Title: DD: The DRS bot data show that X and XX apes as a collective are a massive whale
Author: yurimtoo
Created 2022-01-03 21:06:52 UTC
Permalink: /r/GME/comments/rvc1vd/dd_the_drs_bot_data_show_that_x_and_xx_apes_as_a/
Url:
https://www.reddit.com/r/GME/comments/rvc1vd/dd_the_drs_bot_data_show_that_x_and_xx_apes_as_a/

Hello apes. I'll keep this short and to the point. No rockets. Minimal crayons. Maximum tit jacking.

TA;DR: By aggregating the data from the DRS bot posts, we can infer the distribution of shares held by apes. Phrased differently, we can answer the question, "What percentage of apes hold N shares?" for all values of N. Based on the DRS bot data as of 3 January 2022 19:07 GMT, we can say that ~22.3% of apes are X holders, ~47% of apes are XX holders, ~27.9% of apes are XXX holders, and <3% of apes are XXXX+ holders. [X and XX apes have DRS'd ~11% of the total DRS'd shares. By comparison, the largest whale that has fed the bot (40k shares!) has just ~2.73% of the total DRS'd shares. As a collective, X and XX holders are a massive whale; as a group they are larger than all XXXXX whales combined!](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/shareholders_piechart.png) Feeding the bot will help improve the accuracy of these results, so I encourage all apes to do so whether they have 1 share or 1000 shares. These results will be updated in the future [here](https://github.com/yurimtoo/stonkde). BUY HOLD DRS. THIS IS NOT FINANCIAL ADVICE, I BRUSH MY TEETH WITH CRAYONS

Before getting into the details, I need to acknowledge u/geppetto123. This brilliant ape is responsible for the core of this post, [as they brought up this idea in the first place](https://www.reddit.com/r/Superstonk/comments/qofvwh/liquidity_crisis_for_kenny_approx_39_of_the_free/hjnr5px/?context=3). [A couple months ago, they wrote a solid post on the DRS bot data.](https://www.reddit.com/r/Superstonk/comments/qpri4l/drs_dd_statistics_are_in_why_small_apes_matter/) Not only that, they wrote the orignial code that I built off of to create what you'll see below. Make sure you share some of those internet points with them. And of course, a big thank you to u/Roid_Rage_Smurf for running the bot and making this possible.

Now, let's get to it.

1. Introduction: Why should you trust anything I have written?

Great question. You shouldn't trust me. You should trust the code and the data. [That's why all of the code used for what is in this post is public.](https://github.com/yurimtoo/stonkde) If you are skeptical about the claims made here, go look at the code and verify for yourself exactly what is being done. I'm happy to answer any questions if anything is unclear.

My everyday work is applying statistics to real-world problems. I've published papers utilizing the techniques that will appear below. In this post, we'll be applying those methods to the DRS bot data.

You might be wondering, why should we even trust the DRS bot data? If so, [you didn't read u/geppetto123's post, did you?](https://www.reddit.com/r/Superstonk/comments/qpri4l/drs_dd_statistics_are_in_why_small_apes_matter/) You should go do that. TA;DR: the data looks reliable, no obvious fuckery, X and XX apes as a collective are a whale. The text/figures below do not introduce much new material in that sense, but rather confirms those earlier findings and build upon that work in some ways.

2. Methods

While the [Github repo](https://github.com/yurimtoo/stonkde) provides the exact methodology, here is a plain-text summary:

1. Aggregate and download the DRS bot data
2. Process it into a list of DRS'd shares of each user that has fed the bot
3. Analyze the data
1. Create a normalized histogram of the data, showing the percentage of users who have DRS'd some number of shares within each bin.
2. Perform a kernel density estimation (KDE) to infer the underlying distribution of shares held by apes

4. Plot the results

2.1 ELIA: what are histograms and kernel density estimations?

If you are unfamiliar with these methods, here's the basics of it.

A histogram shows you the number of occurrences of each data value within some data set. Phrased differently, histograms answer the question, "What is the distribution of values within this data set?" For example, if I have a data set like [0, 1, 1, 1, 1, 1, 4, 7, 7, 7], we can see that one of the values is 0, five of the values are 1, one is 4, and three are 7. If we had a histogram with 8 bins, then it would plot exactly as I described it: the bins for values of 2, 3, 5, and 6 would have 0 occurrences, while the other bins would have the aforementioned counts. If the histogram had just 2 bins, then we would see bin 1 (which would span values 0 -- 3) with a count of 6, and bin 2 (which would span values 4 -- 7) with a count of 4. So, in this simple example, we can see that the choice of bins used to create the histogram will have an effect on the resulting plot; we'll look at this more in Section 3. The important part of this is to understand that the histogram logs the number of counts in a given bin, and each bin spans some range of values. The distribution of counts for each of these bins can then be normalized to give the percentage of cases in the data set falling within each bin -- like, for example, what percentage of apes have just 1 share DRS'd, or what percentage of apes are X holders, etc.

KDE is kind of an extension of this. Rather than assume the available data is the exact distribution we are after, it assumes that the finite set of samples in the data set were drawn from the true underlying distribution. KDE attempts to infer that true distribution from the subset of data we have available (which could be over- or undersampled in certain areas). KDEs answer the question, "What distribution was this data set sampled from?"

My hope is that these descriptions are clear enough for all apes to have enough context to understand Section 3. So, if the above descriptions are unclear, I would suggest searching online for resources that describe these techniques in better detail, or ask questions.

2.2 KDE details

Unlike the previous section, this subsection may not make sense to some apes, and that is okay. This subsection is solely to provide additional details to apes with the familiarity to make sense of it.

The most important of the parameters of a KDE are the bandwidth of the kernel and, to a lesser degree, the kernel itself. Here, I am using a Gaussian kernel. Standard approaches of determining the kernel bandwidth are those of Scott's Rule and Silverman's Rule, which are generalized to work in higher dimensions. Here, I am using a modification to Scott's Rule. Scott's Rule defines the kernel bandwidth to be

(N_data)**(-1. / (d + 4))

where N_data is the number of samples in the data set, and d is the dimensionality of the data. I am modifying this to

(N_data)**(-1. / (d * 2))

This provides equivalent values for d=4, and it avoids the main issue with using Scott's Rule or Silverman's Rule that I will demonstrate and discuss in Section 3. Admittedly I do not like that I don't have an analytical proof to support this choice, but empirically it provides more reasonable results. And ultimately, bandwidth selection is arbitrary, so maybe it's not such a big deal that I don't have a proof for this. In my work, Scott's Rule has consistently worked, and consequently I haven't done a deep dive into the literature on the topic of bandwidth selection for KDE. If any apes are knowledgeable on this topic, please share that knowledge if you think the methodology here can be improved. And if I have unknowingly made some grave mistake here, do please point that out so that we can correct it ASAP.

The resulting KDE is normalized such that it integrates to 1 over the range of possible sharecounts (1 --

40022 as of this post) when using trapezoidal integration. While it might be expected that the KDE should already integrate to 1 (the histogram integrates to 1 by nature, and the KDE is similar to that...), the reality is that the KDE places some probability density outside of the aforementioned range due to the kernel behavior. We as humans know that there is no possible probability for < 1 shares DRS'd, so this normalization step takes that into consideration. As a side effect, it also causes 0% probability for someone to DRS more than the current posted maximum (40022 shares), which we know isn't true -- DFV and RC both hold more than that. However, the KDE doesn't know that, and as a result the KDE's probability density in this regime is negligible.

3. Results and Discussion

3.1 Histograms

Now, let's look at the data. Histograms can be sensitive to the chosen bin sizes, so let's take a look at the histogram with different bin choices:

[Normalized histogram of the data, sharecounts < 1000](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_binsize-comp_under1k.png) [(plot of all sharecounts)](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_binsize-comp_full.png)

[Same as above, but with a logarithmic scaling for the y-axis](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_binsize-comp_under1k_ylog.png) [(all sharecounts)](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_binsize-comp_full_ylog.png)

We can see that for all bin choices that there are few XXXX and XXXXX whales compared to X and XX holders, as expected. While small bin counts over-smooth the data and large bin counts introduce noise in the distribution, we can see that the results are generally consistent. For all following plots, we'll be using the histogram with 10000 bins. The biggest whale thus far has 40022 shares, so this puts each bin having a range of about 4 shares.

3.2 Statistics derived from the DRS bot data

Here are some fun stats from the DRS bot data at the time of this post:

- 8882 apes have fed the DRS bot (it's very possible that our code to pull the data has missed some apes)

- the median ape (50th quantile) holds 38 shares. Other quantiles: 10th holds 2 shares, 25th holds 10 shares, 33rd holds 16 shares, 69th holds 97 shares, 90th holds 312 shares, 95th holds 600 shares, and 99th holds ~2000 shares

- ~7.85% of holders have just 1 share -- and they are the most common holder!

- ~22.3% of DRS'd apes are X holders

- ~46.9% of DRS'd apes are XX holders

- Or put differently... X and XX holders make up **~69%** ■ of apes that have fed the DRS bot

- ~27.9% of DRS'd apes are XXX holders

- <3% of DRS'd apes are XXXX+ holders

- There are 9 XXXXX whales, which makes up ~0.1% of apes that have fed the bot

- [X+XX apes hold ~11% of DRS'd shares, XXX apes hold ~44% of DRS'd shares, and XXXX+ apes hold ~45% of DRS'd shares. **X+XX apes have DRS'd more shares than XXXXX whales!**](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/shareholders_piechart.png)

- Apes holding 750 or fewer shares make up 50% of all DRS'd shares

This last statistic surprised me. I think it is most easily explained by sampling bias, that is, I assume that XXX+ whales are more likely to post their positions for karma compared to X or XX holders. For example, I am an XX ape that hasn't posted their position (YET...that changes with this post), and I know other X and XX holders that haven't posted their positions. There are surely many others in that camp. As a result, that would skew this last statistic towards larger values than actual, so I wouldn't put much weight on that number. Regardless, the third- and second-to-last statistics highlight the importance of X and XX apes. If this group is being undersampled as I suspect, then they make up an even more significant portion of holders, and by extension would hold a more significant fraction of DRS'd shares. If X and XX apes feed the bot, that will help improve the accuracy of these results.

3.3 KDEs

Let's look at the KDE to perhaps get a better picture of things.

As I mentioned in Section 2, for the KDE I am using a modification to Scott's Rule. Let's see why I made this decision by looking at the KDEs using Scott's Rule, Silverman's Rule, the proposed modification, and further adjustments to the proposed modification:

[Comparison of KDE bandwidth selection methods, sharecounts < 1000](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_with_kde_under1k.png) [(full plot)](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_with_kde.png)

[Comparison of KDE bandwidth selection methods, log-scaled](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_with_kde_log.png)

We can see that Scott's Rule and Silverman's Rule oversmooths the data and shifts the data peak to larger values. I have used Scott's Rule numerous times in the past without this oversmoothing occuring, but those situations differed from this situation in one primary way: the DRS bot has been fed by fewer than 9000 apes, whereas the data I have worked with in the past was on the order of millions of samples. As a result, I believe that the sparsity is throwing off the Scott/Silverman bandwidth methods, and visually we can see that the adjusted method better captures the behavior seen in the histogram. We know from the DRS posts that apes gravitate towards round numbers like 100, 200, 500, etc. shares; the Scott/Silverman methods do not capture that behavior, while the modification to Scott's Rule does capture that behavior.

As a hand-wavy proof of the modification to Scott's Rule, I also included two "alternate" modifications: Alternate 1 where the exponent's denominator is changed from `d * 2` to `d * 1.5`, and Alternate 2 where the exponent's denominated is changed to `d * 3`. Alternate 1 yields a KDE that very closely follows the histogram; the kernel bandwidth here is small enough that effectively the KDE arrives at nearly the same result as a histogram. On the other hand, Alternate 2 more closely resembles the results for the Scott and Silverman methods while still having slight sensitivity to the peaks seen in the data, which supports the idea that our modification works as intended, at least for 1-dimensional data.

However, there are regimes here where I think the Scott and Silverman methods provide a better picture of things, and that is when the data becomes sparse for sharecounts > 1000. [This plot](https://raw.githubusercontent.com/yurimtoo/stonkde/master/post/figs/histogram_with_kde_log_400-40k.png) illustrates this: the modification that worked well for sharecounts < 1000 begins to overfit the data, whereas the Scott and Silverman methods diffuse some of the probability into surrounding values. I suspect that a variable-bandwidth KDE will be able to combine the best of both worlds and improve on these results, but I haven't had enough time to consider it in deep enough detail to feel confident including that in this post. Maybe there will be a part two if people are interested.

Also, **I want to be clear that I am not saying that any of these bandwidth selection methods are the 'right' or 'best' one**; that would require knowing the true distribution. Only Computershare knows the true distribution, so the best we can do is speculate and consider the range of possibilities.

How will these results change over time? We'll have to wait and see. I am in the process of automating the analysis presented here to repeat on a regular schedule at the [Github

repo](https://github.com/yurimtoo/stonkde), so we will see each day how new feedings to the DRS bot change these results.

4. Conclusions

So, what are the main takeaways here? According to the DRS bot data,

1. X and XX apes are roughly 69% of all DRS'd GME holders, with 1-share apes making up ~7.9% of holders. **Together, X+XX apes hold ~11% of all DRS'd shares, which is more than all XXXXX whales combined.** The largest whale that has fed the bot holds just ~2.73% of all DRS'd shares. **As a collective, X+XX holders are a whale.**

2. XXX apes hold roughly the same number of DRS'd shares as XXXX+ apes.

3. The accuracy of these results depends entirely on the number of apes that feed the bot. I encourage all apes to feed the bot; even just 1 share helps. For apes without much karma, GMEOrphans allows you to feed the bot.

"Talk is cheap, it takes money to buy whiskey." The apes that have read this post know that I haven't fed the bot yet. [So here is me buying that whiskey.](https://imgur.com/a/mcmOj9Y) 100% DRS'd. Shoutout to u/PilgrimBradford1620 for getting me to DRS my first share back in August.

There are still many avenues for additional research on the DRS bot data. If any apes with experience in programming, statistics, and/or machine learning are interested in collaborating, please DM me and let's make it happen. If anyone would like to help improve the code in the [Github repo](https://github.com/yurimtoo/stonkde), feel free to open issues/pull requests, or DM me.

To my fellow DRS'd apes, see you beautiful fuckers on the moon ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ (did you really think I wasn't going to use rocket emojis in this post?)

BUY, HOLD, DRS. THIS IS NOT FINANCIAL ADVICE, I MIX CHEERIOS WITH MELTED CRAYONS FOR BREAKFAST.

Edit: formatting