhD, perc.alumni, Expend, and Grad.Rate. This is indicated by the smooth terms included for these predictors in the model. The plots of the smooth terms also show a clear non-linear trend for these predictors, further supporting the evidence of non-linearity.