Homework 4

#for nicer table output  
library(knitr)  
library(tidyverse)

# Conceptual Problems

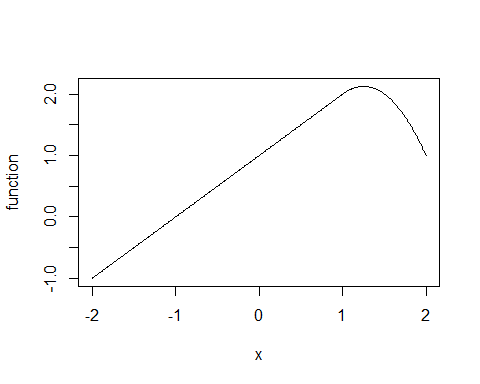
Section 7.9

* Book Problem 3 (you can sketch these by hand and take a picture or use R/python/etc. to do so). The problem is reproduced below.

Suppose we fit a curve with basis functions . (Note that equals 1 for and 0 otherwise.) We fit the linear regression model

and obtain estimates . Sketch the estimated curve between and . Note the intercepts, slopes, and other relevant information.

#quick function to plot  
poly\_to\_plot <- function(x, beta\_0, beta\_1, beta\_2){  
 if (x<1){  
 beta\_0+beta\_1\*x  
 } else {  
 beta\_0+beta\_1\*x+beta\_2\*(x-1)^2  
 }  
}  
poly\_to\_plot\_V <- Vectorize(poly\_to\_plot)  
#sequence to plot  
x <- seq(from = -2, to = 2, by = 0.01)  
plot(x, poly\_to\_plot\_V(x, 1, 1, -2), type = "l", ylab = "function")



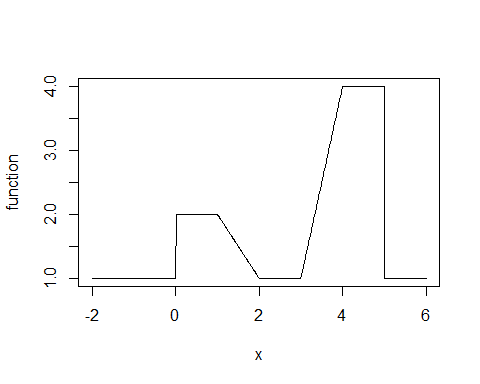
We can see that the function is linear until we reach 1. At that point we become a quadratic function. The slope over the linear region is and the intercept is .

* Book Problem 4 (you can sketch these by hand and take a picture or use R/python/etc. to do so). The problem is reproduced below.

Suppose we fit a curve with basis functions . We fit the linear regression model

and obtain estimates . Sketch the estimated curve between and . Note the intercepts, slopes, and other relevant information.

#quick function to plot  
poly\_to\_plot\_2 <- function(x, beta\_0, beta\_1, beta\_2){  
 if (x > 0 & x <= 2){  
 if (x > 1) {  
 beta\_0 + beta\_1\*(1-(x-1))  
 } else {  
 beta\_0 + beta\_1  
 }  
 } else if (x >= 3 & x <= 4) {  
 beta\_0 + beta\_2\*(x-3)  
 } else if (x > 4 & x < 5){  
 beta\_0 + beta\_2  
 } else {  
 beta\_0  
 }  
}  
poly\_to\_plot\_2V <- Vectorize(poly\_to\_plot\_2)  
#sequence to plot  
x <- seq(from = -2, to = 6, by = 0.01)  
plot(x, poly\_to\_plot\_2V(x, 1, 1, 3), type = "l", ylab = "function")



We have a piecewise constant, and linear function.

* Book Problem 5. The problem is reproduced below.

Consdier two curves, and , defined by

where represents the th derivative of .

1. As , will or have the smaller training RSS?

will be more flexible and therefore, should be able to fit the training data better.

1. As , will or have the smaller test RSS?

This really depends on the complexity of the true relationship! If the true relationship is highly non-linear, then will likely do better. If the relationship is simpler, then will likely do better.

1. For , will or have the smaller training and test RSS?

They will be the same as we aren’t penalizing anything!

# Implementation Problems

#Solutions from Dr. Maity, modified by Dr. Post  
# libraries  
library(ISLR2)  
library(rsample)  
library(boot)  
library(splines)  
library(leaps)  
library(caret)  
library(MASS)  
library(klaR)  
library(class)  
library(gam)  
set.seed(1)

1. 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.
2. 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.

# fit the model  
fit <- lm(nox ~ poly(dis, 3), data = Boston)  
coef(summary(fit)) |>  
 knitr::kable()

|  | Estimate | Std. Error | t value | Pr(>|t|) |
| --- | --- | --- | --- | --- |
| (Intercept) | 0.5546951 | 0.0027594 | 201.020893 | 0e+00 |
| poly(dis, 3)1 | -2.0030959 | 0.0620709 | -32.271071 | 0e+00 |
| poly(dis, 3)2 | 0.8563300 | 0.0620709 | 13.795987 | 0e+00 |
| poly(dis, 3)3 | -0.3180490 | 0.0620709 | -5.123959 | 4e-07 |

Scatter plot between dis (x) and nox (y) is shown with the cubic polynomial fit overlayed. The nox values generally decrease rapidly for smaller values of dis and then level off. The cubic polynomial fit does an excellent job fitting the data.

# plot the resulting data  
dis\_grid <- seq(min(Boston$dis), max(Boston$dis))  
preds <- predict(fit, list(dis = dis\_grid), se = TRUE)  
plot(nox ~ dis, data = Boston, col = "grey")  
lines(dis\_grid, preds$fit, lwd = 2, col = "red")  
title("Cubic Polynomial Fit")

|  |
| --- |
| Scatter plot between dis (x) and nox (y) is shown with the cubic polynomial fit overlayed. The nox values generally decrease rapidly for smaller values of dis and then level off. The cubic polynomial fit does an excellent job fitting the data. |

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

# create 10 separate plots  
MSEs <- rep(0, 10)  
sort\_index <- sort(Boston$dis, index.return = T)$ix  
for (d in 1:10){  
 # scatter plot => data  
 plot(x=Boston$dis, y=Boston$nox, type = "p", lty=1,col="grey",  
 xlab="Dis", ylab="Nox", ylim=c(0.3,0.9),  
 main = paste(c("degree = ", as.character(d)), collapse = ""))  
 # lines => fit  
 fit\_b <- lm(nox~poly(dis,d), data=Boston)  
 MSEs[d] <- sum((fit\_b$fitted.values-Boston$nox)^2)  
 lines(x=Boston$dis[sort\_index],  
 y=fit\_b$fitted.values[sort\_index],  
 type="l", lty=1, col="black")  
}

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

|  |
| --- |
| Ten scatter plots between dis (x) and nox (y) is shown with different polynomial models fit. |

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

par(mfrow = c(1,1))  
MSEs <- rep(NA, 10)  
ind <- sample(1:dim(Boston)[1], floor(0.9 \* dim(Boston)[1]), replace = F)  
train\_set <- Boston[ind,]  
test\_set <- Boston[-ind,]  
for (i in 1:10) {  
 fit <- lm(nox ~ poly(dis, i), data = train\_set)  
 MSEs[i] <- mean((predict(fit, newdata = test\_set) - test\_set$nox)^2)  
}  
plot(1:10, MSEs, type = 'l')  
abline(v = which.min(MSEs), col = "red")

|  |
| --- |
| A plot of MSE values by degree (1-10) is shown. The plot starts with higher MSE values for lower polynomials and then dips down, taking a minimum value at a degree of 8, before slightly increasing for higher values of degree. |

We see the minimum test error happens when degree equals to 8.

1. Fit a smoothing spline model to predict nox using dis using four degrees of freedom. How did you choose the knots? Plot the resulting fit.

# scatter plot => data  
plot(x=Boston$dis, y=Boston$nox, type = "p", lty=1,col="grey",  
 xlab="Dis", ylab="Nox", ylim=c(0.3,0.9),  
 main = paste(c("Smooth Spline Degree = ", as.character(4)), collapse = ""))  
# lines => fit  
sspline1 <- smooth.spline(Boston$dis, Boston$nox, df=4)  
lines(sspline1[[1]], sspline1[[2]])

|  |
| --- |
| Scatter plot between dis (x) and nox (y) is shown with the smoothing spline fit overlayed. The nox values generally decrease rapidly for smaller values of dis and then level off. The cubic polynomial fit does an excellent job fitting the data. |

The knots were chosen by selecting the degrees of freedom and using the defaults from the function. From the help for smooth.spline(): If spar and lambda are missing or NULL, the value of df is used to determine the degree of smoothing. The knots used (scaled into ) are available in sspline1$fit$knot. In the notes for the function it is stated: > In this case where not all unique x values are used as knots, the result is not a smoothing spline in the strict sense, but very close unless a small smoothing parameter (or large df) is used.

1. Now fit a smoothing spline for a range of degrees of freedom, and plot the resulting fits and report the resulting RSS. Describe the results obtained.

df\_range = c(4:9)  
MSEs <- rep(0, length(df\_range))  
sort\_index <- sort(Boston$dis, index.return = T)$ix  
for (d in df\_range){  
 # scatter plot => data  
 plot(x=Boston$dis, y=Boston$nox, type = "p", lty=1,col="grey",  
 xlab="Dis", ylab="Nox", ylim=c(0.3,0.9),  
 main = paste(c("degree = ", as.character(d)), collapse = ""))  
 fit\_b <- lm(nox~bs(dis, df = d), data=Boston)  
 MSEs[d-3] <- sum((fit\_b$fitted.values-Boston$nox)^2)  
 lines(x=Boston$dis[sort\_index],  
 y=fit\_b$fitted.values[sort\_index],  
 type="l", lty=1, col="black")  
}

|  |
| --- |
| Six scatter plots are shown with smoothing splines fit with degrees of freedom 4, 5, …, 9. The models become more complex as the degree increases. |

|  |
| --- |
| Six scatter plots are shown with smoothing splines fit with degrees of freedom 4, 5, …, 9. The models become more complex as the degree increases. |

|  |
| --- |
| Six scatter plots are shown with smoothing splines fit with degrees of freedom 4, 5, …, 9. The models become more complex as the degree increases. |

|  |
| --- |
| Six scatter plots are shown with smoothing splines fit with degrees of freedom 4, 5, …, 9. The models become more complex as the degree increases. |

|  |
| --- |
| Six scatter plots are shown with smoothing splines fit with degrees of freedom 4, 5, …, 9. The models become more complex as the degree increases. |

|  |
| --- |
| Six scatter plots are shown with smoothing splines fit with degrees of freedom 4, 5, …, 9. The models become more complex as the degree increases. |

round(MSEs, 3)

[1] 1.923 1.840 1.834 1.830 1.817 1.826

After setting the range of df from 4-9, df is equal to 8 if we want to use the minimum MSE, but there isn’t a large difference in fit.

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

# train/test tuning for degree of freedom  
MSEs <- rep(NA, length(df\_range))  
for (i in df\_range) {  
 fit <- lm(nox ~ bs(dis, df = i), data = train\_set)  
 MSEs[i - 3] <- mean((predict(fit, newdata = test\_set) - test\_set$nox)^2)  
}  
plot(df\_range, MSEs, type = 'l')  
abline(v = which.min(MSEs) + 3, col = "red")

|  |
| --- |
| A plot of MSE values by degree (4-9) is shown. The plot starts with higher MSE value for df of 4 takes a minimum at df of 5 and then increases as the df increases. |

The best df is equal to 5 when cv is being performed

1. This question relates to the College data set in the ISLR2 library.
2. 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.

# split the data  
data\_split <- initial\_split(College, prop=0.8)  
test\_set <- testing(data\_split)  
train\_set <- training(data\_split)  
# forward selection  
forward <- regsubsets(Outstate ~., data=train\_set, nvmax=17, method='forward')  
f\_summary <- summary(forward)  
metrics <- data.frame(aic = f\_summary$cp, bic = f\_summary$bic, adjR2 = f\_summary$adjr2)  
# coef for best model  
coef\_forward <- coef(forward,6) |>   
 as.data.frame()  
names(coef\_forward) <- c("estimate")  
coef\_forward |>  
 knitr::kable()

|  | estimate |
| --- | --- |
| (Intercept) | -3611.4569185 |
| PrivateYes | 2880.2459988 |
| Room.Board | 0.9648787 |
| PhD | 39.0106159 |
| perc.alumni | 45.6334519 |
| Expend | 0.2304768 |
| Grad.Rate | 25.8625466 |

# best model  
bestmod <- regsubsets(Outstate ~., data=train\_set, nvmax=6)

1. 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.

out <- gam(Outstate ~ Private +   
 s(Room.Board, df=3) +   
 s(PhD, df=3) +   
 s(perc.alumni,df=3) +   
 s(Expend,df=3) +   
 s(Grad.Rate, df=3),  
 data=train\_set)  
plot(out, se=TRUE, col="red")

|  |
| --- |
| Six plots are shown, one for each predictor in the model. The categorical predictor, private, shows box plots with very different locations and spreads for the no and yes groups. The other five plots show the polynomial fits for each predictor. These are smoothed lines in each case. |

|  |
| --- |
| Six plots are shown, one for each predictor in the model. The categorical predictor, private, shows box plots with very different locations and spreads for the no and yes groups. The other five plots show the polynomial fits for each predictor. These are smoothed lines in each case. |

|  |
| --- |
| Six plots are shown, one for each predictor in the model. The categorical predictor, private, shows box plots with very different locations and spreads for the no and yes groups. The other five plots show the polynomial fits for each predictor. These are smoothed lines in each case. |

|  |
| --- |
| Six plots are shown, one for each predictor in the model. The categorical predictor, private, shows box plots with very different locations and spreads for the no and yes groups. The other five plots show the polynomial fits for each predictor. These are smoothed lines in each case. |

|  |
| --- |
| Six plots are shown, one for each predictor in the model. The categorical predictor, private, shows box plots with very different locations and spreads for the no and yes groups. The other five plots show the polynomial fits for each predictor. These are smoothed lines in each case. |

|  |
| --- |
| Six plots are shown, one for each predictor in the model. The categorical predictor, private, shows box plots with very different locations and spreads for the no and yes groups. The other five plots show the polynomial fits for each predictor. These are smoothed lines in each case. |

There are 6 features was being selected by previous step. After fitting them in gam model and setting the df of 3, we can see that they are perform well with the response.

1. Evaluate the model obtained on the test set, and explain the results obtained.

predGAM <- predict(out, test\_set)  
GAM\_teste <- mean((test\_set$Outstate - predGAM)^2)  
predlm <- predict(lm(Outstate~Private + Room.Board + PhD + perc.alumni + Expend + Grad.Rate,   
 data = train\_set),  
 test\_set)  
LM\_teste <- mean((test\_set$Outstate - predlm)^2)  
c(GAM\_teste, LM\_teste)

[1] 4120704 5023464

We found gam model has a better performance since it has a smaller test error compared with the linear model.

1. For which variables, if any, is there evidence of a non-linear relationship with the response?

out\_sum <- summary(out)  
#ANOVA for non-parametric effects  
out\_sum$anova |>  
 knitr::kable()

|  | Npar Df | Npar F | Pr(F) |
| --- | --- | --- | --- |
| (Intercept) | NA | NA | NA |
| Private | NA | NA | NA |
| s(Room.Board, df = 3) | 2 | 2.798824 | 0.0616706 |
| s(PhD, df = 3) | 2 | 1.595671 | 0.2036237 |
| s(perc.alumni, df = 3) | 2 | 1.892964 | 0.1515213 |
| s(Expend, df = 3) | 2 | 42.711355 | 0.0000000 |
| s(Grad.Rate, df = 3) | 2 | 2.869417 | 0.0575055 |

#ANOVA for parametric effects  
out\_sum$parametric.anova |>  
 knitr::kable()

|  | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
| --- | --- | --- | --- | --- | --- |
| Private | 1.0000 | 2843502296 | 2843502296 | 843.04050 | 0e+00 |
| s(Room.Board, df = 3) | 1.0000 | 1966382232 | 1966382232 | 582.99227 | 0e+00 |
| s(PhD, df = 3) | 1.0000 | 765024004 | 765024004 | 226.81403 | 0e+00 |
| s(perc.alumni, df = 3) | 1.0000 | 363453559 | 363453559 | 107.75658 | 0e+00 |
| s(Expend, df = 3) | 1.0000 | 760247970 | 760247970 | 225.39804 | 0e+00 |
| s(Grad.Rate, df = 3) | 1.0000 | 89846945 | 89846945 | 26.63779 | 3e-07 |
| Residuals | 603.9999 | 2037239009 | 3372913 | NA | NA |

For Nonparametric effects, we noticed that all six predictors have non-linear relationships with the response