Emily Robinson
Data Scientist

Follow

Guidelines For Ab Testing

When I was working at Etsy, I benefited from a very robust A/B testing system. Etsy had been doing A/B testing for more than 6 years. By the time I left, Etsy's in-house experimentation system, called Catapult, had more than 5 data engineers working on it full-time. Every morning, I was greeted with a homepage that listed all the experiments that Etsy had run in the prior four years. When you clicked on one, you got a summary of what the experiment was testing (usually written by the product manager). Numbers for all the key metrics, such as conversion rate and add to cart rate, were already calculated. You could easily add any event that happened on the site and get those rates calculated too. You got to see the % change from the control to treatment with its accompanying p-value and how many days until we had 80% power to detect a 1% change. We even had beautiful little confidence intervals that changed color based on whether they overlapped with zero!

And yet sometimes I would spend a majority of my time working on experiments, even though I tried to never be working on more than 3 or 4 at once. How could that take so long when so much was already done for me? The concept of A/B Testing seems pretty simple. A classic example is you change the color of a button and measuring if the click-rate changes. Assuming your assignment of visitors and data collection is working, all you need to do is run a proportion test, right? And if you already have the proportion test calculated, why is a data scientist even needed? Maybe you need one if you want to do some fancy techniques like multi-armed bandits, but how can classic, frequentist A/B Testing be a challenge?

Unfortunately, "generating numbers is easy; generating numbers you should trust is hard!" There are many ways A/B Testing can go wrong, but most of them won't be obvious.

This post outlines some recommended best practices for A/B Testing. I've found that a lot of analysts and data scientists struggle with A/B testing, especially those not classically trained in statistics or who are trying to start their company's A/B testing system. While A/B testing correctly isn't easy, these 12 guidelines will help you guard against some common mistakes and set you up for success.

12 Guidelines for A/B Testing

1) **Have one key metric for your experiment**. You can (and should!) monitor multiple metrics to make sure you don't accidentally tank them, but you should have one as a goal. Revenue is probably the **wrong** metric to pick. It is likely a very skewed distribution which makes traditional statistics

tests behave poorly. See my discussion in my A/B testing talk (around the 23-minute mark). I generally recommend proportion metrics. First, you often you care more about the number of people doing something than how much they do it. Second, you don't have to deal with outliers and changing standard deviations.

Why only one metric? Once you start testing many metrics, you end up with an increased false positive rate. While correction methods can counteract this, they make each test more conservative. That means you become less likely to detect a difference in any given test. One metric also makes decision-making clearer. What would you do if you have three "equally important" metrics, and two go up a small amount while the other goes down a substantial amount?

2) **Use that key metric do a power calculation**. A common mistake in A/B testing is to run a test with such small traffic or base rate you'd need there to be a huge increase to be able to detect it within a week. To avoid this, run a power calculation first to determine how long it would take to detect an X% increase. You'll need the current rate (if a proportion metric) or mean and standard deviation of your key metric, how many visitors you get daily, what type of change you're aiming to get (1% increase? 5%?), the percentage of people you'll be allocating to your new version (e.g. are you doing 50/50 or 75/25 split), desired level of power (usually 80%), and the significance threshold (usually 95% or 90%). If you're doing a proportion metric, experimentcalculator.com is good for this.

Two things will generally happen: 1) you'll find that it will take a few days or weeks or 2) you'll find that it will take 3 years, 5 months, and 23 days. If the latter happens, you may either have to go for a different metric with a higher baseline rate or decide you only care about bigger changes. For example, you can decide that it's okay that you can't detect a 5% increase clicks because only a 10% or greater increase is meaningful. If you want to learn more about power, check out Julia Silge's excellent introductory post. She even created a shiny app so you can calculate how your power level changes with your effect size and population.

3) **Run your experiment for the length you've planned on**. You should monitor it in the first few days to make sure nothing exploded, but plan on running it for the length you planned on in your power calculation. Don't stop as soon as something is significant or you will get a lot of false positives. See the review section in Dave Robinson's Bayesian A/B Testing blog post. Don't be another p-hacking statistic:

> *When p-hacking is taken into account, 73% of A/B tests on Optimizely have no effect https://t.co/YNsx6IGBfk pic.twitter.com/xeLnOal3ba*
>
> *— Josh Kalla (@j_kalla) July 18, 2018*

4) **Pay more attention to confidence intervals than p-values**. They have a 1-1 relationship such that if the p-value is less than .05, the 95% confidence interval does not overlap with 0. But if the confidence interval is wide and very close to zero, you've got a lot less evidence of a change than if it's tiny and far away.

5) **Don't run tons of variants**. Say you want to redesign your homepage and your designers come up with six possibilities. How do you pick one? Well, that's what A/B Testing is for, right? Wrong. You will lower your ability to detect a statistical effect, as each group will have fewer people in it. You'll also raise the likelihood of a false positive if you simply test the control against each treatment group. As a rule of thumb, stick to only a treatment and control most of the time and don't go more than four total groups (control and three variations).

6) **Don't try to look for differences for every possible segment.** If your test doesn't work overall, it can be tempting to hold out hope that it actually did, just not for everyone. Or even if your A/B tests did succeed, you may want to know if it was driven by a big change in one segment. Did we help US visitors? New visitors? Visitors on Saturday? Down that road lies the madness of false positives from multiple testing, also known as detecting differences in health based on astrological signs. If you really think there will be a difference, either pre-specify your hypothesis or run separate tests (e.g. one for new visitors and one for returning).

7) **Check that there's not bucketing skew**. Bucketing skew, also known as sample ratio mismatch, is where the split of people between your variants does not match what you planned. For example, maybe you wanted to split people between the control and treatment 50/50 but after a few days, you find 40% are in the treatment and 60% in the control. That's a problem! If you have lots of users, even observing 49.9% in the control and 50.1% in the treatment can indicate a problem with your set-up. To check if you have an issue, run a proportion test with the number of visitors in each group and check if your p-value is less than .05. If you do have bucketing skew, you have a bug. Unfortunately, it can be difficult to find it, but a good place to start is checking is if the skew differs based on web browser, country, or another visitor factor. Also check if your treatment is significantly slower; it may be that users with slow connections are dropping out before they get bucketed into the treatment. Finding the bug and rerunning the test is very important because generally users aren't going missing at random. If you're systematically dropping people who use internet explorer in the treatment, who also never buy your product, your conversion rate will look artificially better because the population in the control vs. treatment is different.

8) **Don't overcomplicate your methods**. Maybe you have engineers who've read about multi-armed bandit testing, stats nerds who want to use Bayesian methods, or product managers who want the key metric to be a complicated sequence of behaviors. If you're just starting out A/B testing methods, focus on getting the basic, frequentist methods right. Even after a few years, it's usually better to invest in experiment design and education rather than fancy statistical methods.

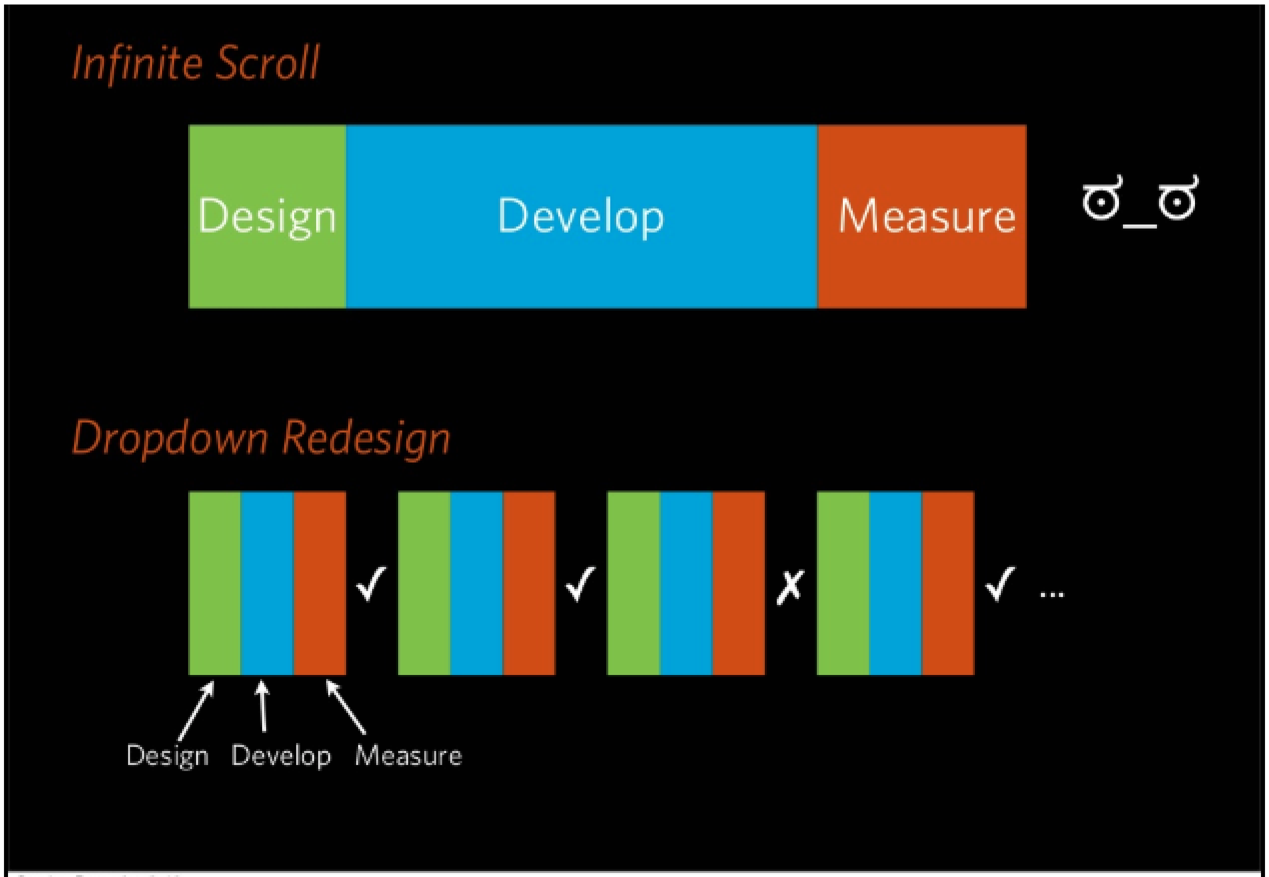
9) **Be careful of launching things because they "don't hurt"**. There may actually be a negative change that's too small to detect but could have a meaningful effect in the long-term. When deciding whether to launch on "neutral," the first step is to look at your non-key metrics. If other metrics you care about have been impacted negatively, you'll probably want to rollback. If not, this is where your product intuition and other data can come in. Is this a change users have been asking for? Does it set the foundation for future changes you want to make? In general, default to rolling it back. This is also where your power analysis comes in - whatever increase you had 80% power to detect, would you be okay if you launch and you actually had decrease of that same size? The smaller changes you were set to detect, the less risky launching on neutral is. You could also look into non-inferiority testing, which is designed to test that your treatment is not worse than the control by a pre-specified amount. While I haven't used it before, this looks like a good resource.

10) **Have a data scientist/analyst involved in the whole process.** As Sir R. A. Fisher once said, "to consult the statistician after an experiment is finished is often merely to ask [them] to conduct a post mortem examination. [They] can perhaps say what the experiment died of." If a team tries to bring in a data scientist after they launched an experiment, they may find the data doesn't exist to measure their key metric, the test is severely underpowered, or there's a design flaw that means they can't draw any conclusions.

11) **Only include people in your analysis who could have been affected by the change**. If you have users in your experiment whose experience could not have been impacted by your change, you're adding noise and reducing your ability to detect an effect. For example, if you're changing the layout of the search page, only add users to the experiment if they visit the search page. In a more complicated example (from this great paper on triggering), let's say you want to experiment with changing the threshold for a free shipping offer (displayed only when they meet the criteria) from $35 to $25. You should only put users in the experiment who have cart sizes between $25 to $35 because those are the only people who would see something different in the treatment vs. control group. Relatedly, start tracking your metrics **after** the user sees the relevant page. Imagine you're running an experiment on the search page, and someone visits your sites, buys something from the homepage, and **then** visits the search page, entering the experiment. You don't want to count their earlier conversion, as it could not have been a result of your change.

12) **Focus on smaller, incremental tests that change one thing at a time**. It's very tempting to launch big changes or a bundle of smaller changes in the hope that they result in big wins. But the problem is that you will often invest tons of effort up front only to find out your change doesn't work. And when it doesn't, it's hard to figure out why - was it just one part that failed? Or was it an interaction of the changes? A better practice is to split them up into smaller tests.

Dan McKinley, former principal engineer at Etsy, gives a great example of this problem in his presentation on continuous experimentation. His team spent weeks working on enabling infinite scroll for the search page. But when they ran the A/B test, they found it performed worse! Their first reaction was that it must be a bug, but while they did find some, the results remained unchanged. So they went back and tested the assumptions behind why they believed infinite scroll would be better. First, are more items actually better? When they changed just the number of items on the search page, they found there were more clicks, but the same number of purchases. Second, were faster results better? Nope, artificially slowing down the search page didn't hurt anything. If they'd checked those first, they would not have invested in infinite scroll. They learned from this and changed to making a series of smaller design-develop-measure (with A/B tests) cycles culminating up to a big change.



Sunday, December 2, 12

Contrasting the two release plans, infinite scroll was a big bet that didn't work out.
The dropdown redesign was a series of small bets: some worked and some didn't, but we didn't have to throw out everything when things didn't work

Next Time

While I learned some of these guidelines in my statistics classes or experience at Etsy and DataCamp, others I picked up from the great set of A/B Testing resources available online. In a future post, I'll share a list of some of my favorite papers, blog posts, and talks, with short summaries of what I took away and suggested audience level.

Updated: August 07, 2018