Linear Modeling with Two Categorical Predictor Variables

ANTH 3720-001 Archaeological and Forensic Science Lab Methods: Data Analysis with R

Elic Weitzel

Jan. 29-31, 2021

If you would like the original R Markdown file, find it on my GitHub page at https://github.com/weitzele/Basic-R-Tutorial

1 Linear Modeling with Two Categorical Predictor Variables

One of the most common questions that scientists ask of their data is whether two groups of observations differ from each other. Archaeologists might be interested in whether the size of carbonized seed remains is larger at one site and smaller at another. Forensic anthropologists might be curious as to whether the stature of one skeletal population was smaller than another. In such cases, our data are grouped into categories and we're interested in whether the existence of these categories predicts variation in our data.

To begin, we're going to use data that are normally-distributed. This is of course not common for many types of data as normal data are unbounded and continuous, but certain types of data like spatial coordinates, stable isotope delta values, and other measurements that can range on either side of a baseline often fit these assumptions.

So let's load the `archdata` package once again:

```
library(archdata)
```

We're going to analyze some data from the `BarmoseI.pp` data set within this package. So let's use the `data()` function to load this dataset. Then, if you run `?BarmoseI.pp` you'll read that these are two dimentional spatial coordinates for 473 piece plotted artifacts from a site in Denmark.

```
data(BarmoseI.pp)
```

```
?BarmoseI.pp
```

Let's inspect the first few rows of each column in this data frame, and then inspect the values of the `Label` column in greater detail:

```
head(BarmoseI.pp)
```

```
##   North East Class    Label
## 1  5.50 3.75     1 Scrapers
```

```
## 2  7.92 1.35     1 Scrapers
## 3 10.50 1.92     1 Scrapers
## 4  8.50 2.42     1 Scrapers
## 5  8.28 3.42     1 Scrapers
## 6  8.35 3.85     1 Scrapers
```

```
table(BarmoseI.pp$Label)
```

```
##
##                     Scrapers                       Burins
##                           38                           25
##       Lanceolate Microliths                  Microburins
##                           36                           16
##                   Flake Axes                    Core Axes
##                           28                            4
##                Square Knives           Blade/Flake Knives
##                          192                           18
## Denticulated/Notched Pieces                        Cores
##                           26                           81
##              Core Platforms
##                            9
```

We can see that this data frame contains Northing and Easting coordinates (relative to an arbitrary datum for this site, not a global coordinate system), a vector of Class codes used by the origial excavator, and a column of artifact types called `Label`. In this latter column, we see the various types of artifacts for which we have spatial information.

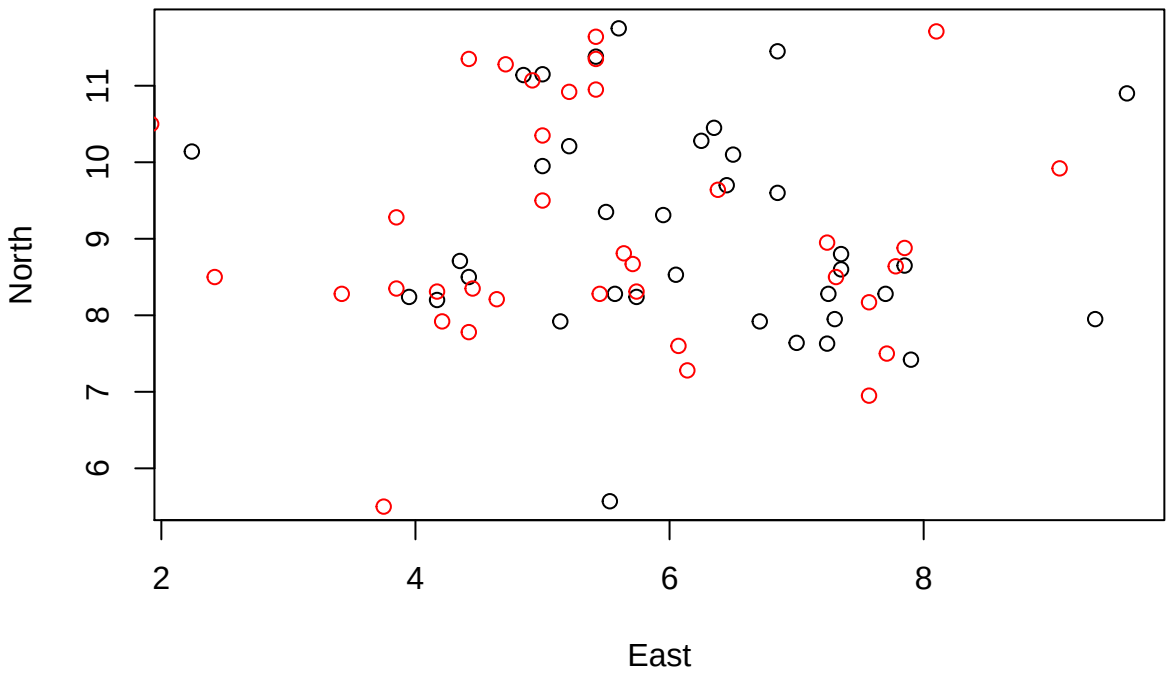
Now, to build our first model, let's say we're interested in the east-west pattern of artifacts at this site. Sites often have spatial patterns in artifact distribution that can reflect past activity areas. So let's say we are curious to know whether scrapers are located further east within this site than lanceolate microliths are.

Let's start by subsetting our data frame to only include the rows pertaining to our two artifact types of interest: scrapers and lanceolate microliths. We can do this by indexing the `BarmoseI.pp` object using logical operations. The following code tells R to pull out only the rows of `BarmoseI.pp` for which the `Label` column contains values of "Lanceolate Microliths" or "Scrapers". The `|` symbol denotes **or** in a logical operation, and remember that `==` is needed to denote equality. Keep in mind that indexing in R is case sensitive, so make sure you're referring to the output from `table(BarmoseI.pp$Label)` to make sure you spell the artifact type correctly.

```
Barm.data.sub <- BarmoseI.pp[BarmoseI.pp$Label == "Lanceolate Microliths" |
                               BarmoseI.pp$Label == "Scrapers", ]
```

Now we have a new data frame object called `Barm.data.sub` that only contains spatial coordinates for the two artifact types that we're interested in. Let's plot these coordinates to visually inspect the spatial patterns in these data. Visualizing this sort of data also shows us that R is perfectly capable of plotting spatial data, and analyzing it as well. One need only input northing coordinates on the y-axis and easting coordinates on the x-axis to produce a map, like this:

```
plot(North ~ East, data = BarmoseI.pp[BarmoseI.pp$Label == "Lanceolate Microliths", ])
points(North ~ East, data = BarmoseI.pp[BarmoseI.pp$Label == "Scrapers", ],
       col = "red")
```



In this very simple map, we can see that lanceolate microliths, plotted in black, are scattered across the site with most falling between 4 and 8 easting. Scrapers, shown in red, have a similar spatial pattern.

Let's compare the easting coordinates for each artifact type:

```
summary(BarmoseI.pp[BarmoseI.pp$Label == "Lanceolate Microliths", ]$East)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.240   5.192   6.150   6.154   7.242   9.600
```

```
summary(BarmoseI.pp[BarmoseI.pp$Label == "Scrapers", ]$East)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.350   4.420   5.420   5.532   7.025  10.920
```

It looks like the mean and median coordinates for lanceolate microliths (plotted in black) are slightly further east than those for scrapers. But is this a statistically significant difference? We need to model the data to find out!

1.1 Building a Model

Let's do this using the basic linear model function in R: `lm()`. The syntax for this function can be viewed by running `?lm`, and there are a few different forms that you can use to specify such a model. We'll use the syntax I prefer, which requires you to specify the response variable *as a function of* the predictor variable. This is written as `y ~ x` with the `~` symbol meaning "as a function of".

In this example, we're interested in whether being a scraper or being a lanceolate microlith better predicts the easting coordinate of an artifact. We therefore have a continuous, unbounded response variable - the easting location - and a categorical predictor variable taking one of two forms - artifact type.

```
mod1 <- lm(East ~ Label, data = Barm.data.sub)

mod1
```

```
##
## Call:
## lm(formula = East ~ Label, data = Barm.data.sub)
##
## Coefficients:
##                 (Intercept)  LabelLanceolate Microliths
##                      5.5321                       0.6218
```

With the subsetted Barmose data `Barm.data.sub`, which includes coordinates for only lanceolate microliths and scrapers, we have now built a model that predicts the effect of `Label` on `East`. In essence, this model tells us whether the mean easting coordinates for lanceolate microliths is significantly different than the mean easting coordinates for scrapers.

1.1.1 Model Diagnostics

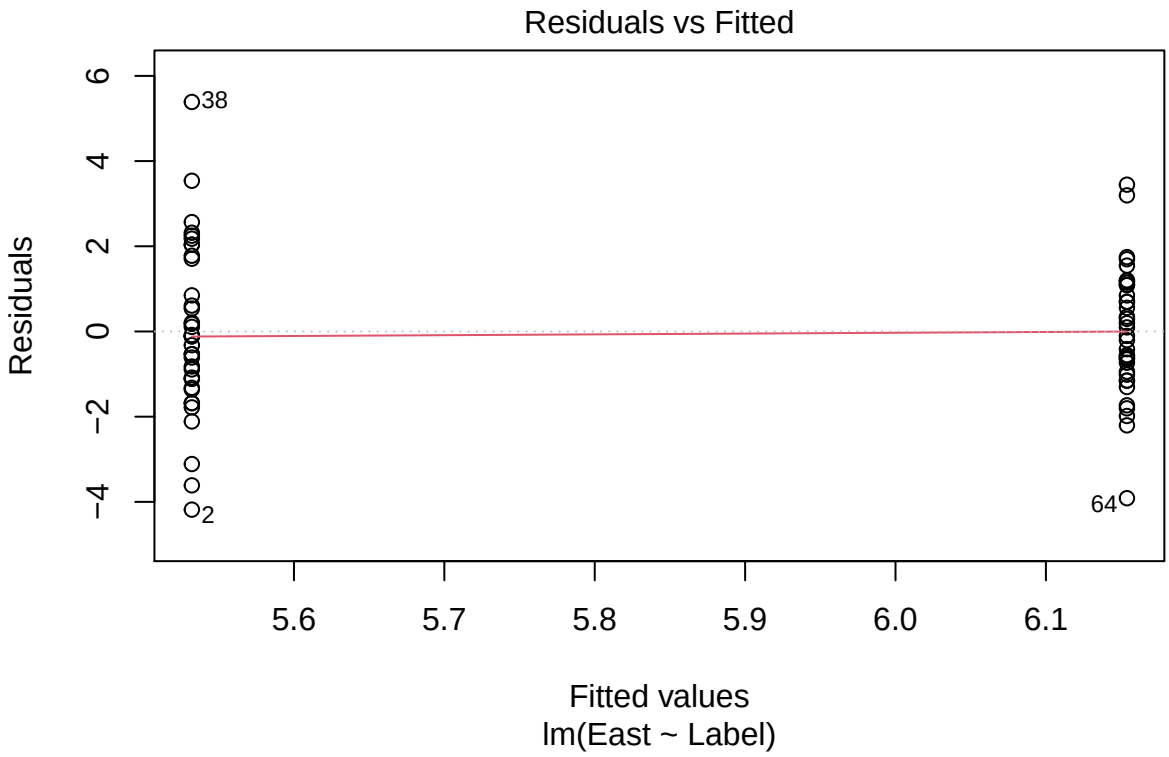
But before we assess the significance of this model, we need to first make sure we're not violating any of the assumptions of linear regression.

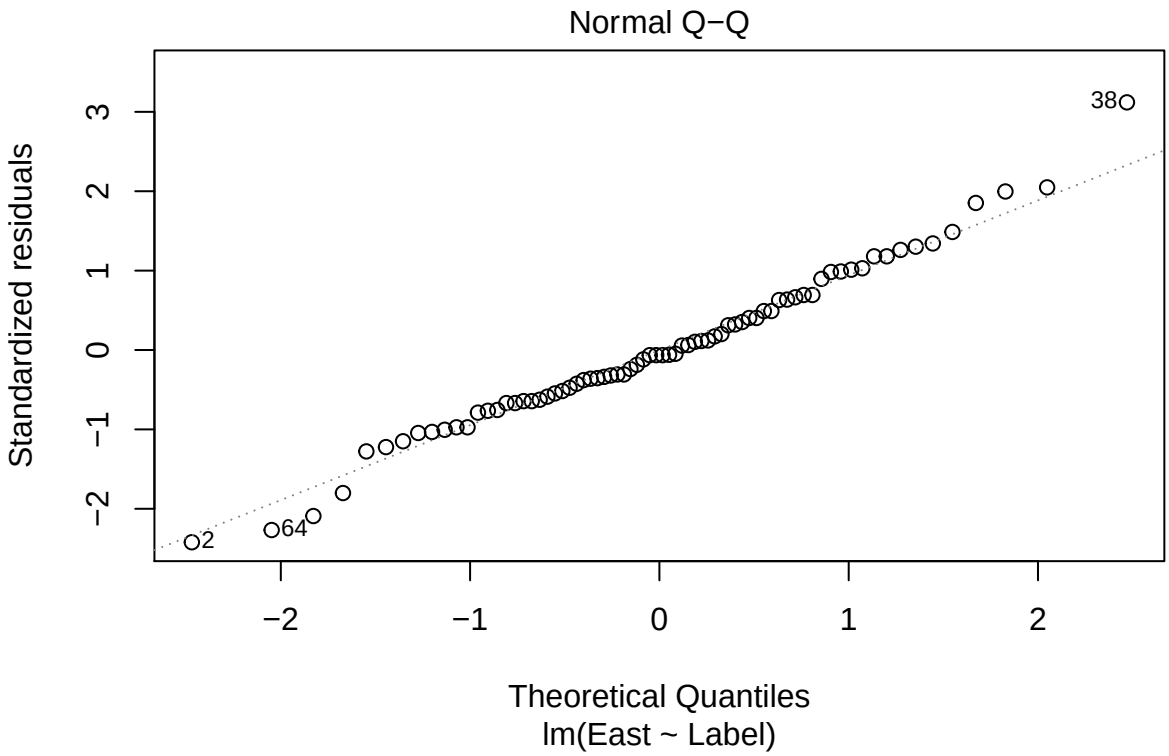
Remember that linear models make several assumptions. They assume that:

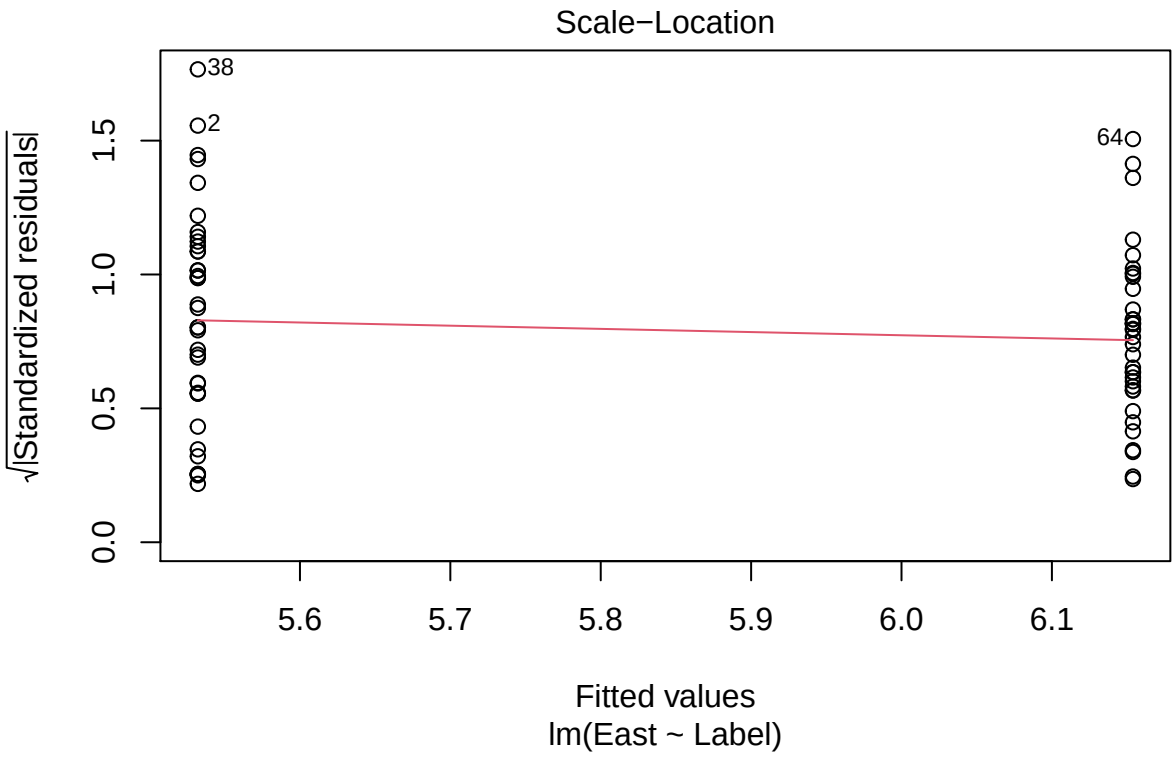
1. The model makes real-world sense
2. Additivity
3. Linearity
4. Independent errors
5. Homoskedasticity (equal variance of errors)
6. Normality of errors

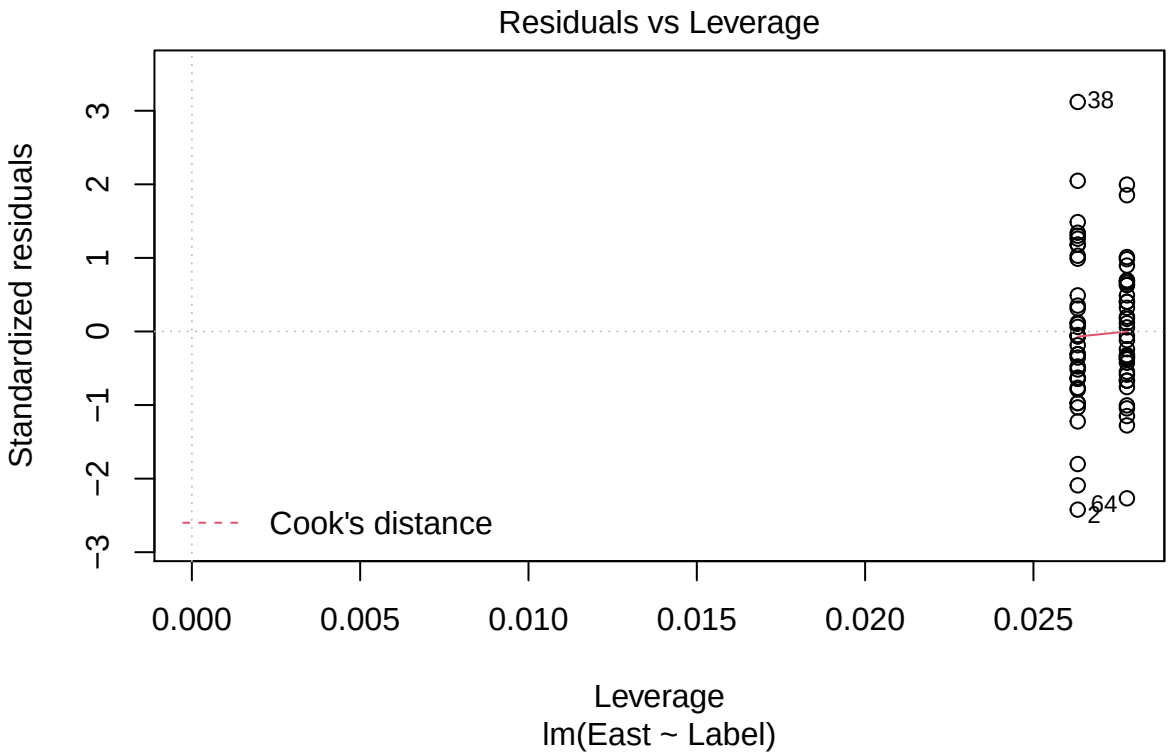
We can confirm that our model doesn't egregiously violate these assumptions by running the `plot()` function on our model object `mod1`. When we do this, R returns four plots but they are not immediately visible. You will see that the R Console now reads `Hit <Return> to see next plot:`. We must place our cursor in the R Console and hit Enter four more times: once to view each of the four plots.

```
plot(mod1)
```


Residuals vs Fitted
Residuals
6
4
2
0
−2
−4
38
2
64
5.6
5.7
5.8
5.9
6.0
6.1
Fitted values
lm(East ~ Label)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(East ~ Label)
38
64
2
−2
−1
0
1
2
3


Scale−Location
38
2
64
√|Standardized residuals|
0.0
0.5
1.0
1.5
5.6
5.7
5.8
5.9
6.0
6.1
Fitted values
lm(East ~ Label)



The first plot is titled Residuals vs Fitted and it shows the residual values (the difference between the data point and the predicted value corresponding to that data point) relative to the fitted values from our model. This plot is useful for assessing whether there is any strange patterning in our residuals. There should not be, and the residuals should be randomly distributed relative to the red line shown in the plot, corresponding to the model fit itself (residual = 0).

The second plot is what's called a Q-Q plot, or quantile-quantile plot. This plot is used to visualize whether the standardized residuals are normally distributed. The dotted black line that runs diagonally across the plot represents normality. Our data points should all fall along this line with no strange deviations.

The third plot, Scale-Locataion, is similar to the first but in this case we are plotting the square root of the residuals. The red line in this plot should be horizontal, indicating that the relationship between the residuals and the model fit does not change across all the values of the predictor.

Finally, the fourth plot shows the leverage of the residuals. This plot is useful for assessing whether there are any data points which have undue influence on our model. These will be shown as falling outside the dashed red lines in this plot, which are labeled "Cook's distance". Often, these lines will not be visible because all data points fall within the range, as is the case here.

1.1.2 Model Output

Now that we have determined that our model is not violating any assumptions, we can move on to assessing the model itself. We do this using the `summary()` function:

```
summary(mod1)
```

```
##
```

```
## Call:
## lm(formula = East ~ Label, data = Barm.data.sub)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1821 -1.1046 -0.1121  1.0936  5.3879
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   5.5321     0.2839  19.484   <2e-16 ***
## LabelLanceolate Microliths   0.6218     0.4071   1.527    0.131
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.75 on 72 degrees of freedom
## Multiple R-squared:  0.03139,    Adjusted R-squared:  0.01793
## F-statistic: 2.333 on 1 and 72 DF,  p-value: 0.131
```

This function returns a lot of useful information. First, it returns our model code. Then, it returns information on the model residuals. More important than these, however, is the information on the model coefficients. We see a table of values with rows labeled `(Intercept)` and `LabelLanceolate Microliths` and columns labeled `Estimate`, `Std. Error`, `t value`, and `Pr(>|t|)`. This is where we can learn what the model intercept value is and what the effect of artifact type is on easting coordinates.

Our model has an intercept of 5.5321 with an error of 0.2839. More importantly, our model has a slope of 0.6218 with an error of 0.4071. This slope is the effect of artifact type on the easting coordinates. To interpret this, we can do some basic math.

If we calculate the mean easting coordinates for scrapers, we see that it is 5.5321. Compare this to the intercept value of our model: they're identical values! Why is this?

This all makes sense when you remember the form of the linear model: `y = b0 + b1 * x`.

```
mean(BarmoseI.pp[BarmoseI.pp$Label == "Scrapers", ]$East)
```

```
## [1] 5.532105
```

```
mod1$coefficients[1] + (mod1$coefficients[2] * 0)
```

```
## (Intercept)
##    5.532105
```

```
#take the intercept and add it to the slope multiplied by 0
```

It's important to remember that with linear modeling, we must add the intercept value (`b0`) to our slope (`b1`) if we want to predict the mean value (`y`) of a condition/group (`x`). To calculate the predicted mean for scrapers, we first multiply the slope by x. In this case, "scrapers" is being treated as the baseline group by R, and is therefore assigned a value of 0. "Lanceolate microliths" is group 1, and is assigned a value of 1. So in the above case, we are adding the intercept to the slope, but first we're multiplying the slope by 0 since we're calculating the predicted mean for group 0 (scrapers). This means that the predicted mean of the baseline group is really just the intercept!

Now let's calculate the mean easting value of lanceolate microliths, which is 6.1539. Our model tells us the effect of artifact type is 0.6218, which is not the same as the mean microlith easting value... because we

need to add it to the slope, of course. If we sum the intercept, 5.5321, and our slope, 0.6218, we get the mean easting value of microliths: 6.1539! As described above, what we're really doing is multiplying the slope by 1. We're calculating the predicted mean for group 1, lanceolate microliths, so x = 1. Then we add this value to the intercept to get our mean easting coordinate.

```
mean(BarmoseI.pp[BarmoseI.pp$Label == "Lanceolate Microliths", ]$East)
```

```
## [1] 6.153889
```

```
mod1$coefficients[1] + (mod1$coefficients[2] * 1)
```

```
## (Intercept)
##    6.153889
```

And sure enough, they match!

1.1.3 P-values from a Linear Model

Let's return to the summary output of our model:

```
summary(mod1)
```

```
##
## Call:
## lm(formula = East ~ Label, data = Barm.data.sub)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1821 -1.1046 -0.1121  1.0936  5.3879
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   5.5321     0.2839  19.484   <2e-16 ***
## LabelLanceolate Microliths   0.6218     0.4071   1.527    0.131
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.75 on 72 degrees of freedom
## Multiple R-squared:  0.03139,    Adjusted R-squared:  0.01793
## F-statistic: 2.333 on 1 and 72 DF,  p-value: 0.131
```

We now understand the coefficient estimates for our model, but now let's look at the next columns. The Std. Error column is showing the error estimates corresponding to the coefficient estimates for the intercept and slope. The intercept estimate is therefore 0.6218 +/- 0.4071. It is then possible to calculate a p-value with this information that can tell us whether the means of these two groups are significantly different from each other. This p-value is shown in the fourth column, `Pr(>|t|)`, and the row for the slope, `LabelLanceolate Microliths`. This p-value is 0.131: not significant at an alpha of 0.05.

A p-value of 0.131 means that if we were to repeat our observations of these two groups of artifacts an infinite number of times, we would obtain this result, or a more extreme result, 13.1% of the time if the null hypothesis were true and there were no difference between the two groups. Quite the mouthful...

In other words, if there is no difference between the easting coordinates of scrapers and lanceolate microliths, we would expect to get these data 13.1% of the time if we sampled these artifact locations a huge number of times. In other words still, we'd get these data 13.1% of the time if our null hypothesis is true.

This actually means that these data are somewhat unlikely if the null hypothesis is true and these two artifact types are identically distributed in space along the east-west axis. Thirteen percent is not a huge number, but it's still larger than the conventional cutoff point of 0.05, or 5%. Therefore, we can say that there is no statistically significant difference in easting location between these two groups of artifacts.

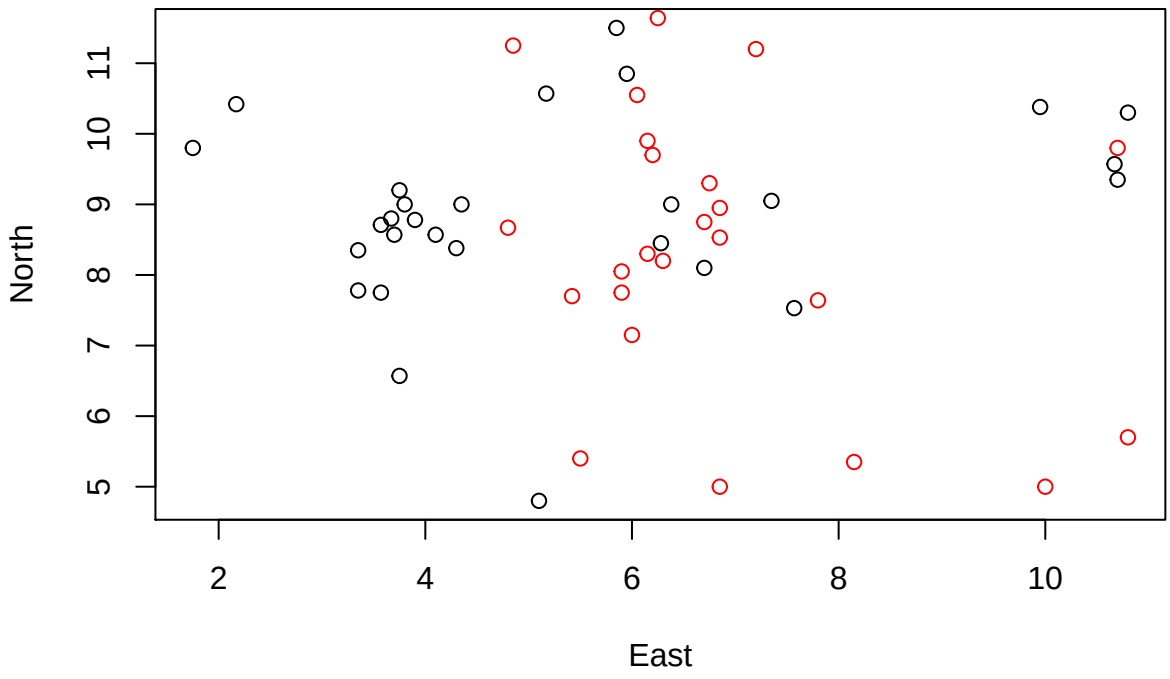
1.2 Building Another Model

Now let's build another model.

Let's say we're now interested in the distribution of flake axes and denticulated/notched pieces, and particularly the easting distribution of these artifacts. We can subset our BarmoseI.pp dataset to just include these two artifact types, and then plot the coordinates of each type in different colors.

```
Barm.data.sub2 <- BarmoseI.pp[BarmoseI.pp$Label == "Flake Axes" |
                                BarmoseI.pp$Label == "Denticulated/Notched Pieces", ]

plot(North ~ East, data = BarmoseI.pp[BarmoseI.pp$Label == "Flake Axes", ])
points(North ~ East,
       data = BarmoseI.pp[BarmoseI.pp$Label == "Denticulated/Notched Pieces", ],
       col = "red")
```

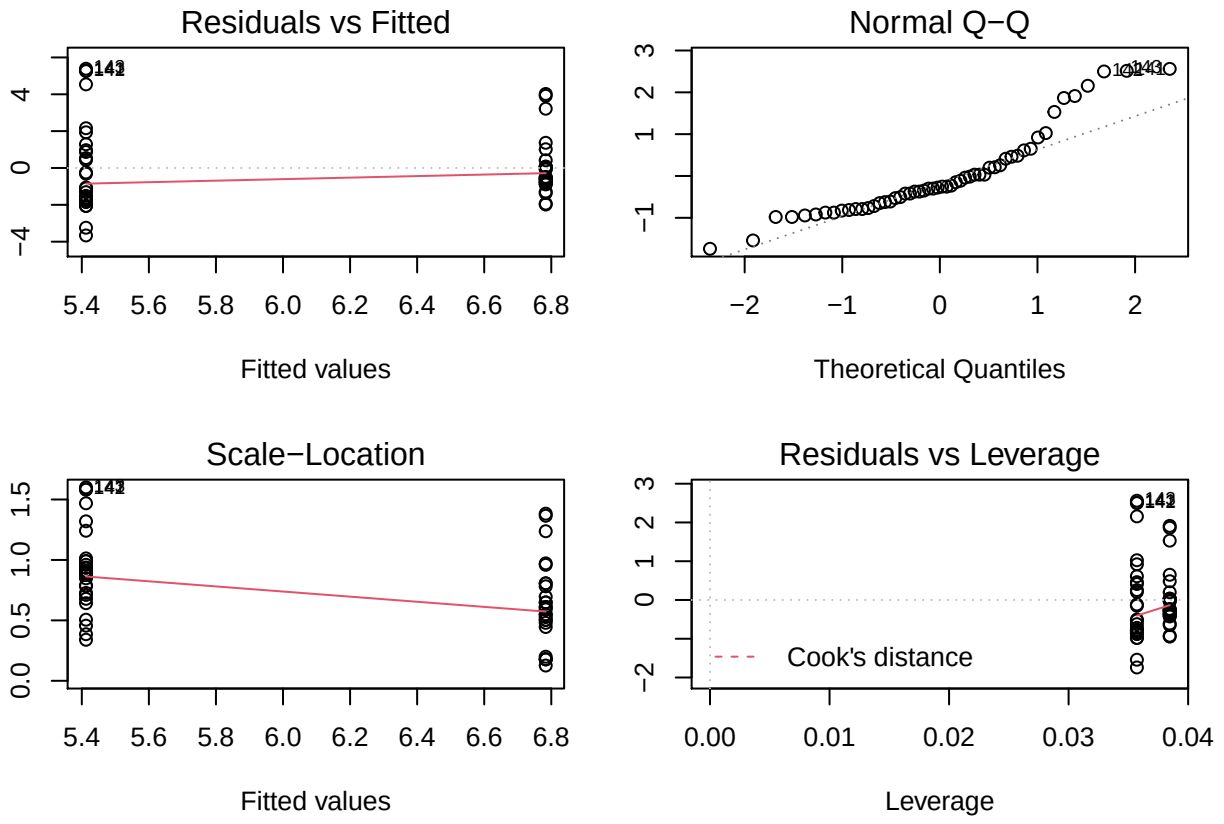


It looks like most flake axes (black) are further west while most denticulated or notched lithics are a bit further east (red). But we want to model this so we can be more confident in this observation. So let's make another linear model and inspect the diagnostic plots.

```
mod2 <- lm(East ~ Label, data = Barm.data.sub2)
```

To inspect the diagnostic plots this time, let's change the plotting parameters so that we can view all four at once without having to click through them in the R Console. We can do this using the `par()` function with the `mfrow` argument. The `mfrow` argument controls how many plots to show at once. The default value is `c(1, 1)`, denoting 1 row and 1 column of plots: i.e., one plot. If we change it to `mfrow = c(2, 2)`, we can plot all four diagnostic plots at once, in two rows and two columns. To do this, we also need to shrink the plot margins a bit using the `mar` argument. Note that after I change the parameters and obtain our plots, I change them back.

```
par(mfrow = c(2, 2), mar = c(4, 3, 3, 1))
plot(mod2)
```




```
par(mfrow = c(1, 1), mar = c(5.1, 4.1, 4.1, 2.1))
```

Now, with this new model, `mod2`, the diagnostic plots don't look quite as good as before. The more of these plots you see, the better you'll be able to spot problems, but while these plots aren't perfect, they're not really problematic in my view. The residuals look to be randomly distributed around the model fit with no troubling heteroskedasticity or outliers. The Q-Q plot is the worst looking of the bunch - check out how the residuals begin to deviate from normality (the dotted line) at higher quantile values. But, it's still not

terrible and the vast majority of points fall along the line. Again, this is all a bit subjective and your eyes will become more practiced with experience.

Now that we're confident that our model isn't behaving poorly, we can inspect the model output:

```
summary(mod2)
```

```
##
## Call:
## lm(formula = East ~ Label, data = Barm.data.sub2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6625 -1.4752 -0.5585  0.7850  5.3875
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       5.4125     0.4049   13.37   <2e-16 ***
## LabelDenticulated/Notched Pieces   1.3710     0.5835    2.35   0.0226 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.142 on 52 degrees of freedom
## Multiple R-squared:  0.09598,    Adjusted R-squared:  0.07859
## F-statistic: 5.521 on 1 and 52 DF,  p-value: 0.02262
```

Taking a look at the model summary, we see that our intercept is 5.4125 +/- 0.4049 while our coefficient for denticulated/notched pieces is 1.3710 +/- 0.5835. Recall from above that R treats the first category as the intercept and the second as the slope. Thus, the mean easting value for flake axes is 5.4125 - our intercept - while the mean easting value for denticulated/notched pieces is the sum of our intercept and the other coefficient (5.4125 + 1.3710 = 6.7834).

More importantly for most of us, the p-value is significant! The model p-value is 0.022, which is less than the conventional cutoff of 0.05. This means that there is only a 2.2% chance that we would observe these easting coordinates if the null hypothesis were true and these two artifact types were distributed identically. That's a small enough chance that it can be considered significant!

2 T-Tests

You might not have realized it, but what you just performed was a t-test!

Perhaps the most basic of statistical hypothesis tests is the t-test. A t-test, at its most basic, is a test to compare the means of two groups of observations. There are several different varieties of t-tests that we won't get into here, but just as with all other statistical hypothesis tests, they're all just variations on a linear model! Thus, if you understand linear modeling, you can understand t-tests, or any other type of test.

If you still don't believe me, check this out. Remember our p-value from above? Run the `summary()` function again if you need to find it:

```
summary(mod2)
```

We obtained this p-value from a linear model using the `lm()` function. It was 0.02262.

Now let's use the `t.test()` function that is built into R. You can check out the help page for this function using the `?` function, and then simply plug in the same information to this `t.test()` function that we fed to the `lm()` function.

```
t.test(East ~ Label, data = Barm.data.sub2, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  East by Label
## t = -2.3496, df = 52, p-value = 0.02262
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.5418028 -0.2001203
## sample estimates:
##                   mean in group Flake Axes
##                                   5.412500
## mean in group Denticulated/Notched Pieces
##                                   6.783462
```

The p-value from this t test is 0.02262: the same as the one we got from our linear model. This is because a t test *is* a linear model! Simply a particular variety of one with normally distributed residuals and two categorical predictor variables.

To report this result, we would say something like:

The result of a t-test reveals that there is a significant difference in easting values between these two artifact types (t = -2.35, df = 52, $p < 0.05$).

You should always report the t statistic and degrees of freedom, which are important pieces of this statistical test and necessary for calculating the p-value. These values are reported in the output of the `t.test()` function. We won't get into what these numbers really mean here, but you should report them nonetheless.

So now you know the basics of linear models, specifically with reference to cases of two categorical predictors, or a t test!