Michael Ippolito
CUNY DATA621
Blog #2

Background

I work at a large university that has many divisions which are very much decentralized from one another when it comes to things like IT and IT security. Each division has its own IT support groups, its own way of maintaining inventory, its own priorities and objectives when it comes to all things technical. Often, there is even further decentralization within divisions, with departments within them doing their own thing. This has its benefits, but it often makes it difficult for units whose job it is to perform central services.

In my role as a cybersecurity analyst, I'm confronted by this fact daily. Our office is ultimately responsible for the overall security of the university, yet we often lack the visibility into or the authority over its divisions and departments. And while we're asked to report on the overall security posture of the university, it's often somewhat of a guessing game.

One aspect of that security posture is in the realm of endpoint security. Our office contracts with security software vendors to license tools such as malware prevention, firewall services, and vulnerability management. We can't force divisions or departments to use them, but we make them available and we strongly encourage their use.

Recently, we have advocated for a stronger, more active approach to deploying these tools to university endpoints. As with any such effort, leadership is interested in gauging progress, namely how many of the university's endpoints are using the security tools, and how many more do we have to go for a "full" deployment.

The first metric is easy. We have consoles and reports and graphs that we can generate using any given tool. It's the second one, the "denominator" in the equation, that is and has been historically difficult. We don't know how many endpoints each division has, because they don't report those counts centrally. Rather, they maintain them locally, if they maintain them at all. So while we know how many endpoints are secured, we don't know if that represents 50%, 75%, or 100% of the total. (As an aside note, we'll never be able to achieve a "full" deployment. There will always be things like printers, IoT devices, or odd flavours of Linux that don't support installation of the tools on whatever one-off operating system they're running. But we hope to get close, i.e., deploying the tools to as many devices that actually support them.)

I was recently tasked with trying to generate an approximation for how many total endpoints each unit has. That number, along with the counts of tools already deployed, will be included in periodic reports to the head of each division, giving him or her a sense of how "secure" their endpoints are. I put that in quotes because a lot of other factors go into a division's overall security; this is just one of them. But it's an important one, and one that we haven't been able to provide. Now we're able to at least provide an approximation that is based on something more than just a random guess out of thin air.

Objective

CThe goal is to generate approximate endpoint estimates for each division. My approach was to gather data from departments and divisions from which we have known endpoint counts, fit a linear model to that data, and predict endpoint counts based on the coefficients generated by the model.

The data we had with which to create this kind of model was of limited quality. First, we had full-time employee (FTE) counts per department and division (we used March 2022 payroll data). Second, we had mac address counts mapped to IP addresses over a 30-day period during July and August 2022. While the quality of the FTE counts itself was high-fidelity, endpoint counts and FTE counts don't correlate precisely. Likewise, the fidelity of the mac-address data suffers from several problems. First, it is difficult to correlate IP addresses (and, hence, mac addresses) back to a particular division. I created a separate model to do this, but the model is only 81% accurate against validated training data and is likely much less accurate against untested data. Second, the mac address data doesn't account for endpoints that are connected to public or wireless networks, nor does it account for VPN-connected or other remote users.

In addition to FTE and mac address data, a categorical variable (large_unit) was added to account for divisions and departments having atypically high endpoint counts, for example units that host services for other campus units that may require large numbers of servers. We also treated the university's central IT division as a special unit, since it runs many services for campus and will, therefore, have an unusually high endpoint count.

Despite the limitations described above, the model may serve as a rough estimate in the absence of any other means of quantifying endpoint counts. These numbers are only to be used with a heavy disclaimer and not to be used as a checklist against which to measure the true progress of any school, college, or division in deploying security software to their endpoints.

<u>Data Sources</u>

For the first data set, we queried mac address counts from a database that tracks mac and IP information for the last 30 days. Example data:

Subnet	Count
10.1.1.0/24	200
10.1.2.0/24	50

There were approximately 32,000 subnet records like the above. These counts had to be correlated back to a division code; historically this has always been a challenge, since the network database does not tie a subnet record to a division code. However, each subnet *is* tied back to the staff members who administer it. And from these staff members we can query the university's WhoIs database to glean the division code for each of those staff members. This isn't always reliable, as the staff member isn't necessarily a member of the division the subnet belongs to. But it is right often enough that the information is useful to the model.

To overcome these challenges, I combined the contact-based division predictions with a natural language processing (NLP) model based on document similarity. The subnet names and descriptions were quantitatively compared with terms scraped from websites mapped back to a particular division To calculate the "best" division match, the division with the smallest Jaccard distance was weighted more heavily than those with larger distances. Additional "institutional knowledge" was also used to weight some divisional predictions more heavily than others.

With the aim of correlating endpoint counts directly to FTE counts, we collected the 2022 FTE counts from university HR records. Example data:

Unit	FTE Count
Axx9100	177.5
Axy3205	22.0

We also have a list of 46 departments and divisions that have “full” or “near-full” deployments. For the purposes of estimating total counts, we’ll consider these counts as a “full” deployment upon which to base our model. Along with this list, we identified 13 departments as being potentially “large” units, i.e. ones that have an atypically high endpoint count.

In addition, we have counts of the security tools we’re interested in deploying:

Unit	Tool	Deployed Count
Axx9100	Antimalware	100
Axx9100	Vulnerability management	98
Axy3205	Antimalware	5
Axy3205	Vulnerability management	0

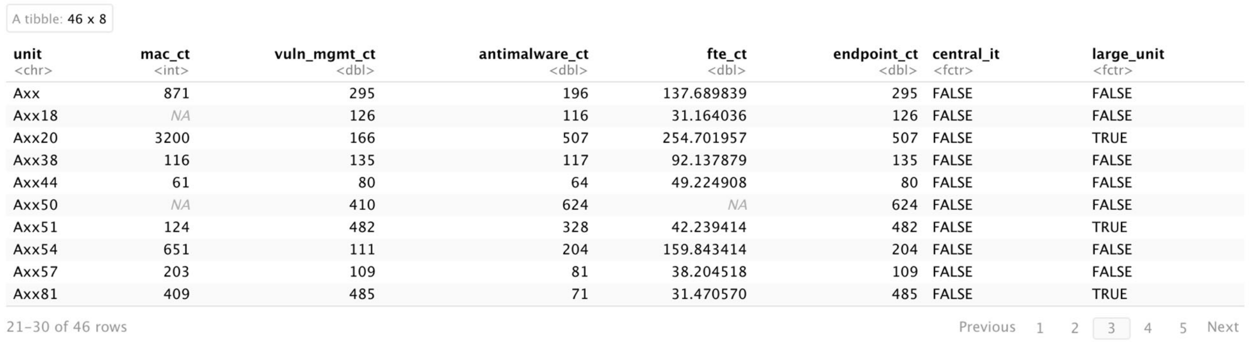
What we want to end up with is something like this:

Unit	Full Unit	Large Unit	Mac Addr Count	FTE Count	Total Endpoints	Antimalware Count	Vuln Mgmt Count
Axx9100	Yes	Yes	210	177.5	100	100	98
Axy3205	No	No	50	22.0	?	5	0
Axz4590	No	Yes	400	275.0	?	200	150

And from there we can calculate the percentage deployment of each tool.

Data Exploration

First, we’ll look at a sample of the data we have for fully deployed units, which we’ll base the model on.

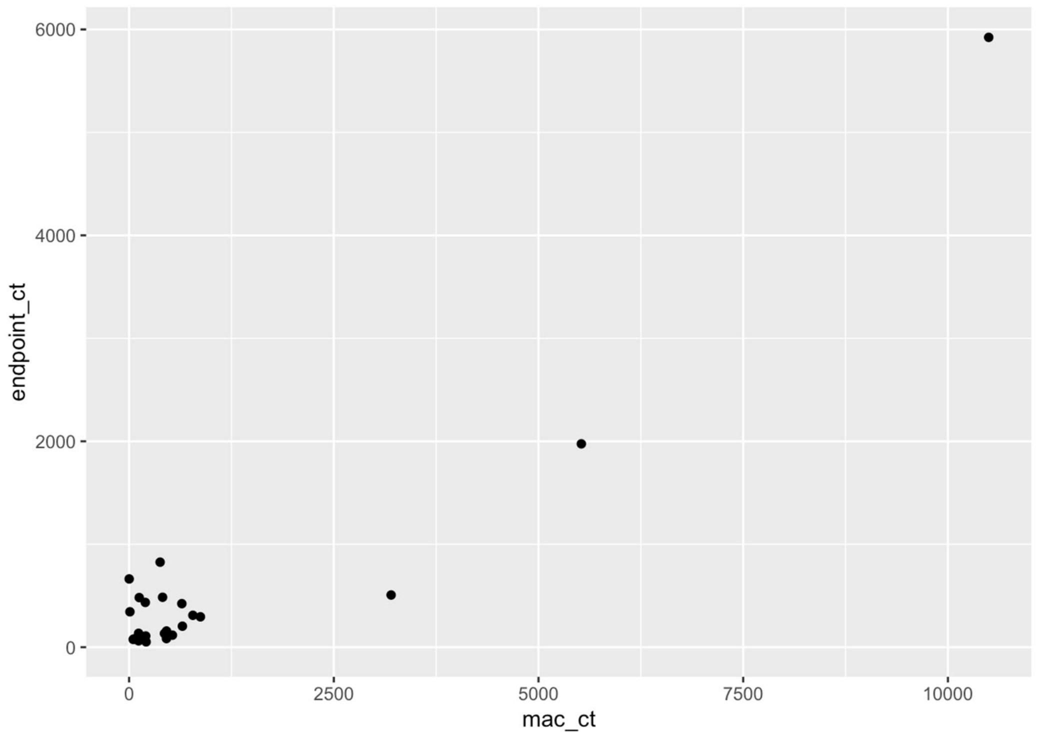


A tibble: 46 x 8

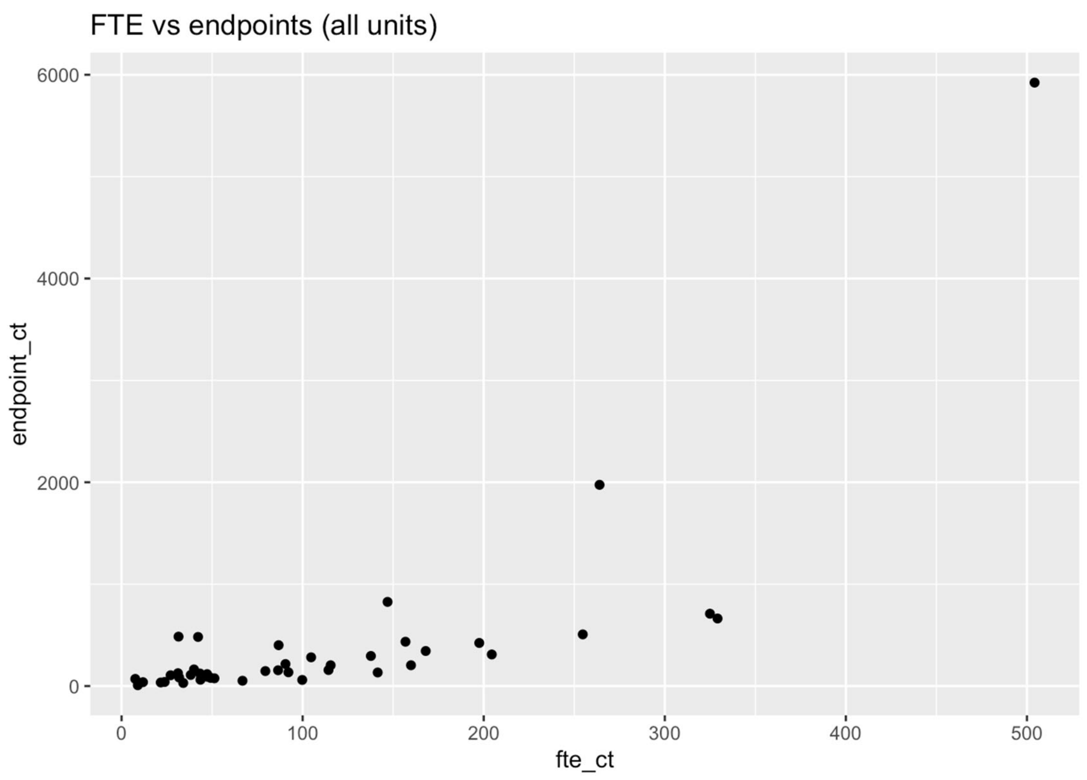
unit <chr>	mac_ct <int>	vuln_mgmt_ct <dbl>	antimalware_ct <dbl>	fte_ct <dbl>	endpoint_ct <dbl>	central_it <fctr>	large_unit <fctr>
Axx	871	295	196	137.689839	295	FALSE	FALSE
Axx18	NA	126	116	31.164036	126	FALSE	FALSE
Axx20	3200	166	507	254.701957	507	FALSE	TRUE
Axx38	116	135	117	92.137879	135	FALSE	FALSE
Axx44	61	80	64	49.224908	80	FALSE	FALSE
Axx50	NA	410	624	NA	624	FALSE	FALSE
Axx51	124	482	328	42.239414	482	FALSE	TRUE
Axx54	651	111	204	159.843414	204	FALSE	FALSE
Axx57	203	109	81	38.204518	109	FALSE	FALSE
Axx81	409	485	71	31.470570	485	FALSE	TRUE

21–30 of 46 rows

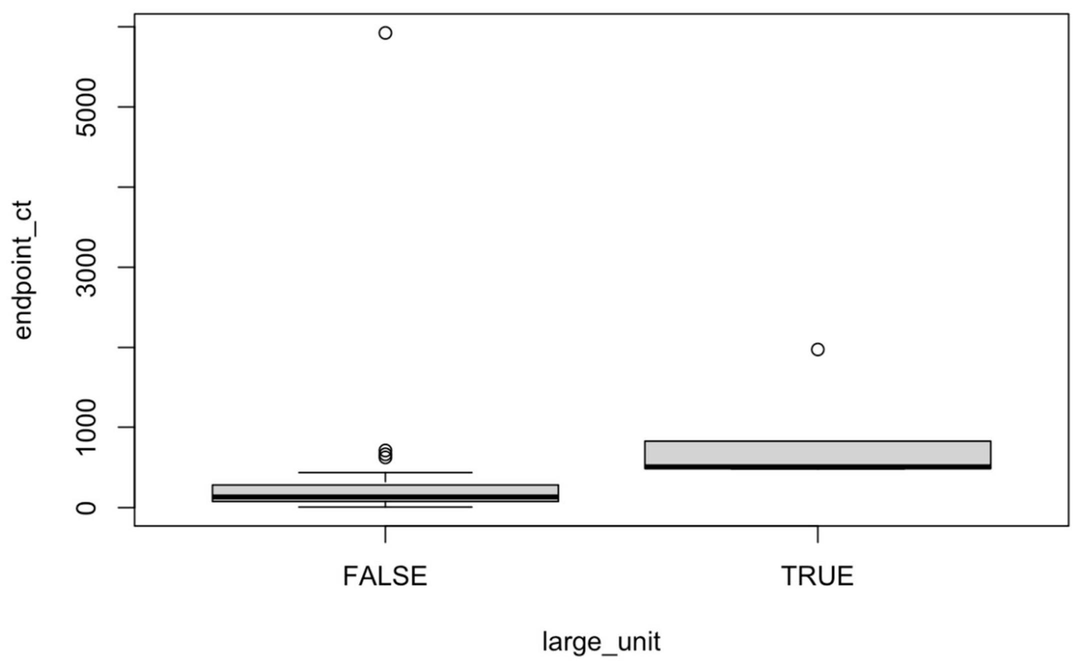
Here are the endpoint counts vs mac address:



And the endpoint counts vs FTE counts:



Box plots of endpoint count vs large_unit. Note that a unit can be large due to sheer size (as in the case of the outlier shown below) or can be disproportionately large given its relatively smaller FTE count. The latter is what we're considering a "large unit."

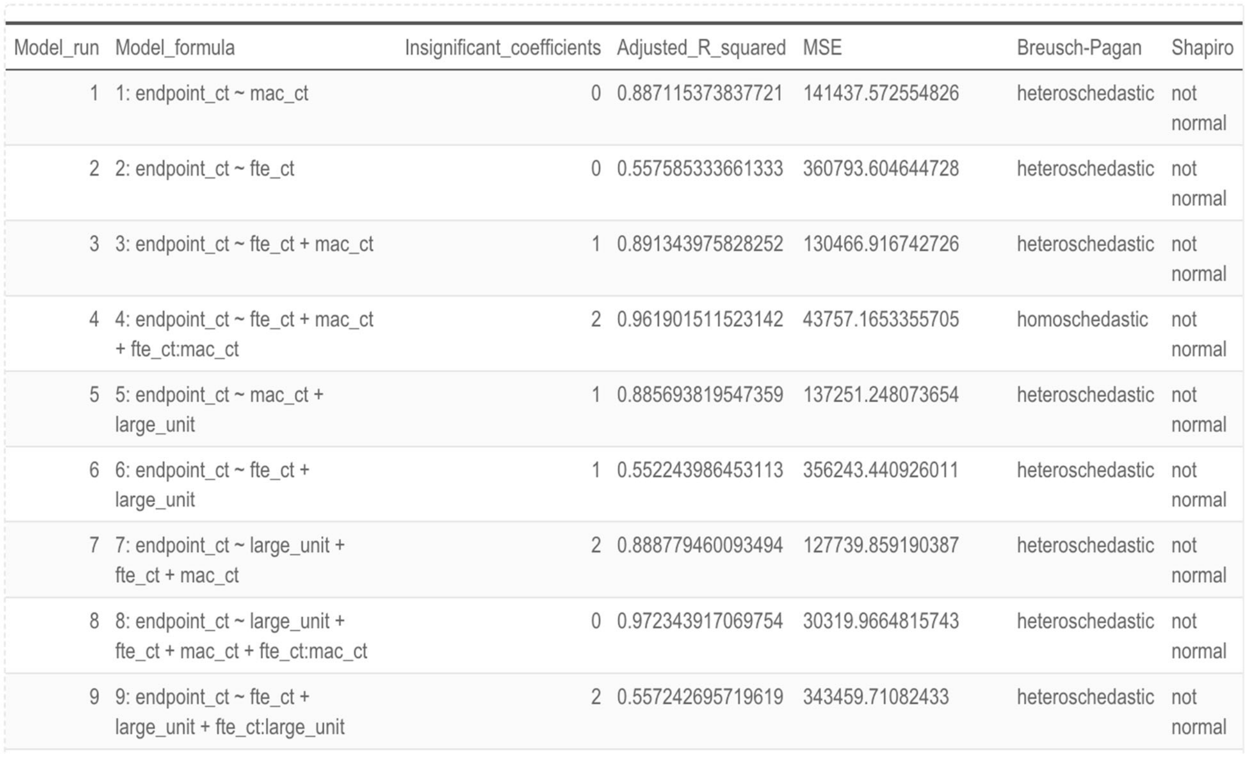


Modeling

To start with, I tried linear models with no transformations, including the following:

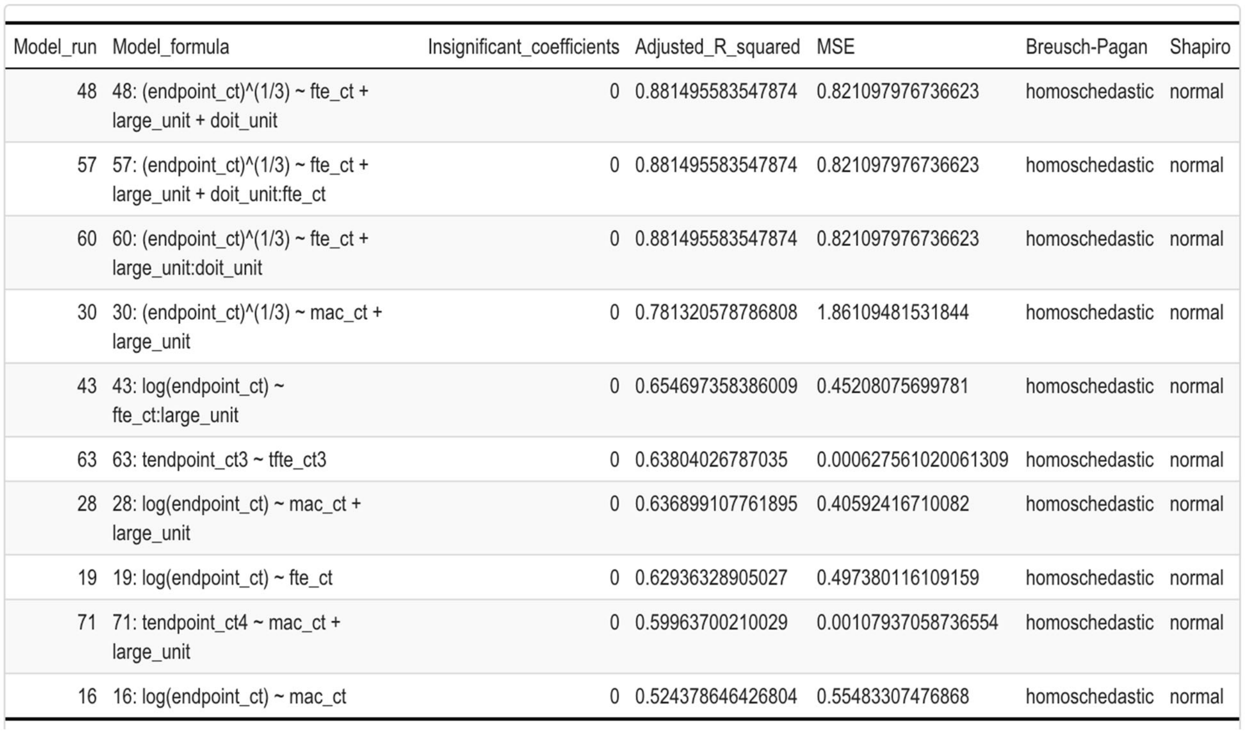
```
   'endpoint_ct ~ mac_ct',
   'endpoint_ct ~ fte_ct',
   'endpoint_ct ~ fte_ct + mac_ct',
   'endpoint_ct ~ fte_ct + mac_ct + fte_ct:mac_ct',
   'endpoint_ct ~ mac_ct + large_unit',
   'endpoint_ct ~ fte_ct + large_unit',
   'endpoint_ct ~ large_unit + fte_ct + mac_ct',
   'endpoint_ct ~ large_unit + fte_ct + mac_ct + fte_ct:mac_ct',
   'endpoint_ct ~ fte_ct + large_unit + fte_ct:large_unit',
   'endpoint_ct ~ fte_ct:large_unit',
   'endpoint_ct ~ fte_ct + large_unit + central_it',
   'endpoint_ct ~ fte_ct + large_unit:fte_ct + central_it:fte_ct',
   'endpoint_ct ~ fte_ct + large_unit:fte_ct + central_it',
   'endpoint_ct ~ fte_ct + large_unit + central_it:fte_ct',
   'endpoint_ct ~ fte_ct + large_unit: central_it’
```

The first few model results are displayed below:



Model_run	Model_formula	Insignificant_coefficients	Adjusted_R_squared	MSE	Breusch-Pagan	Shapiro
1	1: endpoint_ct ~ mac_ct	0	0.887115373837721	141437.572554826	heteroschedastic	not normal
2	2: endpoint_ct ~ fte_ct	0	0.557585333661333	360793.604644728	heteroschedastic	not normal
3	3: endpoint_ct ~ fte_ct + mac_ct	1	0.891343975828252	130466.916742726	heteroschedastic	not normal
4	4: endpoint_ct ~ fte_ct + mac_ct + fte_ct:mac_ct	2	0.961901511523142	43757.1653355705	homoschedastic	not normal
5	5: endpoint_ct ~ mac_ct + large_unit	1	0.885693819547359	137251.248073654	heteroschedastic	not normal
6	6: endpoint_ct ~ fte_ct + large_unit	1	0.552243986453113	356243.440926011	heteroschedastic	not normal
7	7: endpoint_ct ~ large_unit + fte_ct + mac_ct	2	0.888779460093494	127739.859190387	heteroschedastic	not normal
8	8: endpoint_ct ~ large_unit + fte_ct + mac_ct + fte_ct:mac_ct	0	0.972343917069754	30319.9664815743	heteroschedastic	not normal
9	9: endpoint_ct ~ fte_ct + large_unit + fte_ct:large_unit	2	0.557242695719619	343459.71082433	heteroschedastic	not normal

All results above resulted in residuals that were heteroschedastic and not normally distributed. Therefore, I tried transforming the data using Box-Cox, log, square root, and cubed root transforms. The results were much more favourable:

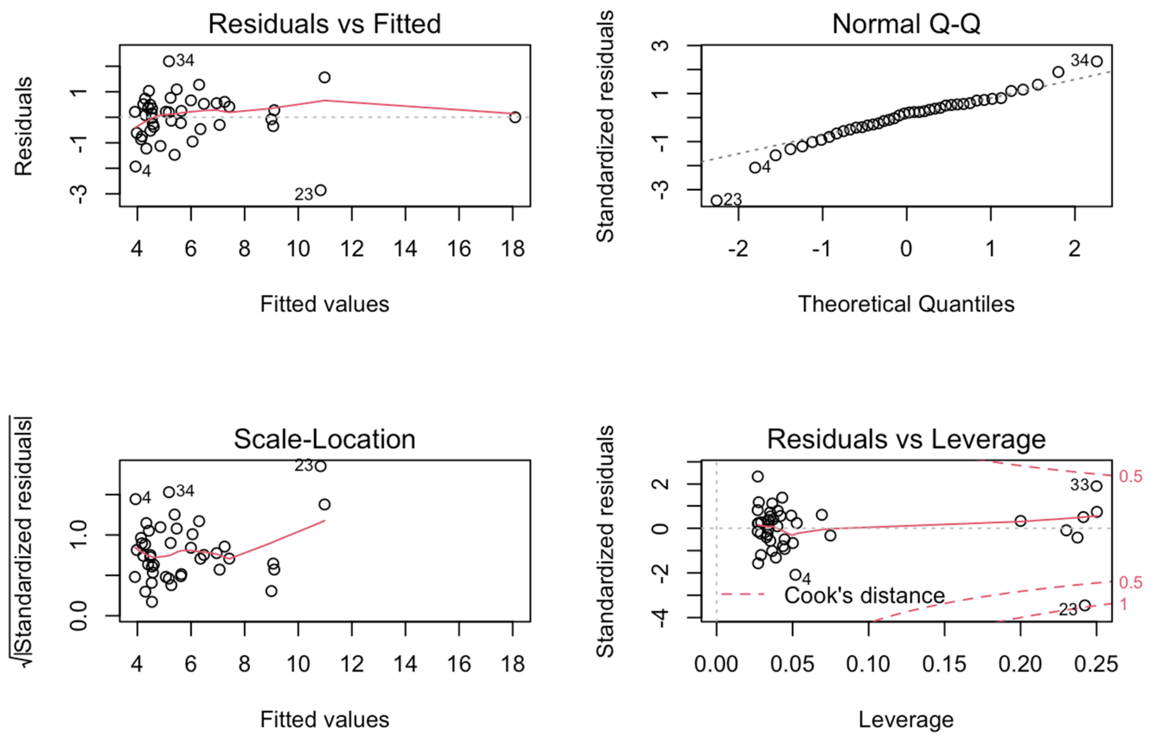


Model_run	Model_formula	Insignificant_coefficients	Adjusted_R_squared	MSE	Breusch-Pagan	Shapiro
48	48: (endpoint_ct)^(1/3) ~ fte_ct + large_unit + doit_unit	0	0.881495583547874	0.821097976736623	homoschedastic	normal
57	57: (endpoint_ct)^(1/3) ~ fte_ct + large_unit + doit_unit:fte_ct	0	0.881495583547874	0.821097976736623	homoschedastic	normal
60	60: (endpoint_ct)^(1/3) ~ fte_ct + large_unit:doit_unit	0	0.881495583547874	0.821097976736623	homoschedastic	normal
30	30: (endpoint_ct)^(1/3) ~ mac_ct + large_unit	0	0.781320578786808	1.86109481531844	homoschedastic	normal
43	43: log(endpoint_ct) ~ fte_ct:large_unit	0	0.654697358386009	0.45208075699781	homoschedastic	normal
63	63: tendpoint_ct3 ~ tfte_ct3	0	0.63804026787035	0.000627561020061309	homoschedastic	normal
28	28: log(endpoint_ct) ~ mac_ct + large_unit	0	0.636899107761895	0.40592416710082	homoschedastic	normal
19	19: log(endpoint_ct) ~ fte_ct	0	0.62936328905027	0.497380116109159	homoschedastic	normal
71	71: tendpoint_ct4 ~ mac_ct + large_unit	0	0.59963700210029	0.00107937058736554	homoschedastic	normal
16	16: log(endpoint_ct) ~ mac_ct	0	0.524378646426804	0.55483307476868	homoschedastic	normal

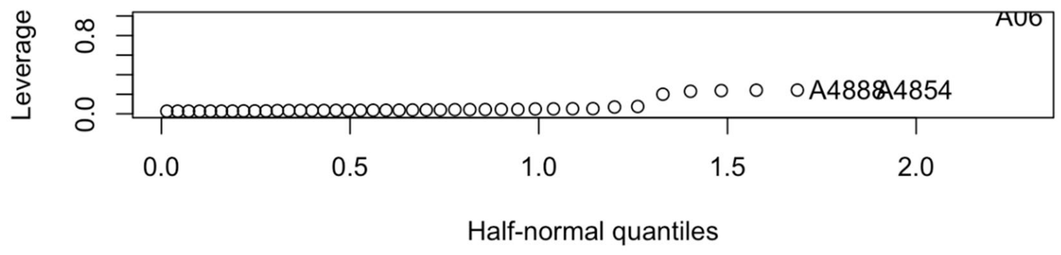
<u>Model Selection</u>

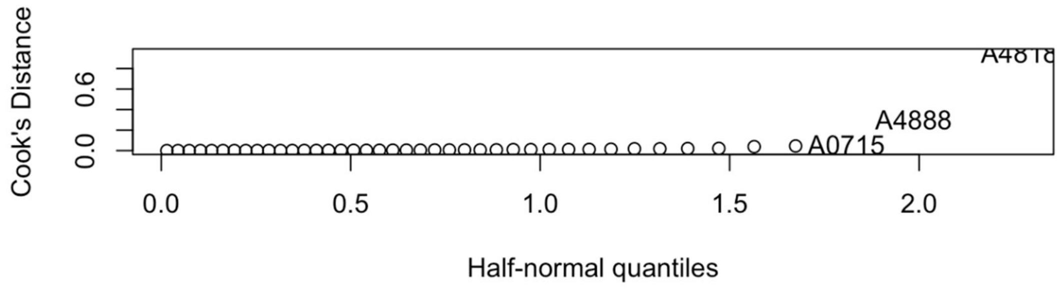
Selecting the model with the best adjusted R-squared value, residuals appear both homoschedstic and normally distributed:

48: (endpoint_ct)^(1/3) ~ fte_ct + large_unit + doit_unit



Half-normal plots showing leverage points:





The model summary is as follows:

```
## [1] "-----------------------------------------------------------------------------------------------"
## [1] "48: (endpoint_ct)^(1/3) ~ fte_ct + large_unit + central_it"
##
## Call:
## lm(formula = lmod_form[i], data = dfagg, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8602 -0.4185  0.1553  0.5189  2.1964
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.786386   0.228337  16.582  < 2e-16 ***
## fte_ct          0.016042   0.001832   8.759 9.49e-11 ***
## large_unitTRUE 2.961911   0.465233   6.367 1.60e-07 ***
## central_itTRUE  6.218259   1.225994   5.072 9.98e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9515 on 39 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:   0.89,  Adjusted R-squared:  0.8815
## F-statistic: 105.1 on 3 and 39 DF,  p-value: < 2.2e-16
```

Model analysis:

```
[1] "--------------------------------------------------"
## [1] "48: (endpoint_ct)^(1/3) ~ fte_ct + large_unit + central_it"
## lm(formula = lmod_form[i], data = dfagg, na.action = na.omit)
## [1] "Shapiro test for normality: The p-value of 0.264955772266619 is > 0.0
5, so do not reject the null; i.e., the residuals are normal."
## [1] "Breusch-Pagan test for homoschedasticity: The p-value of 0.1100138214
92618 is > 0.05 and the test statistic of 6.03303904016363 is < 10, so don't
reject the null; i.e., the residuals are homoschedastic."
## [1] "Adjusted R-squared value: 0.881495583547874"
## [1] "Mean squared error: 0.821097976736623"
## [1] "Leverage point cutoff: 0.232558139534884"
## [1] "Variance inflation factor (VIF):"
##         fte_ct  central_itTRUE large_unitTRUE
##       1.651449       1.621655       1.056396
```

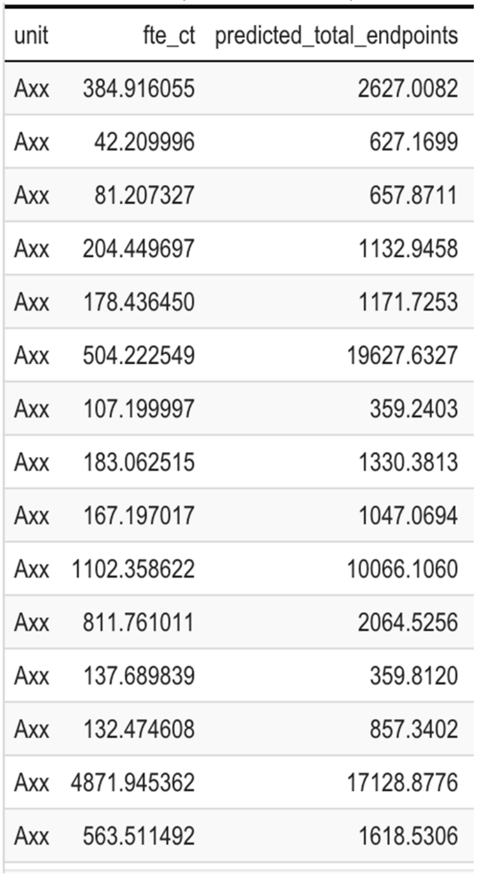
As shown, there is very strong correlation between FTE count and endpoint count. The Shapiro test confirms normality of residuals, and the Breusch-Pagan test confirms homoschedasticity. In addition, the variance inflation factor (VIF) indicates only slight collinearity among predictors, which isn’t a cause for concern.

The formula and confidence intervals for the selected model:

```
## [1] "(endpoint_ct)^(1/3) ~ fte_ct + large_unit + central_it"
## [1] "(Intercept): 3.7863864232126 +/- 0.384718839051336 (3.40166758416127,
4.17110526226394)"
## [1] "fte_ct: 0.0160415146800718 +/- 0.00308588481810908 (0.012955629861962
8, 0.0191273994981809)"
## [1] "large_unitTRUE: 2.96191104682989 +/- 0.783858809519685 (2.17805223731
02, 3.74576985634957)"
```

```
## [1] "central_itTRUE: 6.21825878930115 +/- 2.06564624528395 (4.152612544017
2, 8.2839050345851)"
```

The (redacted) predicted endpoint counts:



unit	fte_ct	predicted_total_endpoints
Axx	384.916055	2627.0082
Axx	42.209996	627.1699
Axx	81.207327	657.8711
Axx	204.449697	1132.9458
Axx	178.436450	1171.7253
Axx	504.222549	19627.6327
Axx	107.199997	359.2403
Axx	183.062515	1330.3813
Axx	167.197017	1047.0694
Axx	1102.358622	10066.1060
Axx	811.761011	2064.5256
Axx	137.689839	359.8120
Axx	132.474608	857.3402
Axx	4871.945362	17128.8776
Axx	563.511492	1618.5306

Conclusion

In conclusion, there is good statistical evidence to support a very strong linear model correlating endpoint count to FTE count, with additional categorical variables of large_unit and central_it. Commonly used techniques to evaluate the model's residuals indicate that the model is valid. However, the 90% prediction interval includes a wide range of values; while I'm redacting the exact numbers from this post, the difference between the upper and lower bounds was about 120,000, greater than the predicted mean. So while there should be no expectation that this estimate is close to accurate, it may nonetheless serve as a basis for further evaluation.