Homework 2 Key

# Conceptual Problems

## Book Problems

Complete the following problems from the Introduction to Statistical Learning with R book (I’m not sure if the problems are in the same order in the python book so use the R book to identify which problems to do).

Section 3.7

* Book Problem 3
* Our fitted model is given by
  1. For those with college our fitted line is
  + For those with high school our fitted line is
  + iii is the answer as the high school predicted values can be larger, given that the GPA is high enough.
  1. College graduate, IQ = 110, and GPA = 4.0 yields
  + ::: {.cell}
  + 50 + 20\*4 + 0.07\*110 + 35+0.01\*4\*110-10\*4
  + ::: {.cell-output .cell-output-stdout}
  + [1] 137.1
  + ::: :::
  1. False, we’d need to look at a hypothesis test or confidence interval to determine this. Without knowing the standard error or other information, we can’t determine whether the effect is significant or not.
* Book Problem 4
  1. We’d expect the training RSS to be smaller for the cubic model as the RSS always decreases with the addition of more predictors.
  2. As the true relationship is linear, we’d expect the test MSE for the linear model to be smaller as the cubic model would be overfitting to the data and should not generalize as well.
  3. Same answer as a.
  4. We don’t have enough information to determine this. It would depend on how well the linear function approximates the true function vs how well the cubic function does.

Section 6.6

* Book Problem 1
  1. The best subset selection model as that method would look include looking at the forward and backward models.
  2. It isn’t clear. We can’t say that one method will produce the best model in terms of prediction error.
  3. True/False
     1. True
     2. True
     3. False
     4. False
     5. False
* Book Problem 2
  1. For the LASSO, iii is the best choice. The LASSO introduces bias in a hope for a reduction in variance. As it shrinks the estimates towards 0 generally, this would mean it is less flexible.
  2. For RR, iii is the best choice with the same reasoning as a.
  3. For non-linear methods, ii is the best choice. They allow for even less bias towards the true f but can introduce more variance.
* Book Problem 3
  1. iv, as the training RSS will steadily decrease. We know the training RSS is the same function that least squares minimizes.
  2. ii, the test RSS will generally be optimized for some s in the range of 0 to the sum of the absolute value of the OLS solutions.
  3. iii, the variance of the predictions will generally get larger as the model increases its flexibility.
  4. iv, the squared bias will decrease as we get closer to the OLS solution
  5. v, the irreducibile error doesn’t change
* Book Problem 4
  1. iii, as the training RSS will steadily increase as we penalize more
  2. ii, the test RSS will generally be optimized for some larger than 0 but will get worse as we penalize too much
  3. iv, the variance of the predictions will generally get less and less the more we penalize the size of the coefficients
  4. iii, the squared bias will increase the further we get from the OLS solution
  5. v, the irreducibile error doesn’t change

# Implementation Problems

## Problem 1

### Fitting an Elastic Net Model

For this problem we’ll fit an Elastic net model to our bike sharing data set. The code below reads the data in and does the modifications from the notes:

library(tidyverse)  
library(lubridate)  
#read in data from previous set of notes  
bike\_share <- read\_csv("https://www4.stat.ncsu.edu/online/datasets/SeoulBikeData.csv",  
 local = locale(encoding = "latin1"))  
  
bike\_share <- bike\_share |>  
 rename("date" = "Date",  
 "rented\_bike\_count" = `Rented Bike Count`,  
 "hour" = "Hour",  
 "temperature" = `Temperature(°C)`,  
 "humidity" = `Humidity(%)`,  
 "wind\_speed" = `Wind speed (m/s)`,  
 "visibility" = `Visibility (10m)`,  
 "dew\_point\_temperature" = `Dew point temperature(°C)`,  
 "solar\_radiation" = `Solar Radiation (MJ/m2)`,  
 "rainfall" = `Rainfall(mm)`,  
 "snowfall" = `Snowfall (cm)`,  
 "seasons" = "Seasons",  
 "holiday" = "Holiday",  
 "functioning\_day" = "Functioning Day"   
 ) |>  
 mutate(date = dmy(date), #convert the date variable from character  
 seasons = factor(seasons),  
 holiday = factor(holiday),  
 functioning\_day = factor(functioning\_day),  
 log\_rented\_bike\_count = log(rented\_bike\_count)) |>  
 filter(functioning\_day == "Yes")

1. First, split the data into a train/test set.

library(caret)

Warning: package 'caret' was built under R version 4.1.3

Loading required package: lattice

Attaching package: 'caret'

The following object is masked from 'package:purrr':  
  
 lift

set.seed(8)  
split <- createDataPartition(bike\_share$log\_rented\_bike\_count,   
 p = 0.8,  
 list = FALSE)  
bike\_train <- bike\_share[split, ]  
bike\_test <- bike\_share[-split,]

1. Create your own grid of tuning parameter values (in R the expand.grid() function can be useful for creating a data frame at every combination of two vectors), use 5 fold CV, preprocess your numeric predictors, and select your optimal tuning parameters on the training set.

tc <- trainControl(method = "CV",  
 number = 5)  
#did a quick fit with a few values of alpha and lambda  
#used that to narrow it down to this grid  
tuning\_grid <- expand.grid(alpha = seq(from = 0, to =1, length = 15),  
 lambda = seq(from = 0, to = 0.5, length = 15))  
  
#train the model!  
en\_fit <- train(log\_rented\_bike\_count ~ hour +   
 temperature + humidity + wind\_speed +  
 dew\_point\_temperature + solar\_radiation +  
 rainfall + snowfall + seasons,   
 data = bike\_train,  
 method = "glmnet",  
 preProc = c('center', 'scale'),  
 tuneGrid = tuning\_grid,  
 trControl = tc  
 )  
#best parameters  
en\_fit$bestTune

alpha lambda  
16 0.07142857 0

en\_fit$bestTune |>  
 knitr::kable()

|  | alpha | lambda |
| --- | --- | --- |
| 16 | 0.0714286 | 0 |

1. Determine the RMSE on the test data set using your best model.

preds <- en\_fit |>   
 predict(newdata = bike\_test)  
metrics <- postResample(preds, obs = bike\_test$log\_rented\_bike\_count)  
metrics

RMSE Rsquared MAE   
0.7186819 0.6033162 0.5202948

data.frame(value = metrics) |>  
 knitr::kable()

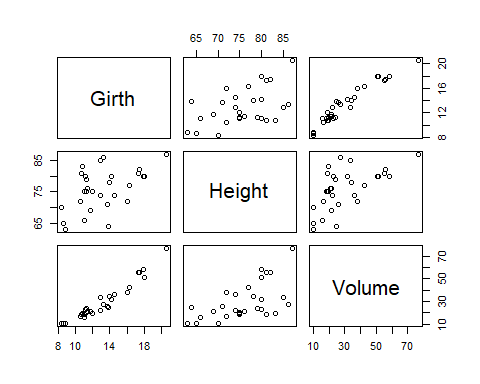
|  | value |
| --- | --- |
| RMSE | 0.7186819 |
| Rsquared | 0.6033162 |
| MAE | 0.5202948 |

## Problem 2

The datasets package in R has a data set called trees (datasets::trees in R). The data is also available at <https://www4.stat.ncsu.edu/online/datasets/trees.csv> if you are using another software.

1. First, create scatter plots between the predictors and the response (two scatter plots).

#baseR pairs function summarizes small datasets well  
pairs(trees)



Now we want to fit a couple of linear regression models to predict Height of the trees.

1. Fit a model with main effects for both Girth and Volume. Are either of these predictors statistically significant? Report the relevant hypothesis tests, p-values, and interpretations. Are the assumptions of inference met (at least roughly) here? Provide evidence (plots!).

tree\_fit <- lm(Height ~ Girth + Volume, data = datasets::trees)  
#summary(tree\_fit)  
summary(tree\_fit)$coef |>   
 knitr::kable()

|  | Estimate | Std. Error | t value | Pr(>|t|) |
| --- | --- | --- | --- | --- |
| (Intercept) | 83.2957705 | 9.0865753 | 9.166905 | 0.0000000 |
| Girth | -1.8615109 | 1.1566879 | -1.609346 | 0.1187591 |
| Volume | 0.5755946 | 0.2208225 | 2.606594 | 0.0144910 |

f\_info <- summary(tree\_fit)$fstatistic  
f\_info |>  
 knitr::kable()

|  | x |
| --- | --- |
| value | 9.820371 |
| numdf | 2.000000 |
| dendf | 28.000000 |

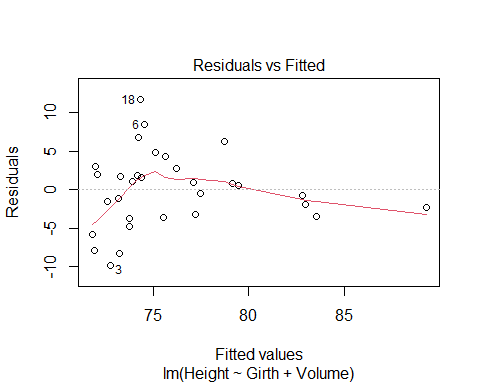
#pvalue manually  
1-pf(f\_info[1], df1 = f\_info[2], df2 = f\_info[3])

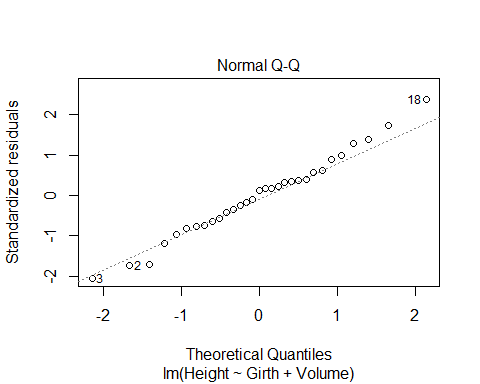
value   
0.0005868313

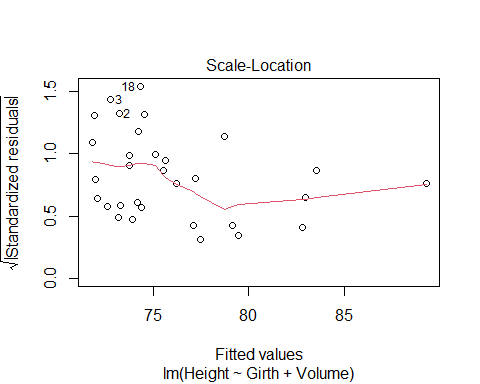
The F-stat is significant (9.82, p-value = 0.0005868, tests if both slopes are zero vs at least one differs) so the model is useful. The Volume variable has a significant p-value (0.0145, tests if that slope is 0 vs not 0) but the Girth variable does not (0.1188, tests if that slope is 0 vs not 0).

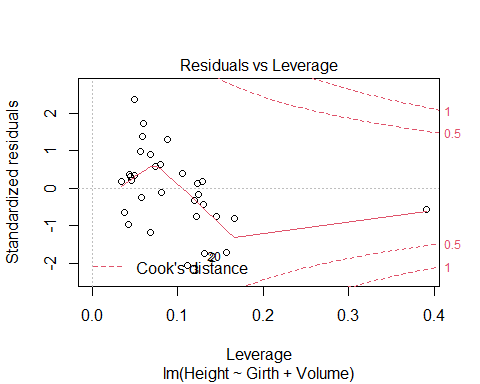
To check assumptions we can run:

plot(tree\_fit)









Looking at these we see that there is a slight pattern in the fitted vs residuals chart but nothing too extreme. The qqplot looks great so the normality assumptions seems reasonable. No points seem to have extreme leverage or appears to be an outlier.

1. Produce a predicted height for a tree with a Girth of 10 and a Volume of 12.

predict(tree\_fit, newdata = data.frame(Volume = 12, Girth = 10))

1   
71.5878

1. Fit a model with main effects for both Girth and Volume along with their interaction. Are either of these predictors statistically significant? Report the relevant hypothesis tests, p-values, and interpretations. Are the assumptions of inference met (at least roughly) here? Can you explain any discrepancies in the importance of the predictors in this model as compared to the first model we fit?

tree\_fit\_int <- lm(Height ~ Girth\*Volume, data = datasets::trees)  
#summary(tree\_fit\_int)  
summary(tree\_fit\_int)$coef

Estimate Std. Error t value Pr(>|t|)  
(Intercept) 75.40148460 8.49146607 8.879678 1.703513e-09  
Girth -2.29632359 1.03600722 -2.216513 3.526965e-02  
Volume 1.86094656 0.47932389 3.882441 6.035269e-04  
Girth:Volume -0.05608418 0.01909084 -2.937753 6.689120e-03

f\_info\_int <- summary(tree\_fit\_int)$fstatistic  
f\_info\_int |>  
 knitr::kable()

|  | x |
| --- | --- |
| value | 11.20784 |
| numdf | 3.00000 |
| dendf | 27.00000 |

#pvalue manually  
1-pf(f\_info\_int[1], df1 = f\_info\_int[2], df2 = f\_info\_int[3])

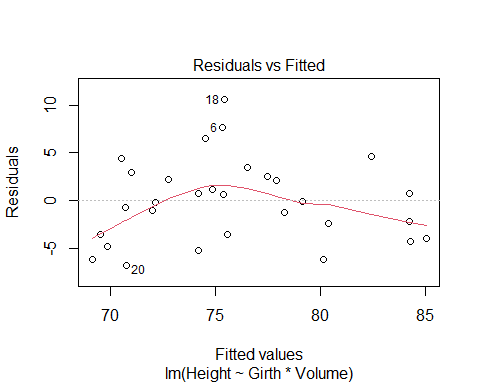
value   
5.898362e-05

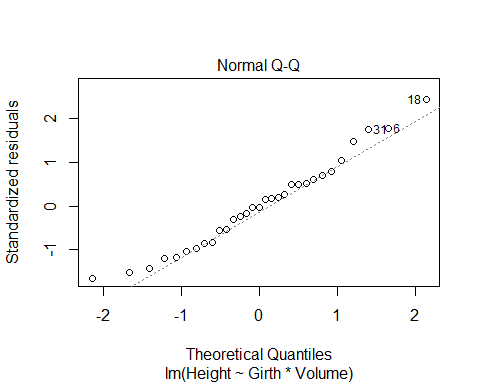
The global F-test is again significant (F = 11.21, p-value very small, tests if the slopes are all zero vs at least one differs). Now we see that every predictor is significant (we reject the individual tests that each slope is 0 in favor of the alternative that they are non-zero).

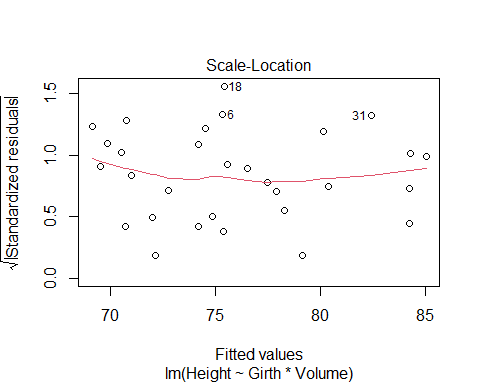
This in some ways contradicts the previous model… However, recall that these are tests of whether that predictor is important given the other predictors are in the model. In this case, Girth is important given that the interaction between Girth and Volume is in the model. This is important! We now know that each predictor is important and the effect of the variable on the tree height depends on the other predictor variable’s value!

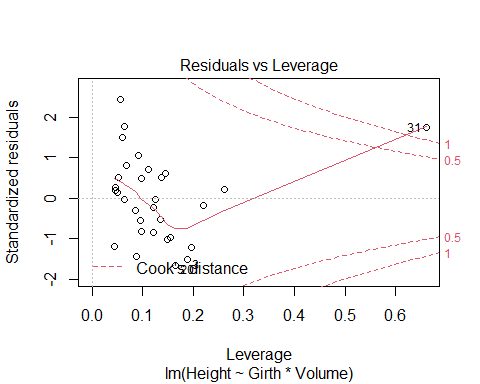
Checking assumptions:

plot(tree\_fit\_int)









We see very little pattern in the residuals vs fitted plot. Normality still looks great in the qqplot. There is one point that seems to have high leverage and is affecting the model fit a bit. That point might be interesting to look at further if we knew more about the data.

1. Produce a predicted height for a tree with a Girth of 10 and a Volume of 12.

predict(tree\_fit\_int, newdata = data.frame(Girth = 10, Volume = 12))

1   
68.03951

1. Fit two different models. First a model with just Volume as a predictor. Second a model with Volume, Volume squared, and Volume cubed. Recall the I() function if you are using R. Perform the F-test discussed in the notes to determine if the squared and cubic terms are important for the model.

linear <- lm(Height ~ Volume, data = datasets::trees)  
cubic <- lm(Height ~ Volume + I(Volume^2) + I(Volume^3),   
 data = datasets::trees)  
anova(linear, cubic) |>  
 knitr::kable()

| Res.Df | RSS | Df | Sum of Sq | F | Pr(>F) |
| --- | --- | --- | --- | --- | --- |
| 29 | 782.0746 | NA | NA | NA | NA |
| 27 | 658.6541 | 2 | 123.4204 | 2.529668 | 0.0984068 |

Here we see a marginally significant p-value (0.09841). This indicates that the cubic model may be useful above and beyond the simple linear model.

## Problem 3

Using the Carseats data from the ISLR2 package, split the data into a training and a test set (80/20). Then use best subset, backward, and forward selection to determine the optimal model size for each type of model (similar to how we did this in the notes, no need to use CV here). Once you’ve determined the best size for each model, fit that to the entire data set.

I’ll just give code for the best subset selection method. The forward and backward are very similar.

set.seed(101)  
library(ISLR2)

Warning: package 'ISLR2' was built under R version 4.1.3

library(leaps)

Warning: package 'leaps' was built under R version 4.1.3

## Create test and training sets  
data\_split <- createDataPartition(Carseats$Sales,   
 p = 0.8,   
 list = FALSE  
 )  
  
test\_set <- Carseats[-data\_split, ]  
train\_set <- Carseats[data\_split, ]  
  
## Best subset selection on the training data  
best\_train <- regsubsets(Sales ~ .,  
 data = train\_set,  
 nvmax = 11)  
  
train\_sum <- summary(best\_train)

* For each model size, estimate the test performance. Pull this function from our notes to help:

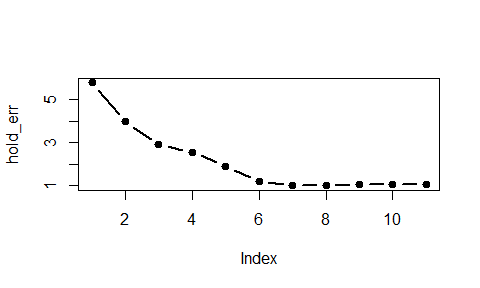
#We'll write a function to predict and estimate the error on the test set.   
#Inputs are   
#- model size (mod\_size),  
#- summary output of the selection process (reg\_summary)  
#- model matrix of the test data (test\_model)  
#- test set response (test\_resp)  
test\_err <- function(mod\_size,   
 reg\_summary,   
 test\_model,  
 test\_resp){  
 # get regression coefs  
 betahat <- coef(reg\_summary$obj, mod\_size)  
 # get best subset of the specified size  
 sub <- reg\_summary$which[mod\_size, ]  
 # Create test model matrix, prediction, test error  
 model <- test\_model[, sub]  
 yhat <- model %\*% betahat  
 err <- mean((test\_resp - yhat)^2)  
 return(err)  
}

* Apply the function to each model size

#define the test model  
test\_model <- model.matrix(~ CompPrice + Income + Advertising +   
 Population + Price + ShelveLoc +  
 Age + Education + Urban + US,  
 data = test\_set)  
  
#define the test response  
test\_resp <- test\_set$Sales  
  
#apply the function to each of the model sizes  
hold\_err <- sapply(1:11, #apply the function to these   
 FUN = test\_err,   
 reg\_summary = train\_sum,  
 test\_model = test\_model,   
 test\_resp = test\_resp)

* Now let’s plot the errors

plot(hold\_err, type = 'b', pch=19, lwd=2)



Holdout error as a function of model size

* Choose the optimal model size and use that model size on a model fit to the full data set

size\_opt <- which.min(hold\_err)  
size\_opt

[1] 7

#fit on the full data set  
bestmod <- regsubsets(Sales ~ .,  
 data = Carseats,  
 nvmax = 11)  
#Use the optimal size  
coef(bestmod, size\_opt) |>  
 round(3) |>  
 knitr::kable()

|  | x |
| --- | --- |
| (Intercept) | 5.475 |
| CompPrice | 0.093 |
| Income | 0.016 |
| Advertising | 0.116 |
| Price | -0.095 |
| ShelveLocGood | 4.836 |
| ShelveLocMedium | 1.952 |
| Age | -0.046 |