Homework 2 R markdown

Jessica Kraker

2021-08-11

Possible Solutions to Selected Questions

Important: This document contains R code solutions and example answers for problems posed in the DS740: Data Mining homework. We are intentionally sharing this document with learners *who have already completed the associated homework*. We want you to be able to review and troubleshoot your code, as well as ask questions to prepare you for later assessments. *By using this document, you are accepting responsibility* **not** *to share it with anyone else, including other students in the course or the program who might not have completed the homework yet*. By upholding this agreement, you are helping us use this tool with learners in future terms.

From Problem 1: Model Assessment

Question

Why is *Volume* the most reasonable **response** variable? *Include real-world reasons (eg. physical practicalities) in your discussion.*

Possible Answer: *Since Volume could only be measured after a tree was cut down and processed (while both Girth and Height can be measured prior to harvesting the tree), we wish to use the easier-to-obtain measures to predict Volume.*

A predictive model could also be useful in identifying trees that are cost-effective (lumber product versus cost).

Question

Use multiple linear regression fit the model to the full data set. Identify the coefficient estimates ($\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{\beta}_3$, $\hat{\beta}_4$, $\hat{\beta}_5$) for the five predictor terms.

How many of the five predictor terms are significant at the $\alpha = 0.10$ significance level?

Code for Possible Answer:

```
Trees <- read.csv("TreesTransformed.csv")  # retain for tables
n=dim(Trees)[1]
lmfit = lm(Volume ~ ., data=Trees)

summary(lmfit)$coefficients[,4] < 0.10
```

We now apply k-fold cross-validation to produce honest predictions, using the process outlined in the next several questions.

Question

Starting with:

```
groups = rep(1:5, length=31)
```

Set R's seed to 2:

```
set.seed(2)
```

and then define cvgroups (random groups for the cross-validation) using the `sample()` function.

With the above definition of cvgroups, use the 5-fold cross-validation method to produce honest predicted values. Provide the predicted-y value for the **first** and **second** observations, along with CV measure:

Code for Possible Answer:

```
groups = rep(1:5, length=n)
set.seed(2)
cvgroups = sample(groups,n)

# use 5-fold CV
allpredictedCV = rep(NA,n)
for (i in 1:5) {
  groupi = (cvgroups == i)
  lmfitCV = lm(formula = Volume ~ .,data=Trees[!groupi,])
  allpredictedCV[groupi] = predict.lm(lmfitCV,Trees[groupi,])
}

allpredictedCV
mean((allpredictedCV-Trees$Volume)^2)
```

We will now use the bootstrap to estimate variability of the coefficients.

Question

Program a function, making use of `lm()` to fit the linear regression model, that outputs the six coefficient estimates. Set R's seed to 2:

```
set.seed(2)
```

and then use `boot()` to produce R = 1000 bootstrap estimates for each of β_0, β_1, β_2, β_3, β_4, and β_5.
Enter your R code below. Use your bootstrap results to estimate the **standard errors** for the coefficients.

Possible Code Answer:

```
# Question 8
library(boot)
set.seed(2)

beta.fn = function(inputdata,index) {
lmfitboot = lm(formula = Volume ~ .,data=inputdata[index,])
return(lmfitboot$coef)
}
boot1000 = boot(Trees,beta.fn,R=1000)

apply(boot1000$t,2,sd)
lmfitfulldata = lm(formula = Volume ~ .,data=Trees);
summary(lmfitfulldata)$coefficients[,2]
```

From Problem 2 - Model Selection

Model 1: $Volume = \beta_0 + \beta_1 \cdot Girth + \beta_2 \cdot Height + \beta_3 \cdot Girth \cdot Height + \beta_4 \cdot Girth^2 + \beta_5 \cdot Girth^2 \cdot Height$

Model 2: $Volume = \beta_0 + \beta_1 \cdot Girth + \beta_2 \cdot Height$

Model 3: $Volume = \beta_0 + \beta_1 \cdot Girth + \beta_2 \cdot Height + \beta_3 \cdot Girth \cdot Height$

Model 4: $Volume = \beta_0 + \beta_1 \cdot Girth + \beta_2 \cdot Height + \beta_4 \cdot Girth^2 + \beta_5 \cdot Girth^2 \cdot Height$

Model 5: $Volume = \beta_0 + \beta_4 \cdot Girth^2 + \beta_5 \cdot Girth^2 \cdot Height$

Model 6: $Volume = \beta_0 + \beta_5 \cdot Girth^2 \cdot Height$

Use LOOCV (note n = 31) method to calculate $CV_{(31)}$ for each of Models 1-6.

Question

Enter your R code, including performing the cross-validation and computing the $CV_{(31)}$ measure for Model 1 below.

Possible Code Answer:

```
#Q12
Model1 = (Volume ~ Girth + Height + GirthHeight + Girth2 + Girth2Height)
Model2 = (Volume ~ Girth + Height)
Model3 = (Volume ~ Girth + Height + GirthHeight)
Model4 = (Volume ~ Girth + Height + Girth2 + Girth2Height)
Model5 = (Volume ~ Girth2 + Girth2Height)
Model6 = (Volume ~ Girth2Height)
```

```r
allpredictedLOOCV = rep(NA,n)
for (i in 1:n) {
  lmfitLOOCV = lm(formula = Model1,data=Trees[-i,])
  allpredictedLOOCV[i] = predict.lm(lmfitLOOCV,Trees[i,])
}

LOOCVmodel1 = mean((allpredictedLOOCV-Trees$Volume)^2); LOOCVmodel1
```

Question

Explain why you chose the model selected in the previous question.

Possible Answer: The value of CV_{31} is at a minimum, with a simple one-term model.

Question

Using the same split of the data into five sets as you performed in Problem 1, use 5-fold cross-validation method to calculate $CV_{(5)}$ for each of Models 1-6.

Possible Code Answer:

```r
#cross-validation
allpredictedCV = matrix(rep(NA,n*6),ncol=6)
for (model in 1:6) {
  for (i in 1:5) {
    groupi = (cvgroups == i)
    lmfitCV = lm(formula = sixModels[[model]],data=Trees[!groupi,])
    allpredictedCV[groupi,model] = predict.lm(lmfitCV,Trees[groupi,])
  }
}

CV5model1 = mean((allpredictedCV[,1]-Trees$Volume)^2); CV5model1
```

Question

Considering the form of the model that was selected by cross-validation, why does this model make sense from a practical standpoint?

Possible Answer: The model is relatively Simple as a geometric interpretation of formula corresponding to volume of cylinder or cone.

---

from Problem 3 - Model Assessment & Selection with KNN

Question

Starting with: `groups = c(rep(1:10),length=392)`

Set R's seed to 2: `set.seed(2)`

and use `sample()` to divide the data into **ten** sets.

Then use 10-fold cross-validation method to calculate $CV_{(10)}$ for **1**-nearest neighbor regression.

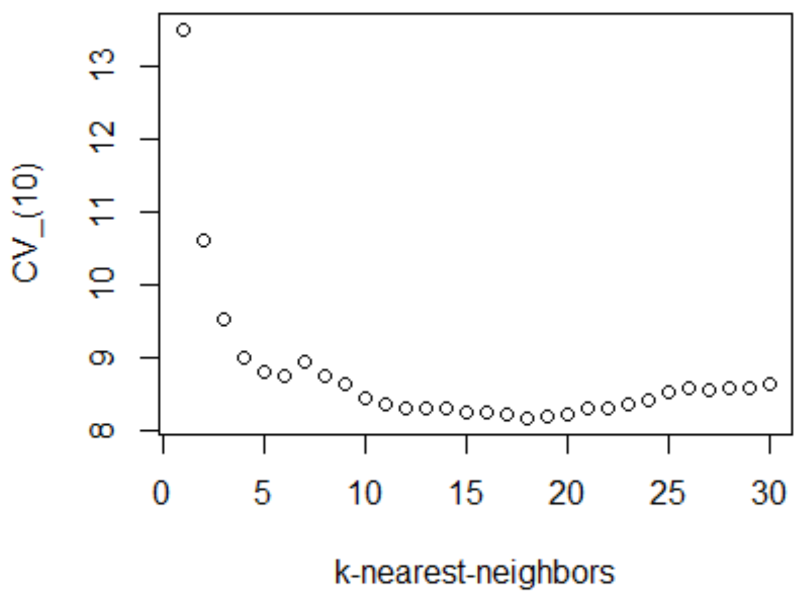
Enter your R code for performing the cross-validation and computing the $CV_{(10)}$ measure below.

Possible Code Answer:

```
library(ISLR)
library(FNN)
data(Auto)
names(Auto)
n=dim(Auto)[1]
AutoUsed = cbind(Auto$weight,Auto$year)

groups = rep(1:10,length=392)
set.seed(2)
cvgroups = sample(groups,n)

CV10all = rep(0,30)
for (j in 1:30) {
allpredictedCV = rep(0,n)
  for (i in 1:10)  {
    groupi = (cvgroups == i)
    train.Auto = AutoUsed[cvgroups != i,]
    train.Auto.std = scale(train.Auto)
    valid.Auto = AutoUsed[cvgroups == i,]
    valid.Auto.std = scale(valid.Auto,
                        center = attr(train.Auto.std, "scaled:center"),
                        scale = attr(train.Auto.std, "scaled:scale"))
    predictedCV = knn.reg(train.Auto.std, valid.Auto.std, Auto$mpg[!groupi],
k = j)
    allpredictedCV[groupi] = predictedCV$pred
  }

  CV10 = sum((allpredictedCV-Auto$mpg)^2)/n
  CV10all[j] = CV10
}

CV10all
```

Question

Consider models 1-30 as the k-nearest neighbors regression for values of k from 1 to 30. Using the same split of the data into ten sets as you performed in the Model assessment section, use 10-fold cross-validation method to calculate CV(10) for each of Models 1-30; remember to re-standardize each training set inside the cross-validation. Make a plot of the CV(10) as a function of k. Embed your plot to the Quiz question.



Question

In general, how should the $CV_{(10)}$ value compare to the value of MSE (computed by reusing the same data used to fit the model)?

Possible Answer: Generally, *MSE under*estimates the amount of error, compared to $CV_{(10)}$

Question

Which k (number of nearest neighbors) would you select based on the values of $CV_{(10)}$ for 10-fold CV?

Explain why you chose the k value specified in the previous question. *Comment on both model predictive ability and model complexity.*

Possible Answer: This is a good mix between minimizing $CV_{(10)}$ and keeping a simpler model (fewer neighbors).