1/22/14 Class Notes

Admin Info:

- You can be in homework groups with people in other sections.
- You don't have to memorize R commands for the exams.
- Calculus will be utilized in this class, but you don't have to remember the details like the Fundamental Theorem of Calculus.
- Laptops must be closed unless the class is working in RStudio or you are scribing for the day.
- You shouldn't print R scripts for homework.
- Dr. Scott has office hours from 3:30-4:30 on MW. David (TA) has office hours on Wednesdays from 5:30-7 in CBA 4.304.
- When importing a dataset, make sure to check for headers at the top. If they exist, click the yes button on the "Header" row. This ensures that the data is correct and can be used in R effectively.

Homework 1 Review:

1. The concept of the first question was to try to read past the journalist's interpretation to see the data. When you are reading a newspaper article, it is always good to try to analyze the data and what could be factors behind it and understand it well even through the journalist's summary because sometimes it isn't the study at fault, but the writing
2. The second question covered many concepts we learned in the previous session. For example, you had to specify that the "Year" variable is categorical, which was reviewed in the Mammal Sleep video posted on the website. The data was designed to be analyzed easily, so the key was to not overthink it.
3. The final question is asking you to find the minimum of the Sum Squared Errors and what this term was. Theta hat is the guess of what the number could be. You are subtracting theta hat from the actual because you want to see what the difference is. You square it to make sure that going over is the same as going over. We then sum it to get the sum of squared errors. The SSE is a parabola with a positive second derivative (looks like a smile). The derivative of the equation ends up being y bar, or the mean. The problem was used to let us understand why you use the mean when you are estimating group data (it is where the errors are at a minimum). There is no penalty for incorrect algebra, as long as you know what needs to be done.

Class Notes:

Wednesday's class mainly dealt with linear regressions. These help create predictions based on current data. We also started looking at "stories", which help us analyze these equations to let us better understand what the data is saying. To work through the problems, one should download the pickup.csv and pickup.R files from the website.

One of the basic goals of statistical modeling is to partition data. We want to understand what the variability is and where is comes from. We need to make sure we look back and see what variability

actually is. The pickup.R data is information from Craigslist of pickup trucks for sale in Austin. Using the hist(pickup) function, we can see the variability in the data. Y1 to YN are data points that show prices in trucks.

If we draw ten dots and imagine they are on a price axis, we can try to understand the coverage interval (or what range covers the percentage you want), in which an 80% coverage interval would cover all but two dots. This will come up again in the context of confidence intervals. Confidence intervals are more complicated than coverage intervals, since the coverage interval concept is very basic. The standard deviation helps demonstrate this concept since it shows on average how far the actual value is from the predicted (or the mean). We make sure to square it so the positive and negative differences don't cancel each other out. The formula is the summation from i=1 to N of (yi – y bar)^2 divided by (N-1). The standard deviation of y is the square root of the equation because you don't want the error or the values to be squared (if the units are elephants, you don't want squared elephants). The standard deviation is the average error.

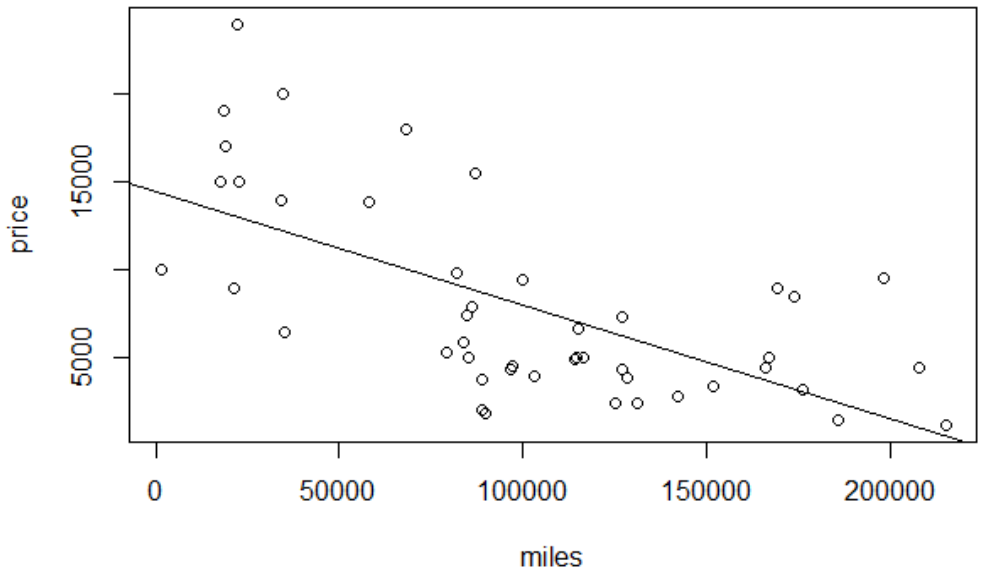
We partition variability, but we want to understand what we mean by variability. We usually use standard deviation to see this variability. To computer standard deviation, it's sd(dataset$variable) or (sd(pickup$price)) for the pickup truck data used in class. The standard deviation of data can help show how far the data extends when added or subtracted from the mean. To find the mean using R code, we use mean(dataset$variable). In the case of the data used in class, the R code is mean(pickup$price).

Furthermore, to find the spread of the data, we can use quartiles to examine what data fits at various percentage points. To do this in R, we use the function called Qdata. The code for this function is qdata(quantile, variable, data=dataset). So if we want multiple quartiles, we say qdata(c(quantile1, quantile2, quantile3), variable, data=dataset). A lowercase "c" means combine in R.

How to create and examine linear regression models:

There are two very common ways to present groupwise data. One is to show all of the groupmeans as a table or as they appear in R (straight-up). The other is baseline/offset form. For the first type, we pull up the groupmeans using the mean function and stratifying it like so: mean(variable1~variable2, data=dataset). This should display a table of groupmeans.

To use the baseline/offset method, you must first find the linear model. In R, the function is lm(variable1~variable2, data=dataset) (the lm stands for linear model). It is usually best to set the linear model equal to a variable, named something to the effect of lm1 or model1 because it helps with the nest few steps. Once you have the linear model ready, you must get the coefficients, fitted values, and residuals to see how the linear model represents the data. The code for these functions are coef(lm1), fitted(lm1), and resid(lm1), respectively. You then use cbind(groupmeans, coef(lm1)) to get the baseline (groupmean) and what the offset is. The offset is the difference between the baseline and the groupmean for that group. Statistical modeling is all about differences, whether between groups or points in the data. The baseline/offset method shows it all without having to further manipulate anything. Then, you would plot the data in a scatterplot using the plot function (code is plot(variable1~variable2, data=dataset) and display the trend line using the code abline(model1).



Any line fits to the equation y = intercept + slope * x. No straight line fits the data perfectly, though. Yi = Bo + B1xi + ei where Bo + B1xi is equal to the trend model value fitted value and the ei is the residual value. i means the cases (B1x1, B1x2, etc.). The residual is the difference between the predicted value from the regression line and the actual data. You want to minimize the summation of ei^2 because this means that the regression line is closely related to the data. This is from the work of a French man name Legendce. We could derive the formula of yi but we're not going to because it involves a lot of algebra. The Pythagorean Theorem also proves this assumption. Now that we recognize that the regression line provides the least sum squared errors, we can go to computing it.

Stories:

1. Once you have the scatterplot up and the line fitted, you can use the function abline(model1) to have the line appear on the scatterplot. There are four stories you can see from the data. Some of the stories are complex and require "digesting" while other don't. The first is plug-in prediction. If we were to try to predict the price of a truck based on its mileage, we would try to follow the least regression line from the x axis to the line to the y axis. We plug in one variable and see the other variable value. If we were to predict a new truck, we can use the function newx = x-value and yhat = intercept + (slope)*newx. If we wanted to do this with 3 trucks, we do the same thing, but with newx = c(x1, x2, x3) and use the same yhat function. This will give you three new values.
2. If we take the 46 trucks and create a line about it (lowering the amount of numbers from 92 to 2), we can summarize the trend, which is story number 2. The intercept is where the line hits the y axis. We can do this using the coef function. The intercept can become tricky when dealing with years because you are technically predicting what happens in year 0. There are many ways

we can see what the slope variable means. Yhati = Bo + B1xi when yi = yhati + ei. Dyi/dxi = 0 + B1 = B1 since B1 is the slope. So for the pickup trucks, the slope means the change in price for the change in mileage. If B1 = .065 for the truck price, price = price + miles*B1. So the units the B1 would need to be in to keep the left price would be price per miles. Story number two is just analyzing what the slope and intercept in the line mean. You can use this information to predict when you should get rid of your car, for example, if you use your car for another 10000 miles, you would lose ~$642.

3. Story #3 is taking the x-ness out of Y (statistical adjusting). When you are comparing things, you get the sense that things are being adjusted but don't know what that means. In the pickup truck case, we would be adjusting the price without miles. If you want to extract the residuals, you can use the resid(model1) function. When there is a systematic trend, there is x-ness in y. To remove the x-ness, plot the residuals. This removes the trend and allows you to see the adjusted data. This allows you to see things without the influence of the x-axis and the trend. If you are looking for a minimum, as you would with price, you want to search for the smallest (or largest negative) residual. From this, you can find a good deal on the pickup trucks since you would need to find the minimum of price. To find the best deal, you can use the min R function for the residuals to display the smallest or most negative residual. This is the best deal because the largest negative residual should give the truck with the best price for the miles. To see which truck is the one with the smallest residual, one must code which.min(resid(model1)), which will give the row number of the table that corresponds to that truck.
4. Story #4 is reducing uncertainty, which is using the standard devation. The functions to use are sd(pickup$price) and sd(resid(model1)). This shows that the uncertainty has been reduced. We will go over this more in-depth on Monday.

After doing stories 1-3, the RStudio should show something similar to this: