Problem 1

Solution:
We can easily compare the RMSE’s for these different models. See the code in Appendix A for details. Besides, anything you can do in R, you can find an analogous function/method in Matlab.

Problem 2 Ex3.2 on pg 94

Solution:

• A 95% CI for $a^T \beta = \sum_{j=0}^{3} \beta_j x_0^j$ is:
  $$\sum_{j=0}^{3} \hat{\beta}_j x_0^j \pm 1.96 \cdot \text{MSE}$$
  which depends on the number of data we have for the regression. Since $E(a^T \beta - a^T \hat{\beta}) = 0$, $\text{Var}(a^T \beta - a^T \hat{\beta}) = \text{MSE}(a^T \hat{\beta})$

• A 95% CI for $\beta$, as in (3.15), is $C_{\beta} = \{ \beta | (\hat{\beta} - \beta)^T X^T X (\hat{\beta} - \beta) \leq \hat{\sigma}^2 \chi^2_{p+1} (1 - \alpha) \}$, which generates a CI for the function $f(x_0) = x_0^T \beta$, namely $\{ x_0^T \beta | \beta \in C_{\beta} \}$. The upper and lower bounds depend not only on the predicted value but also on the specific value of $x_0$.

Problem 3

Solution:
Efron famously introduced the bootstrap by setting a confidence interval on the correlation coefficient between the average LSAT scores and the average GPAs for entering classes at a sample of law schools. Download the population data (use law82 in R, or go to our class website) and use the bootstrap to set the confidence interval for 5 random samples of size 15 at level 95%. Compare your results to Efron’s confidence interval (e.g., in the SIAM monograph) and comment.

Efron’s initial sample was surprisingly lucky. Under repeated draws from the population, there is only about a 7% chance of getting a sample for which the sample correlation is so close to the population correlation. Further, the confidence interval generated by Efron’s initial sample under the bootstrap finds a standard error very close to that appropriate for this population.

Problem 4

Solution:
Consider a nearest-neighbor regression algorithm in $\mathbb{R}^3$ that puts equal weight on the response for the second and third nearest $x$ values, but no weight upon the closest value. Describe the “smoothing” matrix, indicate why it is unusual, check the fine print about smoothing matrices, and diagnose the problem. Also, describe the implications of this procedure for leave-one-out cross-validation.
The smoothing matrix has zeros on the diagonal, and thus appears to have no degrees of freedom. But smoothing matrices must have all eigenvalues in the open interval $(0,1)$, and this does not (since the trace is zero, then the eigenvalues must be zero or have some negative values). In fact, this matrix is not at all a smoother—it is extremely unsmooth.

For loo cross-validation, the full data estimate for a response will be the same as the loo estimate. Thus cross-validation provides no additional information about the predictive error.
A: R code for problem 1

# Problem 1:
### Read in data
ozonedata <- read.table("ozone.dat",header=T)
attach(ozonedata)
### 1. multiple linear regression
lm1 <- lm(ozone~., data=ozonedata)
ynew <- predict(lm1, newdata=ozonedata)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))
# [1] 4.3732
summary(lm1)
#Residual standard error: 4.441 on 320 degrees of freedom

### 2. a stepwise linear regression model
step1 <- step(lm1)
summary(step1)
ynew <- predict(step1, newdata=ozonedata)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))
# [1] 4.389188

#Residual standard error: 4.43 on 324 degrees of freedom
#Step: AIC= 988.24
#ozone ~ humid + temp + ibt + vis + doy

### 3. additive model
library(mgcv)
am1 <- gam(ozone~s(vh)+s(wind)+s(humid)+s(temp)+s(ibh)+s(dpg)+s(ibt)+s(vis)+s(doy))
ynew <- predict(am1, newdata=ozonedata)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))
# [1] 3.469863

### 4. generalized additive model
library(gam)
tozone <- ace1$ty;
gam1 <- gam(tozone~s(vh)+s(wind)+s(humid)+s(temp)+s(ibh)+s(dpg)+s(ibt)+s(vis)+s(doy),data=ozonedata,fam=gaussian)
ynew <- predict(gam1, newdata=ozonedata)
sqrt(sum((ynew-tozone)^2)/(nrow(ozonedata)))
# [1] 0.3919104

#Note HERE!
#for the generalized additive model, we get a quite small RMSE, which
#is not comparable to the other models since we don’t have an inverse
#function for the transformation of the ozone level. You can check
#back to the results of the ACE models to figure out the link function.:)
#
#
### 5. a projection pursuit model
ppr1 <- ppr(ozone~., data=ozonedata, nterms=3)
ynew <- predict(ppr1, newdata=ozonedata)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))
## [1] 3.551325

## 6. a neural network model
require(nnet)
nn1 <- nnet(ozone~., data=ozonedata, size=2, rang=0.1, decay=0.5, maxit=200)
ynew <- predict(nn1, newdata=ozonedata)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))
####### [1] 13.42055

## 7. an ACE (Alternating Conditional Expectation) model
## NOTE: David has given some ACE code in class, you can figure out
## how to do this part by your own
require(acepack)
help(ace)
#ace1 <- ace(ozone~., data=ozonedata)
x <- ozonedata[, 2:10]
y <- ozonedata$ozone;
ace1 <- ace(x, y)
summary(ace1)
par(mfrow=c(3,1))
plot(ace1$y, ace1$ty) # view the response transformation
#plot(ace1$x, ace1$tx) # view the carrier transformation
#persp(ace1$tx, ace1$ty) # examine the linearity of the fitted model
#ynew <- predict(ace1, x)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))

## 8. AVAS model
#Additivity and variance stabilization for regression
help(avas)
avas1 <- avas(x, y)

## 9. a regression tree model
require(tree)
tree1 <- tree(ozone~., data=ozonedata)
ynew <- predict(tree1, newdata=ozonedata)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))
## [1] 3.817251

## 10. a MARS model
###### Multivariate Adaptive Regression Splines
require(mda)
help(mars)
mars1 <- mars(x, y)
ynew <- predict(mars1, x)
sqrt(sum((ynew-ozone)^2)/(nrow(ozonedata)))
# [1] 3.560219

B: R code for problem 2