**Statistical Learning 6A**

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 *j*th observation from the original sample? Justify your answer.

**Bootstrapping draws sample with replacement, which means each subset has equal chance of getting drawn. Drawing exactly the *j*th subset has probability of 1/n, and thus not drawing it is (1-1/n).**

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

**Bootstrapping draws sample with replacement, which means each subset has equal chance of getting drawn, and thus the sequence of drawing do not affect the probability of drawing a particular subset. The second bootstrap observation still has probability of (1-1/n) to not be the *j*th subset.**

(c) Argue that the probability that the *j*th observation is *not* in the bootstrap sample is (1 *−* 1*/n*)*n*.

**As mentioned above, a Bootstrap process draws subsets independently. If we got a Bootstrapped sample without *j*th subset. That means repeating n times of drawing, the *j*th subset repeatedly never been drawn. The probability of that is (1-1/n) to the nth power, or (1-1/n)^n.**

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

**As demonstrated above, a bootstrap process without the *j*th subset has probability of (1-1/n)^n. So a bootstrap sample contains at least one *j*th subset has probability of 1-(1-1/n)^n, and when n=5, 1-(1-1/5)^5=0.67232.**

(e) When *n* = 100, what is the probability that the *j*th observation is in the bootstrap sample?

**1-(1-1/100)^100=0.6339677**

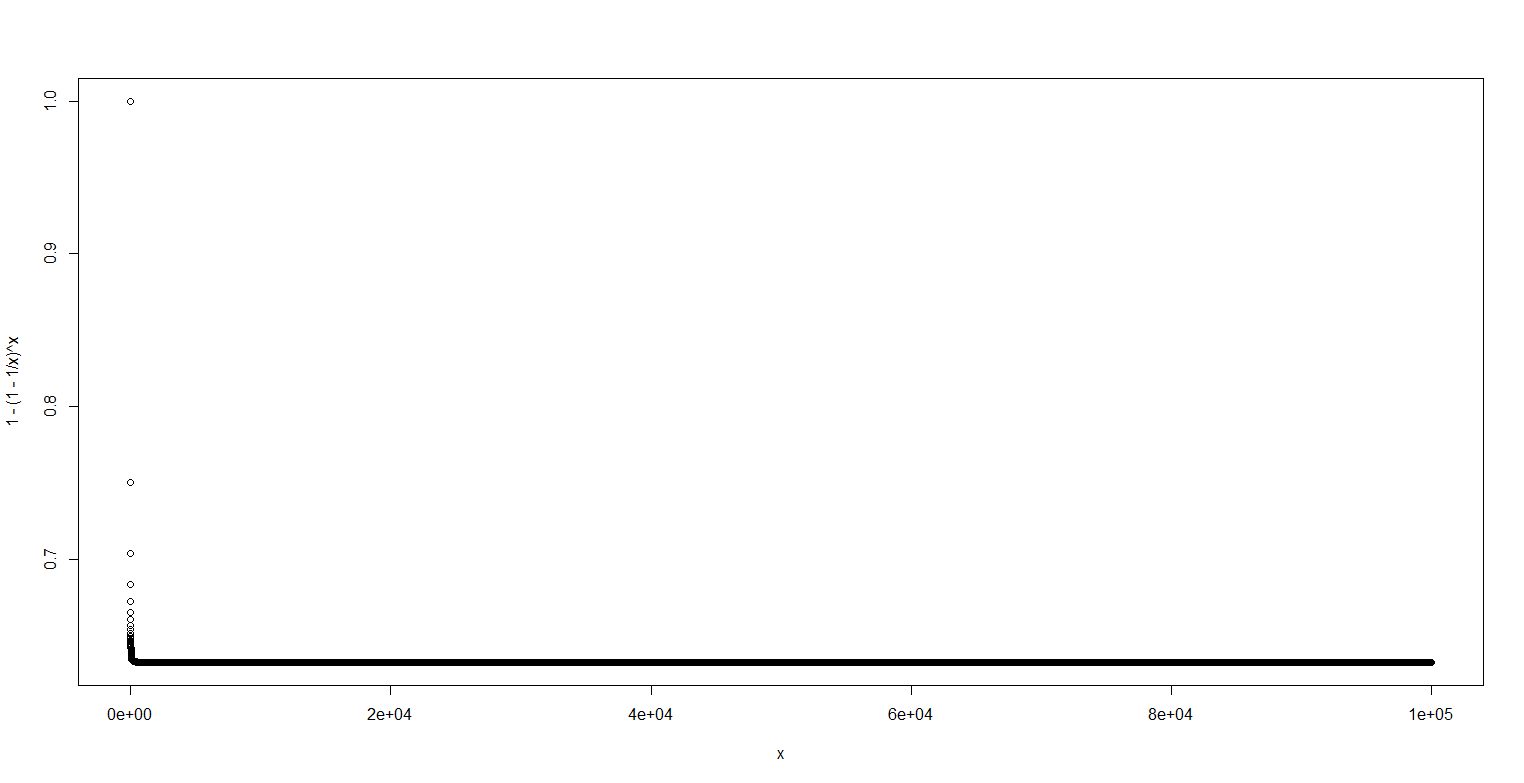
(f) When *n* = 10*,* 000, what is the probability that the *j*th observation is in the bootstrap sample?

**1-(1-1/10000)^10000=0.632139**

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

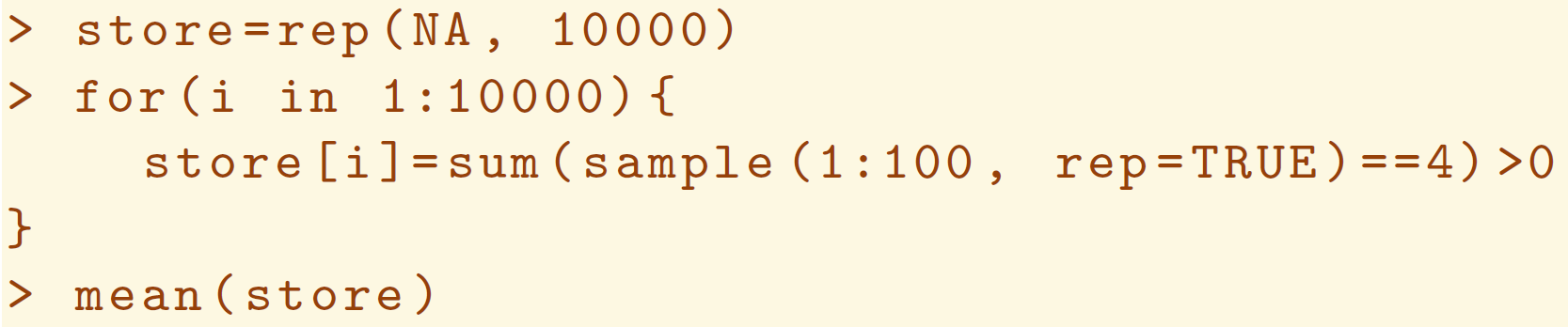
**x <- 1:100000**

**plot(x, 1 - (1 - 1/x)^x)**



**We can see that as n increases, the probability converges quickly. This explains when we change n from 5 to 10000, the probability did not change much.**

(h) We will now investigate numerically the probability that a bootstrap sample of size *n* = 100 contains the *j*th 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.



Comment on the results obtained.

**Each time this process give different answer, but all converge around 0.63. This is consistent with what we found in part (g).**

3. We now review *k*-fold cross-validation.

(a) Explain how *k*-fold cross-validation is implemented.

**We randomly divide our dataset into k subsets, each are not overlapping and roughly the same size. We pick one subset to be test subset, and use the rest as the training subset. For each subsets, we calculate and record the test errors, then repeating the process again with another subset to be test and remaining as controls, until each subsets has been used as test subset once. Then, we calculate and report the mean test error as the crossvalidation error.**

(b) What are the advantages and disadvantages of *k*-fold crossvalidation relative to:

i. The validation set approach?

**Rather than arbitrarily break the dataset into test and train, K-fold crossvalidation allows us to, in a sense, use the whole dataset as test and training, thus reduces test error variance with minimum concern of overfitting. However, the bias maybe higher due to that we used the whole dataset to train the data.**

ii. LOOCV?

**LOOCV is like k-fold when k=n, that individual observations are used to be the test instead of a subset. So obviously variance will be much higher than the k-fold, but less biased. However, this approach is extremely calculation intensive.**

6. 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.

**library(ISLR)**

**attach(Default)**

**set.seed(1)**

**Q6.fit.glm <- glm(default ~ income + balance, data = Default, family = "binomial")**

**Summary(Q6.fit.glm)**

**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 \*\*\***

(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.

**boot.fn <- function(data, index) {**

**fit <- glm(default ~ income + balance, data = data, family = "binomial", subset = index)**

**return (coef(fit))**

**}**

(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.

**library(boot)**

**boot(Default, boot.fn, 1000)**

**Bootstrap Statistics :**

**original bias std. error**

**t1\* -1.154047e+01 -8.008379e-03 4.239273e-01**

**t2\* 2.080898e-05 5.870933e-08 4.582525e-06**

**t3\* 5.647103e-03 2.299970e-06 2.267955e-04**

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

**These std.errors are consistent with the glm estimates.**

7. In Sections 5.3.2 and 5.3.3, we saw that the cv.glm() function can be used in order to compute the LOOCV test error estimate. Alternatively, one could compute those quantities using just the glm() and predict.glm() functions, and a for loop. You will now take this approach in order to compute the LOOCV error for a simple logistic regression model on the Weekly data set. Recall that in the context of classification problems, the LOOCV error is given in (5.4).

(a) Fit a logistic regressionmodel that predicts Direction using Lag1 and Lag2.

**set.seed(1)**

**attach(Weekly)**

**Q7.fit.glm <- glm(Direction ~ Lag1 + Lag2, data = Weekly, family = "binomial")**

**summary(Q7.fit.glm)**

**Coefficients:**

**Estimate Std. Error z value Pr(>|z|)**

**(Intercept) 0.22122 0.06147 3.599 0.000319 \*\*\***

**Lag1 -0.03872 0.02622 -1.477 0.139672**

**Lag2 0.06025 0.02655 2.270 0.023232 \***

(b) Fit a logistic regressionmodel that predicts Direction using Lag1 and Lag2 *using all but the first observation*.

**Q7.fit.glm.2 <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-1, ], family = "binomial")**

**summary(fit.glm.1)**

**Coefficients:**

**Estimate Std. Error z value Pr(>|z|)**

**(Intercept) 0.22324 0.06150 3.630 0.000283 \*\*\***

**Lag1 -0.03843 0.02622 -1.466 0.142683**

**Lag2 0.06085 0.02656 2.291 0.021971 \***

(c) Use the model from (b) to predict the direction of the first observation. You can do this by predicting that the first observation will go up if *P*(Direction="Up"*|*Lag1, Lag2) *>* 0*.*5. Was this observation correctly classified?

**predict.glm(Q7.fit.glm.2, Weekly[1, ], type = "response") > 0.5**

**1**

**TRUE**

**Weekly[1, ]**

**Year Lag1 Lag2 Lag3 Lag4 Lag5 Volume Today Direction**

**1 1990 0.816 1.572 -3.936 -0.229 -3.484 0.154976 -0.27 Down**

**The observation is incorrectly classified as the true first observation is Down.**

(d) Write a for loop from *i* = 1 to *i* = *n*, where *n* is the number of observations in the data set, that performs each of the following steps:

i. Fit a logistic regression model using all but the *i*th observation to predict Direction using Lag1 and Lag2.

ii. Compute the posterior probability of the market moving up for the *i*th observation.

iii. Use the posterior probability for the *i*th observation in order to predict whether or not the market moves up.

iv. Determine whether or not an error was made in predicting the direction for the *i*th observation. If an error was made, then indicate this as a 1, and otherwise indicate it as a 0.

**error <- rep(0, dim(Weekly)[1])**

**for (i in 1:dim(Weekly)[1]) {**

**fit.glm <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ], family = "binomial")**

**pred.up <- predict.glm(fit.glm, Weekly[i, ], type = "response") > 0.5**

**true.up <- Weekly[i, ]$Direction == "Up"**

**if (pred.up != true.up)**

**error[i] <- 1**

**}**

**error**

(e) Take the average of the *n* numbers obtained in (d)iv in order to obtain the LOOCV estimate for the test error. Comment on the results.

**mean(error)**

**0.4499541**

**This is the LOOCV test error. It has 45.00% error rate.**