Paige Blomstedt, Addison Roden

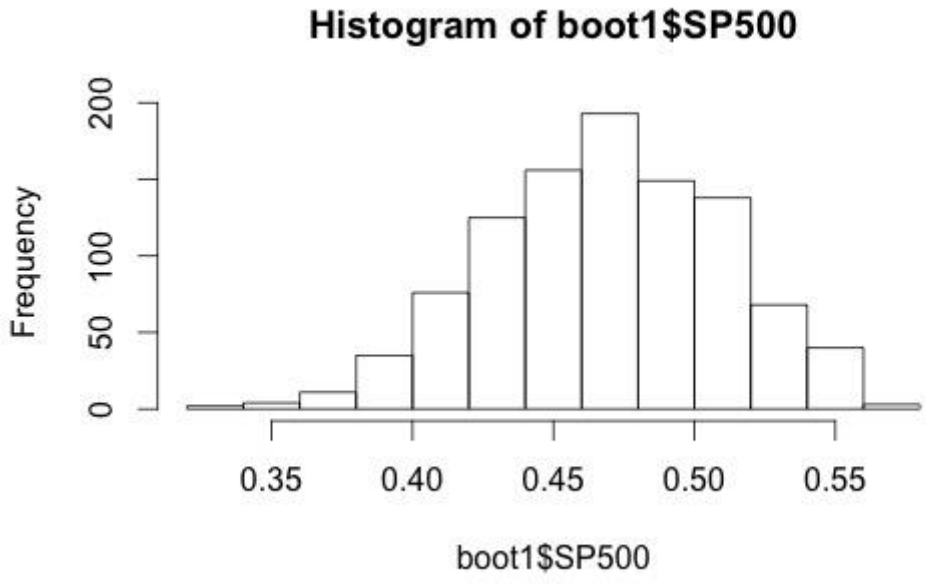
February 10, 2014

Homework review

1. Bootstrapping
    a. In a previous homework, we found a model for the relationship between Walmart's stock returns and those of the S&P 500. A variable that predicts the relationship between these two, or beta, was found to be .47 from the data, but the beta given by Yahoo Finance was .33. Initially, this difference was attributed to sampling variability; however, through the use of bootstrapping, we can see that the difference in betas cannot be attributed to sampling variability because the standard error is only .04 :

    sd(boot1$SP500) = 0.04294441

    A histogram of bootstrapped samples shows that the variance between Yahoo's beta of .33 and the average beta of .47 cannot be attributed to sampling variability. Finding the 95% confidence interval of the beta for the data given shows that .33 is not included.



    b. Part b of this homework question looked at data for an incoming freshman class at UT. The data was divided by which school the student was admitted into. The mean for each data set can easily be found in R: **mean(SAT.Q ~ School, data=ut2000)**. This mean is referred to as the "straight up" mean because it gives only the mathematical average of each group. Another method of finding the mean is the "baseline offset" method. In this case, one college is used as a baseline, and the means for the other colleges are described by how much they vary from the initial college, or the baseline. The question asked for a comparison of SAT scores from students in the Architecture school and the

Liberal Arts school. Using architecture as the baseline, we can use the baseline offset method and the r command **lm(SAT.Q ~ School, data=ut2000)** to find the difference between scores in the architecture school (Intercept) and those in other schools (SchoolEDUCATION, SchoolLIBERAL ARTS, etc.).

i. In order to generalize to a larger population, we can find a more accurate mean for each school by sampling with replacement, or bootstrapping. The sample is bootstrapped 1000 times: **boot2 = do(1000)*lm(SAT.Q ~ School, data=resample(ut2000))**, and then a confidence interval of these 1000 samples was found: **confint(boot2)**.

ii. Another way to ensure the validity of the confidence interval is by creating subsets: **architecture=subset(ut2000, School=="ARCHITECTURE"),** and **liberal.arts=subset(ut2000, School=="LIBERAL ARTS").** Bootstrapping resamples the whole data frame, and therefore does not guarantee the same number of students in each school for each sample. Creating a subset for each college ensures that there will be a consistent number of students in each sample for each college. For example, if there are 32 architecture students in the sample, creating a subset ensures that each resample also has 32 architecture students. The r command for creating a resample is **resample(architecture).** After creating a subset for each college, you resample the data set for each college, and these resample data sets can be used to form a more accurate group mean. Once a mean is found for each subset, you can compute the difference in the means in an r command similar to **boot3 = do(1000)*(mean(resample(architecture)$SAT.Q) - mean(resample(liberal.arts)$SAT.Q)),** and finding the **confint(boot3)**.

2. Parameter and prediction uncertainty in the normal linear regression model

a. In order to determine whether the company should use the cheap test, use the (r^2) statistic because it is the fraction of the total variation in the data that is predicted using the x-variable. Looking at the graph of the expensive test as predicted by the cheap test, you can see that there is some variability in x, or the cheap test. The residual (r^2) is found to be larger than 90%, indicating the cheaper test is acceptable to use. R^2 (0.934) is found by using the following commands in R: **lms=lm(expensive~cheap,data=shocks) and >summary(lms)$r.squared**

b. If the company is to use the cheap test, it needs to be sure that the cheap test does not systematically under- or over-predict the data. A confidence interval for the intercept and the slope of the expensive test on the cheap test, **confint(lms),** shows that 1 is a reasonable assumption for the slope. Next, the prediction interval needs to be narrow enough to closely predict the results of the expensive test. We have been using a naïve prediction interval, meaning we have not taken into account future uncertainty of the error and the parameters. This uncertainty is accounted for using this formula from page 117 of the course packet:

$$\widehat{\beta}_0 + \widehat{\beta}_1 x^\star \pm t^\star \hat{\sigma} \left\{ 1 + \frac{1}{n} + \frac{(x^\star - \bar{x})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \right\}^{1/2}$$

The prediction intervals also need to be taken into account when determining if the cheap test is an accurate predictor. The questions states that if the 95% prediction interval for the value of the expensive test, as given by the cheap test, is no wider than 16.5 units of rebound. To find this in r, take the fitted model (lms), feed it the new data, **newdata=data.frame(cheap = c(510,550,590)),** ask for a prediction interval with a confidence level of 95%, and it will give you upper and lower bounds.

>**Predict(lms,newdata,interval="predict",level=.95)**

The prediction interval for the highest point does not fall within 16.5 units of rebound, so the interval is too wide to be acceptable. The width of the prediction interval changes throughout the interval when we move from the naïve to the actual prediction interval because the systematic information changes. Sampling distributions get buffeted more around the outer end points of the data, and less in the middle. By taking more data, part of the predictive uncertainty that you cannot drive to zero becomes the constants B_0 and B_1. Residual uncertainty will always exist, and although it cannot be driven down to zero, the systematic variation in data becomes zero. The values of B_0 and B_1 will become arbitrarily tight around the true values, which allows us to use B_0 and B_1 in prediction equations.

3. Cross Validation
    a. General idea: cut the data set in half, using one half to fit a model and one half to see if the model accurately predicts data. Assign one half of the data to be data that you "have" seen before, and use it to fit a model. Using your model, try and accurately predict the other half of the data, the half that was assigned to be what you "haven't" seen before. This allows us to mimic out-of-sample variation.
    b. Call one data set the training set and one data set the testing set.
    Trainset = sample(1:N, Ntrain). Trainset returns the data that you've seen before. Whatever isn't included in this training set is in the testing set.
    c. Next, pick out the appropriate rows of the data frame for the training and testing sets
    Util.train = utilities[trainset,]
    Util.test = utilities[-trainset,]
    d. Fit the linear and quadratic model using only the training data set.
    lm1 = lm(gasbill~temp,data=util.train)
    lm2 = lm(gasbill~temp+I(temp^2),data=util.train)… etc.
    e. Predict on the testing data set (the data you're pretending you haven't seen before).
    pred1 = predict(lm1,newdata=util.test)
    pred2 = predict(lm2,newdata=util.test)… etc.
    f. Compute each model's mean squared error.
    mse1 = mean((pred1 - util.test$gasbill)^2)
    mse2 = mean((pred2 - util.test$gasbill)^2)… etc.
    g. Now you can use the "for" loop in order to execute the same set of commands over and over again under slightly different stipulations. This ensures the randomness of choosing which cases get placed into the training set is averaged out over several trials.

```
mysim = do(1000)*{
 trainset = sample(1:N, Ntrain)
 util.train = utilities[trainset,]
 util.test = utilities[-trainset,]

 lm1 = lm(gasbill~temp,data=util.train)
 lm2 = lm(gasbill~temp+I(temp^2),data=util.train)
 lm3 = lm(gasbill~temp+I(temp^2)+I(temp^3),data=util.train)
 lm4 = lm(gasbill~temp+I(temp^2)+I(temp^3)+I(temp^4),data=util.train)

 pred1 = predict(lm1,newdata=util.test)
 pred2 = predict(lm2,newdata=util.test)
 pred3 = predict(lm3,newdata=util.test)
 pred4 = predict(lm4,newdata=util.test)

 mse1 = mean((pred1 - util.test$gasbill)^2)
 mse2 = mean((pred2 - util.test$gasbill)^2)
 mse3 = mean((pred3 - util.test$gasbill)^2)
 mse4 = mean((pred4 - util.test$gasbill)^2)
 c(mse1, mse2, mse3,mse4)
}
```

The loop gets a different split every time that it carries out these commands. To see which model (linear, second-order, third-order, etc.) did best, see which one has the smallest mean squared error (mse).

h. Huge take-away regarding "for" loops:
Write pseudo code first rather than actual code.
→The pseudo code is the English description of what R is doing, indicated after "#" in R

New Material

Linear Regression Assumptions: these assumptions can be true or false, but if they are false, you cannot trust the model.

1. Normality
2. Independence
    a. If a gas-consumption model shows gas consumption by season, the variance in winter is consistently higher than that in summer; there is not independence because there is a trend that can be predicted based on points in near proximity.
3. Constant variance (homoscedasticity): every residual is the drawn from the same distribution with the same variance
    a. If the residuals don't all have the same variance, there is a fan shape at either end.

T-statistic

T-statistics were introduced at the end of class. T-statistics are found by dividing the estimate for B_0 by the standard error of the estimate, and they give us a signal to noise ratio. They will be further discussed in subsequent classes.