Assignment 3

Ethan Tenison

3/20/2020

# Q1: Samples and Resampling (40 points)

In Question 2 on Assignment 2, we used the validation set approach to model assessment and selection.

## (a) How many observations were in the sample? In the training set? In the test set? How many

training sets were there? How many test sets? What are strengths and weaknesses of this approach?

There were 4,422 observations in the dataset from assignment 2. There were 3316 obs in the training and 1106 in the test with a 75% split. FOr the KNN classifier 3538 training 884 test, using a 80% 20% split. There was one set of each in both.

The validation approach is simple and easy, but your estimates can have a lot of variation depending on what is inside your training/test set.

### (b) How many observations would there have been in the training set for LOOCV? In the test set? How many

training sets would there have been? How many test sets? What are strengths and weaknesses of this approach?

With LOOCV there would be 4421 observations in training and 1 in test, but there would be 4422 train/test sets. The main advantage is that you will have less bias because n-1 observations are in your training set, but it is computational expensives. Therefore, it might take a long time to use if you have millions of observations.

### (c) How many observations would there have been in the training set using kfold CV? In the test set? How

training sets would there have been? How many test sets? What are strengths and weaknesses of this approach?

There number of observations and sets depends on how many folds you select. 5 K-fold validation would split the data into 5 random sets of approximately 884 or 885. Each set would get a chance to act as the training set, and all the other sets are added together to function as the test set. The biggest appeal for kfold is its speed. It’s less computationally intensive, but there is more bias. Increasing the number of folds reduces bias close to LOOCV.

### (d) How many observations would there have been in the training set using bootstrapping? In the test set?

How many training sets would there have been? How many test sets? What are strengths and weaknesses of this approach?

The bootstrap method does not separate the data into test and train. Random samples are taken from the original sample k number of times. The total number of bootstrap samples you want depends on how much variance is in your statistics. If there is high variance, you might want more samples than if the variance was low.

# Q2: Resampling in data (40 points)

library(reshape2)  
library(readr)  
library(janitor)  
library(dplyr)  
water <- read\_csv("Austin\_Water\_-\_Residential\_Water\_Consumption.csv")  
water <- clean\_names(water)  
water$customer\_class <- gsub(" - ", "\_", water$customer\_class)  
water$customer\_class <- gsub("-", "\_", water$customer\_class)  
water$year <- as.numeric(substr(water$year\_month, 1, 4))  
water$month <- as.numeric(substr(water$year\_month, 5, 6))  
water <- water[water$postal\_code != "", ]  
water <- na.omit(water)  
water <- as.data.frame(aggregate(data=water, total\_gallons~customer\_class+year+postal\_code, sum))  
water <- water[water$year == 2014,]  
water$hi\_use <- 0  
water$hi\_use[water$total\_gallons > mean(water$total\_gallons)] = 1  
water$zip\_code <- water$postal\_code  
houses <- read\_csv("2014\_Housing\_Market\_Analysis\_Data\_by\_Zip\_Code.csv")  
houses <- clean\_names(houses)  
atx <- inner\_join(water, houses, by="zip\_code")  
atx$hi\_use <- as.factor(atx$hi\_use)

### (a) Split the data into a training set (80%) and a test set (20%). Produce summary statistics of the

variable HiUse for the training and test sets separately.

library(caTools)  
set.seed(555)  
split = sample.split(atx$hi\_use, SplitRatio = 0.80)  
train = subset(atx, split == TRUE)  
test = subset(atx, split == FALSE)  
print("Train")

## [1] "Train"

print(summary(train$hi\_use))

## 0 1   
## 65 41

print("Test")

## [1] "Test"

print(summary(test$hi\_use))

## 0 1   
## 16 10

### (b) Fit a multiple logistic regression model predicting HiUse and using (among other things of your

choice) Median.home.value as a predictor to the training set. Compute the test error – i.e. the fraction of the observations in the test set that are misclassified.

Based on this model, with a threshold of 50%, there were 0 false negatives and 4 false positives.Overall it was correct = of the time.

model <- glm(hi\_use~ customer\_class +median\_home\_value + median\_household\_income + median\_rent ,family=binomial(link='logit'),data=train)  
summary(model)

##   
## Call:  
## glm(formula = hi\_use ~ customer\_class + median\_home\_value + median\_household\_income +   
## median\_rent, family = binomial(link = "logit"), data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.98106 -0.00008 -0.00006 0.67585 1.32839   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -2.076e+01 2.075e+03 -0.010 0.992  
## customer\_classIrrigation\_Residential -3.312e-01 2.949e+03 0.000 1.000  
## customer\_classMulti\_Family 2.051e+01 2.075e+03 0.010 0.992  
## customer\_classResidential 2.102e+01 2.075e+03 0.010 0.992  
## median\_home\_value -4.233e-06 3.940e-06 -1.074 0.283  
## median\_household\_income 2.591e-05 2.062e-05 1.256 0.209  
## median\_rent 7.636e-04 2.696e-03 0.283 0.777  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 141.466 on 105 degrees of freedom  
## Residual deviance: 59.507 on 99 degrees of freedom  
## AIC: 73.507  
##   
## Number of Fisher Scoring iterations: 18

library(caret)  
p <- predict(model, test, type = "response")  
print(summary(p))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.0000 0.0000 0.6365 0.4233 0.8068 0.9045

p\_class <- ifelse(p > 0.5, "Predict 1", "Predict 0")  
print(table(p\_class, test[["hi\_use"]]))

##   
## p\_class 0 1  
## Predict 0 12 0  
## Predict 1 4 10

### (c) Redo the split so that you have different observations in the training and test sets and produce

summary statistics of the variable HiUse in the new training and test sets. Fit the same multiple logistic regression model in (b) to the new training set and compute the new test error. Comment on your results with respect to the results from (a) and (b).

Based on this model, with a threshold of 50%, there were 4 false negatives and 4 false positives. Overall it was correct = of the time.

set.seed(2)  
split = sample.split(atx$hi\_use, SplitRatio = 0.80)  
train = subset(atx, split == TRUE)  
test = subset(atx, split == FALSE)  
print(summary(train$hi\_use))

## 0 1   
## 65 41

print(summary(test$hi\_use))

## 0 1   
## 16 10

model <- glm(hi\_use~ customer\_class +median\_home\_value + median\_household\_income + median\_rent ,family=binomial(link='logit'),data=train)  
summary(model)

##   
## Call:  
## glm(formula = hi\_use ~ customer\_class + median\_home\_value + median\_household\_income +   
## median\_rent, family = binomial(link = "logit"), data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.11522 -0.00002 0.00000 0.47203 1.59661   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.062e+01 2.820e+03 -0.011 0.9913   
## customer\_classIrrigation\_Residential -3.121e+00 3.789e+03 -0.001 0.9993   
## customer\_classMulti\_Family 2.321e+01 2.820e+03 0.008 0.9934   
## customer\_classResidential 2.490e+01 2.820e+03 0.009 0.9930   
## median\_home\_value -5.066e-06 4.369e-06 -1.160 0.2462   
## median\_household\_income -1.149e-05 2.719e-05 -0.423 0.6725   
## median\_rent 1.013e-02 4.277e-03 2.369 0.0178 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 141.466 on 105 degrees of freedom  
## Residual deviance: 49.244 on 99 degrees of freedom  
## AIC: 63.244  
##   
## Number of Fisher Scoring iterations: 19

p <- predict(model, test, type = "response")  
print(summary(p))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.0000 0.0000 0.3689 0.3758 0.7636 0.9996

p\_class <- ifelse(p > 0.5, "Predict 1", "Predict 0")  
print(table(p\_class, test[["hi\_use"]]))

##   
## p\_class 0 1  
## Predict 0 12 4  
## Predict 1 4 6

### (d) Split the data into k folds for a value of k that you choose. Produce summary statistics of the

variable HiUse for each of the k training and validation sets separately. Fit the same multiple logistic regression model in (b) to each of the k training sets and compute the k new test errors. Comment on your results.

The test error rates are: 0.153847, 0.148149, 0.185186, 0.1538462, 0.1538462. The summary statistics of hi\_use are below. Despite having a variation of hi\_use values in each kfold set, the test errors are very similar.

set.seed(2)  
rand <- sample(nrow(atx))  
  
k1row <- rand[rand %% 5 + 1 == 1]  
k2row <- rand[rand %% 5 + 1 == 2]  
k3row <- rand[rand %% 5 + 1 == 3]  
k4row <- rand[rand %% 5 + 1 == 4]  
k5row <- rand[rand %% 5 + 1 == 5]  
  
k1fold <- atx[k1row,]  
k2fold <- atx[k2row,]  
k3fold <- atx[k3row,]  
k4fold <- atx[k4row,]  
k5fold <- atx[k5row,]  
  
print("Summary Statistics for 5 folds")

## [1] "Summary Statistics for 5 folds"

summary(k1fold$hi\_use)

## 0 1   
## 17 9

summary(k2fold$hi\_use)

## 0 1   
## 14 13

summary(k3fold$hi\_use)

## 0 1   
## 15 12

summary(k4fold$hi\_use)

## 0 1   
## 17 9

summary(k5fold$hi\_use)

## 0 1   
## 18 8

#model with k1fold as test   
model <- glm(hi\_use~ customer\_class +median\_home\_value + median\_household\_income + median\_rent ,family=binomial(link='logit'),data=rbind(k2fold, k3fold,k4fold,k5fold))  
p <- predict(model, k1fold, type = "response")  
p\_class <- ifelse(p > 0.5, "Predict 1", "Predict 0")  
print(table(p\_class, k1fold[["hi\_use"]]))

##   
## p\_class 0 1  
## Predict 0 13 0  
## Predict 1 4 9

#model with k2fold as test  
model <- glm(hi\_use~ customer\_class +median\_home\_value + median\_household\_income + median\_rent ,family=binomial(link='logit'),data=rbind(k1fold, k3fold,k4fold,k5fold))  
p <- predict(model, k2fold, type = "response")  
p\_class <- ifelse(p > 0.5, "Predict 1", "Predict 0")  
print(table(p\_class, k2fold[["hi\_use"]]))

##   
## p\_class 0 1  
## Predict 0 11 1  
## Predict 1 3 12

#Model with k3fold as test  
model <- glm(hi\_use~ customer\_class +median\_home\_value + median\_household\_income + median\_rent ,family=binomial(link='logit'),data=rbind(k2fold, k1fold,k4fold,k5fold))  
p <- predict(model, k3fold, type = "response")  
p\_class <- ifelse(p > 0.5, "Predict 1", "Predict 0")  
print(table(p\_class, k3fold[["hi\_use"]]))

##   
## p\_class 0 1  
## Predict 0 12 3  
## Predict 1 3 9

#Model with k4fold as test  
model <- glm(hi\_use~ customer\_class +median\_home\_value + median\_household\_income + median\_rent ,family=binomial(link='logit'),data=rbind(k2fold, k1fold,k3fold,k5fold))  
p <- predict(model, k4fold, type = "response")  
p\_class <- ifelse(p > 0.5, "Predict 1", "Predict 0")  
print(table(p\_class, k4fold[["hi\_use"]]))

##   
## p\_class 0 1  
## Predict 0 13 0  
## Predict 1 4 9

#Model with k5fold as test  
model <- glm(hi\_use~ customer\_class +median\_home\_value + median\_household\_income + median\_rent ,family=binomial(link='logit'),data=rbind(k2fold, k1fold,k3fold,k4fold))  
p <- predict(model, k5fold, type = "response")  
p\_class <- ifelse(p > 0.5, "Predict 1", "Predict 0")  
print(table(p\_class, k5fold[["hi\_use"]]))

##   
## p\_class 0 1  
## Predict 0 14 0  
## Predict 1 4 8

# Q3: Bootstrapping in simulated data (30 points)

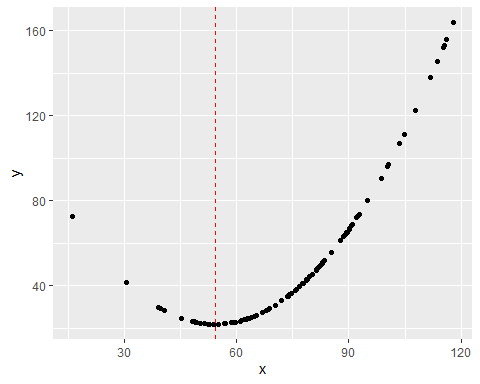
In 1993, Barcelona began a long program to implement “Superblocks” aimed at reducing through traffic in urban areas with many pedestrians. Vox has interesting write up (<https://www.vox.com/energy-andenvironment/2019/4/9/18300797/barcelona-spain-superblocks-urban-plan>). Suppose you are employed by a hypothetical city that had implemented Superblocks along with a parking permit system for residentialtraffic. You notice two interesting things over the course of a year: 1) that vehicle-pedestrian accidents are surprisingly high when the number of permits issued are very low – perhaps so low that pedestrians forget that there are ever any cars at all, and 2) vehicle-pedestrian accidents are UN-surprisingly high when the number of permits issued are very high. This suggests a U-shaped relationship between vehicle-pedestrian accidents and the number of permits issued.

### (a) Generate 100 observations of the number of permits issued distributed N(mew =75, sigma =20).

Generate an error term distributed N(mew =0, sigma =7). Generate a number of accidents from: Y = B0 + B1X1 + B2x1^2 + e where y is the number of accidents, B0 = 125, B1 = −3.8, X1 is the number of permits issued, B2= 0.035, and e is the error term. Plot the scatterplot. Plot the function in Equation 1 – but without the error term – over the scatterplot (hint: use stat\_function() together with xlim() in ggplot). Estimate the number of permits issue that corresponds to the fewest number of accidents and draw a vertical line on the figure.

The number of permits with the fewest number of accidents is approximately 54.

library(ggplot2)  
set.seed(666)  
x <- rnorm(100, 75, 20)  
x2 <- x\*x  
e <- rnorm(100, 0, 7)  
y <- 125 + -3.8\*x + 0.035\*x\*\*2  
df <- data.frame(y, x, x2)  
ggplot(data = df, aes(x=x, y=y)) + geom\_point() + geom\_vline(xintercept = 54.2857143, linetype = "dashed", color = "red")



### (b) Conduct a linear regression of the number of accidents on the number of permits and the

squared number of permits and present and interpret the results. This is certainly an inference problem, but the coefficients b1 and b2 are not very interesting by themselves. What is more interesting is the vertical line at x= -b1/2b2 which identifies the minimum Y in terms of X. Calculate this value and interpret it.

124.99999999999995736744 -3.79999999999999760192(54.2857143) + 0.03500000000000001721(54.2857143)^2 = 21.8571428571

*21.8571428571* is the number of accidents we would expect to see with approx 54 permits issued.

library(boot)

##   
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':  
##   
## melanoma

model <- lm(y ~ x + x2, data = df)  
summary(model)

##   
## Call:  
## lm(formula = y ~ x + x2, data = df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.615e-13 -4.068e-14 -1.397e-14 1.360e-15 1.713e-12   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.250e+02 1.834e-13 6.817e+14 <2e-16 \*\*\*  
## x -3.800e+00 5.035e-15 -7.547e+14 <2e-16 \*\*\*  
## x2 3.500e-02 3.341e-17 1.048e+15 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.882e-13 on 97 degrees of freedom  
## Multiple R-squared: 1, Adjusted R-squared: 1   
## F-statistic: 1.687e+30 on 2 and 97 DF, p-value: < 2.2e-16

### (c) Linear regression gives us a standard error for the coefficients b1 and b2, but not for this

important and interesting value of x that minimizes the number of accidents. Use bootstrapping to estimate a standard error for this value. Interpret the results – make and explain a recommendation for a the number of permits to issue.

The standard error and bias is extremely lower. Therefore, we can be confident that 54or 55 permits would result in the lowest number of accidents.

library(boot)  
  
bs <- function(data, indices) {  
 d <- data[indices,] # allows boot to select sample  
 fit <- lm(y ~ x + x2, data=d, subset= indices)  
 return(-coef(fit)["x"]/(2\*coef(fit)["x2"]))  
}  
  
results <- boot(data=df, statistic=bs,  
 R=1000)  
  
results

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = df, statistic = bs, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 54.28571 4.973799e-14 4.039066e-14