HOMEWORK 7

Andrew Guo

3/31/2022

PROBLEM 1

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? Provide a brief mathematical justification.

Bootstrapping resamples with replacement, meaning that there is always going to be n observations as a whole. Thus, the probability that the observation is not in the jth observation from the original sample will be $(1-(1/n)$.; with 1/n representing the probability that the jth observation is being selected.

(b) What is the probability that the second bootstrap observation **is not** the jth observation from the original sample? Provide a brief mathematical justification.

If you are sampling with replacement, then the probability is the same as the first bootstrap observation; the probability shouldn't change based on the order of which observation is being observed.

(c) Argue that the probability that the jth observation **is not** in the entire bootstrap sample is $(1 - 1/n)^n$.

If the probability for one specific observation to not occur in the first observation is is (1-(1/n)), and if this value holds for all specific observations in the boostrap sample since there is replacement, and because a bootstrap contains exactly n observations, the probability for that specific observation to not be in the sample would be the probability for that one specific observation, multiplied by itself n times, therefore, we would get $(1-(1/n)^n$.

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

This value would equal to 1 minus the probability that it does not occur at all in the entire bootstrap model. Thus, the probability would be, in general terms, $1-(1-(1/n)^n$. If we have 5 observations, this would mean that the probability equals:

1 - [(1 - (1/5)]^5

1 - (4/5)^5 = 0.67232

(e) When $n = 25$, what is the probability that the jth observation **is** in the bootstrap sample?

1 - [(1 - (1/n)]^n

1 - [(1 - (1/25)]^(25)

1 - (24/25)^(25) = 0.6396032831

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

1 - (1 - (1/100)^(100)

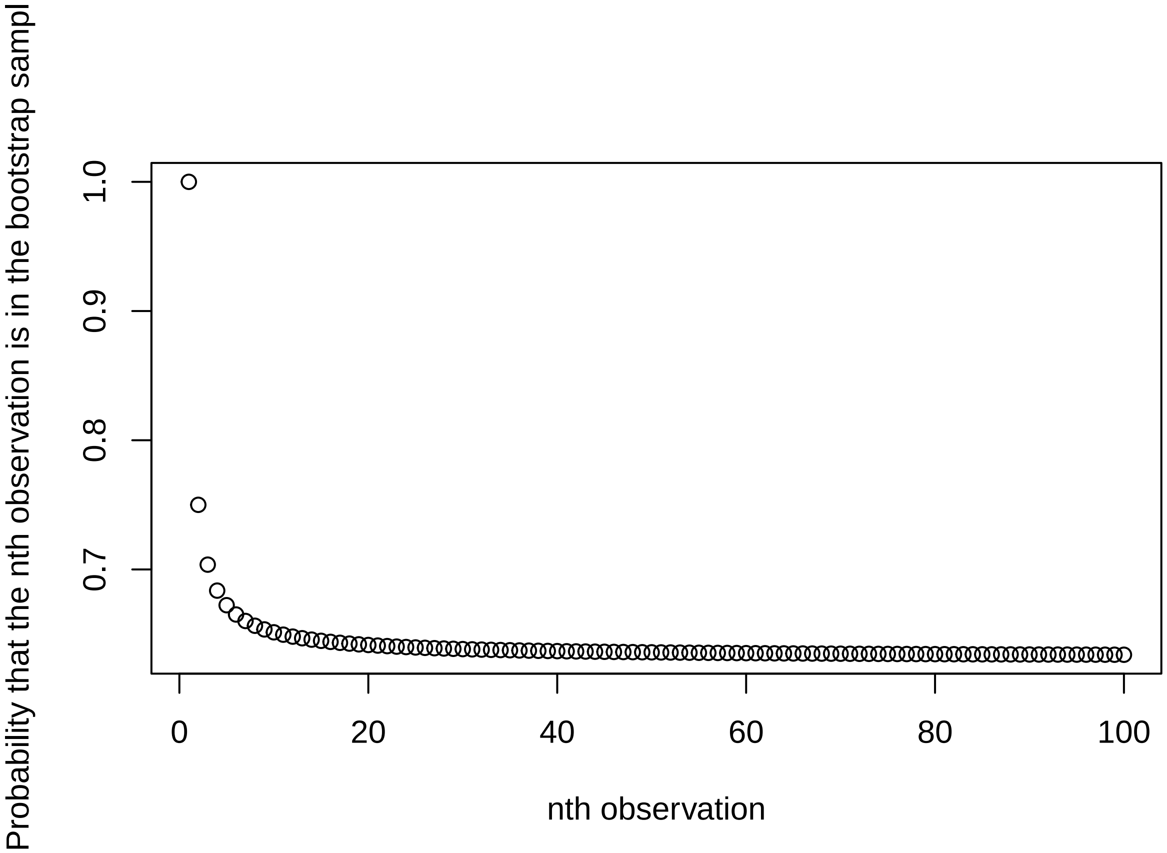
1 - (99/100)^(100) = 0.6339676587

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

```r
y = rep(0, 100)

for (i in 1:100) {
  y[i] = 1-(1 - (1/i))^i
}

plot(1:100, y, xlab = 'nth observation', ylab = 'Probability that the nth observation is
```



I observe that the probability that the jth observation is in the bootstrap sample goes down as n observations increase, reaching a minimum probability of around 63%.

PROBLEM 2

We will now 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)
```

```
library(boot)

set.seed(1)

x <- rnorm(100)

y <- x - 2 * x^2 + rnorm(100)
```

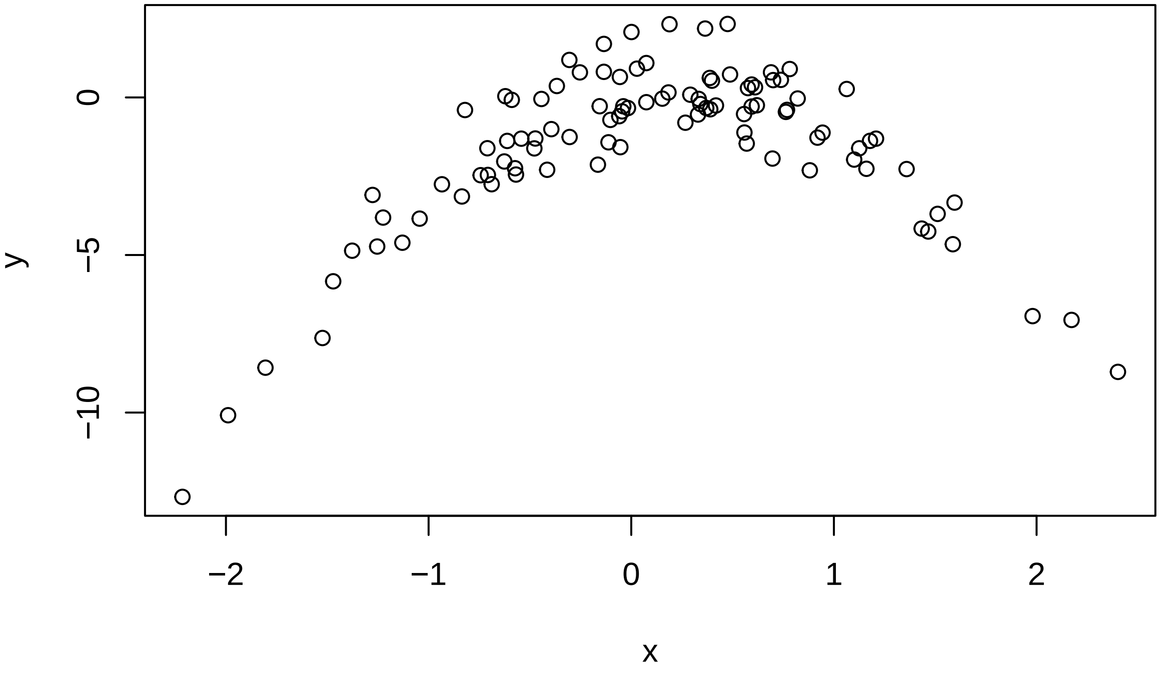
In this data set, what is n and what is p? Write out the model used to generate the data in equation form.

In this model, n = 100 and p is equal to 2.

General model used to generate data: y = b0 + b1x + b2x^2

(b) Create a scatterplot of `x` against `y` . Comment on what you observe.

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



I notice a strong, non-linear, negative parabolic relationship between x and y.

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

1. $y = \beta_0 + \beta_1 x + \epsilon$
2. $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \epsilon$
3. $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \epsilon$
4. $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \epsilon$

Note you may find it helpful to use the `data.frame()` function to create a single data set containing both `x` and `y`. Consult with the LOOCV section of the Chapter 5 lab.

```
set.seed(2)
dataxy <- data.frame(x,y) #dataframe containing both x and y values
attach(dataxy)
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     x, y
```

```
cv.error <- rep (0, 4)
for (i in 1:4) {
  glm.fit <- glm(y ~ poly(x , i), data = dataxy)
  cv.error[i] <- cv.glm(dataxy , glm.fit)$delta[1]
}
cv.error
```

```
## [1] 7.2882 0.9374 0.9566 0.9539
```

The LOOCV errors that result from fitting the following four models using least squares are 7.2882, 0.9374, 0.9566, and 0.9539.

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

```
set.seed(3)
dataxy2 <- data.frame(x, y) #dataframe containing both x and y values

cv.error <- rep (0, 4)
for (i in 1:4) {
  glm.fit <- glm(y ~ poly(x , i), data = dataxy2)
  cv.error[i] <- cv.glm(dataxy2 , glm.fit)$delta[1]
}
cv.error
```

```
## [1] 7.2882 0.9374 0.9566 0.9539
```

The LOOCV errors that result from fitting the following four models using least squares with a different seed are 7.2882, 0.9374, 0.9566, and 0.9539.

The results are the same as what I got in (c) because each of the training sets are not that much different from each other; only one value in each of the training sets is different across the board; LOOCV has very little 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.

The quadratic model had the smallest LOOCV error. Since the true relationship between x and y is quadratic, it would make sense for the quadratic model to have the smallest LOOCV error.

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

```r
lm.fit.xy1 = lm(y ~ poly(x,1), data = dataxy)
summary(lm.fit.xy1)
```

```
##
## Call:
## lm(formula = y ~ poly(x, 1), data = dataxy)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.516 -0.680  0.681  1.549  3.818
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.55       0.26   -5.96    4e-08 ***
## poly(x, 1)      6.19       2.60    2.38    0.019 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.6 on 98 degrees of freedom
## Multiple R-squared:  0.0547, Adjusted R-squared:  0.045
## F-statistic: 5.67 on 1 and 98 DF,  p-value: 0.0192
```

```r
lm.fit.xy2 = lm(y ~ poly(x,2), data = dataxy)
summary(lm.fit.xy2)
```

```
##
## Call:
```

```
## lm(formula = y ~ poly(x, 2), data = dataxy)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.965 -0.625 -0.129  0.580  2.270
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5500     0.0958  -16.18  < 2e-16 ***
## poly(x, 2)1   6.1888     0.9580    6.46  4.2e-09 ***
## poly(x, 2)2 -23.9483     0.9580  -25.00  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.958 on 97 degrees of freedom
## Multiple R-squared:  0.873,  Adjusted R-squared:  0.87
## F-statistic:  333 on 2 and 97 DF,  p-value: <2e-16
```

```r
lm.fit.xy3 = lm(y ~ poly(x,3), data = dataxy)
summary(lm.fit.xy3)
```

```
##
## Call:
## lm(formula = y ~ poly(x, 3), data = dataxy)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.976 -0.630 -0.123  0.554  2.284
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5500     0.0963  -16.10   <2e-16 ***
## poly(x, 3)1   6.1888     0.9626    6.43    5e-09 ***
## poly(x, 3)2 -23.9483     0.9626  -24.88   <2e-16 ***
## poly(x, 3)3   0.2641     0.9626    0.27     0.78
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.963 on 96 degrees of freedom
## Multiple R-squared:  0.873,  Adjusted R-squared:  0.869
## F-statistic:  220 on 3 and 96 DF,  p-value: <2e-16
```

```r
lm.fit.xy4 = lm(y ~ poly(x,4), data = dataxy)
summary(lm.fit.xy4)
```

```
##
```

```
## Call:
## lm(formula = y ~ poly(x, 4), data = dataxy)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.055 -0.621 -0.157  0.595  2.227
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5500     0.0959  -16.16  < 2e-16 ***
## poly(x, 4)1   6.1888     0.9591    6.45  4.6e-09 ***
## poly(x, 4)2 -23.9483     0.9591  -24.97  < 2e-16 ***
## poly(x, 4)3   0.2641     0.9591    0.28     0.78
## poly(x, 4)4   1.2571     0.9591    1.31     0.19
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.959 on 95 degrees of freedom
## Multiple R-squared:  0.875,  Adjusted R-squared:  0.87
## F-statistic:  167 on 4 and 95 DF,  p-value: <2e-16
```

As we can see, only the linear and quadratic coefficients are statistically significant; all other higher order coefficients are not. This falls in line with the conclusions drawn from the cross-validation results, since it shows that the quadratic model is the superior model of the four: the linear model, while having a predictor that is statistically significant, holds an LOOCV error far greater than the other models, and of the higher order models, only the quadratic model is statistically significant while also holding the smallest LOOCV error.