**Good Data Analysis**

Author: Patrick Riley

**Technical**

**Look at your distributions**

Most practitioners use summary metrics (for example, mean, median, standard deviation, and so on) to communicate about distributions. However, you should usually examine much richer distribution representations by generating histograms, cumulative distribution functions (CDFs), Quantile-Quantile (Q-Q) plots, and so on. These richer representations allow you to detect important features of the data, such as multimodal behavior or a significant class of outliers.

**Consider the outliers**

Examine outliers carefully because they can be canaries in the coal mine that indicate more fundamental problems with your analysis. It's fine to exclude outliers from your data or to lump them together into an "unusual" category, but you should make sure that you know why data ended up in that category.

For example, looking at the queries with the lowest number of clicks may reveal clicks on elements that you are failing to count. Looking at queries with the highest number of clicks may reveal clicks you should not be counting. On the other hand, there may be some outliers you will never be able to explain, so you need to be careful in how much time you devote to this task.

**Consider noise**

Randomness exists and will fool us. Some people think, “Google has so much data; the noise goes away.” This simply isn’t true. Every number or summary of data that you produce should have an accompanying notion of your confidence in this estimate (through measures such as confidence intervals and *p-values*).

**Look at examples**

Anytime you are producing new analysis code, you need to look at examples from the underlying data and how your code is interpreting those examples. It’s nearly impossible to produce working code of any complexity without performing this step. Your analysis is abstracting away many details from the underlying data to produce useful summaries. By looking at the full complexity of individual examples, you can gain confidence that your summarization is reasonable.

How you sample these examples is important:

* If you are classifying the underlying data, look at examples belonging to each class.
* If it's a bigger class, look at more samples.
* If you are computing a number (for example, page load time), make sure that you look at extreme examples (fastest and slowest 5% perhaps; you do know what your distribution looks like, right?) as well as points throughout the space of measurements.

**Slice your data**

Slicing means separating your data into subgroups and looking at metric values for each subgroup separately. We commonly slice along dimensions like browser, locale, domain, device type, and so on. If the underlying phenomenon is likely to work differently across subgroups, you must slice the data to confirm whether that is indeed the case. Even if you do not expect slicing to produce different results, looking at a few slices for internal consistency gives you greater confidence that you are measuring the right thing. In some cases, a particular slice may have bad data, a broken user interaction, or in some way be fundamentally different.

Anytime you slice data to compare two groups (such as experiment vs. control, or even “time A” vs. “time B” ), you need to be aware of mix shifts. A *mix shift* is when the amount of data in the slices for each group is different. [Simpson's paradox](http://wikipedia.org/wiki/Simpson's_paradox) and other confusions can result. Generally, if the relative amount of data in a slice is the same across your two groups, you can safely make a comparison.

**Consider practical significance**

With a large volume of data, it can be tempting to focus solely on statistical significance or to home in on the details of every bit of data. But you need to ask yourself, "Even if it is true that value X is 0.1% more than value Y, does it matter?" This can be especially important if you are unable to understand/categorize part of your data. If you are unable to make sense of some user-agent strings in your logs, whether it represents 0.1% or 10% of the data makes a big difference in how much you should investigate those cases.

Alternatively, you sometimes have a small volume of data. Many changes will not look statistically significant, but that is different than claiming these changes are “neutral.” You must ask yourself, “How likely is it that there is still a *practically significant* change?”

**Check for consistency over time**

You should almost always try slicing data by units of time because many disturbances to underlying data happen as our systems evolve over time. (We often use days, but other units of time may also be useful.) During the initial launch of a feature or new data collection, practitioners often carefully check that everything is working as expected. However, many breakages or unexpected behavior can arise over time.

Just because a particular day or set of days is an outlier does not mean you should discard the corresponding data. Use the data as a hook to determine a causal reason why that day or days is different before you discard it.

Looking at day-over-day data also gives you a sense of the variation in the data that would eventually lead to confidence intervals or claims of statistical significance. This should not generally replace rigorous confidence-interval calculation, but often with large changes you can see they will be statistically significant just from the day-over-day graphs.

**Acknowledge and count your filtering**

Almost every large data analysis starts by filtering data in various stages. Maybe you want to consider only US users, or web searches, or searches with ads. Whatever the case, you must:

* Acknowledge and clearly specify what filtering you are doing.
* Count the amount of data being filtered at each step.

Often the best way to do the latter is to compute *all* your metrics, even for the population you are excluding. You can then look at that data to answer questions like, "What fraction of queries did spam filtering remove?" (Depending on why you are filtering, that type of analysis may not always be possible.)

**Ratios should have clear numerator and denominators**

Most interesting metrics are ratios of underlying measures. Oftentimes, interesting filtering or other data choices are hidden in the precise definitions of the numerator and denominator. For example, which of the following does “Queries / User” actually mean?

* Queries / Users with a Query
* Queries / Users who visited Google today
* Queries / Users with an active account (yes, I would have to define *active*)

Being really clear here can avoid confusion for yourself and others.

Another special case is metrics that can be computed only on some of your data. For example "Time to Click" typically means "Time to Click given that there was a click." Any time you are looking at a metric like this, you need to acknowledge that filtering and look for a shift in filtering between groups you are comparing.

**Process**

**Separate Validation, Description, and Evaluation**

I think of data analysis as having three interrelated stages:

1. **Validation**[1](https://developers.google.com/machine-learning/guides/good-data-analysis#fn1)**:** Do I believe the data is self-consistent, that it was collected correctly, and that it represents what I think it does?
2. **Description:** What's the objective interpretation of this data? For example, "Users make fewer queries classified as X," "In the experiment group, the time between X and Y is 1% larger," and "Fewer users go to the next page of results."
3. **Evaluation:** Given the description, does the data tell us that something good is happening for the user, for Google, or for the world?

By separating these stages, you can more easily reach agreement with others. Description should be things that everyone can agree on for the data. Evaluation is likely to spur much more debate. If you do not separate Description and Evaluation, you are much more likely to only see the interpretation of the data that you are hoping to see. Further, Evaluation tends to be much harder because establishing the normative value of a metric, typically through rigorous comparisons with other features and metrics, takes significant investment.

These stages do not progress linearly. As you explore the data, you may jump back and forth between the stages, but at any time you should be clear what stage you are in.

**Confirm experiment and data collection setup**

Before looking at any data, make sure you understand the context in which the data was collected. If the data comes from an experiment, look at the configuration of the experiment. If it's from new client instrumentation, make sure you have at least a rough understanding of how the data is collected. You may spot unusual/bad configurations or population restrictions (such as valid data only for Chrome). Anything notable here may help you build and verify theories later. Some things to consider:

* If the experiment is running, try it out yourself. If you can't, at least look through screenshots/descriptions of behavior.
* Check whether there was anything unusual about the time range the experiment ran over (holidays, big launches, etc.).
* Determine which user populations were subjected to the experiment.

**Check for what shouldn't change**

As part of the "Validation" stage, before actually answering the question you are interested in (for example, "Did adding a picture of a face increase or decrease clicks?"), rule out any other variability in the data that might affect the experiment. For example:

* Did the number of users change?
* Did the right number of affected queries show up in all my subgroups?
* Did error rates change?

These questions are sensible both for experiment/control comparisons and when examining trends over time.

**Standard first, custom second**

When looking at new features and new data, it's particularly tempting to jump right into the metrics that are new or special for this new feature. However, you should always look at standard metrics first, even if you expect them to change. For example, when adding a new universal block to the page, make sure you understand the impact on standard metrics like “clicks on web results” before diving into the custom metrics about this new result.

Standard metrics are much better validated and more likely to be correct than custom metrics. If your custom metrics don’t make sense with your standard metrics, your custom metrics are likely wrong.

**Measure twice, or more**

Especially if you are trying to capture a new phenomenon, try to measure the same underlying thing in multiple ways. Then, determine whether these multiple measurements are consistent. By using multiple measurements, you can identify bugs in measurement or logging code, unexpected features of the underlying data, or filtering steps that are important. It’s even better if you can use different data sources for the measurements.

**Check for reproducibility**

Both slicing and consistency over time are particular examples of checking for reproducibility. If a phenomenon is important and meaningful, you should see it across different user populations and time. But verifying reproducibility means more than performing these two checks. If you are building models of the data, you want those models to be stable across small perturbations in the underlying data. Using different time ranges or random sub-samples of your data will also tell you how reliable/reproducible this model is.

If a model is not reproducible, you are probably not capturing something fundamental about the underlying process that produced the data.

**Check for consistency with past measurements**

Often you will be calculating a metric that is similar to things that have been counted in the past. You should compare your metrics to metrics reported in the past, even if these measurements are on different user populations.

For example, if you are looking at query traffic on a special population and you measure that the mean page load time is 5 seconds, but past analyses on all users gave a mean page load time of 2 seconds, then you need to investigate. Your number may be right for this population, but now you have to do more work to validate this.

You do not need to get exact agreement, but you should be in the same ballpark. If you are not, assume that you are wrong until you can fully convince yourself. Most surprising data will turn out to be an error, not a fabulous new insight.

**New metrics should be applied to old data/features first**

If you create new metrics (possibly by gathering a novel data source) and try to learn something new, you won’t know if your new metric is right. With new metrics, you should first apply them to a known feature or data. For example, if you have a new metric for user satisfaction, you should make sure it tells you your best features help satisfaction. If you have a new metric for where users are directing their attention to the page, make sure it matches to what we know from looking at eye-tracking or rater studies about how images affect page attention. Doing this provides validation when you then go to learn something new.

**Make hypotheses and look for evidence**

Typically, data analysis for a complex problem is iterative.[2](https://developers.google.com/machine-learning/guides/good-data-analysis#fn2) You will discover anomalies, trends, or other features of the data. Naturally, you will develop theories to explain this data. Don’t just develop a theory and proclaim it to be true. Look for evidence (inside or outside the data) to confirm/deny this theory. For example:

* If you see something that looks like a learning trend, see if it manifests most strongly with high frequency users.
* If you believe an anomaly is due to the launch of some features, make sure that the population the feature launched to is the only one affected by the anomaly. Alternatively, make sure that the magnitude of the change is consistent with the expectations of the launch.
* If you see growth rates of users change in a locale, try to find an external source that validates that user-population change rate.

Good data analysis will have a story to tell. To make sure it’s the right story, you need to tell the story to yourself, then look for evidence that it’s wrong. One way of doing this is to ask yourself, “What experiments would I run that would validate/invalidate the story I am telling?” Even if you don’t/can’t do these experiments, it may give you ideas on how to validate with the data that you do have.

The good news is that these theories and possible experiments may lead to new lines of inquiry that transcend trying to learn about any particular feature or data. You then enter the realm of understanding not just this data, but deriving new metrics and techniques for all kinds of future analyses.

**Exploratory analysis benefits from end-to-end iteration**

When doing exploratory analysis, perform as many iterations of the whole analysis as possible. Typically you will have multiple steps of signal gathering, processing, modeling, etc. If you spend too long getting the very first stage of your initial signals perfect, you are missing out on opportunities to do more iterations in the same amount of time. Further, when you finally look at your data at the end, you may make discoveries that change your direction. Therefore, your initial focus should not be on perfection but on getting something reasonable all the way through. Leave notes for yourself and acknowledge things like filtering steps and unparseable or unusual requests, but don't waste time trying to get rid of them all at the beginning of exploratory analysis.

**Watch out for feedback**

We typically define various metrics around user success. For example, did users click on a result? If you then feed that data back to the system (which we actually do in a number of places), you create lots of opportunities for evaluation confusion.

You can not use the metric that is fed back to your system as a basis for evaluating your change. If you show more ads that get more clicks, you can not use “more clicks” as a basis for deciding that users are happier, even though “more clicks” often means “happier.” Further, you should not even do slicing on the variables that you fed back and manipulated, as that will result in mix shifts that will be difficult or impossible to understand.

**Mindset**

**Data analysis starts with questions, not data or a technique**

There’s always a motivation to analyze data. Formulating your needs as questions or hypotheses helps ensure that you are gathering the data you should be gathering and that you are thinking about the possible gaps in the data. Of course, the questions you ask should evolve as you look at the data. However, analysis without a question will end up aimless.

Avoid the trap of finding some favorite technique and then only finding the parts of problems that this technique works on. Again, creating clear questions will help you avoid this trap.

**Be both skeptic and champion**

As you work with data, you must become both the champion of the insights you are gaining and a skeptic of them. You will hopefully find some interesting phenomena in the data you look at. When you detect an interesting phenomenon, ask yourself the following questions:

* What other data could I gather to show how awesome this is?
* What could I find that would invalidate this?”

Especially in cases where you are doing analysis for someone who really wants a particular answer (for example, "My feature is awesome!"), you must play the skeptic to avoid making errors.

**Correlation != Causation**

When making theories about data, we often want to assert that "X causes Y"—for example, "the page getting slower caused users to click less." [Even xkcd knows](http://xkcd.com/552/) that you can not simply establish causation because of correlation. By considering how you would validate a theory of causation, you can usually develop a good sense of how credible a causal theory is.

Sometimes, people try to hold on to a correlation as meaningful by asserting that even if there is no causal relationship between A and B, there must be something underlying the coincidence so that one signal can be a good indicator or proxy for the other. This area is dangerous for multiple hypothesis testing problems; as [xkcd also knows](http://xkcd.com/882/), given enough experiments and enough dimensions, some of the signals will align for a specific experiment. This does not imply that the same signals will align in the future, so you have the same obligation to consider a causal theory such as “there is a hidden effect C that causes both A and B” so that you can try to validate how plausible this is.

A data analyst must often navigate these causal questions for the people that want to consume the data. You should be clear with those consumers what you can and can not say about causality.

**Share with peers first, external consumers second**

The previous points suggested some ways to get yourself to do the right kinds of soundness checking and validation. But sharing with a peer is one of the best ways to force yourself to do all these things. A skilled peer can provide qualitatively different feedback than the consumers of your data can, especially since consumers generally have an agenda. Peers are useful at multiple points through the analysis. Early on you can find out about gotchas your peer knows about, suggestions for things to measure, and past research in this area. Near the end, peers are very good at pointing out oddities, inconsistencies, or other confusions.

Ideally, you should get feedback from a peer who knows something about the data you are looking at, but even a peer with just general data-analysis experience is extremely valuable.

**Expect and accept ignorance and mistakes**

There are many limits to what we can learn from data. Nate Silver makes a strong case in [*The Signal and the Noise*](http://www.amazon.com/dp/159420411X) that only by admitting the limits of our certainty can we make advances in better prediction. Admitting ignorance is a strength not usually immediately rewarded. It feels bad at the time, but it’s a great benefit to you and your team in the long term. It feels even worse when you make a mistake and discover it later (or even too late!), but proactively owning up to your mistakes earns you respect. That respect translates into credibility and impact.