Lappeenranta-Lahti University of Technology
School of Engineering Science
Industrial Engineering and Management

Miki Kaukanen

Evaluating the impacts of machine learning to the future of A/B testing

Master's thesis

Examiners: Professor D.Sc. (Tech.) Marko Torkkeli
Associate Professor D.Sc. (Tech.) Kalle Elfvengren

ABSTRACT

Author: Miki Kaukanen
Title: Evaluating the impacts of machine learning to the future of A/B testing

Year: 2020 **Place:** Espoo, Finland

Master's thesis. Lappeenranta-Lahti University of Technology, Industrial Engineering and Management
115 pages, 16 figures and 13 tables
Examiners: Professor D.Sc. (Tech.) Marko Torkkeli
Associate Professor D.Sc. (Tech.) Kalle Elfvengren

Keywords: A/B testing, software product development, machine learning, multi-armed bandit

The incremental nature of contemporary software development necessitates companies to assess and validate where to place their development efforts. A/B testing is an established, widely used practice within the software industry to evaluate and learn the impact of product changes to the customer behavior and ultimately to the overall business performance. Lately, machine learning methods have started to gain wider attention across the industry, enabling new opportunities in evaluating and optimizing different product features.

This study establishes a detailed overview on the current A/B testing practices as well as examines how and where can machine learning methods potentially be leveraged in companies' experimentation and product development activities in the coming years. The topic is first studied through a comprehensive literature review, which is then followed by a single case study utilizing both quantitative and qualitative evidence from a contextual multi-armed bandit experiment conducted using real end users of a game application.

The findings of the literature review as well as the practical industry evidence from the case study indicate that the multi-armed bandit machine learning algorithms complement existing A/B testing practices by having their distinct use cases as well as providing an option for evaluating the more simple changes. The contextual bandit approach is particularly interesting as it shifts the focus on personalizing the features to the end users based on their predicted preferences. The framework for the use cases of multi-armed bandits established in the thesis together with guidelines from existing research show that the companies can benefit from the use of both A/B testing and multi-armed bandits jointly in their product development activities.

TIIVISTELMÄ

Tekijä: Miki Kaukanen
Työn nimi: Koneoppimisen vaikutuksien arvioiminen A/B-testaukseen tulevaisuudessa

Vuosi: 2020 **Paikka:** Espoo, Suomi

Diplomityö. Lappeenrannan-Lahden teknillinen yliopisto, LUT School of Engineering Science, Tuotantotalouden koulutusohjelma
115 sivua, 16 kuvaa ja 13 taulukkoa
Tarkastajat: Professori (TkT) Marko Torkkeli
Apulaisprofessori (TkT) Kalle Elfvengren

Hakusanat: A/B-testaus, ohjelmistotuotekehitys, koneoppiminen, multi-armed bandit

Nykyaikaisen ohjelmistokehityksen inkrementaalinen luonne edellyttää yrityksiä arvioimaan ja validoimaan mihin asettaa kehityspanoksensa. A/B-testaus on vakiintunut ja laajalti käytetty menetelmä ohjelmistoalalla arvioimaan ja saamaan selville tehtävien tuotemuutosten vaikutukset asiakaskäyttäytymiseen ja perimmiltään tuotteen kokonaisliiketoimintaan. Viime aikoina koneoppimismenetelmät ovat alkaneet saada laajempaa huomiota alalla, avaten uusia mahdollisuuksia tuotemuutosten ja ominaisuuksien arvioimiseen sekä optimoimiseen.

Tehty tutkimus rakentaa yksityiskohtaisen kuvan tämänhetkisistä A/B-testauksen käytännöistä ja tarkastelee kuinka sekä missä tapauksissa yritykset potentiaalisesti voivat käyttää koneoppimismenetelmiä hyväkseen ohjelmistotuotekehitykseen liittyvässä variaatioiden vertailussa. Aihetta tarkastellaan ensin perusteellisen kirjallisuuskatsauksen kautta, jota seuraa kvantitatiivista ja kvalitatiivista dataa hyödyntävä tapaustutkimus peliapplikaatiossa loppukäyttäjillä tehdystä, kontekstuaalista multi-armed bandittia hyödyntävästä testistä.

Tulokset kirjallisuuskatsauksesta sekä käytännön näyttö tapaustutkimuksesta indikoivat että multi-armed bandit koneoppimisalgoritmit tukevat ja täydentävät nykyisiä A/B-testaus käytäntöjä mahdollistaen erillisiä selkeitä käyttötapauksia sekä vaihtoehtoisen tavan arvioida yksinkertaisempia muutoksia. Kontekstuaaliset multi-armed bandit algoritmit ovat erityisesti merkillepantavia sillä ne nykyisestä poiketen siirtävät fokuksen tuoteominaisuuksien personointiin loppukäyttäjille perustuen algoritmin arvioon käyttäjän preferensseistä. Työssä esitetty viitekehys multi-armed bandittien käyttötapauksista yhdessä olemassa olevan tutkimuksen suuntaviivojen kanssa näyttävät, että yrityksille on hyötyä yhdistää A/B-testausta ja multi-armed bandittien käyttöä eri tapauksissa tuotekehitystoiminnassaan.

ACKNOWLEDGEMENTS

As the tradition goes, it is the time and place to express thanks to my supervisors and to the group of people who have supported me in writing the thesis.

First and foremost, I want to thank studio lead Tero Raij at Rovio for providing an opportunity to write this thesis, and giving me the trust and considerably free hands to construct it as I best see fit. The thesis wouldn't exist as it is without the flexibility and support from the company. I want also to give thanks to my supervisor at Rovio, Asko Relas, for supporting and pointing me towards useful resources, as well as Professor Marko Torkkeli for being there to ensure the content turns out academically appropriate. I would like to also extend my thanks to all my colleagues at Rovio who have contributed or given feedback during different stages of the project.

The whole thesis ended up being on the lengthy side, and with the effort put into it, I hope the content finds itself useful going forward, and also insightful to anyone in the industry looking to find themselves a bit wiser regarding the topic.

Espoo, 14.08.2020

Miki Kaukanen

TABLE OF CONTENTS

FIGURES

TABLES

ABBREVIATIONS

API	Application programming interface
ARPDAU	Average revenue per daily active user
ARPPU	Average revenue per paying user
ARPU	Average revenue per user
CVR	Conversion
IAP	In app purchase
ID	Identifier
KPI	Key performance indicator
MAB	Multi-armed bandit
MVF	Minimum viable feature
MVH	Minimum viable hypothesis
MVP	Minimum viable product
MVT	Multivariate test
OEC	Overall evaluation criteria
R&D	Research and development
ROI	Return on investment
SaaS	Software-as-a-service
UI	User interface

1 INTRODUCTION

The introductory chapter serves to guide the reader to the background and purpose of the thesis to better reflect the content of it. Moreover, the chapter presents the research objectives as well as the research questions the study seeks to answer, with the related methodology and data additionally described in brief. Lastly, finishing the chapter is the structure of the thesis with the contribution of each chapter being detailed.

1.1 Background

Companies can arguably only be successful if they are able to understand their customers' needs and develop products and services that can fulfill them, and accurately learning about customer needs has long been recognized as a vital part of product development (Fabijan et al., 2018a). Software companies during the last decade have increasingly shifted to develop and create products for new and highly dynamic domains with many technical and business uncertainties tied to them. Software products satisfying new customer needs or offering novel solutions that have not existed before can however often find themselves in a position where requirements are not always obvious and can't be defined in advance. This creates a situation where it is difficult or next to impossible for the company to evaluate and predict which product features or attributes create value for the customers, even if the customers were asked. (Lindgren & Münch, 2016)

Software companies have throughout decades been evolving their product development practices to answer the emerging needs in the changing environment (Fabijan et al., 2017a). Most recently, agile software development methods have risen in popularity to answer to the need of increased flexibility in determining and constantly updating software requirements (Wasserman, 2016). This contemporary nature of software development allows increased flexibility in types of services that can be delivered and optimized even after the software has been launched, enabling companies to continuously improve their software and solve problems that are relevant and deliver value for the customers. Developing the solutions to the problems however has often been haphazard and based more or less on educated guesswork. (Fagerholm

et al., 2014) According to Kohavi et al., the decisions regarding features in software development were not too long ago still commonly determined similar to prescribing medicine prior to World War II: by people regarded as experts making the call based on their experience-based guess, rather than the use of transparent, evidence based methods. (Kohavi, Longbotham, et al., 2009) Despite the goals and benefits of the contemporary agile development, the agile methods themselves fail to provide the tools and the framework towards developing software that can provide value to customers (Fagerholm et al., 2014).

Studies have shown that most development ideas in reality lead to provide negative or no value for the customer (Kohavi et al., 2013). Justifiably, data collection and analysis practices have become increasingly important as more than just supporting tools. They are widely used to learn in detail about customer behavior, usage patterns and ultimately product performance, as well as how these factors evolve throughout the lifecycle a software product. (Dmitriev et al., 2016; Holmström Olsson et al., 2017) Product usage data enables software companies to become more accurate in evaluating whether developed features and ideas add value to customers, ultimately raising the odds of success in developing products that satisfy intended outcomes for the customer (Lindgren & Münch, 2016).

In addition to collecting product data, companies can identify, prioritize and validate product assumptions by controlled experimentation. Software companies in a variety of domains have over the years been adopting product experimentation such as A/B testing to evaluate ideas and to accelerate innovation cycles (Holmström Olsson et al., 2017). Experimentation in software product development as a research area has been increasingly active in academia (Fabijan et al., 2018b), with several case studies also published on companies' experimentation success. Published research on the topic has mainly been focusing on challenges, statistical methods, design of experiments and technical infrastructure involved in experimentation and A/B testing (Ros & Runeson, 2018). Despite the prominent amount of research on technical topics and design of experiments, Ros & Runeson (2018) found on their mapping study a research gap especially in real world evaluation of technical topics discussed by researchers. Moreover, machine learning techniques such as multi-armed bandits present new intriguing opportunities to approach the subject of A/B testing (Scott, 2010). Indeed, machine learning is challenging the way experimentation is traditionally done in online systems (Issa Mattos et al., 2019).

This study aims to contribute existing research on these topics by shedding additional light on practical implications of conducting A/B testing in an organization and adopting machine learning practices to support a more advanced approach to online experimentation. To date, the research on these areas is still quite scattered with the exception of few active research teams, and there is a lack of aggregated view on the subject as a whole. This study establishes a solid baseline on A/B testing best practices and how is experimentation conducted effectively and robustly, as well as covering the practical benefits and limitations of novel machine learning approach to the subject of experimentation. Moreover, the research expands knowledge on how and where can machine learning be leveraged to realize advantages in online business development through experimentation, as well as recognizing what are the potential impacts of it.

1.2 Research objectives and scope

The objectives of this study can be summarized to consist of exploring and establishing what practices constitute a sustainable approach to A/B testing in an organization, which is then used to build upon and examine the possibilities opened up by applying machine learning methods to support the process. In addition, the goal is to further assess how and where can the different machine learning approaches be feasibly applied and what purpose do they serve in contrast to the traditional A/B testing approach. Typically, the methodology of A/B testing in the modern world is largely associated with companies in the online and software industry. The study seeks to contribute to existing research on A/B testing and machine learning within these industries and improve the comprehension of the topic as a whole from a practical industry point of view.

With these objectives, three research questions presented in Table 1 were formulated for the study. The first research question aims at identifying the current state of best practices and processes as well as the accompanied benefits and limitations associated with A/B testing. The purpose of the second research question is to narrow down the scope of machine learning approaches to those practically applicable in the context of A/B testing. Furthermore, the research question's objective is to characterize these approaches and make distinctions for a managerial level understanding between the available options. This also further means

determining the related challenges and drawbacks that need be understood with different approaches. The third research question intends to evaluate the practical implications and benefits for an organization applying machine learning to accompany its A/B testing practices. By utilizing the knowledge from the first two research questions, the third research question ultimately is set to align how do traditional A/B testing and machine learning based approaches coexist to enable a successful and sustainable approach to product development.

Table 1. Research questions and objectives

Research question	Objective
RQ 1. How do companies in software- and games industry utilize A/B testing to generate business insight?	Identify the underlying processes as well as the main benefits and limitations realized for the companies conducting A/B testing
RQ 2. What are the different types of machine learning approaches that can contribute to A/B testing and how are they differentiated?	Identify the scope of machine learning approaches that can be leveraged in the context of A/B testing and the synergies and challenges associated with each approach
RQ 3. What type of benefits are capable of being realized by utilizing a machine learning approach in A/B testing operations?	Evaluate the practical use cases and benefits for applying machine learning practices in real-life industry context to complement traditional A/B testing.

The findings of the study aim to provide an understanding of the applicability of machine learning based techniques within A/B testing operations for different organizations in the online and software industry. Thus, the scope of the study is accordingly narrowed down to these specific industries. Furthermore, the empirical part focuses particularly on evaluation of the topic in the software industry, or more specifically games industry, on the case company level. Effort is made throughout the thesis to allow reflecting the findings of the empirical part to the existing literature to allow better generalizability of the overall results.

1.3 Methodology and data

The methodology of the study can be divided into two parts. First, a semi-systematic literature review is carried out to provide a comprehensive picture of the discussed topics to understand the intricacies and implications on which the findings of the second part will be reflected and compared against. A semi-systematic approach to the literature review allows synthesizing relevant research findings on broader topic that has been conceptualized differently and studied within diverse disciplines (Snyder, 2019). The second half of the study is the empirical part which follows the principles of a case study. It combines both quantitative and qualitative approach by triangulating data from numerical results of the research as well as interpretive, descriptive data from participant observations as well as document analysis. The case study consists of execution of two A/B tests in the case company Rovio Entertainment Corporation. The first A/B test is executed with a traditional A/B testing methodology to establish a baseline on the experimented subject, with the following test ran utilizing a machine learning based approach replicating the same experimental setup.

Case study is a well-suited methodology for understanding the studied subject in-depth within its real-life context, as it typically combines different types of data to fully understand the dynamics of the case (Yin, 1993; Saunders et al., 2016). The research method still however poses limitations to the empirical study by relying only on one case and data exclusively from the case company, increasing also the potential of biases inherent with A/B testing methodology itself to influence the findings. Accordingly, the acquired quantitative data is accompanied with qualitative data to build a more profound understanding of the case on a practical, applied level and generate findings that are evaluated and validated against the existing research.

1.4 Structure of the report

The report consists of six main chapters. After the introductory chapter which presented the background with objectives and scope of the thesis, chapters two and three constitute the literature review part of the thesis. Chapter two introduces the concept of A/B testing as well as describes the main processes and practices related to it in detail, effectively building the big picture of the current state of A/B testing with its benefits and flaws. Chapter three moves to

cover machine learning methodology addressing the different approaches that can be utilized in the context of A/B testing, with their purpose and use cases examined. Chapter four presents the empirical research approach in more detail covering the research process, data collection and execution of the case study, followed by results of the case study. With chapter five the main objective is to build upon the previous chapters and discuss the findings from literature review and empirical study and arrive at an assessment of the future relationship between A/B testing and machine learning as well as point out identified suggestions for future research. Lastly, chapter six concludes the main learnings of the study. The contribution of each chapter and the inputs they are based on are summarized for convenience in Figure 1.

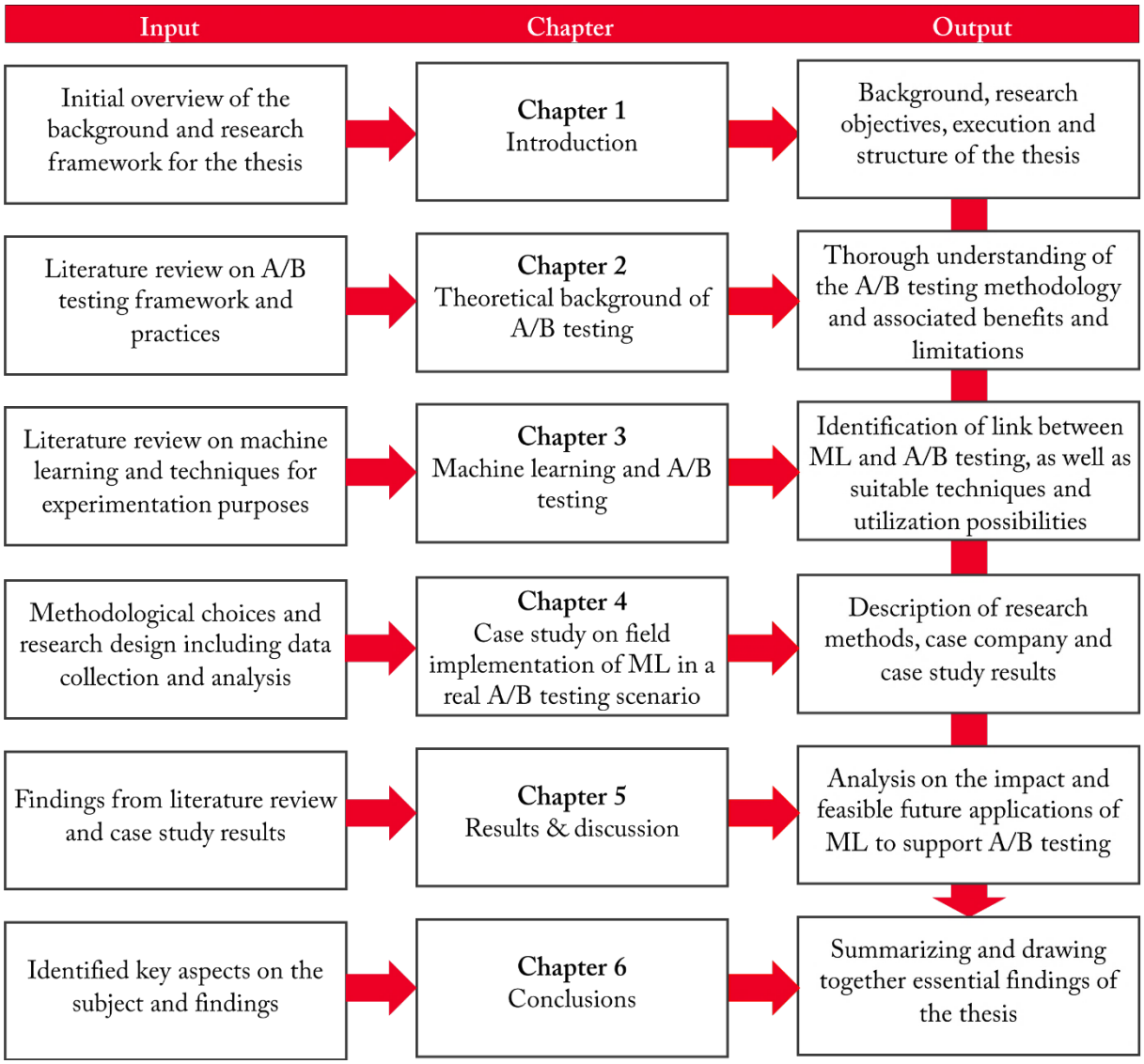

Figure 1. Structure of the thesis with input and output of each chapter

2 THEORETICAL BACKGROUND OF A/B TESTING

In software product development, choices on which features to develop, optimize and prioritize have to be constantly made. There are significant risks involved in deciding and prioritizing what should be developed in the product in order to sustain and create customer value. In addition, customers typically have a hard time of knowing what they would actually want in a software product as a result from lack of awareness on potential solutions, poor ability to predict what they want, as well as a gap between the actual actions and what the customer thinks and says. Thus, qualitative assessment through interviews or focus groups can fail to produce the optimal product decisions. (Lindgren & Münch, 2016) Although these methods remain essential in product development for understanding customer motives more in depth, basis for product decisions should originate from actual customer behavior and their patterns of using products and services (Fabijan et al., 2018a; Lindgren & Münch, 2016). According to Xie & Aurisset, running controlled experiments and basing product decisions on business metrics is the most effective way to bridge the prementioned gaps (Xie & Aurisset, 2016).

Software companies are increasingly collecting and using customer and product data in various ways to support decision making throughout the product lifecycle (Fabijan et al., 2015). Continuous data collection from the customer using the product in its real environment enables an unprecedented opportunity to evaluate ideas with customers in a fast and accurate way. Based on the data and any changes made to the product, it is possible to derive causal conclusions between the changes made to the product and the customers' reactions on them. (Fabijan et al., 2018a) Typically, these causal relationships on the changes to the product are established and verified through the use of A/B testing – a widely used controlled experimentation framework to evaluate new ideas and to make data driven decisions (Xu & Chen, 2016).

This chapter goes through the theory and different aspects of experimentation and A/B testing. The chapter starts by introducing the principal concept of A/B testing on a general level, and then proceeds to examine in detail the process models and the considerations in executing A/B tests. Followingly, the aspects of design and analysis of A/B test experiments are elaborated for enabling a more profound perception of the methodology. After having gained the detailed

understanding of the A/B testing framework, the last two parts of the chapter focus on defining the different benefits and limitations in utilizing A/B testing as part of the product development.

2.1 Experimentation and A/B testing

In software development, the term "experimentation" refers to many different techniques used to evaluate product assumptions (Schermann et al., 2018). These methods include techniques for eliciting both qualitative and quantitative data in a variety of ways, with the choice of method(s) depending on the intended purpose and context of the experiment (Lindgren & Münch, 2016). The purpose for experimentation is for the company to gain a more profound understanding on the related issue by analyzing and interpreting the experiment results, in order to ultimately support the decision-making when it comes to product decisions. One of the most common techniques in evaluating product hypotheses are online controlled experiments, known commonly as 'A/B tests', 'split tests', 'randomized experiments', 'control/treatment tests' or 'online field experiments' (Kohavi & Longbotham, 2017). Out of the many synonyms with slightly distinct semantics in each, A/B testing or split testing are the two most commonly used and widely known terms for the practice.

The underlying theory of controlled experiments dates back to 1920s and Sir Ronald A. Fisher's experiments at the Rothamsted Agricultural Experimental Station in England. Fisher's ideas widely transformed the agricultural experimentation, with many other fields of science quickly also adopting Fisher's statistical principles (Box, 1980), which to this day are considered as fundamental. Whether it be agricultural experimentation or online testing, the general idea of any experimentation technique is to transform assumptions into testable hypotheses, with a scientific method then applied to support or refute the hypotheses (Lindgren & Münch, 2016). In the context of the simplest form of A/B testing, the experimentation method is based around randomly assigning live users of the software to two different variants of the software which are being evaluated to determine the best performing one (Holmström Olsson et al., 2017). The two variants of the software are most commonly known as the "control" and "variant", with variant also sometimes referred as "treatment". In this setup, the users in the control group are seeing the existing version of the software, and respectively the users in variant group are using

a modified version of the software with a change or a different configuration introduced to it. (Kohavi, Longbotham, et al., 2009).

Whether the user sees control or variant is fully managed by a server and is thus entirely independent of the end user behavior, with additionally the users themselves being unaware of belonging to an A/B test group. (Xu & Chen, 2016). The users are split between groups in a persistent manner, meaning they continue to receive the same experience in every visit and differences in behavior between groups can be observed (Kohavi et al., 2014). In order to do this, the measured interactions in the software are instrumented to a set of measurable metrics. These metrics of interest could include for example things such as sessions per user or revenue per user, that can be used to inform decisions about the changes. After the metrics have been collected from the groups over a period of time, statistical tests are conducted on the collected data to evaluate whether there is a statistically significant difference between the two variants of the software. (Kohavi et al., 2014) Once the A/B test ends and the winning option is decided, the system frees the users from the A/B test and treats them in the same way, serving the new baseline to everyone (Dmitriev et al., 2016). Figure 2 below further showcases the general level view of the process and the experiment arrangement taking place in the execution of an A/B test.

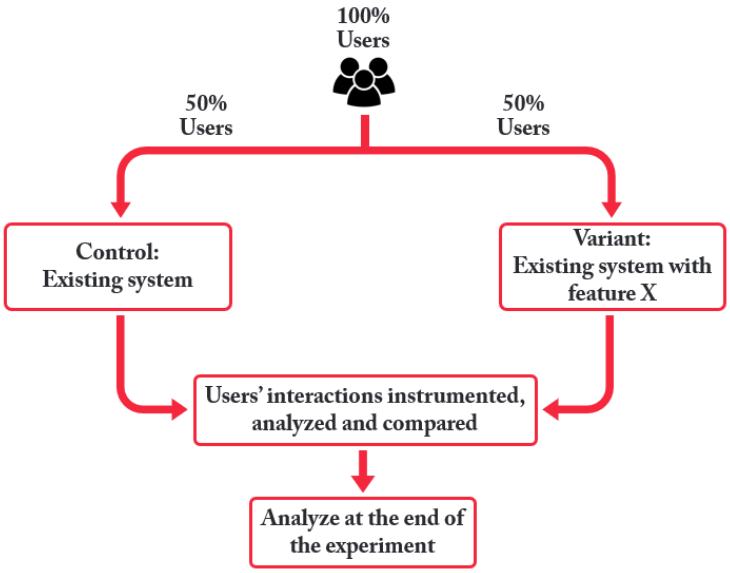

Figure 2. General level overview of an A/B test arrangement (Kohavi, Longbotham, et al., 2009, p. 149)

In essence, A/B testing is thus about testing variants of functionalities with customers in order to learn from customer behavior and make conclusions about the optimal software configuration (Holmström Olsson et al., 2017). The key thing to note in the experiment setup is the word "random". With the users randomly assigned to the groups and the experiment designed and executed correctly, the only thing consistently different between the groups is the introduced change. Any external factors during the experiment period such as seasonality, impact of other product changes and competitor or market moves are evenly distributed between control and variant. Consequently, any differences observed in the metrics between the groups can be attributed through statistical analysis to the introduced change. (Kohavi, Longbotham, et al., 2009; Fabijan et al., 2019) The causal relationship between the product changes and the measured changes in user behavior or business performance creates a more accurate understanding of what the customers value, and moreover provides sufficient evidence to draw conclusions on the impact of the change (Kohavi & Longbotham, 2017).

Controlled experimentation shifts the decision-making from subjective decision-making towards an evidence-driven process (Kohavi & Thomke, 2017). Moreover, running frequent A/B tests and using the results as an integral part of company decisions and product planning can have a substantial impact on the company culture (Kohavi, Longbotham, et al., 2009). Similar argument is made by Bakshy et al., who state that for some organizations controlled experiments stand at a central role throughout the design and decision-making process (Bakshy et al., 2014). The ability to access large customer samples and automatically collect vast amounts of data about user interactions and behavior on websites and apps through experiments has given companies a remarkable opportunity to evaluate many ideas rapidly and with great precision. The organizations utilizing controlled experiments are able to iterate rapidly, fail fast and pivot accordingly on their product development, which can be a significant competitive advantage when used correctly. In some areas of the software industry where controlled experiments are commonplace nowadays, rigorous experimenting should even be considered a standard operating procedure in order to be able compete with the competitors (Kohavi & Thomke, 2017).

The importance of controlled experimentation has been demonstrated a number of times by both the academia as well as the industry (Fabijan et al., 2017a). In the industry, mobile

applications, desktop applications, services and operating system features are regularly evaluated with A/B testing (Dmitriev et al., 2016). A/B testing is widely used especially by companies in the field of social media, search engines, e-commerce and online publishing (Machmouchi & Buscher, 2016). The methodology is also well adopted within companies in mobile gaming industry (Hynninen & Kauppinen, 2014) and by software-as-a-service (SaaS) providers (Lindgren & Münch, 2016). Simply put, A/B testing has begun affecting the development of all internet-connected software, and according to Holmström Olsson et al. (2017), has become mainstream in the industry with companies nowadays running frequent and parallel experiments.

The large internet companies of this era like Amazon, eBay, Facebook, Google and Microsoft are each running more than 10,000 controlled experiments annually to evaluate and improve their sites continuously (Kohavi & Longbotham, 2017; Kohavi & Thomke, 2017). Microsoft's practices and success with systematic large-scale controlled experimentation are acknowledged and studied in a number of academic releases, and Google reportedly has considered experimentation practically as a mantra, to the extent of evaluating almost every change that potentially affects user experience through experiments (Tang et al., 2010). The conceptually rather simple methodology of A/B testing can thus be an integral part of a company's product development toolbox. In some organizations, A/B testing is even considered as the single most important technique in learning about customer behavior and preferences (Holmström Olsson et al., 2017).

2.2 Continuous experimentation

New feature releases for a product can happen constantly and continuously on a software product. In order to evaluate the impact of each change and iterate to improve the features, A/B tests need to be accordingly run continuously. A term encompassing the practice of doing so, continuous experimentation, according to Fagerholm et al. refers to constant testing of the value of product changes as an integral part of the product development process, with the goal to continuously evolve the products towards high-value creation (Fagerholm et al., 2014). Ros and Runeson (2018) in turn consider continuous experimentation to refer to conducting experiments in iterations and testing continuously even the small changes. The main idea in

continuous experimentation is to have the mentality of constantly developing hypotheses on value creation and product changes, which are then tested continuously and validated through experimentation techniques such as A/B testing.

During the last decades, software advancements such as continuous integration and continuous deployment enabled companies to deliver changes to the software continuously on rapid iterations (Fabijan et al., 2018b), and continuous experimentation can be considered as an extension to these software trends (Ros & Runeson, 2018). The general feedback loop and lifecycle of A/B testing is based on three cyclic phases (Figure 3).

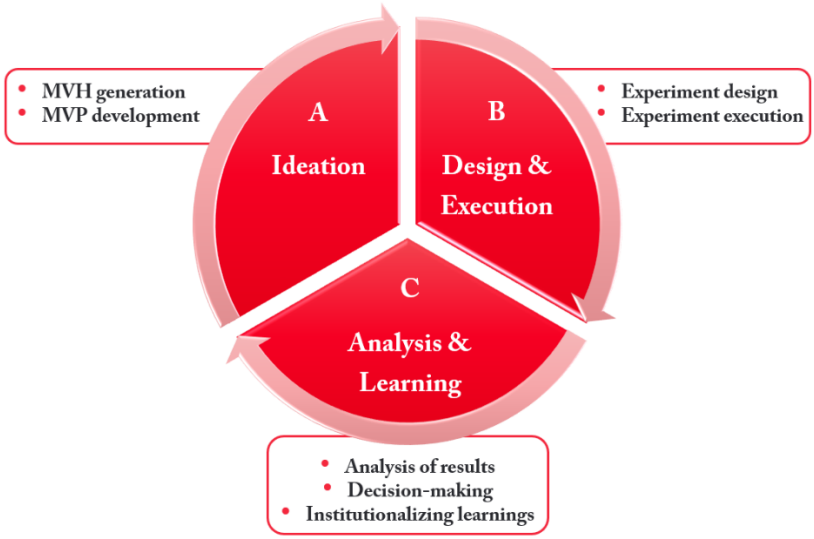

Figure 3. A/B test lifecycle (Fabijan et al., 2020)

Any experiment begins with the ideation of the test, which includes proposing changes to the product and developing minimum viable hypotheses (MVH), the simplest adequate-enough treatment and criteria to validate and trust the impact of the proposed changes. In ideation phase it is also established what is needed to develop in order to test the hypotheses, which often includes defining the minimum viable product (MVP), the adequate-enough version of the feature or change to validate the idea. Next, in the design and execution phase the configuration is decided and checked for any validity concerns, and the A/B test is launched live for users. After sufficient data is collected, the last phase consists of gaining a thorough understanding of the results and learnings through statistical analyses and examining the

outcome. The results are used in decision making, and more importantly, institutionalized, which means capturing and sharing the analysis results and learnings from the experiment with the relevant units and individuals in the organization. (Fabijan et al., 2019, Fabijan et al., 2020) In continuous experimentation, the learnings of the previous experiments are actively used in the planning of the next A/B test loop in order to effectively accumulate learnings. This drives the product management to efficiently conduct experiments and pursue continuous improvements that can be made based on the data from users of the software (Holmström Olsson et al., 2017).

The two most prevalent frameworks for continuous experimentation include the HYPEX model proposed by Holmström Olsson and Bosch as well as the continuous experimentation RIGHT model by Fagerholm et al. The HYPEX model, or "The Hypothesis Experiment Data-Driven Development" model (Figure 4), is a model developed for integrating feature experimentation with customers into the software development process. The HYPEX model is built on a systematic set of practices that shorten the customer feedback loop that seeks to ensure development effort is better in correspondence to the actual customer needs. (Lindgren & Münch, 2016)

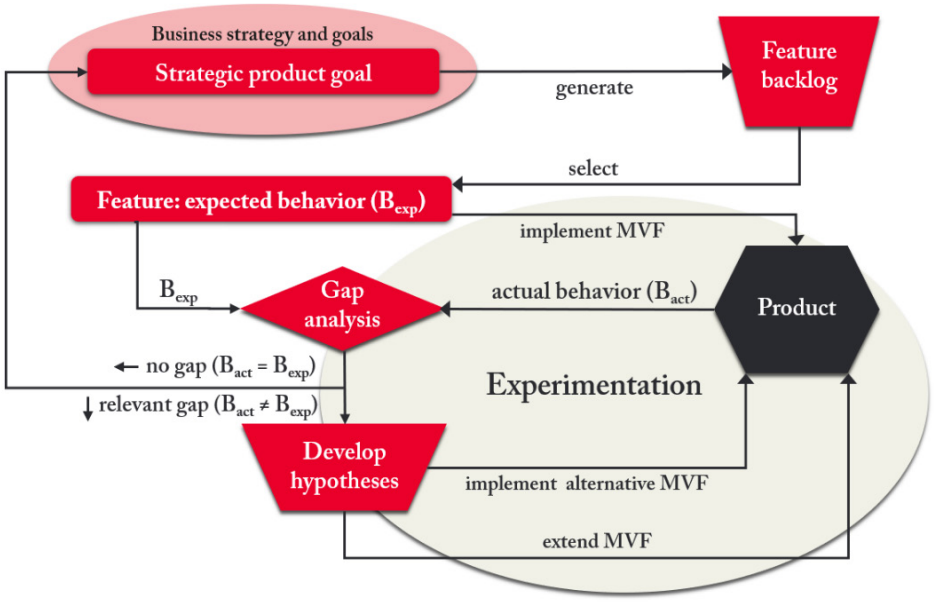

Figure 4. The HYPEX model for experiment driven development (Holmström Olsson & Bosch, 2014)

The generation of features is based on the strategic business goals as well as the in-depth understanding of customer needs. The features and the feature backlog consisting of ideas to be tested serves as a basis for selection of the next experiment. After a feature is selected from the backlog, the hypothesis is developed regarding the expected behavior, and the experiment is designed and instrumented. Entering the experimentation domain, HYPEX model introduces the concept of minimum viable feature (MVF), the smallest possible part of the feature that adds value to the customer. The MVF, essentially a slightly different take on the definition for an MVP, is then implemented for the experiment group in order to collect data about the actual behavior. The experiment is analyzed in gap analysis to determine how the actual behavior differed from the expected behavior stated in the hypothesis, based on which decisions are made about the full implementation of the feature. If there is no negative gap and the feature change is sufficient to achieve expected behavior, the feature is finalized and released for users. In case of a significant gap however, the team starts developing new hypothesis to explain the gap, tries to resolve the believed causes for the gap and launches a follow-up experiment with the new, modified feature. The third option is that the team decides to abandon the feature altogether based on the results. (Holmström Olsson & Bosch, 2014)

The gap analysis is central for the overall process. It ensures informed decision-making and promotes organizational learning through contemplating on what caused the difference between expected and actual user behavior. The model overall allows the product management team to align their efforts and strive for improving their understanding of customer behavior. Furthermore, the continuous experimentation and constant, quantifiable feedback provides a better focus for work in the team. (Holmström Olsson & Bosch, 2014)

The RIGHT (Rapid Iterative value creation Gained through High-frequency Testing) model suggested by Fagerholm et al. is consisted of "Build-Measure-Learn" feedback loops (Figure 5). The Build-Measure-Learn blocks structure the experimentation activity, and connect product vision, business strategy and technological product development through the experimentation. (Fagerholm et al., 2014) The process is supported by a technical infrastructure which enables lightweight releasing of MVPs, provides means for product instrumentation and supports the design, execution, and analysis of experiments (Lindgren & Münch, 2016).

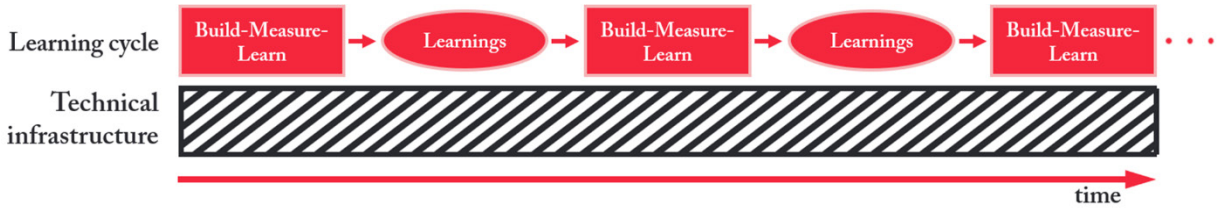

Figure 5. Continuous experimentation cycle (Fagerholm et al., 2017, p. 298)

Within each Build-Measure-Learn block, assumptions are derived from product strategy and previous experiments. The assumptions are used to formulate a hypothesis that can be systematically tested through an experiment, with the intention to gain knowledge regarding the derived assumptions. Next, the hypothesis serves as a basis to implement and deploy an MVP, in parallel with the experiment being designed and instrumented. The experiment is then launched for the users, and data is collected in accordance to the experiment design. Concluding the Build-Measure-Learn block, the data is analyzed and the results utilized on the strategy level to support decision making to pivot, change assumptions or decide to roll forward to deploy the feature or change. The results of each experiment are reflected back to the strategy and vision of the product, accumulating insight to be utilized in the next repeated Build-Measure-Learn blocks. (Fagerholm et al., 2014, Fagerholm et al., 2017)

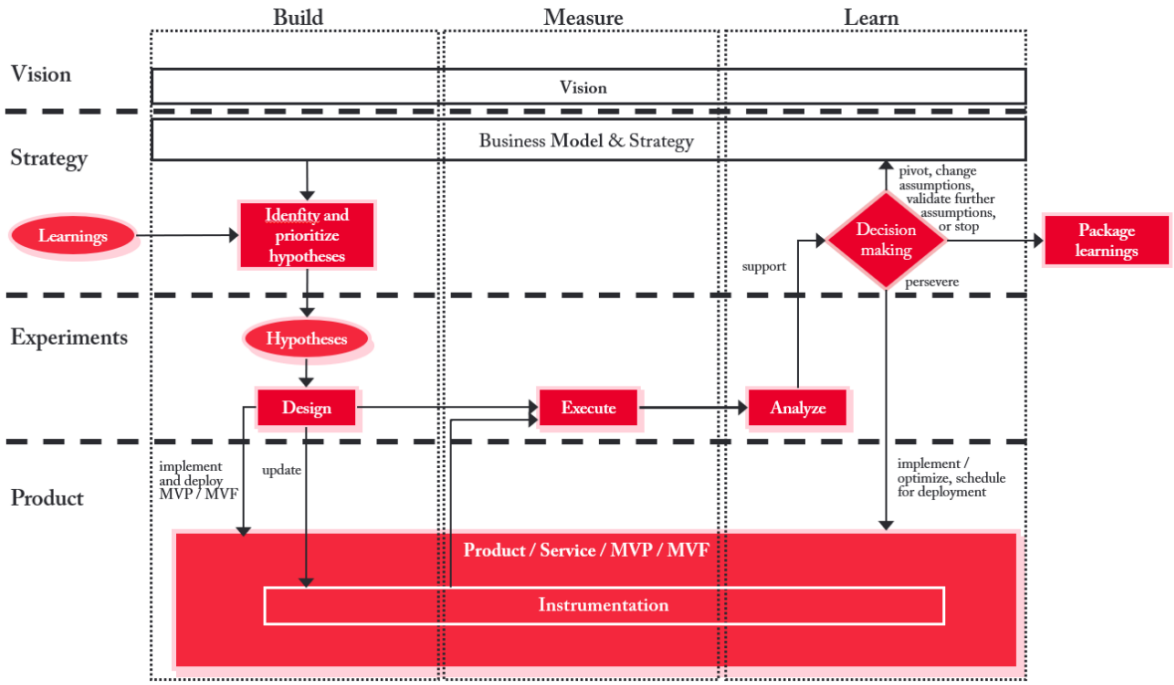

Figure 6. Build-Measure-Learn block (Fagerholm et al., 2017, p. 298)

Fagerholm et al. additionally define the typical roles and the technical infrastructure involved in conducting controlled experiments. Figure 7 displays an overview of the experiment infrastructure and the connections of the elements. The roles indicated especially can vary based on the type and size of the company: in a small company typically a small number of persons will handle the different roles, and one person may assume more than on role. In a large organization the roles can on the contrary be handled by multiple teams instead. (Fagerholm et al., 2014)

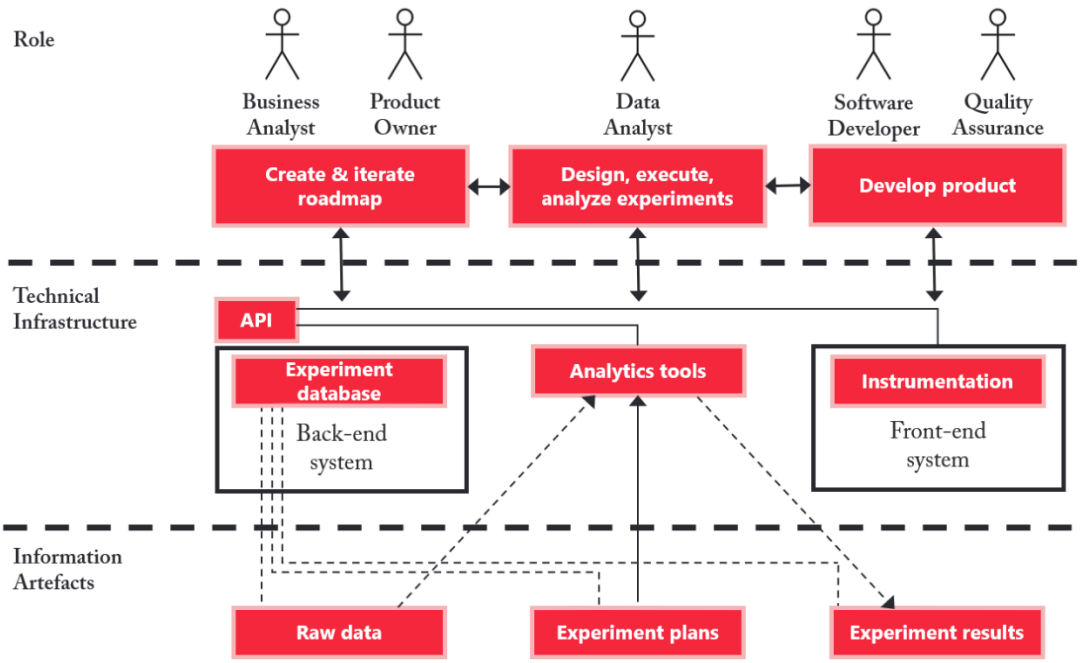

Figure 7. Continuous experimentation infrastructure (Fagerholm et al., 2014, p. 32)

A business analyst and a product owner, or a product management team, handles the creation of the A/B test roadmap and updates the roadmap iteratively based on results accumulated from other A/B tests. The product management works closely with a data analyst, who is responsible for designing, executing and analyzing the experiments. However, the design and execution of experiments can also be under product management's responsibilities depending on the organization and skillsets, with the data analyst mainly responsible for analysis of the tests. The development of a tested feature or change is handled by the developers, while quality assurance ensures no issues exist in the feature that could deteriorate and affect the experiment results. During the experiment and after it is finished, the data analyst employs a variety of tools in

accessing and retrieving the raw data in the back-end system, analyzing data and performance metrics as well as producing a report of the result. (Fagerholm et al., 2014) The motivations to analyze the experiment before it has concluded could include detecting any issues in instrumentation, overall sanity checking the data, and in some cases seeing if any preliminary learnings can be gained earlier that could be utilized in planning upcoming or follow-up experiments.

In order for the organization to conduct continuous experimentation, it needs to have the abilities to frequently release MVPs with suitable instrumentation, rapidly design and manage experiment plans, link experiment results with the product roadmap, and utilize a flexible business strategy. Furthermore, the organization must possess a proper understanding of what to test and why, coupled with skilled individuals to analyze the results and draw connections from the results to the context of the whole product and customer behavior. The organization must also be able to properly define decision criteria and act based on data-driven decisions. (Fagerholm et al., 2014) Kohavi and Thomke note that if a company develops the technical infrastructure and organizational skills to conduct continuous experimentation, it will be able to assess product decisions with a scientific, evidence-driven process relatively inexpensively. Without continuous experimentation, several breakthroughs might be missed, many failing ideas could get implemented and ultimately resources are being wasted on the development. (Kohavi & Thomke, 2017) Finally, Table 2 rounds up and lists the critical factors in successfully conducting continuous experimentation within an organization.

Table 2. Critical success factors in continuous experimentation (based on Fagerholm et al., 2014; Lindgren & Münch, 2016)

Domain	Factors
Development of features	• Integrating experiments to product development cycle • Developing and releasing MVPs regularly • Perform instrumentation to collect, analyze and store relevant data

Design and execution of experiments	• Assumptions need to be tied to high-level business considerations and prioritized based on them • Assumptions need to be transformed into testable hypotheses • Properly designing experiments based on hypotheses and previous results • Managing, iterating and updating experiment plans • Ability to analyze quantitative data reliably through statistical methods • If the experiment shows unexpected results, analyzing the reasons to explain the result
Updating product roadmap	• Experiment results used as input for decision making and follow-up actions • Iterating product strategy based on insight from experiments • Feedback loops pass relevant information from experiments to the organization

2.3 Executing product improvements through A/B tests

In the experiment-driven approach, business development and customer development are closely linked. The software development tends to focus on what to develop, and product roadmaps are seen as lists of untested assumptions that are systematically tested with experiments. In order for the customer behavior to be observed to determine if the software delivers value, it is necessary to easily deploy software. Agile software development methods allow quickly deploying and determining what to develop through the emphasis on incremental development process. (Lindgren & Münch, 2016) The very principle behind agile methods is that the "highest priority is to satisfy the customer through early and continuous delivery of valuable software" (Beck et al., 2001). Therefore they allow to quickly orient and make adjustments when the requirements change or any other need such as reprioritization presents itself. However, while agile methods allow reprioritizing which features to develop and implement, the methods themselves provide little guidance on what to develop to deliver value.

A/B testing and the related experiment-driven approach drives the development effort towards value delivery through testing and learning. The agile development methods focus more on the building aspects, while the experiment-driven approach focuses on the testing and learning aspects. Combining the agile methods with constant validation of product assumptions through A/B testing drives the development effort towards value delivery (Lindgren & Münch, 2016).

However, adopting experimentation is not trivial to companies. In addition to suitable software development knowledge and practices, conducting reliable and statistically robust controlled experimentation requires for example domain and data science expertise. Luckily, recent research has shed light and gathered knowledge from industry leaders on how to operate and conduct experiments. (Schermann et al., 2018) Starting with the motivation, product teams should be experimenting with their design decisions, parameter modifications, infrastructure changes and other types of features with the long term objective to learn about customer preferences and behaviors (Holmström Olsson et al., 2017; Fabijan et al., 2018a). Generating insight or understanding a relationship between specific actions ultimately helps improving and optimizing the product by reaching goals, such as delivering monetizable value to users (Lindgren & Münch, 2016; Holmström Olsson et al., 2017).

A/B tested things could more specifically include things such as user interface (UI) changes, backend algorithmic changes, new features or in some cases even new business models (Kohavi & Thomke, 2017). Depending on the tested change, the requirements for A/B testing it vary on the implementation side. On server-side the code changes happen on the backend only meaning that it only takes a server-side deployment in order to activate the changes, which can be deployed to take place instantaneously for the targeted users. This applies to most changes on websites, or features in application software that are backend driven. On application domain, this specifically means that the change can happen independent of an app update being released. Client-side changes, which mainly concern applications and include features that need to be controlled from the app itself, need to be however coupled with an app release. Hence, the changes are activated for users only after the update is released and the users has updated the application. Similar limitations naturally apply in introducing new client-side feature whether it is A/B tested or not. This has followingly led to new features always rolled out under A/B tests if possible, as it enables minimizing the risks involved with the new feature releases.

A/B testing allows to roll out the new feature to a small, randomized user group to evaluate it. If the feature has severe degrading effects on user experience or is downright faulty, the experiment population can be instantly directed to the baseline version of the application without the faulty feature, without having to go through another client release cycle which can take from days to weeks. The benefit of being able to prevent end users from being stuck with a faulty app for weeks has strongly promoted the "test everything" culture and use of A/B testing to evaluate changes. Certain limitations on A/B testing app changes however still exist, as a set amount of big changes on the application side can't be A/B tested. This includes cases where large changes have to be bundled together and it is impossible to separate them due to infrastructure changes or limitations, which in turn makes it impossible to A/B test them. (Xu & Chen, 2016) Because of some of the prementioned limitations, there are some differences in conducting A/B tests on applications compared to web that need to be considered in the process. Generally speaking, however, the A/B testing itself is conducted similarly on web and application domain.

When speaking of experiments in general, a distinguishing should also be made between regression-driven experiments and business-driven experiments. Regression driven-experiments are used to identify technical issues and are fundamentally a quality assurance technique, while business-driven experiments are mainly requirements-engineering techniques used to validate business hypotheses and evaluate impact of changes, i.e. the domain of A/B testing. (Schermann et al., 2018) Furthermore, the business-driven experiments can be further classified to feature introduction experiments and feature optimization experiments. The names are fairly self-explanatory; in feature introduction experiments a new functionality that hasn't previously existed in the particular context is added to the software, whereas in feature optimization experiments an existing functionality has been modified with the intention to improve a defined aspect of it. Both type of A/B tests are common in companies, and combined they help to better understand the effect of developed features, as well as what to develop next. Operationally, it should be noted that with feature introduction experiments a product team typically needs to invest time and effort into initial feature development, meaning that the feature introduction experiments must be planned further ahead, whilst feature optimization experiments are often be set up by changing existing feature parameters which can be performed quicker. (Fabijan, Dmitriev, McFarland, et al., 2018)

To optimize and assess many options simultaneously, especially feature optimization A/B tests are often run as univariable tests. Univariable test such as A/B/C and A/B/C/D tests have more than one variant group and thus assess more than one modification of a feature or variable at the same time. (Kohavi & Thomke, 2017) The benefit of univariable tests is that they help shifting towards the optimal configuration by forking changes. Impact of small feature improvements shouldn't be underestimated as they are inexpensive to implement and assess compared to development of a new feature, and yet they can yield a significant impact. The absolute impact of small improvements can in some cases exceed the impact of initial feature introduction, thus having major return-on-investment (ROI) (Fabijan, Dmitriev, McFarland, et al., 2018). Furthermore, Kohavi et al. point out that even negative experiments that degrade the user experience in the short term can sometimes be run due to learning value and long-term benefits (Kohavi et al., 2013).

To evaluate either regular or incremental product improvements through A/B tests, a company must have decided requirements that can be transformed into a solution (Hynninen & Kauppinen, 2014). The requirements themselves typically can't be defined in detail and a more based on educated guesses based on previous learnings. Naturally projects where requirements can be determined upfront still exist, but represent a very small percentage of all software projects (Wasserman, 2016). In addition to detailing the feature change or optimization procedures, the plan of an A/B test should be accompanied with a hypothesis of it improving a set of specified metric or metrics.

These specified metrics are often referred as 'overall evaluation criteria' (OEC), 'evaluation metrics' or 'performance metrics', and consist of quantitative measures of the experiment's objective (Kohavi, Longbotham, et al., 2009). Evaluation metrics used vary between web and application domain as well as based on the evaluated feature in case. Typical examples of evaluation metrics in web domain include conversion rate, repeat usage, customer retention, click-through rate or time to perform a certain task (Holmström Olsson et al., 2017; Kohavi & Thomke, 2017). Common examples of application domain evaluation metrics include conversion rate, customer retention or average session length (Hynninen & Kauppinen, 2014; Lindgren & Münch, 2016).

When A/B testing is introduced, many ideas will naturally start ending up being disproven and therefore it is also critical for the product team including designers, managers and product leads to be prepared to learn from the experiment and accept that most ideas fail to deliver what they were intended to do (Fabijan, Dmitriev, McFarland, et al., 2018). Moreover, the most valuable outcome of every experiment should not be whether the change made an impact or not, but the learnings that can be captured in a series of experiments. The mindset of accumulating learnings from the A/B tests by capturing and sharing them is vital in successfully improving the product through A/B testing. Especially the tests that do not have the desired impact and show unexpected outcomes should be shared and discussed. (Fabijan, Dmitriev, McFarland, et al., 2018; Fabijan et al., 2019)

Metadata such as screenshots, descriptions of functionality of the variations should be stored in addition to experiment hypothesis, results and impacts to metrics as well as final ship decisions. In a small scale, this could be handled through office tools and cloud drives, but for larger scale experimentation a dedicated ticketing tool in experimentation platform is necessary. A dedicated approach enabling to search across vast amounts of different experiments enables future experimenters to see what has already been tried, use the accumulated knowledge and apply it in a new context, prioritize new experiments as well as update metrics definitions to improve capturing customer value and missing details (Fabijan et al., 2019, Fabijan et al., 2020). Furthermore, at very large scale, much of the capturing of experiment learnings should be automated and well-integrated in the experimentation platform to reduce non-productive work (Gupta et al., 2018). In large-scale experimentation where hundreds of concurrent experiments are run with millions of users, Kohavi et al. note that quality assurance process should also be changed. Classical testing and debugging techniques no longer are feasible on their own due to the number of live variants of the system in production, and instead of heavy up-front testing, Kohavi et al. suggest utilizing issue alerts and post-deployment fixing. (Kohavi et al., 2013)

Overall the toolkit for executing A/B tests should cover product management tools, e.g. documentation tools and validation boards, technical infrastructure, e.g. feedback channels, data analysis tools, data storage capabilities, and optimally a platform to incorporate many of the tools in one convenient place (Lindgren & Münch, 2016). Lindgren and Münch also have found that in addition to supportive organizational culture and in-depth customer and domain

knowledge, good availability of technical tools and competence facilitates experimentation (Lindgren & Münch, 2016). The findings by Bakshy et al. support the fact that the availability of easy-to-use tools for analyzing the experiments is a major factor in adoption of A/B testing, and that attention should be paid to tools for designing, running, analyzing and automating experiments (Bakshy et al., 2014). Companies can either build the infrastructure for A/B testing either in-house or acquire it from a third-party provider. Notably, an increasing amount of third-party A/B testing tools are available in the market (Dmitriev et al., 2016). In addition to commonly known tools such Google Analytics and Adobe Target (Bakshy et al., 2014), there are also other companies specializing in A/B testing tools such as Apptimize, Optimizely and Mixpanel (Xu & Chen, 2016).

Typically, both small and large companies start with a centralized team for A/B testing and use third party tools to begin integrating A/B testing into their development practices (Lindgren & Münch, 2016; Kohavi & Thomke, 2017). However, findings by Lindgren & Münch signal that small startups are more likely to start with a broader and more integrated adoption of A/B testing from the beginning (Lindgren & Münch, 2016). Third-party tools and services allow easily to begin A/B testing, but when A/B testing becomes a corporate priority, the capabilities are further developed in-house and tightly integrated into company's other processes in order to scale things up and have the ability to customize the tools to better suit the company's needs. Similarly, A/B testing often is rolled out from central unit to the business units as the practices have been established. (Kohavi & Thomke, 2017)

2.4 Design of experiments and metrics

Experimental design is a major influencing factor in arriving at reliable and meaningful results from an A/B test. In the design phase of an experiment it is determined what is experimented on, what is the goal of the experiment, targeted population and traffic split among variants, as well as an estimating the duration of the experiment. (Xu & Chen, 2016) As described previously, A/B testing on any client-side changes requires coding, testing and shipping all variants for each of this kind of experiments with the app build. Thus, any code-side changes to the variants in experiments to be launched would require the next app version release. This has lead to parameterization being used extensively, as it allows flexibility in modifying and

creating new variants to upcoming A/B test without an app release. As new configurations can be passed to the client through parameters as long as the client understands how to parse the configurations, code-side changes and thus the need for app release is avoided. (Xu & Chen, 2016)

Experiment design encompasses many different key things to check both on the conceptual design of the experiment as well that the changes made, metrics gathered and expectations of the impact based on the previous two are logical. First, aspects of experiment validity are considered on the experiment design. Fabijan et al. (2019) provide a checklist of aspects which should be considered before launching an experiment, presented in Table 3.

Table 3. Experiment design analysis (Fabijan et al., 2019, p. 4)

• Experiment hypothesis is defined and falsifiable
• Experiment design to test the hypothesis is decided
• Metrics and their expected movement are defined
• Required data can be collected
• The minimum size effect and A/B test duration are set
• Overlap with related experiments in handled
• Risk associated with testing the idea is managed
• Criteria for alerting and shutdown are configured
• Experiments owners are known and defined

Each change for A/B testing should be introduced with a description of what the change that will be evaluated is (e.g. price point of a conversion offer) , who will see the change (e.g. new users after a defined date), what the expected impact (e.g. increase in conversion) is and how the impact is connected to overall product or business goals (e.g. increase in lifetime revenue). Numerical estimation also helps in prioritizing changes and evaluating later on the reliability of estimations as well as contemplating about how well are the customers being understood in that particular area. Most importantly however, it should be explained why a change is expected to have an impact on the defined metrics and understanding why the change is made in the first place. This way, a hypothesis combining the change in an experiment with its impact and

reasoning behind the expectation can be defined and formed, which can be explicitly be falsified or validated by the experiment. (Fabijan et al., 2019)

The data collected from the experiment should allow the tracked key metrics to be analyzed, which can be achieved most easily by creating a centralized catalog of log events and implementing those events in product. This ensures relevant analytics in the product to analyze the key metrics the organization uses to evaluate their A/B tests. Furthermore, defining the experiment duration and minimum effect size (Δ%) looking to be detected helps managing and planning the process. The running periods needs to be long enough that the experiment can detect the expected changes, but on the other hand is in the interest of product development to know the results early. The size of the effect the A/B test is looking to detect affects the duration of the experiment, as smaller changes will typically require more data and thus longer experiment duration for more users to get into the experiment. It is also good to note that the minimum effect size differs from the expected effect size. Fabijan et al. suggest that the latter may be difficult to predict and can differ greatly, whereas minimum effect size the organization is interested in is typically more consistent across experiments and determined by business goals and number of active users of the product. (Fabijan et al., 2019)

Any possible overlap between other experiments should be detected and coordinated to avoid with the A/B test targeting, as two or more experiments interacting may cause issues in the validity of the results if the tested changes are even remotely connected in changing behavior (Fabijan et al., 2019). Additionally, even though the experiments will run on a limited number of users, the risks involved with a potentially very bad experience for users causing business losses should be taken into account (Fabijan et al., 2019) and weighted in especially more uncertain and exploratory experiments. A common practice to mitigate the risk of a bad change is to target new experiments initially only for a small percentage of users, and then ramping up the percentage up gradually in order to speed up the data collection and consequently experiment running time. (Kohavi, Longbotham, et al., 2009; Kohavi & Longbotham, 2017) Furthermore, having a criteria for alerting or shutting down experiments – with either the experimenters or experimentation platform itself aware of them – helps mitigating most alarming situations where the experiment is unintentionally having a significantly negative effect. This is also why every experiment should have a defined individual or a group as the

experiment owners. The experiment owners are responsible for monitoring the experiment and for any operations, such as starting and stopping the experiment or acting on any alerts. Consequently, having several experiment owners for a single experiments ensures availability of one to contact in situations that may require more urgent action. (Fabijan et al., 2019)

The type of metrics the experiment uses is one of the other key factors in arriving at trustworthy and interpretable results. The metrics help discerning whether the effect of the change was desired or not and therefore guide shipping decisions, which is why good A/B test metrics are critical in order to make sound data-driven decisions. (Machmouchi & Buscher, 2016) Yet according to Dmitriev et al., one of the key challenges for organizations running A/B test is to select the OEC by which to evaluate A/B tests. The main difficulty is arriving at metrics that are in short-term able to predict the long-term impact of changes. Short-term improvements in metrics such as increase in revenue due to raising prices likely contradictingly reduces long-term revenue and customer lifetime value as users abandon. (Dmitriev et al., 2016) Kohavi et al. likewise advocate that good metrics should include factors that predict long-term goals rather than being short-term focused (Kohavi, Longbotham, et al., 2009). Another option for evaluating long-term impact of features can be to run long-term A/B tests, which however makes learning slower and experimenting less effective. (Dmitriev et al., 2016)

Metrics commonly try to capture abstract and subjective concepts such as success, delight, loyalty, engagement or life-time value, which represent goals for serving customers but have no standard way to formally define them. This creates and additional challenge in arriving at solid metrics. (Dmitriev & Wu, 2016) Organizations need to succeed in finding metrics that capture the essence of their business, which is why there is no one-size fits all solution available. Moreover, evaluating tests with ad-hoc metrics typically results in conflicts and unreliable, incomparable results (Fabijan et al., 2018b). When designing a metric, a profound awareness of changes it is supposed to measure it is needed. Seemingly even a very good predictor might still fail to pick up certain user behavior changes and consequently miss measuring an A/B difference. (Machmouchi & Buscher, 2016)

No single metric is without their weaknesses or loopholes, which can make the metric incorrectly move, or oppositely blind, for certain treatments the metric is not designed for.

Hence, designing a good metric system, i.e. a collection of metrics that measure treatment effects from various different angles, is important for gaining a comprehensive understanding of the implications of the experiment from the entirety. (Machmouchi & Buscher, 2016) By looking at a defined small group of metrics to evaluate the experiment, the decisions can be made more accurately and effectively. Additionally, if any data issues arise in the experiment, typically different metrics will respond and reflect on them, making them easier to spot. (Fabijan et al., 2018b) However, having a large number of metrics means that the odds of some metrics moving statistically significantly by chance is increased, and for some treatment effects some metrics can move also in seemingly contradictory ways. This easily leads to experiments cherry-picking the metrics that are most in-line with their expectations, leading to seemingly data-driven but in effect unsound decisions and interpretations. Therefore, defining the hypothesis and underlying assumptions in the experiment design is critical in avoiding this particular pitfall. To further address the issue, metric systems can be designed in a hierarchical way so that at the top are the most robust metrics which are defined at the user-level, have the fewest built-in assumptions and are usually the least sensitive. A system of metric capturing different scopes ensures experiments are not missing global effects that can't be captured by feature-level metrics, or the details that the more general level metrics fail to capture. (Machmouchi & Buscher, 2016)

The experiment and metric design are typically devised with the goal to get statistically significant and applicable learnings fast to speed up development. Sensitivity is a factor that refers to the amount of data needed for a metric to show differences between the groups. Considering sensitivity is important because more sensitive metrics allow detecting small changes sooner, thus shortening the time to run experiments and improving decision making agility. (Dmitriev & Wu, 2016) Sensitivity can be affected by three ways: increasing sample sizes, designing product changes that lead to larger differences in metrics, or reducing variance of the metrics. The simplest way is to increase sample sizes, which however means tying more users to the experiment. This puts an emphasis on product management to design product changes that can make clear impacts on the used metrics to make experiments more effective to run. (Xie & Aurisset, 2016)

On the metric design, simply counting the number of events such as queries, sessions or clicks doesn't result in sensitive metrics. To obtain more sensitive metrics, the counts can be normalized or capped. (Dmitriev & Wu, 2016) On metric capping, events otherwise without an upper bound are limited to a certain bound to reduce noise and variance generated by outliers. Two typical methods to achieve this are through truncation (any value >x treated as x) or functional transformation (use of log function to reduce effects of outliers). (Machmouchi & Buscher, 2016) Normalizing the metrics can be done with different denominators, i.e. dividing the metric per user or per session for instance. Similarly, bounding value between 1 and 0 by further increases the sensitivity. (Dmitriev & Wu, 2016)

Using different denominators on counting metrics can affect the sensitivity drastically. Depending on the denominator used, metrics can move in contradictory directions or not at all. According to Machmouchi and Buscher there is a trade-off, where generally the more fine-grained the denominator the higher the sensitivity of the metric is, but the chance for any anomaly movements is also increased. As an example, measuring "clicks per session" instead of "clicks" improves the sensitivity of the metric. Further fine graining to use "clicks per pages" further increases the sensitivity, but however leads to unintended contradictory metric movements. Based on this Machmouchi and Buscher further suggest using the metric with the most fine-grained denominator that doesn't cause it to move on its own. In a metrics scorecard, the metrics on the lower level should have more fine grained denominators and should be interpreted more cautiously if top level metrics already show movement. (Machmouchi & Buscher, 2016)

Dmitriev & Wu describe a metric evaluation framework to improve robustness of the metrics used in A/B testing. The framework and methods define and measure important characteristics of good metrics and proposes meta-metrics to evaluate metric sensitivity and alignment with sensitivity and user value. By having a data-driven approach to metric evaluation they report to have improved metrics substantially at Microsoft. The details of the framework are considered less relevant for the scope of this thesis, yet the framework illustrates important consideration points for metric design. Generally, users issuing more queries in an internet browser and therefore an increase in "queries per user" and "queries per session" are considered as an improvement in engagement. However, the framework suggests that these metrics rather

correlate with user value better in negative direction. Improving the quality of search results typically means that users do not have to reformulate queries as often, which leads to fewer "queries per session" and "queries per user". On the opposite, degrading search quality or making search results hard to examine leads to increase in these metrics as users have the reformulate queries more often to find what they want. (Dmitriev & Wu, 2016)

Moreover, questions on how to moderate a certain metric can be hard to discern. "Duplicate queries", i.e. the same query issued by the user twice in a row within the same session are fairly common in search engines. Some of the queries are real user-initiated queries, while some could be only due to log errors. The difficulty from metrics point of view is in deciding whether or not to deduplicate the duplicate queries by merging duplicate queries and taking a union of user actions from the data. Intuitively deduplicating will eliminate noise which should have positive effects on metrics. However, that will also result in losing signal from real user-initiated queries and affect for example "number of queries" and "number of clicks per query" that are part of many key search engine metrics. It's not clear to evaluate such tradeoffs, and there lies one of the difficulties faced in designing metrics. Ultimately in this example case, evaluation using the framework suggested that deduplicated versions of metrics performed better for most metrics regarding sensitivity and alignment with user value, but not in all cases. Using such evaluation frameworks can improve the quality of metrics used by the organization, and while the particular framework was applied in the area of web search by Dmitriev and Wu, according to them the framework is applicable to use in any domain. (Dmitriev & Wu, 2016)

Once an organization has developed an A/B metric system, all the experiments executed will rely on the metrics in arriving at trustworthy learnings and results. Thus, it is important for the metric system to be reliable, easily debuggable and that the metrics' movements can be clearly understood. Consequently, as a general guideline the metrics should be easily decomposable to the different signals that affect it, which is best achieved by constructing metrics of simple linear functions that combine basic attributes representing users' interactions. This ensures that the main drivers behind a metric movement are able to be pinpointed accurately if needed. (Machmouchi & Buscher, 2016)

2.5 Analyzing the A/B test results

In order to learn about the impact of A/B tests, the experiments need to be accurately analyzed using statistical tests when the experiment is being concluded. The analysis is concerned with identifying how the metrics that the experimenter expected to impact in the experiment actually reacted to the changes (Fabijan et al., 2018a). Since the assignment of users to experiment groups is random, the cause of the difference in metrics can be ruled out to a causal effect due to the change or random chance. Statistical tests and statistical significance in the results serves to rule out the random chance, making it possible to say based on evidence that there is a causal effect between the change and the differences in metrics. (Fabijan, Dmitriev, McFarland, et al., 2018) In other words, if the delta between the metric values for variant and control groups is statistically significant, a conclusion can be made that with high probability the introduced change caused the observed effect (Kohavi et al., 2014).

To understand the basic level of analyzing such tests, one first needs to be familiar with the concepts related to analysis. There are statistical principles behind how especially the statistical significance of A/B tests is determined, but this chapter mainly adopts a managerial perspective to understanding necessary basics. The following Table 4 summarizes the basic concepts involved in discussing A/B test analysis.

Table 4. Basic concepts of A/B test analysis

Concept	Description
Population	All users that can potentially be impacted by the change (Xie & Aurisset, 2016)
Sample	Number of users that are part of the A/B test (Kohavi, Longbotham, et al., 2009)
Factor	Controllable experiment variable, the change which is different between A/B test groups (Kohavi, Longbotham, et al., 2009)
Variant	A group where the factor is different from control group. Multiple variants can exist within same test (Kohavi, Longbotham, et al., 2009)

Experimental unit	Entity over which metrics are calculated before averaging over the entire experiment for each variant, commonly unique user ID (Kohavi, Longbotham, et al., 2009)
Null hypothesis (H_0)	Hypothesis that the variants do not perform differently and there is no causal difference due to the change (Kohavi, Longbotham, et al., 2009)
Confidence interval	Range of plausible values for the size of the effect (Kohavi, Longbotham, et al., 2009)
Confidence level	Share of the confidence intervals correctly containing true mean value (Kohavi, Longbotham, et al., 2009)
Power	Probability of correctly rejecting H_0, detecting a difference when it indeed exists (Kohavi, Longbotham, et al., 2009)
P-value	Observed significance level, probability that the observed results would be higher or equal rather due to random chance (Fabijan, Dmitriev, McFarland, et al., 2018)
Standard error	Deviation of the sampling distribution, accuracy of the sample in representing the population (Kohavi, Longbotham, et al., 2009)

The statistical tests, such as t-tests, involved in analyzing the test evaluate whether one of the variants is statistically significant from control, enabling to reject null hypothesis. Unit of analysis in A/B tests is most commonly user IDs, which allow analyzing behaviors associated with users in different A/B test groups. (Bakshy et al., 2014) When analyzing different metrics depicting changes in user behavior, the confidence level for statistical significance is usually set to 95%. The 95% confidence level implies that statistically 5% of the time it is incorrectly concluded that there is a difference, when there actually wasn't. This type of false conclusion is also known as type I error. Power of the experiment analysis is commonly desired to be around 80-95%, although it can't be directly controlled. However, raising the confidence level and decreasing standard error increases the power of the experiment and decreases the chance of a type II error, where the null hypothesis is retained when in reality it being false. (Kohavi, Longbotham, et al., 2009) If the analysis shows a variant outperforming the control on the set level of significance, user preference can be inferred to be the variant option (Kharitonov et al., 2017).

Experiment methodologies such as A/B testing typically rely on means, which are assumed to be normally distributed in the sample. According to Central Limit Theorem, the mean of a variable has an approximately normal distribution when the sample size is large enough. Depending on the metrics used, the sensitivity and therefore also the skewness of the distribution changes, requiring different amounts of users in the test for the standard error to decrease. Many metrics in online experiments have long tailed distributions and can be quite skewed, which may require a higher lower bound for the sample size before one can assume normality. (Kohavi et al., 2014)

There are different formulas to estimate required sample sizes. For example, van Belle (2002, p. 31) suggests a following rule of thumb approximation formula ($n = 16\delta^2/\Delta^2$), where n is sample size of each variant, δ^2 the variance of metric and Δ the size of difference looking to be detected. The formula is assuming confidence level is 95% and desired power is 80% for the experiment. (van Belle, 2002) Alternatively, Kohavi et al. (2014, p. 8) suggest a rule of thumb formula based on skewness coefficient ($n = 355 \times s^2$), where s is the skewness coefficient of the distribution of the variable X defined as ($s = E[X - E(X)]^3/[Var(X)]^{3/2}$). Evaluating sample sizes can show for instance that capping metrics can greatly increase sensitivity and reduce the skewness so that the average converges to normality faster. (Kohavi et al., 2014) Improving sensitivity of an experiment by various variance reduction techniques related to sampling techniques and metrics can additionally reduce required sample size for A/B tests significantly (Xie & Aurisset, 2016; Kharitonov et al., 2017).

As the basis of an A/B test is dividing users randomly to the different A/B test groups and the statistics assume random sample of end users on each variant, randomization quality is critical in an experiment (Kohavi, Longbotham, et al., 2009). Randomization quality isn't something concerned on analysis in experiment to experiment basis, but rather more holistically in the experiment system. Fabijan et al. advocate validating experiment system quality through a series of A/A test when introducing a new randomization algorithm or new metrics. (Fabijan et al., 2019) A/A test is a test setup where the different groups are served exactly the same experience. A/A tests, sometimes called also a null tests, can be used to collect data and assess its variability for power calculations and also to test the experimentation system. (Kohavi, Longbotham, et al., 2009) In particular, the distribution of p-values should be close to uniform in an A/A test

(Fabijan et al., 2019). If the system is working intendedly, null hypothesis should be rejected roughly 5% of the time when a 95% confidence level is used in the analysis (Kohavi, Longbotham, et al., 2009). Ensuring A/A alternatives show close results keeps type I error low and eliminates possible biases in the experiment system (Kharitonov et al., 2017). There are further multiple techniques and guidelines documented for ensuring this (see e.g. Kohavi, Longbotham et al., 2009; Xie & Aurisset, 2016). Making sure the metrics and the system works expectedly allows the analysis to focus on how product changes affected the metrics, rather than on metrics' sensitivity to external factors (Dmitriev & Wu, 2016).

When proceeding to analyze an A/B test, a certain number of practices should be carried out to ensure trustworthiness of the analysis. First, it must be ensured that no data quality issues are present in experiment and that outliers in the data skewing certain metrics are treated by e.g. capping (Kohavi & Thomke, 2017). Certain data quality metrics can be used here to ensure the experiment results can be trusted. Moreover, in terms of trusting the results, the analysis must determine that the experiment has sufficient power to be looking at the results yet at that point. If there are no data issues present and the experiment has enough data the analysis can be carried out in detail. (Fabijan et al., 2019)

In the analysis, contextual awareness is important. For instance, to evaluate the real sustained impact of a new feature or change to the product, novelty effects should be excluded. Certain new features can increase usage of the feature at first, with the usage however over time regressing. Analysis over a time horizon helps determining whether the change made had a sustaining, actual long-term value increasing impact. (Fabijan et al., 2019) Another concern are other events in the world and seasonality affecting user behavior. Although the effects of these are distributed evenly between the groups, user might be behaving different to normal in the particular time, i.e. what works during that period of time might not work otherwise. In such cases comparing cohorts in the post-period if possible can have less bias and provide a more accurate picture. (Dmitriev et al., 2016)

"Any figure that looks interesting or different is usually wrong" (Ehrenberg, 1974, p. 543). The Twyman's law is equally a highly applicable generalization when looking at analysis results. By nature, humans are inclined to resist and question negative results to a feature that was thought

to have success, so analysis is drilled deeper to find the cause. When the effect is strongly positive however, the inclination is to celebrate, rather than drilling deeper looking for anomalies. (Kohavi et al., 2014) Hence, Twyman's law is a good rule of thumb to remember when looking at test results. Overall failing to recognize any possible violations in the experiment design and analysis can steer the product team into making wrong conclusions, possibly actually causing harm to the business (Kluck & Vermeer, 2017). In the long run, poor data or analysis can effectively be worse than no data, by blinding decision-makers with pseudoscience (Fabijan et al., 2019).

On the subject of drilling deeper to the results, the decision on which variant should be shipped based on metrics should be only one concern of A/B testing. The experiments should also regardless be analyzed in depth to discover more insight that can be used to gather learnings, iterate on the feature and provide ideas for new features. (Fabijan et al., 2019) For instance in mobile applications such as games, it is not unusual to see a positive lift on one platform and a negative impact on the other. Thus, it is important to examine per-segment impacts in addition to the overall impact only. (Xu & Chen, 2016) Sometimes the critical insights for reliably making a decision on the winning variant can similarly be buried and discovered through analyzing different homogenous segments of users (Fabijan et al., 2018a).

Particularly, when the overall results of the experiment yield no statistically significant difference between variants a deeper analysis may uncover changes in different segments. However, it could also mean that there truly is no impact from the different variants, or simply that the experiment did not have sufficient power to detect the change. Sometimes even the result showing no significant differences validates making the change, for example in cases where the change allows making other changes down the line or the change being a requested feature by the users, and it provenly doesn't hurt anything. (Kohavi, Longbotham, et al., 2009) Continuously also collecting qualitative customer feedback for instance through surveys or emails can also compliment A/B testing by providing additional context and insight. Qualitative data can aid in interpreting ambiguous quantitative movements and results, understanding customer behavior to come up with new hypotheses to test as well as improving the metrics that are used. (Levin, 2014; Fabijan, Dmitriev, McFarland, et al., 2018)

2.6 Benefits of A/B testing

The internet and software industries by nature have distinct advantages in how organizations can utilize experiments. Companies can effortlessly introduce numerous variations of the service without substantial engineering or distribution costs, and conveniently access a large random sample of users to observe their behavior to make guided decisions. In a number of ways, it can be claimed that experimentation with internet services is easy. (Bakshy et al., 2014) The practice of testing product ideas is not new. The widespread adoption of web-based systems has however enabled a much wider range of techniques to experiment with customers and perform statistical testing instead of relying on anecdotal feedback. Moreover, the experiments can be administered automatically without involving customer actions or even agreement. (Schermann et al., 2018)

Several technological trends and advancements such as continuous deployment, increasing use of software-as-a-service as a delivery model or DevOps development practices support the ability to rapidly deploy software to customers and the ability elicit feedback in the form of product usage data. The ability to deliver product changes incrementally and the closer integration of product discovery, product validation, and delivery activities in the development allow taking into account the observations from experiments when making product decisions. Many different methods are in fact available for conducting the experiments with customers, including A/B tests, multivariate tests, product analytics, landing pages, fake door tests, problem interviews, solution interviews, and tests with wireframes, mockups, or Wizard of Oz minimum viable products. The selection of method depends on the type of hypothesis looking to be tested as well as the purpose and context of the experiment. (Lindgren & Münch, 2016) When it comes to estimating changes in user behavior, seasonality is one of the main confounding factors. The difference in the change observed and perceived trends over time in the experiments can't be entirely attributed to changes in user behavior without the random allocation to followable and controllable groups (Dmitriev et al., 2016), hence A/B testing steps in to the picture.

The most well-known benefit of A/B testing in software product development are incremental product improvements (Fabijan et al., 2017a). As previously highlighted, even tiny changes can

yield great impacts, despite people commonly assuming that greater investment will results in larger impact. Although the business world glorifies disruptive big ideas, in reality most progress is achieved by implementing continuous minor improvements. (Kohavi & Thomke, 2017) Many organizations can have many ideas, but the ROI for them is often unclear. As also stated earlier, many of the thought improvements – regardless whether they are larger or smaller – in fact may yield negative or no returns. For instance at Microsoft, only one third of the ideas tested improved the metrics as they were designed to (Kohavi, Crook, et al., 2009). By A/B testing the ideas and new feature releases, the company gains statistically significant evidence which can underpin the actual impact of them. Thus, A/B testing a powerful tool in providing guidance to the value of ideas, which can also guide further decisions on where to invest for greatest return-on-investment. (Kohavi, Longbotham, et al., 2009)

The impact of A/B testing however extends beyond direct revenue gain. It fundamentally changes how research and development (R&D) is planned, prioritized and measured in the company. Fabijan et al. identify benefits on three different levels, including portfolio, team and product level. On the portfolio level, A/B testing supports accurately identifying constituting factors for customer and business value, and how the work of product teams contributes to it. On the team level, A/B testing can fundamentally change and improve how R&D is executed and how the product roadmap is planned. This lastly impacts development on the product level where decisions on product functionality, complexity, quality and infrastructure are made. (Fabijan et al., 2017a)

Generally, there are multiple different cognitive biases that take place during the decision-making process that can affect the rationality of the decisions made. These human biases can affect decision-making in the negative way by basing decision on fallacies rather than rational evidence. (Manjunath et al., 2018) The data-driven approach of A/B testing can provide later verifiable reasoning and transparency to the decision-making and the criteria the decision was made based on. Moreover, in order to know where to invest and prioritize development effort in the product level, it is critical for companies to measure how customers react on software product changes to link product development and business aspects to verify that the product is moving towards a direction with higher business value. (Rissanen & Münch, 2015) The benefit of A/B testing is that is focuses on what customers do rather than what they say. This allows

improving the understanding of the value that the product provides and how different features affect product use in a certain context (Fabijan et al., 2017a) as well as quantifying hypotheses and analytically deriving the answers, effectively enabling the use of hypotheses to steer the development process (Rissanen & Münch, 2015).

On the team level, the experiment approach on development affects team activity planning. Knowing what type of changes and in what areas improved the key metrics in the past and by how much allows the team to generalize and organize their work to focus on type of changes that are known to have the most beneficial effect on the product. Experimenting can also benefit in defining performance goals for the team. By setting goals to improve certain key metric or metrics provides a clear focus on changes to make and a way of tracking the progress by monitoring the product metrics and the effect of statistically significant changes on them. Speaking of metrics, A/B testing can also help identifying "leading" and "lagging" measures. (Fabijan et al., 2017a) Leading measures are metrics that are predictive and influenceable, changing within a short period after a change occurred. Lagging measures on the other hand are metrics that measure the goal but change with a delay, typically as a consequence of several leading measures changing. Lagging measures are characterized with a performance gap and a time (e.g. increase revenue to X by Q4). (McChesney et al., 2012)

Executives typically fixate on lagging measures even though there is no way to directly affect them. However, it is much harder to come up with lead measures that able to predict how certain things play out over time. (McChesney et al., 2012) Still, companies should instead try to focus in setting product team goals on lead measures and avoid lagging indicators for performance goals. Without controlled experimentation, changes to metrics and correlations between metric movements are much more challenging to detect in complex online environment due to large variance, seasonality effects, and other factors, making establishing a clear relationship between leading and lagging indicators difficult. (Fabijan et al., 2017a)

In their extensive article, Levinthal and March (1993) have examined the topic of organizational learning. In the article, they found out that learning typically tends to sacrifice the long run to the short run. Effective learning requires exploration to happen, but at the same there is an issue that exploration starts at some point to be restrained by increased exploitation. Moreover,

organizational learning tends to oversample successes and undersample failures. Learning processes tend to eliminate failures, which is accentuated by the way learning produces confidence and confidence further produces favorable anticipations. (Levinthal & March, 1993) Consequently, A/B testing can be seen to aid in organizational learning as it promotes the organization to continuously explore through testing and new hypotheses, as well as through test results highlight also the occurring failures.

Some of the experiments more directly inform decisions regarding parameters they involve, but other well-designed experiments can instead aim to provide more lasting and generalizable knowledge through broader, longer-term influences on beliefs of designers, developers and managers (Bakshy et al., 2014). Furthermore, evaluating users' reactions to different changes not only provides beneficial insight into customer preferences with the service, but also allows the identification of segments that benefit from a feature or estimating the impact which the changes have on customer behavior on larger scale (Kohavi, Longbotham, et al., 2009; Hynninen & Kauppinen, 2014). These insights can lead to a virtuous cycle of improvements in features and potentially to a degree, also better personalization (Kohavi, Longbotham, et al., 2009). Fabijan et al. further point out that by experimenting on different product development teams across multiple products in the portfolio, the company can learn what is valuable for the business and what is valuable for the customers across the product portfolio. By validating hypotheses in multiple experiments across multiple products, the company can potentially build generalizable knowledge that is directly actionable to also other products in the portfolio as well as in further products and development projects. (Fabijan et al., 2017a)

As one other major benefit, experimentation and the validation of ideas through A/B testing reduces the risk of deploying software changes and additionally accelerates the cycle of innovation in data-driven companies (Kohavi, Longbotham, et al., 2009; Fabijan et al., 2017a). The risk management of releasing changes through A/B tests mainly comes from the ability to verify that the features deliver increased business or customer value, but also from rolling the changes to a smaller percentage of users and the ability to quickly roll back to baseline. Although the changes to products are typically well tested through quality assurance, this effectively decreases the exposure and magnitude of harm from possible defects or otherwise negative changes by limiting the impact to only a defined number of customers. As a side benefit, missing

defects in quality assurance and building heavy virtual testing infrastructure isn't as business critical, as the risk of defects is lower and moreover rolling back to baseline for those users is possible through simple configuration change without the need of a client update. (Fabijan et al., 2017a) In most of the cases, "breaking the product" still actually happens through implementing features that diminish the value of the product. These can be often be deceptively simple cases, such as UI changes that are thought to improve the user experience but fail to do so due to incorrect presumptions. (Rissanen & Münch, 2015) Managing the feature risk by continuously measuring new features to the product through A/B testing is a relatively easy way to prevent this.

Furthermore, as product instrumentation and product data are increasingly relied on in measuring the product performance overall, A/B testing can help in verifying that the also otherwise tracked metrics are up to date and react as it is expected from them. By performing A/A tests or observing results from the tests where the outcome is highly expected can be used to verify the instrumentation quality and logic by comparing the expected and actual outcomes. (Fabijan et al., 2017a) Lastly, companies utilizing A/B testing typically see increased velocity in innovation largely thanks to many of the other discussed benefits. The innovation cycle can be seen to be benefitted through A/B testing lowering the cost of testing and experimental failures, which encourages increased experimentation of innovative ideas. Moreover, it is suggested that failing fast enables knowing what ideas didn't work and helps providing necessary course adjustments to identify other more successful ideas to be proposed and implemented. (Kohavi, Longbotham, et al., 2009)

Thus, the impact of A/B testing extends to many areas on top of the typically conceived use of deciding whether or not to deploy a change (Fabijan et al., 2020). A/B testing allows rapidly making evidence-based decisions which help guiding the evolution of the software, reaching product goals and accelerate learning (Ros & Runeson, 2018). Experiments can point the team in the right direction when answers aren't obvious, people have conflicting opinions or are uncertain about the value of the idea. Kohavi and Thomke followingly summarize a view on the learning aspect of A/B testing: "if you really want to understand the value of an experiment, look at the difference between its expected outcome and its actual result. If you thought something was going to happen and it happened, then you haven't learned much. If you thought

something was going to happen and it didn't, then you've learned something important. And if you thought something minor was going to happen, and the results are a major surprise and lead to a breakthrough, you've learned something highly valuable." (Kohavi & Thomke, 2017, p. 82) Finally, Table 5 summarizes the main benefits of A/B testing discussed in this chapter.

Table 5. Benefits of A/B testing in portfolio, product and team level (based on Fabijan et al., 2017a, p. 24)

	Benefits	Guidelines to achieve
Portfolio	Value discovery and validation	(1) Customer and business value are hypothesized on portfolio level (2) Hypotheses are evaluated on multiple experiments across multiple products (3) Measurement of the value is formalized in terms of leading metrics
Product	Incremental product improvements	(1) Experiments are run and analyzed as part of development cycle (2) Statistical differences between variants are determined (3) Variants with improvements to key metrics are deployed
	Ensuring product quality	(1) Risky changes are initially deployed to a low % of users (2) Product changes that degrade key metrics are rolled back to baseline
	Stabilizing product complexity	(1) Product improvements with no impact on key metrics don't get deployed (2) Reverse experiments are run to remove features that have become obsolete
	Product instrumentation quality assurance	(1) A/A experiments are run to identify noisy instrumentation (2) Experiments with highly expected outcomes validate instrumentation quality
	Accelerate learning and innovation	(1) Experimentation is utilized to test new ideas actively (2) Running experiments that provide valuable, generalizable knowledge
Team	Team activity planning	(1) Value and impact of different ideas in found out (2) Changes / features that improve key metrics are shared among team (3) Team generalizes learnings to identify feature areas to prioritize and improve
	Defining performance goals for the team	(1) Measuring amount of impact from changes over a period allows setting realistic goals for next period to improve the key metrics

2.7 Limitations and challenges in A/B testing

As demonstrated in the previous chapter, there are number of benefits that an organization can realize through utilization of A/B testing. However, there also exists certain challenges and limitations that constrain A/B testing and its implementation. These can vary from organizational challenges to inevitable restrictions on the methodology itself that constrain running the tests in operational level or can distort the results if left unacknowledged.

First, there are some obvious prerequisites and limitations for a company to run A/B tests. The organization has to have the adequate domain knowledge and technical infrastructure in place to have the capability of running the tests in the first place, as well as the capability to iteratively improve the product and ship the releases at adequately short time intervals for the customers. (Hynninen & Kauppinen, 2014) Other key challenges are related to transforming the organizational culture and decision-making processes to use the test results and learnings to their advantage along with ensuring the collected customer and product data are carefully analyzed with robust practices (Lindgren & Münch, 2016). As such, investments in experimentation platform, instrumentation of existing products and expertise such as data scientists and data engineers are likely required (Fabijan et al., 2017a).

Assuming the company has the necessary prerequisites and capabilities for A/B testing, there are a number of limitations that need to be understood. The first, previously emphasized key area is the selection of metrics by which the experiments are evaluated with. Getting the metrics right takes thoughtful consideration and detailed scrutiny to understand both the business goals as well as trade-offs in the metrics (Kohavi & Thomke, 2017) as was demonstrated in chapter 2.4. Although usually the importance of clearly defined and suitable metrics is acknowledged well by practitioners, their identification remains often challenging (Lindgren & Münch, 2016). Some metrics might be able to tell variant is better and by how much, but miss the context on actually making conclusions on "why". Primacy and newness effects are another concern when measuring new features or changes during a short experimentation period, which can lead to misleading conclusions about sustained value of the feature. Similar risk is involved with the common pitfall of the metrics failing to predict and measure long-term goals. (Kohavi, Longbotham, et al., 2009) Moreover, otherwise suitable metrics might fail to align with

business level KPIs. In other words, a team might optimize for certain outcomes using metrics they can influence, but without verifying the relationship between the metrics used and the high-level business KPIs. (Holmström Olsson et al., 2017)

Getting the metrics right on its own isn't enough either – the results from experiments need to reliable and trustworthy. Experimentation pays no benefit no matter how good the metrics are, if people don't trust the experiment results. Validating the experimentation system e.g. through A/A tests and setting up automated checks and safeguards is necessary to prove that everything adds up, meaning about 95% of the time the system correctly identifies no statistically significant difference in the A/A tests run. (Kohavi & Thomke, 2017) A/A tests that consistently show non-uniform distribution of p-values are a symptom of serious issue in the experimentation system, which requires debugging the experimentation system to discover the root cause. Constantly failing to decide based on experiment results can be another symptom of issues in metrics or trustworthiness of the system. When a used key metric has statistically significant change in the end of an experiment but consensus among stakeholders can't be reached about, it signals the experiment results aren't trusted in the organization. In such cases people might argue with some thinking the result was good while others arguing it was not, by further trying to convince against the statistically significant results with far-fetched explanations or counter examples in the data. (Fabijan et al., 2019)

Additionally, there exists a number of biases related to how the experiments are run that can cause A/B tests in an otherwise valid system to show invalid results. Two of the common biases to acknowledge related to A/B tests are survival bias and selection bias (Dmitriev et al., 2016; Fabijan et al., 2017a). In World War II, there was a decision to add armor to bombers with military concluding to add the armor where the planes were hit the most based on recordings made after battles. However, the identified locations were actually on the contrary the worst places to reinforce the plane with armor. Bullet holes in the planes were in fact almost uniformly distributed, whereupon armor would've needed to be added to the places where there were no bullet holes in the recordings. This at first counterintuitive conclusion comes from the fact that the bombers hit in those places never made it back to take recordings of. (Denrell, 2005) In an A/B testing context, similar survivorship bias can take place in long-term experiments. In an extreme, users who don't like the feature abandon, and the cohorts are left with users that either

like the feature or don't care. If users abandon at different rates between control and variant, the remaining surviving population is different, and the conclusion made based on the results can be completely wrong if the abandonment is not understood. (Dmitriev et al., 2016)

When choosing which cohorts to analyze and by which granularity to analyze, selection bias can take place. Choosing what to analyze can lead analyzing samples that are not representative of the effect of the change to the population. For example, to avoid the previously mentioned primacy and newness concerns, the long-term impact on all users should be measured through changes at the end of the experiment period when targeting the A/B test to both old and new users. The users who appear in that period consist that way of old users who have had time to adapt to the feature and get past the newness excitement, as well as new users that have recently joined who were not exposed to the old experience before and such do not have primacy effects. (Dmitriev et al., 2016; Xie & Aurisset, 2016) Otherwise, it will be difficult to discern whether a movement in metrics is simply due to a newness excitement, or whether it is caused by the new better experience itself (Xie & Aurisset, 2016). This further implies that in order to arrive at trustworthy results on changes where primacy and newness effects are a concern, the experiment needs to be run for multiple weeks. (Kohavi, Longbotham, et al., 2009)

While experimenting on new users only avoids primacy and newness effects and thus additionally removes the need to long the experiment longer (Kohavi, Longbotham, et al., 2009), it leads to an uncertainty how old users will react to the change when it is rolled out, as old users and new users may often react to a change very differently (Dmitriev et al., 2016). There is however always also a risk involved with showing old users a different experience and then reverting again back to the old experience, in case the change didn't perform better than the control as often happens. With experiments consistently run on old users, the experience changing all the time due to various changes that get reverted may degrade the overall user experience. Reusing control and treatment populations from one experiment to another might also lead to carryover effects, in which people's experience in an experiment alters their future behavior (Bakshy et al., 2014). When experimenting mainly on new users only, the same users aren't able to subsequently be part of many experiments. Moreover, as new users are not familiar with the old version, they often don't notice having being part of experiments. Yet A/B testing

only on new users slows down the pace of innovation, as the usable sample of users is more limited. (Xie & Aurisset, 2016)

With A/B testing being highly iterative, follow-up experiments optimizing the same feature are often run rapidly based on the results. This necessitates changing or launching new experiments. However, changing live A/B test often easily leads in statistical inferences that invalidate the results. Running follow-up experiments requires collecting another fresh sample of users and can require some effort from development perspective, with experimental logic sometimes mixed in with application code. (Bakshy et al., 2014) Moreover, A/B tests are often targeted based on more attributes than new or existing users, for example in application domain by platform, operating systems or app versions. In addition to experiments often only existing in certain app versions, some features may be unique to a particular country or user demographic which consequently leads to specific targeting being required in such cases. (Xu & Chen, 2016) This further narrows down the collection of users available for certain experiments.

Sharing users concurrently between multiple experiments depending on the experimented changes can lead to interferences that invalidate the causal relationships inside the tests. Consequently, experimenters need to take into consideration which experiments can overlap and interact with each other without affecting the results. (Kohavi, Longbotham, et al., 2009) Especially in larger organizations, experimentation for the same product can be distributed across multiple teams, making it more difficult to run experiments simultaneously without interacting with other's experiments or complicating application logic (Bakshy et al., 2014). Experiments without a sufficient number of users on the other hand lead to underpowered results (Fabijan et al., 2017a), and therefore limited number of users is one of the key constraints limiting the number of A/B tests that can be run.

Multivariate tests (MVT) are a design which is often suggested to remedy the issue by using the core mechanism of A/B testing but including multiple factors within the test setup. Being able to test many factors simultaneously avoids the need to run multiple changes back to back and waiting for statistically significant results for each change, thus accelerating improvement. However, MVTs are harder to analyze and interpret, and the interactions between factors can't be estimated without fractional factorial design which increases the amount of combinations

manyfold. (Kohavi, Longbotham, et al., 2009) Building code for a complex MVTs requires more time setting up and is prone to bugs (Kohavi et al., 2014), as well as limits when the test can be started as all factors need to be developed ready for the experiment to commence. Furthermore, some combinations of factors may give a poor user experience. (Kohavi, Longbotham, et al., 2009) Google reportedly has developed and uses a design of partitioned overlapping experiments instead of multi-factorial design to test the smaller changes, which tries to remedy some of the issues with MVTs (Tang et al., 2010).

When there are no significant interactions such as in case of smaller insignificant changes, MVTs can allow to simultaneously measure each factor with a single collection of experimental units. However, the currently accepted best practice among A/B testing frameworks still is that experiments should be confined to one factor at a time, on the basis that changing multiple factors will make it difficult to determine which factors are affecting the outcome measured by the experiment (Scott, 2015). Kohavi et al. similarly advocate against complex designs. While some literature and commercial products promote the benefits of MVTs, it is typically more beneficial to run simple univariable test iteratively as they are easier to understand, run sanity checks, and thus in most cases are more trustworthy. Complex designs not only make it hard to determine cause-and-effect relationships, but also make experiments more vulnerable to errors. (Kohavi et al., 2014)

Another shortcoming due to the nature of A/B testing is the difficulty in segmenting and personalizing features or variables to users with A/B testing itself. Finding and deriving segments from the A/B test results requires significant effort and deep-diving (Kohavi, Longbotham, et al., 2009), and is dependent again on number of users to establish statistical significance. As such, A/B testing typically is used to identify the winning variant and applying that to all users, although analysis and apply decision could also be made on a quite high-level user segment basis. Related to segmenting, another raised concern regarding A/B testing is that users may notice they are getting a different variant than their friends or family, or see multiple variants when using different devices. It is however relatively rare that users will notice any difference. (Kohavi, Longbotham, et al., 2009)

With all that, A/B testing requires additional coordination on when can experiments be stopped on gathering users, which experiments to start next, which experiments are allowed to overlap, and how are the experiments targeted. While in smaller scale this can be relatively simple, performing it in a larger scale where multiple teams are running multiple experiments can be challenging. (Fabijan et al., 2019) Furthermore, in application domain further challenges in scheduling compared to web domain come from the need to roll out the app new versions through the app stores. In addition to submitting application builds to the app stores for review, waiting them to pass review and to be released, the end users still need to update the app. Even if new app versions can be built and released every other week, from the users' perspective it can be annoying to constantly update the app. (Xu & Chen, 2016) Lastly, it is worth reminding that not all features can be A/B tested, and attempting to replicate stellar results reported by others will most likely not bring similar success (Kohavi et al., 2014). To summarize the limitations discussed in this chapter, Table 6 lists the key shortcomings of A/B testing.

Table 6. Characterizing limitations of A/B testing

Cause	Effect
Difficulty of getting metrics right	(1) Challenge in arriving at trustworthy results (2) Difficulty in measuring and predicting long-term impacts (3) Difficulty in understanding "why" and accurately learning based on results
Potential biases distorting results	(1) Increased need to pay attention in analysis that the results actually are what they seem to be to avoid wrong conclusions (2) Managing biases that could affect the results requires paying attention in planning the tests
Limitations on number of users	(1) Old users shouldn't be continuously re-used for subsequent tests to avoid carryover effects and bad user experience (2) Limited number of new users lengthens the duration of collecting adequate sample sizes and limits the amount of test that can be run
Cumbersome personalization	(1) Segmenting features with A/B tests is limited to a high-level and requires effort
More complexity in coordination	(1) Additional level of planning and coordination required to manage tests and avoid interference between tests

3 MACHINE LEARNING AND A/B TESTING

Automated optimization through machine learning adds additional intricacy, complexity and challenges to the experimentation process. Before a company considers incorporating machine learning into their A/B testing practices, the basic prerequisite is to have in place a robust experimental approach to product development. Ensuring a sustainable experimental approach to product development, the various subjects and practices covered in the previous chapters should be well regarded and thought out in the company (Lindgren & Münch, 2016). With that said, capturing user behavior and choosing among product variations can be aided by new experimentation tools (Bakshy et al., 2014).

As the goal of the thesis is to answer the question regarding the impact of utilizing machine learning methods in A/B testing, this section adopts a view mainly describing characteristics and differences on a high level in available techniques that could be utilized and focuses on practical implications. First, the chapter starts with an introduction to machine learning and suitable methods for experimentation, followed by a slightly more detailed description of different strategies available. Then, similar to previous chapter, benefits and limitations involved with the adoption of machine learning in experimentation are discussed. Lastly, the chapter covers some practical use cases on where machine learning methods have been found advantageous in the context of controlled experimentation.

3.1 Suitable machine learning approaches for experimentation

The field of machine learning in general is concerned with the question of how to construct computer programs that automatically improve with experience (Mitchell, 1997). Much like the name suggests, machine learning thus involves computer algorithms that are able to "learn" based on experience and then make a determination or a prediction based on it. A more widely quoted and more formal definition given by Mitchell defines machine learning by stating that "a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." (Mitchell, 1997, p. 2). There is an increasing use of machine learning techniques

in customization and recommender systems online, with topics such as reinforcement learning and multi-armed bandits receiving wide attention with companies successfully reporting the use of said techniques in their systems (Issa Mattos et al., 2017).

Generally, machine learning types are divided into three categories: supervised learning, unsupervised learning and reinforcement learning (Mueller & Massaron, 2016). Additionally, some sources mention semi-supervised learning, a method falling between supervised learning and unsupervised learning as a category of its own. Whereas supervised learning uses labeled data and unsupervised learning uses unlabeled data, semi-supervised learning utilizes both labeled and unlabeled data. Effectively semi-supervised learning can thus be considered rather a hybrid approach to attempt to improve the performance in one these two of these approaches. (Chapelle et al., 2006; van Engelen & Hoos, 2020) However, there have been arguments made recently also for re-defining and adding fourth main category of self-supervised learning, a relatively recent and promising form of machine learning techniques (Asano et al., 2019; Dickson, 2020). Without going into too much detail about each style of machine learning, there are underlying aspects that define which type of problems are the different styles applicable for. Schaal (2020) presented a visualization for the different machine learning styles with the inclusion of self-supervised learning, highlighting the underlying differences between their use cases (Table 7).

Table 7. Categorization of machine learning styles (Schaal, 2020)

Objective is	**Ground-truth exists**	
	Yes	**No**
Explicit	Supervised learning	Reinforcement learning
Implicit	Self-supervised learning	Unsupervised learning

The four machine learning styles are characterized in the framework by two main questions – whether the objective is explicitly or implicitly given, and whether there exists a ground-truth,

i.e. a definitive answer to a problem. If not yet directly apparent from the categorization, after a brief introduction it is quite obvious how the framework provides the answer to why specifically reinforcement learning is the area of interest when looking into applications within experimentation and A/B testing.

The machine learning area of reinforcement learning is largely originated by Sutton and Barto for solving sequential decision making problems (Sutton & Barto, 1998). At its simplest, reinforcement learning can be regarded as a process of trial-and-error coupled with feedback provided from the environment that indicates the utility of the outcome. It enables the learning of optimal behavior through a set of sequential decisions that result in the achievement of a goal or the best possible outcome. (Gatti, 2015) More precisely, the approach is based on an agent which repeatedly interacts with the environment to learn and make decisions (Figure 8).

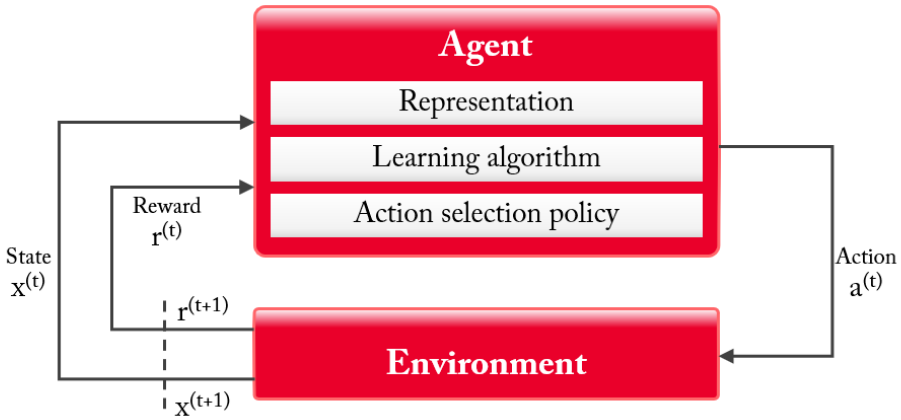

Figure 8. The reinforcement learning paradigm (Gatti, 2015, p. 8)

The basic idea is that the agent is observing the state of the environment $x^{(t)}$ at time t, selecting an action $a^{(t)}$ which it believes likely to be beneficial based on the algorithm's action selection policy and transitions to state $x^{(t+1)}$. At $x^{(t+1)}$, the environment issues feedback in a form of reward $r^{(t+1)}$ to the agent which provides an indication of the utility of the action. The feedback provided to the agent is used to improve its estimation of the value of actions in each state. Over time with repeated interaction with the environment and corresponding rewards, the agent's estimation of true state values slowly improves, enabling it to learn and select more optimal actions in future interactions that ultimately lead to the greatest cumulative reward. (Gatti,

2015) Accordingly, trial and error search and delayed rewards are the two most important and distinguishing characteristics of reinforcement learning (Sutton & Barto, 1998).

Reinforcement learning focuses on online improvement and presents a trade-off between exploration and exploitation. The trade-off is involved with finding a balance on exploring different solutions in order to find the optimal solution and moving to exploit the best solution to gain largest cumulative rewards. Ultimately, reinforcement learning methods attempts to map situations to actions to maximize a numerical reward signal. Unlike in most forms of machine learning, the learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them. (Sutton & Barto, 1998) Referring back to Table 7, with reinforcement learning there isn't a ground-truth known that would guide which actions to take, yet the agent is trying to optimize for an explicit goal: maximizing the reward. Comparing this back to controlled experimentation, there similarly exists an explicit goal of discovering the initially unknown best variant from the available options.

From a broad class of reinforcement learning problems, one common form of this exploration-exploitation trade-off is called the multi-armed bandit problem. (Burtini et al., 2015) The multi-armed bandit (MAB) problem is a fundamental dynamic optimization problem in reinforcement learning (Sutton & Barto, 1998), which is being widely explored in the context of online experiments (Issa Mattos et al., 2019). MAB problem describes a sequential decision-making problem where the agent has the choice to play one of multiple options with different reward probabilities. The name multi-armed bandit derives from "one-armed bandit", a colloquial term for a slot machine. (Scott, 2010) Accordingly, the multi-armed bandit problem can be seen as a collection of slot machines – or arms – which the agent has a choice to play, with the goal of achieving the largest possible reward from a payoff distribution with unknown parameters. (Burtini et al., 2015)

The straightforward analogy in online world is to see different website or application configurations as a row of slot machines, each with their own probability of producing a reward. Some of the arms might have higher reward probabilities than others, but the agent initially has no knowledge regarding the expected payoff of the arms, and thus must play the different arms in order to learn about the arms' reward probability. Thus, the agent must in each stage decide

which arm of the experiment to observe next. (Scott, 2010) Effectively, the agent experiments with a system, and the system responds to the experimentation with rewards (Kubat, 2017). The choice of arm however involves the trade-off between utility gain from exploiting arms that appear to be doing well based on the limited sample information, and exploring arms that might potentially be optimal, but currently appear to be inferior due to sampling variability. (Scott, 2010) The agent's goal therefore must be to find the optimal balance between pure exploration to pure exploitation (Kaibel & Biemann, 2019). This is the behavior that the reinforcement learning paradigm also seeks to emulate (Kubat, 2017).

In all different variants of the MAB problem, only the payoff for the one selected arm at any step is observed and not the payoffs for non-selected arms. This partial information nature of the bandit problem distinguishes it from generalized reinforcement learning, where observing the payoff of all arms, i.e. full information, is usually assumed. This property of the multi-armed bandits specifically makes them the appropriate method for experimental design out of the collection of all other machine learning approaches (Burtini et al., 2015), as many real-world learning and optimization problems are modeled in this way (Vermorel & Mohri, 2005).

3.2 Multi-armed bandit techniques

Many authors attribute the introduction of the bandit problem to Robbins (Robbins, 1952), but the fundamental idea behind the problem dates back at least to Thompson (Thompson, 1933). The motivation for Thompson to study bandit problems originated from clinical trials on deciding which treatment to use on the next patient, but modern technologies have created opportunities for new applications and enabled bandit algorithms to play an important role in several industrial domains. Online services in particular are suitable targets for bandit algorithms (Bubeck & Cesa-Bianchi, 2012), and any experiments with a finite number of variants can in fact be formulated as bandit problems (Issa Mattos et al., 2017).

The explore-exploit trade-off inherent in many sequential decision-making problems is well studied within MABs (Issa Mattos et al., 2017), and many techniques or approaches under multi-armed bandit problems have been introduced and developed (Agrawal & Goyal, 2013). There are three categories that are mainly discussed with MABs: the "traditional" basic

stochastic bandits, contextual bandits and adversarial bandits. To set the initial frame of mind on how these differ, Figure 9 on the following page displays the very basic idea of the bandit methodologies in the context of A/B testing. It is further worth noting that adversarial bandits aren't included in the figure for a particular reason described later in chapter 3.2.2.

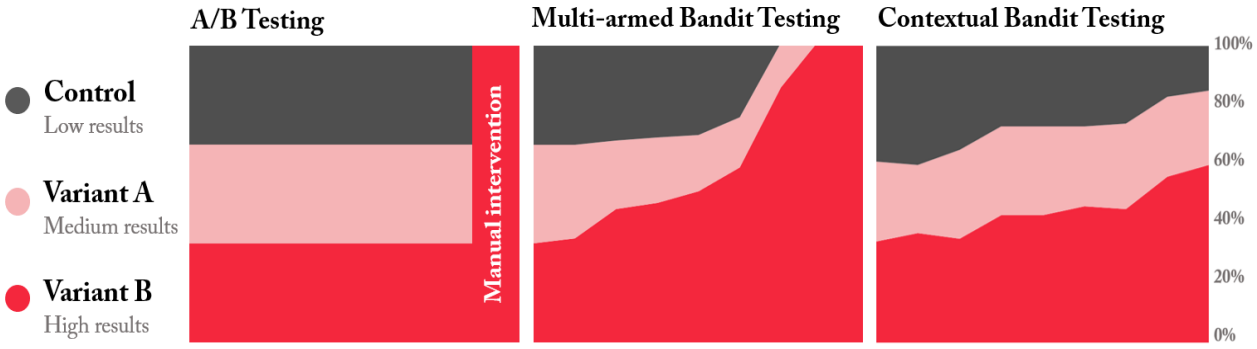

Figure 9. Variant group allocation in test methodologies over time (Navot, n.d.)

The visualization presented here shows how bandit tests differ in results over time compared to traditional A/B testing. While the classic A/B test approach requires waiting a statistically significant winner is found for a manual intervention to take place, bandit algorithms learn while the test runs, dynamically allocating traffic for each variant – i.e. arm – based on its performance. Moreover, contextual bandit algorithms change the population exposed to each variant to continuously maximize each variant's performance, essentially personalizing the experience for different people. Notably in contextual bandit's case there isn't a single winning variation, nor necessarily not a single losing variation. (Navot, n.d.)

Under each MAB approach, there are several algorithms that have been proposed and evaluated. (Scott, 2010) The logic of an algorithm trying to solve the multi-armed bandit problem is often also referred as a strategy or a policy (Burtini et al., 2015). More specifically, policy can be defined to be a particular sequence of actions that is selected throughout the decision making process (Gatti, 2015). A policy tries to adaptively select the arms to achieve the best profit, with the objective function being to minimize statistical regret (Misra et al., 2019). Regret in this case is the difference of the drawn arm and the optimal arm, and similarly cumulative regret can be measured as the difference between the cumulative total reward

achieved if the optimal arm was chosen every time, and the actual total reward received (Issa Mattos et al., 2019). Similarly, Auer et al. define regret as the expected loss due to the fact that the globally optimal policy is not followed at all the times (Auer et al., 2002), with the optimal policy being the set of actions during the process that maximizes the total cumulative reward (Gatti, 2015).

In order to minimize regret, the algorithms exploit past experience to select the arm that appears the best. However, the seemingly optimal arm may still be suboptimal due to imprecision in algorithm's knowledge, hence the algorithm has to explore the suboptimal arms to gather more information about them. Exploration thus increases the short-term regret as suboptimal arms are chosen, but ultimately tries to reduce long-term regret by obtaining information about the arms' payoffs to refine the algorithm's estimates. (Li et al., 2010) In general, regret measures can only be monitored in a simulation environment because of their requirement of an oracle which can unambiguously identify the optimal arm. (Burtini et al., 2015). The different algorithms are therefore developed to minimize regret in different environments and maximize the effectiveness in achieving highest possible reward.

3.2.1 Basic stochastic bandits

A stochastic bandit is finite-armed multi-armed bandit which works according to the reinforcement learning paradigm by repeatedly choosing between independent arms over a time horizon to minimize total regret. More specifically, at each time step, the policy plays a single arm and receives a reward based on predefined, desirable performance. The arms drawn and the learned reward distribution are then used by the policy to inform further decisions. (Burtini et al., 2015)

There is a plethora of different strategies that have been used with stochastic bandits, including pure exploration strategies, purely greedy strategies, hybrid strategies and methods based on upper confidence bounds (Scott, 2010). To understand differences in choice of algorithm, some of the popular and more studied stochastic bandit strategies are presented in very brief detail next. Furthermore, it is good to acknowledge that adversarial and contextual bandits discussed later largely build upon the same basic MAB algorithms.

First multi-armed bandit strategy to cover is an index policy developed by Gittins, called Gittins index. (Gittins, 1979) It is a policy that is computing the expected discounted present value of playing a given arm based on assumptions of optimal play in the future – value known as the Gittins index. Thus, playing the arm with the largest Gittins index maximizes the expected present value of discounted future rewards. Gittins index hasn't seen a widespread adoption due to practical issues in the policy. (Scott, 2010) Mostly, Gittins index has a high computational complexity of maintaining the indices, practically constraining it to a limited set of known distributions (Burtini et al., 2015). Furthermore, Gittins index is an inconsistent estimator of the optimal arm and has further practical limitations on the assumptions it makes on arms and on the discounting scheme. Thus, in applied use there are other better policies to utilize. (Scott, 2010)

A more feasible class of strategies are UCB algorithms first introduced by Lai and Robbins (Lai & Robbins, 1985). UCB or Upper Confidence Bound algorithms assign each arm with an upper confidence bound from the confidence interval of its estimated reward, and accordingly the arm with largest bound is played. UCB algorithms not only explore how much reward each arm receives, but also the confidence in each arm. The algorithms balances the exploration-exploitation trade-off by selecting the arm with the highest empirical reward estimate and playing optimistically. (Issa Mattos et al., 2019) Thus, UCB algorithms play initially more systematically to reduce uncertainty but as uncertainty reduces over time, starts to play optimistically more to highest confidence bound arm. UCB algorithms have been an active research area and many different variations of these algorithms have been developed that differ on the distributional assumptions. (Scott, 2010).

Scott (2010) groups several decision-rule based policies under heuristic strategies. These include equal allocation, play-the-winner, deterministic greedy-strategies and hybrid strategies. Equal allocation policies are simple naïve methods emphasizing over-exploration by equally allocating observations to all arms until maximum optimality probability exceeds a determined threshold, from which point onwards the winning arm is played. Play-the-winner is another straightforward policy where an arm is played at time $t+1$ if it resulted in a success before at time t. Otherwise, then the next arm is chosen either at random or the arms are cycled through deterministically. (Scott, 2010) The obvious limitation of this strategy is that success has to

determined dichotomously, i.e. being either 0 or 1 (Kaibel & Biemann, 2019). It is also straightforward to see that play-the-winner easily tends towards equal allocation between arms and thus over-explores. However, play-the-winner can theoretically be nearly optimal policy when the best arm has a very high success rate. (Scott, 2010)

Greedy algorithms on the other hand are policies that purely focus on exploitation. Deterministic greedy strategies could for example always choose the arm with highest sample mean reward. This approach has shown to perform poorly as it fails to adequately explore the other arms (Sutton & Barto, 1998). Moreover, greedy algorithms suffer when batch updating is used instead of real time updates due to delayed feedback. Delayed feedback means that the results of the play of an arm become available only after a time delay, such as in case of batch updates, but the decisions on which arm to play next are required immediately. (Agrawal & Goyal, 2012) Thus, a bandit has to play multiple times before it can learn from recent activity. Greedy algorithms perform especially poorly in the early phases of batch updating because they only learn about one arm per update cycle due to their heuristic strategy on selection of arm. (Scott, 2010)

Hybrid strategies on the other hand are a collection of policies based on greedy algorithms but have been modified to force some amount of exploration (Scott, 2010). The most commonly known one is the ε-greedy strategy, which has also probably been the most widely used policy with stochastic MAB problems in practice (Vermorel & Mohri, 2005; Burtini et al., 2015). In ε-greedy rules the policy chooses with probability 1- ε the arm with highest expected reward, and otherwise randomly pulls one. For example in case of ε being 0.6, the arm with the highest expected outcome is selected in 60% of cases and otherwise pure random exploration strategy takes place. The ε is determined differently based on which variant of the ε-greedy strategy is used. (Vermorel & Mohri, 2005) One well performing strategy is the ε-decreasing strategy where the value of ε decreases over time, as the decreasing ε leads closer to the optimal strategy asymptotically. With carefully chosen ε function, the total regret here can even be near optimal bound. (Scott, 2010) On the other hand, one very simple and less optimal version is the ε-first strategy, where for the first N rounds ε remains 1 and the levers are randomly pulled. After N rounds, during the remaining rounds the lever of highest estimated mean is pulled. (Vermorel & Mohri, 2005) From multi-armed bandit methods, ε-first strategy most closely resembles a

classical A/B test experiment with obvious similarities of pure exploration leading to an intervention to switch to pure exploitation.

Despite the wide usage, ε-greedy strategies can be wasteful as they use simple random sampling as the basis for exploration. A stratified approach that under-samples the likely sub-optimal arms is a way to trying to remedy this. (Scott, 2010) Softmax learning (Luce, 1959) is a randomized strategy resembling ε-greedy strategy, but with the difference that it explores by weighting other arms according to their value estimates. (Issa Mattos et al., 2019) While in ε-greedy strategy all alternatives to the best arm have an equal probability to be chosen, Softmax tries to take into account the differences between inferior arms. This means Softmax rule chooses each arm with a probability that is proportional to its expected outcome Similar to the ε-greedy strategy, Softmax also features a tuning parameter to determine how much exploration or exploitation is done at any given time. (Kaibel & Biemann, 2019)

Considerable amount of newer bandit literature focuses on Bayesian decision rule, which seeks to address the MAB problem by applying Thompson sampling. Scott (2010) also refers this approach as randomized probability matching. This Bayesian decision rule randomly chooses arms based on the Bayesian posterior probability that an arm is optimal. (Scott, 2010) Basically, the idea is to assume a simple prior distribution on the underlying parameters of the reward distribution of each arm, and at every time step play an arm according to its posterior probability of being the best arm (Agrawal & Goyal, 2013). This approach combines many positive attributes of the other policies mentioned above. Thompson sampling takes into account the differences between inferior arms and decreases exploration behavior naturally as the experiment proceeds by also taking into account the number of trials per arm and the variance in prior rewards when estimating the expected rewards of each arm. Then, based on the number of trials per arm as well as their mean effectiveness and variance, the Bayesian posterior probability of an being optimal is updated constantly during the experiment. Thus, arms that yield high mean rewards with small variance based on prior rounds have a higher probability of being the best arm and being played. (Scott, 2010; Kaibel & Biemann, 2019)

Being a randomized algorithm, Thompson sampling is also more robust to batch updating and delayed feedback than other methods as it doesn't as likely get trapped to an early bad decision

during the delay. (Agrawal & Goyal, 2012) The randomization aspect of Thompson sampling is according to Scott (2015) an overlooked advantage of the policy, as in real world experiments high traffic and delayed updates due to technical reasons are common. While non-randomized algorithms choose a single arm to bet before updating, Thompson sampling spreads the observations across arms according to the posterior probability while waiting for updates. Additionally, Thompson sampling doesn't require tuning parameters set by the analyst that can lead to inefficiencies. However, tuning parameters can still be introduced into Thompson sampling if desired. (Scott, 2015)

Thompson sampling has been shown mathematically and empirically to perform competitively against other well performing policies such as UCB or ε-greedy (Burtini et al., 2015; Scott, 2015). As inferior arms are less frequently played than arms that are more likely to be optimal, Thompson sampling improves the economic performance of the experiment while also offering a better experience to customers as less of them get assigned to inferior arms. Avoiding inferior arms also produces greater sample sizes among the better arms, which helps distinguishing the best arm(s) from the merely good ones. (Scott, 2015) Still, there isn't a consensus on a single best policy with some studies suggesting deterministic algorithms such as UCB and its variants are favorable since they have a feasible closed-form index policy, while others promote the Bayesian Thompson sampling variants due to their randomized allocation and resilience to delayed feedback (Hejazinia et al., 2019).

3.2.2 Adversarial bandits

Adversarial bandits are one of the strongest generalizations of the multi-armed bandit problem. Rather than the rewards being picked from a prior set distribution, in adversarial bandit problem an agent chooses an arm and an omniscient adversary simultaneously chooses the payoff structure for each arm. The bandit problem is transformed into an iterated three step process, which starts by the adversary picking reward distributions, followed by the agent picking an arm without awareness of the adversary's selections, and lastly rewards being assigned. This removes the distribution dependence on the arms, making adversarial bandits MAB problems in which there is no assumption of statistical reward generating process. Any policy for the

adversarial bandit problem must thus acknowledge the potential asymmetry in information between the agent and the adversary. (Burtini et al., 2015)

The adversarial or non-stochastic bandit problem was proposed as a way of playing an unknown game against an opponent, which is a classical topic in game theory (Bubeck & Cesa-Bianchi, 2012). Consequently, without constraints, the adversarial bandit can be seen as a competition game between the algorithm and an omniscient adversary with unlimited computational power and memory, making it capable of always staying ahead of any strategy the agent selects (Burtini et al., 2015). It can be argued that the adversarial bandit is not even a statistic problem, but rather a problem of game theory in how to behave in order to defeat an omniscient adversary (Stucchio, 2014). Adversarial bandits overall have hardly, if any, practical use cases already simply due to lack of omniscient adversaries in real life applications, and generally the adversarial bandit can be seen more as a scientific thought experiment.

3.2.3 Contextual bandits

A natural extension of the multi-armed bandit problem is to associate side information with each arm (Bubeck & Cesa-Bianchi, 2012). Such extensions of the "basic" MAB problem are contextual bandits, where the surrounding environment is taken into account when choosing arm in. The environment – i.e. the context – affects the reward associated with each arm in contextual bandits. (Slivkins, 2019) Similarly as in basic MAB, in contextual bandit the agent is presented with the choice of N arms to select with the goal to minimize regret over time. Before choosing which arm to play, the agent however evaluates also multi-dimensional feature vectors associated with each arm, referred more commonly as "context". The agent uses these feature vectors along with earlier rewards and earlier feature vectors to make the choice of arm. Over time, the agent tries to gather enough information on how the contexts and rewards relate to each other in order to estimate which arm is likely to give the best reward by looking at the feature vectors. (Agrawal & Goyal, 2013)

To conceptualize the problem again with the slot machine analogy, contextual bandits model situations where the machine have different properties believed to affect their payoff. These properties are typically divided into arm-context and world-context, which may include

multiple different dimensions: some slot machines might be old, some might be new; some might be blue, some might be red (categorical context), and machines closer to the back of the casino might seem to pay better (linear context). The categorical and linear context form together the arm-context. In addition to arm-context, the payoffs could vary by other environmental factors such as time of day or day of the week, known as world-context. Arm-context can be used to learn share properties across arms, whereas world-context interacts with arm context and is declared on a per step basis. (Burtini et al., 2015) Based on this contextual side information, a concept of contextual regret is introduced, and the optimality in contextual bandits is defined with respect to the best mapping from contexts to arms, instead of simply having a single best arm. (Bubeck & Cesa-Bianchi, 2012).

In general, contextual variables enable a more elaborate learning process with the vector of contextual variables used to guide learning. Even if the contextual variables are incomplete or are not strongly covariant to the variable of interest, they do not significantly affect the learning process negatively. The simple non-contextual bandit models are not efficient in many cases when a complication has been introduced. (Burtini et al., 2015) What makes the contextual bandit particularly useful version of the MAB problem is the ability to define which is the best arm specifically in the given circumstances (Agrawal & Goyal, 2013). This makes the contextual bandit also notably interesting from experiment design perspective by allowing the consideration of other dimensions which plausibly covary with the selected treatment. (Burtini et al., 2015)

From practical perspective the main motivation for contextual bandits in experimental design is that a user with known user profile arrives each round. The user profile is used as the context (among other factors) by which the algorithm can personalize the user's experience to the expected best variant according to the context. (Slivkins, 2019) Take personalized news article recommendation for instance as an example. In this problem, the different articles correspond to the bandit's arms, and a reward is obtained if the user clicks on the shown article. The context the bandit utilizes in selection of arm may include user's historical activities, demographic information or geolocation as well as content information and categories of the article and for instance proximity to a major event such as sports events or elections. (Bubeck & Cesa-Bianchi, 2012; Slivkins, 2019) The presence of contexts typically creates number of different possible

variations of the best arm (Bubeck & Cesa-Bianchi, 2012), thus effectively leading to personalization based on multiple known factors of the user and the environment.

As stated previously, many of the contextual bandit strategies can be seen as extensions of the traditional MAB algorithms with the inclusion of contextual information. To shortly highlight the affiliation to non-contextual bandit strategies, LinUCB (Li et al., 2010) for instance is a linear type, contextual upper confidence bound solution derived from the UCB policies for non-contextual bandits. Consequently, LinUCB largely builds on the upper confidence bound work of the non-contextual bandit policies. Like all UCB methods, LinUCB similarly chooses the arm based on the highest upper confidence bound. (Burtini et al., 2015).

In turn, Thompson sampling with linear payoffs (Agrawal & Goyal, 2013), a linear Thompson sampling strategy (also known as LinTS), extends the Bayesian Thompson sampling algorithm to estimate reward with contextual variables. In the contextual version of Thompson sampling, a linear predictor defined by a multi-dimensional vector – or the context – is used to predict the mean reward of an arm. The linearity in both LinUCB and LinTS simply means that in the algorithm's expectation of the reward of each arm is linear with respect to the arm's features. (Zhou, 2015) Respectively, non-linear contextual bandits model more complicated problems and the rewards are approximated by different non-linear functions (Liu et al., 2018). There is a wide spread of different linear and non-linear contextual bandit models presented in academic research to solve the contextual multi-armed bandit problem, all in the end with the shared same goal to minimize contextual regret.

3.3 Benefits of multi-armed bandits in experimentation

The utilization of machine learning in experimentation can be seen to be focused on learning and optimizing business goals to drive post-deployment innovation (Issa Mattos et al., 2017). One of the key challenges discussed with A/B testing is the high number of users required to run A/B tests that are able to yield results with high power and high confidence levels. Collecting the required amount of data can be challenging especially for small to medium sized companies. Multi-armed bandits have the potential to deliver faster results with better allocation of resources compared to A/B testing. (Issa Mattos et al., 2019) As traffic allocation

to variants in non-contextual MABs starts to increase or decrease according to each variant's performance, traffic is not wasted on precisely determining the exact degree to which the suboptimal variants underperform, but instead more traffic is devoted to high performing variants increasing the ability to determine the difference between them. (Hejazinia et al., 2019) This effectively help differentiating good arms from the best ones quicker and can help decreasing the time and total sample size required for the experiment (Issa Mattos et al., 2019). Hence, multi-armed bandits are especially effective when there is a limited time-window and the decisions have to made quickly, with the decision being justifiable without high confidence levels due to the cost of type I errors being lower. (Kohavi, Longbotham, et al., 2009).

Dynamically changing the user allocation to the best performing variants and minimizing the exposure of users to sub-optimal variants also facilitates conducting more efficient and ethical experiments (Kaibel & Biemann, 2019). Assigning users to inferior variants only as often as absolutely necessary not only improves the user experience (Kaibel & Biemann, 2019), but also decreases the opportunity costs involved with assigning and holding users in worse performing variants. Thus, testing product improvements with MABs can reduce the cost of experimentation to the organization, especially if the differences in performance between variants are drastic. (Scott, 2010). The previously discussed fractional factorial design of A/B tests that strives for more efficient testing, but suffers from rigidity and complexity, is therefore additionally challenged directly by the alternative MAB approach. (Hill et al., 2017)

For a non-contextual bandit test that is not constrained by a limited window of opportunity, Hejazinia et al. suggest an example of an experiment cycle illustrated below in Figure 10. First, developing the bandit model for the use case as well as offline simulation and integration into production environment might be required depending on experimented change and desired actions for reward. After the MAB model is developed and validated, an experiment with MAB including multiple variant is launched to determine efficiently the best variant out of those. As MAB algorithms almost without exception optimize for one reward – or metric – to make them more robust, the winner of the MAB experiment can then be tested against the current version with an A/B test to statistically determine and ascertain that the variant is indeed performing better also in other metrics of interest. (Hejazinia et al., 2019)

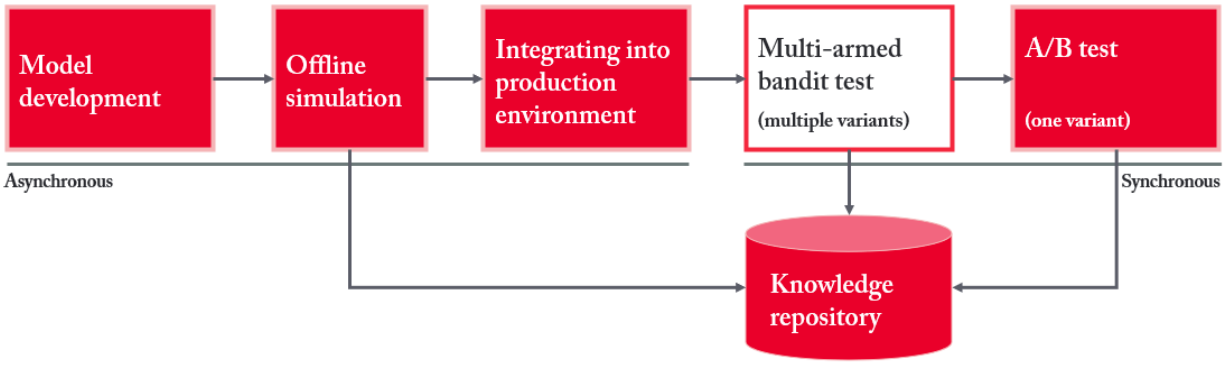

Figure 10. Non-contextual multi-armed bandit experiment cycle (Hejazinia et al., 2019, p. 2)

The hybrid MAB and A/B test approach eliminates quickly the under-performing variants and significantly speeds up testing, as based on empirical results a MAB test with multiple variants can achieve results in fraction of the time compared to an A/B test with even only a single variant group (Hill et al., 2017; Hejazinia et al., 2019). The hybrid approach also enables overcoming the partly black box that is MAB testing, enabling to determine there are no degrading effects to other behavior and that the effect correlates with a longer-term predicting metrics used in evaluation of an A/B test. (Hejazinia et al., 2019)

In addition to speeding up tests, an increasing number of researchers and practitioners recognize that one treatment does not fit all (Eden, 2017). While controlled A/B tests are well suited to binary decisions, they are limited in terms of contextualization and personalization (Hill et al., 2017). Instead of only allowing to test various different variants and determining a winner, contextual bandits can efficiently individualize allocation of the variants to each person based on their characteristics (Kaibel & Biemann, 2019). Continuous personalization of the user experience on a user to user basis theoretically thus allows better suiting each user's needs and improving desired metrics.

Thanks to the context captured by the contextual bandits, it is possible also react to changes in preferences over time and start routing users differently if the performance of the arms changes, unlike in traditional A/B tests where the winning variant is settled upon after the test concluded. Thus, ongoing optimization can effectively be achieved by having ongoing contextual bandit experiment live on the optimized feature or a parameter. Taking into account the changes in performance of the variants over time can also be done in non-contextual bandit tests by

incorporating resetting and forgetting aspect known as fading memory. This non-stationarity allows considering only the information from a defined period of time or from the last specified number of users to re-train policies as new data becomes available. Hence, policy training de-emphasizes older data points over time and allows adjusting for more present behavior. (Kaibel & Biemann, 2019; Slivkins, 2019) Kaibel and Biemann suggest that this could be particularly useful in some cases where new variants are desired to be added at a later stage to the experiment, for instance as iteration based on current results (Kaibel & Biemann, 2019).

Many bandit algorithms discussed and evaluated for both non-contextual and contextual bandit problem are open-source and available online, providing an easy starting point. For example, bandit algorithms and some of the core functionality is provided via Vowpal Wabbit, an open-source machine learning library majorly contributed by Microsoft Research (Slivkins, 2019). Justifying the credibility, Vowpal Wabbit has been successfully utilized to implement machine learning in production systems. Number of other similar systems and libraries also exists online. (Agarwal et al., 2016) Additionally, many other commercial products such as Adobe Target or Google Analytics allow utilizing MAB approaches on A/B tests (Scott, 2013; Hickey, 2019).

3.4 Limitations of the multi-armed bandit approach

Bandits problems differ from normal controlled experiments in the conceptual level. Whereas normal controlled experiments focus on hypothesis testing and obtaining insight and detailed understanding of the tested subject with a statistical confidence level, bandit problems focus on optimization and do not balance the experiments to accurately estimate the inferior treatment effects. (Scott, 2010)

While traditional A/B tests fully focus on exploration during the experiment, MABs use information gathered during the experiment to blend exploration and exploitation. This means a trade-off is made between maximizing the statistical power for all variants groups and maximizing the effects. (Kaibel & Biemann, 2019) Thus, the statistical power to identify differences in treatment effects is lower, making the MAB approach less useful for analyses which involve comparing variants or determining specific degree of effects for generating insight (Burtini et al., 2015). That said, it is not always critical to know why or how exactly a

change caused an uplift to benefit from the knowledge of the specific change resulting in it. The motivations behind the behavior of users in some optimization cases can be particularly difficult to determine or are not of interest nevertheless. (Kohavi & Thomke, 2017)

With the fundamental difference between traditional A/B testing and MAB-based experiments, a common pitfall is still to assume that MAB experiments have the same assumptions as normal A/B tests. MAB algorithms have several assumptions that must be met for a correct inference, related for instance on independence of the arms, reward distribution, and reward time invariance. (Issa Mattos et al., 2019) A common violation is assuming the underlying distributions and parameters are stationary. However, the state of the world is changing around the learning process and the best arm in one time period may not be the best arm in another. Rapid changes of the distribution and switching-type models can results on poor performance on many fixed policies. (Burtini et al., 2015) For example, the distribution of clicks in a website could change between weekdays when the system has more business activity and weekends when there is more home activity. This is a possibility regardless of whether the experiment is run as a bandit or as a traditional experiment, however the bandit is more susceptible to it due to making conclusions about arms in a shorter period of time (Scott, 2015). With the temporal changes the MAB algorithm might learn and allocate exploitation to a suboptimal arm depending on when the experiment was started. (Issa Mattos et al., 2019)

This also sometimes referred as Saturday/Tuesday problem can invalidate the experiment if the underlying assumptions are not understood and validated. Novelty effects discussed with A/B tests can similarly cause too early exploitation of an arm due to increase in metric in the beginning which then rather drops over time. After a period of time, the variant which having novelty effects might in fact have a statistically significant lower performance in the evaluated metric compared to the control group. (Issa Mattos et al., 2019) Non-stationary bandit models have been proposed to partially overcome the issue by allowing the data to decay out of the model with a time-weighted component. However, this solution requires an accurate model of the appropriate rate of decay to be efficient. (Burtini et al., 2015)

Other more specific violated assumptions studied by Issa Mattos et al. include presence of lagging metrics or rescaling and normalization of metrics. Commonly tracked metrics such as

user retention which have a dependency on time are incompatible reward metrics with algorithms that assume instant feedback. Metrics that represent more abstract concepts usually start to lag more, making them in general less attractive for MAB-based experiments as lagging metrics might lead to incorrect conclusions even with algorithms that are well functional with delayed feedback. Rescaling and normalizing metrics might be necessary for some MAB algorithms such as UCB where the reward has to be constrained between 0 and 1. Normalization process needs to be validated to perform as expected, as depending on the assumptions the transformation logic might not be valid as the experiment progresses. Consequently when there are uncertainties in the assumptions, traditional A/B testing provides a more robust framework to conduct the experiment and evaluate the changes. (Issa Mattos et al., 2019)

On an A/B test, several defined metrics of interest are tracked to evaluate the impact of a change. In a typical MAB-based experiment, the algorithm does the allocation based on one success metric, the reward, and incorporating additional metrics on top of the MAB can be compromised because of the previously discussed lack of power in some arms. This is only not preventing the use of other supporting metrics for an in-depth analysis, but also the choice of reward greatly affects the outcome and conclusion of the experiment. (Issa Mattos et al., 2019) One suggested potential solution is the use of MAB extensions for multi-objective optimization for example with Pareto relations, first proposed by Drugan and Nowe (Drugan & Nowe, 2013). However, providing a validated Pareto curve to a MAB algorithm requires a deep understanding of the system and users, as well as adds great technical complexity and additional failure modes to an experiment, and the approach hasn't yet seen industrial use. With contextual bandit tests additional information from other metrics can be added as exogenous variables, however this solution also has the same drawback of potential lack of power in arms. (Issa Mattos et al., 2019)

Typically there are also concerns related to traffic being sent to suboptimal arms, and the lack of abilities to be able to identify if it happens. The source for this might not necessarily even be in invalid assumptions or a poor choice of reward metric, but inherent from the algorithm itself. The choice of algorithm to manage the exploitation-exploration trade-off reliably and efficiently to minimize regret is thus as is important. (Scott, 2010) Overall the exploration-

exploitation trade-off can be seen as one of the main sources for both benefits as well as disincentives to use MAB experiments.

Additionally, the literature notes that MAB-based experiments introduce other technical limitations as well. For instance, according to Issa Mattos et al. several MAB algorithms do not support deterministic randomization reassignment as they do not use pure randomization in their assignment process. Instead, these algorithms rely on cross-checking and caching methods to keep the variants and user experience consistent (Issa Mattos et al., 2019), which introduces restrictions on the use of ramp-up procedures and automated shutdown (Kohavi, Longbotham, et al., 2009). If the algorithm uses caching or cross-checking methods, it might be necessary to implement an extra layer of application code in order to handle ramp-up and automated shutdown cases (Issa Mattos et al., 2019).

Most MAB algorithms however allocate lower number of users to bad variants and thus it is often considered that MAB can replace ramp-up procedures. With additional measures, such as enabling hard allocation boundaries for all variants, comes in the danger of degrading the exploration and decision-making process of the algorithm. Issa Mattos et al. remark that MAB algorithms also prevent the use of sample ratio mismatch quality check which allows investigating the traffic split of users to each variant and detect any possible randomization bias, in case the MAB algorithm uses a randomized component. As regret minimization in MAB-based experiments depends on asymmetric sampling and allocation between variants, it is hard to identify any randomization or instrumentation bias in the system during the experiment. With this in mind, additional care should be taken if MAB is combined with ramp-up procedures to avoid biases. (Issa Mattos et al., 2019)

The main limitation to understand however still remains to be when to use bandit testing and when classical A/B testing. One of the main reasons to use a traditional A/B test to subsequently evaluate the winning arm of a prior MAB-based experiment is to control the possibility of type I error. Whereas A/B tests constrain type I error to the pre-defined significance level set typically to 5%, in MAB-based experiments type I errors can be more common. The pitfalls related to naïve implementation based only on MAB experiment results are typical for organizations that introduce MAB algorithms without in-depth knowledge of the methodology

and without explicit goals for outputs. Accordingly, it is worth noting that MAB experiments shouldn't be used for exploration purposes or for understanding user behavior as in such cases the control of type I error and predetermined power are more important than regret minimization. (Issa Mattos et al., 2019)

Experiences from companies suggest that estimating the consequences of committing to type I or type II error, as well as identifying how the outcome of the experiment will be used in future development are the first fundamental steps in deciding which type of experiment to use (Issa Mattos et al., 2019). In case of the estimated cost of type I error being high, using classical A/B tests is typically a good strategy as they are designed to be analyzed using tools and processes that tightly control the type I error rate. However, when the goal of the experiment is not understanding the feature and user behavior, the estimated cost of type I error is low and on the other hand the opportunity costs related to type II error high, MAB-based approaches can provide value to the experiment. (Scott, 2010; Issa Mattos et al., 2019)

Finally, it is also worth noting that while machine learning systems can seem relatively easy to develop and implement, they can increase technical debt, maintenance costs and increase uncertainty in reliability. (Sculley et al., 2015) Many of the bandit algorithms used in experimentation have mathematical proofs, but system validation can become a problem in the everchanging online environment the algorithms interact with (Issa Mattos et al., 2017). Moreover, use of machine learning approach in experimentation can further be hindered by cultural and training barriers. Whereas A/B tests are typically better understood by managers, developers and other stakeholders, MAB-based experiment may be found less transparent and not properly understood with their associated limitations. Issa Mattos et al. additionally note that popular experiment models such as HYPEX or RIGHT model may not be fully aligned with MAB-based experiments, as these models assume the experimentation process should be minimizing type I errors rather than minimizing opportunity cost related to not exploiting the best solution. (Issa Mattos et al., 2019)

3.5 Practical use-cases for multi-armed bandit experiments

Despite the limitations associated with multi-armed bandits, they still provide various benefits in appropriated situations. Delivering value to customers with MAB experiments is still an emerging research area and practice in the industry, however several best practices and appropriate use-cases have already been identified by researchers. (Issa Mattos et al., 2019) Multi-armed bandits have successfully been adopted for various kinds of experiments (Burtini et al., 2015). The commonality in all MAB-based experiments is each arm of the MAB corresponding to a variant of the product, and with the reward being a defined trackable user metric. The reward metric furthermore should have a positive direction so that a higher value of the metric is considered better than a lower value. (Issa Mattos et al., 2019)

Online software such as websites, cloud services or games are especially compliant with continuous bandit optimization as experimental variation is easy to introduce, and user responses to the changes can quickly be observed (Scott, 2010). Typical application scenarios for MAB-based experiments for instance in web domain include news article recommendation or personalization, displaying ads, product recommendation or webpage layout optimization (Slivkins, 2019). Website optimization is a canonical use case for traditional A/B testing, where MAB based experiments can display improved performance. Optimizing a website is motivated by actions defining a successful visit, often referred as conversion, such as making a purchase, visiting a particular section of the site or signing up for a newsletter. MAB-based experiments can decrease the amount of lost conversions during experimentation period significantly compared to a traditional experiment. (Scott, 2015)

In website optimization often only small differences are observed, which is hard from statistical point of view and leads to lengthy experiments to gain statistical significance. Further, as the baseline probabilities are often small as well, even the small difference between variants on a website with a large amount of traffic can lead to a relatively high number of lost conversions. The MAB approach is more efficient in these cases on finding the best arm than traditional statistical experiments and able to exploit the advantages of it earlier. (Scott, 2015) Furthermore, the advantage for the MAB approach only increases as the number of variants in the experiment grows (Hill et al., 2017). This promotes also the use of MAB-based experiments

on multivariate testing where a high number of layout options are tested in conjunction (Scott, 2015).

For example, Scott (2010) demonstrated handling multifactor experiments using Thompson sampling with a probit regression model. Similarly, Hill et al. (2017) evidenced an approach to layout optimization using Thompson sampling -based model incorporating interactions between components of the page. In their experiment it took non-contextual MAB algorithms three to nine days to converge to the winning variant, whereas in contrast in their case arriving at statistically significant results on all combinations with a traditional A/B test would have taken an estimated 66 days. Furthermore, the winning layout for the particular experiment showed a 21% lift over the median layout and a 44% lift over the worst performing layout. In addition, a contextual version of the algorithm also applied seemed to perform well in scenarios where the influence of context is significant. Given the noteworthy results, Hill et al. argue that there seems to be a considerable business opportunity in combinatorial optimization of web page layouts with MABs. The underlying MAB model in question is reportedly utilized within various Amazon services, demonstrating continuous learning that is out of reach for traditional randomized experiments. (Hill et al., 2017)

Another well-known case to run bandit-based experiments are short-term campaigns where there is a limited time to exploit an opportunity. A fixed time window to exploit the variants means failing to display for instance the optimal news recommendations or limited time offer that would lead to higher conversion is costlier. (Issa Mattos et al., 2019) Traditional A/B tests fail to exploit such situations as acquiring statistically significant results takes more time and exploiting the best variant can only be started after acquiring the results from the A/B test. Thus, short-term campaigns are well suited for MAB-based experiments where it is desired to maximize the cumulative reward for the short period of time (Issa Mattos et al., 2019).

For contextual bandit experiments, personalization is the main use case. One version of the software feature or website page may perform better for different users based on their geolocation or other preferences. Depending on the granularity of the contextual data available, contextual bandit approach may be used to personalize the software versions down to the individual level. (Scott, 2015) For example Li et al. conclude the contextual bandit approach to

be effective for personalized web-based services such as news article recommendations (Li et al., 2010). Similarly, content-serving systems such as news headlines or serving ads have the goal to provide the users with the most relevant content from a pool of possible options. In such systems the cost of type II error is again costlier than type I error. Additionally, the system typically needs to operate automatically without manual intervention. Thus, a MAB approach for experiments involving such systems is more well-suited than a traditional A/B test (Issa Mattos et al., 2019). Schwartz et al. demonstrated utilizing various policies in optimizing advertiser's resource allocation over time across many ad creatives and websites by sequentially learning about ad performance. This allows to maximize customer acquisition rates by testing many ads on many websites while learning which ad works best on each website. (Schwartz et al., 2017) Misra et al. on the other hand demonstrate the use of multi-armed bandit in dynamic online pricing with an UCB-based algorithm (Misra et al., 2019).

Sometimes an organization also has prior knowledge about preferences across user segments it wants to utilize in the experiment. Contextual multi-armed bandit can also be used in such cases to specifically incorporate prior knowledge into the experiment design with contextual information and target variants based on it to different users. (Issa Mattos et al., 2019) Prior knowledge might include learnings from previous experiments, organization's or experts' experience and beliefs or evidence from other organizations. (Kaibel & Biemann, 2019) With traditional A/B tests confined to random uniform variant assignment process (Issa Mattos et al., 2019), utilization of other knowledge to target variants to users would not be technically possible without splitting the experiment to a number of separate experiments hard-targeted to users based on certain criteria, and within experiments randomly assigning users to control or variant groups.

Outside these substantiated cases, there are naturally other opportunities to use MABs in online experiments when the discussed limitations are acknowledged and addressed. MAB-based experiments can provide a more efficient optimization method than traditional A/B tests by explicitly optimizing value (Scott, 2015) and typically achieving faster decisions regarding the best variant when the effect size and the difference in mean-reward are high enough (Issa Mattos et al., 2019). Finally, Table 8 below serves to summarize simplified guidelines for

selecting between traditional A/B tests, non-contextual MAB-experiments and contextual MAB-experiments.

Table 8. Guidelines for selecting controlled experimentation method (based on Issa Mattos et al., 2019, p. 78)

Aspect of the experiment	Method selection		
	Traditional A/B test	Multi-armed bandit	Contextual bandit
Goal of the experiment	(1) Learning (2) Innovation (3) Optimization	(3) Optimization	(3) Optimization (4) Personalization
Cost of type I or type II error	(1) Costs of both are high (2) Only the cost of type I is high	(3) Only the cost of type II is high	(3) Only the cost of type II is high
Knowledge of problem and assumptions	(1) Not well-known or understood	(2) Well understood and matches MAB assumptions	(3) Well understood and context is well-known and validated
Number of decision metrics needed	(1) Multiple metrics that can't be grouped into one (2) Single delayed or less sensitive metric	(3) Sufficiently sensitive single metric	(3) Sufficiently sensitive single metric
Time box of the experiment	(1) Short-term with no time pressure (2) Long term exploration for high sample size	(3) Short window of opportunity	(4) Long to adaptively change the variation

4 CASE STUDY ON FIELD IMPLEMENTATION OF MACHINE LEARNING IN A REAL A/B TESTING SCENARIO

The empirical part of the thesis aims to examine the discussed topics through a case study on practical implementation of machine learning in A/B testing, providing evidence especially to the third research question. This chapter focuses on the detailing and covering the research methodology, execution and results of the case study.

The case study takes place at Rovio Entertainment Corporation, a Finnish mobile-first games company that creates, develops and publishes mobile games. As one of the leading Finnish gaming companies, Rovio is best known for the global Angry Birds brand, originating back to the popular mobile game launched in 2009. Since then, the company has been striving to build competitive edge through technology – and reflecting that, currently Rovio's vision for machine learning aims that "in 2022 Rovio has a game tailor made for individual players" (Rovio, 2019). The company started building data capabilities to enable the use of machine learning back in 2011, with the technology first applied three years later. While Rovio has a well-established practice of continuously A/B testing new features and changes on live users, the case study examines the first implementation of a contextual bandit-based approach on experimentation within the company to supplement traditional A/B testing.

The executed case study consists of two experiments that are looking to optimize the in-app purchase (IAP) conversion through showing one of four different conversion offers for new players. The first experiment was ran using a traditional A/B test methodology seeks to find a baseline global optimum for the best performing conversion offer, while the follow-up experiment utilized a contextual bandit approach to optimize the variation of the same set of conversion offers individually for the players. The experimented subject is intentionally rather simple and straightforward to allow better inference of the results by eliminating excess complexity and convolution from the test setup itself. The bandit algorithm used is Online Cover, a sophisticated contextual algorithm shown to be robust and competitive when pitted against other policies (Agarwal et al., 2014; Bietti et al., 2018).

The Online Cover algorithm is an online approximation of the "Importance-weighted Low-Variance Epoch-Timed Oracleized Contextual Bandits" or ILOVETOCONBANDITS (Agarwal et al., 2014), an UCB-based contextual bandit algorithm with strong theoretical guarantees of efficient regret bounds (Burtini et al., 2015). Although it can be argued whether the academics particularly excelled at convincingly naming the algorithm, Burtini et al. consider in their work ILOVETOCONBANDITS as the state-of-the-art in contextual bandit algorithms with respect to regret and computational complexity (Burtini et al., 2015). Accordingly to the algorithm it is based on, Vowpal Wabbit library denotes Online Cover as the most sophisticated of the available options for real-time online contextual bandit learning (Vowpal Wabbit.org, n.d.). While this doesn't ensure the algorithm's performance by any means, an evaluated algorithm argued to manage the exploration-exploitation trade-off both robustly and effectively increases the initial confidence on the ability to deliver results and in handling technical complexity.

4.1 Case study research approach and methodology

The case study approach for the empirical part seeks to especially further evaluate role of machine learning within A/B testing practices on an applied scenario, with the goal to expand on learnings from the theoretical part. As a research method, case study is designed to enable an in-depth investigation of a topic within its real-life setting (Yin, 2014) which is seeking to generate deep understanding and insightful appreciation of the case, hopefully resulting in new learning about real-world behavior and its meaning (Yin, 1993). Utilizing case study methodology is valuable for business and management research as it allows the researcher to examine the problem or question in a practical situation (Farquhar, 2012). With the goal to evaluate realized impact of applying a multi-armed bandit approach to variant optimization, the case study provides the framework to conduct a detailed investigation through examining the issue in a real-life use case.

As described above, the case study in question is based on two field experiments executed sequentially with the A/B testing methodology in a production environment, using real customers as the sample. With field experiments a key concern raised from the research point of view however is the generalizability of the results. Accordingly, many researchers using field

experiments have opted for mixed methods approaches to utilize multiple sources of evidence, offsetting the weaknesses inherent to using a single approach. (Eden, 2017) Case study as a research method intrinsically enables combining different methods to collect both qualitative and quantitative evidence (Eisenhardt, 1989). In fact, Yin (1993) emphasizes the relevance of both qualitative and quantitative data in case studies to triangulate evidence and establish converging lines of discovery, generally leading to more reliable and generalizable results. In addition to supporting generalizability, using qualitative evidence can typically aid in interpreting and corroborating findings from qualitative data (Eisenhardt, 1989; Saunders et al., 2016). Notably, research regarding engineering systems has been particularly found to benefit from the use of qualitative methods to build insights not obtainable otherwise (Szajnfarber & Gralla, 2017).

Although case studies are typically associated with the inductive approach aiming to develop theory (Hamel et al., 1993), case studies have a distinctive place in evaluation research (Yin, 2014). It has been noted that case studies are well suited for answering evaluative questions such as "how", "why" or "to what extent" with a relatively full understanding of the nature and complexity of the issue (Farquhar, 2012; Saunders et al., 2016). Albeit this and the fact that the case study methodology aids in the generalizability of results, a single-case design as the one used here inherently trades off generalizability in particular for a more in-depth and insightful investigation of the phenomenon (Eisenhardt & Graebner, 2007). Using a single case can typically raise concerns about the representativeness of the case and thus about the generalizability of the results beyond the particular case (Hamel et al., 1993).

However, Yin (1993) argues that case studies typically tend to generalize to other situations through analytic generalizations, compared to typical qualitative research methods such as surveys which tend to generalize to populations through statistical generalizations. The analytic generalizations depend on the study's theoretical framework to establish a logic potentially applicable to other situations. Consequently, a thought-out theoretical framework and the use of theory helps generalizing the findings from the case study (Yin, 1993). More specifically, generating findings where the existing theory specifies certain type of results should occur provides generalizability for the particular set of results through analytic generalization and findings from previous studies (Yin, 2014). The criticism overall regarding the generalizability

of case studies is furthermore typically related to interpretive, qualitative research, with the criticism being more rare when it comes to qualitative and mixed methods case studies that play by the rules in their execution and design (Saunders et al., 2016).

4.2 Case study data collection

The data collection in case studies typically involves multiple data sources often consisting of multiple different data collection techniques (Farquhar, 2012). Case studies emphasize the study of the phenomenon within real-life context by collecting data in natural settings instead of relying on derived data (Bromley, 1986). Out of the common sources of evidence used in case studies, in this particular case three sources consisting of field experiments, related documentation as well as participant observation are utilized in the execution to collect quantitative and qualitative data. From research point of view, the goal of the experiment strategy is to study the effect of an intervention in measured variables based on formulated hypothesis to discover and observe causal links through quantitative data. Second, the document sources provide a secondary source for additional context and for facilitating the analysis of the quantitative data. Lastly, participant observation involves participating and engaging in the researched activities of the organization to gain a deep understanding of the context and learning directly from the research setting. (Saunders et al., 2016)

Aligned with the positivist approach, the empirical part and evaluation are largely based on measured, quantitative information (Farquhar, 2012), which is supported by the qualitative data to provide further meaning and explanation on the results. This aims to remove bias from the research by being objective on what is being observed, supported further through backing up the made observations with relevant theory (Farquhar, 2012). The quantitative basis of the research further makes it more reliable by allowing more explicit description of the research setting and procedures (Yin, 2014). The quantitative nature of the conducted field experiments also means the reliability of the study can be effectively controlled with an adequate sample size and treating outliers, while the validity of the findings can be ensured through appropriate experimental setup and choice of statistically evaluated metrics that match the assumptions and formed hypotheses (Saunders et al., 2016). With that, the detailed experiment setting and procedures are elaborated next along with going through the results of the case study.

4.3 Case study execution and results

The case study was executed between March and June 2020, with the data collection for the first A/B test experiment taking place between March 18th and April 15th, and the second MAB-based experiment ran from May 18th to June 4th. The experiments were ran within one of Rovio's current top grossing free-to-play mobile games, Angry Birds Dream Blast, using internal A/B testing platform and tools. In order interpret the results, the first thing to define with the execution any experiment is the hypothesis and the expected impact of the made intervention.

In free-to-play mobile games, revenue is often generated through optional in-app purchases that provide value for the players in various ways. The in-app purchases are typically available at all times through the game's in-game shop, although sometimes the players are shown limited-time offers that provide a better value through discounted price for a set of items. An example of such an offer is a conversion offer, which is typically a low price point offer shown to the players early on in the game to lower the barrier for purchases, allowing the player to also assess whether such transactions provide value they are happy to spend on.

In case of both the traditional A/B test experiment and the MAB-based experiment the goal of the experiment was to increase IAP conversion through slightly different conversion offers shown to the new players. The hypothesis was that different contents and more value through items in the offer, while maintaining the same price point, can make the offer more interesting and desirable for the player to purchase, affecting the measured metric of IAP conversion. Thus, the null hypothesis H_0 looking to be refuted with the test can be formulated as that 'there is no difference in conversion with different contents of the offer', while the alternative hypothesis H_1 in that case stating that 'there is a causal relationship between contents of the offer and IAP conversion'. The first traditionally executed A/B test included three different variants shown in Table 9, which were tested against the control conversion offer currently in place within the game.

Table 9. Baseline conversion offer A/B test experiment groups

Experiment group	Conversion offer content	Price point
Control	300 coins 	0.99$
Variant A	300 coins & one of each of the 3 powerups 	0.99$
Variant B	300 coins & one of each of the 3 pre-level boosters 	0.99$
Variant C	300 coins & 2 hours of unlimited lives 	0.99$

The conversion offer is shown in the test to the player after their first fail in a level and returning back to the game's main hub screen, which typically occurs within the first day of playing. If the player doesn't convert from the initial display of the offer pop-up, the offer still remains available through an icon in the main hub screen for a limited time of 48 hours before entering cooldown. The value proposition of the offer is communicated through a reference price point and a discount tag of -75%, which was calculated in this case based on coin value and thus the same for each offer in order not to introduce a second factor distorting the causality.

Ultimately the objective and expected key learning of the A/B test was to find the globally optimal conversion offer contents to show to the whole population. By increasing the overall IAP conversion of the new players through a conversion offer most convenient for them, the life-time value of the cohorts can be increased as converted players typically more likely continue to monetize after the initial conversion, linking the goal of the experiment to the more high-level business goals. On top of the overall impact to IAP conversion percentage, offer conversion was measured as a more detailed metric to inform how much conversion (CVR) through that specific entry point were affected. The overall IAP revenue was another key metric evaluated to

determine if the change in offer conversion translated to actual business impact, and didn't cannibalize revenues from other conversion entry points.

On top of these three metrics, other guardrail metrics such as average revenue per daily active user (ARPDAU), second IAP conversion percentage, daily IAP conversion percentage, average revenue per user (ARPU), average revenue per paying user (ARPPU) and number of purchases per user were measured , but are not the centerpiece of the analysis. The experiment was ran on both Android and iOS platforms in all countries, on new users first seen on the game after the initialization of the A/B test. As per standard A/B testing procedure, the users were automatically randomly distributed to the experiment groups and continued to stick in the group for the duration of the experiment, continuing to receive the same configuration. The IAP conversion percentage within new users is quite small to begin with, thus requiring a larger sample size to determine a difference between the groups. Consequently, the collected total sample size for the test was approximately 181k users per group.

The A/B test analysis was conducted according to the standard analysis practices in the company, in which binomial metrics are assessed by fitting Beta-binomial distributions (see e.g. Gupta & Nadarajah, 2004) with naive uniform prior to the sample data, and similar statistics are calculated for continuous metrics by using bootstrapping approach for random sampling with replacement to estimate the underlying distribution (see e.g. Efron & Tibshirani, 1993). The data is assumed to be normally distributed. The key criteria used to assess the results are the lift in comparison to the control and the probability that the variant is better than control. The statistical power assessments are already done when deciding the sample sizes for the experiment.

Results of the baseline conversion offer A/B test are presented in Table 10 along with the guardrail metrics in Table 11. The results are reported here as percentage changes compared to the control group, with the confidence level of variant being better presented in square brackets for the key metrics. The results presented in the tables measure the difference between metrics in tier 1 countries, an internal defined set of countries that comprise most of the generated revenue, which the company uses as a standard practice to eliminate noise from countries with mainly non-monetizing users and consequently measure monetization differences more

accurately. The sample size from tier 1 countries was approximately 44k users per group. Interpreting the confidence level, 0.95 or above is signaling statistical significance for variant being better, while 0.05 or below oppositely indicating a statistically significant value for variant being worse. Moreover, values over 0.80 can be interpreted to show the variant likely being better, and values under 0.20 to signal the variant likely being worse. IAP revenue was capped in the analysis at gross 300 USD per player to moderate outliers skewing the results. Any novelty effects on different offer configurations are intrinsically ruled out by the constraint of running the test on new users only.

Table 10. Results of the baseline conversion offer A/B test

	Sample size (tier 1)	Offer CVR	IAP CVR	IAP revenue
Control	44789			
Variant A	44513	+12.35% [0.887]	-6.43% [0.027]	-4.58% [0.237]
Variant B	44451	+6.07% [0.722]	-5.98% [0.041]	-11.80% [0.051]
Variant C	44428	-0.26% [0.480]	-3.77% [0.141]	-3.46% [0.314]

[probability variant better]

Table 11. Guardrail metrics on baseline conversion offer A/B test

	ARPDAU	CVR 2nd	CVR daily	ARPU	ARPPU	Purchases per user
Control						
Variant A	-4.12%	-4.72%	-6.05%	-4.98%	+1.71%	-5.66%
Variant B	-10.96%	-0.81%	-5.87%	-11.89%	-6.33%	-9.00%
Variant C	-3.54%	-0.52%	-1.49%	-3.69%	-0.14%	+3.85%

The results of the A/B test show the offer conversion being positively impacted in Variant A and Variant B, although the results are not quite statistically significant. However, both the total IAP conversion and IAP revenue seem to have been negatively impacted by the variants with the guardrail metrics also supporting the findings, indicating these setups to be worse overall business wise. Generally it should be kept in mind that the percentage differences show as relatively high due to the base values being quite small, with Variant C in fact displaying quite similar performance as the Control. Regardless of the offer conversion seeing an uplift in the two variants, the findings support refuting the null hypothesis and the control group in the test was considered as the winning configuration due to the overall business implications.

The main interest and purpose of this traditional A/B test was to set a baseline to which to compare the MAB-based approach and the impact of it. More specifically, the best performing offer configuration found by running the A/B test was the one to be used as the control group in the follow-up experiment against MAB-based optimization. In the follow-up MAB-based experiment, the goal was to evaluate if and how can personalizing which offers to show via contextual bandit learning increase the offer conversion as well as total IAP conversion over the traditional A/B test and the globally optimal conversion offer. Setting the contextual bandit optimization as a variant group against the control in the test setup allows comparison between the two approaches, with the contextual bandit still optimizing independently per player between its four arms and the different offers.

The Online Cover contextual bandit algorithm used for the MAB-based experiment was configured to use offer conversion as a reward to optimize which offers to show. In addition, the context used includes a number of variables related to demographic data, behavioral statistics, game specific information such as the player's inventory, as well as world context such as time of day, day of the week or calendar holidays. Before running the actual contextual bandit test on real users, the algorithm was first offline validated by running an offline simulation on a simulated player population with different attributes attempting to replicate a real-world environment. The Online Cover algorithm was simulated against a few other well-performing algorithms, as well as an oracle and pure random selection (Figure 11). The results of the offline simulation verified the implemented Online Cover algorithm worked technically as well as was further reassuring the claimed efficiency in minimizing regret.

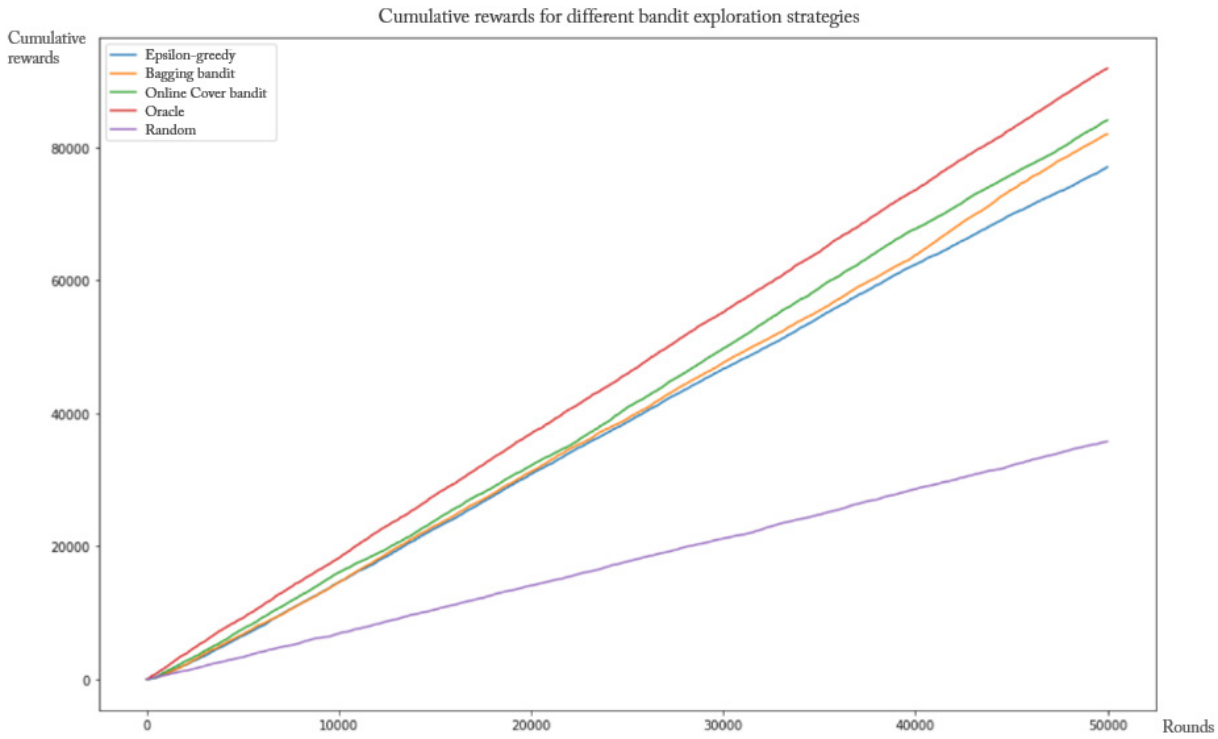

Figure 11. Algorithm offline simulation results

For the actual experiment, targeting included new users first seen in the game after initialization of the test from both Android and iOS platforms and from all countries, similarly to the prior A/B test. Further, the same four conversion offers were used as the four arms of the bandit, with the bandit optimization approach being compared to the control which used the global optimum found with the traditional A/B testing approach. The null hypothesis H_0 for the experiment can therefore be stated as "personalization of conversion offers with contextual bandit doesn't affect the key metrics of offer conversion, overall IAP conversion and IAP revenue compared to serving the offer globally", and accordingly alternative hypothesis H_1 being that "contextual bandit personalization of offers affects the key metrics compared to serving the offers globally". The results between the control group and the contextual bandit were analyzed using the same principles as described with the first A/B test analysis. The same set of key decision and guardrail metrics were also used to that of the prior conversion offer A/B test. Total collected sample size for the test was approximately 244k user per group.

The results are presented in Table 12 and Table 13 in the similar format as previously, with the tier 1 countries' sample size landing at approximately 57k users per experiment group. IAP

revenue per player was capped for the analysis again at gross 300 USD per player, and any novelty effects can again be ruled out by the test having a fresh set of new users collected as the sample.

Table 12. Results of the contextual bandit approach A/B test

	Sample size (tier 1)	Offer CVR	IAP CVR	IAP revenue
Control	57166			
Variant A - Contextual bandit	56648	+8.35% [0.974]	+3.90% [0.913]	+4.22% [0.756]

[probability variant better]

Table 13. Guardrail metrics on the contextual bandit approach A/B test

	ARPDAU	CVR 2nd	CVR daily	ARPU	ARPPU	Purchases per user
Control						
Variant A - Contextual bandit	+3.65%	+3.89%	+4.39%	+3.96%	-0.10%	+1.87%

The results of the test display a statistically significant uplift on offer conversion and an uplift to overall IAP conversion with a high probability when personalizing the offers with contextual bandit approach. This time with the bandit approach, IAP revenue and total IAP conversion don't seem to be cannibalized, although the metrics don't reach statistical significance. However, the guardrail monetization metrics also support the overall monetization to be likely positively impacted by consistently displaying an uplift from the duration of the experiment. Hypothesis H_1 can thus be determined being supported. ARPPU is likely being slightly negatively impacted in the results due to the variant converting and sustaining more spending users, resulting in a higher number of players that spend lower amounts to the game and thus decreasing the average revenue per paying user.

Further analyzing the contextual bandit reveals how the bandit has distributed the offers. Based on the results, it seems the bandit has recommended the offer quite versatilely to players (Figure 12). There is quite a bit of hourly and daily fluctuation in the recommendation, which could be mostly explained by time of day, type of traffic arriving at different times, the players' sessioning and thus the players' varying consumption and inventory of items in the game. Seeing each of the offers have continued to be served until the end of the experiment, the bandit seems to have determined based on rewards each of the offers to be valuable to some group of players.

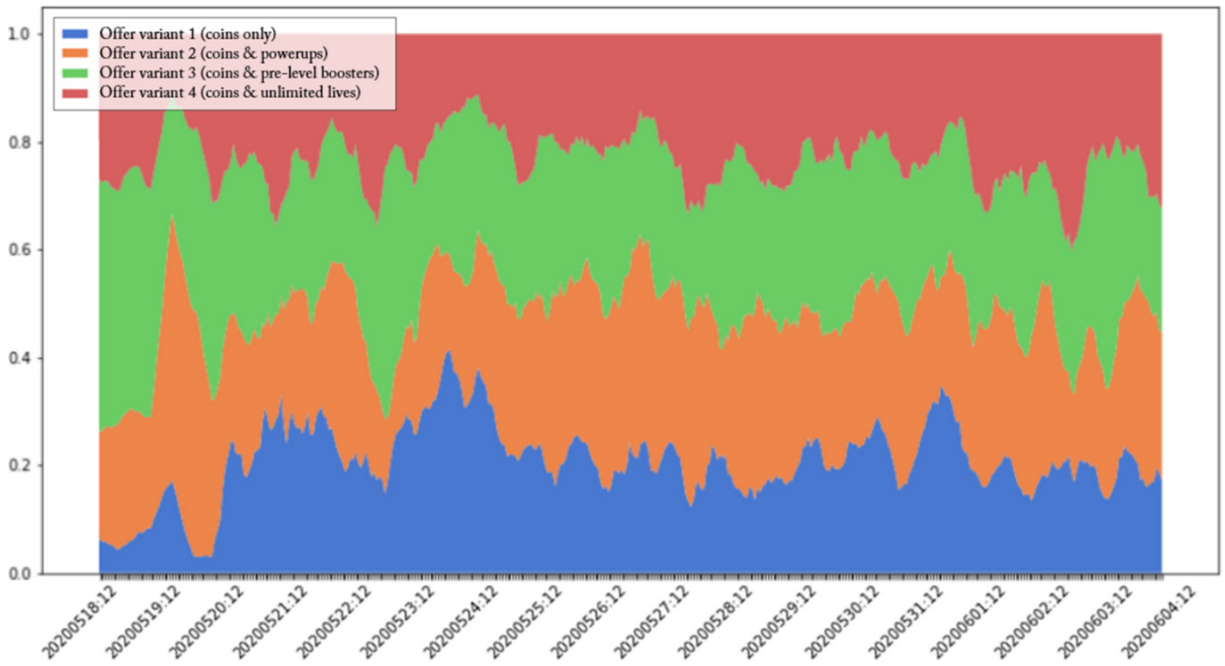

Figure 12. Distribution of recommended offers in the contextual bandit

Below on the next page, Figure 13 aggregating the distribution of offers from the duration of the experiment further shows that the bandit algorithm has distributed offers quite similarly both globally and in tier 1 countries, so the tier 1 sample is also very much representative of the whole group. Only the share of players who have been shown the conversion offer with coins and pre-level boosters can be observed to be slightly higher globally than in tier 1. The offer most commonly shown by the contextual bandit can be observed to be the coins and powerups offer, with the coins only offer surprisingly the least shown. Looking back the Figure 12, this seems to be mostly due to lower recommendation rates later down the line in the experiment.

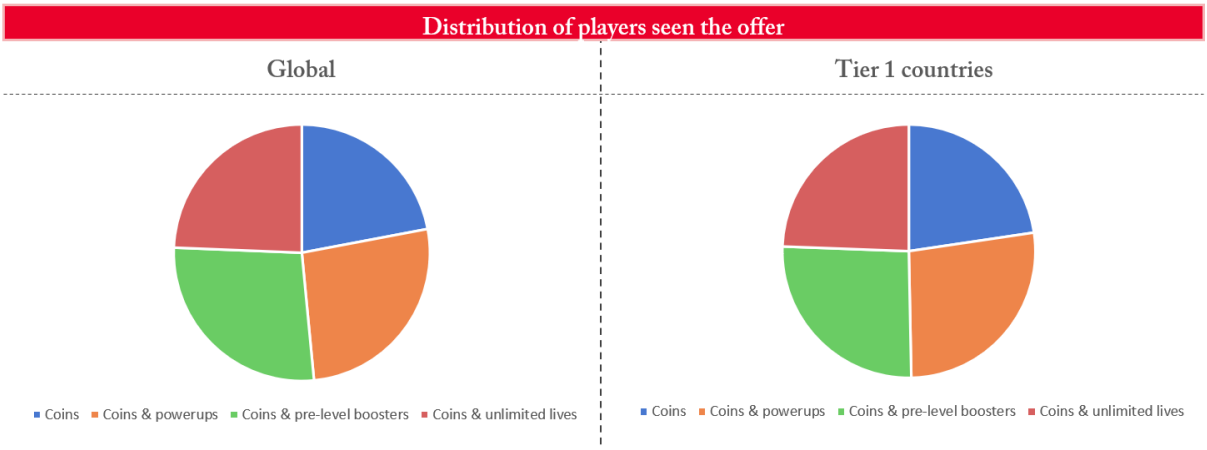

Figure 13. Total distribution of recommended offers in the contextual bandit

Figure 14 further displays the distribution of realized number of conversions for each type of offer in the contextual bandit as well as their conversion percentages. The distributions look again quite similar between global and tier 1 countries, which in this case results from most of the conversions globally in fact coming from tier 1 countries. The proportional difference in offer conversion percentages between global and only tier 1 can also be observed from here.

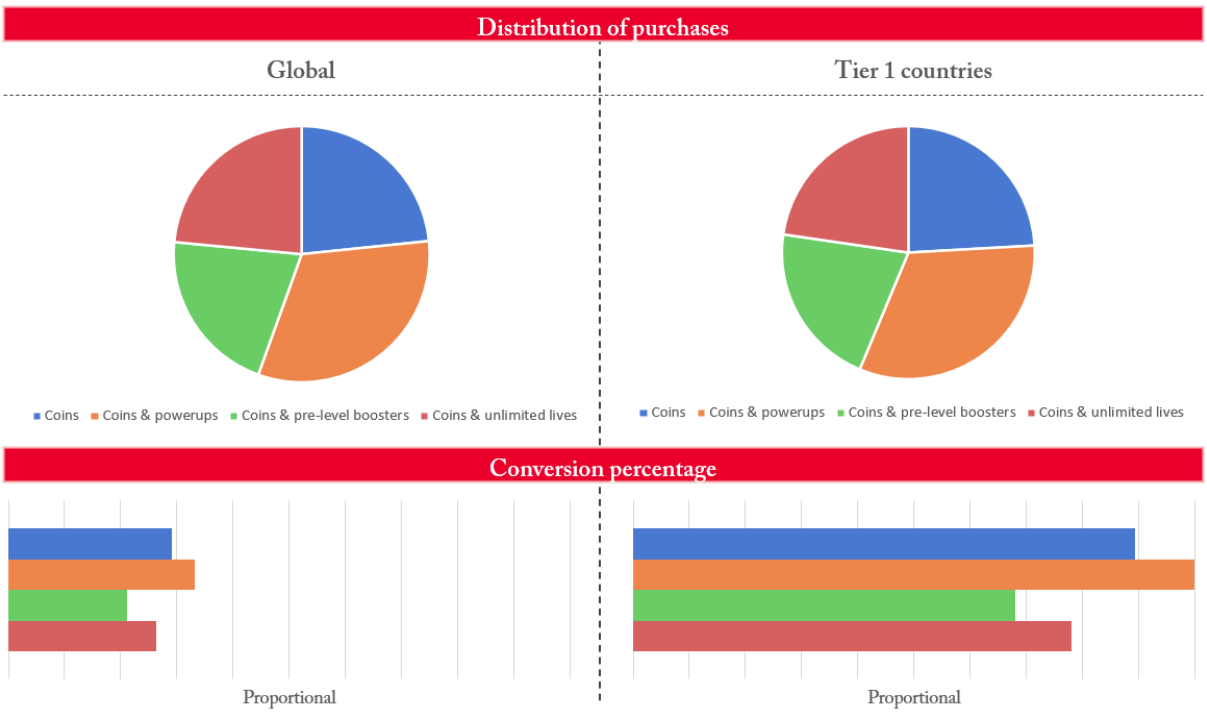

Figure 14. Overview of conversions in the contextual bandit

Overall, the most successful bandit's recommendations in terms of conversion rate have been with the coins and powerups, with the previous test's winning control offer with only coins following closely on the conversion rate performance. These two offers also have the highest number of purchases made on them. Looking at how the offers were served, the importance of all the contextual variables which the bandit utilized on deciding on the offer recommendation were additionally measured. From the dataset the most important contextual variables in personalizing the offer to be shown can be observed to be related to both the users' behavior in the game as well as known factors before downloading the game such as how they found it.

In the analysis there were also few hiccups noticed with the implementation of the bandit. First, it should be noted that a small latency was observed with the system and how the conversion offer was being shown by the server. While the control group served the offer at the exact time when the player fails the first time and returns to the main hub screen, in the bandit group a considerable amount of players have first seen the offer one to two minutes after the serving request as visible in Figure 15.

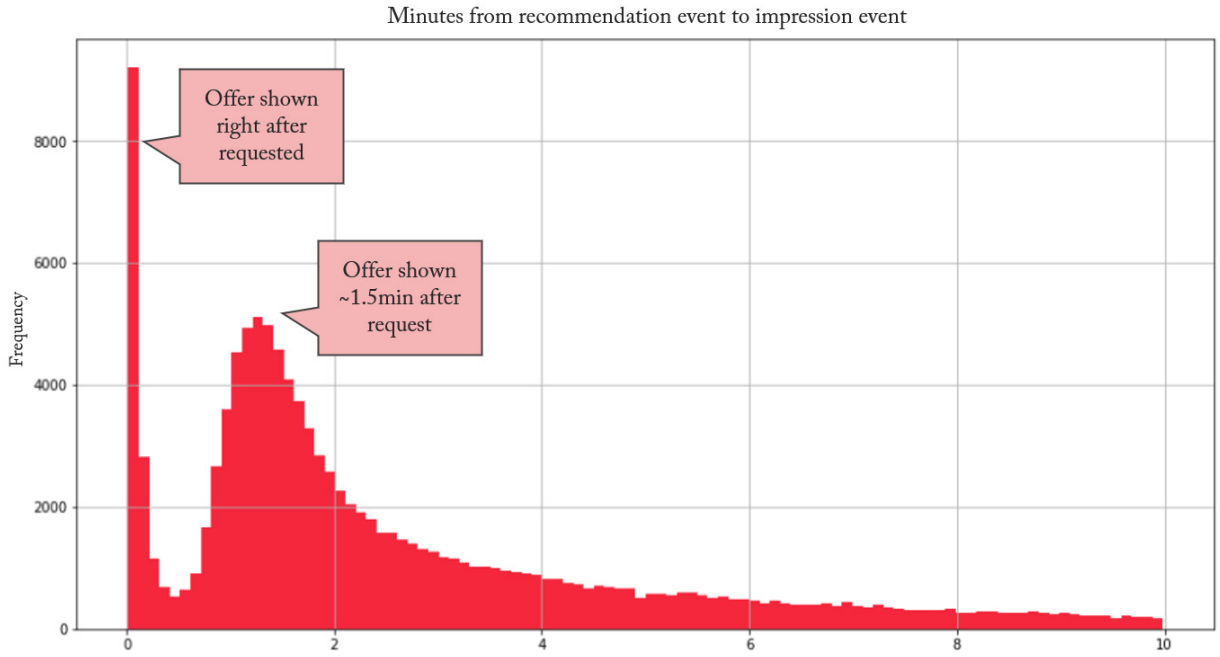

Figure 15. Time from bandit offer recommendation request to impression

The spike observed at 1.5min between the serving request and the player seeing the offer is most likely caused by the offer not being returned in time by the server before the player has entered the next level from the hub screen. Thus, the players are seeing the offer in these cases only after playing one more level, coinciding with the one to two minute delay on the impression. The steady long tail on the figure on the other hand is nothing exceptional to pay attention to, as it is a result of the player staying in the level they made their first fail in, and attempting it multiple times without entering the main hub screen, hence not seeing the offer before exiting the level. The delay and the offer being shown in some cases to users' on slightly different stage than in the control group might slightly impact the reliability of the results.

When analyzing the test it was also observed that the bandit has been able to recommend a different offer if the player has not converted and the cooldown of the conversion has passed to serve the conversion offer again. To ensure best comparability between control and the bandit approach and thus reliability of the results, optimally the player would've received the same offer the bandit has first recommended to them, as is the behavior in the control group. The number of conversions occurring from the conversion offer being shown to the players again later however is then already lower, as many players have converted through the first conversion offer or other IAP:s available in the shop by that point. It is also worth noting that the group of players who have seen a different offer and then converted to it might have behaved similarly even if shown the same offer the second time, as was intended. Yet this introduces a second factor to the experiment for the players that have received a different offer in the second conversion offer round. The ramifications of these will be further discussed in the next chapter along with discussing the results of the test.

5 RESULTS & DISCUSSION

Before the execution of the case study, the literature review suggested certain type of behavior and certain types of benefits should be realizable by engaging with contextual multi-armed bandit experimentation. However, with A/B testing it has been noted that while there is previous research regarding the topic, the research has almost predominantly focused on web domain and it has remained unclear to what extent this is applicable for companies outside of this domain (Holmström Olsson et al., 2017). So far it is not evident that methods described to work well for companies such as Microsoft or Google are equally applicable and useful for smaller companies and other domains (Schermann et al., 2018). A similar issue can be argued to currently hold even more true regarding the research on multi-armed bandit approaches. Furthermore, the limited amount of practical evidence on MAB approaches has particularly been focused on web domain and website optimization. Meanwhile previous case studies have established that although academic research suggests MAB algorithms providing several benefits, the MAB approach may not in all cases provide and realize the expected benefits for companies (Issa Mattos et al., 2019).

The executed case study sought to provide practical evidence within the application domain regarding experimentation with a machine learning based approach, and compare the results to the traditional A/B test experimentation to evaluate the behavior against existing theory. The results from the case study elaborated in the previous chapter depict in this case largely expected behavior for the contextual multi-armed bandit in personalizing conversion offers, showcasing that the bandit approach can minimize long-term regret compared to a fully random A/B testing approach. As suggested by the literature, the contextual bandit allocation seems to have found groups with different preferences and better performing variants to serve for each of the groups in order to increase the cumulative reward. For example, if a player is seen to use the items part of a specific offer more or if the player has previous experience of similar games, the player might consider differently the value of a particular offer compared to other players.

From the contextual bandit's results, out of the four different arms of the bandit the offer with coins and powerups bundled in appears as the arm with the best performance in terms of offer

conversion. In the first experiment of the case study set to discover the best performing offer via traditional A/B testing, the same offer configuration performed the best in terms of offer conversion, but saw a negative impact to the overall business performance. Here it must be noted that majority of the IAP conversions in the game happen through an in-level flow directing players to the in-game shop, where the offers were not available. Thus, seeing an offer with more items before that occasion might introduce additional cognitive load for the purchase decision. On the other hand, the perceived value of other IAP products for some players could be decreased if the offer contains considerably more value than other standard IAPs in the game. Thus, it might be that the overall IAP conversion and business performance is impacted more from showing offers with more items to all of the new players, compared to what is the amount of total lost amount of conversions from showing coins only offer to the players that would've in fact found the additional items more compelling and affecting their decisions. It is hard to determine all the possible causes behind the observed change in player behavior, and there might be many other reasons that could explain the different behavior as well.

The contextual bandit test showed an overall improvement on the monetization metrics, including offer conversion, compared to serving the globally optimal offer in the control group. While the differences in overall IAP revenue or IAP conversion are not statistically significant, they still show great confidence over the variant performing better regarding the said metrics. The data between tier 1 and global show a coherent picture regarding the bandit's performance. Moreover, each of the guardrail metrics in the test showed quite expected uplifts compared to the changes in key metrics. With the negative difference in ARPPU being so marginal despite the high conversion uplift, it would furthermore suggest that the players have been content with their offer purchases and have continued to spend. As for the scale of the impact, an average of 8.35% lift in offer conversion and 3.90% uplift in IAP conversion can be considered as relatively good results when comparing to the prior A/B test. However, as mentioned in the results of the case study, the execution ended up containing two minor flaws which affect how the results can be interpreted.

The latency observed in serving the offer and the resulting delay in players seeing the offer in theory could have an effect on offer conversion, introducing a question regarding its effect to the validity of the results. However, in this case showing the offer a level later to some of the

players due to latency is more likely to affect the conversion metrics negatively if anything. The hypothesis on the impact of the latency however can't be fully verified with the data, and can be argued to introduce some minor degree of unknown impact to the results to either direction. Then again I don't see this to introduce an alarming issue by any means for the overall validity of the case study, seeing the goal of the experiment in the first place wasn't to evaluate the exact extent of the uplifts, but rather validate theoretical assumptions of the observable behavior, benefits and implications of experimentation with the contextual bandit approach.

The second anomaly observed in the test involved the bandit being able to serve a different offer than was first shown to the player when they are able to receive the conversion offer again. In the control group, the players continued to receive the same offer later on if they hadn't converted, which accordingly was the intended behavior for players in the contextual bandit group as well. The number of conversions that would be due to this however likely is on the lower side due to the lower number of impressions, together with the probability of the player buying the offer regardless of its contents. This to an extent limits the impact to the results and yields it not to introduce a major threat to the overall validity of the study. However, the distribution of offers shown is likely affected by this, which could be one factor explaining the low proportion of coins only offer in Figure 12 and Figure 13. Overall, the bandit being able to serve a different offer in this case undoubtedly has at least some degree of impact and valid reliability concerns when it comes to accurately measuring the scale of the uplift.

Being impartial, the two issues encountered in the experiment affect the reliability and validity of the case study results in terms of the observed lifts and exact business impact. However, looking at the validity and bandit's behavior more closely, the execution and general results of the study are sound when comparing to the prior implications from theory, and support the existing research on contextual multi-armed bandits. The case study also demonstrates the chance of unintentional behavior occurring within an experiment, which is why it is critical to do a deeper analysis on the experiments to spot any anomalies. When implementing any new system this fact is only greatly emphasized, even when the company already has established experimentation practices in place. In terms of generalizability, the study supports that findings suggested by existing research can be replicated within application domain in an appropriate use case with real end users, leading to support generalizability to other similar cases. Being the

first identified researched applied implementation of contextual MAB within application domain, similar research undertakings and more practical evidence from different experiment cases optimizing different factors would need to be gathered to extend the analytic generalizability.

In prior research feature experimentation in general has proven useful for optimization of product performance (Holmström Olsson et al., 2017), with companies utilizing A/B testing reportedly generating a profound impact on their annual revenue (Fabijan et al., 2017b). With machine learning entering the field, new opportunities are opened up in optimizing the software product for the end users. Instead of directly replacing most of the existing A/B testing practices, machine learning and multi-armed bandits first and foremost enable new use cases and opportunities for feature experimentation.

Generally speaking, three distinct novel use cases for multi-armed bandit experimentation were identified from current research. First, MAB-based approaches are beneficial when there is a limited time window for experimentation and exploitation. One such example could be optimizing between different seasonal campaigns or sales lasting only a few days or weeks, with the campaign often being already over before any results from a traditional A/B test can be obtained. With the bandit approach, the best performing variants in the test can begin to be exploited right from the beginning of the experiment.

Second, different combinatorial experiments or other experiments with a high number of variants can be ran more effectively by exploring the high number of variants first with the bandit approach, which is then followed by a traditional A/B test experiment with the few most promising looking variants from the bandit experiment pitted against the control group to determine the best performing variant with statistical confidence. These types of experiments usually involve a number of smaller changes that are interlinked between each other, one example for instance could be optimizing layout or appearance of number of UI elements across the system.

Third, contextual bandits enable personalization or optimizing to find local optimums for different groups of users instead of serving the same configuration for all. This can be useful in

cases when there are great differences in user preferences, the user base is rather heterogenous or it is desired to constantly react to potential changes in user preferences. For instance, reacting to seasonality or changes in traffic affecting demand in online stores could be a case where the optimal variant could change within non-predetermined periods of time, and changing the user allocation to better performing variants could drive up the business performance.

On top of the three distinct use cases, the results from theory and practical evidence so far in general show promise in using bandit optimization especially for use cases where there is a simple problem to find the optimal configuration for. In these cases MAB experiments could particularly solve limitations of traditional A/B testing related to limited number of users and the duration of the tests, as well as cumbersome personalization. On more complex problems MAB approaches face the difficulty of modelling the reward as a single well-rounded variable for the agent to use as a reward, without incorrectly the agent learning to exploit something in the modelling. Overall based on the theoretical findings and their discussed implications on this thesis, a generalizable framework for bandit optimization can be suggested as in Figure 16.

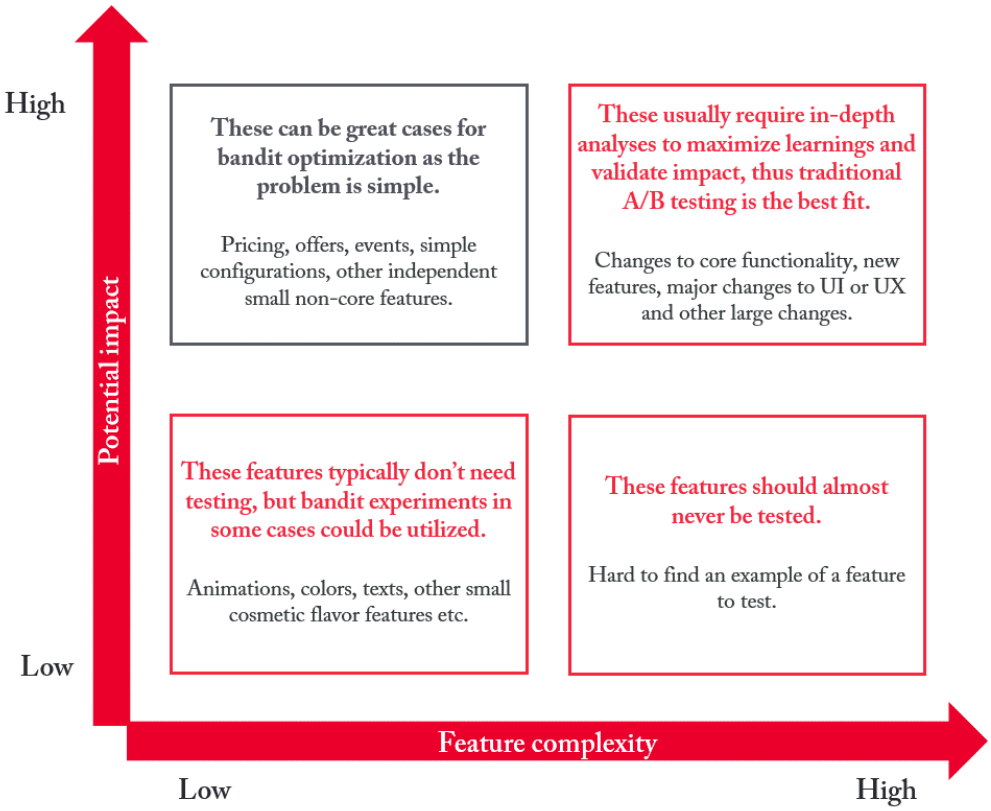

Figure 16. Framework for bandit optimization use cases

The examples in the framework are mainly developed here with software and specifically game applications in mind, but the general framework is applicable within all domains as the underlying theory and assumptions apply. The subjective division in the framework assumes features in two axes based on their complexity and potential impact, which ultimately are the main criteria on how the change should be tested. In some cases, it isn't enough to know that B is better than A, but instead one needs to know exactly by how much and under which circumstances. In such cases when there is a need for additional insight and deep analysis on the impact of the change, or a desire to rank the performance of all the variants (Hickey, 2019), traditional A/B testing simply still is the best solution.

Other than the three distinct use cases detailed earlier, multi-armed bandits and contextual multi-armed bandits can be in general great for simple optimization cases where the saved time and user allocation may prove to more useful than learning the details of how the change affected the metrics. Furthermore based on the research findings thus far, this generalization can be extended to apply to most typical use cases in both web and application domain, with the choice of bandit algorithm rather potentially playing a bigger role. Ultimately however it comes down to the line being hard to draw between the simple enough experiments where the learnings probably are insignificant and the experiments where it is valuable to build learnings and additional hypotheses from the analysis. Opting for a traditional A/B test comes with the cost of longer test duration and opportunity costs from not being able to exploit the winning variant earlier, but on the other the gained insights can be immensely more valuable.

With online software products remaining in a perpetual state of development throughout their lifecycle, there likely is always more ideas and features to test and iterate on than there is capability to. Regarding low impact changes, iterating on features and areas that have been proven to yield an impact is often times more beneficial than engaging on an experiment with changes projected or known to have low potential impact. However with that being said, the impact of a given type of change should never be underestimated before proven otherwise. Depending on the type of software or application, even some of the smallest changes may yield large impacts, as can be highlighted in the case of the well-known Bing's color A/B test. In 2013 Bing ran a set of experiments on font colors on their browser, introducing merely hue changes to the existing palette. On the contrary to what probably most would expect from such

a change, the experiment results showed the difference between winning variant and current color set to improve monetization by over $10M annually. With the results rightly met with skepticism the experiment was replicated, generating again a similar result and proving the high potential impact from simply altering the color set on the fonts. (Kohavi et al., 2014)

Then there are separate cases outside the framework where some features can't be A/B tested due to constraints on their implementation and parameterization. However, the Google Play Store on Android has the capability to roll out a new version available of an application to only a percentage of users, applying to both existing and new users downloading the app. By setting up a non-randomized targeting based on app version with control group being populated by players that are still in the old version and a variant group with the players who have the new version, it is possible to conduct A/B test-like evaluation on new features with the randomization and serving of different feature set between groups effectively handled by the staged roll out. Meanwhile Apple hasn't supported such staged roll outs on their App Store, limiting the workaround only to be applicable within Android platform at least to this date. (Xu & Chen 2016) The workaround also can't be applied with machine learning as instead of assuming pure random distribution between groups, the agent handles variant assignment by observing the rewards from different arms. Using the workaround also means rolling out any other features with the new releases isn't possible for the duration of the test. Albeit being quite limited and rarely used, the technique enables the possibility to assure that a big new feature which can't be A/B tested affects the metrics roughly in a way as expected.

Based on what has been mentioned, it is safe to say that A/B testing will continue to coexist with the machine learning approaches as a standalone practice with its own purposes. Moreover, traditional A/B tests can be used as support tool for MAB experiments. With basic MAB experiments, Hejazinia et al. (2019) suggested to run a follow-up A/B test to validate the impact of the winning variant of the MAB test also on other metrics. Running follow-up A/B tests can be a good practice to validate the overall impact of the change especially with bandit optimization tests that may present a trade-off between some metrics. However, it partially defeats the bandit's benefit of faster exploitation of the best performing variants and saved number of users to obtain the results, and thus doesn't make sense in case of tests with low

number of variants. In these cases opting for an A/B testing only approach usually gets the job done quicker.

With contextual bandits, a follow-up A/B test is not feasible due to the intended long or infinite running time of the bandit. In the case of contextual bandits, a prior A/B test could be run instead to validate none of the variants degrade the user experience on other metrics, a practice surprisingly not being brought up in the existing literature. If there is a possibility or risk concerned with the trade-off between two or more metrics, a traditional A/B test done prior to the contextual bandit test could ascertain none of the variants severely affect for example user retention in favor of monetization. However it should be noted that results on the variants might differ globally and within personalized segments in some cases quite drastically, so the prior A/B test isn't a silver bullet to avert the risks. One of the shortcomings of the bandit approach indeed is that the bandit requires a single metric that is sensitive to change in a short period of time, which in turn may not correlate with a longer-term metric(s) used as an evaluation criterion for success (Hejazinia et al., 2019). Moreover, having the requirement of a single metrics creates difficulty in evaluating situations when there is a trade-off between metrics.

Typical case in feature development is optimizing for an optimal trade-off between retention and monetization, which currently isn't possible to evaluate using MAB testing. Instead of optimizing using retention and monetization as rewards, the MAB approach would require optimizing towards life-time value (LTV) by developing an LTV model, adding uncertainty with a mathematical model that has to be validated and balanced to model the trade-off in a desired way. Trying to model a reward based on multiple metrics further easily leads to the agent finding a loophole in the logic and exploiting it, a behavior also known as reward hacking. Complex models additionally make it harder to verify the MAB is choosing the arms in a desired way.

There is some existing research on multi-objective applications of MABs that are trying to solve the multiple rewards issue. The current research on multi-objective bandits according to Tekin and Turgay can be categorized under two approaches, the Pareto approach and the scalarized approach (Tekin & Turgay, 2018). However, these extensions seem to also still very much be

academic research implementations described only through simulations and datasets, still some way from any potential industrial use (Issa Mattos et al., 2019). The topic deserves further research in the future, as establishing a robust multi-objective optimization algorithm could potentially solve some of the current limitations of MAB experimentation and broaden the possible use cases for it. When it comes to future research, another topic deserving further attention is experimentation models designed for MAB-based experiments. Current experimentation models assume a fairly rigid process from hypothesis to launching an experiment, analyzing it and applying the new baseline. However for example in case with the contextual bandits, ideally there is no intervention to the experiment before a possible follow-up iteration on the same feature or change is launched. In conjunction with developing experimentation models, for future research there is still also a general lack of evidence regarding MAB-experimentation practices, especially from the application domain.

Some other questions are also open on applied use when there are a number of contextual bandits run parallel. For example if multiple contextual bandits run parallel in the system optimizing different parameters and features, a challenge comes to track each and spot any possible issues that might emerge over time. In theory one possible solution could be to run A/A/B, A/A/B/B or A/B/B test setups to keep validating similar experiences continue to output similar performance, which in case of MABs however to an extent slows down learning in the early phases. Handling detection of anomalies in ongoing bandit experiments and possible automation of it is a topic likely worth research in the future, as many authors conclude detection of issues and handling error cases being one of the key aspects for an efficient and sustainable experimental approach (Tang et al., 2010; Kohavi et al., 2014; Fabijan et al., 2017b).

Lastly, some of the identified limitations on the experimentation in general are related to managing and coordinating the experiments and the whole experiment landscape within the product, with experimentation requiring decent amount of manual effort to get from hypothesis formulation to analyzed data and conclusions. Regarding automation, MABs provide a step closer towards fully automated experiments as the methodology itself leads to conclude the winning variant without a separate analysis. Although other pieces of the puzzle and creating an integrated system likely still have still a long way to go (Issa Mattos et al., 2017), automation within experimentation systems is likely a topic worth following in the future.

6 CONCLUSIONS

Software products in the present day are continuously developed and optimized throughout their lifecycle to deliver additional value to the end users. Correspondingly, the importance of validating the value and impact of new features and changes made to the product in an unbiased and robust way has been recognized widely throughout the industry. A/B testing and continuous experimentation provide to means to conduct evidence-driven evaluation of changes to the product underpinned by statistical analyses on the behavior of real users, providing insights that will help developing the product to the right direction. Meanwhile, machine learning techniques have lately become more than just a buzzword in the industry, enabling new opportunities and benefits to be realized in the area of feature experimentation and product development. Out of the various machine learning styles, reinforcement learning and more specifically multi-armed bandits are the suitable techniques for experimentation as they are capable of modelling the objective of discovering the best performing variant from a number of different options with unknown performance.

The literature identified number of distinct use cases in which multi-armed bandits and contextual multi-armed bandits can deliver new benefits where A/B testing has been found to be unsuitable. The basic stochastic bandits operate with largely similar objective and purpose as traditional A/B tests, converging to find the one optimal variant out of the tested options. The basic stochastic bandits however allow minimizing time and sample size required to determine the winning variant as the algorithms start allocating more traffic to the well performing ones while avoiding to distribute any more traffic than necessary to the bad variants. However, using bandit optimization limits the capability to conduct analyses on the behavior and impact of the different variants and thus limiting its potential use cases. The other further limiting aspect of the bandit approach is the requirement for a single metric that is sensitive to change in a short period of time, meaning the bandit can't evaluate trade-offs between multiple metrics.

With the introduction of contextual variables and context, the contextual bandit algorithms are able to personalize the variants by addressing the estimation of optimality through each user's context. This shifts the paradigm of experimentation as it entails the experiment should be kept

indefinitely running to keep optimizing the variants for the users. Contextual bandits are able to address the limitation with personalization in traditional A/B testing, and thus provide intriguing new opportunities for experimentation. From the different applications of the multi-armed bandits in experimentation, contextual bandits are likely to most fundamentally change how feature experimentation and product development is approached in many organizations.

The executed case study provided further practical evidence from application domain on the benefits of utilizing a contextual bandit approach on a suitable use case to personalize content for the users over a one-size-sits-all approach with traditional A/B testing. As a contribution to existing literature the thesis also propounded a framework for the use of multi-armed bandits in experimentation and optimization, suggesting how feature complexity and the potential impact of a change can be used to determine the choice of experimentation method. The framework paired with guidelines for choosing between experimentations methods presented at chapter 3.5 provide a clear indication on which cases to lean towards bandit experimentation.

The findings of the thesis and the suggested framework also establish that traditional A/B testing still holds true under the area which it is first and foremost designed for – generating statistically significant validation and insights about the change in behavior introduced by a new feature or configuration. Accordingly, it remains a more secure and robust way to determine winning variant under normal circumstances, where determining the best performing variant with high confidence is more important than saving few days or weeks in experiment duration. There remains a number of limitations regarding both the A/B testing as well as the multi-armed bandit approaches, and choosing the experimentation method when both options are available comes down to weighing the inherent trade-offs.

As a whole, the thesis presented a thorough overview of the existing A/B testing practices and evaluated how the multi-armed bandits will reshape the experimentation within product development, presenting the use cases, benefits and limitations associated with them. The findings of the thesis will hopefully serve to help organizations steer their practices towards better feature experimentation, as well as provide a solid resource and foundation to use for future research.

REFERENCES

Agarwal, A., Bird, S., Cozowicz, M., Hoang, L., Langford, J., Lee, S., Li, J., Melamed, D., Oshri, G., Ribas, O., Sen, S. & Slivkins, A. (2016). Making Contextual Decisions with Low Technical Debt. arXiv:1606.03966v2.

Agarwal, A., Hsu, D., Kale, S., Langford, J., Li, L. & Schapire, R. E. (2014). Taming the monster: A fast and simple algorithm for contextual bandits. *31st International Conference on Machine Learning, ICML 2014*, 5, pp. 3611–3619.

Agrawal, S. & Goyal, N. (2012). Analysis of thompson sampling for the multi-armed bandit problem. *Journal of Machine Learning Research*, 23, pp. 1–26.

Agrawal, S. & Goyal, N. (2013). Thompson sampling for contextual bandits with linear payoffs. *30th International Conference on Machine Learning, ICML 2013*, (Volume 28), pp. 1220–1228.

Asano, Y. M., Rupprecht, C. & Vedaldi, A. (2019). A critical analysis of self-supervision, or what we can learn from a single image. arXiv:1904.13132v3.

Auer, P., Cesa-Bianchi, N. & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2), pp. 235–256.

Bakshy, E., Eckles, D. & Bernstein, M. S. (2014). Designing and deploying online field experiments. *WWW 2014 - Proceedings of the 23rd International Conference on World Wide Web*, pp. 283–292.

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J. & Thomas, D. (2001). Manifesto for Agile Software Development. Online. [Last accessed: 06.08.2020]. Available at: http://agilemanifesto.org/principles.html

van Belle, G. (2002). Statistical rules of thumb. Hoboken: John Wiley & Sons, Inc.

Bietti, A., Agarwal, A. & Langford, J. (2018). A Contextual Bandit Bake-off. arXiv:1802.04064v4.

Box, J. F. (1980). R. A. Fisher and the Design of Experiments, 1922-1926. *The American Statistician*, 34(1), pp. 1–7.

Bromley, D. B. (1986). The Case-study Method in Psychology and Related Disciplines. Chichester: John Wiley & Sons, Inc.

Bubeck, S. & Cesa-Bianchi, N. (2012). Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1), pp. 1–122.

Burtini, G., Loeppky, J. & Lawrence, R. (2015). A Survey of Online Experiment Design with the Stochastic Multi-Armed Bandit. arXiv:1510.00757v4.

Chapelle, O., Schölkopf, B. & Zien, A. (2006). Semi-Supervised Learning. London: MIT Press.

Denrell, J. (2005). Selection bias and the perils of benchmarking. *Harvard Business Review*, 83(4), pp. 114–119.

Dickson, B. (2020). Self-supervised learning The plan to make deep learning data-efficient. TechTalks. Online. [Last accessed 06.08.2020]. Available at: https://bdtechtalks.com/2020/03/23/yann-lecun-self-supervised-learning/

Dmitriev, P., Frasca, B., Gupta, S., Kohavi, R. & Vaz, G. (2016). Pitfalls of long-term online controlled experiments. *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pp. 1367–1376.

Dmitriev, P. & Wu, X. (2016). Measuring metrics. *International Conference on Information and Knowledge Management. Proceedings, 24-28 October*, pp. 429–437.

Drugan, M. M. & Nowe, A. (2013). Designing multi-objective multi-armed bandits algorithms: A study. *Proceedings of the International Joint Conference on Neural Networks*, pp. 1-8.

Eden, D. (2017). Field Experiments in Organizations. *Annual Review of Organizational Psychology and Organizational Behavior*, 4(1), pp. 91–122.

Efron, B. & Tibshirani, R. J. (1993). An Introduction to the Bootstrap. Boca Raton: Chapman & Hall/CRC.

Ehrenberg, A. S. C. (1974). The Teaching of Statistics : Corrections and Comments. *Journal of the Royal Statistical Society*, 138(4), pp. 543–545.

Eisenhardt, K. & Graebner, M. (2007). Theory Building from Cases: Opportunities and Challenges. *Academy of Management Journal*, 50(1), pp. 25–32.

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *Academy of Management Review*, 14(4), pp. 532–550.

van Engelen, J. E. & Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning*, 109(2), pp. 373–440.

Fabijan, A., Dmitriev, P., Holmström Olsson, H. & Bosch, J. (2017a). The benefits of controlled experimentation at scale. *Proceedings - 43rd Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2017*, pp. 18–26.

Fabijan, A., Dmitriev, P., Holmström Olsson, H. & Bosch, J. (2017b). The Evolution of Continuous Experimentation in Software Product Development: From Data to a Data-Driven Organization at Scale. *Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering, ICSE 2017*, pp. 770–780.

Fabijan, A., Dmitriev, P., Holmström Olsson, H. & Bosch, J. (2018a). Effective online controlled experiment analysis at large scale. *Proceedings - 44th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2018*, pp. 64–67.

Fabijan, A., Dmitriev, P., Holmström Olsson, H. & Bosch, J. (2018b). Online controlled experimentation at scale: An empirical survey on the current state of A/B testing. *Proceedings - 44th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2018*, pp. 68–72.

Fabijan, A., Dmitriev, P., Holmström Olsson, H. & Bosch, J. (2020). The Online Controlled Experiment Lifecycle. *IEEE Software*, 37(2), pp. 60–67.

Fabijan, A., Dmitriev, P., Holmstrom Olsson, H., Bosch, J., Vermeer, L. & Lewis, D. (2019). Three Key Checklists and Remedies for Trustworthy Analysis of Online Controlled Experiments at Scale. *Proceedings - 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice, ICSE-SEIP 2019*, pp. 1–10.

Fabijan, A., Dmitriev, P., McFarland, C., Vermeer, L., Holmström Olsson, H. & Bosch, J. (2018). Experimentation growth: Evolving trustworthy A/B testing capabilities in online software companies. *Journal of Software: Evolution and Process*, 30(12), pp. 1–23.

Fabijan, A., Holmström Olsson, H. & Bosch, J. (2015). Customer feedback and data collection techniques in software R&D: A literature review. *Software Business : 6th International Conference, Icsob 2015, Braga, Portugal, June 10-12, 2015, Proceedings, 210(210)*.

Fagerholm, F., Guinea, A. S., Mäenpää, H. & Münch, J. (2014). Building blocks for continuous experimentation. *1st International Workshop on Rapid Continuous Software Engineering, RCoSE 2014 - Proceedings*, pp. 26–35.

Fagerholm, F., Sanchez Guinea, A., Mäenpää, H. & Münch, J. (2017). The RIGHT model for Continuous Experimentation. *Journal of Systems and Software*, 123, pp. 292–305.

Farquhar, J. D. (2012). Case study research for business. London: SAGE Publications.

Gatti, C. (2015). Design of Experiments for Reinforcement Learning. Cham: Springer International Publishing.

Gittins, J. C. (1979). Bandit Processes and Dynamic Allocation Indices. *Journal of the Royal Statistical Society: Series B (Methodological)*, 41, pp. 148–177.

Gupta, A. K. & Nadarajah, S. (2004). Handbook of Beta Distribution and Its Applications. New York: Marcel Dekker, Inc.

Gupta, S., Ulanova, L., Bhardwaj, S., Dmitriev, P., Raff, P. & Fabijan, A. (2018). The Anatomy of a Large-Scale Experimentation Platform. *Proceedings - 2018 IEEE 15th International Conference on Software Architecture, ICSA 2018*, pp. 1–109.

Hamel, J., Dufour, S. & Fortin, D. (1993). Case study methods. Newbury Park: SAGE Publications.

Hejazinia, M., Eastman, K., Ye, S., Amirabadi, A. & Divvela, R. (2019). Accelerated learning from recommender systems using multi-armed bandit. arXiv:1908.06158v1.

Hickey, J. (2019). Beyond A / B testing : Automation within testing in Adobe Target. White Paper, pp.1-7. [Last accessed: 06.08.2020]. Available at: https://www.adobe.com/content/dam/www/us/en/marketing/target/automation/pdfs/54658.en.target.whitepaper.automation-10.27.pdf

Hill, D. N., Nassif, H., Liu, Y., Iyer, A. & Vishwanathan, S. V. N. (2017). An efficient bandit algorithm for realtime multivariate optimization. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2017*, pp. 1813–1821.

Holmström Olsson, H. & Bosch, J. (2014). The HYPEX Model: From Opinions to Data-Driven Software Development. In: Bosch J. (eds) Continuous Software Engineering. Cham: Springer International Publishing, pp. 155–164.

Holmström Olsson, H., Bosch, J. & Fabijan, A. (2017). Experimentation that matters: A multi-case study on the challenges with A/B testing. In: Ojala A., Holmström Olsson H., Werder K. (eds) Software Business. ICSOB 2017. Lecture Notes in Business Information Processing, vol. 304. Cham: Springer International Publishing, pp. 179–185.

Hynninen, P. & Kauppinen, M. (2014). A/B testing: A promising tool for customer value evaluation. *2014 IEEE 1st International Workshop on Requirements Engineering and Testing, RET 2014 - Proceedings*, pp. 16–17.

Issa Mattos, D., Bosch, J. & Olsson, H. H. (2017). Your system gets better every day you use it: Towards automated continuous experimentation. *Proceedings - 43rd Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2017*, pp. 256–265.

Issa Mattos, D., Bosch, J. & Olsson, H. H. (2019). Multi-armed bandits in the wild: Pitfalls and strategies in online experiments. *Information and Software Technology*, 113(September), pp. 68–81.

Kaibel, C. & Biemann, T. (2019). Rethinking the Gold Standard With Multi-armed Bandits: Machine Learning Allocation Algorithms for Experiments. *Organizational Research Methods*, (June 2019), pp. 1–26.

Kharitonov, E., Drutsa, A. & Serdyukov, P. (2017). Learning sensitive combinations of A/B test metrics. *WSDM 2017 - Proceedings of the 10th ACM International Conference on Web Search and Data Mining*, pp. 651–659.

Kluck, T. & Vermeer, L. (2017). Leaky Abstraction In Online Experimentation Platforms: A Conceptual Framework To Categorize Common Challenges. arXiv:1710.00397v1.

Kohavi, R., Crook, T. & Longbotham, R. (2009). Online Experimentation at Microsoft. ThinkWeek Paper, pp. 1-16. [Last accessed: 06.08.2020]. Available at: https://exp-platform.com/Documents/ExP_DMCaseStudies.pdf

Kohavi, R., Deng, A., Frasca, B., Walker, T., Xu, Y. & Pohlmann, N. (2013). Online controlled experiments at large scale. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1168–1176.

Kohavi, R., Deng, A., Longbotham, R. & Xu, Y. (2014). Seven rules of thumb for web site experimenters. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1857–1866.

Kohavi, R. & Longbotham, R. (2017). Online Controlled Experiments and A/B Testing. *Encyclopedia of Machine Learning and Data Mining*, pp. 922–929.

Kohavi, R., Longbotham, R., Sommerfield, D. & Henne, R. M. (2009). Controlled experiments on the web: Survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1), pp. 140–181.

Kohavi, R. & Thomke, S. (2017). The Surprising Power of Online Experiments. *Harvard Business Review*, 2017/09, pp. 74–82.

Kubat, M. (2017). An Introduction to Machine Learning, Second Edition. Cham: Springer International Publishing.

Lai, T. L. & Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1), pp. 4–22.

Levin, M. (2014). Designing Multi-Device Experiences: An Ecosystem Approach to User Experiences Across Devices. Sebastopol: O'Reilly Media, Inc.

Levinthal, D. A. & March, J. G. (1993). The myopia of learning. *Strategic Management Journal*, 14(S2), pp. 95–112.

Li, L., Chu, W., Langford, J. & Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, pp. 661–670.

Lindgren, E. & Münch, J. (2016). Raising the odds of success: the current state of experimentation in product development. *Information and Software Technology*, 77(C), pp. 80–91.

Liu, B., Yu, T., Lane, I. & Mengshoel, O. J. (2018). Customized nonlinear bandits for online response selection in neural conversation models. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 5245–5252.

Luce, R. D. (1959). Individual Choice Behavior. New York: John Wiley & Sons, Inc.

Machmouchi, W. & Buscher, G. (2016). Principles for the design of online A/B metrics. *SIGIR 2016 - Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 589–590.

Manjunath, A., Bhat, M., Shumaiev, K., Biesdorf, A. & Matthes, F. (2018). Decision Making and Cognitive Biases in Designing Software Architectures. *Proceedings - 2018 IEEE 15th International Conference on Software Architecture Companion, ICSA-C 2018*, pp. 52–55.

McChesney, C., Covey, S. & Huling, J. (2012). The 4 disciplines of execution : achieving your wildly important goals. New York: Free Press.

Misra, K., Schwartz, E. M. & Abernethy, J. (2019). Dynamic online pricing with incomplete information using multiarmed bandit experiments. *Marketing Science*, 38(2), pp. 226–252.

Mitchell, T. (1997). Machine Learning. New York: McGraw-Hill.

Mueller, J. & Massaron, L. (2016). Machine learning for dummies. Hoboken: John Wiley & Sons, Inc.

Navot, Y. (no date). Beyond A/B testing: Multi-armed bandit experiments. Online. [Last accessed: 06.08.2020]. Available at: https://www.dynamicyield.com/blog/contextual-bandit-optimization/

Rissanen, O. & Münch, J. (2015). Continuous Experimentation in the B2B Domain: A Case Study. *Proceedings - 2nd International Workshop on Rapid Continuous Software Engineering, RCoSE 2015*, pp. 12–18.

Robbins, H. (1952). Some Aspects of the Sequential Design of Experiments. *Bulletin of the American Mathematical Society*, 58(5), pp. 527–535.

Ros, R. & Runeson, P. (2018). Continuous experimentation and A/B testing: A mapping study. *Proceedings - International Conference on Software Engineering*, pp. 35–41.

Rovio (2019). Capital Markets Day 2019, Games. Online. [Last accessed 06.08.2020]. Available at: http://assets-production.rovio.com/s3fs-public/rovio_cmd_games.pdf

Saunders, M., Lewis, P. & Thornhill, A. (2016). Research methods for business students. Seventh edition. Harlow: Pearson Education.

Schaal, S. (2020). The Four 'Pure' Learning Styles in Machine Learning. Towards Data Science. Online. [Last accessed 06.08.2020]. Available at: https://towardsdatascience.com/the-four-pure-learning-styles-in-machine-learning-a6a1006b9396

Schermann, G., Cito, J. & Leitner, P. (2018). Continuous Experimentation: Challenges, Implementation Techniques, and Current Research. *IEEE Software*, 35(2), pp. 26–31.

Schwartz, E. M., Bradlow, E. T. & Fader, P. S. (2017). Customer acquisition via display advertising using multi-armed bandit experiments. *Marketing Science*, 36(4), pp. 500–522.

Scott, S. L. (2010). A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6), pp. 639–658.

Scott, S. L. (2013). Multi-armed bandit experiments. Google Analytics Solutions. Online. [Last accessed 06.08.2020]. Available at: https://analytics.googleblog.com/2013/01/multi-armed-bandit-experiments.html

Scott, S. L. (2015). Multi-armed bandit experiments in the online service economy. *Applied Stochastic Models in Business and Industry*, 31(1), pp. 37–45.

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J. F. & Dennison, D. (2015). Hidden technical debt in machine learning systems. *NIPS'15: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, 2, pp. 2503–2511.

Slivkins, A. (2019). Introduction to multi-armed bandits. *Foundations and Trends in Machine Learning*, 12(1–2), pp. 1–286.

Snyder, H. (2019). Literature review as a research methodology: An overview and guidelines. *Journal of Business Research*, 104(March), pp. 333–339.

Stucchio, C. (2014). The Adversarial Bandit is not a Statistics Problem. Online. [Last accessed 06.08.2020]. Available at: https://www.chrisstucchio.com/blog/2014/adversarial_bandit_is_not_statistics_problem.html

Sutton, R. S. & Barto, A. G. (1998). Reinforcement Learning: An Introduction. Cambridge: MIT Press.

Szajnfarber, Z. & Gralla, E. (2017). Qualitative methods for engineering systems: Why we need them and how to use them. *Systems Engineering*, 20(6), pp. 497–511.

Tang, D., Agarwal, A., O'Brien, D. & Meyer, M. (2010). Overlapping experiment infrastructure: More, better, faster experimentation. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 17–26.

Tekin, C. & Turgay, E. (2018). Multi-objective Contextual Multi-armed Bandit with a Dominant Objective. *IEEE Transactions on Signal Processing*, 66(14). pp. 3799–3813.

Thompson, W. R. (1933). On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika*, 25(3–4), pp. 285–294.

Vermorel, J. & Mohri, M. (2005). Multi-armed bandit algorithms and empirical evaluation. *ECML'05: Proceedings of the 16th European conference on Machine Learning*, pp. 437–448.

Vowpal Wabbit.org (no date). Vowpal Wabbit Machine Learning Library Wiki. Contextual bandit algorithms. Online. [Last accessed: 06.08.2020]. Available at: https://github.com/VowpalWabbit/vowpal_wabbit/wiki/Contextual-Bandit-algorithms

Wasserman, A. I. (2016). Low ceremony processes for short lifecycle projects. In: Kuhrmann M., Münch J., Richardson I., Rausch A., Zhang H. (eds) Managing Software Process Evolution: Traditional, Agile and Beyond - How to Handle Process Change. Cham: Springer International Publishing, pp. 1–13.

Xie, H. & Aurisset, J. (2016). Improving the sensitivity of online controlled experiments: Case studies at Netflix. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 13-17 August*, pp. 645–654.

Xu, Y. & Chen, N. (2016). Evaluating mobile apps with A/B and quasi A/B tests. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 13-17-August*, pp. 313–322.

Yin, R. K. (1993). Applications of Case Study Research. Newbury Park: SAGE Publications.

Yin, R. K. (2014). Case study research: Design and methods (5th ed.). Thousand Oaks: SAGE Publications.

Zhou, L. (2015). A Survey on Contextual Multi-armed Bandits. arXiv:1508.03