Utilizing Data Science To Assess Software Quality

Renato Martins

renatopmartins@gmail.com

Abstract

Data Science is a "hip" term these days. Most people automatically associate Data Science with financial and customer behavior applications. It turns out that Data Science provides a set of very powerful tools that can be used to assess software quality, especially in the cloud/online world.

Most cloud companies that supply online applications use the “canary” release technique to assess release quality on a limited set of hosts before fully deploying a new release candidate. These online applications also provide a great wealth of metric data that can and should be used to assess the quality of the release candidate before it is fully deployed. A green set of passing test case results is not enough to make the deployment call. Many release engineering teams find themselves staring at graphs trying to assess if the new release candidate build is at least as good as the current production build. Sometimes what they think they see is not what it really is.

This paper will go over breaking down metric data and statistical methods that will aid in making quality decisions and boost confidence on go/no-go deployment decisions. It will cover recommended methods, the process used to select the methods, and key learning points.

Biography

Renato Martins has over 15 years of experience in the software industry where he has worked on many types of devices and applications, from embedded systems to highly available and scalable cloud systems. He spent 11 years at Microsoft in many different software quality roles. He was also the CTO and Co-Founder of Stringr Inc, a crowd-sourced video marketplace for media outlets, and he also ventured outside of the software industry by co-founding and operating a Craft Brewery in Seattle, WA. Renato currently works for Groupon, Inc. as a Senior Engineering Manager on the set of services that form Groupon’s Relevance platform.

Renato holds a bachelor’s degree in Electrical Engineering from Inatel, a master’s degree in Computer Science from DePaul University, and a master’s degree in Business Administration from the Wharton School at the University of Pennsylvania.

1 Introduction

Most Cloud Companies utilize a “canary” release technique to assess the quality of a new release before it is rolled out to all of the production hosts. This technique, allegedly, comes from underground mining. Before high tech sensors and other sorts of elaborate protection mechanisms, miners brought a canary (a kind of bird) down to the mining grounds. Every now and then one would “poll”, or observe, the canary. If the bird was lively, they’d carry on. If the bird was lethargic or even dead, they’d get out of there as soon as possible, hopefully remembering to bring the bird along if it was still alive.

In cloud computing we use a similar technique in which we deploy a new release candidate to a small set of hosts, the least amount possible, and then we compare the metrics between a host running a new release candidate and one running the current production release. In doing that we find ourselves staring at graphs and trying to determine if what we are looking at is okay or not. That’s easy when we’re looking at very stable metrics, like disk utilization. But when it comes to more volatile metrics, the decision making process is not always simple and we find ourselves drawing lines and other types of artifacts to help us make a decision.

In order to improve this decision-making process and help make decisions one would not regret later, we can adopt Data Science concepts and tools. With the adoption of such concepts and tools, the release process reliability has been shown to improve dramatically as well as the level of confidence in making go/no-go decisions.

2 Picking The Tools

When looking at the realm of Data Science tools we can find ourselves overwhelmed by the sheer amount of tools available out there. Naïve Bayesian models, K Clustering, Linear and Logistic Regression, etc. It’s easy to get lost and be overwhelmed. Where do we start? How do we decide which are the right and best methods? We have to get down to the basics and use the KISS (Keep It Simple S…) concept. The rest will follow.

Take a look at the graphs below showing error rates of two different hosts running two different builds; one is a canary and one is a reference host. They compare the 3XX, 4XX and 5XX error rates between the two hosts. If you had to pick the best one, which one would you pick? Why?

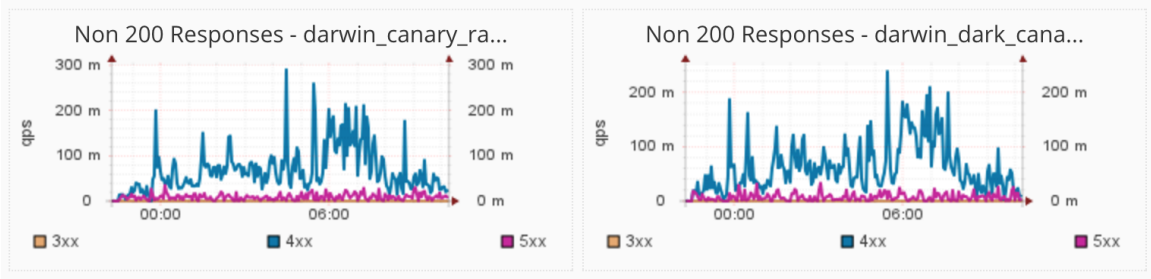

Figure 1 - Error Rates From Canary And Reference Hosts

Sometimes things are not really what they appear to be. Or are they? The goal of this paper is to remove any guessing from the picture by teaching the reader to utilize tools to bring better visibility into the data and make better informed decisions that can stand challenges.

2.1 Percentiles, Mean, Median, and Standard Deviation

We are able to easily calculate these metrics for each data set, so why wouldn't we? This is like putting an X-Ray through the data to help us see where it's more volatile and where it differs the most between the reference and the canary.

The percentiles to be used need to include some basic reference ones, like 25th, 75th, and 95th percentiles. The other percentiles depend on what metric you're using and what is the expectation your clients have about that metric. For example, if your service needs to have 99% availability, then you need to calculate the 99th percentile as well to make sure that the data up until that point meets the *fill in the metric* expectations.

Let's take the previous two graphs showing the same metrics for the same period of time. How do you figure out which one is better? Is one even better than the other or about the same? Take a look. Can you tell?

2.1.1 5XX Metrics Comparison

Metric	Canary Host	Reference Host
00th percentile	0.00000	0.00000
25th percentile	0.00002	0.00004
75th percentile	0.00355	0.00471
90th percentile	0.00643	0.00890
95th percentile	0.00923	0.01172
Mean	0.00236	0.00300
Median	0.00110	0.00159
Standard Deviation	0.00317	0.00377

2.1.2 4XX Metrics Comparison

Metric	Canary Host	Reference Host
00th percentile	0.00000	0.00000
25th percentile	0.01020	0.01029
75th percentile	0.02911	0.02823
90th percentile	0.04800	0.04993
95th percentile	0.06251	0.05956
Mean	0.02202	0.02193
Median	0.01739	0.01697
Standard Deviation	0.01734	0.01772

2.1.3 Source Of Metrics

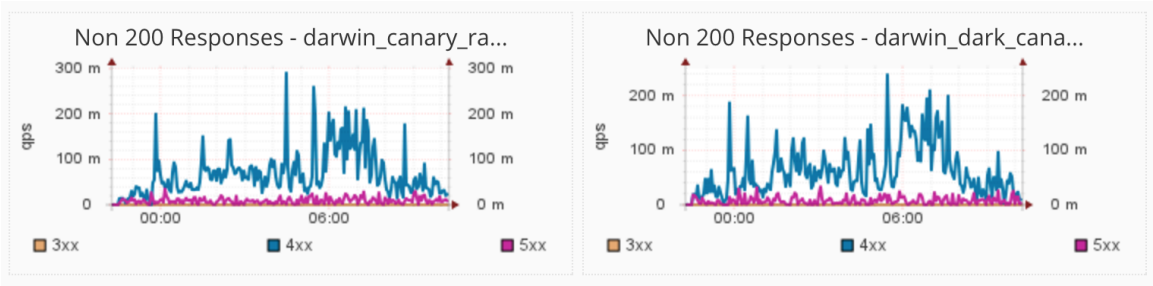

Figure 1 - Error Rates From Canary And Reference Hosts

2.1.4 Conclusions from the breaking down of percentiles, mean, median, and standard deviation

2.1.4.1 5xx

In terms of 500s, the canary host appears to be better. How can we tell? All metrics are better, varying from up to 40% better in the case of the 25th percentile, down to about 16% better in case of standard deviation. Because of that, it is already safe to assume that the build under test there improves this metric by non-trivial amounts, and therefore it should be considered for deployment.

2.1.4.2 4xx

In terms of 400s, the canary host is almost identical, without any significant difference in the metrics between the two hosts. Because of that, it is safe to assume that the new build under test there does not significantly impact this metric; therefore it should be considered for deployment.

2.2 Percentiles, Mean, Median, and Standard Deviation Ratios

Now that we have started breaking down the percentiles, mean, median, and standard deviation of the data, it only makes sense to calculate ratios (new/reference), so it's even easier to spot and quantify differences.

The previous section surfaced that the 5XX metric is better, so let's take a look at the ratios.

2.2.1 5XX Metrics Comparison With Ratios

Metric	Canary Host	Reference Host	Canary/Reference
00th percentile	0.00000	0.00000	0.00000
25th percentile	0.00002	0.00004	0.42086
75th percentile	0.00355	0.00471	0.75467
90th percentile	0.00643	0.00890	0.72199
95th percentile	0.00923	0.01172	0.78787
Mean	0.00236	0.00300	0.78600
Median	0.00110	0.00159	0.69066

Standard Deviation	0.00317	0.00377	0.84007

By adding the ratios, we now have a quantifiable way to express how much better each metric is and thus make a much more informed decision on the merits of shipping this build or not.

2.3 Boxplot

Sometimes looking at a table full of numbers is not so appealing and it can also make it difficult to determine when differences are too much or too little. Boxplots remove that guesswork and the saying "*a picture is worth a thousand words*" (source unknown) couldn't be truer.

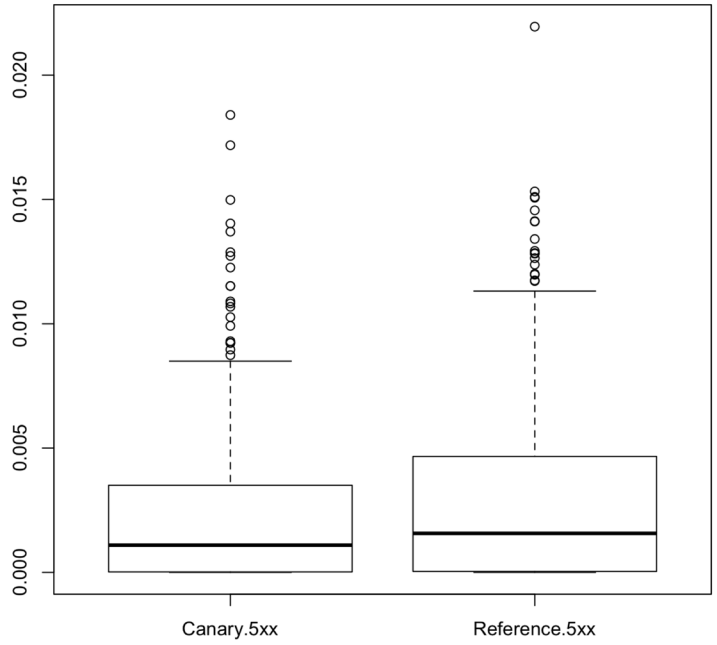

Figure 2 - Boxplot Showing Data Distribution Of Canary And Reference Hosts

The boxplot above gives us a few extra insights here:

- The canary mean is below the reference build.
- The canary's 75th percentile is below the reference build.
- The canary's max percentile is below the reference build.
- The canary's outliers also seem to be consistently below the reference build.

The conclusion from this boxplot comparison is that the canary build is slightly better than the reference build and thus should be considered for deployment.

2.4 Histograms

Following on the thought and subsequent conclusion that a picture is worth a thousand words, we can plot out histograms of the data so we can observe any yet unnoticed insights about the data.

Histograms can be a very powerful tool because, if done right, they'll always be unique for each data set and that, invariably, gives us new insights. Some of those insights include the following:

- The data should not be normally distributed.
- Are there any meaningful gaps in the data set?
- Where is the highest concentration of samples?
- Where is the lowest concentration of samples?

Let's take a look at the current data set and see what their histograms show us.

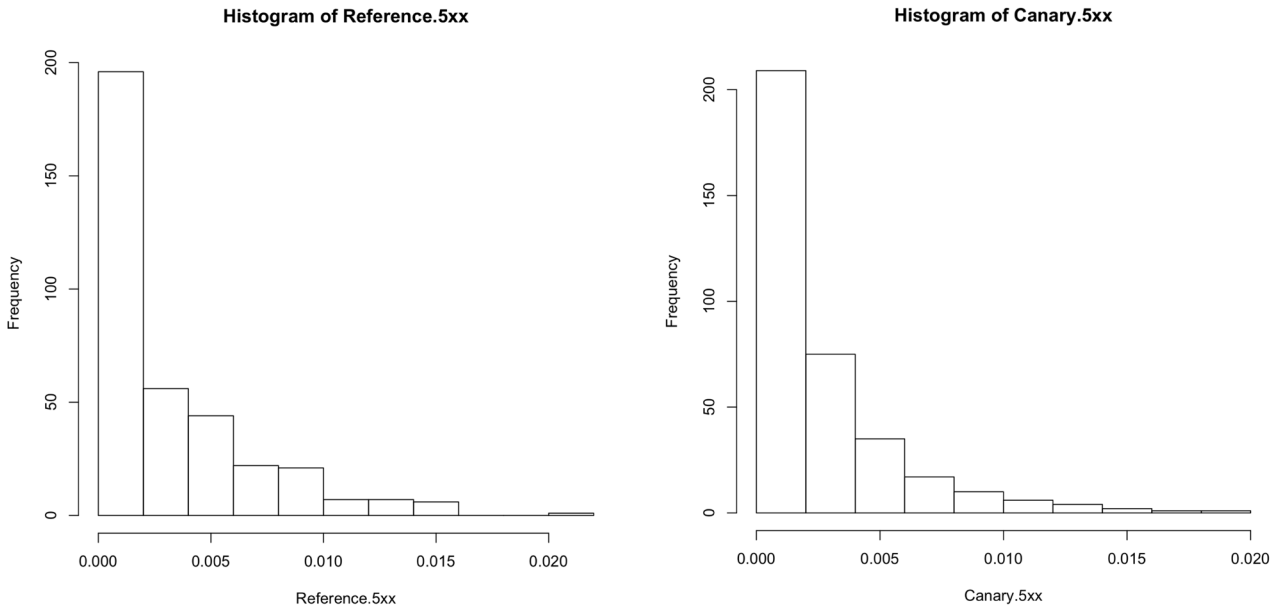

Figure 3 – Histograms Showing Distribution Of Data For All Hosts

Analyzing the data that we have here helps us notice the following:

- Canary has a higher concentration of samples at the lowest percentile of the data set.
- The reference host seems to have a set of outliers at the highest percentile.
- The canary seems to have a better distribution of the data as it progressively drops as values go up, whereas the reference host's data seems grouped across two or more bins.

2.5 ANOVA – Analysis of Variation

Now that we have looked at the meaningful percentiles and other metrics, we can start working with means and ANOVA is a great statistical test tool to start with.

According to Wikipedia[1], ANOVA "is a collection of statistical models used to analyze the differences among group means and their associated procedures (such as "variation" among and between groups), developed by statistician and evolutionary biologist Ronald Fisher"

If we run ANOVA on the data that has been plotted in the two graphs, we get the following result:

[1] https://en.wikipedia.org/wiki/Analysis_of_variance

```
             Df   Sum Sq   Mean Sq F value Pr(>F)
ind           1 0.000073 7.304e-05   6.032 0.0143 *
Residuals   718 0.008695 1.211e-05
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> TukeyHSD(anova_results)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = values ~ ind, data = stacked_errors)

$ind
                                  diff          lwr         upr     p adj
Reference.5xx-Canary.5xx 0.0006370231 0.0001277988 0.001146247 0.0142855
```

Figure 4 - ANOVA Test Output

While the sheer amount of data from this output might seem overwhelming at first, there are some key pieces which we must focus on and determine which other numbers we should look at, if any. The obvious first metric to look at is the p-value, which indicates the probability of a value of F greater than or equal to the observed value. The null hypothesis is rejected if this probability is less than or equal to the significance level. In this case our p-value is well within the confidence interval (95%), so we now turn our attention to the F-stat, which has a value of a bit over six. Ideally this metric should be as close as possible to one, but while it isn't, it still isn't far enough for us to reject the null hypothesis of no significant difference in variance between the two data sets.

2.6 Welch's T-Test

Continuing with our analysis of means, we'll use this test for its ability to remove any guessing that we could still have on the comparison of the data sets.

According to Wikipedia[2], "in statistics, Welch's t-test, or unequal variances t-test, is a two-sample location test which is used to test the hypothesis that two populations have equal means."

The null hypothesis here is that there is no significant difference in the means of the two data sets. As we have already calculated in section 2.2.1, we have noticed a lower mean in the canary data set when compared to the reference host. So let's see what a statistical test that checks difference in means says about that.

```
        Welch Two Sample t-test

data:  values by ind
t = -2.456, df = 697.29, p-value = 0.01429
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.0011462730 -0.0001277733
sample estimates:
   mean in group Canary.5xx mean in group Reference.5xx
                0.002359227                 0.002996250
```

Figure 5 - T-Test Output

Just like in the ANOVA test looked over in the previous section, there is a great deal of data that the test outputs and that can be overwhelming to look at first. But like in that case, there are key pieces of data we look at first to determine if the rest of the data set matters or not. The first piece of data we look at is the p-value, which here is well within our 95% confidence interval. That

[2] https://en.wikipedia.org/wiki/Welch%27s_t-test

indicates the t-stat is significant. The t-stat is -2.456, which is a good result since we want that number to be around two. Like in the ANOVA test, we don't want a number that is too large here, so the null hypothesis is not rejected which means, according to this test, the true difference in means is equal to zero.

3 What We Have Learned

3.1 What Percentiles to Calculate

Picking the percentiles was pretty straightforward and didn't really yield a lot of learning opportunities. The biggest takeaway here is that we should err on the side of caution and should calculate as many percentiles as possible. It's also important to strike a balance and not generate too much data that overwhelms us. The most meaningful percentiles when it comes to decision-making are the higher percentiles because they usually are the most volatile ones. It's best to always calculate the 90^{th}, 95^{th}, 99^{th} and max (or 100^{th}) percentiles so we can more clearly and easily assess the volatility of the data in question.

More importantly, for cloud systems that need to have over 99% reliability, we need to make sure that we're always looking at the higher percentiles to make sure we're not regressing this very important metric. Mean, median and lower percentiles may all be the same, or even better, but that doesn't necessarily imply that the 99^{th} and above percentiles are also better. **Takeaway: never make assumptions without generating the actual numbers**.

3.2 How Much Data Is Needed

This is the question one struggles with the most. In a recent project we started collecting and comparing 12 hours of data, but that proved to be too much. Figuring out the ideal data set size was a long and laborious process. We started at 12 hours of data and went down in one-hour increments, calculating the metrics and running the tests for each of them. In the end it became clear that a data set of at least three hours had enough data for us to run the tests. We also noticed that we wanted to have less than six hours of total data, so there is such a thing about having too much data. Because the number of samples over time will vary on logging solutions and policies, in terms of numbers of samples, we want to have around 160 samples.

Samples	t-stat	p-value
360 (12hr)	3.8951	0.0001078
260 (9hr)	3.5781	0.0003806
160 (6hr)	2.9022	0.003972
100 (3.1hr)	2.9415	0.003676
90 (3hr)	3.2008	0.001642
60 (2hr)	2.1255	0.03568
50 (1.6hr)	1.4877	0.1401

Table 1 - Samples, Time, And Test Results

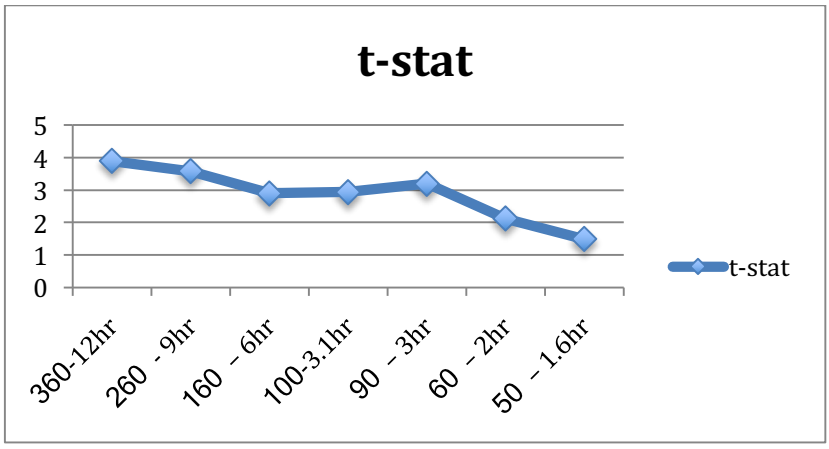

Figure 6 - Plot Of T-Stats x Samples - Time

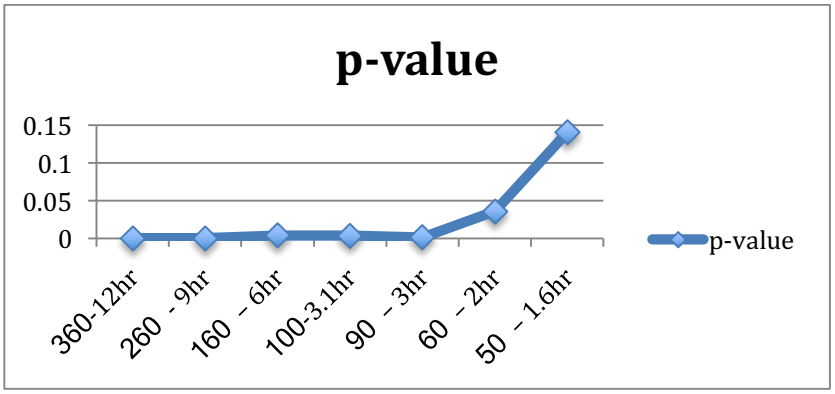

Figure 7 - Plot Of P-Values x Samples - Time

In the table and graphs above we can see t-stat values increasing with the number of samples and p-values dropping instead. It's easy to see that once we have too many samples, we end up with high confidence intervals, and as a result we have difficulty rejecting data unless it's extremely different. That is not ideal for high availability systems where the slightest shifts can make a big impact.

3.3 What Time Frame Is Best To Collect Your Data

In the previous exercise we discovered that more is less, but we didn't determine what time intervals are best. Is data taken from 2:00 AM to 5:00 AM ideal? Or is data from 9:00 AM to 12:00 PM better? The real answer is: *it depends*. The data should come from the time interval that has the most traffic. Why? Because you want to analyze the data under the most demanding conditions as those are more likely to yield the most valuable data. We should already know that testing under ideal conditions usually yields ideal results, often failing to expose issues uncovered by less than ideal conditions. If we take data when things are quiet and not really stressing out the service much, then it's less likely to expose problems simply because there isn't enough traffic to expose them in the first place.

The sampling of the data is up to you. Anything that is less than a metric sample for every 5 minutes is not recommended. Also, too many samples may be too much. Anything in the one sample every one to two minutes is appropriate for the purposes covered in this paper.

4 Automating This Stuff

There are many tools available that will do this work, spanning a different variety of budgets. This list covers some tools that have been successfully used.

4.1 Excel

The basic metrics come built-in and the rest can be done via many available statistics add-on packages, including some free versions. Unfortunately this support is not uniform across all operating systems. The Mac version of Excel has less statistics packages available and less free versions as well.

4.2 R

This is a popular open source statistics tools package. It's very powerful and covers all of the metrics and tests covered in this paper. This tool is supported across multiple operating systems with very similar functionality and best of all, it's completely free!

This was the tool used to generate all the statistical data present in this paper.

4.3 Python

Python has some very powerful libraries capable of calculating all of the metrics listed here and many more. Because it is a programming language with plenty of other libraries and capabilities, it then becomes the best option for automation.

Here are some libraries that we have successfully used for calculation of data and execution of tests listed in this paper:

- Pandas
- Numpy
- SciPy

5 Conclusion

In my most recent project, these tools and methods covered in this paper have been used for daily decision making for over six months and there is absolutely no turning back. Ever since adoption of the methods described in this paper, no release has gone out that significantly regressed a tested metric and no releases have had to be rolled back either.

Ongoing work involves increasing the number of metrics being analyzed and automation of the procedure so this can become an integral part of a continuous testing, integration and delivery platform instead of a side tool that simply helps make a decision to pull the plug on a new build or not. More importantly, this has provided the data used to push back when a release needs to go out, however, causing more harm than good. The tools described here very decisively remove any guess-work and speculation from any discussion about the impact of a metric regressing, even ever so slightly.

These powerful tools have helped maintain and increase quality with minimal overhead, while giving its users maximum confidence in their decisions. The experience shows that through the use of easily and readily available tools, great impact can be brought to the quality of the releases! This in turn results in greater customer satisfaction and increased revenue lift!