MATH 4322 Homework 4 Fall 2021 Solutions

Cathy Poliak

10/28/2021

Problem 1

We will review k-fold cross-validation.

(a) Explain how k-fold cross-validation is implemented.
(b) What are the advantages and disadvantages of k-fold crossvalidation relative to:

i. The validation set approach?
ii. LOOCV?

Answer

(a) This approach involves randomly k-fold CV dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining $k-1$ folds. The mean squared error, MSE_1, is then computed on the observations in the held-out fold. This procedure is repeated k times; each time, a different group of observations is treated as a validation set. This process results in k estimates of the test error, $MSE_1, MSE_2, \ldots, MSE_k$. The k-fold CV estimate is computed by averaging these values,

$$CV(k) = \frac{1}{k}\sum_{i=1}^{k} MSE_i.$$

(b) Advantages:

- Less computational intensive.
- Does not lose in estimation quaility
- The variability in the estimates are negligible.

Disadvantages: * More bias

Problem 2

Suppose that we use some statistical learning method to make a prediction for the response Y for a particular value of the predictor X. Carefully describe how we might estimate the standard deviation of our prediction.

Answer

If we suppose using some statistical learning method to make a prediction for the response Y for a particular value of the predictor X we might estimate the standard deviation of our prediction by using the bootstrap approach. The bootstrap approach works by repeatedly sampling observations (with replacement) from the original data set B times, for some large value of B, each time fitting a new model and subsequently obtaining the RMSE of the estimates for all B models.

Problem 3

We will perform cross-validation on a simulated data set.

(a) Generate a simulated data set as follows:

```
set.seed(1)
x=rnorm(100)
y=x-2*x^2+ rnorm(100)
```

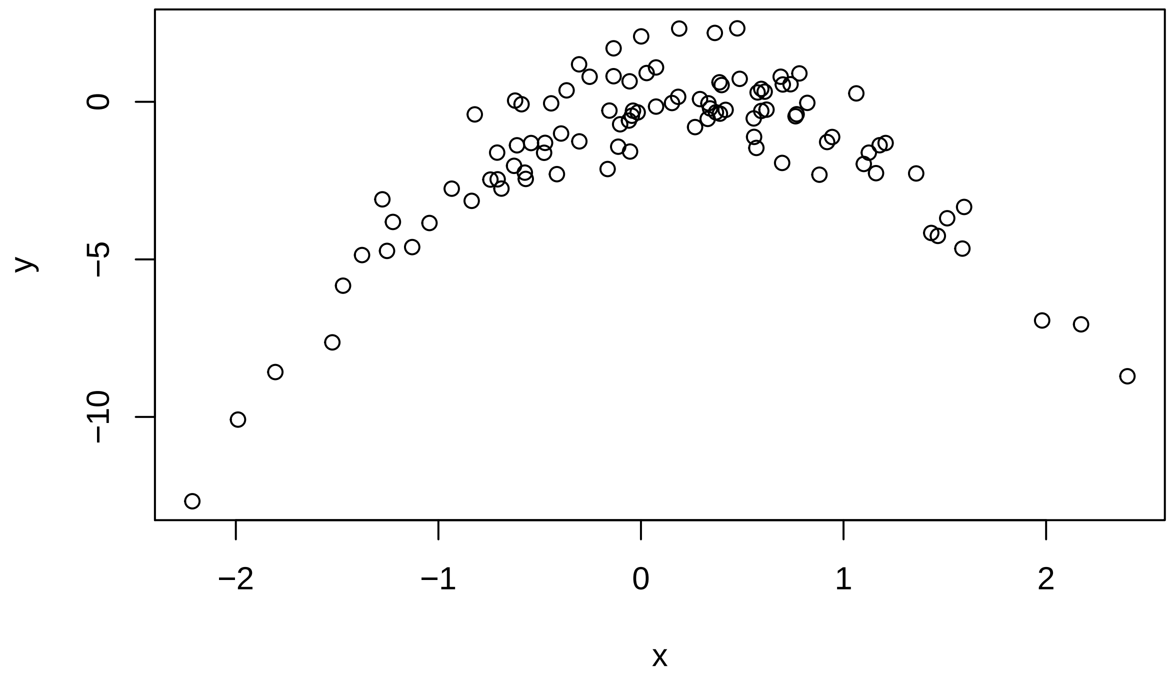
In this data set, what is n and what is p? Write out the model used to generate the data in equation form.

Answer $Y = X - 2X^2 + \epsilon$, $n = 100$, $p = 2$

(b) Create a scatterplot of X against Y . Comment on what you find.

Answer

```
plot(x,y)
```



This is parabolic shape, non-linear.

(c) Set a random seed, and then compute the LOOCV errors that result from fitting the following four models using least squares:

i. $Y = \beta_0 + \beta_1 X + \epsilon$
ii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \epsilon$
iii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$
iv. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \epsilon$

Note: you might find it helpful to use the `data.frame()` function to create a single data set containg both X and Y.

```
set.seed(10)
library(boot)
xy = data.frame(x,y)
cv.error = rep(0,4)
for (i in 1:4) {
  glm.fit = glm(y~poly(x,i),data = xy)
  cv.error[i] = cv.glm(xy,glm.fit)$delta[1]
}
cv.error
```

```
## [1] 7.2881616 0.9374236 0.9566218 0.9539049
```

We see a sharp drop in the estimated MSE between the linear and quadratic fits, but then no clear improvement from using higher-order polynomials.

(d) Repeat (c) using another random seed, and report your results. Are your results the same as what you got in (c)? Why?

Answer

```
set.seed(100)
library(boot)
xy = data.frame(x,y)
cv.error = rep(0,4)
for (i in 1:4) {
  glm.fit = glm(y~poly(x,i),data = xy)
  cv.error[i] = cv.glm(xy,glm.fit)$delta[1]
}
cv.error
```

```
## [1] 7.2881616 0.9374236 0.9566218 0.9539049
```

The values are the same as before. There is no randomness in the training/validtaion set splits.

(e) Which of the models in (c) had the smallest LOOCV error? Is this what you expected? Explain your answer.

Answer

The smallest LOOCV error is the quadratic polynomial. Yes, because we set y as a result of x^2.

(f) Comment on the statistical significance of the coefficient estimates that results from fitting each of the models in (c) using least squares. Do these results agree with the conclusions drawn based on the cross-validation results?

```
glm.fit = glm(y~poly(x,4),data = xy)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 4), data = xy)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0550  -0.6212  -0.1567   0.5952   2.2267
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.55002    0.09591 -16.162  < 2e-16 ***
## poly(x, 4)1   6.18883    0.95905   6.453 4.59e-09 ***
## poly(x, 4)2 -23.94830    0.95905 -24.971  < 2e-16 ***
## poly(x, 4)3   0.26411    0.95905   0.275    0.784
## poly(x, 4)4   1.25710    0.95905   1.311    0.193
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9197797)
##
##     Null deviance: 700.852  on 99  degrees of freedom
## Residual deviance:  87.379  on 95  degrees of freedom
## AIC: 282.3
##
## Number of Fisher Scoring iterations: 2
```

Yes the terms with x and x^2 are statistically significant. The other terms are not. This confirms with what was stated in part (c).

Problem 4

We will use a logistic regression to predict the probability of `default` using income and `balance` on the `Default` data set in the ISLR package. We will now estimate the test error of this logistic regression model using the validation set approach. Do not forget to set a random seed before beginning your analysis.

(a) Fit a logistic regression model that uses income and balance to predict default.

Answer

```
library(ISLR)
default.reg = glm(default ~ income + balance,
                  data = Default,family = "binomial")
summary(default.reg)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default)
```

```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4725  -0.1444  -0.0574  -0.0211   3.7245
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
## income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
## balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1579.0  on 9997  degrees of freedom
## AIC: 1585
##
## Number of Fisher Scoring iterations: 8
```

(b) Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps:

i. Split the sample set into a training set and a validation set.
ii. Fit a multiple logistic regression model using only the training observations.
iii. Obtain a prediction of default status for each individual in the validation set by computing the posterior probability of default for that individual, and classifying the individual to the default category if the posterior probability is greater than 0.5.
iv. Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

Answer

```
set.seed(2)
train = sample(1:nrow(Default),nrow(Default)/2)
default.test = Default[-train,]
default.train = Default[train,]
default.reg2 = glm(default ~ income + balance,
                 data = default.train,family = "binomial")
pred.test = predict(default.reg2,newdata = default.test,type = "response")
yhat = ifelse(pred.test<0.5,"No","Yes")
(conf.mat = table(yhat,default.test$default))
```

```
##
## yhat    No  Yes
##   No  4819  101
##   Yes   18   62
```

```
(conf.mat[1,2]+conf.mat[2,1])/5000
```

```
## [1] 0.0238
```

The test error rate for my validation set is 2.3%.

(c) Repeat the process in (b) three times, using three different splits of the observations into a training set and a validation set. Comment on the results obtained.

Answer Sample 1

```
##
## yhat    No  Yes
##   No  4822  109
##   Yes   23   46
```

```
## [1] 0.0264
```

Sample 2

```
##
## yhat    No  Yes
##   No  4821  110
##   Yes   20   49
```

```
## [1] 0.026
```

Sample 3

```
##
## yhat    No  Yes
##   No  4815  125
##   Yes   19   41
```

```
## [1] 0.0288
```

All of these stayed close between 2.5% and 3.0%.

Problem 5

We continue to consider the use of a logistic regression model to predict the probability of `default` using income and `balance` on the `Default` data set. In particular, we will now compute estimates for the standard errors of the income and balance logistic regression coefficients in two different ways: (1) using the bootstrap, and (2) using the standard formula for computing the standard errors in the *glm()* function. Do not forget to set a random seed before beginning your analysis.

(a) Using the *summary()* and *glm()* functions, determine the estimated standard errors for the coefficients associated with income and balance in a multiple logistic regression model that uses both predictors.

Answer

```r
default.reg = glm(default ~ income + balance,
                  data = Default,family = "binomial")
summary(default.reg)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4725  -0.1444  -0.0574  -0.0211   3.7245
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
## income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
## balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1579.0  on 9997  degrees of freedom
## AIC: 1585
##
## Number of Fisher Scoring iterations: 8
```

Standard error for $\hat{\beta}_2 = 0.000005$ and $\hat{\beta}_3 = 0.00023$

(b) Write a function, *boot.fn()*, that takes as input the Default data set as well as an index of the observations, and that outputs the coefficient estimates for income and balance in the multiple logistic regression model.

Answer

```r
boot.fun = function(data,index)
  return(coef(glm(default ~ income + balance,
                  data = data, subset = index,family = "binomial")))
boot.fun(Default,1:nrow(Default))
```

```
##   (Intercept)        income       balance
## -1.154047e+01  2.080898e-05  5.647103e-03
```

(c) Use the *boot()* function together with your *boot.fn()* function to estimate the standard errors of the logistic regression coefficients for income and balance.

Answer

```
set.seed(1)
boot(Default,boot.fun,1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Default, statistic = boot.fun, R = 1000)
##
##
## Bootstrap Statistics :
##          original        bias     std. error
## t1* -1.154047e+01 -3.945460e-02 4.344722e-01
## t2*  2.080898e-05  1.680317e-07 4.866284e-06
## t3*  5.647103e-03  1.855765e-05 2.298949e-04
```

(d) Comment on the estimated standard errors obtained using the *glm()* function and using your bootstrap function.

Answer

The standard errors using both methods are very close.

Problem 6

We will now consider the `Boston` housing data set, from the `MASS` library.

(a) Based on this data set, provide an estimate for the population mean of *medv*. Call this estimate $\hat{\mu}$.

```
library(MASS)
(hat.mu = mean(Boston$medv))
```

```
## [1] 22.53281
```

(b) Provide an estimate of the standard error of $\hat{\mu}$. Interpret this result. *Hint*: We can compute the standard error of the sample mean by dividing the sample standard deviation by the square root of the number of observations.

```
(se.hatmu = sd(Boston$medv)/sqrt(nrow(Boston)))
```

```
## [1] 0.4088611
```

(c) Now estimate the standard error of $\hat{\mu}$ using the bootstrap. How does this compare to your answer from (b)?

```
mean.fun = function(data,index)
  mean(data[index],na.rm = TRUE)
(boot.out = boot(data = Boston$medv,mean.fun,1000))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston$medv, statistic = mean.fun, R = 1000)
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1* 22.53281 0.0109415    0.414028
```

Both standar errors are close to 0.41.

(d) Based on your bootstrap estimate from (c), provide a 95% confidence interval for the mean of *medv*. Compare it to the results obtained using `t.test(Boston$medv)`. *Hint*: You can approximate a 95% confidence interval using the formula $[\hat{\mu} - 2 \times \text{SE}(\hat{\mu}), \hat{\mu} + 2 \times \text{SE}(\hat{\mu})]$.

Answer

```
hat.mu + c(-1,1)*2*sd(boot.out$t)
```

```
## [1] 21.70475 23.36086
```

```
t.test(Boston$medv)
```

```
##
##  One Sample t-test
##
## data:  Boston$medv
## t = 55.111, df = 505, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  21.72953 23.33608
## sample estimates:
## mean of x
##  22.53281
```

Also can do `boot.ci`

```
(boot.ci(boot.out))
```

```
## Warning in boot.ci(boot.out): bootstrap variances needed for studentized
## intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.out)
##
```

```
## Intervals :
## Level      Normal              Basic
## 95%   (21.71, 23.33 )   (21.75, 23.36 )
##
## Level     Percentile            BCa
## 95%   (21.71, 23.32 )   (21.71, 23.32 )
## Calculations and Intervals on Original Scale
```

(e) Based on this data set, provide an estimate, $\hat{\mu}_{\text{med}}$, for the median value of *medv* in the population.

Answer

```
(hat.med = median(Boston$medv))
```

```
## [1] 21.2
```

(f) We now would like to estimate the standard error of $\hat{\mu}_{\text{med}}$. Unfortunately, there is no simple formula for computing the standard error of the median. Instead, estimate the standard error of the median using the bootstrap. Comment on your findings.

```
median.fun = function(dat,idx) median(dat[idx],na.rm = TRUE)
set.seed(10)
boot.out.median = boot(data = Boston$medv,median.fun,1000)
boot.out.median
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston$medv, statistic = median.fun, R = 1000)
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1*     21.2 -0.00665   0.3745779
```

The standard error is estimated to be 0.375. That means the estimated median for `medv` may be off by on average 0.375 or so.

(g) Based on this data set, provide an estimate for the tenth percentile of medv in Boston suburbs. Call this quantity $\hat{\mu}_{0.1}$. (You can use the quantile() function.)

```
(hat.quant1 = quantile(Boston$medv,0.1,type = 5))
```

```
##   10%
## 12.71
```

(h) Use the bootstrap to estimate the standard error of $\hat{\mu}_{0.1}$. Comment on your findings.

```r
quant.fun = function(data,index) quantile(data[index],0.1,type = 5)
set.seed(100)
(boot.out = boot(Boston$med,quant.fun,10000))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston$med, statistic = quant.fun, R = 10000)
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1*    12.71 0.018014   0.5089463
```

The estimate may be off by 0.51 or so.