Analysis of rank-score data for the TU Delft Aerospace Selection Process*

erentar2002

2023-04-15

The exam scores on the TU Delft Aerospace Selection process are released a day before the ranks are. This day of waiting is usually extremely painful and to get around that, I have collected and compiled data from various discord and whatsapp channels to produce this document. I dont know what i am doing, so take it with a huge pinch of salt, and please do suggest better methodology. Compiling this data will be only the simplest step we can take in understanding how the entrance process really works, as very little information about it is released to the public.

Table of contents

Source: https://github.com/erentar/tud-aero-selection-rankings

*Special thanks to Django van der Plas and everyone on the TU Delft Study server. If at any time the word "we" is used, it refers to my conversations with various people in the server.

Data for previous years

Data collection

I went into all of the TU Delft discords i know of, searched for "rank". Clusters of messages around activity spikes (usually over a few days) included a lot of screenshots of scores with ranks included. These were added to a spreadsheet `data.ods`, and used to create following plots.

Analysis

Expected distribution would be a gaussian distribution. Ranking each member would involve finding the percentile of every point on this gaussian distribution, i.e how many people are above them. Cumulative distribution function of the normal distribution would give percentile of every point. To find this, one would integrate the normal distribution.

Thus, the cumulative distribution function is equivalent to the indefinite integral of the gaussian distribution, which is known as the error function erf.

The scavenged data was already ranked, the expected fit would be an erf fit.

To find the model below, sagemath is used.

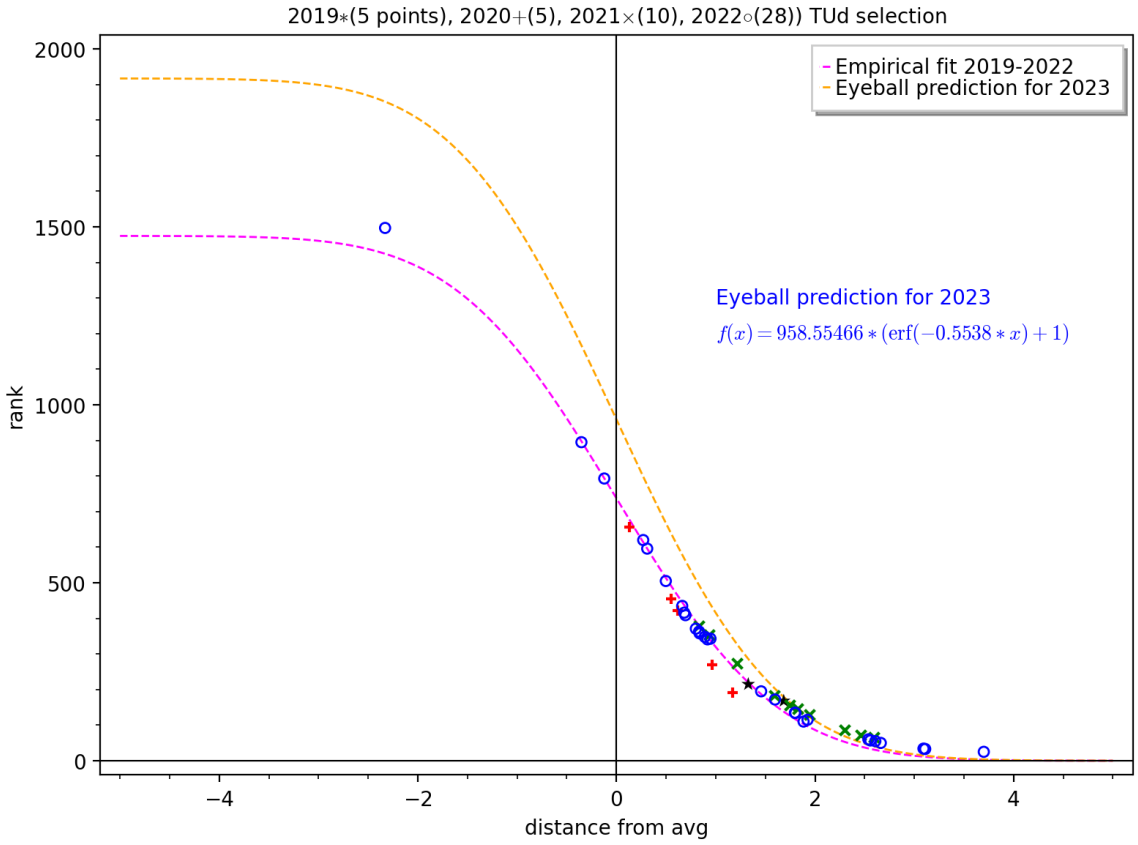
```
var("a,b")
model(x) = a * ( erf(b*x)+1 )
fit1 = find_fit(year:list,model)
```

The full code can be inspected in the source of this document.

This model seemed to be accurate to within 40 ranks when tried with data not included in the training set.

This model will be inaccurate when applied blindly to 2023 ranks, due to higher amount of people applying, as lowest rank would be higher. To have a rough idea of the landscape in 2023, we scale the model by increasing the first coefficient by the factor expected.

The ceiling (i.e the lowest rank) for 2022 was 1600. We had naively eyeballed that about 1.2-1.4 times as much people would apply this year (without any backing evidence). Thus, scaling the function respectively (by a factor of 1.3) would yield



As will be laid out in a later section, this turned out to be accurate as long as the scaling factor is correct.

Data for 2023

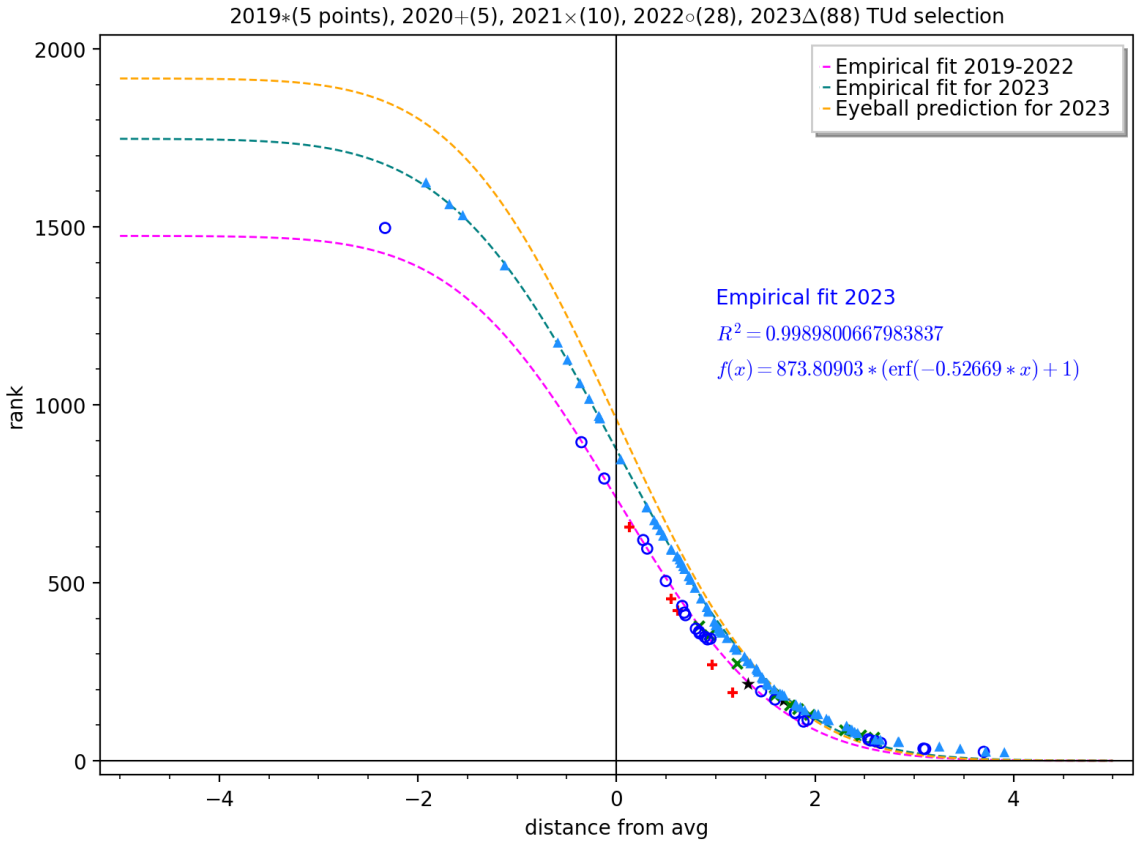
Data collection

Spammed google form https://forms.gle/tzp7KgC5CznU8Q7VA at the TU Delft discords, and also the whatsapp chat for 2023 applicants. Apologies for the annoyance.

I am willing to say it paid off, because as of 2023-04-16, there are 92 responses (some of which were unusable due to trolling and/or invalid entries), which is great. Thanks to everyone who participated and donated data. I wish i could credit everyone individually. The persons who included their name will be in the thanks section.

Analysis

Same as the previous section, data is different while the code is identical.



Playing around

Now that i knew the fit line for 2023, i had a rough idea how many more people applied.

My fit for 2019-2022 yielded

```
1474.6994782364534
```

as the lowest rank.

The e-mail sent last year showed that 2300 people applied and about 1800 finished the mini-mooc. I suspect that some people dropped after that as well, however a drop of

```
325.3005217635466
```

seems a little too much.

Yet still, let us asssume this number is correct for now.

The highest rank yielded by the 2023 fit is

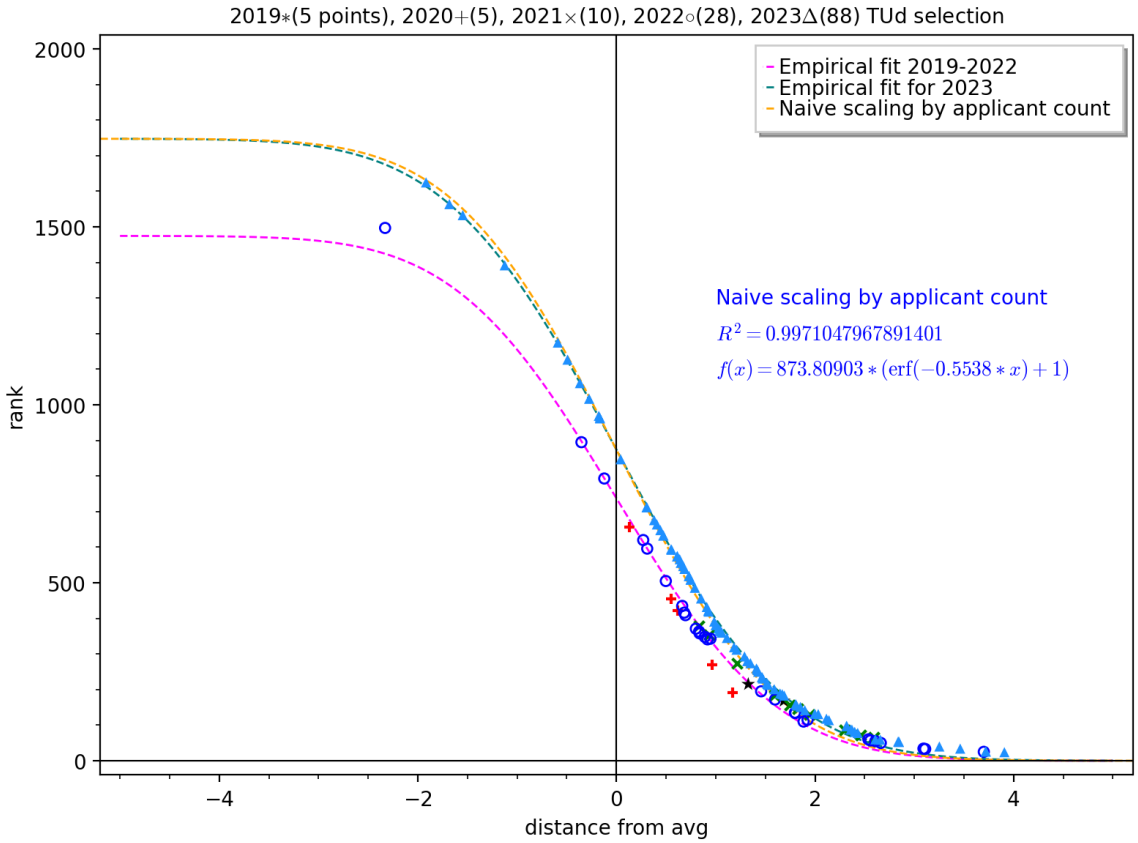
`1747.6180518955043`

If we are to assume this number is correct,

`1.1850672477252284`

times as much people applied.

My eyeball estimation was 1.3 times. Instead if my naive prediction had used the number above as the coefficient, the following line would appear

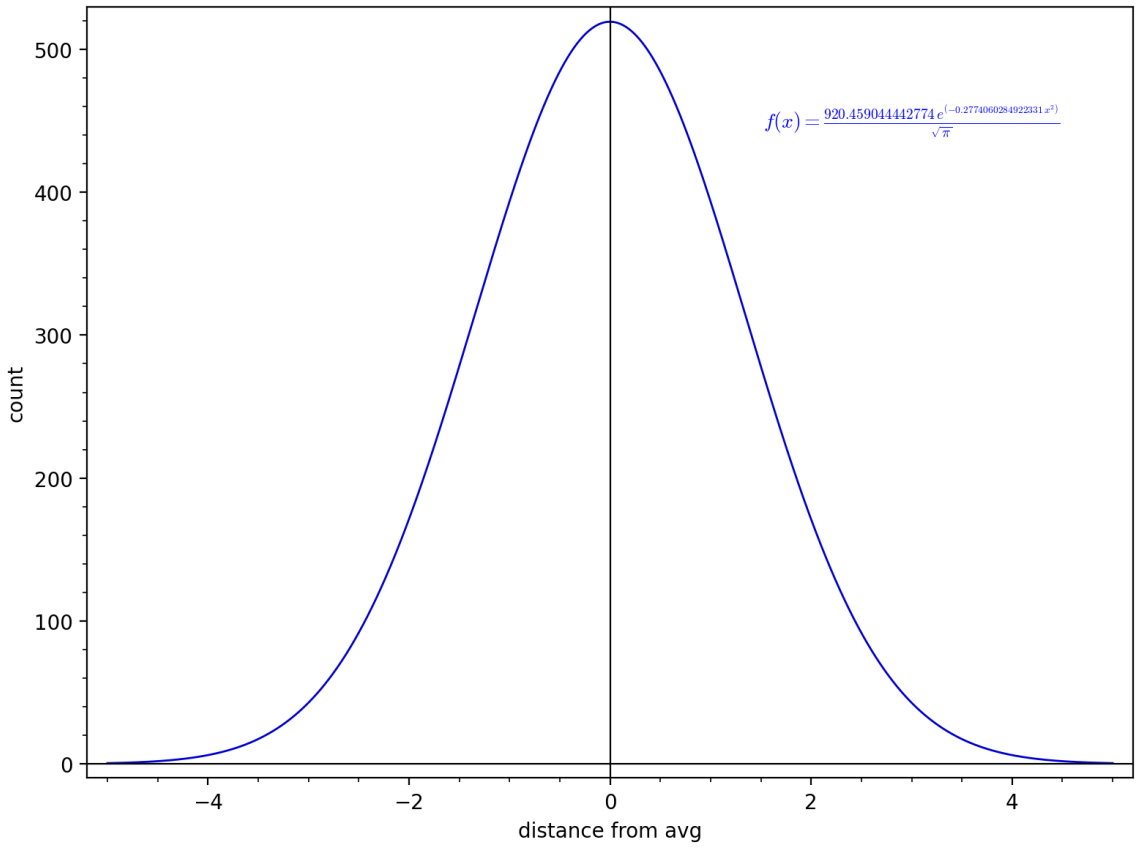


This is very accurate, probably because the underlying statistical distribution didnt change much.

Extracting statistical characteristics of the fit

The fit provided is a numerical fit, and does not give the standard deviation.

Deriving the fit would yield the underlying bell curve, as can be seen below



The standard deviation is found by solving

$$-1 * \frac{d}{dx}\left(\texttt{fit2023}\right) == \texttt{ceil2023} * \frac{1}{\sigma * \sqrt{2\pi}} * \exp\left(-\frac{1}{2} * \left(\frac{x}{\sigma}\right)^2\right)$$

which yields

$\sigma = 1.34253944581253$

Conclusion

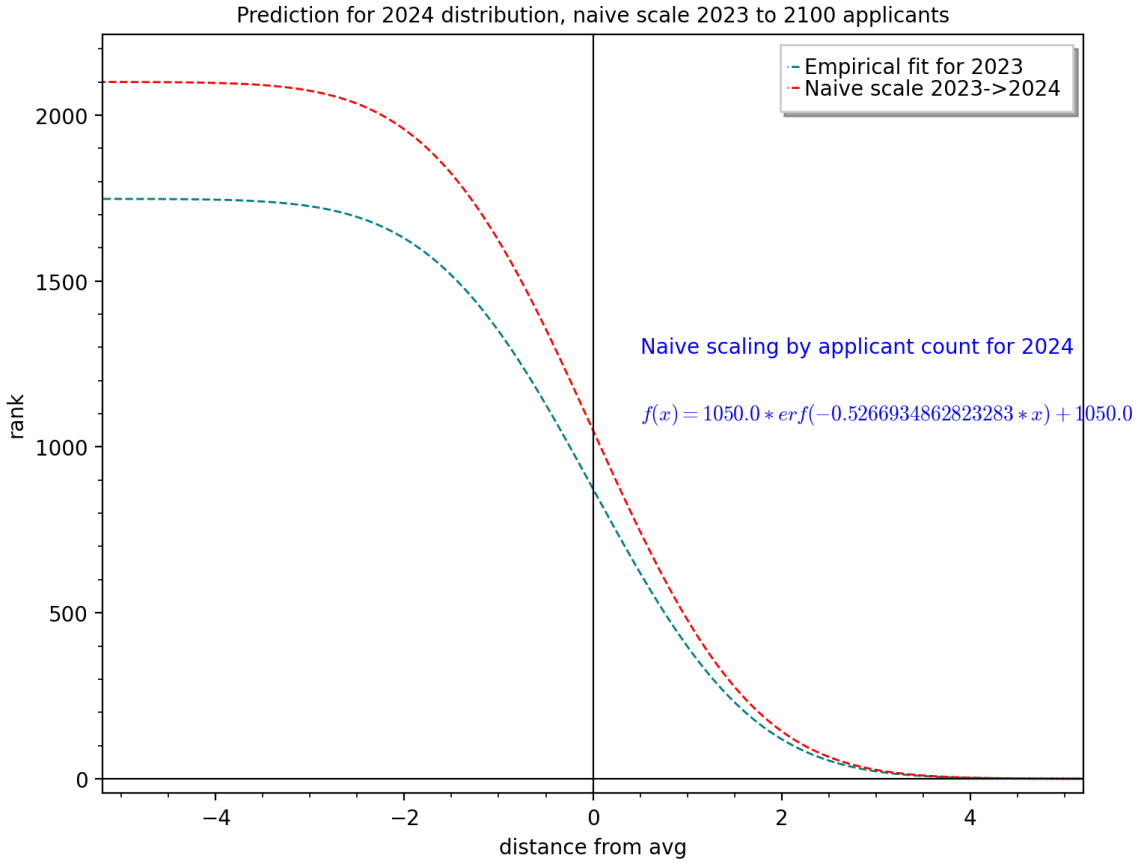
Predicting rank using this methodology, i.e getting the last year's curve and scaling it proportionally to this year's applicant count, seems to work reliable enough for estimating whether one gets in or not.

The greatest limiting factor in applying this method to future years would be the fact that the applicant counts are not released, makes this method rather inaccurate.

2024

Predictions

Since naive scaling to applicant count is the best guess, assuming the distribution characteristics have not changed.



This fit yields for the 440 cutoff point as being

1.08487207118712

This number is the distance from average. To get the score required for 440, we add the average score from 2024.

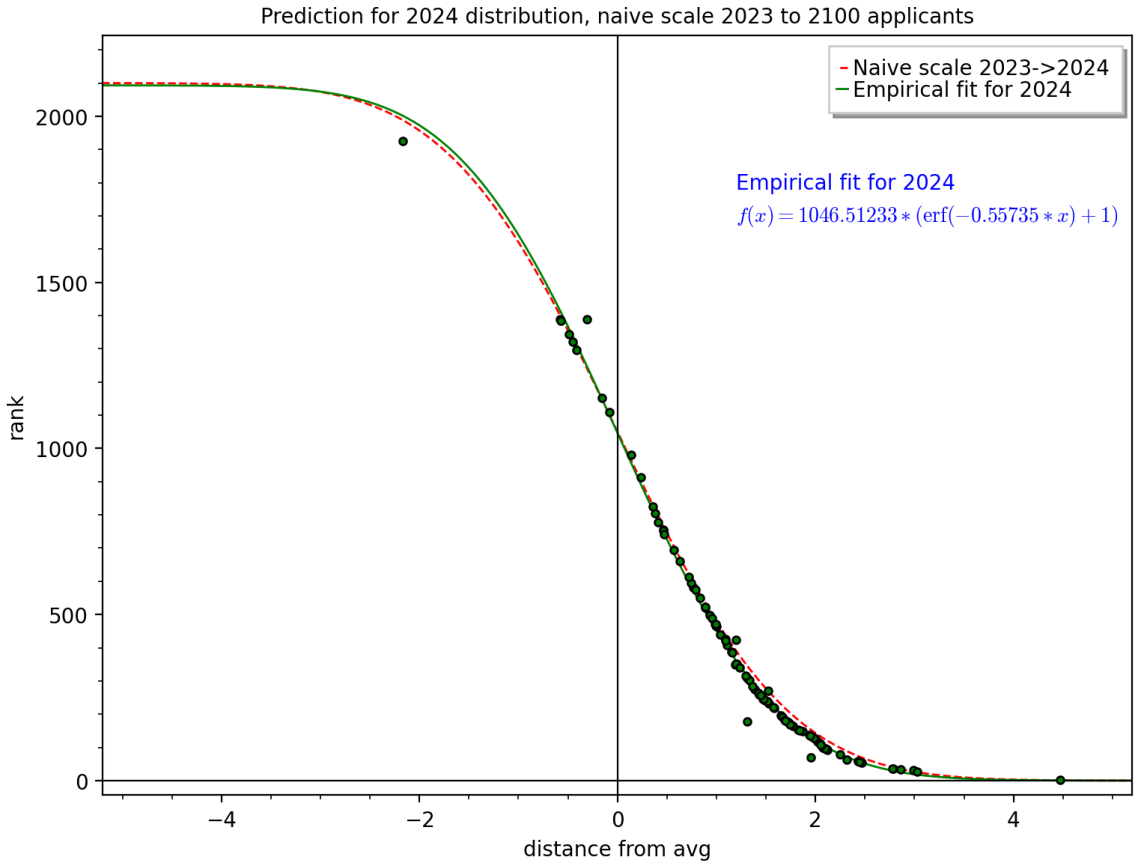
6.17287207118712

Based on this model, this is the score to be ranked 440.

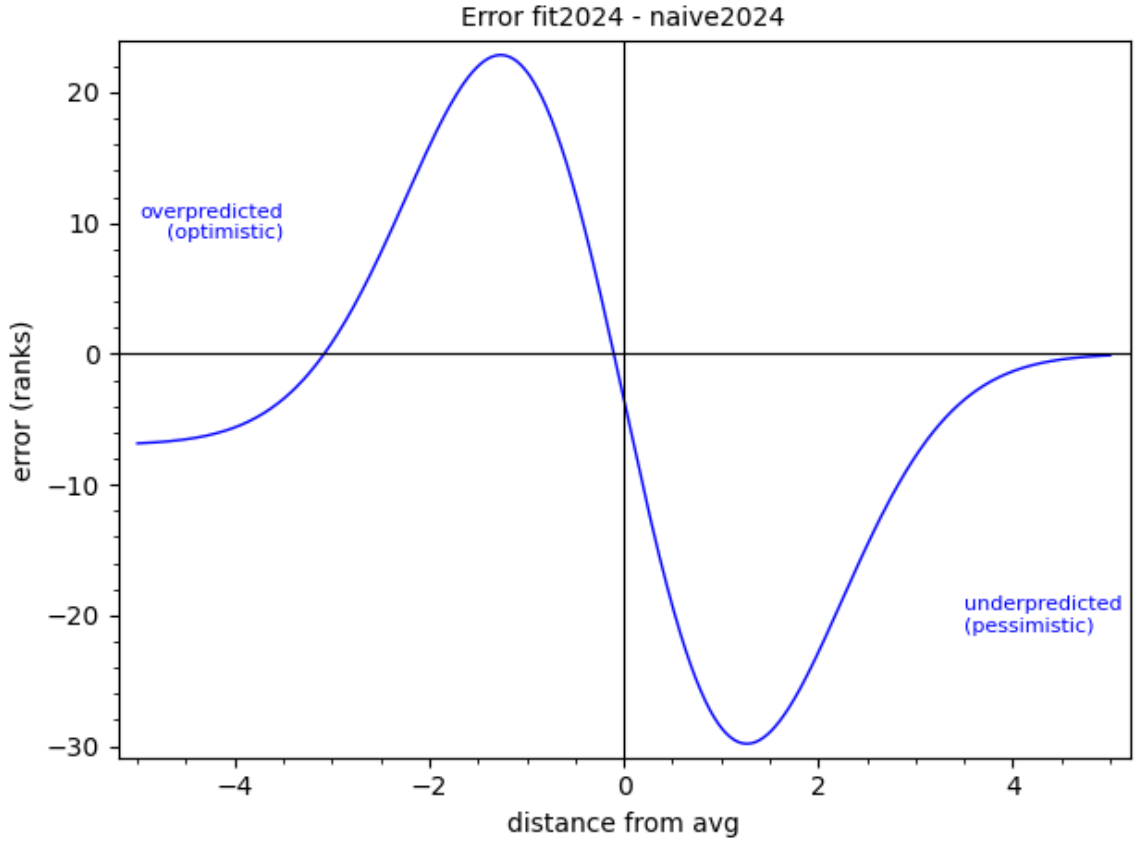
Comparison to empirical data

Data was collected in the exact same fashion as last year, spammed this google form

raw data from the spreadsheet
also at **`data2024.ods`**



The error between the naive scale and empirical fit is shown below



thats pretty fuckin accurate