2_assignement

Chapter 5

Exercise 1 ???

Exercise 2

a) 1-1/n

The probability that it is the jth is 1/n, the probability that it is not is 1-1/n

b) Same as before 1-1/n

The bootstrap always takes a random observation from the original sample, no matter what came before

c) For it not to be in the bootstrap sample, it mustn't be picked at any step. Thus it will be (1-1/n)*(1-1/n)…….**(1-1/n) n times.

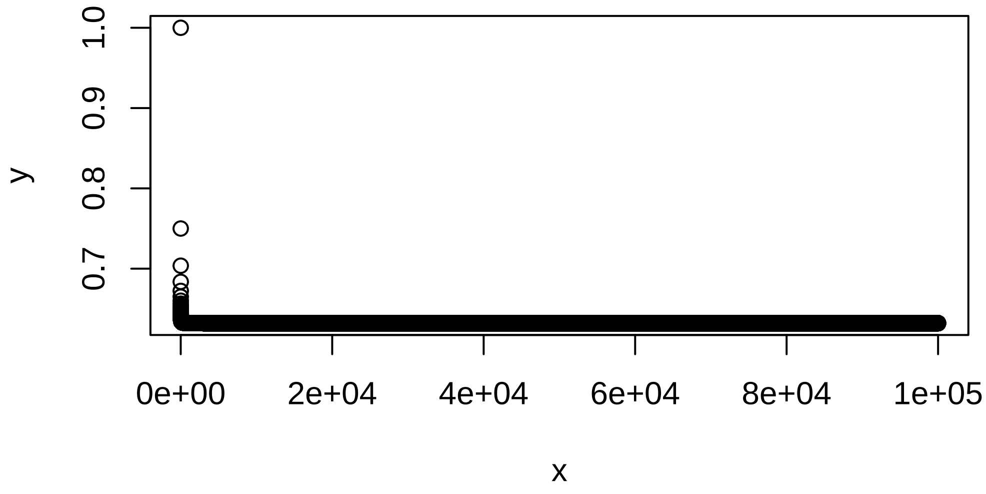
d) 0.67232

e) 0.6339

f) 0.6321389

g)

```r
x <- seq(1, 100000, 1)
y <- 1-(1-(1/x))^x
plot(x, y)
```



The value approaches approx 0.6321

h)

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

```
[1] 0.6374
```

As expected it is around the value we calculated.

Exercise 3

a) We split the training set into k subsamples. We create a model based on every sample, except the ones in the k-th group. After this we run the model on the k-th set, and measure the MSE. We repeat this with each of the groups.

b)

Exercise 7

```
library(ISLR2)
dat <- Weekly
#a
fit1 <- glm(Direction ~ Lag1 + Lag2, data = dat, family = binomial)
#b
fit2 <- glm(Direction ~ Lag1 + Lag2, data = dat[-1,], family = binomial)
#c
if (predict(fit2, newdata = dat[1, 2:3], type = "response") > 0.5){
  pred <- "Up"
  print(pred)
} else {
  pred <- "Down"
  print(pred)
}
```

```
[1] "Up"
```

it was not correctly predicted

d)

```
preds = c()
ers = c()
for (i in 1:nrow(dat)) {
  mod <- glm(Direction ~ Lag1 + Lag2, data = dat[-i,], family = binomial)
  if(predict(mod, newdata = dat[i, 2:3], type = "response") > 0.5) {
    tmp <- "Up"
  } else {
    tmp <- "Down"
  }
  preds[i] <- tmp
  if(tmp == dat[i, 9]){
    ers[i] <- 0
  } else
    ers[i] <- 1
}
```

e)

```
mean(ers)
```

```
[1] 0.4499541
```

Slightly lower than 50 %

Chapter 6

Exercise 1

a)

```
n =  1000
p = 20
X = matrix(rnorm(n*p), n, p)
B = rnorm(20)
B[c(4,7,13,19)] = 0
Y= X %*% B
X <- cbind(X,Y)
colnames(X) <- c(1:20, "Y")
```

b)

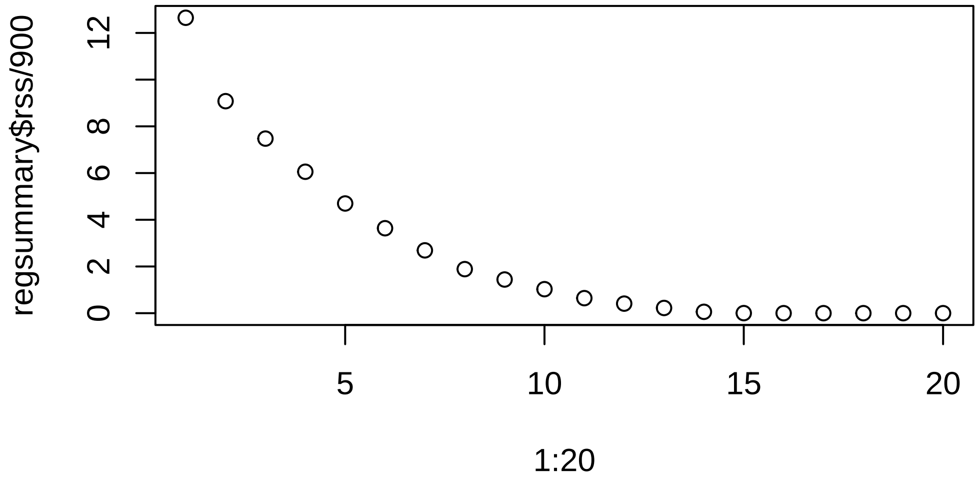
```
test <- sample(1:nrow(X), 100)
X_test <- X[test,]
X_train <- X[-test,]
X_train <- as.data.frame(X_train)
X_test <- as.data.frame(X_test)
```

c)

```
library(leaps)
regsubset <- regsubsets(Y ~., data = X_train, nvmax = 20)
regsummary <- summary(regsubset)
```

```
Warning in log(vr): NaNs produced
```

```
plot(1:20, regsummary$rss/900)
```



```
mse_train <- regsummary$rss/900
```

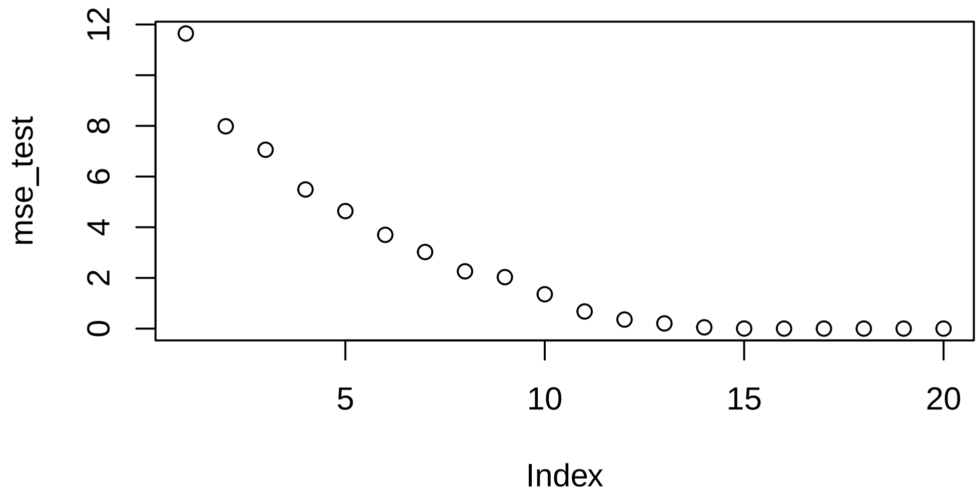
d)

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.0     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
bestpredictors <- regsummary$which
colnames(bestpredictors) <- c("Intercept", 1:20)
bestpredictors <- as.data.frame(bestpredictors)
predictions <- list()
```

```
fits_list <- list()
regs_list <- list()
mse_test <- c()
fits_list <- list()
 for(i in 1:20) {
  vars <- bestpredictors[i,-1]
  regs <- which(vars == TRUE)
  regs_list[[i]] <- regs
  tmp_fit <- lm(Y ~ ., data = X_train[,c(regs, 21)])
  fits_list[[i]] <- tmp_fit
  predictions[[i]] <- predict(tmp_fit, newdata = X_test)
  mse_test[i] <- mean((X_test$Y - predictions[[i]])^2)
}
plot(mse_test)
```



e)

```
which(mse_test == min(mse_test))
```

```
[1] 20
```

f)

```r
summary(fits_list[[18]])$coef
```

```
                 Estimate   Std. Error       t value    Pr(>|t|)
(Intercept) -1.998401e-16 1.672601e-16 -1.194786e+00 0.232492015
`1`          4.247120e-01 1.617951e-16  2.624999e+15 0.000000000
`2`         -4.668803e-01 1.645487e-16 -2.837339e+15 0.000000000
`3`          2.341582e-01 1.632204e-16  1.434614e+15 0.000000000
`4`         -2.242107e-16 1.607895e-16 -1.394436e+00 0.163537475
`5`         -6.998505e-01 1.675749e-16 -4.176344e+15 0.000000000
`6`          8.504465e-01 1.632420e-16  5.209729e+15 0.000000000
`8`          4.156738e-01 1.746534e-16  2.379992e+15 0.000000000
`9`         -1.390081e+00 1.684003e-16 -8.254622e+15 0.000000000
`10`         1.145777e+00 1.641904e-16  6.978344e+15 0.000000000
`11`        -4.787422e-02 1.629740e-16 -2.937537e+14 0.000000000
`12`        -9.398470e-01 1.601449e-16 -5.868730e+15 0.000000000
`13`        -4.477662e-16 1.619536e-16 -2.764781e+00 0.005815095
`14`        -2.443527e+00 1.715052e-16 -1.424754e+16 0.000000000
`15`        -6.624851e-01 1.643573e-16 -4.030761e+15 0.000000000
`16`         6.177619e-01 1.658413e-16  3.725018e+15 0.000000000
`17`        -1.060497e+00 1.632728e-16 -6.495245e+15 0.000000000
`18`         8.986797e-01 1.772673e-16  5.069630e+15 0.000000000
`20`        -1.969895e+00 1.673790e-16 -1.176907e+16 0.000000000
```

```r
B
```

```
 [1]  0.42471199 -0.46688035  0.23415817  0.00000000 -0.69985052  0.85044650
 [7]  0.00000000  0.41567378 -1.39008072  1.14577733 -0.04787422 -0.93984700
[13]  0.00000000 -2.44352669 -0.66248509  0.61776192 -1.06049668  0.89867968
[19]  0.00000000 -1.96989469
```

They are very similar, the 0 values are close to 0, and are not significant.

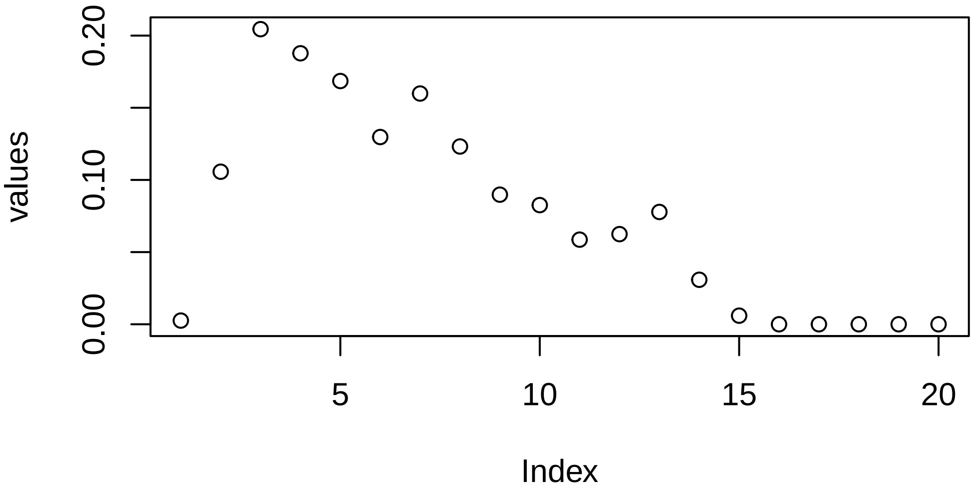
g)

```r
values <- c()
for(i in 1:20){
  values[i] <- sqrt(sum((B[regs_list[[i]]] - summary(fits_list[[i]])$coef[-1, 1])^2))
}
```

```
Warning in summary.lm(fits_list[[i]]): essentially perfect fit: summary may be
unreliable

Warning in summary.lm(fits_list[[i]]): essentially perfect fit: summary may be
unreliable
```

```
plot(values)
```


values
Index