UNIVERSITÄT OSNABRÜCK

Institute of Cognitive Science
Intuit Data Engineering and Analytics

Bachelor's Thesis

Comparing Models for A/B Testing

Andrea Suckro

June 25, 2015

First supervisor: Prof. Dr. Frank Jäkel
Second supervisor: Mita Mahadevan

Comparing Models for A/B Testing

This thesis is centered around the frequently used method of A/B Testing. We will start by describing a tool used by Intuit for that matter. Then different methods will be compared on a slightly simplified model. The discussion will show pro's and con's for any methods and try to sketch a useful picture for future development.

Contents

Chapter 1

Introduction

Motivation for the project. General motivation for automated tests in the field of software development. hope for an objective criterion to make business decisions. can be applied to design measured in clickrates, can be internal process , can be ... anything that can be measured. intuit has an own framework to conduct test scenarios called abntest. it is a webservice that can be easily integrated by any product and uses http requests to measure the characteristic numbers for a test scenario.

1.1 Terminology

certain terms will reoccur throughout the thesis and will be explained here. Test: Bucket: Action: A user that is assigned to a bucket may perform an action that is measured. For example - the clicking on a specific button may trigger the recording of this action. Sampling Rate: The sampling rate determines what percentage of all the users participate in the test. This decision takes place before each user is assigned to a bucket.

1.2 Intuit's A/B Testing Platform

Users have the possibility to create test cases for their projects. By encapsulating the whole process of performing and A/B Test the users can perform tests in any area of the product. It could be centered around UI and design choices or test backend functionality.

1.2.1 Creating a Test

The platform for creating the tests and measuring the metrics along the way is organized as a web service. The users built in code to their products to show the users different experiences based on their assignment that is determined by the A/B Testing service. When creating the test the users define the number of different experiences, the controll setting and how long the test should be run.

1.2.2 Running the Test

For any new user that is visiting the product the service determines if she is part of the experiment or not based on the sampling rate. If it is decided that she takes part in the testing the next step decides the bucket she is assigned to. The user will from now on always stay in the bucket to avoid shifting experiences from visit to visit.

1.2.3 Analyzing the Result

The service provides metrics and current states of the test cases throughout the testing phase. Though it is not encouraged, customers can still update their tests and for example increase the sampling rate or disable buckets.

Chapter 2

Classic AB-Testing

2.1 Assumptions and Setup

A test is set up with at least two buckets and a sampling rate that determines what percentage of all possible users should participate in the testing. If for a new user was decided that she takes part in the test the next part is assigning her to a bucket. Later in this chapter different mechanisms for this step will be discussed and evaluated. Now certain actions of the users are recorded. In the following we will be concerned with the easy case where there is basically only one action that happens, or does not happen.

One is generating actions with a probability $q1$ the other generates them with a probability $q2$. The prior distribution for all q is a Beta distribution, which is a convenient choice because it is a prior distribution for binomial proportions (the Beta distribution is the conjugate family for the binomial likelihood). We also assume no prior knowledge when creating the test case (uninformative prior $Beta(1,1)$). $N1$ customers get assigned to Bucket 1 and $N2$ to Bucket 2. They generate $k1$ and $k2$ actions. The posterior distributions for each individual case is:

$$q = f(k|N) \propto f(N|k)f(k)$$

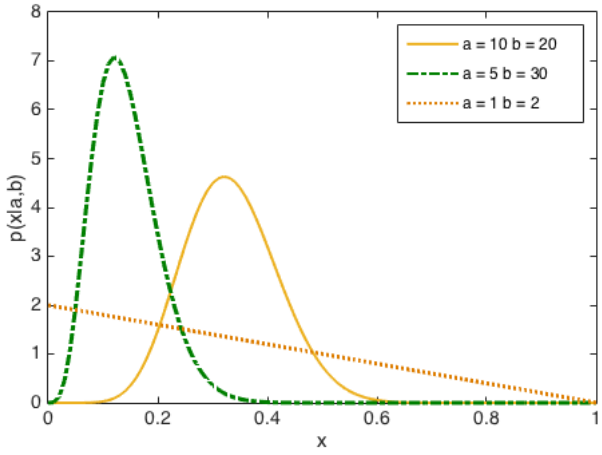
$$q1 = Beta(k1+\alpha, N1-k1+\beta) = \frac{x^{k1+\alpha-1}(1-x)^{N1-k1+\beta-1}}{B(k1+\alpha, N1-k1+\beta)}$$

$$q2 = Beta(k2+\alpha, N2-k2+\beta) = \frac{x^{k2+\alpha-1}(1-x)^{N2-k2+\beta-1}}{B(k2+\alpha, N2-k2+\beta)}$$

So each new assignment and each new event would directly alter the distributions parameters. *Questions: What is with returning users? Or users with a high action rate? Is every user a new one?* What is the posterior for $P(q1 > q2|k1, N1, k, 2, N2)$ - that Bucket 1 performs better than Bucket 2 given the collected data? There are in principal two different ways to get an answer.

2.2 Best Bucket

Finding the best Bucket among the other test scenarios can be achieved in several ways. It follows a set of methods. Each will be explained first for the simple case with two Buckets and then expanded to the general setting with n different cases.



2.2.1 Analytic

In general for two random variables X, Y with corresponding probability density function f_X, f_Y and cumulative density function F_X, F_Y it holds:

$$\begin{aligned} P(X \geq Y) &= \iint_{[x>y]} f_X(x) f_Y(y)\, dy\, dx \\ &= \iint_{-\infty}^{x} f_X(x) f_Y(y)\, dy\, dx \\ &= \int_{-\infty}^{\infty} f_X(x) F_Y(x)\, dx \end{aligned}$$

In our case with the two given buckets and the Beta Distribution that has only non-zero values in the range from 0 to 1 this formular resolves to:

$$P(q1 \geq q2) = \int_0^1 Beta_{q1}(k1, N1 - k1) I_{q2}(k2, N2 - k2)\, dx$$

Where I_{q2} is the regularized incomplete beta function. In the paper "Numerical Computation of Stochastic Inequality Probabilities" the author John D. Cook uses symmetries of the distribution to arrive at a set of equations that can be used to calculate the problem recursively.

$$\begin{aligned} g(k1, N1, k2, N2) &= P(X > Y) \\ h(k1, N1, k2, N2) &= \frac{B(k1 + k2, N1 + N2)}{B(k1, N1) B(k2, N2)} \end{aligned}$$

From that one could calculate a base case for a small sample and then continue with:

$$
\begin{aligned}
g(k1+1, N1, k2, N2) &= g(k1, N1, k2, N2) + h(k1, N1, k2, N2)/k1 \\
g(k1, N1+1, k2, N2) &= g(k1, N1, k2, N2) - h(k1, N1, k2, N2)/N1 \\
g(k1, N1, k2+1, N2) &= g(k1, N1, k2, N2) - h(k1, N1, k2, N2)/k2 \\
g(k1, N1, k2, N2+1) &= g(k1, N1, k2, N2) + h(k1, N1, k2, N2)/N2
\end{aligned}
$$

2.2.2 Approximation

For *sufficient large* samples one could again approximate the Beta Distribution with a normal distribution. This means that the following equation should be fulfilled for $B(k, N)$:

$$\frac{k+1}{k-1} \approx 0, \frac{N+1}{N-1} \approx 0$$

The Gaussian Distribution to a given Beta Distribution has the following shape:

$$B(k, N) \approx N\left(\frac{k}{k+N}, \sqrt{\frac{kN}{(k+N)^2(k+N+1)}}\right)$$

The inequality for this case then could be solved by:

$$
\begin{aligned}
P(X > Y) &= P(0 > Y - X) \\
&= P(0 > \mu_Y - \mu_X + (\sigma_X^2 + \sigma_Y^2)^{\frac{1}{2}} Z) \\
&= P\left(Z < \frac{\mu_X - \mu_Y}{(\sigma_X^2 + \sigma_Y^2)^{\frac{1}{2}}}\right) \\
&= \Phi\left(\frac{\mu_X - \mu_Y}{(\sigma_X^2 + \sigma_Y^2)^{\frac{1}{2}}}\right)
\end{aligned}
$$

I am not sure if this is really a smart way to do it but what would be maybe interesting is finding out which of the possibilities is the fastest for growing N.

2.2.3 Sampling

One could also sample from the different distributions and take the mean of the of the samples to determine the best distribution. The mean of a Beta distribution $B(k, N)$ is given by:

$$E[X] = \frac{k}{k+N}$$

Open question here: how exactly is the sample size related to the accuracy of the estimate?

2.2.4 Discussion

The generalization would again depend on the chosen method to evaluate the inequality.

Analytic

I found examples for 3 Beta distributed random variables. But basically any formula would calculate the probability that one variable is greater than the others which results in $n-1$ equations to solve the problem for n buckets. The same number of equations would occur if we would compute the inequality pairwise with the already presented formula.

Sampling

Additional Buckets would not change the underlying computation but would result in additional draws and new means to compare.

2.3 Best Assignment

It is not obvious what the best strategy is to assign a new user to a specific bucket. I will try to compare several intuitive approaches in this section. Interesting cases for the different strategies are: How fast does each strategy converge? What happens if there is no clear winner?

2.3.1 Uniform

For each new user we alternate the assignment uniformly between the buckets. So for two buckets the assignment alternates between bucket 1 and bucket 2.

2.3.2 Random

The assignment is based on a random draw that yields the next bucket. The two methods comparison is shown in the following figure:

2.3.3 Entropy

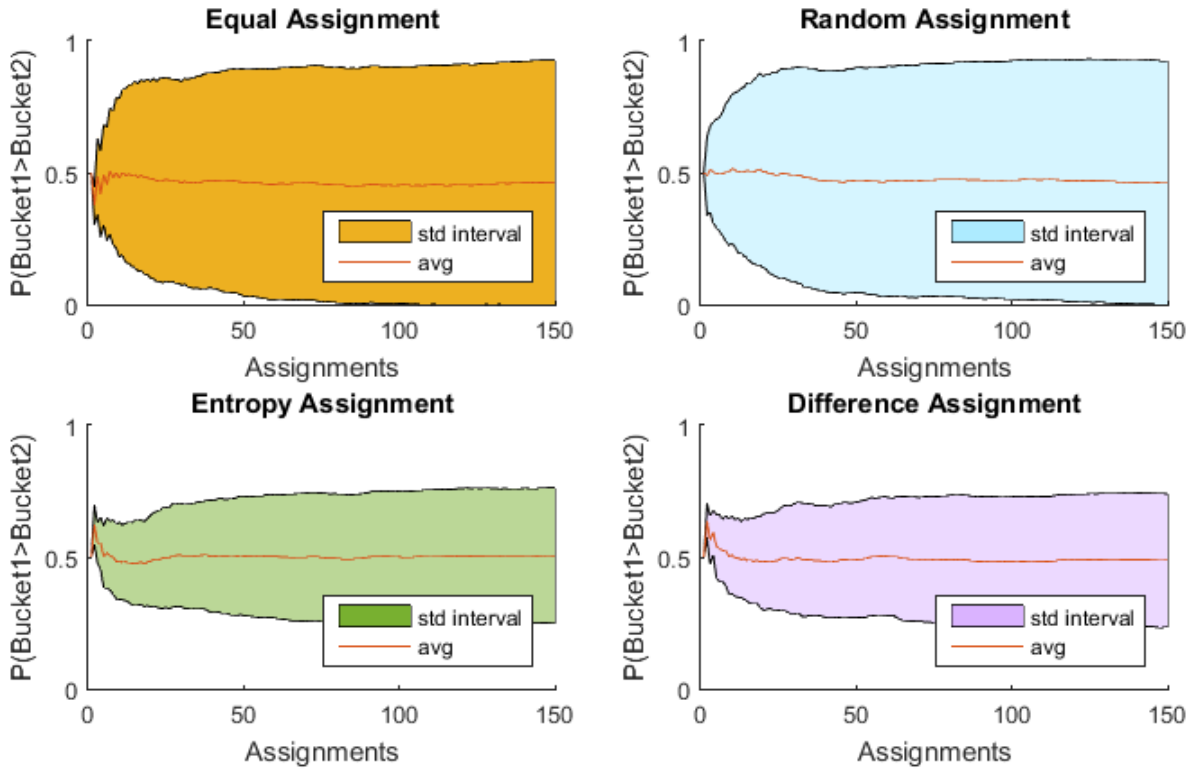
The best assignment for a not yet tracked user would be a bucket that minimizes the entropy over the different buckets. The entropy of $P(q1 \geq q2)$ is given by:

$$\begin{aligned} h(q) &= \int_X f(x) \log f(x) dx \\ &= \int_0^1 P(q1 \geq q2) log P(q1 \geq q2) dx \end{aligned}$$

We will use the natural logarithm (result in Nit). The assignment will be given to the bucket with the lowest result.

2.3.4 Discussion

and why it's wrong...



Different Beta Distributions

Chapter 3

Bandit AB-Testing

3.1 Underlying Idea

The previous chapter dealt with optimization concerning classical AB-Testing. The following will describe a fundamentally different approach in which there is no separation between the exploration and the exploitation phase. This is motivated by the idea that valuable costumers could be lost during exploration. Basically this means that we want to already maximize our profit while gaining information about the other buckets. When combining exploration and exploitation one has to chooce on each step if a known good bucket should be used again or if one invests in other unknown buckets to gain more information about their reward function. This balancing is at the core of the now described algorithms.

3.1.1 Bandit Algorithms

Bandit algorithms originate from Machine Learning where they serve learning agents to show sensible behavior while exploring the environment. In each time step an agent is forced to make a choice among several actions. An action will lead to a reward. The reward function for any action is not known from the beginning and the agent can only estimate the true reward function of any action by trying this action. The objective is of course to maximize the received reward over the whole experiment. The name of this class of algorithms stems from their relation to the famouse casino slot machines. Each action can be imagined as one lever in a row of many one-armed bandits. Each machine has a hidden reward function and a player wants to maximize their earned money at the end of the evening. The update rule for each step can be roughly formulized as:

$$NewEstimate \leftarrow OldEstimate + StepSize[Target - OldEstimate]$$

A K-armed bandit problem is defined by random variables $X_{i,n}$ for $1 \leq i \leq K$ and $n \geq 1$ where each i is the index of a slot machine. Successive plays of machine i yield reward $X_{i,1}, X_{i,2}, ...$ which are independent and identically distributed according to an unknown law with unknown expectation μ_i. The machines are also independent.

3.1.2 Reformulating AB-Testing to Bandits

Each new user is our chance to either explore or exploit what we know about the buckets of the test. Each assignment becomes the pull on the lever. The assignments are also independent and identically distributed. We do not know when or if at all the user's behavior will lead to the recording of an action. We will have to assign users before getting the feedback of our previous assignments which leads to a class of bandit algorithms that is concerned with *delayed feedback*. We deal with a stationary environment which means our $StepSize$ is $\frac{1}{k}$ where k denotes the number of assignments. Since we use no prior knowledge there is also *no side information*.

Terms

We want to find the real distributions underlying the variations we made in the buckets. Assume this distributions to be of the form $P_B(y|w)$ where w are the unknown parameters that describe the distribution. By assigning users we generate random observations $D_B = (y_1, y_2..y_t)$.

3.2 Bandit algorithms

3.2.1 epsilon-greedy

One of the simplest algorithms exploits always the best performing bucket (with the highest conversion rate) exept for ϵ's fraction of cases where the next bucket is choosen uniformily. For example: if $\epsilon = 0.1$ every tenth assignment would not be to the best performing bucket. This also means that even after convergence for the bucket probabilities 10% of the assignments would not always hit the optimal bucket. For n buckets this would be:

$$P(exploration) \cdot P(\neg bestmachine|exploration) = \epsilon \cdot \frac{n-1}{n}$$

A simulation with 1000 times 1000 assignments gives the following plot for this algorithm:

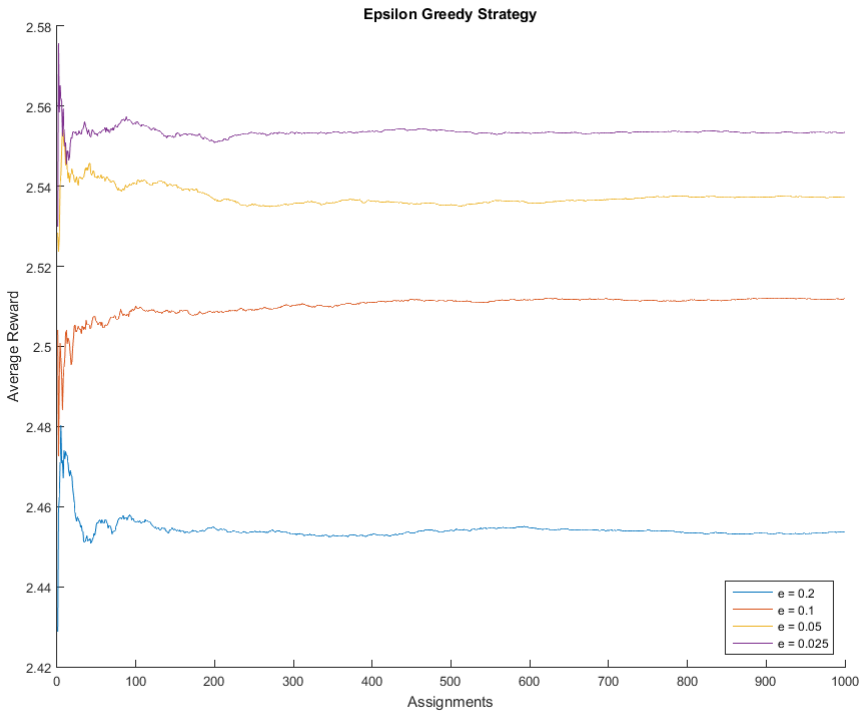
3.2.2 epsilon-first

Another algorithm that is also basically used by Visual Website Optimizer is called the $\epsilon - first$ algorithm. This is close to A/B-Testing since the exploration phase proceeds the exploitation phase for a finite number of steps and afterwards just exploit.

3.2.3 UCB

This algorithm also explores first, but not until a finite number of assignments. The Hoeffding's inequality gives us a confidence bound that models hot sure we are about the estimated payoff across multiple plays. This way - the longer we do not play a machine the more likely it will be played in the future (but only once). The probability for each machine is defined as:

$$mean(b_n) + \sqrt{\frac{2ln(N)}{n_p(b_n)}}$$

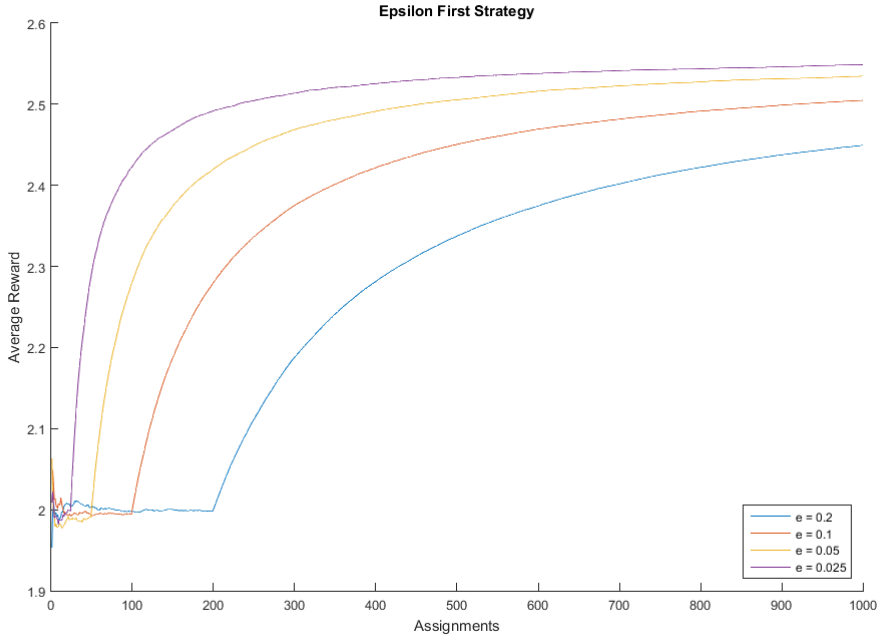


Here b_n is one of n buckets and $n_p(b_n)$ is the number of times this bucket has been played. N corresponds to the number of assignments across all buckets. This algorithm requirs $mean(b_n)$ to be bounded by 1 so that the confidence bound overpowers the first term. The fraction of time that is spend on exploring decreases exponentially.

3.2.4 Thompson sampling

To determine the next bucket to be played, the buckets probability distribution itself is used. They generate numbers that naturally balance between exploitation and exploration, because less performing buckets are demoted and winners promoted by the information already collected.

3.2.5 Discussion



Epsilon First

Chapter 4

Discussion

Hier schreiben dass die normalen AB-Tests schneller signifikante Ergebnisse liefern. Unterschied zwischen dem was man untersucht. Nicht reiner Gewinnnutzen.

Die allgemeine Meinung geht ja zu bandits aber die haben warscheinlich auch nachteile- nochmal genau rauslesen, Paper von Frank- ich glaube irgendwas von, dass man halt ne menge daten für nen vernünftigen prior braucht etc.

List of Figures

Declaration of Authorship

I hereby certify that the work presented here is, to the best of my knowledge and belief, original and the result of my own investigations, except as acknowledged, and has not been submitted, either in part or whole, for a degree at this or any other university.

Osnabrück, June 25, 2015