Homework 4 - MATH 4322

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 cross-validation relative to:

i. The validation set approach?
ii. LOOVC?

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

We will now derive the probability that a given observation is part of a bootstrap sample. Suppose that we obtain a bootstrap sample from a set of n observations.

(a) What is the probability that the first bootstrap observation is not the jth observation from the original sample? Justify your answer.

(b) What is the probability that the second bootstrap observation is not the jth observation from the original sample?

(c) Argue that the probability that the jth observation is not in the bootstrap sample is $\left(1 - \frac{1}{n}\right)^n$.

(d) When $n = 5$, what is the probability that the jth observation is in the bootstrap sample?

(e) When $n = 100$, what is the probability that the jth observation is in the bootstrap sample?

(f) When $n = 10,000$, what is the probability that the jth observation is in the bootstrap sample?

(g) Create a plot that displays, for each integer value of n from 1 to 100,000, the probability that the jth observation is in the bootstrap sample. Comment on what you observe.

(h) We will now investigate numerically the probability that a bootstrap sample of size $n = 100$ contains the jth observation. Here $j = 4$. We repeatedly create bootstrap samples, and each time we record whether or not the fourth observation is contained in the bootstrap sample.

```
store <- rep(NA, 10000)
 for(i in 1:10000){
store[i] <- sum(sample (1:100 , rep=TRUE) == 4) > 0
}
mean(store)
```

Comment on the results obtained.

Answer

(a) The probability that the first bootstrap is not the jth observation from the original sample is $1 - \frac{1}{n}$. This is because there are n samples, and we are equally likely to pick

each observation when taking our first bootstrap observation.

(b) Since the bootstrap observations are sampled from the original observations *with replacement*, the probability that the second bootstrap observation is the on the jth observation from the original sample is still $1 - \frac{1}{n}$.

(c) As the probability that any particular bootstrap observation is not the jth observation from the original sample is still $1 - \frac{1}{n}$. We are taking a total of n bootstrap observations from the original set of n observations, the probability that the jth observation is not in the bootstrap sample is $\left(1 - \frac{1}{n}\right)^n$. Note that we are multiplying probabilities because sampling with replacement means that the individual bootstrap observations in the bootstrap sample are *independent*.

(d) The probability that the jth observation is in the bootstrap sample is $1-(1-\frac{1}{n})^n$. When $n = 5$, This probability is

$$1 - (1 - \frac{1}{5})^5 = 0.6723$$

(e) When $n = 100$ probability that the jth observation is in the bootstrap sample is

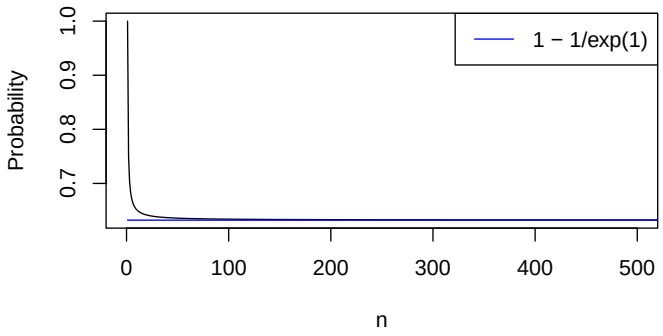
$$1 - (1 - \frac{1}{100})^{100} = 0.634$$

(f) When $n = 1000$ probability that the jth observation is in the bootstrap sample is

$$1 - (1 - \frac{1}{1000})^{1000} = 0.6323$$

(g) See the plot below

```
x = seq(1:100000)
p.x = 1 - (1 - 1/x)^x
plot(x,p.x,xlab = "n",ylab = "Probability",type = "l",xlim = c(0,500))
lines(x,y=rep(1-1/exp(1),length(x)),type = "l",col = "blue")
legend("topright",legend = "1 - 1/exp(1)",col = "blue",lty =1)
```



By L'Hopital's rule $lim_{n\to \text{inf}}(1-\frac{1}{n})^n = \frac{1}{exp(1)}$.

(h) Simulation

```
store <- rep(NA, 10000)
 for(i in 1:10000){
store[i] <- sum(sample (1:100 , rep=TRUE) == 4) > 0
}
mean(store)
```

```
[1] 0.6341
```

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)
```

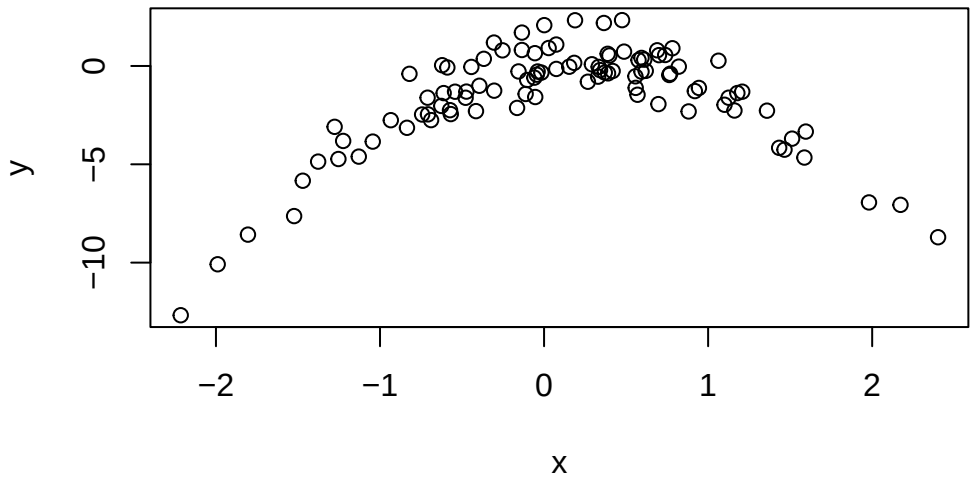
In this data set, what is n and what is p? Write out the model used to generate the data in equation form.

Answer $Y = X - 2X^2 + \epsilon$, $n = 100$, $p = 1$

(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 containing both X and Y.

Answer

```
set.seed(10)
library(boot)
```

```
Warning: package 'boot' was built under R version 4.3.2
```

```
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/validation 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?

Answer

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

```
Call:
glm(formula = y ~ poly(x, 4), data = xy)

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

```r
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)

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
```

(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 will now consider the `Boston` housing data set, from the `ISLR2` library.

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

(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.

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

(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 $[\hat{\mu} - 2SE(\hat{\mu}), \hat{\mu} + 2SE(\hat{\mu})]$.

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

(f) We now would like to estimate the standard error of $\hat{\mu}_{med}$. 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.

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

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

Answer

(a) The estimate for μ should be the sample mean $\bar{x}$ from the observations.

```
library(ISLR2)
attach(Boston)
(hat.mu = mean(medv))
```

```
[1] 22.53281
```

$\hat{\mu} = 22.5328$

(b) The standard error of $\hat{\mu}$ can be estimated by the standard deviation of the observations divided by the square root of the number of samples.

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

```
[1] 0.4088611
```

$SE(\hat{\mu}) = 0.4089$

(c) Using the bootstrap

```
library(boot)
mean.fun = function(dat, idx) mean(dat[idx], na.rm = TRUE)
(out.boot = boot(medv,mean.fun,1000))
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = mean.fun, R = 1000)


Bootstrap Statistics :
    original      bias    std. error
t1* 22.53281 -0.01295593   0.3964158
```

This is close to the estimated standard error from part (b).

(d) Confidence Intervals

```
t.test(medv)$conf.int
```

```
[1] 21.72953 23.33608
attr(,"conf.level")
[1] 0.95
```

```
out.boot$t0 + c(-1,1)*2*sd(out.boot$t)
```

```
[1] 21.73997 23.32564
```

These values are close.

(e) Estimate of the median.

```
median(medv)
```

```
[1] 21.2
```

$\hat{\mu}_{med} = 21.2$

(f) Bootstrap for median

```
median.fun = function(dat, idx) median(dat[idx],na.rm = TRUE)
(out.boot.median = boot(medv,median.fun,1000))
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = median.fun, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*     21.2 -0.0059   0.3777869
```

We expect the median to be 21.1941 give or take 0.3778.

(g) Estimate of the tenth percentile.

```
quantile(medv,.1)
```

```
  10%
12.75
```

$\hat{\mu}_{0.1} = 12.75$

(h) Bootstrap of the tenth percentile

```
fun10 = function(dat, idx) quantile(dat[idx],0.1, na.rm = TRUE)
(out.boot10 = boot(medv,fun10,1000))
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = fun10, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*    12.75 0.03375   0.4762985
```

We expect the 10th percentile to be 12.7838, give or take 0.4763.