Context-based A/B Test Validation

Michael Nolting
Sevenval Technologies GmbH
Köpenicker Str. 154
10997 Berlin, Germany
michael.nolting@sevenval.com

Jan Eike von Seggern
Sevenval Technologies GmbH
Köpenicker Str. 154
10997 Berlin, Germany
eike.seggern@sevenval.com

ABSTRACT

Data-driven and continuous development and deployment of modern web applications depend critically on registering changes as fast as possible, paving the way for short innovation cycles. A/B testing is a popular tool for comparing the performance of different variants. Despite the widespread use of A/B tests, there is little research on how to assert the validity of such tests. Even small changes in the application's user base, hard- or software stack not related to the variants under test can transform on possibly hidden paths into significant disturbances of the overall evaluation criterion (OEC) of an A/B test and, hence, invalidate such a test. Therefore, the highly dynamic server and client run-time environments of modern web applications make it difficult to assert correctly the validity of an A/B test.

We propose the concept of test context to capture data relevant for the chosen OEC. We use pre-test data for dynamic base-lining of the target space of the system under test and to increase the statistical power. During an A/B experiment, the contexts of each variant are compared to the pre-test context to ensure the validity of the test. We have implemented this method using a generic parameter-free statistical test based on the bootstrap method focussing on frontend performance metrics.

Keywords

Frontend performance monitoring; A/A testing; A/B testing; bootstrapping; dynamic base-lining

1. INTRODUCTION

A/B testing is a common pattern for gradient-based, data-driven optimization of user experience. Interpreting the results of A/B tests pose challenges concerning statistical inference and the reduction of the variability of the results. As has to be done for every statistical test and even before analyzing an A/B test, its validity must be asserted [5]. To validate an A/B test for variants of some feature X, all metrics have to be checked that might impact the overall evaluation criterion (OEC) and are not dependent on X. If any of these metrics differs significantly between the samples for variants A and B the A/B test might be invalid. For example, consider an A/B test of a backend feature of a web application using the conversion rate as OEC. Such a test should not influence the Javascript error rate of the client application. If the error rates differ between samples A and B, e.g. due to different browser-variant shares in the samples, this might affect the conversion rate and, thus, enhance, balance or even invert the effect of the backend feature that is the subject of the test — that is invalidating the test.

We propose the concept of a *test context* to assert the validity of an A/B test. This concept is intended to assist other A/B testing frameworks (e.g. PlanOut [2]). A context will contain the metrics that might impact the overall evaluation criterion (OEC) of the A/B test. A pre-test sample is used to define the *target space* of the tested system (dynamic base-lining) because, first, this procedure relieves the user from defining the target space by hand (a tedious and complicated task even for a test context containing only a handful of metrics) and, second, it increases the statistical power of the validation, i.e. the required sample size to detect a given effect size if one of the metrics worsens [3].

In this paper, we present our implementation of this concept, `history-diagnostics` [1]. Focussing on frontend centric metrics, we will shortly introduce the method in Section 2 followed by an exemplary application in Section 3 and our conclusions and outlook in Section 4.

2. CONTEXT-BASED TEST VALIDATION

Using pre-experiment data to improve the statistical power of A/B tests has been used elsewhere [3]. But to our knowledge, pre-experiment data has not been used to improve the validation of A/B tests. In the following we will give a general overview of the method followed by an exemplary application to frontend-centric metrics.

The method expects three different data samples: the pre-experiment sample S_{TS} to define the target space and the test-variant samples S_{A} and S_{B}. Each sample can contain data from different sources, e.g. front- and backend. Based on these samples, the metrics relevant to the test under validation are calculated. The user can define a set of functions of the samples to be used as metrics. Each metric m is expected to be scalar and ordinal, i.e. there is a single direction in which the metric improves. Possible metrics are the performance or error rates of the front- and backend. The method consists of the following steps:

WWW'16 Companion, April 11–15, 2016, Montréal, Québec, Canada.
ACM 978-1-4503-4144-8/16/04.
http://dx.doi.org/10.1145/2872518.2889306.

1. Dynamic base-lining of the target space by drawing bootstrap samples from S_{TS} and estimating the distributions of the metrics [4].
2. Calculate the metrics for samples S_{A} and S_{B} and, based on the bootstrapped distributions, the probability to find worse metric values.
3. Report the smallest of these probabilities,

$$\min_{m \in \mathrm{metrics}} P_{\mathrm{BS}}\left[M \text{ worse than } m(S)\right], \qquad (1)$$

for S_{A} and S_{B} individually.

If one of these two reported figures is below 5 %, we have a strong indication that the test is not valid. This method is flexible in two ways: It makes no assumptions about the distribution of the sampled data or metrics[1] and it can take into account data from many different sources.

Focussing on frontend-centric metrics, the data samples consist of frontend requests. For each request, we record its time t, the frontend performance p and any number of events v_k, e.g. Javascript errors that occurred during the request.[2] Typically, the event records are binary (1 or 0), i.e. event k did occur or not. A possible set of metrics based on this data is the following: To estimate the overall performance, we use the median,

$$\pi(S) = \mathrm{median}(p_1, \ldots, p_n), \qquad (2)$$

to be more robust against outliers. The traffic rate is estimated by

$$\tau(S) = n/T\,, \qquad (3)$$

where n is the number of requests in sample S and T is the sampling duration. We estimate the event rates similarly but normalize them to the observed traffic,

$$\phi_k(S) = \sum_{i=1}^{n} v_{k,i}/n\,. \qquad (4)$$

When drawing bootstrap samples care has to be taken that not only the performance and event rates vary but also the traffic rate. In our implementation this is achieved by reweighting the sampled data with Poissonian weights.

3. EXEMPLARY APPLICATION

To illustrate the performance of context-based A/B test validation, we consider an A/B test in which partial page loading (PPL) was enabled for a medium-traffic (160 page views per hour) web application to improve the application's frontend performance. In this test, the application's OEC did not improve significantly. Fig. 1 shows the response of the minimal probability, Eq. (1), for the frontend-centric metrics defined above in a semi-logarithmic plot. The only event taken into account was the occurrence of Javascript errors. The curve of the PPL-enabled test case B drops below the significance threshold of 5 % (black line in Fig. 1) early on during the test, while the original variant A stays at values 1, indicating no significant change. Further investigations of this case showed, that the drop of variant B was due an increased frontend error rate, which counter-acted improvements of the application's OEC due to an improved frontend performance.

[1] Except for metrics being scalar and ordinal.

[2] Our repository [1] contains Javascript code to collect the frontend data.

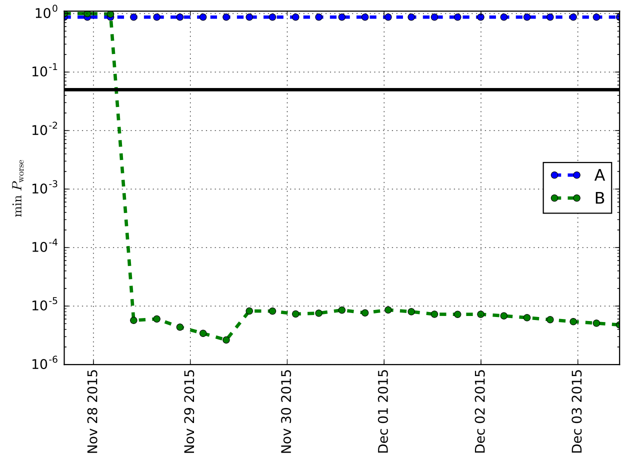

Figure 1: Exemplary application of context-based A/B test validation. Variant A is un-modified, B is with PPL enabled.

4. CONCLUSIONS AND OUTLOOK

We presented a method for context-based A/B test validation that uses pre-experiment data to increase statistical power. Our implementation is available on Github [1] together with a tutorial. We use this method in our frontend-performance monitoring dashboard FDX[3] to validate A/B tests. From that usage, we selected an exemplary use case that shows the power of this method to detect disturbances in the context of an A/B test that invalidate this test. We plan to improve the method's power by adding more metrics and using statistical tests more suiting to specific metrics in addition to the generic non-parametric bootstrap method. Furthermore, we explore a tighter integration with PlanOut [2] to simplify usage.

5. ACKNOWLEDGMENTS

This work was partially funded by the ProFIT program of the Investitionsbank Berlin.

6. REFERENCES

[1] `https://github.com/sevenval/history-diagnostics`.

[2] Bakshy, E., Eckles, D., and Bernstein, M. S. Designing and Deploying Online Field Experiments. In *Proceedings of the 23rd International Conference on World Wide Web*, (2014), ACM, pp. 283{292.

[3] Deng, A., Xu, Y., Kohavi, R., and Walker, T. Improving the sensitivity of online controlled experiments by utilizing pre-experiment data. In *Proceedings of the sixth ACM international conference on Web search and data mining* (2013), ACM, pp. 123{132.

[4] Efron, B., and Tibshirani, R. J. *An Introduction to the Bootstrap*. Chapman & Hall, New York, 1993.

[5] Kohavi, R., and Longbotham, R. Online controlled experiments and A/B tests. In *Encyclopedia of Machine Learning and Data Mining*, C. Sammut and G. Webb, Eds. 2015.

[3] FDX: Frontend Data Analytics