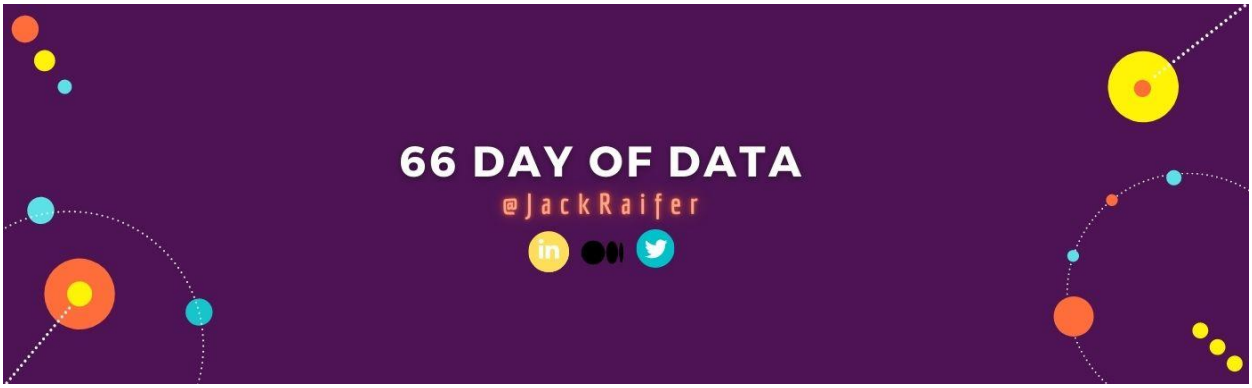

#66DaysOfData – Days 15 to 17: Building an OCEAN – Attrition Formula

Last time we discussed how a biostatistics risk formula is built using the Lung Cancer Risk for smokers built by Maria Maraki. Now, it is time to start building our own risk model for attrition based on the OCEAN personality model. To do this, we need to take all the info we can from the available research papers.

Since we will want to use this formula to later build a formula-based prediction model, then why not do it with a Jupyter Notebook.

As usual, we set up our notebook with the libraries we believe will be useful, this time around, I will only use NumPy since it allows us to build arrays and do easy and quick statistics operations. The statistics library or just coding from scratch are also a possibility in this case, but I'm a NumPy kind of person.

Before we start, we need to define which variables we will contemplate for creating the formula. Initially, let us just focus on defining, as variables, the 5 traits of OCEAN personality:

O – Openness to Experience (How open a person is to new experiences)
C – Conscientiousness (How much is someone conscious of the consequences of actions)
E – Extraversion (How outgoing a person is and/or needs to be)
A – Agreeableness (How much a person tries to be agreeable and friendly)
N – Neuroticism (High neuroticism describes a person's lack of emotional regulation)

Please do consider these are very wide and succinct explanations of the variables, there are whole books, thick, long and dense, about the subject, but for our purposes, this will do.

Now that we have our variables of interest, we can go ahead and start adding the information we discovered in the research papers. Although there is no access to the original data, we do have the correlation results of each of these variables to the attrition or turnover event, so let us create each variable as an array with the correlation from each study. The reason I decided to use

arrays, is because it will allow me to run statistical operations easily to create the variables we need for the formula:

```
[14]: Odata = np.array([-0.20, -0.285, -0.214, -0.19, -0.232])
      Cdata = np.array([-0.23, -0.219, -0.257, -0.153, -0.21])
      Edata = np.array([-0.19, -0.065, -0.128, -0.095, -0.165])
      Adata = np.array([-0.26, -0.238, -0.257, -0.24, -0.236])
      Ndata = np.array([0.28, 0.132, 0.228, 0.176, 0.20])
```

To build a risk formula, we need just one number for each of variables, or it would become unwieldly, the best way to do this, is to create just one-off variables for each of the OCEAN traits using the mean of the correlations we found:

```
[15]: Omean = Odata.mean()
      Cmean = Cdata.mean()
      Emean = Edata.mean()
      Amean = Adata.mean()
      Nmean = Ndata.mean()
```

Our means, in a way, define how much each of the variables affects the risk of attrition, so, with this data, if we had to build a formula right now, it would look like this:

AttRisk = O*-0.0422 + C*-0.2138 + A*-0.1462 + E*-0.0714 + N*0.3032

If we use the averages from our world OCEAN results dataset, we can test out our formula and see what happens:

```
[207]: O,C,E,A,N = 0.733941, 0.701998, 0.672329, 0.696807, 0.574399

[209]: AttRisk = O*Omean + C*Cmean + A*Amean + E*Emean + N*Nmean

[210]: AttRisk

[210]: -0.15677917979999997
```

Hmm, something seems off here, how can we have a negative risk? It's not like there is an actual risk of a person clinging to the company (at least not that has been studied yet), so the lowest percentage should be zero, and that would be a very unlikely result. Of course, what we are missing here is a common practice in statistics, biostatistics and of course Data Science: Normalizing. This means all the results should be constrained to be within 0 and 1. To do that we use the following normalization formula:

x normalized = (x – x minimum) / (x maximum – x minimum)

Applying this normalization to the end of our formula, this is what we get:

```
[140]: AttRisknorm = (((O*Omean + C*Cmean + A*Amean + E*Emean + N*Nmean) - -0.4736) / (0.3032 - -0.4736))

[141]: AttRisknorm

[141]: 0.407853785015448
```

Now we seem to be getting somewhere. Still, we are using the OCEAN data of the worldwide average, so a 40% Risk of Attrition for an average person seems incredibly high, after all, that would mean that, on average, 40% of the people you hire would leave the company, this would be crippling to practically every business out there. How can we fix this? Simple, we need to adjust our formula to real world data. And what is the worldwide turnover average? The most comprehensive data I found for this says it was 10.9% in 2020.

So, we need to adjust our formula so that an average person has a 10.9% risk of attrition, making our new formula look like this:

```
[204]: AttRiskFinal = ((((O*Omean + C*Cmean + A*Amean + E*Emean + N*Nmean) - -0.4736) / (0.3032 - -0.4736))) / 3.74

[205]: AttRiskFinal

[205]: 0.10905181417525348
```

And there we have it, a data driven, hand made formula for calculating attrition risk based on the OCEAN personality model. Is it perfect? Not even close. Is it good? I have absolutely no idea, we need to build a model with this formula and test it out on real data. Is it better than building an ML model? Probably not, but this exercise allowed me to understand what I am looking for even better, to build something that, once converted into a model, can help me benchmark ML models to make sure we can build something better.

What I do hope to have proven here is how exhausting it can be to create any kind of algorithm from scratch, it takes a lot of domain knowledge, as well as a lot of reading (got to read about 30 papers on the subject the past week), and some creative tweaking. Also, we can already see problems. What happens if we want to use this in a specific population with different averages? Do the averages change? What happens when they do? Do we have to tweak the formula every year or so?

And, this is why we love all the new and amazing tools we have for Data Science, since they can answer most of those questions and allows us to automate a lot of the processes and make improvements a lot faster.

Next Time – Building a Predictive Model with our Risk Formula

Jack Raifer Baruch

Follow me on Twitter: @JackRaifer

Follow me on LinkedIN: jackraifer

About the Road to Data Science - #66DaysOfData Series

Road to Data Science series began after I experienced the first round of Ken Jee´s #66DaysOfData challenge back in 2020. Since we are starting the second round of the challenge, I thought it would be a good idea to add small articles every day where I can comment my progress.

I will be sharing all the notebooks, articles and data I can on GitHub: https://github.com/jackraifer/66DaysOfData-Road-to-Data-Science

Please do understand I might have to withhold some information, including code, data, visualizations and/or models, because of confidentiality regards. But I will try to share as much as possible.

Want to follow the #66DaysOfDataChallenge?

Just follow Ken Jee on twitter **@KenJee_DS** and join the #66DaysOfData challenge.

You can also reach out to me at any time through **LinkedIN** or **Twitter.**