Linear Regression & Correlation

INFO 640 | Pratt Institute | McSweeney

```
#install.packages("broom")
#install.packages("GGally")
```

```
library(tidyverse)
library(lubridate)
library(broom) #to help clean things up and remerge our dataframes
library(GGally) #to help run multiple pair-wise correlations
```

This lab explores correlation and regression in R, understanding the output, reporting the findings, and visualizing the results. By the end of this lab, you will be able to run regression analysis on 2 variables to understand the extent to which one predicts the other.

We are going to work with the trees dataset, this is a measure of the height, girth (diameter) and volume of black cherry trees. More information is available in the documentation. We want to know if the girth predicts the height. This would be very useful because it is much easier to measure the girth of a tree than it is to measure the height of a tree.

```
?trees
```

```
data(trees)
glimpse(trees)
```

We are interested in the Height and Girth measurements, wanting to predict how tall a tree might be given its diameter. We'll start with a visualization.

```
ggplot(trees, aes(x=trees$Girth, y=trees$Height)) + geom_point()+
  labs(title="Tree height by girth")
```

Well, those points are all over the place, surely there is not a strong correlation, but let's dig a little deeper.

Now we will add the line of best fit which we will use to visualize where new points might be. We will set standard error to FALSE for now.

```
ggplot(trees, aes(x=trees$Girth, y=trees$Height)) + geom_point()+
  stat_smooth(method="lm", se=FALSE)+
  labs(title="Tree height by girth")
```

This gives us a good idea of how much the diameter of a tree contributes to its height. We see a mdoderate positive correlation.

If the fact that we are in feet and inches bothers you, you can change the units of the variable. However, it does not matter, the slope of the regression line is units agnostic. Secondly, sometimes this will not be possible (i.e., when comparing poverty rates to measures like high school attainment).

Change units if you want, this lab continues to use 'Height', not 'Height_inch'.

```
trees$Height_inch <- trees$Height*12
```

Now let's make a linear model with the trees data. This is always specified lm(y ~ x, data). y is your dependent variable, and x is your independent. In our case, we have stated the question so that Height is dependent on Girth (that's what we're hoping at least). For our purposes, Time is aways independent (exceptions found in Physics).

When building the linear models, do not use the dataset name when calling the variables. It will become apparent later why that is essential.

```
lm_trees <- lm(Height ~ Girth, data=trees)
lm_trees
```

Simply calling the model gives the coefficients. This number means that for every inch wider a tree is, it is 1.054 feet taller, assuming a base height of 62.031 feet. If you remember your algebra classes, this is the slope and intercept or the 'm' and 'b' in the classic, y=mx+b formula.

If we want to get more information about our model:

```
summary(lm_trees)
```

We'll address each of these values and what they mean later in this tutorial. If we want to see just the coefficients, call the coefficients functions.

```
coef(lm_trees)
```

To get a vector with all the fitted values (y'), for each data point, call for the fillted values. These tell us what the model predicted rather than what was measured. So, for each girth, there's a predicted height. The fitted values give us that predicted height.

```
fitted_trees <- fitted.values(lm_trees)
fitted_trees
```

Residuals - each fitted value returns a residual. The residual is the difference between the actual, measured value and the predicted (fitted) value. This can be useful for identifying outliers and modeling the distribution of the noise. This is much of the usefulness of regression analysis. By understanding how much the model is off, you can start to make it better.

```
residual_trees <- residuals(lm_trees)
```

```
residual_trees
```

At some point, we probably want one unified dataframe with our original response & explanatory variables along with the fitted values, responses, and residuals.

We need to rejoin the lm calculations with the original dataframe. Augment is a really common command, so we need to specify that we want the broom package's version of augment.

```
lm_matrix_trees <- broom::augment(lm_trees)
head(lm_matrix_trees)
```

We can use this augmented dataframe to examine the residuals and identify the outlier(s). A large residual will be a big difference between what was predicted and what was measured. We only want to review this, not save it, so we'll use some shorthand and pipe the arranged dataframe over to the head function all in the same sentence.

```
lm_matrix_trees %>%
arrange(desc(.resid)) %>%
head()
```

Note that residuals are raw values, so are both positive and negative. We'd like to be able to see all of them at once - compare the absolute value of the residuals. We need to sort the absolute values of the residuals to identify the outliers - or the most extreme values.

```
lm_matrix_trees$.resid_abs <- abs(lm_matrix_trees$.resid)


lm_matrix_trees %>%
arrange(desc(.resid_abs)) %>%
head()
```

It looks that the tree that is the most outlier is 13.8 inches diameter. Let's inspect this tree. If we found something noteworthy about this tree - maybe it was cultivated in a different way, or otherwise stood out from the remainer of the dataset in a way that made it different from the others, then it could be removed. However, there is no evidence of that here, so we will just inspect it, but keep it in our model.

```
trees %>%
filter(Girth == 13.8)
```

Now let's say that we found a Black Cherry Tree that was exactly 19 inches in diameter. How tall do we think it will be?

For our purposes, we only have one value, but you could have a whole, giant dataframe of such values. In order to merge the dataframes and predict the new heights from the old Girths, we have to have a column that matches our linear model exactly. Therefore, we need a dataframe that has a "Girth" column. Had

we used “trees$Girth" above, we would need a dataframe that had "trees$Girth” as one of the columns.

We will make a new dataframe with just this one value, and then call predict on it using the linear model we have built. This will return a vector of predictions. You can attach it to the dataframe of new values with the augment function.

```
head(lm_trees)
new_trees <- data.frame("Girth" = 19.5)

predict(lm_trees, newdata=new_trees)

mytree <- broom::augment(lm_trees, newdata=new_trees)
mytree
```

We can take this new predicted value and represent it on the same plot with our other values. In order to visualize where it would be situated. Let’s color it red so its easier to see. We could also make it larger so it’s easier to see.

```
ggplot(data = trees, aes(x = Girth, y = Height)) +
geom_point() + geom_smooth(method = "lm") +
geom_point(data = mytree, aes(y = .fitted), size = 3, color = "red")+
labs(title="Tree height by girth")
```

Now let’s say that we want to know how well our model performed compared to the null model. The null model in this case is the situation where every single tree is the same height. It’s the mean height. If you had to take a wild guess about the height of a Blac Chetty tree and you couldn’t see it, didn’t know the girth, or anything else about it, you’d probably guess the mean of all Black Cherry trees. That is the null model.

R makes a model of the null model by basically modelling the Height as related to nothing, for which we use the placeholder, 1.

```
tree_null <- lm(Height ~ 1, data = trees)
```

Unfortunately, it’s very hard to represent this on a plot. But that’s ok, it’s just the mean of the dependent values. We’ll use the mean to create a horizontal line at the yintercept of the mean height.

```
mean_height <- mean(trees$Height)

ggplot(data = trees, aes(x = Girth, y = Height)) +
geom_point() +
geom_hline(yintercept=mean_height) +
labs(title="Tree height null model")
```

Finally, if we want to assess how much error this model accounts for, we can call the summary again, and look for the Multiple R-squared.

```
summary(lm_trees)
```

So far, we have assumed that we already know what variables we want to investigate. For *exploratory data analysis*, we would want to examine all of the variables that we have reason to believe are either correlated with each other or are different measures of similar underlying factors.

To do this, you want to make a correlation grid and then focus on the variables that are the most correlated.

For us, we will look at all 3 variables from the trees dataset.

```
ggpairs(data = trees, columns=1:3)
```

We know mathematically that Volume and Girth are the most likely to be directly correlated, and they are. This gives us a roadmap to get started.

Congratulations!! You now have the tools to analyze how variables influence each other. If you want to work with another dataset, the longley dataset is a great place to start. It it about employment and various social factors.