Introduction to Statistical Learning 7/18/2017

Chapter 7: Nonlinear Methods

Name: roninlaw

Email: roninlaw

1

(a)

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

(b)

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

(c)

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

(d)

$$\begin{aligned}
f_1'(x) &= \beta_1 + 2\beta_2 x + 3\beta_3 x^2 \\
f_1'(\xi) &= \beta_1 + 2\beta_2 \xi + 3\beta_3 \xi^2 \\
f_2'(x) &= (\beta_1 + 3\beta_4 \xi^2) + 2(\beta_2 - 3\beta_4 \xi) x + 3(\beta_3 + \beta_4) x^2 \\
f_2'(\xi) &= \beta_1 + 3\beta_4 \xi^2 + 2\beta_2 \xi - 6\beta_4 \xi^2 + 3\beta_3 \xi^2 + 3\beta_4 x^2 \\
f_2'(\xi) &= \beta_1 + 2\beta_2 \xi + 3\beta_3 \xi^2
\end{aligned}$$

(e)

$$\begin{aligned}
f_1''(x) &= 2\beta_2 + 6\beta_3 x \\
f_1''(\xi) &= 2\beta_2 + 6\beta_3 \xi \\
f_2''(x) &= 2\beta_2 - 6\beta_4 \xi + 6\beta_3 x + 6\beta_4 x \\
f_2''(\xi) &= 2\beta_2 - 6\beta_4 \xi + 6\beta_3 \xi + 6\beta_4 \xi \\
&= 2\beta_2 + 6\beta_3 \xi
\end{aligned}$$

3

```
# Plot function
x = -2:2
y = 1 + x - 2 * (x - 1)^2* I(x >= 1)
plot(x,y)
```

4

```
# Plot function
x = -2:2
y = c(1, 1, 2, 2, 1) # plug x values into function
plot(x,y)
```

6

(a)

```
# Cross Validation to determine degree of polynomial
# LOOCV
library(boot)
for (i in 1:10) {
   glm.fit = glm(wage ~ poly(age, i), data = Wage)
   print(cv.glm(Wage, glm.fit)$delta[2]) # d = 6 has the lowest LOOCV value
}

# ANOVA Analysis
fit.1 = lm(wage ~ age, data = Wage)
fit.i = lm(wage ~ poly(age, i), data = Wage) # for i in [2, ..., 10]
anova(fit.1, fit.2, fit.3, fit.4, fit.5, fit.6, fit.7, fit.8, fit.9, fit.10)

# Fit polynomial regression
agelims = range(age)
age.grid = seq(from = agelims[1], to = agelims[2])
preds = predict(fit.6, newdata = list(age = age.grid), type = "response", se = TRUE)
par(mfrow = c(1, 2), mar = c(4.5,4.5,1,1), oma = c(0,0,4,0))
plot(age, wage, xlim = agelims, cex = .5, col = "darkgrey")
title("Degree-6 Polynomial", outer = T)
lines(age.grid, preds$fit, lwd = 2, col = "blue")
```

The ANOVA analysis indicates that there is no need to use a polynomial with degree > 3 to fit the data.

(b)

```
# Cross Validation to determine # of cuts
for (i in 1:10) {
  glm.fit = glm(wage ~ cut(age, i), data = Wage)
  print(i)
  print(cv.glm(Wage, glm.fit)$delta[2])
}

# 7 Knot Step Function
glm.fit = glm(wage ~ cut(age,7))
preds = predict(glm.fit, newdata = list(age = age.grid), se = TRUE)
par(mfrow = c(1,2), mar = c(4.5, 4.5, 1, 1), oma = c(0,0,4,0))
plot(age, wage, xlim = agelims, cex = .5, col = "darkgrey")
title("7 Knot Step Function")
lines(age.grid, preds$fit, lwd = 2, col = "green")
```

7

```r
# Feature Analysis
summary(race)
summary(jobclass)
summary(maritl)
plot(maritl, wage)
plot(race, wage)
plot(jobclass, wage)

# Models
fit.1 = glm(wage ~ race, data = Wage)
fit.2 = glm(wage ~ jobclass, data = Wage)
fit.3 = glm(wage ~ race + jobclass, data = Wage)
fit.4 = glm(wage ~ race + jobclass + poly(age, 3))
```

8

```r
# Imports
library(splines)

# Feature Analysis
summary(cylinders) # slightly skewed right
summary(acceleration)
summary(displacement) # skewed right
summary(horsepower)
cor(subset(Auto, select = - name))

# Variables of Interest plotted against mpg
plot(cylinders, mpg)
plot(displacement, mpg)
plot(horsepower, mpg)

# Displacement degree Analysis (polynomial)
dfit.1 = lm(mpg ~ poly(displacement, 1))
dfit.i = lm(mpg ~ poly(displacement, i)) # for i in [2, ..., 10]
anova(dfit.1, dfit.2, dfit.3, dfit.4, dfit.5,dfit.6, dfit.7, dfit.8, dfit.9, dfit.10)

# Displacement df Analysis (spline)
for (df in 1:20) {
  fit = glm(mpg ~ ns(displacement, df = df))
  print(cv.glm(Auto, fit)$delta[2] )
}

# Cylinder degree Analysis (spline)
ss1 = gam(mpg ~ s(cylinders, 1))
ssi = gam(mpg ~ s(cylinders, i)) # # for i in [2, ..., 10]
anova(ss1, ss2, ss3, ss4, ss5, ss6, ss7, ss8, ss9, ss10)

# Models
fit.1 = lm(mpg ~ poly(displacement, 3)) # deviance = 8378.822
fit.2 = lm(mpg ~ cylinders + poly(displacement, 3)) # deviance = 7412.263
fit.3 = smooth.spline(displacement, mpg) #
fit.4 = glm(mpg ~ ns(displacement, df = 12)) # deviance = 6318.941
fit.5 = gam(mpg ~ s(cylinders, 4)) # deviance = 8544.487
fit.6 = gam(mpg ~ year + s(cylinders, 4) + poly(displacement, 3)) # deviance = 3898.179
```

9

(a)

```
# Cubic Polynomial Regression of nox on dis...
Boston = read.csv("BostonHousing.csv", header = T, na.strings = "?")
attach(Boston)
fit.1 = lm(nox ~ poly(dis, 3))
summary(fit.1) # B_0 = .55, B_1 = -2.00, B_2 = .856, and B_3 = -.31
dislims = range(dis)
dis.grid = seq(from = dislims[1], to = dislims[2])
preds = predict(fit.1, newdata = list(dis = dis.grid), type = "response", se = TRUE)
par(mfrow = c(1,2))
plot(dis, nox, xlim = dislims, cex = .5, color = "darkgrey")
lines(dis.grid, preds$fit, lwd = 2, col = "blue")
```

(b)

```
# Different degrees for Regression of nox on dis...
par(mfrow = c(1,2))
plot(dis, nox, xlim = dislims, cex = .5, color = "darkgrey")
colarr = c("burlywood4", "gold", "red", "green", "darkviolet", "mediumvioletred",
    "steelblue1", "skyblue", "sienna1", "thistle")
resids = rep("NA",10)
for (i in 1:10) {
  fit = lm(nox ~ poly(dis, i), data = Boston)
  preds = predict(fit, newdata = list(dis = dis.grid), type = "response", se = TRUE)
  resids[i] = sum(fit$residuals^2)
  lines(dis.grid, preds$fit, lwd = 2, col = colarr[i])
}
print(resids)
```

(c)

```
# Cross Validation to see best polynomial degree...
library(boot)
cv.errs = rep("NA", 10)
for (i in 1:10) {
  fit = glm(nox ~ poly(dis, i))
  cv.errs[i] = cv.glm(Boston, fit)$delta[2]
}
which.min(cv.errs) # d = 3
```

(d)

```
# Spline Regression w/ df = 4...
sfit = lm(nox ~ bs(dis, df = 4))
plot(dis, nox, xlim = dislims, cex = .5, col = "darkgrey")
preds = predict(sfit, newdata = list(dis = dis.grid), type = "response", se = TRUE)
lines(dis.grid, preds$fit, lwd = 2, col = "blue")
```

(e)

```
# Spline Regression for varying degrees of freedom...
resids = rep("NA",10)
par(mfrow = c(1,2))
plot(dis, nox, xlim = dislims, cex = .5, color = "darkgrey")
colarr = c("burlywood4", "gold", "red", "green", "darkviolet", "mediumvioletred",
    "steelblue1", "skyblue", "sienna1", "thistle")
resids = rep("NA",10)
for (i in 1:10) {
```

```
  sfit = glm(nox ~ bs(dis, df = i))
  preds = predict(sfit, newdata = list(dis = dis.grid), type = "response", se = TRUE)
  resids[i] = sum(fit$residuals^2)
  lines(dis.grid, preds$fit, lwd = 2, col = colarr[i])
}
```

The resulting lines seem much more closer to the other lines, than the resulting lines for the polynomial regression. The resulting RSS values also seem to be much closer than they were for the polynomial regression.

(f)

```
# Cross Validation to see best df value...
cv.errs = rep("NA", 10)
for (i in 1:10) {
  sfit = glm(nox ~ bs(dis, df = i))
  preds = predict(sfit, newdata = list(dis = dis.grid), type = "response", se = TRUE)
  cv.errs[i] = cv.glm(Boston, sfit)$delta[2]
  lines(dis.grid, preds$fit, lwd = 2, col = colarr[i])
}
which.min(cv.errs) # df = 10 was the best value
```

10

(a)

```
# Factor selection
library(ISLR)
library(leaps)
train = 1:(dim(College)[1] / 2)
test = (dim(College)[1] / 2) : dim(College)[1]
College.Train = College[train,]
College.Test = College[test,]
names(College)
regfit.fwd = regsubsets(Outstate ~ ., data = College.Train, nvmax = 19, method =
    "forward")
regsummary = summary(regfit.fwd)
which.min(regsummary$bic) # 9
# Features Selected: PrivateYes, Room.Board, Expend, Accept, Enroll, Personal, Terminal,
    Grad.Rate, perc.alumni
```

(b)

```
# Model Creation

# Basic EDA
summary(Private) # No: 212, Yes: 565 (significantly more Yes than No)
summary(Room.Board)
summary(Expend)
plot(Room.Board, Outstate) # direct relationship between Room.Board and Outstate
plot(Private, Outstate) # Private schools have a higher Outstate value on average
plot(Expend, Outstate) # direct relationship between expenditure price and Outstate
hist(Expend) # data is heavily skewed right
hist(Room.Board) # unimodal distribution
plot(density(Accept)) # skewed right (better schools let in less people....seems
    intuitive)
plot(density(Expend)) # heavily skewed right (seems intuitive)
plot(density(Enroll)) # heavily skewed right (seems intuitive)
plot(density(Personal)) # heavily skewed right (seems intuitive)
```

```
plot(density(Terminal)) # heavily skewed left (most professors have to get PhDs...seems
    intuitive)
plot(density(Grad.Rate)) # data is unimodal and centered
plot(density(perc.alumni)) # slightly skewed right (not that intuitive)


# Degree selection for Room.Board Polynomial
attach(College)
cv.errs = rep("NA", 10)
for (i in 1:10) {
  fit = glm(Outstate ~ poly(Room.Board, i))
  cv.errs[i] = cv.glm(College, fit)$delta[2]
}
which.min(cv.errs) # i = 2

# Degree selection for Expend Polynomial
which.min(cv.errs) # i = 3

# df selection for Expend Spline
cv.errs = rep("NA", 10)
for (i in 1:10) {
  fit = glm(Outstate ~ bs(Expend, df = i ))
  cv.errs[i] = cv.glm(College, fit, K = 10)$delta[2]
}
which.min(cv.errs) # i = 5

# df selection for Accept Spline
which.min(cv.errs) # i = 9

# df selection for Enroll Spline
which.min(cv.errs) # i = 5

# df selection for Personal Spline
which.min(cv.errs) # i = 3

# df selection for Terminal Spline
which.min(cv.errs) # i = 6

# df selection for Grad.Rate Spline
which.min(cv.errs) # i = 4

# df selection for perc.alumni
which.min(cv.errs) # i = 2

# Final Model
model = gam(Outstate ~ Private + Expend + Enroll + Personal + Grad.Rate + perc.alumni +
    Terminal + bs(Accept, 9), data = College.Train)
# did not use splines for features with low degrees of freedom (df <= 6)

# Plot Model
par(mfrow = c(3,3))
plot(model, se = T, col = "green")
```

(c)

```
# Model Predictions
preds = predict(model, newdata = College.Test, type = "response", se = TRUE)
```

```r
rss1 = sum((College.Test$Outstate - preds$fit) ^ 2)
tss1 = sum((preds$fit - outstateMean)^2)
r = rss1 / tss1
r2 = 1 - r # .582
```

This is significantly better than simply using just linear regression, which gives a R^2 of .248

(d) The data suggests there is a non-linear relationship between Accept and Outstate.

11

(a)

```r
# Generate Response
x1 = sample(1:10, 100, replace = TRUE)
x2 = sample(1:10, 100, replace = TRUE)
eps = rnorm(100, mean = 0, sd = 10)
y = 2*x1 - x2 + eps
```

(b)

```r
# Initialize Beta_1
a = y - (beta1*x1)
beta2 = lm(a ~ x2)$coef[2]
a = y - beta2*x2
beta1 = lm(a ~ x1)$coef[2]
```

(c)

```r
# Fit Model w/ Beta_1 Fixed
a = y - beta2*x2
beta1 = lm(a ~ x1)$coef[2]
```

(d)

```r
# Fit Model w/ Beta_2 Fixed
beta1 = 5
a = y - beta2*x2
beta1 = lm(a ~ x1)$coef[2]
```

(e)

```r
# Plot values
for (i in 1:1000){
  a = y - beta1[i]*x1
  beta2[i] = lm(a ~ x2)$coef[2]
  a = y - beta2[i]*x2
  model = lm(a ~ x1)

  if (i != 1) {
    beta1[i] = model$coef[2]
  }

  beta0[i] = model$coef[1]
}

plot(1:1000, beta0, type = "l", col = "red")
lines(1:1000, beta1, col = "green")
lines(1:1000, beta2, col = "blue")
```

(f)

```
# Multiple Linear Regression
fit = lm(y ~ x1 + x2 + eps)
fit$coefficients # [1.421085e-15, 2.000000e+00, -1.000000e+00 ]
abline(h = fit$coef[1])
abline(h = fit$coef[2])
abline(h = fit$coef[3])
```

(g) The coefficients in the model seemed to converge after just two iterations. This is most likely because the true relationship between the predictors was linear.