Data-Driven Metric Development for Online Controlled Experiments: Seven Lessons Learned

Alex Deng *
Microsoft
One Microsoft Way
Redmond, WA 98052
alexdeng@microsoft.com

Xiaolin Shi*†
Yahoo Research
701 1st Ave
Sunnyvale, CA 94089
shixiaolin@gmail.com

ABSTRACT

Online controlled experiments, also called A/B testing, have been established as the mantra for data-driven decision making in many web-facing companies. In recent years, there are emerging research works focusing on building the platform and scaling it up [34], best practices and lessons learned to obtain trustworthy results [19; 20; 23; 26], and experiment design techniques and various issues related to statistical inference and testing [6; 7; 8]. However, despite playing a central role in online controlled experiments, there is little published work on treating metric development itself as a data-driven process. In this paper, we focus on the topic of how to develop meaningful and useful metrics for online services in their online experiments, and show how data-driven techniques and criteria can be applied in metric development process. In particular, we emphasize two fundamental qualities for the goal metrics (or Overall Evaluation Criteria) of any online service: directionality and sensitivity. We share lessons on why these two qualities are critical, how to measure these two qualities of metrics of interest, how to develop metrics with clear directionality and high sensitivity by using approaches based on user behavior models and data-driven calibration, and how to choose the right goal metrics for the entire online services.

Keywords

Controlled experiment; A/B testing; web measurement; online metrics; user experience evaluation

1. INTRODUCTION

As the quote by Lord Kelvin stated, "If you cannot measure it, you cannot improve it", it is also true for online services in industry. For various online services such as search engines, e-commerce sites, and social networking sites, metrics are used to indicate how well the systems are serving real users for their needs. The impact of any feature change of the service can be quantitatively measured by the metrics. Depending on how the scores of metrics are generated, there are two types of metrics for most online services: one type is offline metrics, whose scores are based on labels of offline data sets (usually generated by a small group of judges); and the other type is online metrics, whose scores are usually based on the behavioral data log of millions of users who interact with the system in real time. In industry, the most popular and widely accepted way of monitoring and comparing online metrics is using online controlled experiments (i.e. A/B testing experiments) [5; 22; 34]. Figure 1 shows the relationship between the services and online metrics: based on the data logged by the services, online metrics are able to measure how well the services are doing; and based on the metrics, the online services are able to know the direction of making further improvement.

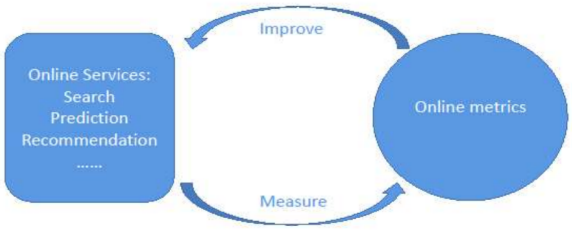

Figure 1: The relationship between online services and online metrics.

Nowadays, many online service companies are using online metrics as the pointers toward the North Star (i.e. the success of the business) to improve their products, including Amazon [21], eBay, Etsy [28], Facebook [3], Google [34], Groupon, Intuit [29], LinkedIn [30], Microsoft [27], Netflix [2], and Yahoo. The *Overall Evaluation Criteria (OEC)* (also known as *goal metrics* or *key metrics*) of an online service are metrics defined to help the system move toward the North Star. The choice of good OEC is not easy. As an online service grows over time, its OEC should also evolve, and the choice of its OEC depends on how far the system is away from the North Star (as shown in Figure 2).

Many previous works have been focusing on the experimentation platform [22; 25; 34], various best practices and lessons learned [20; 24; 26] and various issues related to the statistical inference and testing [3; 7; 8; 13]. Little work has been published on the topic of data driven metric development for online controlled experiments. The work of [31] proposes the metric framework HEART, which first points

*Authors contributed equally to this work.
†Work primarily done when working at Microsoft.

KDD '16, August 13-17, 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2939672.2939700

out that user-centric metrics are more direct in measuring user experience of using the product than metrics focusing on business or technical aspects of the product. Our work further demonstrates how to develop metrics that well align with user experience in a data-driven approach, and how to choose the right metrics for online services. To the best of our knowledge, this paper is the first work on how to develop online metrics systematically in a data-driven way.

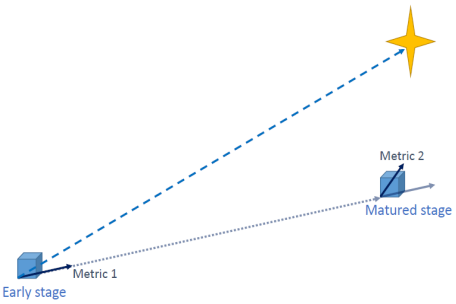

Figure 2: The choice of OEC evolves as the service grows over time toward the North Star. When the service (the cubic) is at an early stage, metric 1 is used as the OEC. Another OEC (metric 2) is used when the service is closer to the North Star.

2. PRELIMINARIES

2.1 Offline metrics vs. online metrics

Before the large-scale online experiments are available, the most popular way to measure the performance of online systems in industry is using offline metrics such as area under curve (AUC), root mean squared error (RMSE), and normalized discounted cumulative gain (NDCG), etc. The evaluation approach of using offline metrics is borrowed from academia, where the sizes of the data sets are much smaller and whose targets are not the real users. However, in industry, when the online services are deployed, the ultimate performance of interest is the success of the business, and the real users are the core of the business. Offline metrics certainly have great values in guiding the services when they are at relatively early stages and not many users are available for experimentation yet. For example, Bing used the offline metric NDCG for a long time at the early stage of the search engine, and NDCG had been playing a critical role in guiding the improvement of the search system for several years. However, at a certain point, it was found that NDCG could not give us the directions with high confidence any more, and very often it did not align with what was shown by online metrics either. In fact, it has been realized that in many scenarios, offline metrics have poor correlations with online metrics. According to [4], it could be due to two reasons. First of all, offline metrics such as AUC and NDCG ignore human factors, and thus cannot capture the qualities of services that are designed for personalization. Another reason is offline metrics are based on incomplete and biased offline data sets, and thus offline metrics cannot always predict the results of online metrics that are based on large-scale real users. It is still an open question in this area on how to correlate offilne metrics with online metrics.

2.2 Types of online metrics

Because of the special role of metrics in online services, most work in this community is done from the business perspective. For example, the AARRR model [1] defines metrics according to their different roles in conversion and monetization in a web service.

However, because of the diversity of online services, the link between user status and monetization is not always as straightforward as the AARRR model describes. Moreover, these standard web metrics may be too generic to be used on online applications with different service purposes. In this paper, we categorize online metrics into the following three types based on how they are defined:

Type 1: Business report driven metrics. This type of metrics are defined based on the long-term goals of the online services, and thus they are usually directly related to the success of the business, such as Revenue per User, Queries or Visits per User. Recently there is growing interest in reporting active user count metrics such as Monthly Active Users (MAU) and Daily Active Users (DAU).

This type of metrics are mainly used as reports to business owners, executives and investors or shareholders as they are directly related to the success of the online services. However, this type of metrics are often not actionable because the short-term movements of these metrics measured in online experiments are often different from the long-term effects. As a result, product teams whose purpose of using online metrics is to chase short-term signals can hardly use these metrics to improve their features efficiently. For example, long-term revenue per user is crucial for the success of a search engine service such as Bing; however, if revenue per user is used as the goal metric, the product teams may find it very hard to tune their ranking algorithms toward the improvement of this goal in short terms (such as a couple of weeks). This is because improved search results may immediately increase the clicks on the search results and decrease the probability of clicking on the ads results, and thus decrease revenue per user in short term. The same reversed direction happens to market share metrics. For a search engine, bad ranking algorithms often result in short-term increases in query volumes [32], since users tend to issue more reformulation queries to satisfy their information needs. Thus, using business report driven metrics to measure the long-term impact usually requires running experiments for long periods of time, which is not desired for agile online experiments whose purpose is to show the impact of new features with limited time and cost.

Type 2: Simple heuristic based metrics. This type of metrics are usually based on the simple facts of the interaction between users and the online services, such as Click-Through Rate (CTR) of a web page, performance delay, count of activities per user.

This type of metrics give us straightforward descriptions on how users interacting with the services. They are mostly actionable and also important for us to monitor the system from different perspectives. For example, the count of activities per user within a certain period of time shows how frequently an average user is active on the service, while performance delay indicates how soon the system responds to users' requests. However, this type of metrics usually lack clear interpretations regarding user experience and are usually ambiguous in the relationship with the North Star. For example, having more spams and misleading captions with the URLs in online service may increase its click through

[1] http://piratemetrics.com/

rate. However, this is actually hurting user experience, and will hurt the revenue and market share of the service eventually. For another example, a longer page load time often is considered bad. However if the delay is caused by a new feature addition that greatly generates user satisfaction, it should be interpreted as a reasonable trade-off.

Business report driven metrics and simple heuristic based metrics represent the first generation of online metrics. They are mostly simple, obvious and align with our common sense. Usually at the early stage of an online service, when it does not have enough users and the improvements are easy to observe, these two types of metrics may be good enough. However, when the service accumulates enough users and the market share is relatively stable, these metrics are either insensitive to small feature improvements (especially for the revenue based metrics), or mis-align with real improvement of user experience (especially for the simple heuristic based metrics). At such stages, a third type of online metrics, user-behavior-driven metrics, are in great need.

Type 3: User-behavior-driven metrics. This type of metrics are based on user behavior models such as satisfaction models and frustration models. Ideally, a well designed user-behavior-driven metric should be able to measure user experience directly, and these experiences are highly correlated with the long-term success of the online service. At the same time, it should be sensitive enough and actionable so that it can be used in agile online experiments with limited time and cost. Metric definitions of this type are often more involved, relying on highly customized rules or algorithms trained from user behavioral models of using the particular online services, and thus more complexity in the design process. For example, we need a user satisfaction model to define whether a search result page satisfies a user. Clicking signals and dwell time signals are used but there are also cases where user can be satisfied by just looking at the page (i.e. good abandonment) [9; 33]. Thus, it is essential to analyze user search behavior instead of simply using clicks as the only heuristic in the metrics of user search satisfaction. More details about the process and rules of designing user-behavior-driven metrics will be described in Section 3.5.

3. SEVEN LESSONS OF ONLINE METRIC DEVELOPMENT

3.1 Lesson 1: Define metrics for metrics

As we have mentioned at the beginning, metrics are used to evaluate the online services quantitatively. However, when we develop online metrics, what are the criteria we can use to evaluate the metrics quantitatively? Especially, how can we make comparison and choose one or a couple of metrics as OEC or the goal metrics for the online service from hundreds or even thousands of metrics we are monitoring? Generally speaking, there are two mandatory qualities we have to take into consideration when choosing the OEC or goal metrics for an online service: directionality and sensitivity. The directionality and sensitivity of a metric are like the direction and magnitude of a vector, as both are critical in pushing the system improving toward the North Star effectively (Figure 2).

Mandatory quality 1 – directionality: a good goal metric (or OEC) should have a clear directional interpretation that aligns with user experience in most common cases. In another word, when the metrics are used in A/B experiments, the goal metric should show consistent direction when the experiments positively impact user experience, and show the opposite direction for the experiments that negatively impact user experience. It is not uncommon to see metrics whose directions are ambiguous in indicating user experience. One example is the metric Distinct Queries per Unique User (DSQ/User) in search. Intuitively, users issue more distinct queries if they are more satisfied with the search results and this is why some search engines gain more market share than others. However, user behavior analysis shows that, in a short term after relevance improved, users tend to issue less queries within a search task while the total number of search tasks remains almost the same [32]. Thus the metric DSQ/User decreases almost immediately after the relevance gets improved. It is important to make sure that the goal metrics should not have such ambiguous directional interpretation and have consistent directions for both short term and long term. A common mistake in picking a goal metric is to beat the control in short term by doing something clearly "wrong" from a business perspective (such as increasing DSQ/User by degrading the relevance of a search engine), as it is quite often that short-term and long-term objectives diverge diametrically [24].

Mandatory quality 2 – sensitivity: a good goal metric (or OEC) needs to be sensitive to most of the improvement of user experience. A sensitive goal metric helps the organization and feature teams being able to make decisions quickly and with limited cost. As we have mentioned before, report or revenue driven metrics usually require long experimentation time and high user traffic to observe the impact of new features. For example, the metric Sessions per User (Sessions/User) is a metric believed to be closely correlated with the market share of a search engine as well as user satisfaction. However, deep-dive analysis shows that instead of improving the total number of sessions, which requires much longer time to change, the search success rate improves more immediately and significantly when search relevance changes [32]. In another word, metrics based on successful sessions are much more sensitive than the metric Sessions/User, and thus we prefer metrics such as Successful Sessions per User or Successful Session Rate as the goal metric for an online service.

In order to measure the two qualities of metrics qualitatively and convince the entire organization to adopt a new metric, a relatively objective and systematical approach for metric evaluation is critical. At Bing, there is an evaluation framework called MetricLab. More details will be discussed in Section 3.3.

In addition to evaluating a new metric by quantitatively reporting its direction and sensitivity based on a validation corpus, it is also very important to have a thorough understanding on the applicability of the metrics. Generally speaking, no metric can be applicable everywhere, even the metrics with the most state-of-the-art models. Thus, it is important to fully evaluate the pros and cons of a new metric, including its *directionality*, *sensitivity* and *applicability*.

3.2 Lesson 2: Understand roles of metrics

In the previous subsections, we have introduced the goal metrics (or OEC) of an online service and the two mandatory qualities of good OEC. We also have pointed out that no metric is applicable in all scenarios and thus in order to have a more objective and complete understanding of the change of user experience with the service, in addition to the goal

metrics, it is also very important to have the following two types of metrics: *guardrail metrics* and *debugging metrics*.

Guardrail metrics are metrics that help to guard against situations when the goal metrics may give us wrong signals. They are usually used in two scenarios. The first scenario is to replace the goal metric (or OEC) in cases when the goal metric is not applicable. In this scenario, we should not read too much into the goal metrics, and use the guardrail metrics as substitutes. For example, the entire organization of a search engine service may use the metrics with long dwell time clicks as the goal metrics, as users having more long dwell time clicks in web search usually shows that the search engine is doing a better job. However, this rule is not applicable to the engineering team that works on the instant answers (e.g. weather, stock, dictionary), whose goal is to provide users with the right information without having clicks. Thus, instead of using metrics with long dwell time clicks as the goal metrics, these teams need some guardrail metrics that can better address the user experience in using instant answers. The second scenario of guardrail metrics is to capture the dimensions of user experience that the goal metrics are not able to measure. In this scenario, usually the goal metrics are still the keys to look at, and the guardrail metrics help to make sure that some other important dimensions are not going in the wrong way. For example, metrics such as Revenue per User and performance latency are good candidates for guardrail metrics when the metric of user satisfaction rate is used as the goal metric. These guardrail metrics can make sure that the service is not improving the user experience by sacrificing revenue, nor anything unexpected happening on the server side. Having a clear directional interpretation is very important for a good guardrail metric in both scenarios.

Debugging metrics are usually used to help us understand why some important metrics, especially the goal metrics, move or do not move, and how to interpret their movements. For goal metrics whose models are based on multiple user behavior signals, it is helpful to keep track of how these individual signals contributed to the overall metric movement (and thus, it is desirable for a goal metric to be debuggable or decomposable, which we will discuss more in Section 3.5). Debugging metrics can be as simple as making sure certain assumptions we make are correct. For example, http response size or log size typically should change when new features are added for the treatment. Debugging metrics are especially important for rate metrics. If we use a rate metric such as Query Success Rate for a search engine, it is important to keep track of the change of the plain count of the denominator, which is the total number of queries per user in this case. We will discuss more regard this in Section 3.6. Instead of having an clear directional interpretation, it is more important for a good debugging metric to be sensitive. This is a distinct difference between debugging metrics and guardrail metrics.

In summary, it is always important for us to fully understand the characteristics of each individual metric, including when it is applicable and when it is not, and its direction and sensitivity in different scenarios, so that we are able to assign it to the right role.

3.3 Lesson 3: Evaluate metric qualities

In Lesson 1 we show two mandatory qualities that we need to measure for each metric under development: sensitivity and directionality. If a metric rarely shows movement with statistical significance (i.e. is insensitive), it is not actionable in practice no matter how good the metric is in all other aspects. On the other hand, if a metric frequently gives us statistically significant signals, but we have little confidence on how to relate the movement to the success of the business, then the metric is disqualified to act as a goal metric. The next question is how to measure these two qualities of metrics systematically and convincingly. According to our experience, there are two effective approaches.

3.3.1 Validation Corpus

Motivated by supervised learning, we could have a validation dataset with known labels. In online controlled experiments, this translates to a set of existing experiments for which we know whether the feature is indeed good for the users (in terms of average treatment effect[2]) or not. In reality, we can never be 100% sure about the goodness of the features, so the best we can do is to collect a set of high quality past experiments for which we have high confidence on whether their features are good for the users or not. This set of past experiments is called the *validation corpus*.

Such a validation corpus is extremely valuable as it provides straightforward way for us to systematically test a new metric and see whether the movement of the tested metric agrees with the "labels" in the corpus. In Bing we have developed a tool called MetricLab to routinely measure new and old metrics. There are three cases: (1) the metric shows statistically significant movement with a consistent direction (either positively or negatively correlates with the "labels"), (2) the metric shows statistically significant movement with ambiguous directions and (3) the metric is not statistically significant. Sensitivity can be measured by proportion of cases in (3), or by computing average or medium of t-statistics. If the ratio of average t-statistics between two metrics X and Y are r, then we can say movements of Y *on average* requires r times of traffic than those of X to be detected. More details about developing and using the validation corpus can be found in [10].

3.3.2 Degradation experiment

A problem with the validation corpus described above is that this approach is not applicable when we adventure into a new experimentation scenario. For instance, when we onboard a new partner with a new product, we need to design and implement a set of first generation of metrics to start with. Obviously there exists no validation corpus. People commonly resort to simple business reporting metrics and believe the directions of these metrics are obvious to interpret. However as we mentioned in Section 3.1, even a standard reporting metric as simple as DSQ/User might be misleading[3]. Hence it is important to verify the directions of these metrics before we make any decision based on them. Fortunately there is a way to validate the directions without having the validation corpus. This solution lies in a lesson we learned over years. Most of the time it is hard to come up with a good feature that impacts user experience with statistically detectable improvement in a short period, but it is much easier for us to degrade or even screw up user

[2]Note that there might be heterogeneous treatment effect among users, i.e., some subpopulations of users might like the feature while others dislike. For feature ship decision making, here we only focus on average treatment effect.

[3]Distinct query volume is part of ComScore search share report.

experience deliberately. In another words, it is hard to find positive labels but there are plenty of cases with negative labels. This leads us to the idea of using degradation experiments to validate metric direction and sensitivity[4]. The type of user experience degradation experiments depends on the type of web services. For search engine and online websites, typical degradation experiments include delaying a web page [25], downgrading to a known inferior service [32] to more extreme choices such as deliberately giving user a buggy experience such as crashing a mobile app.

People often feel reluctant to run degradation experiments because deliberately hurting user experience is against the organization goal. We argue that if we control the degradation within an acceptable degree, we only hurt short term user experience without making permanent damage, and the knowledge gained from running such experiment outweighs the short term loss in the long run. In Bing we've experimented delaying return of search result page by 250ms and didn't observe any long term user effect [25]. The long running search relevance degradation experiment [32] showed a short term left over effect on user search behavior but no sign of losing user base. Both experiments successfully helped us validate the directions of many metrics, and also cast lights on the development and adoption of new goal metrics.

3.4 Lesson 4: Decompose metric sensitivity

When there is a good metric candidate that is not sensitive enough to be used as an actionable goal metric, we need to better understand its sensitivity before giving it up. There are two components of metric sensitivity:

1. Statistical Power: Given a true effect moving the metric, how likely are we able to detect the effect?
2. Movement Probability: How often does a change we test really moves the metric (has a treatment effect)?

The separation of the two components above helps us to isolate two sources impacting metric sensitivity. On one hand, statistical power accounts for the probability that we can detect the movement *given that there is a true underlying movement (treatment effect) on a metric*. On the other hand, the movement probability shows *how likely a metric is indeed moved or not*. The two components together help us to compare sensitivity of different metrics. For example, we know Sessions per User is a relatively insensitive metric. Surprisingly, its insensitiveness is not mainly due to its lacking statistical power: its noise level, measured by the coefficient of variations (Standard Deviation divided by Mean), is not among the highest. Instead, the reason we cannot detect its movement is because people's daily search needs are limited and it is hard to change users engagement within a short time. Google [16] reported similar results that even for long running experiments it is hard to move sessions per user. In contrast, metrics like Revenue per User is relatively less sensitive than other revenue focus metrics such as Revenue per Search due to its high variance (highly skewed distribution) and low statistical power. If statistical power is the reason of insensitiveness, we then can focus on *variance reduction*. Increasing traffic size is an obvious option since the variance of a metric decreases at the rate of $\sqrt{N}$ where N is the number of independent samples. Methods such as capping the metric to reduce skewness and other more sophisticated transformation methods may also apply.

[4]We first heard this idea in a discussion with Ron Kohavi and Brian Frasca.

To measure the two components, we first need to lay some foundations. Suppose for both treatment and control groups we observe i.i.d. observations from two distributions with unknown mean τ_T and τ_C respectively. Denote our observations by $Y_i, i = 1, \ldots, N_T$ for treatment group and $X_i, i = 1, \ldots, N_C$ for control group. We want to test the null hypothesis $H_0 : \tau_T - \tau_C = 0$ against the alternative $H_1 : \tau_T \neq \tau_C$.

Without assuming distributions of X and Y, in A/B testing we resort to the central limit theorem and hence use Wald test which can be seen as large sample version of the well-known t-test. The test statistic is

$$\mathrm{Z} := \frac{\overline{X} - \overline{Y}}{\sqrt{\sigma_T^2/N_T + \sigma_C^2/N_C}} = \frac{\Delta}{\sqrt{\sigma_T^2/N_T + \sigma_C^2/N_C}},$$

where σ_C and σ_T are variances of X and Y and Δ is the observed metric difference between treatment and control. The variances are unknown but in large sample scenario we can assume they are known by using their estimates which will not change the asymptotic normal approximation. Note that metrics are often in different scales and to evaluate different metrics we need to unify the scale. We first define $N_E = 1/(1/N_T + 1/N_C)$ to be the **effective sample size**. Let σ^2 be the **pooled variance** such that $\sigma^2/N_E = \sigma_T^2/N_T + \sigma_C^2/N_C$. The two definitions above are used to convert a two-sample problem into an equivalent one-sample problem, where we can define **effect size** to be $\delta = \Delta/\sigma$. With these shorthands, Z-statistics can be rewritten as

$$\mathrm{Z} = \frac{\delta}{\sqrt{1/N_E}}. \tag{1}$$

Note that the effect size δ is the observed difference Δ scaled by pooled standard deviation σ and thus is scaleless. We also treat σ as a fixed constant. Finally, define

$$\mu := E(\delta) = E(\Delta)/\sigma = (\tau_T - \tau_C)/\sigma \tag{2}$$

is the average treatment effect scaled by σ. When σ is treated as known, inference on $\tau_T - \tau_C$ and μ are equivalent and μ is the scaleless treatment effect.

We adopt a model where we assume there are two states of the truth. When there is no treatment effect $\mu = 0$ and this is the null hypothesis H_0. The alternative hypothesis H_1 is the state where there is non-zero treatment effect μ. Unlike fixing an alternative treatment effect as in most textbook power analysis, we also assume under H_1 μ follows certain distribution g. This is because the true treatment effect in general is unknown.

Recall the Z-statistics is the ratio of δ to $\sqrt{1/N_E}$. We typically declare a metric moved statistically significantly if $|Z| > 1.96$[5]. We see the probability of the event that *a treatment has an effect and we can detect successfully* is

$$P(H_1) \times P(|Z| > 1.96|H_1) \tag{3}$$

where the first term $P(H_1)$ is the probability of a true movement and the second term $P(|Z| > 1.96|H_1)$ is the statistical power. These two terms corresponds to the two components of sensitivity introduced at the beginning.

To calculate the statistical power, one need to know the distribution of δ under alternative H_1. We found it useful

[5]1.96 and the corresponding 0.05 p-value cutoff is arbitrary and we can replace 1.96 by other thresholds in (3)

to assume $\mu \sim N(0, V^2)$ under H_1. The parameter V is unknown and need to be measured and we claim that V is a good quantity to measure the statistical power component of metric sensitivity. The larger the V, the more disperse the treatment effect is under H_1, and hence more easily we can detect the movement. Note that under H_1, $\delta \sim N(0, (1/N_E + V^2))$. Let S be a standard normal random variable,

$$P(|Z| > 1.96|H_1) = P\left(|S| > 1.96 \times \sqrt{\frac{1/N_E}{1/N_E + V^2}}\right) \quad (4)$$

For any given effective sample size N_E, the larger the V, the larger the above probability. We make remark that unlike traditional power analysis concerning sample size needed in an experiment for a given presumed effect size, here we are not interested in the sample size. Instead, we are interested in for any given sample size, how we can compare expected statistical power of different metrics. The result above identified V to be a good metric of metrics for this component.

The component $P(H_1)$ and V together allows us to systematically order and compare sensitivity of metrics. To estimate both quantities for each metric, we can use the same validation corpus mentioned above. However such corpus tend to be small due to the cost of expert labeling. For the sake of decomposing sensitivity, label is not required and we can use a vast unlabeled historical experiments data. The technical detail is in [6]. Note when comparing two metrics, a metric A might have higher $P(H_1)$ but smaller V than metric B. To combine the two sensitivity metrics into one, we fix an effective sample size N_E and use (4) to calculate the expected statistical power for the given sample size and then multiply it with $P(H_1)$ to get (3). The effective sample size is chosen to reflect a typical experiment, e.g. the sample size we typically allocate for an experiment.

Metric	P(True Movement)	$V^2/(1/N_E)$	P(Detect True Movement)
Count Metric A(whole page)	3%	25.1	2.1%
Count Metric A(sub region)	17%	37.2	12.8%
Metric B	31%	2.8	9.8%
Metric B with Utility Weights	45%	3.3	15.6%
Metric C	12%	9.2	5.9%
Metric C with VR	11%	19.7	8.0%
Metric D	13%	29.3	9.4%
Metric D Capped	16%	38.6	12.1%

Table 1: Examples of detailed sensitivity decomposition. Results align with theory and experience.

Table 1 shows results of a few example metrics when we apply the method to Bing historical experiments. We found the results consistent with theory and our experiences and we listed a few here.

1. It is easier to move a metric measuring part of a page than a whole page metric [26]. The sub region version of Metric A is way more sensitive than the whole page version. It has significantly higher $P(H_1)$, which means more experiments can move sub region metric.
2. Carefully designed utility weighting helps improving sensitivity. Metric B is a session level metric and after giving different weights to different sessions according to a custom utility function, we found the revised version significantly improved sensitivity.
3. Variance Reduction (VR) increase sensitivity by reducing variance σ^2 and hence increasing V. The work by Deng et al. [7] proposes a variance reduction framework usng pre-experiment data. In the above table comparing Metric C with and without VR we found V^2 doubled after VR, aligning well with the theory.
4. Capping of highly skewed metrics helps improving sensitivity by reducing variance. Comparing Metric D with and without capping, we also found a significant increase of V^2. See [26] for more details about capping.

We make a remark that there is a naive but relatively biased approach to measure sensitivity using unlabeled historical experiment data. Given a list of historical experiment, just count the proportion of experiments a metric moved statistically significantly. On surface it seems this will give the same measure we tried to calculate in (3). But imagine if a metric never truly moved, i.e. $P(H_1) = 0$, then because of Type-I error, this naive approach will in expectation gives 5%. In other word, the naive approach is biased, with the bias being $P(H_0) \times 5\% \leq 5\%$. Since the bias is bounded, the naive method can still be used to quickly compare sensitivity of different metrics when the more sophisticated measurement (3) is not available. For example, if metric A using the naive approach has a sensitivity score more than 5% higher than another metric B, we can say metric A is more sensitive than metric B, ignoring the estimation error of the two sensitivity measurements, which is often small when the set of unlabeled data is large. The two parameter approach presented here has the advantage that it separates two sources of sensitivity apart. When lack of sensitivity of a metric is due to lack of statistical power, we can then investigate where we can improve the statistical power by ways of reducing variance.

3.5 Lesson 5: Learn from offline data

As we have mentioned in Section 2.2, a more advanced type of metrics – user-behavior-driven metrics – are getting more importance when the systems are at matured stages. User-behavior-driven metrics are based on user behavioral models that can predict or classify user experience of using the system, such as satisfaction, frustration, gain or cost [9; 11; 15; 17]. Thus, a carefully designed user-behavior-driven metric usually has good alignment with user experience and high sensitivity, i.e. the two qualities of good OEC we have described in Section 3.1. However, because users seldom give explicit feedback regarding their experience at the same time of using the online services, we propose the approach of utilizing offline labeled data to associate user behavioral signals (i.e. activity logs collected when they use the services) to their experience in developing this type of metrics.

Generally speaking, there are three major steps in developing user-behavior-driven metrics:

- **Step 1**: Have some hypotheses about the user experience based on preliminary observations. Start with the simplest model.
- **Step 2**: Design user study experiments and collect labeled data sets. Test the model from Step 1 on the collected labeled data sets.
- **Step 3**: Design online metrics based on the model learned from Step 2. Validate the metrics based on the validation corpus and try out the metrics on new A/B testing experiments as much as possible.

Iterate these three steps till satisfying results achieved based on the validation corpus described in Section 3.3. Unlike building user behavioral models for other applications such as prediction or recommendation, during the process of building models for online metrics, there are some principles that we need to follow.

First of all, the hypotheses we start in Step 1 should capture important *collective behavioral patterns* of most users. By "collective behavioral patterns", we mean the behavioral patterns that can be observed from most users. For example, for the majority of users of using a search engine, they spend longer time in reading more relevant documents [11], and issue more reformulation queries when they are not satisfied with the results of previous queries [15]. It is absolutely possible that there are some users whose clicking or reformulation behavior is different from the majority; however, we do not have to worry too much about the behavioral patterns of the minority of the users unless there are some special cases we want to address (such as the feature of instant answer in search). In other words, the behavioral models for online metrics need to capture users' homogeneous behavior instead of their heterogeneous behavior.

Secondly, when we collect labeled offline data and learn models from them in Step 2, we should be extra cautious and keep the following several rules in mind:

Rule 1: If possible, collect the labeled data from multiple different ways. This is because for many inevitable reasons, such as different user populations and data distributions, any user or judge labeled data is biased comparing with the real production data. Thus, the phenomena that can be observed from data sets collected in different ways are more likely to be true for the real production data as well. There are two most common ways to collect labeled data. One is collecting first-hand labels by taking surveys from the users or by doing lab study. The studies in [1; 11; 12] use this approach. The other is using third-party judges to label the logged data by re-presenting the information and behavioral traces to the trained crowd workers, such as the work in [14; 17; 36]. Both approaches have their own pros and cons. For example, using the lab study generates more accurate labels as they are labeled by the users themselves, but this approach can hardly simulate the real scenarios of using the services; on the other hand, employing third-party crowd workers to annotate logged search sessions is inexpensive and can better capture the real usage scenarios, the limitation of this method is being less accurate than real users' own ratings. In the work of [15], multiple labeled data sets collected by both approaches are used to verify the relationship between query reformulation and search success, which makes the conclusion more convincing and reliable, and hence Bing employs this feature in defining the search success model for its OEC.

Rule 2: Avoid making the models as black-boxes. In Section 3.2 we have mentioned that it is important to have debugging metrics for OEC; on the other hand, it is very desirable for OEC to be "debuggable" or "decomposable". By being "debuggable", we mean it is easy and straightforward to define debugging metrics to track and explain the movements of a metric. In order to get a "debuggable" metric, the behavioral model that the metric is based on should be decomposable and intuitive. For example, we learn that both clicking behavior and reformulation behavior are highly correlated with user's search satisfaction [15], a decision tree classifier as Figure 3 is much better than a machine learned classifier when being used as the model for metrics. This is because a decision tree model as Figure 3 is very intuitive and easy to be decomposed into several branches: queries followed by reformulations (unsuccessful queries of branch 1), queries not followed by reformulations and without long-dwell-time clicks (unsuccessful queries of branch 2), and queries not followed by reformulations and with long-dwell-time clicks (successful queries of branch 3). If we use Successful Queries per User as the goal metric, then it is fairly easy to understand the movement of this metric by using Queries per User, Reformulation Queries (i.e. queries followed by reformulations) per User and Queries with Long-dwell-time Clicks per User as the debugging metrics.

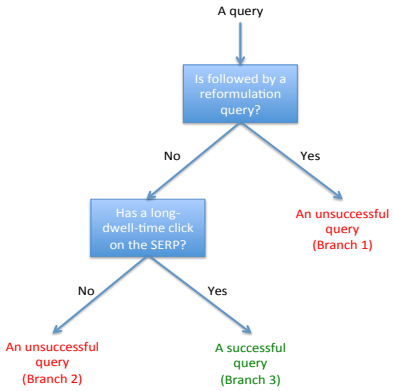

Figure 3: An intuitive and decomposable decision tree model combining two behavioral signals, reformulation and long-dwell-time clicks, to decide if a query is successful or not.

In practice, based on the offline labeled data, we can first have some machine-learned models, and then we simplify and hand-tune the machine-learned models into clear, human-readable models for online metrics. It is acceptable if the human-readable simplified models have reasonably lower accuracies than the machine-learned models when evaluated by the offline labeled data. This is because the goodness of a model for metrics is eventually evaluated by the quality described in Section 3.1 instead of fitting the offline labeled data.

Rule 3: When choose features for the models, make sure that only exogenous features can be used. Exogenous features are features from the user side, such as users' clicking behavior, browsing patterns or dwell time. In contrast, endogenous features are those from the service side, such as the information shown on the webpages. If the endogenous features are used in the models for metrics, then the product teams have the chance of gaming the metrics either intentionally or unintentionally, and thus the metrics lose the required quality of impartiality. For example, it has been shown that displaying good instant answers on the search result pages has high probability in making the users end their search tasks without clicks but with satisfaction [33]. However, if we use an endogenous feature such as "an instant answer of weather being displayed" as a feature in the model shown in Figure 4, then if the feature team has a powerful algorithm in predicting when the user would not have any click, they can manipulate the metric by displaying a weather answer to improve the probably of classifying a query into branch 2 even the query has nothing to do with weather. Although this kind of manipulation is hard in practice, we have seen some "accidental or unintentional" manipulation of our metrics when some endogenous features are employed in the models in reality.

Finally, even with the most state-of-the-art models, it is still halfway to a good metric. Rigorously validating

the behavior-driven metrics based on the validation corpus and extensively testing them on new controlled experiments is the must step. Moreover, with the same user behavior model, we can define multiple metrics. For example, with the successful query model shown in Figure 3, we can define metrics such as Successful Queries per User, Successful Query Rate, Successful Sessions per User and Successful Session Rate. When we want to choose the right one for OEC, in addition to the qualities discussed in Section 3.1, there are a few important factors we need to take into consideration. First of all, usually metrics with bounded values are preferred, as metrics without upper bounds are much more sensitive to outliers than those having bounded values [19]. Thus, comparing with the count metrics such as Sessions per User and Queries per User, rate metrics such as Session Success Rate and Query Success Rate have the advantage of being bounded and less likely to be affected by outliers. Second, if we want to use rate metrics for OEC or goal metrics, there are some rules we need to follow, which will be discussed in the next subsection.

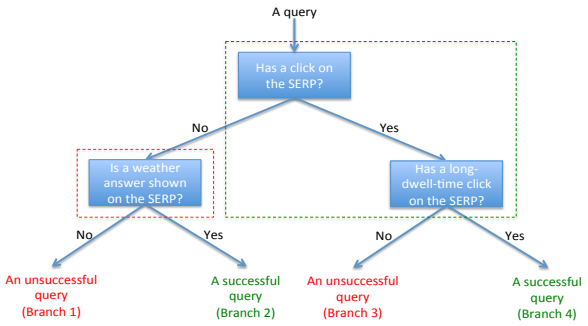

Figure 4: A decision tree model having both exogenous features (within the green dashed box) and an endogenous feature (within the red dashed box). The endogenous feature gives the feature team a chance to manipulate the metric.

3.6 Lesson 6: Choose the right rate metrics

A rate metric is a metric that has two parts in its definition: the numerator and the denominator (and the denominator not be the count of users), such as Click Through Rate, Views per Movie, or Success Query Rate (i.e. the count of success queries divided by the count of total queries). Rate metrics are widely used and in many cases they are good candidates for OEC, as rate metrics have good properties such as with bounded values, less skewed (e.g. the value of rate metric Query Success Rate is bounded between 0 and 1 and its distribution is less skewed than the count metric of Successful Queries per User) and relatively more sensitive. However, when we design and use rate metrics, there are some traps we need to be extremely careful about.

First of all, when a rate metric increases with statistical significance, it could be due to the following reasons: (a) the numerator increases and the denominator remains stable; (b) the denominator decreases and the numerator remains stable; (c) the numerator increases and the denominator decreases; (d) both the numerator and denominator increase, and (e) both the numerator and denominator decrease. Among these five possibilities, except for (a), the goodness of all other cases are ambiguous. Take the rate metric of Query Success Rate as an example, case (a) means that the total number of queries a user issues (i.e. the denominator) remains the same while the number of successful queries a user issues (i.e. the numerator) increases. This is clear and very likely a positive scenario. However, if the denominator (i.e. the total number of queries) is not stable, then it could be due to any reason from (b) to (e). As we have mentioned previously, the move direction of total queries per user is ambiguous, and thus it is hard to decide if any scenario from (b) to (e) is positive, negative or mixed without further analysis. Generally speaking, if the denominator of a rate metric changes between the control and treatment groups, then comparing the rate metric between the control and treatment groups makes as little sense as comparing apples and oranges. Thus, there are two rules we have to follow when dealing with rate metrics:

1. We should always keep the denominator and the numerator as debugging metrics (discussed in Section 3.2) for a rate metric;
2. When we choose a rate metric to be a goal metric, we should choose the one whose denominator is relatively more stable in most cases.

For example, when we want to decide the goal metric between Query Success Rate (QSR) and Session Success Rate (SSR) for Bing, we examined whose denominator is more stable in addition to look at their directions and sensitivities as mentioned in Section 3.1. Based on the observations from hundreds of A/B experiments, we find that the denominator of SSR, i.e. number of sessions per user, is much more stable than the denominator of QSR, i.e. number of queries per user. In fact, 10% of statistically significant QSR movement happened together with statistically significant Distinct Queries per User movement! In other word, interpreting QSR is tricky for 10% of cases. Thus, although QSR is more sensitive than SSR in many cases, SSR is more appropriate for being a goal metric. We use QSR as a debugging metric to help better understand the goal metric.

The second issue is that there are two ways to compute the metrics for rate metrics: Average of Ratios and Ratio of Averages. For instance, Click Through Rate (CTR) can be defined as A) Average CTR per User, and B) #(Clicks of all users)/#(Pageviews of all users). The former is average of ratios where for each individual user we calculate the ratio first and then take average across all users. The latter is equivalent to the ratio of Clicks/User to Pageviews/User, hence is a ratio of averages. Both definitions are used in practice, and sometimes both versions are computed for the same rate metric and therefore it is important to understand the differences. Their main difference is how users are weighted. For average of ratios, users have the same weight on the final metric because each user has a single value of ratio. For ratio of averages, mathematically it is equivalent to weighted average of ratio per user, where weight is proportional to the denominator value for each user, e.g. user with more pageviews will have a higher impact on CTR. Because of the difference in weighting, optimizing ratio of averages puts more emphasize on heavy users while optimizing average of ratios treats each user equally important.

In our experience, these two versions of rate metrics move in the same direction for most experiments. When they don't, we get an interesting signal that the treatment effect is different between heavy users and average users, which begs our further investigation. In general average of ratios tends to be more sensitive than ratio of averages when treatment effects are homogeneous for all users. This is because

for ratio of averages, the metric is dominated by those heavy users, and hence the *effective sample size* is the same as the number of heavy users. On the other hand, average of ratios is influenced equally by all the users and its *effective sample size* is the total number of users. The concept of effective sample size is better explained in the literature of weighted samples and importance sampling. As a rule of thumb, when using weights in sample average, effective sample size is smaller than the number of independent samples. The larger the skewness of weight distribution is, the smaller the effective sample size is. Also note that the variance for ratio of averages should be calculated using Delta Method [35] if users are used as the randomization units, while for average of ratio, we just need to use the standard sample variance formula.

3.7 Lesson 7: Use combo metrics as surrogate for OEC

In spite of all the techniques we have stated above, finding a good goal metric that has both clear direction and actionable sensitivity can still be very hard because no metric is applicable in all scenarios. In the absence of a single OEC to use as our guide toward the North Star, one approach is to use a set of success criteria metrics and by looking at these metrics altogether, we can have a better estimation on the experiment result. For this strategy to work, first, we need to have a fairly comprehensive understanding of all the success criteria metrics. For each metric, we need to know the intended direction interpretation and more importantly scenarios where the metric fail to give clear direction and therefore should be either discarded or replaced by another metric in the decision making process. Second, the set of individual metrics should cover all known scenarios. When one metric fail to give clear direction, there must be at least one other metric that can fill the gap. At last, the most ambiguous situation is when two metrics shows different directions, i.e. one shows user experience is improved and the other suggests degradation. In such cases experts' opinions are often required to carefully trade off two things: the signal strength and the signal value.

This procedure described above requires a panel experts to gather together regularly to review the experiment results together and thus is not scalable. For an entire organization to align, a more objective and standardized approach is still preferred. **Combo metrics** are designed to address this issue. It is a derived metric consists of the most important success criteria metrics used in experts' decision rule. Given a set of metrics $X_i, i = 1, \ldots, k$, a combo metric is of the form

$$f(\Delta\%(X_i), \ldots, \Delta\%(X_k))$$

where the $\Delta\%(X_i)$ is the Percent Delta of metric X_i defined as $\frac{X_i^T - X_i^C}{X_i^C} \times 100\%$ (X_i^T and X_i^C here are metric X_i's values for treatment and control). The functional form of f is what we need to design. It is generally recommended to make f differentiable and with continuous first derivative. This is because we need to perform statistical test on the combo metric and continuity makes the test of the combo metric as easy as each component. When f is not differentiable, the metric might not even have an asymptotically normal distribution which significantly complicates the analysis. We show the asymptotic normality of a combo metric when f is differentiable with continuous partial derivatives. Our derivation also gives the formula of the variance. When combo metric is asymptotically normal, Wald's Z-test described in Section 3.4 can be applied.

The key of the derivation is to use the delta method [35]. We can rewrite the combo metric as

$$f(X_1^T, \ldots, X_k^T, X_1^C, \ldots, X_k^C)$$

where X^T and X^C is the component metrics calculated in treatment and control groups respectively. The asymptotic distribution of the vector $X = (X_1^T, \ldots, X_k^T, X_1^C, \ldots, X_k^C)$ is multivariate normal with known covariance matrix from multivariate central limit theorem. Because treatment and control groups are two independent groups of subjects, the covariance matrix of this vector has the structure

$$\begin{pmatrix} \Sigma_T & 0 \\ 0 & \Sigma_C \end{pmatrix}$$

where Σ_T and Σ_C are covariance matrix of $(X_i^T, \ldots, X_k^T)$ and $(X_i^C, \ldots, X_k^C)$ respectively and both can be estimated using sample covariance from the experiment data. From the delta method, the variance of $f(X_i^T, X_i^C, \ldots, X_k^T, X_k^C)$ can be calculated as:

$$\left(\frac{\partial f}{\partial x}\bigg|_{x=X}\right)^T \times \begin{pmatrix} \Sigma_T & 0 \\ 0 & \Sigma_C \end{pmatrix} \times \frac{\partial f}{\partial x}\bigg|_{x=X}$$

If the function f does not have a continuous first derivative, then there is no guarantee that the combo metric has a asymptotic normal distribution. An example of such combo metric is any function that involves MAX or MIN, or conditional rules. For those irregular combo metrics, more computationally intensive method such as bootstrap [18] should be used instead.

The simplest type of combo metrics are linear combinations, i.e. combo metrics of the form

$$\sum w_i \Delta\%(X_i)$$

where the weight w_i has a utility based interpretation. If a validation corpus is available to accurately measure direction and sensitivity of a combo metric, then we can try different weights or functional forms and pick the one with the best performance. If such a validation corpus is not available, we can use degradation experiments as described in Section 3.3. For example, if X and Y are two success criteria metrics and from several degradation experiments the ratio of degradation between the two are r, then it means 1% change of Y is comparable to r% change of X. This gives us a good starting point of the weights.

4. SUMMARY

Nowadays, controlled experiments are used heavily by many data-driven Internet companies to understand the impact of the business on real users. In this paper, we present the process of designing online metrics for controlled experiments could also be data-driven by showing methodologies we have adopted and lessons we have learned over years. We define two mandatory qualities that goal metrics (i.e. OEC) have to meet, directionality and sensitivity, and show empirical ways to measure these qualities systematically and effectively. We also show how to design and improve metrics by utilizing user behavior modeling and statistical techniques, and how to choose the right and most appropriate metrics based on different characteristics they have. We hope the

insights and lessons we share in this paper could benefit the successful use and deployment of online metrics for controlled experiments in various online services in industry.

Acknowledgment

The authors want to thank Pavel Dmitriev and Xian Wu for their work on MetricLab, Shouyuan Chen for his related internship work in alternative sensitivity and alignment measurement framework not published here, and Patrick Pantel and Yi-Min Wang for discussions and support when one of the authors was working at Microsoft Research. We also thank anonymous reviewers who gave helpful feedback on the presentation and contents of this paper.

References

[1] Mikhail Ageev, Qi Guo, Dmitry Lagun, and Eugene Agichtein. Find it if you can: A game for modeling different types of web search success using interaction data. In *Proceedings of the 34th ACM SIGIR Conference*, pages 345–354, 2011.

[2] Xavier Amatriain and Justin Basilico. Netflix recommendations: beyond the 5 stars. *Netflix Tech Blog*, 6, 2012.

[3] Eytan Bakshy and Dean Eckles. Uncertainty in online experiments with dependent data: an evaluation of bootstrap methods. In *Proceedings of the 19th ACM SIGKDD Conference*, pages 1303–1311, 2013.

[4] Joeran Beel, Marcel Genzmehr, Stefan Langer, Andreas Nürnberger, and Bela Gipp. A comparative analysis of offline and online evaluations and discussion of research paper recommender system evaluation. In *Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation*, RepSys '13, 2013.

[5] Brian Christian. The a/b test: Inside the technology that's changing the rules of business, April 2012. URL http://www.wired.com/business/2012/04/ff_abtesting/.

[6] Alex Deng. Objective bayesian two sample hypothesis testing for online controlled experiments. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 923–928, 2015.

[7] Alex Deng, Ya Xu, Ron Kohavi, and Toby Walker. Improving the sensitivity of online controlled experiments by utilizing pre-experiment data. In *Proceedings of the 6th ACM WSDM Conference*, pages 123–132, 2013.

[8] Alex Deng, Tianxi Li, and Yu Guo. Statistical inference in two-stage online controlled experiments with treatment selection and validation. In *Proceedings of the 23rd international conference on World Wide Web*, pages 609–618, 2014.

[9] Abdigani Diriye, Ryen White, Georg Buscher, and Susan Dumais. Leaving so soon?: Understanding and predicting web search abandonment rationales. In *Proceedings of the 21st ACM CIKM*, pages 1025–1034, 2012.

[10] Pavel Dmitriev and Xian Wu. Measuring metrics, 2016. https://archive.org/details/MeasuringMetricsCIKM2016.

[11] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. Evaluating implicit measures to improve web search. *ACM Trans. Inf. Syst.*, 23(2):147–168, April 2005.

[12] Qi Guo and Eugene Agichtein. Ready to buy or just browsing?: Detecting web searcher goals from interaction data. In *Proceedings of the 33rd ACM SIGIR Conference*, pages 130–137, 2010.

[13] Yu Guo and Alex Deng. Flexible online repeated measures experiment. *arXiv preprint arXiv:1501.00450*, 2015.

[14] Ahmed Hassan, Rosie Jones, and Kristina Lisa Klinkner. Beyond dcg: User behavior as a predictor of a successful search. In *Proceedings of the Third ACM WSDM Conference*, pages 221–230, 2010.

[15] Ahmed Hassan, Xiaolin Shi, Nick Craswell, and Bill Ramsey. Beyond clicks: Query reformulation as a predictor of search satisfaction. In *Proceedings of the 22nd ACM CIKM Conference*, pages 2019–2028, 2013.

[16] Henning Hohnhold, Deirdre O'Brien, and Diane Tang. Focusing on the long-term: It's good for users and business. In *Proceedings of the 21th ACM SIGKDD Conference*, pages 1849–1858, 2015.

[17] Jiepu Jiang, Ahmed Hassan Awadallah, Xiaolin Shi, and Ryen W. White. Understanding and predicting graded search satisfaction. In *Proceedings of the 8th ACM WSDM Conference*, pages 57–66, 2015.

[18] Ariel Kleiner, Ameet Talwalkar, Purnamrita Sarkar, and Michael I Jordan. A scalable bootstrap for massive data. *J. R. Stat. Soc. Ser. B (Statistical Methodol.)*, 2014.

[19] R. Kohavi, R. Longbotham, and T. Walker. Online experiments: Practical lessons. *Computer*, 43(9):82–85, Sept 2010.

[20] Ron Kohavi and Roger Longbotham. Unexpected results in online controlled experiments. *ACM SIGKDD Explorations Newsletter*, 12(2):31–35, 2011.

[21] Ron Kohavi and Matt Round. Front line internet analytics at amazon.com, 2004.

[22] Ron Kohavi, Randal M Henne, and Dan Sommerfield. Practical guide to controlled experiments on the web: listen to your customers not to the hippo. In *Proceedings of the 13th ACM SIGKDD Conference*, pages 959–967, 2007.

[23] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining Knowledge Discovery*, 18:140–181, 2009.

[24] Ron Kohavi, Alex Deng, Brian Frasca, Roger Longbotham, Toby Walker, and Ya Xu. Trustworthy online controlled experiments: Five puzzling outcomes explained. *Proceedings of the 18th ACM SIGKDD Conference*, 2012.

[25] Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker nd Ya Xu, and Nils Pohlmann. Online controlled experiments at large scale. *Proceedings of the 19th ACM SIGKDD Conference*, 2013.

[26] Ron Kohavi, Alex Deng, Roger Longbotham, and Ya Xu. Seven rules of thumb for web site experimenters. In *Proceedings of the 20th ACM SIGKDD Conference*, KDD '14, pages 1857–1866, 2014.

[27] Ronny Kohavi, Thomas Crook, Roger Longbotham, Brian Frasca, Randy Henne, Juan Lavista Ferres, and Tamir Melamed. Online experimentation at microsoft. In *Proceedings of the Third International Workshop on Data Mining Case Studies, held at the 5th ACM SIGKDD Conference*, pages 11–23, 2009.

[28] Dan McKinley. Design for continuous experimentation: Talk and slides.[online] dec 22, 2012.

[29] Mike Moran. Multivariate testing in action: Quicken loan's regis hadiaris on multivariate testing. *Biznology Blog by Mike Moran*, 2008.

[30] Christian Posse. Key lessons learned building recommender systems for large-scale social networks. In *Proceedings of the 18th ACM SIGKDD Conference*, pages 587–587, 2012.

[31] Kerry Rodden, Hilary Hutchinson, and Xin Fu. Measuring the user experience on a large scale: User-centered metrics for web applications. In *Proceedings of CHI 2010*, 2010.

[32] Yang Song, Xiaolin Shi, and Xin Fu. Evaluating and predicting user engagement change with degraded search relevance. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, pages 1213–1224, 2013.

[33] Yang Song, Xiaolin Shi, Ryen White, and Ahmed Hassan Awadallah. Context-aware web search abandonment prediction. In *Proceedings of the 37th International ACM SIGIR Conference*, pages 93–102, 2014.

[34] Diane Tang, Ashish Agarwal, Deirdre O'Brien, and Mike Meyer. Overlapping experiment infrastructure: More, better, faster experimentation. *Proceedings of the 16th ACM SIGKDD Conference*, 2010.

[35] Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000. ISBN 0521784506.

[36] Hongning Wang, Yang Song, Ming-Wei Chang, Xiaodong He, Ahmed Hassan, and Ryen W. White. Modeling action-level satisfaction for search task satisfaction prediction. In *Proceedings of the 37th ACM SIGIR Conference*, 2014.