# AB Testing - Business Learning for Data Science

Avinash Barnwal

March 2019

## 1 Motivating Examples

### 1.1 Checkout page at Doctor FootCare

Conversion rate of users is the percentage of visits to website that include a purchase.
One variant had coupon code option and other variant didn't have it. Variant having coupon code had lower conversion rate as users started thinking twice about whether they were paying more.

### 1.2 Ratings of Microsoft Office help articles

Classical problem of putting the more ads comparing against users experience. This problem is categorized as Overall Evaluation Criteria(OEC). Page Views and Clicks are the key metrics considered and have been assigned monetary value.
Page Views have an assigned value based on ads and clicks to destination from the MSN are determined based on two ways:-

- **Gain** - Monetary value that the destination property assigned to a click from the MSN home page.
- **Cost** - The cost paid to search engines to generate leads to reach the destination links.

Idea is to first analyze the click through rate(CTR) and then convert the CTR to monetary value to decide on different variants.
**Final result** - A controlled experiment was run on 5% for 12 days.
Clickthrough rate decreased by 0.38% and p-value = 0.02.
Translating the lost clicks to their monetary value , it was higher than the expected ad revenue, so idea of adding more ads to MSN home page was scrapped.

### 1.3 Behavior-Based Search at Amazon

Its a new kind of recommendation algorithm where algorithm suggests item based on "People who searched for X bought item Y." It gave some spurious products but using controlled experiment it increased by revenenue by 3

## 2 Controlled experiments

### 2.1 Terminology

**Overall Evaluation Criterion (OEC)** - A quantitative measure of the experiment's objective. Metrics can be short-term(i.e. clicks) but we should also consider long-term aspects like predicting life-time value and repeat visits.

**Factors** - A controllable experimental variable that is thought to influence the OEC. Factors are assigned values, leading to single factor with two values : A and B.

**Variant** - A user experiences different features which we are testing, where one is Control and one which we are testing is the Treatment. In case of bug, for example, the experiment is aborted and all users should see the Control variant.

**Experimental unit** - The entity over which metrics are calculated. Sometimes called an item. The units are assumed to be independent.

**Null hypothesis.** - The hypothesis,that the OECs for the variants are not different and any observed differences during the experiments are due to random fluctuations.

**Confidence level** - The probability of failing to reject(i.e., retaining) the null hypothesis when it is true.

**Power** - Power measures the our ability to detect a difference when it indeed exists.

**A/A test** - Instead of an A/B test, you exercise the experimentation system, assigning users to one of two groups, but expose them to exactly the same experience. An A/A test can be used to (i) collect data and assess its variability for power calculation. (ii) test the experimentation system.

**Standard-deviation** - A measure of variability, typically denoted by $\sigma$.

**Standard-Error** - It is standard deviation of sampling distribution of sample statistics.

### 2.2 Hypothesis testing and sample size

Factors impacting the test:-

- Confidence Level :- Commonly set to 95%, this implies tht 5% of the

time we will incorrectly conclude that there is a difference when there is none(Type I error). All else being equal , increasing this decreases the power.

- Power :- The probability of determining that the difference is statistically significant.
- Standard error :- The smaller the Std-Err, the powerful the test. Three ways to reduce Std-Err
  - Increase the sample size.
  - Choose better OEC such as proportion.
  - Exclude users which are not impacted in the experiments.
- Effect :- The difference in OECs for the variants, i.e. the mean of Treatment minus the mean of the Control.

Two formulas are useful to share in this context. The first is t-test , used in A/B tests.

$$t = \frac{\bar{O}_B - \bar{O}_A}{\hat{\sigma_d}} \tag{1}$$

We assume throughout that the sample size is large enough to apply the Normal distribution even when it is skewed distribution.
Second formula is about the minimum sample size required for confidence level = 95% and power = 80%.

$$n = \frac{16\sigma^2}{\Delta^2} \tag{2}$$

A more conservative formula for sample size(for 90% power) has been suggested

$$n = \frac{4r\sigma^2}{\Delta^2} \tag{3}$$

### 2.2.1 Example: impact of lower-variability OEC on the sample size

Consider 2 types OECs - Purchasing amount and conversion rate.
Sample Size calculation - Purchasing amount For 5% change in revenue with 95% Confidence level and 80% power. Assume standard deviation is 30 , average spend is 3.75 and conversion rate is 5%.

$$n = \frac{1630^2}{3.75 * 0.05^2} n = 409,000 \tag{4}$$

Sample Size calculation - Conversion Rate

$$n = \frac{16 * (0.05) * (1 - 0.05)}{0.05^2} n = 122,000 \tag{5}$$

#### 2.2.2 Example: impact of reduced sensitivity on the sample size

$\Delta^2$ is inversely proportional to sample size (n), where if we increase the sensitivity then we less sample size and we can detect the bug early also.

#### 2.2.3 Example: filtering users not impacted by the change

Exclude the users which are not impacted by change in the features. This can lead to lower sample size with same power.

#### 2.2.4 The choice of OEC must be made in advance

It should be made in advance and with the corrections.

### 2.3 Confidence intervals for absolute and percent effect

#### 2.3.1 Confidence intervals for absolute effect

$$CILimits = \bar{O_B} - \bar{O_A} + -1.96 * \hat{\sigma_d} \tag{6}$$

#### 2.3.2 Confidence intervals for percent effect

The percent difference is calculated by =

$$PctDiff = \frac{\bar{O}_B - \bar{O}_A}{\bar{O}_A} * 100\% \tag{7}$$

$$CV_B = \frac{\hat{\sigma_B}}{\bar{O}_B} \tag{8}$$

$$CV_A = \frac{\hat{\sigma_A}}{\bar{O}_A} \tag{9}$$

C.I. for Percent Effect =

$$(PctDiff + 1) * \frac{1 + -1.96 * \sqrt{CV_A^2 + CV_B^2 - (1.92)^2 * CV_A^2 * CV_B^2}}{1 - 1.96 * CV_A^2} - 1 \tag{10}$$

### 2.4 Effect of robots on experimental results

Robots can skew the estimates, enough to render the assumptions invalid. Even it can have significant effect in A/A test. It is especially important to filter out robots, that interact with the user-id.

#### 2.4.1 JavaScript versus server-side call

It is generally thought that very few robots will be included in the experiment if the treatment assignment is called by JavaScript so those experimental setups shouldn't be affected as much by robots.

#### 2.4.2 Robots that reject cookies

It is recommended to exclude unidentified requests from analysis, so that robots that reject cookies will not be part of experimental results.

#### 2.4.3 Robots that accept cookies

If a robot accepts cookies and doesn't delete them, the effect can be profound, especially if the robot has a large number of actions on the site. It can lead to large standard deviation of many metrics, thus reducing the power.
Thus, it is recommended to remove robots accepting cookies and have large number of actions.

### 2.5 Extensions for online settings

#### 2.5.1 Treatment ramp-up

An experiment can be initiated with a small percentage of users assigned to the Treatment(s), and then that percentage can be gradually increased. For example, if you plan to run an A/B test at 50%/50%, you might start with a 99%/0.1% split, then rampup the Treatment from 0.1% to 0.5% to 2.5% to 10% to 50%. At each step, which could run for, say, a couple of hours, you can analyze the data to make sure there are no egregious problems with the Treatment as effect has inversely square relationship with sample size.

#### 2.5.2 Automation

Once an organization has a clear OEC, it can run experiments to optimize certain areas amenable to automated search. Multi-armed bandit algorithms and Hoeffding Races can be for such optimization.

#### 2.5.3 Software migrations

Experiments can be used to help with software migration. If a feature or a system is being migrated to a new back-end, a new database, or a new language, but is not expected to change user-visible features. A/B tests can be used to show the difference in software migration, helping identify bugs in the port.

### 2.6 Limitations

Following are the limitations :-

- Quantitative metrics, but no explanations. - "Why" is not known.
- Short term vs long term effects - Controlled experiments measure the effect on OEC during the experimentation period, typically a few weeks. Long-term should be part of the OEC. Like include some kind of penalty over not used advertisements.

- Primacy and newness effect - These are opposite effects that need to be recognized. It can bring "newness" bias to experienced users. Therefore, it is advised to run experiments only for new users.
- Features must be implemented - The feature may be a prototype that is being tested against a small portion, or may not cover all edge cases. Paper prototyping can be used for qualitative feedback and quick refinements of designs in early stages.
- Consistency - Users may notice they are getting different variant than their friends and family. It is relatively rare that users will notice the difference.
- Parallel experiments - Strong interactions are rare in practice. Pairwise statistical tests can also be done to flag such interactions automatically.
- Launch Events and Media Announcements. - If there is a big announcement made about a new feature, such that the feature is announced to the media, all users need to see it.

# 3 MultiVariable testing

Consider testing five factors on the MSN homepage in a single experiment. There are two primary benefits of a single MVT vs multiple sequential A/B tests to test the same factors:-

- Test as many factors in a short period of time.
- You can estimate interactions between factors.

There are three common limitations :-

- Some combinations of factors may give a poor user experience - Interaction can lead to worst user experience.
- Analysis and interpretation are more difficult - For an MVT you have the same metrics for many Treatment-Control comparisons (atleast one for each factor being tested) plus the analysis and interpretation of the interactions between the factors.
- It can take longer to begin the test.

## 3.1 Traditional MVT

Fractional factorial designs that are specific subsets of full factorial designs. These designs are popularized by Genichi Taguchi. The user must be careful to choose a design that will have sufficient resolution to estimate the main effects and interactions that are of interest.

With Five factors we can test it using full factorial, fractional factorial or a Plackett-Burman design.
For K=5, there are $2^5 = 32$ user groups are required. K=3, fractional factorial design exits. Generally, -1 denotes the control and 1 denotes the treatment. **Plackett-Burman** designs can be constructed where the factors are all at two levels with the number of users groups being a multiple of 4, so 4, 8, 12, 16, 20, etc. The number of factors that can be tested is number of groups minus 1. Similar properties in **Plackett-Burman** design is also a fractional factorial. Main goal is to reduce the number of user groups to test main effects and interactions with little or no confounding.
If we want to estimate all two factor interactions with five factors , we will need a fractional factorial design with 16 treatment combinations.

## 3.2 MVT by running concurrent tests

Fractions of the full factorial are used in offline testing because there is usually a cost to using more treatment combinations even when the number of experimental units does not increase. This doesn't happen with web where users are cheap. A side benefit is you can shut down the factors which are behaving egregious. The experiment that includes the remaining factors is not affected.
Factors not impacting Power :-

- Number of treatment combination cells.
- If sample size is same. It doesn't matter if you are using single factor or many or whether you are conducting eight run MVT.

Factors impacting Power :-

- Number of Variants for a factor. This effectly decreases the sample size for any comparison we want to make, whether the test is MVT or A/B test.
- Assigning less than 50% of the test population to the treatment. It is especially important for treatments in an MVT to have the same percentage of the population as the Control.

## 3.3 Overlapping experiments

This approach is to simply test a factor as a one-factor experiment when the factor is ready to be tested with each test being independently randomized. This is the approach you should take if you want to maximize the speed with which ideas are tested and you are not interested in or concerned with interactions.

### 3.4 Summary of MVT

The two alternatives are better than traditional MVT. The one you use would depend on your priorities. If you want to test ideas as quickly as possible and aren't concerned about interactions, use the overlapping experiments approach. If it is important to estimate interactions run the experiments concurrently with users being independently randomized into each test effectively giving you a full factorial test.

## 4 Implementation architecture

Implementing an experiment on website involves three components. Following are the three components:-

- **Randomization Algorithm** - The function that maps end users to variants.
- **Assignment Algorithm** - Uses the output of randomization algorithm to determine the experience that each user will see on the website.
- **Data Path** - Captures raw observation data as the users interact with website, aggregates it, applies statistics, and prepares reports of the experiments outcome.

### 4.1 Randomization Algorithm

There are three key properties to support statistically correct experiments:-

- End users must be equally likely to see each variant of an experiment(assuming a 50-50 split). There should be no bias toward any particular variant.
- Repeated aasignments of a single end user must be consistent; the end user should be assigned to the same variant on each successive visit to the site.
- No correlation between experiments.
- The algorithm may support monotonic ramp-up, meaning that the percentage of users who see a Treatment can be slowly increased without changing the assignment of users who were previously assigned to that Treatment.
- The algorithm may support external control, meaning that users can be manually forced into and out of the variants.

### 4.1.1 Pseudorandom with caching

A standard pseudo-number generator can be used as the randomization algorithm when coupled with a form of caching. A good pseudo-number would satisfy first and third requirements. We can test validity of Pseudo-random number generators by running simultaneous experiments and check that two experiments are correlated or not. It has been found that the random number generators built into many popular languages(for example , $C\#$) work well as long as the generator is seeded only once at server setup. Seeding the random number generator on each request may cause adjacent requests to use the same seed which may introduce noticeable correlations between experiments.
To satisfy second requirement, the algorithm must introduce state: the assignments of end users must be cached once they visit the site.
The caching can be accomplished either on the server side or on the client side(by storing a user's assignment in a cookie).
Cookie is cheaper but it will not work for users with cookies turned off.
Both forms of this approach are difficult to scale up to a large system with a large fleet of servers.
The server making the random assignment must communicate its state to all the other servers in order to keep assignments consistent.

### 4.1.2 Hash and partition

This method eliminates the need for caching by replacing the random number generator with hash function. Hash function produces randomly distributed numbers as a function of a specific input. Following are the steps:-

- User Identifier - Each user is assigned a single unique identifier which is maintained either through a database or a cookie.
- Experiment Identifier - Each experiment is assigned a single unique identifier which is maintained either through a database or a cookie.
- Hash Function - Apply function to combination of user identifier and experiment identifier to obtain an integer that is uniformly distributed on a range of values. The range is then partitioned, with each variant represented by a partition.

This produces range of numbers then partitioned, with each variant represented by a partition.
The method is very sensitive to the choice of hash function. Hash functions are not supposed to have below properties:-

- Funnels - Instances where adjacent keys map to the same hash code.
- Characteristics - Instances where a perturbation of the key produces a predictable perturbation of the hash code, then correlations may occur between experiments.

MD5 generated no correlations between experiments. SHA256 came close. Following are the limitations of MD5:-

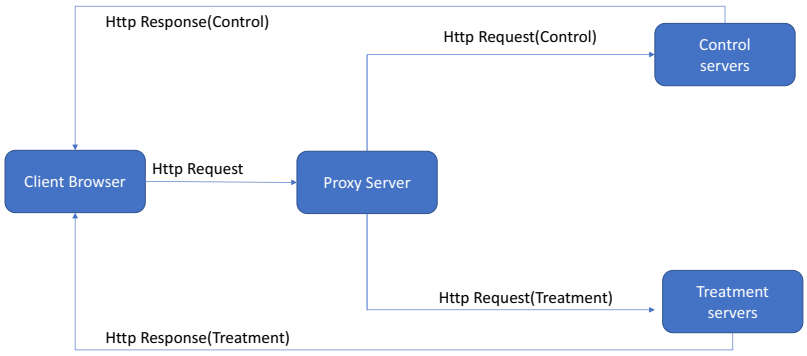
- Running-time performance of the hash function - Expensive and Partial caching fails.

Solution is to use a hybrid approach, combining the hash and partition method with either a small database or limited use of cookies.

## 4.2 Assignment method

The assignment method is the piece of software that enables the experiment website to execute a different code path for different end users.

### 4.2.1 Traffic splitting



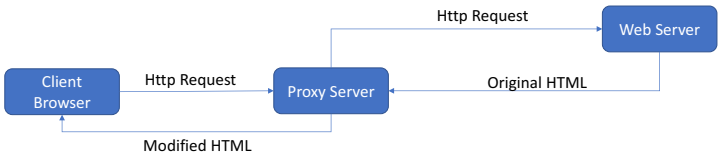

Following are the disadvantages:-

- Replicating application for small changes in features.
- Setting up and configuring parallel fleets is typically expensive.
- Running multiple experiments requires the fleet to support one partition for each combination of variants across all experiments.
- Any differences between the fleets used for each variant may confound the experimental results.

It is recommended to use it when we have significant code change.

### 4.2.2 Page rewriting

Page rewriting is an assignment method that incorporates a special type of proxy server that modifies HTML content before it is presented to the end user.

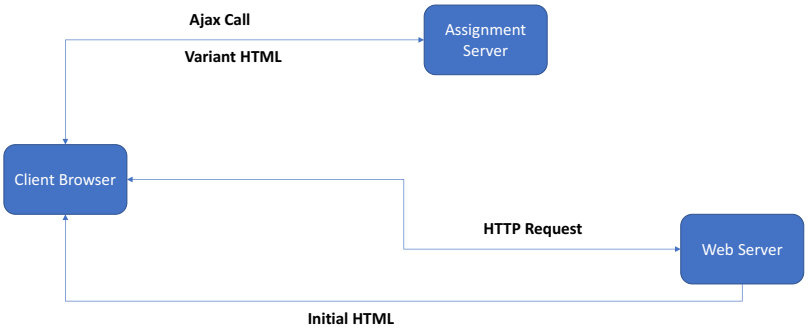



Like traffic splitting, this method is non-intrusive. Following are the disadvantages:-

- Page render time is impacted by the action of the proxy server.
- Experimentation on large sites requires significant hardware.
- Development and testing of variant content is more difficult and more error-prone.
- Running experiment on back-end algorithms is difficult.
- Running experiment on encrypted traffic is resource-intensive.

### 4.2.3 Client-Side assignment

All of these products can run an experiment without making any decisions on the server. A developer implements an experiment by inserting JavaScript code that instructs the end user's browser to invoke an assignment service at render time.
assignment.pdf



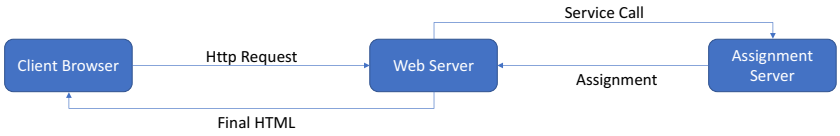

Following are the key limitations:-

- The client-side assignment logic executes after the initial page is served and therefore delays the end user experience.
- The method is difficult to employ on complex sites that reply on dynamic content because complex content can interact with the JavaScript.
- End users can determine that a page is subject to experimentation.

This method is best for experiments on front-end content that is primarily static.

#### 4.2.4 Server-Side assignment

Server-side assignment refers to a family of methods that use code embedded into the website's server to produce a different user experience for each variant. assignment.pdf




Following are the advantages:-

- Extremely general Method
- It places the experimentation code in the best logical place.
- Experimentation is completely transparent to end users.

Following are the disadvantages:-

- Initial implementation is expensive.
- Experimentation introduces risk.
- Some variations of this method require code changes to be manually undone to complete an experiment.

Server-side assignment can be integrated into a content management system to greatly reduce the cost of running experiments using this method. When so integrated, experiments are configured by changing *metadata* instead of code.

#### 4.2.5 Summary

| Family | Intrusive? | Implementation cost of first experiment | Implementation cost of subsequent experiments | Hardware Cost | Flexibility render time | Impact on |
|---|---|---|---|---|---|---|
| Traffic splitting | No | Moderate to high | Moderate to high | High | High | Low |
| Page rewriting | No | Moderate | Moderate | Moderate to high | Moderate | High |
| Client-side assignment | Yes | Moderate | Moderate | Low | Low | High |
| Server-side assignment | Yes | High | Moderate to low | Low | Very High | Very Low |

### 4.3 Data path

The website must collect raw data such as page views, clicks, revenue, render time, or customer-feedback selections. The system must then convert this raw data into metrics-numerical summaries that can be compared between variants of each experiment to determine the outcome.

#### 4.3.1 Event-triggered filtering

Data collected from web traffic on a large site typically has tremendous variability, therefore difficult to detect effects on smaller features.
One way to improve this variability is to restrict the analysis to only those users who were impacted by the experiment.
Event-triggered filtering is implemented by tracking the time at which each user first saw content that was effected by the experiment.

#### 4.3.2 Raw-data collection

#### 4.3.3 Using existing(external) data collection

Websites already have some data collection in place, either through an in-house system or an external metrics provider like Omniture or Webmetrics. For there websites, a simple approach is to push the treatment assignment for each user into this system so that it becomes available for analysis.

#### 4.3.4 Local data collection

using this method, the website records data locally, either through local database or log files. The data is collected locally on each server in the fleet and must be sorted and aggregated before analysis can begin.

#### 4.3.5 Service-based collection

Under this model, the website implements a service designed to record and store observation data. Service calls may be placed in a number fo locations, including web servers, application servers, backend algorithm services, and even the end user's browser.
Service-based collection is the most flexible and therefore preferred when possible.

# 5 Lessons Learned

## 5.1 Analysis

#### 5.1.1 Mine the data

A controlled experiment provides more than just a single bit of information about whether the difference in OECs is statistically significant. For example, an experiment showed no significant difference overall, but a population of users with a specific browser version was significantly difference overall, but a population of users with a specific browser version was significantly worse for the Treatment.

#### 5.1.2 Speed matters

A Treatment might provide a worse user experience because of its performance. If time is not directly part of your OEC, make make sure that a new feature that is losing is not losing because it is slower.

#### 5.1.3 Test one factor at a time (or not)

We believe the advice, interpreted narrowly, is too restrictive and can lead organizations to focus on small incremental improvements. Conversely, some companies are touting their fractional factorial designs and Taguchi methods, thus introducing complexity where it may nor be needed. Following are the recommendations are therefore:-

- Conduct single-factor experiments for gaining insights and when you make incremental changes that could be decoupled.
- Try some bold bets and very different designs.

- Use full or fractional factorial designs suitable for estimating interactions when several factors are suspected to interact strongly. Limit the number of values per factor and assign the same percentage to the treatments to the control. This gives your experiment maximum power to detect effects.

## 5.2 Trust and execution

### 5.2.1 Run continuous A/A tests

Run A/A tests and validate the following.

- Are users split according to the planned percentages?
- Is the data collected matching the system of record?
- Are the results showing non-significant results 95% of the time?

### 5.2.2 Automate ramp-up and abort

It is recommended that experimenters gradually increase the percentage of users assigned to the Treatments(s). An experimentation system that analyses the experiment data in near-real-time can automatically shut-down a Treatment if it is significantly under performing relative to the Control.

### 5.2.3 Determine the minimum sample size

Decide on the statistical power, the effect you would like to detect, and estimate the variability of the OEC through an A/A test. For these metrics it is important that we get an adequate number of users into the test per day and that the Treatment and Control groups are of equal size.

### 5.2.4 Assign 50% of users to treatment

In order to maximize the power of an experiment and minimize the running time, we recommend that 50% of users see each of the variants in an A/B test. Assuming all factors are fixed, a good approximation for the multiplicative increase in running time for an A/B test relative to 50%/50% is $\frac{1}{4p(1-p)}$ where teh Treatment receives portion p of the traffic. For example, if an experiment is run at 99%/1%, then it will have to run about 25 times longer that if it ran at 50%/50%.

### 5.2.5 Beware of day of week effects

Long experiments might show effects of weekend and analyzing them separately may lead to interesting insights.

## 5.3 Culture and business

### 5.3.1 Agree on the OEC upfront

A good technique is to assess the lifetime value of users and their actions. For example, a search from a new user may be worth than an additional search from an existing user.

### 5.3.2 Beware of launching features that "do not hurt" users

In the face of a **non significant difference** result, sometimes the decision is made to launch the change anyway. It is possible that experiment is negative but under-powered.

### 5.3.3 Weight the feature maintenance costs

An experiment may show a statistically significant difference between variants, but choosing to launch the new variant may still be unjustified because of maintenance costs.

### 5.3.4 Change to a data-driven culture

In a web world, we can integrate customer feedback directly through prototypes and experimentation. If an organization has done the hard work to agree on an OEC and vetted an experimentation system, experimentation can provide real data and move the culture towards attaining shared goals rather than battle over opinions.