Simulating the Robustness of Various Combination Algorithms for Funding Opportunity Cost-Effectiveness

Koji Flynn-Do

August 2022

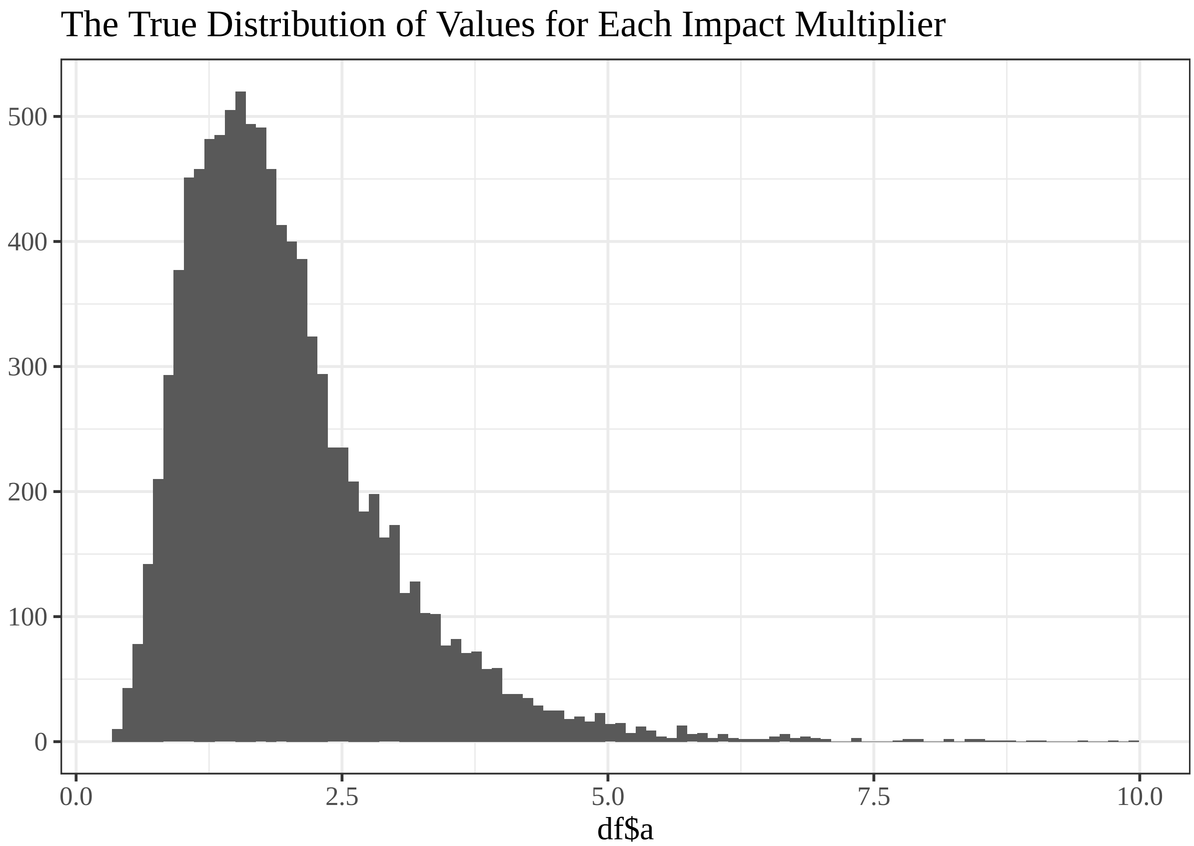
1. Simulate Underlying Data

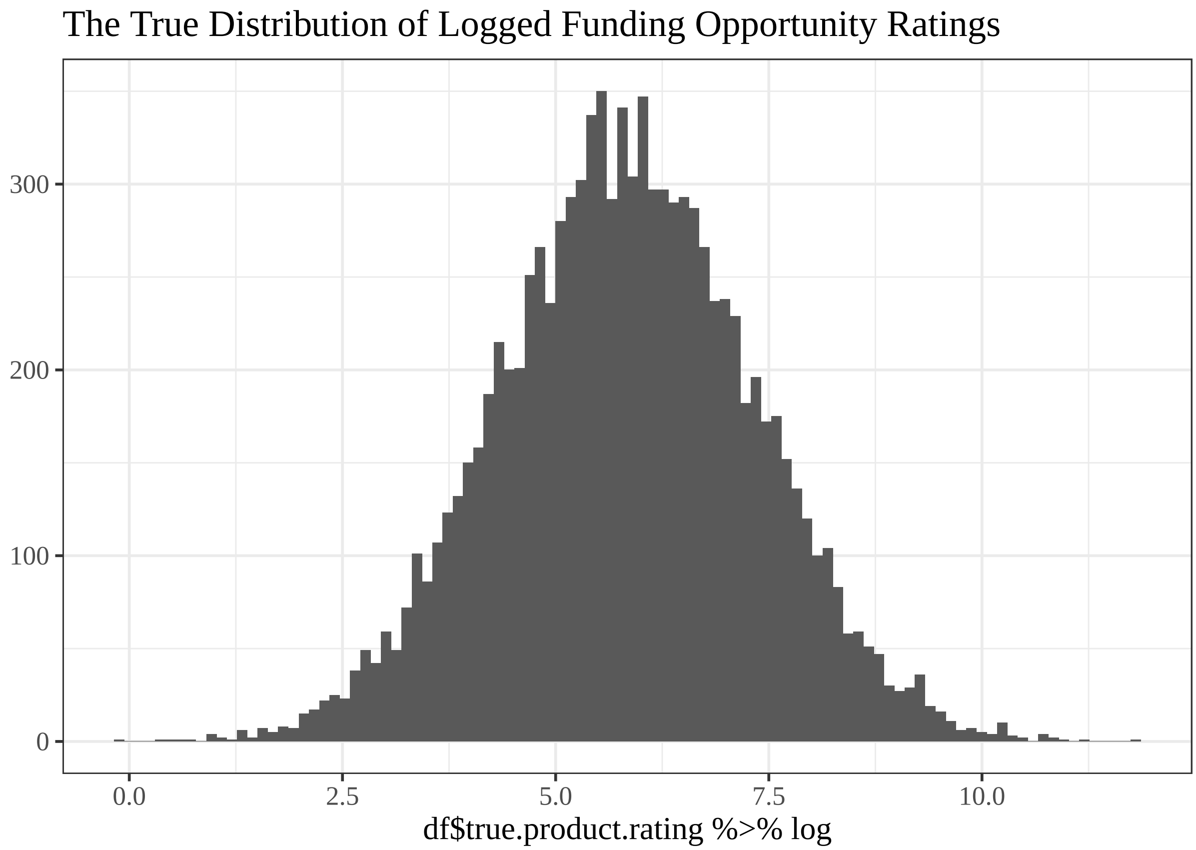
I simulated 10,000 funding opportunities, each with cost-effectiveness described by 10 impact multipliers ("attributes"). The impact multiplier values are all drawn from a log-normal distribution with mean 2 and standard deviation 1.

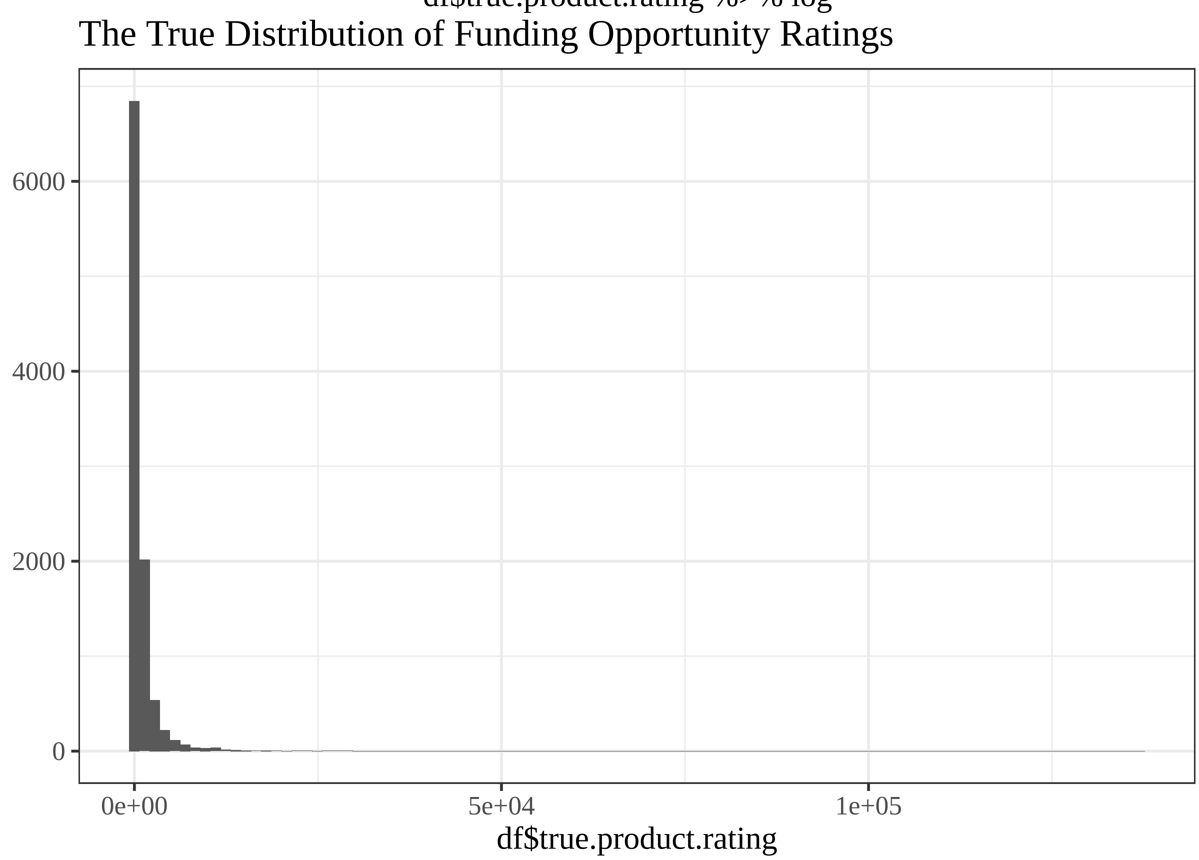
With those true impact multipliers, I computed the true rating of each funding opportunity and the true rankings.

```
set.seed(13)

mean <- 2
sd <- 1

log_mean <- log(mean^2 / sqrt(sd^2 + mean^2))
log_shape <- sqrt(log(1 + (sd^2 / mean^2)))

n = 10000

components_str <- c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j")

for (component in components_str) {
  assign(x = component,
         value = rlnorm(n = n,
                 meanlog = log_mean,
                 sdlog = log_shape))
}

components <- c(a, b, c, d, e, f, g, h, i, j)

# true rating if the data generating process is multiplicative
true.product.rating <- a*b*c*d*e * f*g*h*i*j

# true rating is the data generating process is multiplicative and additive
true.mixed.rating <- a*b*c*d*e + 5*f*g*h*i*j


ratings <- list(true.product.rating, true.mixed.rating)

# for (r in ratings) {
#
```

```r
#   cat('mean: ', mean(r), ', median: ', median(r), ', sd: ', sd(r), '\n\n')
# }


df <- data.frame(a, b, c, d, e, f, g, h, i, j, true.product.rating, true.mixed.rating)

df['true.rank.product'] <- rank(-df$true.product.rating)
df['true.rank.mixed'] <- rank(-df$true.mixed.rating)
```




The True Distribution of Logged Funding Opportunity Ratings
300
200
100
0
0.0
2.5
5.0
7.5
10.0
df$true.product.rating %>% log


The True Distribution of Funding Opportunity Ratings
6000
4000
2000
0
0e+00
5e+04
1e+05
df$true.product.rating

2. Simulate Noisy Measurements of Each Attribute

I simulate various kinds of noise to see if the results are different.

- One version has noise *inversely proportional to the true value.* Maybe larger values should have smaller noise, since bigger multipliers seem like they should be easier to detect.
- Another version has *noise equal to the square root of the true value.* I don't have a motivating causal model for this, I just thought it'd be worth checking.
- The final three have *constant noise*, no matter the true value of the specific attribute. The first of these has *sd = 0.125*, the second has *sd = 0.25*, and the third has *sd = 1.* Recall that each attribute is lognormally distributed with a mean of 2 and a standard deviation of 1.

3. Apply Various Combination Rules to Get Ratings and Rankings

For each noise-scenario, I combine the noisy estimates to get ratings in two ways:

- Adding all the terms together
- Multiplying all the terms together

Then, for each true underlying data-generating process, I compute the difference between the rank given by each combination rule and the true rank.

- Noise Inversely Proportional to True Value (`.p`)
- Noise Equal to Square Root of True Value (`.s`)
- Constant Noise 0.125 (`e`)
- Constant Noise 0.25 (`q`)
- Constant Noise 1 (`o`)

4. Compare Performance of Combination Rules In Different Scenarios

a. The Underlying True Ratings Are the Product

Here, the true cost-effectiveness is given by the product of all ten attributes.

I check the performance of the additive and multiplicative models in each noise scenario and at three percentiles: all the funding opportunities, the top 10% of funding opportunities (by *true* rating), and the top 1% of funding opportunities (by *true* rating).

Standard Error of the Ranking

This is an ordinal consideration. Note that there are 10,000 synthetic funding opportunities, so the top 1% is 100 funding opportunities. We get the standard error by taking standard deviation of the ranking error (the difference between the ranking given and the true ranking).

Correlation of Ratings With True Ratings

b. The Underlying True Ratings Are Mixed

Here, the true cost-effectiveness is given by the product of the first five attributes and 5 times the product of the last five attributes. I'm just doing the exact same thing as above.

Table 1: The SE of the ranking, so 10 means ranking is off by 10 spots on average. True data structure is multiplicative Attribute Mean, SD: 2 1

	Inv. Prop.	Square Root	0.125	0.25	1
All Funding Opportunities					
Multiplicative	2029	2469	496	864	2219
Additive	1968	2725	944	1080	2346
Top 10%					
Multiplicative	1033	1615	178	324	1299
Additive	798	1810	344	418	1385
Top 1%					
Multiplicative	275	934	24	56	644
Additive	224	1175	72	81	589

Table 2: Correlation Between Log(True Ratings) and Log(Estimates). True Data Structure is Multiplicative Attribute Mean, SD: 2 1

	Inv. Prop.	Square Root	0.125	0.25	1
All Funding Opportunities					
Multiplicative	0.76	0.66	0.96	0.99	0.72
Additive	0.76	0.58	0.93	0.95	0.69
Top 10%					
Multiplicative	0.59	0.38	0.86	0.95	0.42
Additive	0.64	0.36	0.77	0.81	0.41
Top 1%					
Multiplicative	0.54	0.28	0.80	0.93	0.36
Additive	0.59	0.24	0.68	0.75	0.42

Table 3: The SE of the ranking, so 10 means ranking is off by 10 spots on average. True data structure is mixed. Attribute Mean, SD: 2 1

	Inv. Prop.	Square Root	0.125	0.25	1
All Funding Opportunities					
Multiplicative	2436	2755	1654	1786	2578
Additive	2322	2913	1682	1759	2626
Top 10%					
Multiplicative	2017	2138	1484	1572	2069
Additive	1563	2148	1181	1217	1855
Top 1%					
Multiplicative	1596	1675	848	937	1423
Additive	788	1501	453	516	1077

Table 4: Correlation Between Log(True Ratings) and Log(Estimates). True data structure is mixed. Attribute Mean, SD: 2 1

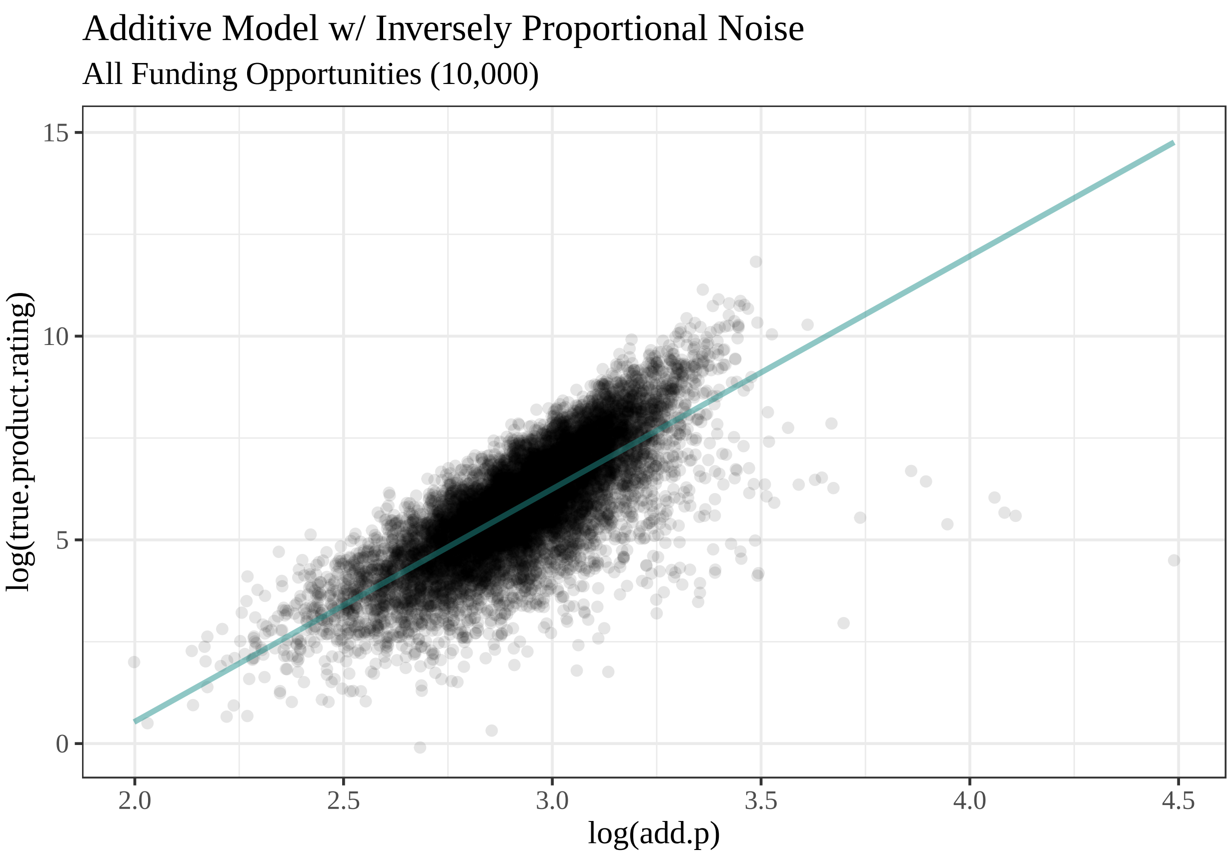
	Inv. Prop.	Square Root	0.125	0.25	1
All Funding Opportunities					
Multiplicative	0.65	0.56	0.81	0.84	0.61
Additive	0.68	0.51	0.82	0.84	0.60
Top 10%					
Multiplicative	0.23	0.19	0.29	0.31	0.19
Additive	0.36	0.22	0.42	0.43	0.27
Top 1%					
Multiplicative	0.11	0.01	0.15	0.15	0.02
Additive	0.33	0.11	0.36	0.36	0.19

Standard Error of the Ranking

Correlation of Ratings With True Ratings

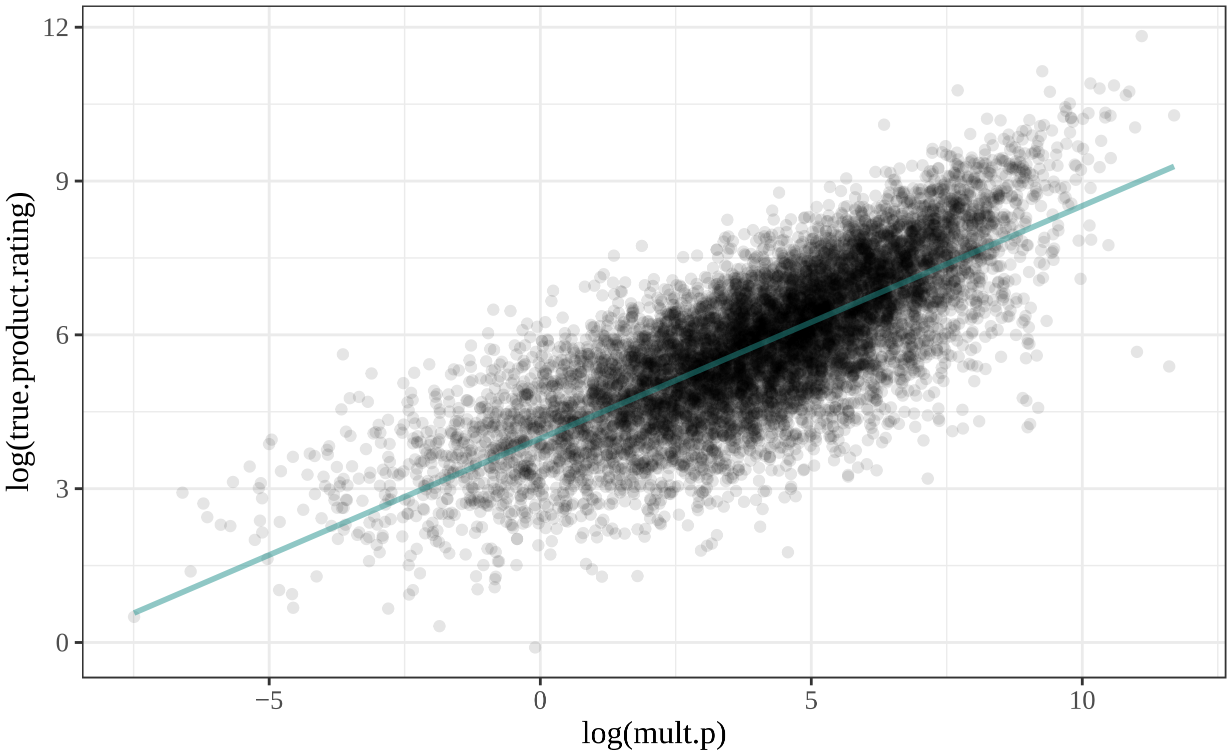
5. Some Charts

I think that the first situation, where noise is inversely proportional to the size of the multiplier, is the most like the one we're in.



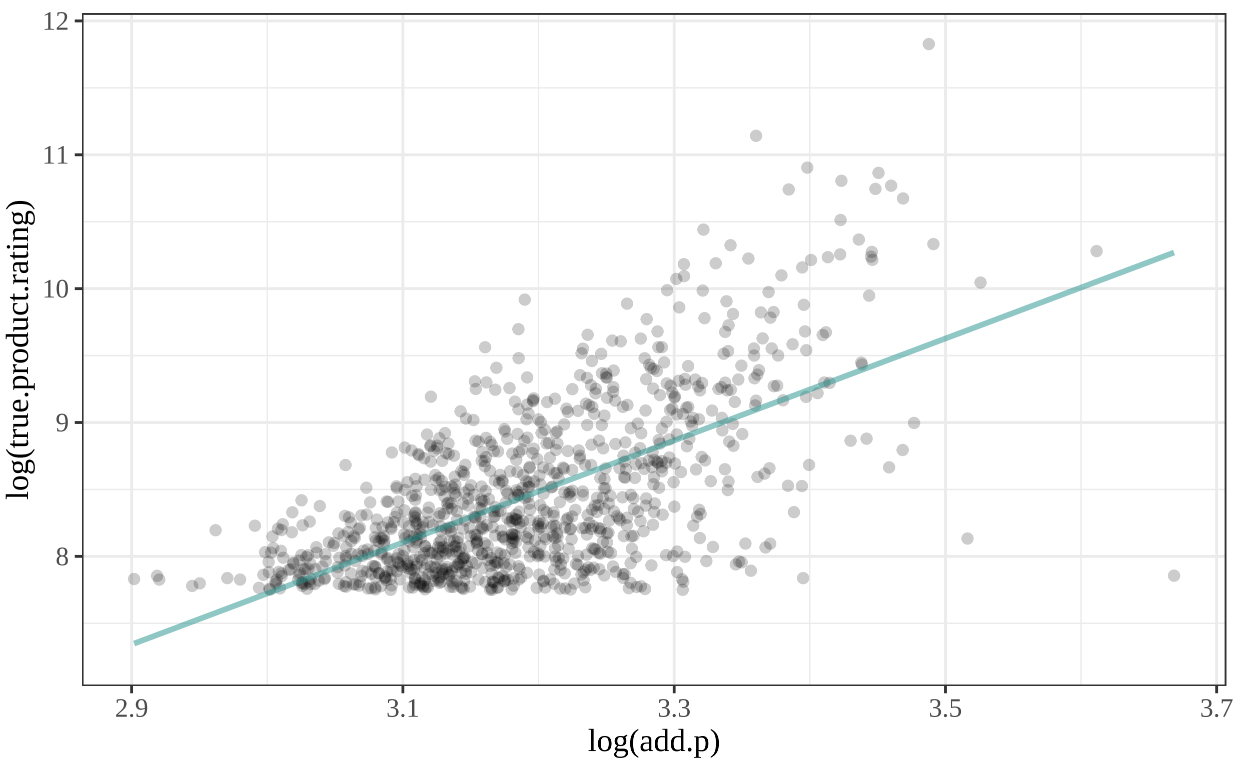
Multiplicative Model w/ Inversely Proportional Noise

All Funding Opportunities (10,000)



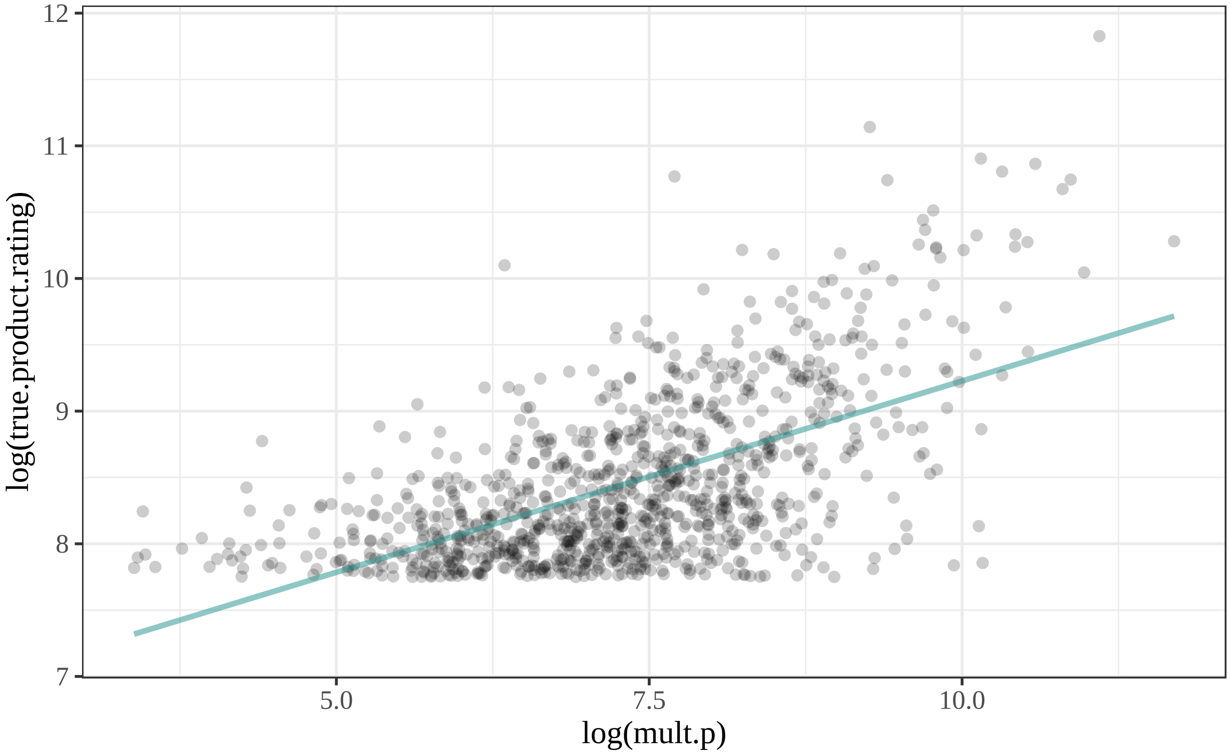
Additive Model w/ Inversely Proportional Noise

Top 10% of Funding Opportunities (1,000)



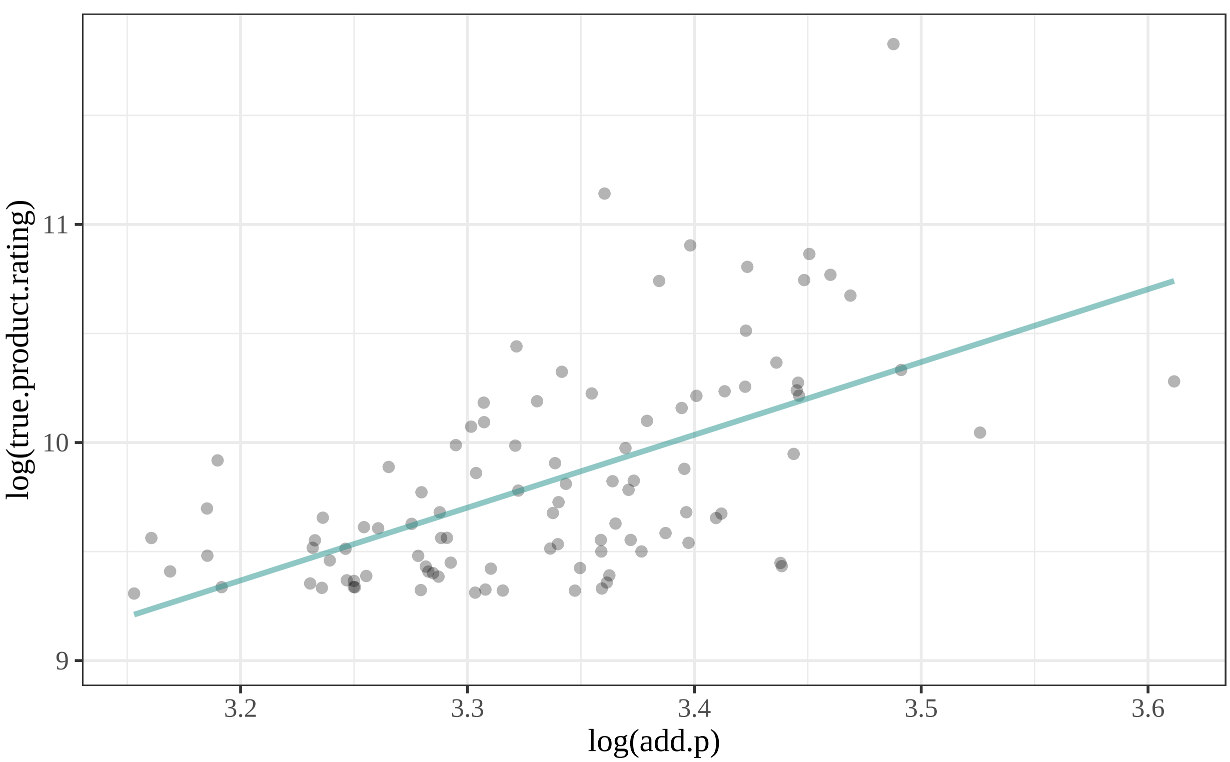
Multiplicative Model w/ Inversely Proportional Noise

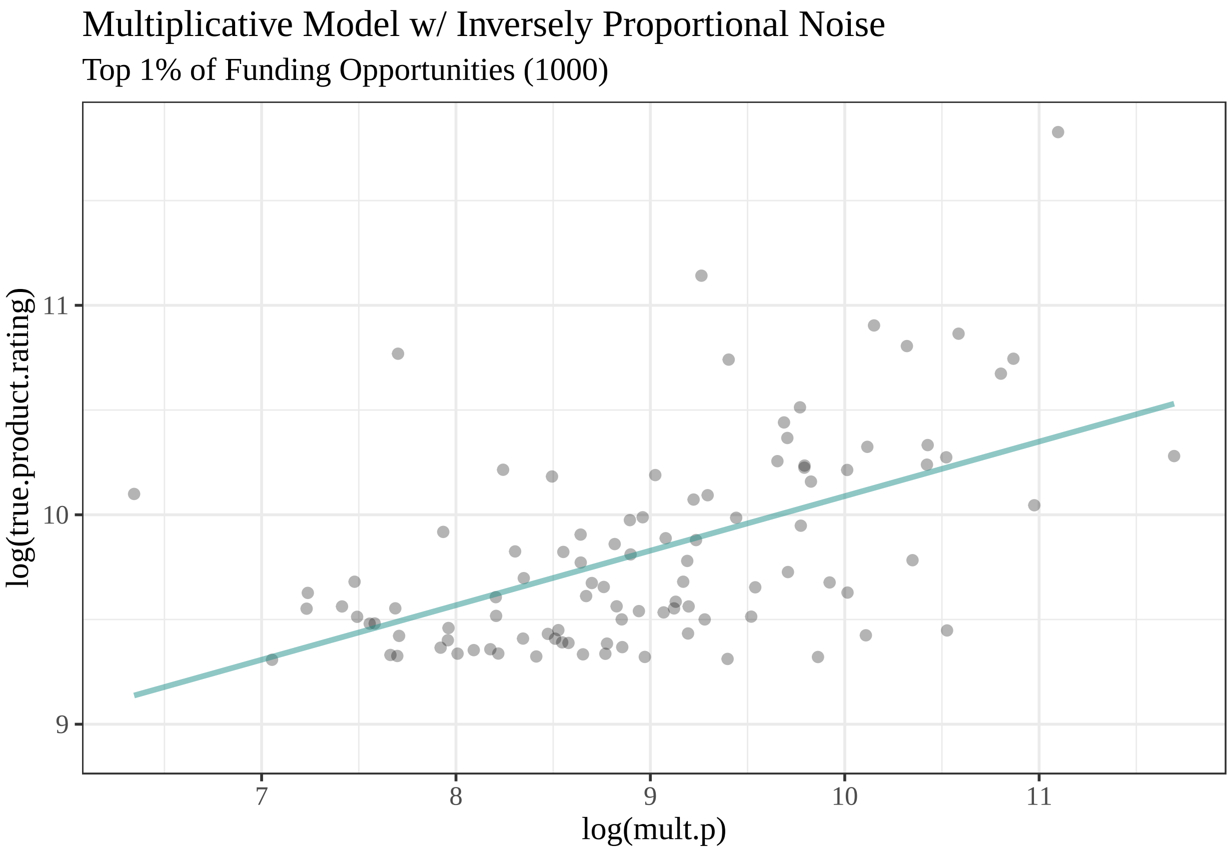
Top 10% of Funding Opportunities (1,000)



Additive Model w/ Inversely Proportional Noise

Top 1% of Funding Opportunities (1000)





Multiplicative Model w/ Inversely Proportional Noise
Top 1% of Funding Opportunities (1000)
log(true.product.rating)
11
10
9
7
8
9
10
11
log(mult.p)