Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

William Tallis Falch

Measuring the Effect of Recommender Systems in Online Video Learning Platforms

A Case Study with Utdannet.no

Master's thesis in Master of Science in Informatics
Supervisor: Özlem Özgöbek
Co-supervisor: Mateja Stojanovic
September 2022

NTNU
Norwegian University of
Science and Technology

William Tallis Falch

Measuring the Effect of Recommender Systems in Online Video Learning Platforms

A Case Study with Utdannet.no

Master's thesis in Master of Science in Informatics
Supervisor: Özlem Özgöbek
Co-supervisor: Mateja Stojanovic
September 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Declaration of own work

I declare that the work in this MSc dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

William Tallis Falch, September 1st 2022

Acknowledgement

I would like to thank my supervisors Özlem and Mateja for their continued guidance and excellent support throughout the duration of the thesis. I would not have finished it without their unrelenting commitment to me and my work. Next I would like to thank Utdannet; Espen and Lars Vidar, for their cooperation. By opening up Utdannet to me I was given an opportunity that is afforded to few. I would also like to thank my friends for their moral support during a difficult year.

Finally, I would like to thank my mom, Anne. She has never been more than a phone call away, and has helped me through difficult times. I would also like to thank my cats, JD and Leni, for being good cats.

Abstract

Recommender systems are ubiquitous in today's society. Their utility makes them see use in a number of domains, from search engines, to commerce, to education. The advent of cheap, reliable technology has made way for e-learning and robust systems to facilitate the presentation of quality learning materials. But evaluating nuanced questions about the effects of personalised recommendation in the e-learning space can be difficult. In cooperation with Utdannet, this thesis aims to investigate the effect of personalised recommendations on user engagement and time spent on the Utdannet platform.

To investigate this effect, two live experiments were performed using A/B testing. Separated into groups, two recommendation models were employed to measure the CTR and a modified version of the dwell time metrics.

The results of the experiments were mostly inconclusive. Sparse data, a too simplistic model, and a low adoption rate are hypothesised causes as to why the results are indecisive. While no conclusions can be drawn, there is a clear discrepancy in the number of recorded observations between the strategies employed. As such, future work calls for a model that is better at producing recommendations with sparse data, and longer experiments.

Sammendrag

Anbefalingssystemer er over alt i dagens samfunn. Deres nytteverdi gjør at de ser bruk i mange domener, fra søkemotorer, til handel, til utdanning. I dag finnes det mye billig, pålitelig teknologi, og dette har banet vei for e-læring og robuste systemer som presenterer gode læringsmaterialer. Men å evaluere nyanserte spørsmål om e-læring kan være vanskelig. I samarbeid med Utdannet utforsker denne avhandlingen personaliserte anbefalinger og deres effekt på brukerengasjement og tid brukt på Utdannet sin plattform.

For å undersøke denne effekten ble to A/B-tester utført. To forskjellige anbefalingsstrategier ble brukt for å måle klikkraten og dveleraten mellom de forskjellige strategiene.

Resultatene av eksperimentene er inkonklusive. Mangel på data, en for simplistisk modell, og en lav adopsjonsrate er hypotetiserte årsaker til hvorfor resultatet er som det er. Selv om ingen konklusjoner kan bli tatt, er det en signifikant forskjell i antall observasjoner mellom de forskjellige strategiene som ble brukt i hybridmodellen. På bakgrunn av dette anbefaler jeg mer forskning på mer tilpassede modeller, og en lengre eksperimentperiode.

Contents

Page

1 Introduction **11**

1.1 Motivation . . . 11

1.2 Problem Outline . . . 12

1.3 Research questions . . . 12

1.4 Thesis outline . . . 13

1.5 Utdannet . . . 13

2 Theoretical background **15**

2.1 Recommender systems . . . 15

2.1.1 Recommendation strategies . . . 15

2.1.2 Similarity metrics . . . 18

2.1.3 Neighbourhood functions . . . 18

2.1.4 Offline evaluation . . . 19

2.1.5 Common challenges . . . 19

2.1.6 Recommending videos . . . 20

2.2 E-learning . . . 20

2.2.1 Properties of recommender systems in e-learning . . . 21

2.3 Online evaluation and A/B testing . . . 22

2.3.1 A/B testing: The process . . . 22

2.3.2 Considerations . . . 26

3 Related work **28**

3.1 Recommender systems in education . . . 28

3.1.1 Utdannet . . . 28

3.2 Video recommendation . . . 29

3.3 A/B testing in RS and e-learning . . . 30

4 Method **31**

4.1 Tools 31

4.1.1 LensKit 31

4.1.2 A/B testing tools 31

4.2 System architecture 32

4.2.1 Utdannet 32

4.2.2 Designed system 32

4.3 Data 36

4.3.1 Data description 37

4.3.2 Data statistics 40

4.4 Preprocessing 43

4.4.1 Filtering 43

4.4.2 Formatting 44

4.5 Models 44

4.5.1 User-based collaborative filtering using explicit ratings . 44

4.5.2 User-based collaborative filtering using implicit ratings . 45

4.5.3 Popularity-based models 46

4.6 Evaluation metrics 47

5 Experiments **48**

5.1 Experiment environment 48

5.2 Experiment 1 48

5.2.1 Hypothesis 48

5.2.2 Variables 48

5.2.3 Logging policy 49

5.2.4 Variations 49

5.3 Experiment 2 54

5.3.1 Hypothesis 54

5.3.2 Variables 54

5.3.3 Logging policy 54

5.3.4 Variations 54

6 Results **57**

6.1 Experiment 1 57

6.1.1 General statistics 57

6.1.2 Metrics 58

6.2 Experiment 2 59
6.2.1 General statistics 59
6.2.2 Metrics 60

7 Discussion 62
7.1 Experiment 1 62
7.2 Experiment 2 64
7.3 Choice of models 66
7.3.1 Popularity-based model 66
7.3.2 Hybrid model 67
7.3.3 Single-strategy CF 69
7.4 Limitations 69
7.4.1 Technical challenges 69
7.4.2 Working around existing systems 70
7.5 Future work 70
7.5.1 Different model 70
7.5.2 Running several, longer experiments 71

8 Conclusion 72
8.1 Summary of contributions 72
8.2 Lessons learned 73

A Appendix 74

References 81

List of Tables

4.1 Description of table Scores 38
4.2 Description of table Reviews 39
4.3 Data statistics 40
4.4 Most popular courses 41
4.5 Ratings statistics 42

6.1 General statistics about the experiment 57
6.2 General statistics about the variants 58
6.3 Statistics supporting the CTR 58
6.4 Statistics supporting the dwell time 59
6.5 General statistics about the experiment 59
6.6 General statistics about the variants 60
6.7 Statistics supporting the CTR 60
6.8 Statistics supporting the dwell time 61

List of Figures

4.1 Overview of what onInit does 33

4.2 Overview of what onNewRating does 34

4.3 Overview of what onHalfDay does 35

4.4 Overview of what onRecommend does 36

4.5 The number of times a course has been started, in descending order 42

4.6 The number of ratings each course received, in descending order 43

4.7 Diagram showing the process of generating recommendations based on user-based collaborative filtering using explicit ratings 45

4.8 Diagram showing the process of generating recommendations based on user-based collaborative filtering using implicit ratings 46

4.9 Diagram showing the process of generating recommendations based on the popularity of courses with regard to their mean completion rate 46

4.10 Diagram showing the process of generating recommendations based on the popularity of courses with regard to how many participants they have 47

5.1 The independent variable shown to the control group. 50

5.2 The independent variable shown to the experimental group. It looks the same as the control variable, but the recommendations themselves are different. 51

5.3 The control variable and the independent variable 55

7.1 The different parameters of equation 2.3 for the first experiment. 62

7.2 The different parameters of equation 2.3 for the second experiment. 65

7.3 The site pre-experiment 67

7.4 The site during the experiment 67

Abbreviations

CB Content-Based

CF Collaborative Filtering

CTR Clickthrough Rate

DLR Digital Learning Resource

ICT Information and Communications Technology

IFS Information Filtering System

LO Learning Object

MDE Minimum Detectable Effect

MOOC Massive Open Online Course

OCE Online Controlled Experiment

RS Recommender/Recommendation System

SVD Singular Value Decomposition

1 Introduction

This chapter provides an introduction to the main aspects of the thesis, including my motivation, an outline of the problem, my chosen research questions, and an outline of the thesis. A section is also provided to the company I worked with to produce this thesis.

1.1 Motivation

In today's society, recommender systems (RSs) are ubiquitous. Their versatility and usefulness makes them see use in a number of domains, from commerce, to social networks, in search engines, and in education. Their value makes them highly sought after, and there is plenty of active research to identify new applications and optimise promising recommendation strategies [1].

With the advent of affordable technology and the wide spread of internet access, a new way of learning has emerged. Technology-assisted learning techniques go under many names, but common to all of them, Digital Learning, Technology Enhanced Learning, E-learning, amongst other variants, is their use of technology to replace, assist, or enhance learning for the subject [2]. In its most unorganised format these technologies can provide individual articles and videos, but with time complex systems evolving around the presentation of good, quality learning resources have emerged, and large repositories such as MERLOT [3] and OER Commons [4] are free to browse and develop around. The availability of such collections has encouraged actors to create systems that use these digital learning resources (DLRs) in a more organised way. These systems often have private collections of DLRs too to supplement their public counter parts.

E-learning also offers a unique opportunity for people who previously have not had access to education for various reasons. Since restrictions placed on time and place often are not present in these systems, people in underprivileged communities suddenly have a viable way to pursue education. The recent COVID-19 pandemic proved to many that digital learning works, and much research is being done to look at how e-learning can supplement traditional learning in the future [5].

A relatively recent invention, Massive Open Online Courses (MOOCs) [6], enables learners all over the world to enrol in online courses, often for free. Other providers cater to a more specific audience, offering specialised post-graduate education, training, or certification. In this domain as in others, RSs are starting to be employed to offer personalised suggestions. These RSs sort through ever-increasing collections of DLRs to offer quality learning resources. Depending on the type of education offered, the format of the course, and how users engage with the platform, creating an effective RS can be difficult. Keeping users engaged and ensuring that the content is consumed properly requires quality recommendations.

1.2 Problem Outline

The domain of online, non-formal, video-based, platform-based applications is mostly comprised of private actors. While the natural competition encourages innovation and research, it also encourages closed systems where evaluating the effects of employing personalised recommendations can be difficult. In the e-learning space, it is vital to keep users engaged to ensure a good understanding of the provided DLRs. While research of similar use-cases that utilises offline evaluation exists, limiting the research to historical data also limits the possibilities of understanding such an environment, as the results of offline evaluation do not necessarily transfer to live environments [7]. It makes answering more nuanced questions about user engagement unattainable, and makes it more difficult to design systems that respond well to live users.

1.3 Research questions

This thesis aims to challenge some of the assumptions that are made about the use of personalised recommendations in the e-learning space:

Research Question 1: Does the use of personalised recommendations affect user engagement and time spent on the platform in non-formal education?

Research Question 2: When using personalised recommendations, how does the use of implicit ratings compare to the use of explicit ratings with CF in affecting user engagement and time spent on the platform?

1.4 Thesis outline

Introduction This chapter provides an introduction to the thesis; its foundations, its aims, and other general information.

Theoretical background This chapter provides a theoretical background for many of the topics that are being discussed throughout the thesis; concepts that are used in other work and in this thesis.

Related work This chapter provides insight into work that is being done in similar domains; work that inspired this thesis and work that serves as a foundation for it.

Method This chapter discusses the methods used in pursuit of exploring the research questions. It also discusses dataset properties.

Experiments This chapter describes the experiments that have been performed to test the research questions.

Results This chapter presents the findings of the experiments.

Discussion This chapter discusses the results and how they relate to the research questions.

Conclusion This chapter concludes the thesis by summarising the work and its results.

1.5 Utdannet

The work on this thesis was done in cooperation with Utdannet [8]. Utdannet is a company based in Oslo, Norway, that offers a subscription based service akin to Netflix, but for online video courses. They service thousands of customers monthly through their several hundred courses, with more in development. They sell their services to both enterprise and stand-alone customers, and their catalogue covers a wide range of topics - everything from business, to visual media, to software development. Their mission statement reads: "Our goal is to make the market for courses and competencies available to all". In the realm of e-learning, they are referred to as providing a service in the online, non-formal, video-platform space.

2 Theoretical background

Since the invention of modern computers there has been a need for ways to filter digitally stored information. As time passed, and hobbyists and scientists needed to make sense of ever-growing collections of data, Information Filtering Systems (IFSs) were proposed. Early systems were focused on two issues; the overabundance of information and the propensity for noisy results. These days IFS has become an umbrella term for a wide variety of technologies, all focused on aspects of the information retrieval process.

This chapter covers topics related to this thesis. It will give an introduction into a branch of IFSs called recommender systems (RSs); what they are and how they work. Furthermore it will provide an introduction into the domain of e-learning, and how e-learning has taken advantage of RSs. Finally, it will discuss a branch of online evaluation called A/B-testing and how it can be utilised to optimise system performance and user satisfaction.

2.1 Recommender systems

Recommender systems are a sub-class of Information Filtering Systems, and are tasked with analysing data to produce preferences given different criteria. Common use-cases involve recommending merchandise in online stores, media on platforms such as YouTube [9] or[10], or textual data from search engines such as Google [11]. These systems vary widely in their complexity. While early systems such as Grundy [12] were simple in nature, we today see a broad range of recommender systems utilising state of the art methods in highly specialised scenarios.

2.1.1 Recommendation strategies

When deploying a recommender system there are many strategies one can choose between to generate recommendations. These strategies all have their strengths and weaknesses, and the choice of strategy greatly affects the effectiveness of the recommender. Selecting a strategy can be the result of many factors; the domain of the system, the available input data and the complexity of the recommendation problem.

Collaborative filtering

Collaborative filtering (CF) is a well-known and much user recommendation strategy. As its name suggests the method compares patterns amongst entries in its data bank and yields results based on similarity, the intuition being that similar historical patterns should produce similar future results and that those results would reflect shared interests. CF is used in many domains, for various reasons. It requires no explicit feedback, and can be implemented with little ease, which makes it good for a variety of applications.

The study of CF has been extensive, and over the years a multitude of techniques have been developed within the domain. We usually separate the techniques into two groups, based on their strategy for recommending items [13].

Memory-based filtering Memory-based CF makes use of user ratings in its predictions, and can be split into two types; **user-based** and **item-based**. Usually when people discuss memory-based CF, they are referencing memory-based CF using explicit ratings, where "explicit" refers to the type of data used to recommend. In other words, the data, usually ratings, have been made by the users themselves, and we use those to predict new ratings. The alternative, memory-based CF using implicit ratings, uses implicit data to predict new ratings. Implicit data in this context refers to data that the system has collected about the user, but does not infer preference. This could be the user's purchase history, or their current shopping cart [14].

User-based CF starts with a user, and tries to find users that have rated content similarly to recommend new items. One drawback of this approach is that for larger datasets we expect the number of users to be very large. Not only will this make comparison an expensive operation, we can also find too much diversity and it can be difficult to recommend effectively. In addition we can run into cold-start issues as we do not have anything to compare with [14] [13].

Item-based CF puts items in the centre. Starting from an item, we look at similar items based on the preferences of other users. An advantage of item-based CF is that item matrices are more sparse, making comparison a cheaper operation. We also will negate most issues stemming from cold-start as a single rating can be enough to produce other recommendations. The drawback lies in the diversity of recommendations. Where user-based CF can have too much diversity, item-based can suffer from a lack of diversity [14] [13].

Content-based filtering

Content-based filtering (CB), as the name implies, is a strategy used to recommend items based on the content of the system. In particular, the strategies are devised based on descriptions of the items, and profiles made of the users' preferences. While explicit data can be used, CB excels in domains devoid of explicit data. Like CF using implicit ratings, CB makes use of historical data to create user profiles, rather looking at the relationship between user data points [14].

Deep learning

In the last few years we have seen an exponential growth in the development of Deep Learning (DL), and its applications grow by the day. Modern DL uses complex layering mechanisms stacked in hierarchies. In DL there is not "one" network topology or "one" method, but rather different topologies that perform different predictive analyses [14].

Hybrid systems

In this section we have only discussed some of the existing filtering methods used in RS. Sometimes, the use of a single method proves inadequate in the development of a RS. Often this comes about when we deal with complex data for which no one method is perfectly suited. Luckily we can use several strategies in what is called a hybrid system. The application of one depends entirely on the problem and the resources available to the designers. In some systems, these co-opted strategies may be used simultaneously, e.g. one can be used as a fallback strategy to solve the same problem, while in other systems they may be used sequentially to solve different parts of the same problem.

2.1.2 Similarity metrics

Some recommendation strategies require us to establish a relationship between two or more entities, and to measure that relationship quantitatively. This is done in the hopes that the relationship can yield information which will lead us to produce better recommendations. For that we use a branch of functions called similarity measures, or similarity metrics. The choice of similarity metric depends on several factors, including the domain of the recommendation task, the complexity of the input data, and experimental testing.

Cosine similarity

When dealing with vectors consisting of real values, the cosine similarity measure is commonly used. The measure is simple, but effective, as it measures the cosine of the angle between two vectors - their dot product, divided by the product of the vectors' lengths.

$$cosine\ similarity = S_C(A,B) := cos\theta = \frac{A \cdot B}{\|A\|\,\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{2.1}$$

Calculating the cosine similarity of two vectors is an efficient operation which allows the measure to be utilised in a wide array of applications [15].

Jaccard similarity

Another similarity measure commonly used in computer science and RS is the Jaccard similarity index. This measure takes a different approach and looks at the ratio of the intersection of the sets to their union [16].

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{2.2}$$

2.1.3 Neighbourhood functions

K-nearest neighbours

K-nearest neighbours (k-NN) is a non-parametric supervised learning algorithm. Its use in RS is widespread, where it can be used both in regression and classification. As a tool in regression, the output of the algorithm is a variant of the average of the k nearest neighbours [17].

2.1.4 Offline evaluation

Evaluating one's RS is an important task. Offline evaluation refers to evaluation using already observed data. While such evaluation reduces the metrics one is able to test for, it is still a vital part of developing a RS. The general approach to offline evaluation consists of splitting the dataset to be used in the RS into two; one for training the RS model and one for testing it. After the model has been trained, an evaluation metric is chosen and the test set is run through the trained model for observation and to calculate the value of the metric. Many metrics exist, and choosing the correct one for the recommender at hand can be arduous work [14] [18].

2.1.5 Common challenges

Cold start

Cold start concerns the problem of recommending items in systems that have sparse data, and manifests itself in several ways. In young systems, cold start can refer to a lack of items or users to use for comparison. In mature systems, the problem of cold start can occur if a user presents themselves as unusual compared to other users, making it difficult to yield satisfactory inferences, or if new items about which little is known are added to the catalogue. Much work has been done to study solutions to the cold start problem [14] [19].

Long tail

The long tail problem describes the difference in the frequency of recommendations between items in the recommendation space, where some items are recommended more frequently than others. At its most extreme, some items are recommended frequently, while others only see the light of day in edge cases or in fallback scenarios. There are many reasons for why long tail occurs. In some scenarios, the domain of the system excludes items that are less relevant to that domain. In other cases, a similarity between items causes some to be more popular while others are neglected. A lack of overlap with other items can also cause a recommender system to overlook certain items [14].

2.1.6 Recommending videos

As technology evolves and access to high-speed internet has become normalised, the introduction of videos as DLRs is a natural step. Videos can enhance learning in many ways, the foremost of which being in its ability to offer access to visual learning aids, as well as certified tutors and their learning materials.

But recommending items from video system catalogues comes with a unique challenge, namely videos' inherent lack of textual data to be analysed for use in recommending. While it is true that many video libraries are transcribed or come with quality metadata describing its content, a lot of these systems are unable to rely on the content of the videos themselves, and creating recommender systems around video catalogues can therefore be a laborious process.

An alternative to dealing with the videos themselves is to focus on the users and their behaviour within the system. As with other recommender systems, we can look at explicit and implicit information given to us by the users, such as ratings, completion rates, and metadata such as the category of the video, to name some [20] [21] [22].

2.2 E-learning

E-learning is becoming more and more prevalent in today's ubiquitously technological world. The adaptation of technology into the learning space has transformed it entirely in many ways, including when and where we learn. For many, access to traditional education has been difficult due to the restrictions placed on its time and location, but with the advent of cheap technology e-learning has become a viable alternative to traditional methods. It has also proven to be a viable alternative in times of crisis, such as during the COVID-19 pandemic[5].

As with traditional learning, e-learning comes with its own weaknesses and challenges. Internet access, the absence of a tutor, the personal responsibility put on time management, and the ensured quality of the learning resources, among other problems, are problems that in the e-learning space can become magnified. As such it is important to employ technology that reduces the impact of these issues.

In recent years, large repositories of DLRs have been collected and made available for use. MERLOT [3] and OER Commons [4] are among the more famous repositories, but there exists a number of public and private archives. Making sense of them however is no easy task. Depending on the collection, hundreds of thousands of learning resources are available, and often there are multiple resources for the same query. With the help of RS, companies and researchers are investigating new ways to filter these collections and provide quality learning resources for a number of different learning scenarios. The use of systems that recommend personalised educational resources can potentially improve the learning process and learning outcome of a student [23] [24].

2.2.1 Properties of recommender systems in e-learning

Formal and non-formal systems of learning

In e-learning, and in education in general, we distinguish between formal and non-formal systems. Formal systems are systems used in formal educational settings, such as in schools and universities. On the other hand, non-formal systems often refer to systems deployed by corporations for training of its work force, or to provide as a service, generally paid, to the general population. While these systems share many characteristics, their origins and objectives are different, and as such come with their own unique challenges.

Online and offline systems

In a similar vein, there is a difference between online and offline learning. It refers to the method of teaching, where offline learning requires physical attendance, while online learning usually allows for more leniency in when and where the learning takes place.

2.3 Online evaluation and A/B testing

Making changes to an application can be a gruelling process. Without the proper feedback channels, changes that can have adverse or negligible effects are difficult to detect. Most evaluation performed is so-called offline evaluation. It refers to evaluation that is being done based on historical data. While a lot can be gained from such evaluation, some studies suggest a reduced correlation to a live environment [25].

The alternative, online evaluation, suggests doing live experiments on system-specific data. As a live scenario mirrors the user experience, live evaluation can often give us insight that is unattainable with its offline counterpart. If the user distribution is the same as recorded during an evaluation and with some statistical analysis performed, we can ensure that any measured effect should remain after the evaluation. Live evaluation can give us insight offline evaluation methods do not necessarily support.

A popular method to observe and measure the effects of inquired changes is through the use of A/B testing. A/B testing consists of communicating two competing ideas, and measures the difference in effect between the two. It is considered the gold standard in industry settings [7].

2.3.1 A/B testing: The process

Formulating a hypothesis

Before testing can begin, a hypothesis needs to be postulated. While one in theory can test anything of any order, smaller more focused tests usually perform better. This is because a smaller test limits the differences between the presented variations, and as such it is easier to establish the eventual cause of a measured effect.

Variables

In online evaluation we speak of several types of variables:

Independent variable is the element we will test for in the experiment. This variable is controlled by the people performing the experiment, and is closely related to the hypothesis.

Dependent variable is the variable that captures an effect. The variable depends on the value of the independent variable, and its value can not be predetermined by the people performing the experiment. The dependant variable goes by many names, most famous of which is the performance metric. The performance metric determines the result of the experiment, and it is paramount that it is selected before testing begins. This is vital as a proposed change can alter many metrics, some for the better and others for the worse, and as such it can be difficult to determine the success of the experiment. As previously mentioned you can test for anything, but if the difference between the two versions is big it can make it difficult to understand why a measured effect is observed.

Control variable is closely related to the independent variable, and describes the unaltered, or original, version of the component you want to test.

Selecting the dependant variable is not a straightforward task. It is an active field of research, and choosing the right one is vitally important depending on what you want to measure. Its selection also depends on the available data.

Common metrics

Clickthrough rate: One of the most common dependant variables is the clickthrough rate, or the CTR. The clickthrough rate measures the rate at which a document is clicked on relative to its views. While the CTR is the simplest metric to measure, it is also quite powerful as a click is a definite, purposeful action. In research it is often used as a baseline against other, more advanced metrics. As with other metrics, the CTR introduces bias, especially in situations where document position is relevant [26], or even due to document presentation [27].

Dwell time: A more nuanced metric is the dwell time. While the CTR measures interest, it does not capture information beyond that. Capturing dwelling time on a clicked page can tell us something about a pursued interest for the recommended item, often referred to as a *satisfied click*. There are several ways of measuring dwelling time. It can be as easy as measuring the time spent on the clicked item [28], or it can be measured in more complex ways, e.g. by measuring the pattern of interactions with the recommended item.

Others: Designing new metrics is an active field of research. Advanced metrics can be designed depending on one's needs and available resources. Sophisticated metrics are implemented in a variety of systems, some using statistical approaches, others using AI models. Of course one can also use a combination of metrics, either independently, or combined to produce a more nuanced goal to pursue [7].

Performing necessary calculations

To ensure that the experiment is statistically significant it has to have roots in statistical analysis. This requires some knowledge about the data, some assumptions about the data based on similarly performed experiments, as well as some predictions about the outcome of the experiment. The big question we need to answer is about the sample size for each variant. Performing this calculation also allows us to estimate the length of the experiment.

$$n = \frac{(Z_{\alpha/2}\sqrt{2p_1(1-p_1)} + Z_\beta\sqrt{p_1(1-p_1)+p_2(1-p_2)})^2}{|p_2 - p_1|^2} \tag{2.3}$$

p_1 The baseline conversion rate, the conversion rate of the control variable before the start of the experiment.

p_2 The conversion rate lifted by the absolute Minimum Detectable Effect (MDE). The MDE is the smallest effect that can be detected, and as such it is a vital parameter.

α The significance level. This parameter signifies the strength of the evidence that must be present in the sample before you reject the null hypothesis, i.e. that the measured effect is statistically significant.

β The statistical power, 1 - β. The power of the test is the probability that an effect can be detected in the sample of the test if it exists in the overall population.

$Z_{\alpha/2}$ The z-score from the z-table corresponding to $\alpha/2$

Z_β The z-score from the z-table corresponding to β

If we have access to historical data we can predetermine the expected length of the experiment, by dividing the sample size by the average number of samples per day. These days we can find several online calculators from reputable sources that will do the mathematics for you [29], but understanding the different parameters are vital to ensuring a correct calculation.

Designing and performing the experiment

After the variables have been determined, the experiment needs to be designed. A controlled environment is important to ensure the validity of the experiment as there can be many causes for why a measured effect is observed. Maybe the value of the independent variable in fact does cause the effect, or maybe the difference is caused by outside factors - something as simple as changes in weather patterns caused by a new season can interfere with the composition of the audience. Here one also has to take into account the available resources to determine whether an experiment at all can be run. Some companies, like Netflix, have the resources to run continuous experiments, while other companies, e.g. startups, might not have the money or the time to prioritise elaborate experiments. Does one design a system that is quick to develop but only runs a specific scenario, or does one spend some extra time designing a more versatile solution that can be improved and re-used.

The system also needs to be able to capture all the necessary information to determine the value of the dependent variable. While some studies note that the use of client-side software (such as an add-on) is most suitable for academic purposes [30], it necessarily depends on the situation at hand. Server side logging, while requiring intimate access and knowledge of the underlying system, provides full access to necessary data.

When a test has been designed, its necessary engineering has been completed, and a trial run of the system has been executed, the experiment itself can start. Based on equation 2.3, we can calculate the approximate run time of the experiment. The influx of data needs to be monitored. While the minimum number of observations required according to equation 2.3 needs to be recorded, generally a surplus of data is welcome. If the number of observations is unattainable, a new experiment will have to be designed, or, we can relax some of the parameters.

Post-experiment analysis

After the end of the experiment, an analysis of the collected data needs to be performed. The data collected should be rich enough to allow you to calculate the value of the dependant variable(s). After assessing the results of the analysis, you can conclude the experiment or perform more.

2.3.2 Considerations

Run time of the experiment

While the formula for calculating the number of observations required per variant, and as such the run time of the experiment, is presented in equation 2.3, studies show that running longer experiments bears more statistical significance [31]. This is because longer experiments marginalise fluctuations in user behaviour. Depending on the audience of the system, these behaviours can greatly affect the outcome of the experiment. As an example, adults and parents generally log on their computers later in the day and more prominently on the weekends, while the summer and winter seasons see large shifts in user activity depending on weather conditions.

Negative effects

When designing an experiment there are certain negative effects one needs to be aware of:

The carryover effect concerns a noticeable effect in user behaviour. When a new change is introduced, some users might avoid to interact with the changed feature, either because they do not understand it, or because they do not feel the need. An online experiment might suffer from this if it is introducing features that are new to the user, or if they are confusing [31].

The network effect concerns the inter-connectivity of test subjects. If users within a system know each other, their behaviour within that system might not yield statistically significant data. Extra work is required to ensure that the prevalence of such a scenario is low [32].

Adoption rate concerns the introduction of a new feature. Depending on the size of the feature, its utility to the user, how disrupting it is to existing technology, and how familiar users are with it in theory are all variables that will affect how fast people will adopt it [33].

Ethical considerations

Unless explicitly told so, users are not aware of their participation in online evaluation. Depending on what is recorded, users may give up personal information without knowing that it is being recorded, analysed, and used for further development. This is a problem in many domains, but online evaluations, where one often records an abundance of personal data, can be prone to abuse or negligence, and as such it is vital to respect the safety and security of the recorded data.

3 Related work

This chapter presents an overview of related work upon which this thesis draws inspiration. The case study of Utdannet is unique, but an effort has been made to find work that resembles different parts of Utdannet's environment, and their strengths and challenges are discussed briefly.

3.1 Recommender systems in education

As we have seen, the prevalence of ICTs has enabled us to access educational resources easier than ever before. With an abundance of information available, the need for filtering and quality assurance mechanisms is growing, and as such we are starting to see complex systems built around these information catalogues in an attempt to offer good, relevant information on an individual basis. In their survey of educational recommenders, Urdaneta-Ponte et al. [34] list the most common approaches toward educational recommenders. Traditional methods such as collaborative filtering and knowledge-based systems are still in fashion due to their low complexity, but they also note the onset of new types of recommenders. Hybrid recommenders, systems composed of more than one strategy, provide more nuance and often perform better than any individual strategy would. Recent advances in AI are also utilised in the learning space, where U A et al. [35] note a number of new techniques used, including Bayesian techniques, artificial neural networks, ML techniques, genetic algorithms, and fuzzy set techniques. These developments look to either replace traditional methods, or to enhance them through the use of hybrid systems. The use of AI in these hybrid systems already seems to garner support, especially in systems that contain large quantities of data [36]

3.1.1 Utdannet

Utdannet's case is unique. It is a paid service that offers courses in the online, non-formal, platform-based space. As such, little research directly coincides with its use-case, though some MOOC systems come close.

In their 2019 survey of recommender systems in MOOC, Khalid et al. [24] observe that a lot of systems struggle with implementing good recommenders. More than half the systems surveyed that have recommenders use CB as a standalone strategy or in a hybrid model. While CB provides some degree of personalisation, it is usually used because of a lack of access to explicit user data such as ratings. They conclude that not enough work is done in live environments, and that a lot of previous work has been done on older, outdated datasets.

Onah et al. implement a RS framework for use with MOOCs [37]. In their paper, they note the proliferation of systems using CF and CB, or a hybrid of both. They also highlight the preference in research of using CF over CB because of the efficacy of CF in scenarios where appropriate data is available.

3.2 Video recommendation

As mentioned in the previous chapter, recommending in the video space is most often done without the use of the actual videos. In essence it becomes a prediction task like any other; recommending based on metadata and user behaviour. Depending on the video system one can also look at creator interactions. In their 2018 paper, Gupta et al. explore the MovieLens dataset [38]. They found that an item-based CF approach can be used to effectively recommend videos, and that this approach trumps the use of CB methods. As they are using historical data and offline evaluation methods, it is difficult to say how well these results would translate to live environments.

Others are using similar approaches. Molina et al. [39] show similar results working on recently live data from Netflix. They discuss several strategies, from popularity-based measures, to different implementations of CF, to the use of SVDs. They conclude that CF is the best approach. They also note that a strictly popularity-based measure is a good approach, though the recommendations would be the same for all and as such impersonal. Once again their evaluation is done offline.

3.3 A/B testing in RS and e-learning

A/B testing has long been used in the online domain to evaluate recommender systems, and is an active field of research [40].

A/B testing is used for many purposes; in one-off tests to improve metrics such as conversion rates [41], or in big companies like Netflix, where continuous A/B testing is used to improve user engagement and retention rates [42]. While the use of A/B testing is widespread, researching insight about more nuanced questions can be difficult to achieve. Online evaluation requires access to live systems and live users, which a lot of private companies are unwilling to provide. As such, several researchers are forced to limit their research to offline evaluation while commenting on the shortfalls [13].

In e-learning, researchers are also opening their eyes to the benefits such testing can provide as a tool to evaluate different metrics. While some researchers discuss their intention to implement A/B testing in future systems to perform online evaluation, [43] [44] others are already experimenting with it. In their 2016 paper, Renz et al. devised an A/B testing framework that could run on modern micro-service architectures [45]. It was implemented into two MOOC environments, and while some of their experiments proved inconclusive, others showed the advantage of using such testing to pursue new features.

4 Method

This chapter introduces the Utdannet dataset and the system including models that will be used in the later experiments.

4.1 Tools

Several tools and libraries were used during the research and development parts of this thesis.

4.1.1 LensKit

LensKit[46] is a set of Python tools for experimenting with and studying recommender systems. It provides support for training, running, and evaluating recommender algorithms in a flexible fashion suitable for research and education [47]. For this thesis, several of the model implementations were based on their counterpart in LensKit to ensure that no mistakes were made designing and engineering the system.

4.1.2 A/B testing tools

Several online A/B testing tools were consulted to make sure the results of the experiment were statistically significant. Most significantly, an A/B testing sample size calculator was used to perform preliminary sample size calculations [29] and to assist in verifying the results after the conclusion of the experiments. The results were compared against the results of equation 2.3 to ensure the validity of the results.

4.2 System architecture

4.2.1 Utdannet

The proposed system was enabled by, and laid on top of Utdannet's live service. The web application runs on Amazon's services; more specifically an S3 bucket [48]. Utdannet also utilises a number of other amazon services to power their application, including AWS IOT [49], Amazon DynamoDB [50], and AWS lambda [51].

4.2.2 Designed system

The designed system is outlined in the various figures below. As the system runs on a live service, parts of the system needs to run online and respond to live events, while other parts can run in the background. Data is collected directly from Utdannet's back-end through an internal API. It is then preprocessed, filtering out incompliant data points, mostly legacy data, before it is formatted and ready for use in the prediction task. A number of sub-routines were designed and implemented to run the system and ensure that the system updated when new data was presented to it. Bellow follows an account of the different sub-routines and their responsibilities within the system.

onInit This sub-routine was run as the experiment started. It was tasked with setting up all necessary data structures for logging purposes, generating the initial recommendations, and generating an initial filter that would be matched against the output from the recommendation strategies. The structure of the sub-routine can be seen in figure 4.1.

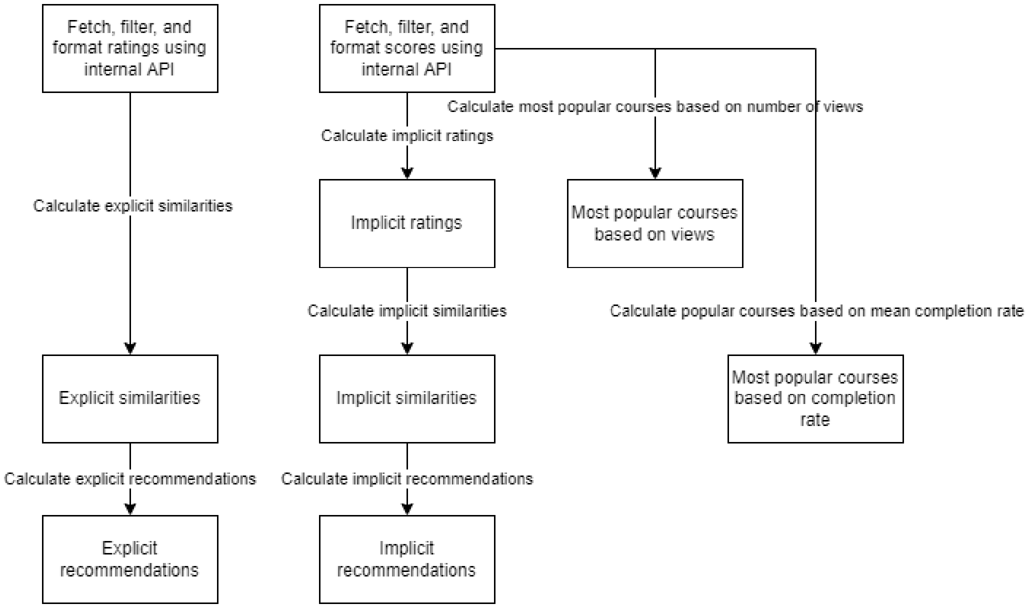

Fetch, filter, and format ratings using internal API
Fetch, filter, and format scores using internal API
Calculate most popular courses based on number of views
Calculate implicit ratings
Calculate explicit similarities
Implicit ratings
Most popular courses based on views
Calculate implicit similarities
Calculate popular courses based on mean completion rate
Explicit similarities
Implicit similarities
Most popular courses based on completion rate
Calculate explicit recommendations
Calculate implicit recommendations
Explicit recommendations
Implicit recommendations

Figure 4.1: Overview of what onInit does

onNewRating This sub-routine was run every time a new rating was registered in the system. Its purpose was to update the recommendations that were based on explicit ratings. It also updated the filter. The structure of the sub-routine can be seen in figure 4.2.

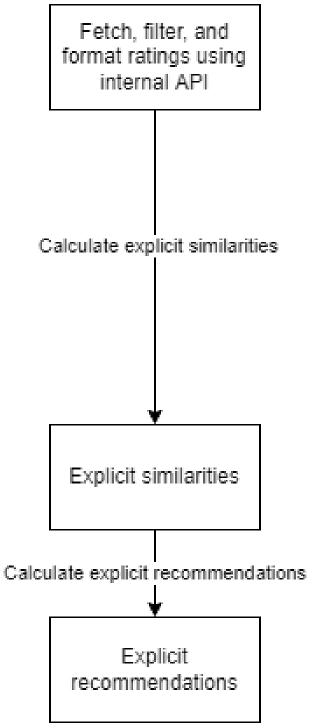

Fetch, filter, and format ratings using internal API
Calculate explicit similarities
Explicit similarities
Calculate explicit recommendations
Explicit recommendations

Figure 4.2: Overview of what onNewRating does

onHalfDay This sub-routine was run every 12 hours. Its purpose was to update the recommendations that were dependant on the scores object. This included recommendations that used implicit ratings, and the popularity-based strategies. It also updated the filter. More about this sub-routine can be found in in chapter 7. The structure of the sub-routine can be seen in figure 4.3.

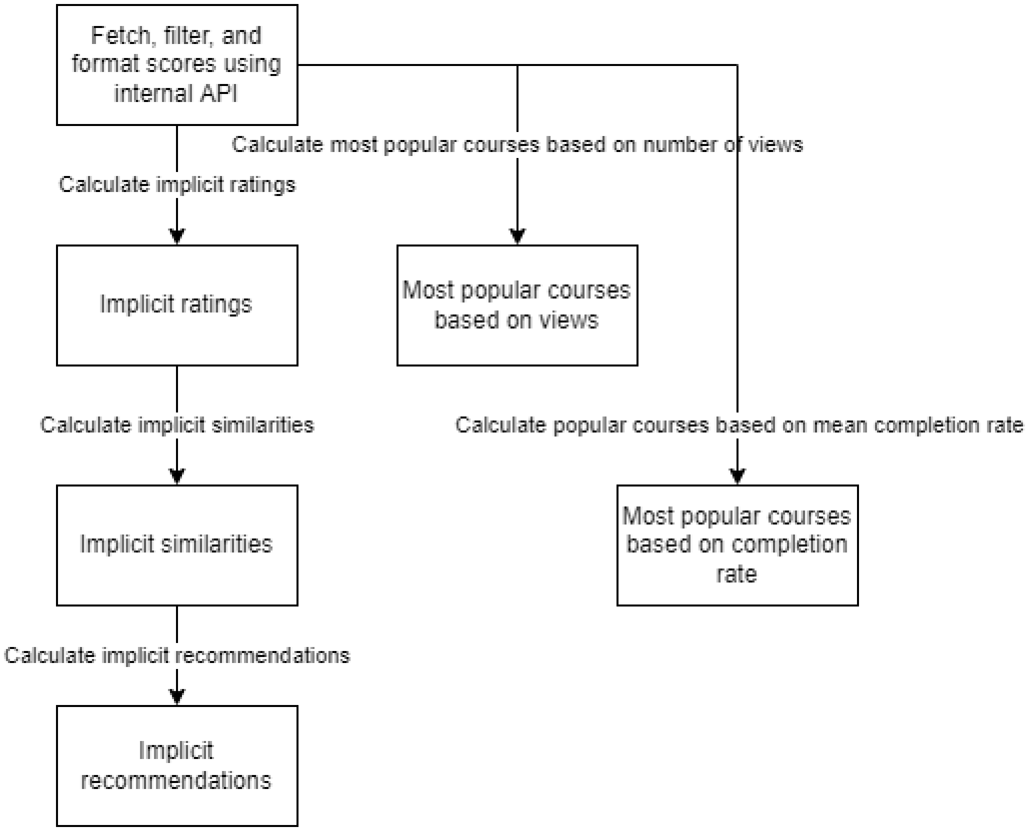

Fetch, filter, and format scores using internal API
Calculate most popular courses based on number of views
Calculate implicit ratings
Implicit ratings
Most popular courses based on views
Calculate implicit similarities
Calculate popular courses based on mean completion rate
Implicit similarities
Most popular courses based on completion rate
Calculate implicit recommendations
Implicit recommendations

Figure 4.3: Overview of what onHalfDay does

onRecommend This sub-routine was run every time a request was made to show the view that contained the independent variable. The sub-routine had two main tasks. First, this task was responsible for assigning a variant when a new user was presented. As no information about the user was available, the assignment was random. Second, a course list was generated according to the user's variant. For each variant, the filter was also queried to make sure courses that the user already had started were not presented. The structure of the sub-routine can be seen in figure 4.4.

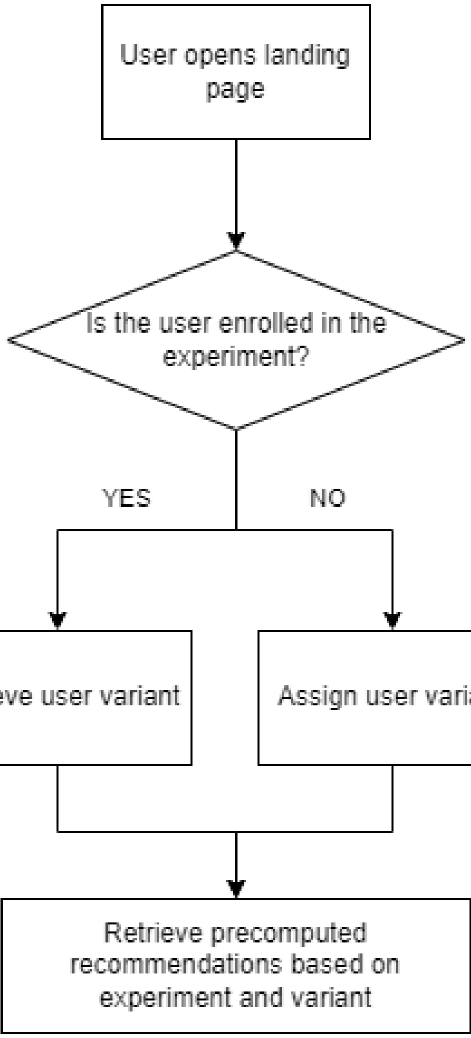

User opens landing page
Is the user enrolled in the experiment?
YES
NO
Retrieve user variant
Assign user variant
Retrieve precomputed recommendations based on experiment and variant

Figure 4.4: Overview of what onRecommend does

4.3 Data

Part of the cooperation with Utdannet included access to all of their data - both historical and real-time, through an internal API.

4.3.1 Data description

Below you can find a description of the data tables that were pertinent to the thesis.

Scores

Scores is tasked with keeping track of user course progression. Each scores object tracks the progress of a user's progress across one course. As such there can be several scores objects per user, or there can be none, according to the activity of the user. The structure of a scores object can be seen in table 4.1.

Table 4.1: Description of table Scores

Attribute	**Type**	**Description**
bookmarked	Boolean	Tracks if course is bookmarked by user
_id	String	MongoDB ID of object
username	String	Username of associated user
parent	String	Course name
total_sub_score	Number	Total score accumulated
status	String	Status wrt. course (completed/active)
number_of_items	Number	Number of lessons in course
watched_items	Number	Number of lessons completed
is_free	Boolean	Decides if course can be watched for free
chapters	Array	Metadata about completed chapters
chapters/chapterNumber	Number	Chapter number within course
chapters/lessons	Array	Metadata about completed lessons
chapters/lessons/lessonNumber	Number	Lesson number within chapter
chapters/lessons/finished	Number	Epoch when user finished lesson
chapters/lessons/score	Number	Score received for lesson
chapters/lessons/processed	Boolean	Score received for lesson
chapters/lessons/videoLength	Number	Length of lesson in seconds
createdAt	String	Datetime when object was created
updatedAt	String	Datetime when object was last updated
course	String	ID of associated course
progress	String	ID of associated progress object
id	String	ID of object

* Entries written in bold were used during post-experiment analysis

Reviews

Reviews is a table tasked with keeping track of user reviews. These reviews are displayed on the web page, and consist of a rating from 1 to 5 and a paragraph justifying the rating. As with scores there can be multiple reviews objects per user, or there can be none, depending on whether the user reviewed a course they completed. The structure of a reviews object can be seen in table 4.2.

Table 4.2: Description of table Reviews

Attribute	Type	Description
_id	String	MongoDB ID of object
title	String	Title of review
text	Integer	Text of review
rating	Number	Rating, out of 5, in increments of 0.5
createdAt	String	Datetime when object was created
updatedAt	String	Datetime when object was last updated
course	String	ID of course related to review
metauser	String	ID of associated user object
id	String	ID of object
username	String	Username of associated user

* Entries written in bold were used during post-experiment analysis

4.3.2 Data statistics

Table 4.3 shows some general statistics about the data. These include general statistics about the data, as well as specific statistics about the portion of the data in use. As these statistics change constantly, the ones recorded below were measured at the start of the first experiment.

Table 4.3: Data statistics

Statistic	**Value**
Number of unique users	4153
Number of courses available	211
Number of lessons available	7545
Number of content categories	37
Average number of lessons per course	35.76
Number of courses completed 100% / 80%	3226 / 5753
Average completion rate	54.6%
Median completion rate	47.5%
Median completion rate w/users grouped and average of users' entries	46.6%
Median completion rate w/users grouped and median of users' entries	35.6%
Mean number of courses completed 100% / 80% per user	0.78 / 1.39
Mean number of courses started per user	5.17
Median number of courses started per user	2
Number of lessons completed	312264
Average number of lessons completed per user	75.19

Table 4.4: Most popular courses

Rank	Name	Number of completions (100% / 80 %)
1	Kurs i Excel 2016: grunnleggende	141 / 204
2	Gratiskurs i digital markedsføring	102 / 163
3	Regnskapskurs: grunnleggende	100 / 142
4	Kurs i søkemotoroptimalisering: komplett	91 / 152
5	Kurs i Teams: komplett	82 / 131
6	Kurs i møteledelse: bedre møter	75 / 97
7	Kurs i Instagram: for bedrifter	73 / 119
8	Kurs i Excel 2016: viderekommen	70 / 92
9	Kurs i presentasjonsteknikk	67 / 82
10	Kurs i innholdsmarkedsføring: komplett	66 / 118
11	Kurs i kommunikasjon: unngå misforståelser	63 / 90
12	Gratiskurs i HTML: introduksjon	62 / 94
13	Gratiskurs i Google Analytics: introduksjon (2016)	56 / 97
14	Gratiskurs i Facebook: markedsføring	55 / 92
15	Kurs i Word 2019: grunnleggende	51 / 66

Table 4.5: Ratings statistics

Statistic	**Value**
Number of ratings	3179
Number of rated courses	197
Mean number of ratings per user	0.77
Median number of ratings per user	1
Most reviews left by one person	53
Average rating	4.65
Median rating	5.0
Average number of ratings per course, including / excluding non-rated courses	14.69 / 15.74
Median number of ratings per course, including / excluding non-rated courses	7 / 8
Number of courses with at least 2 / 5 / 10 ratings	177 / 132 / 75
Number of users with at least 2 / 5 / 10 produced ratings	514 / 110 / 16

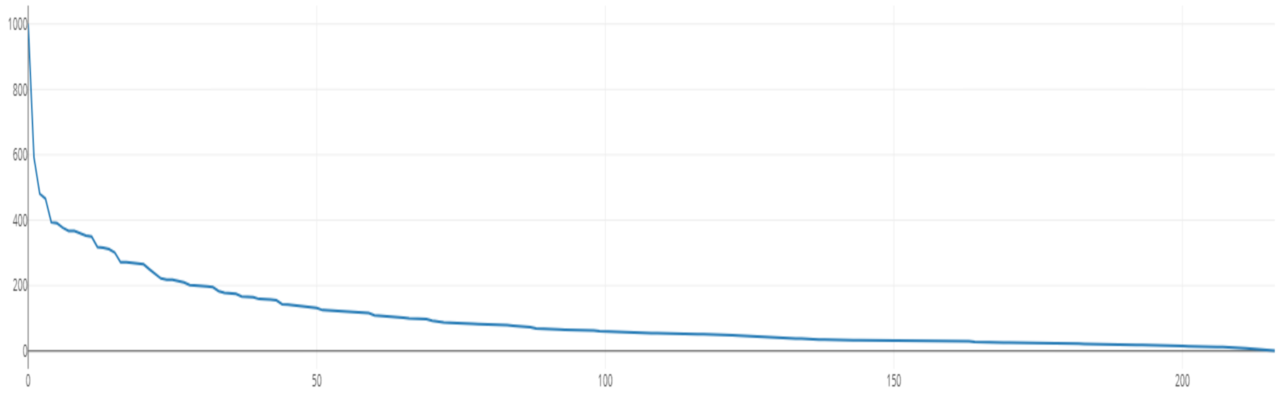


Figure 4.5: The number of times a course has been started, in descending order

In Figure 4.5 the frequency of a course in the scores objects list is shown, or more succinctly, how many times a course has been started. The shape of the graph indicates that using implicit ratings based on the scores objects might incur problems with the long tail problem as discussed in chapter 2.

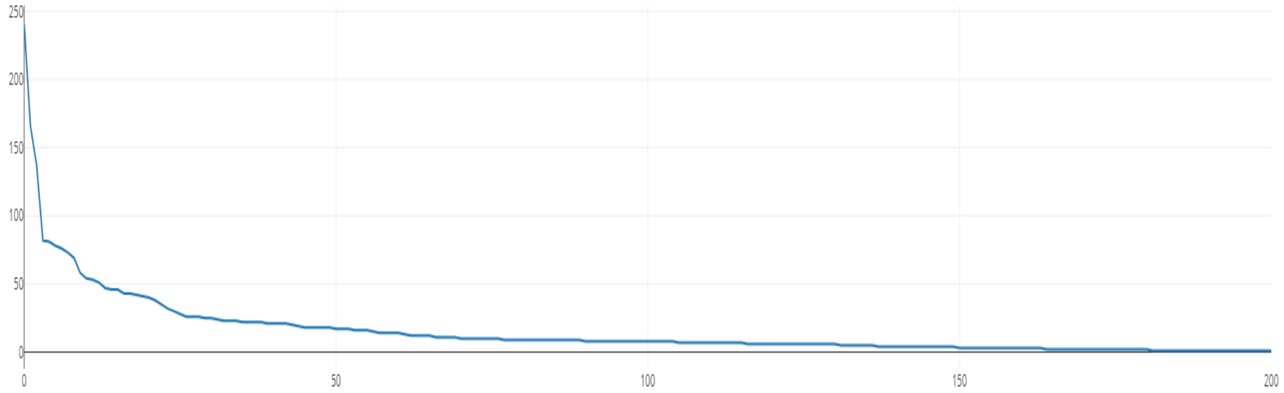


Figure 4.6: The number of ratings each course received, in descending order

Figure 4.6 shows the frequency of a course in the ratings objects list, or more succinctly, how many times a course has been rated. The shape of the graph indicates that using explicit ratings based on the scores objects might incur problems with the long tail problem as discussed in chapter 2.

4.4 Preprocessing

A lot of the data had to undergo preprocessing before it could be used. Below is an overview of what data had to be preprocessed and why.

4.4.1 Filtering

The first step in the preprocessing was the task of filtering certain data points. As live data was pulled for the experiments and their sub-routines, the preprocessing had to be undertaken every time one of the sub-routines in chapter 4.2 was run.

Ratings

Users with no ratings were filtered out to ensure that the ensuing matrix would be smaller and more efficient to run.

Scores

Some of the legacy data did not include all the necessary properties for the system to work. As such it was filtered out. This represented a small fraction of the data points, < 100.

4.4.2 Formatting

Several changes had to be made to the dataset before it could be used in the work. This stemmed from the fact that Utdannet's internal systems had undergone several iterations over the years, and legacy data that was usable required formatting before it could be properly recognised. In essence this consisted of flattening several object structures which had changed over the years, and renaming certain properties to follow the newest standard. As the task was simple in nature it could be done without the use of external tools.

4.5 Models

For the live experiments several models were prepared. More detailed information about their use, as well as a discussion about the parameters used will be found in the next chapter concerning the experiments.

4.5.1 User-based collaborative filtering using explicit ratings

The first model that was prepared is based on a well-known design; user-based collaborative filtering using explicit ratings. First, course ratings from the website are aggregated into a user-course matrix. From here, a user-user similarity matrix is generated through the use of a similarity function. In this model, the cosine similarity function is used. Having found a set of similar users, we iterate over a curated list of courses they have rated that we have not. This list of courses is run through a neighbourhood function. For the neighbourhood function, the standard method to use with CF is k-NN [13]. The output of the neighbourhood function is the same list of courses, but attached with it an average weighted score to indicate preference. This list is sorted and returned as the output of the model, and can be recommended to the user.

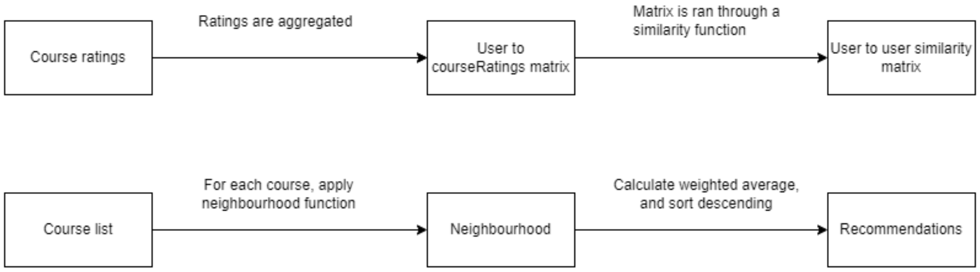

Course ratings
Ratings are aggregated
User to courseRatings matrix
Matrix is ran through a similarity function
User to user similarity matrix
Course list
For each course, apply neighbourhood function
Neighbourhood
Calculate weighted average, and sort descending
Recommendations

Figure 4.7: Diagram showing the process of generating recommendations based on user-based collaborative filtering using explicit ratings

4.5.2 User-based collaborative filtering using implicit ratings

The second model that was prepared is also based on a well-known and similar design to the first; user-based collaborative filtering using implicit ratings. First, implicit ratings are generated. In this work, a list of implicit ratings is generated from the scores object, where a course that is in progress is counted as a rating. The absence of a course in the scores object does not impact the model and its calculations negatively. From here, a user-user similarity matrix is generated through the use of a similarity function. In this model, the cosine similarity function is used. Having found a set of similar users, we iterate over a curated list of courses they have rated that we have not. This list of courses is run through a neighbourhood function. For the neighbourhood function, the standard method to use with CF is k-NN [13]. The output of the neighbourhood function is the same list of courses, but attached with it an average weighted score to indicate preference. This list is sorted and returned as the output of the model, and can be recommended to the user.

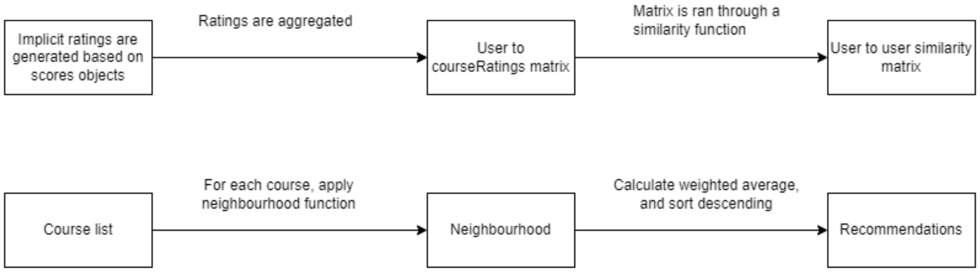

Implicit ratings are generated based on scores objects
Ratings are aggregated
User to courseRatings matrix
Matrix is ran through a similarity function
User to user similarity matrix
Course list
For each course, apply neighbourhood function
Neighbourhood
Calculate weighted average, and sort descending
Recommendations

Figure 4.8: Diagram showing the process of generating recommendations based on user-based collaborative filtering using implicit ratings

4.5.3 Popularity-based models

Two popularity-based models were implemented:

Popularity based on average completion rate

One model was developed focused on the average completion rate. This model strictly looked at instances where people had started the course, and calculated the mean completion rate for each course. The list of courses was then sorted in a descending order and the output could be recommended to the user.

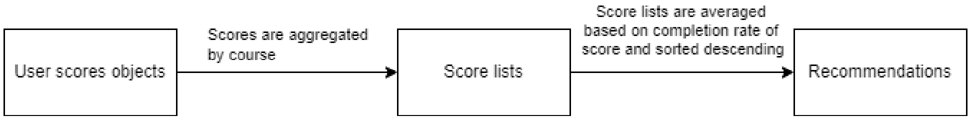

User scores objects
Scores are aggregated by course
Score lists
Score lists are averaged based on completion rate of score and sorted descending
Recommendations

Figure 4.9: Diagram showing the process of generating recommendations based on the popularity of courses with regard to their mean completion rate

Popularity based on number of views

Another model developed focused on the number of views registered for the course. This model looked at the number of people who had started the course, and counted a view regardless of how far into the course they had gotten. Views were aggregated, and the list of courses was then sorted in a descending order. The output could then be recommended to the user.

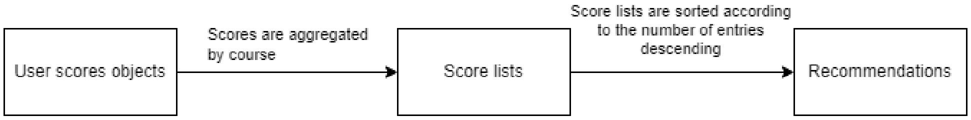

User scores objects
Scores are aggregated by course
Score lists
Score lists are sorted according to the number of entries descending
Recommendations

Figure 4.10: Diagram showing the process of generating recommendations based on the popularity of courses with regard to how many participants they have

4.6 Evaluation metrics

As discussed in Chapter 2, there are many viable metrics to choose between. For these experiments, the two most natural ones to ponder are the CTR and the dwell time.

CTR: An entry is stored every time someone is shown the independent variable. If a click is registered, another entry is stored.

Dwell time A modified version of the dwell time metric is also registered. If someone clicks on the independent variable, a course intro page is shown. Instead of storing the subject's time spent on the course page, we register information about the user's interactions with the course itself. As such we are able to determine an interest through their progress - or lack thereof - in the course. This metric can be problematic, more on that in the Discussion chapter.

5 Experiments

This chapter describes the experiments conducted in furtherance of the laid out research questions.

5.1 Experiment environment

Working with live data is a unique opportunity. While offline evaluation methods are robust, studies show that their conclusions are not necessarily reflected in live data [25]. Live evaluation also affords freedom in that an experiment can be designed and performed to test specific metrics, which historical data necessarily cannot because of limits on what has been recorded. As such the experiments below were conducted in a live environment; designed to challenge the research questions laid out in chapter 1.

5.2 Experiment 1

5.2.1 Hypothesis

The first experiment was conceptualised to test the premier research question, i.e. if the use of personalised recommendations would improve user engagement and time spent on the platform. Online evaluation with A/B testing offered a good chance to explore this question, as detailed information about user interactions with the system could be recorded and applied to evaluate appropriate metrics.

5.2.2 Variables

Independent variable: The independent variable of this experiment was a row of suggested courses for the user to complete. The list consisted of personal recommendations.

Dependent variable: The dependant variable of this experiment was twofold. The first metric to be measured was the CTR. While not telling a complete story, the CTR could tell something about the interest of the test subjects in the produced recommendations. The CTR neither paints a complete picture nor tests the research question in a meaningful way, and as such it was determined that an additional metric should be included; the dwell time. The dwell time could tell us something about the continued interest of a test subject in a particular clicked course. It would do this by measuring the progress of the test subject in the course, and establish a comparative metric for looking at user engagement with the platform and subsequently time spent on the platform.

Control variable: The control variables of this experiment was a row of suggested courses for the user to complete. The list consisted of impersonal recommendations.

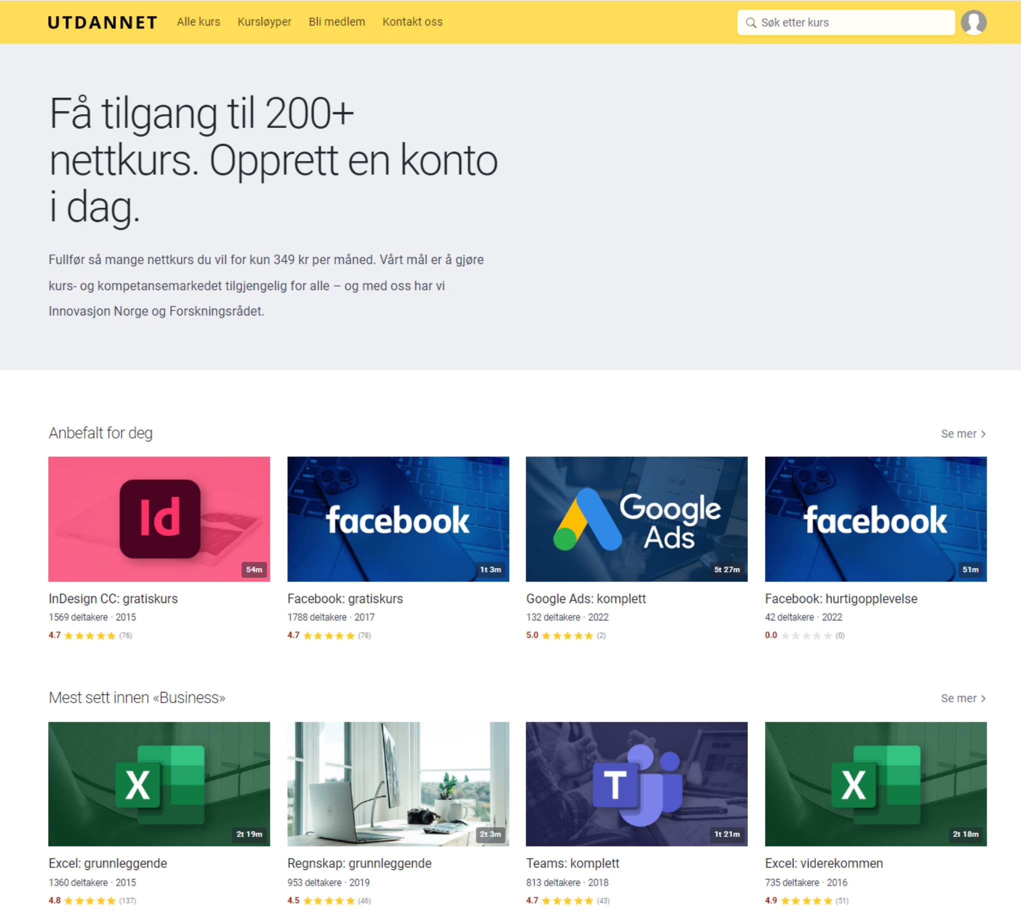
5.2.3 Logging policy

To determine the dependant variable, several new pieces of information had to be logged. Every time the view was shown to a test subject, a new entry was logged. This contained metadata about the object creation, as well as the list of courses shown to the subject. Another entry was created if a subject when presented with the view, clicked on one of the courses in the view and proceeded to a course page. Here some more metadata was stored, along with information about the course.

5.2.4 Variations

Control group

The control group, or group A, were shown a row of impersonal recommendations. The strategy used was a popularity-based measure as described in chapter 4, more specifically popularity based on the mean completion rate of each course.



Få tilgang til 200+ nettkurs. Opprett en konto i dag.

Fullfør så mange nettkurs du vil for kun 349 kr per måned. Vårt mål er å gjøre kurs- og kompetansemarkedet tilgjengelig for alle – og med oss har vi Innovasjon Norge og Forskningsrådet.

Anbefalt for deg

Se mer ›

InDesign CC: gratiskurs
1569 deltakere · 2015
4.7 ★★★★★ (76)

Facebook: gratiskurs
1788 deltakere · 2017
4.7 ★★★★★ (78)

Google Ads: komplett
132 deltakere · 2022
5.0 ★★★★★ (2)

Facebook: hurtigopplevelse
42 deltakere · 2022
0.0 ☆☆☆☆☆ (0)

Mest sett innen «Business»

Se mer ›

Excel: grunnleggende
1360 deltakere · 2015
4.8 ★★★★★ (137)

Regnskap: grunnleggende
953 deltakere · 2019
4.5 ★★★★★ (46)

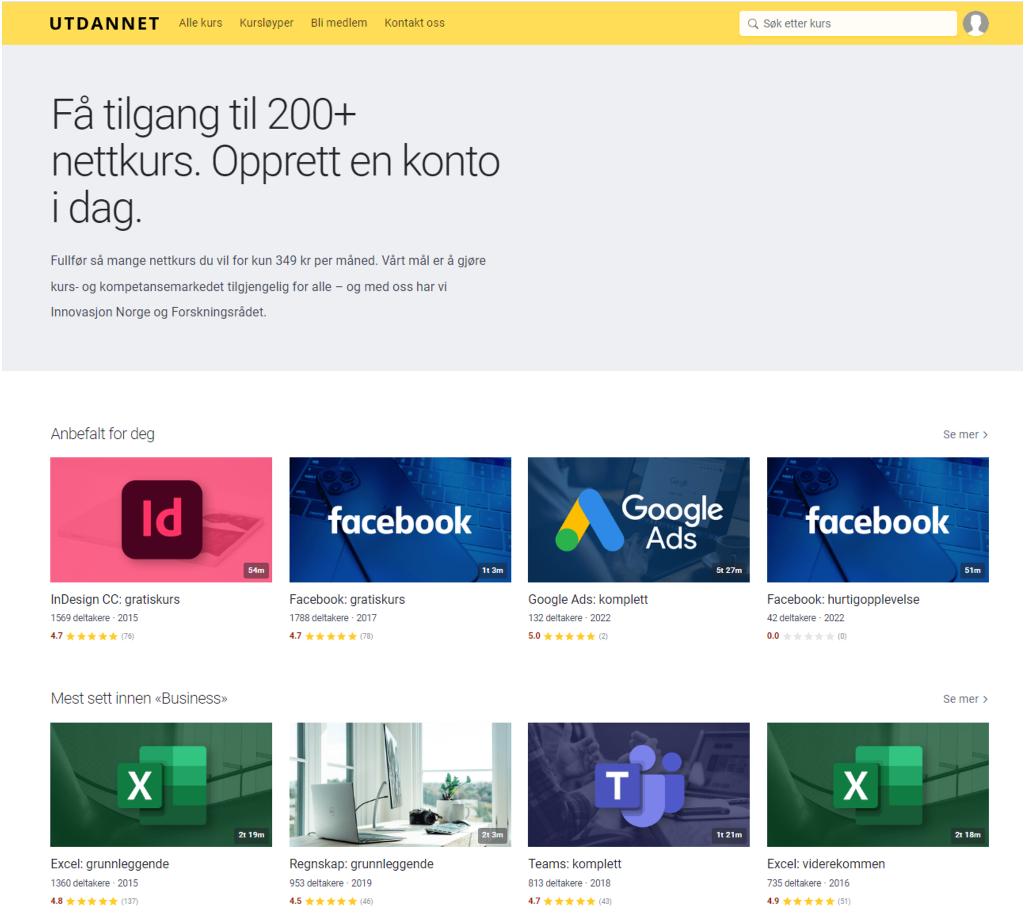
Teams: komplett
813 deltakere · 2018
4.7 ★★★★★ (43)

Excel: viderekommen
735 deltakere · 2016
4.9 ★★★★★ (51)

Figure 5.1: The independent variable shown to the control group.

Experimental group

The experimental group, or group B, were shown a row of personal recommendations. Offline analysis of historical data suggested that the use of a single recommendation strategy could prove insufficient. This was primarily the effect of two unique challenges, both rooted in the cold start problem. Firstly, as can be seen in table 4.3, the median number of courses started by a user was 2. Secondly, looking at tables 4.3 and 4.5, the median number of ratings per user was 1. Demanding personalised predictions on the basis of so little data could prove challenging, and so to ensure a degree of personalisation the decision was made to combine several strategies in a hybrid system. The hope was that the effect of an insufficiency in one area could be helped by utilising more than one strategy. In total, three strategies were utilised to ensure that the experimental group would always be shown recommendations. The premier strategy used was user-based collaborative filtering using explicit ratings. If not enough recommendations could be produced, the system would additionally produce recommendations according to user-based collaborative filtering using implicit ratings. Finally, if the user still could not be provided with enough recommendations, the system would revert to the same popularity-based metric as the control group were offered.



Få tilgang til 200+ nettkurs. Opprett en konto i dag.

Fullfør så mange nettkurs du vil for kun 349 kr per måned. Vårt mål er å gjøre kurs- og kompetansemarkedet tilgjengelig for alle – og med oss har vi Innovasjon Norge og Forskningsrådet.

Anbefalt for deg

InDesign CC: gratiskurs
1569 deltakere · 2015
4.7 ★★★★★ (76)

Facebook: gratiskurs
1788 deltakere · 2017
4.7 ★★★★★ (78)

Google Ads: komplett
132 deltakere · 2022
5.0 ★★★★★ (2)

Facebook: hurtigopplevelse
42 deltakere · 2022
0.0 ☆☆☆☆☆ (0)

Mest sett innen «Business»

Excel: grunnleggende
1360 deltakere · 2015
4.8 ★★★★★ (137)

Regnskap: grunnleggende
953 deltakere · 2019
4.5 ★★★★★ (46)

Teams: komplett
813 deltakere · 2018
4.7 ★★★★★ (43)

Excel: viderekommen
735 deltakere · 2016
4.9 ★★★★★ (51)

Figure 5.2: The independent variable shown to the experimental group. It looks the same as the control variable, but the recommendations themselves are different.

Model parameters

The CF models had parameters associated with them that could be altered. Below follows an account of which parameters were available to alter and which values were used in the experiment.

Mean centre ratings: Mean centring the ratings is a trick used to reduce the presence of user bias and provide a user-centric scale for calculating similarity. Take the example of two persons. Person A rates their courses with 4.5 stars every time, while person B rates their courses with different values in the range [2-4]. Without mean-centred ratings, the system would weigh 5 star ratings from persons A and B the same, while a system using mean-centred ratings would see that while person A previously had a mean of 4.5, person B had a mean of 3, signalling a strong difference (0.5 and 2) in relative rating. Mean-centring is common in CF systems and is the standard in LensKit's implementation [46], and as such it was performed here as well.

Similarity function: Both the cosine similarity function and the Jaccard index were implemented for this thesis. For the experiments the cosine function was used. The Jaccard index does not respond to duplicate entries, and since there was an overwhelming probability of duplicate ratings in a set of user ratings, the cosine similarity measure was chosen. It is also an industry standard because of its light footprint and good accuracy.

Neighbourhood size: As the k-NN neighbourhood function was used, a parameter of that function could be set to determine how many entries to consider when calculating the neighbourhood. Preliminary analysis showed that > 90% of the user-base would average four neighbours per prediction, with the rest showing a low similarity with lower ranked neighbours. As such the (max) neighbourhood size was set to 4.

Minimum neighbourhood size: In certain scenarios, no neighbours could be find for the prediction task. A threshold was set where at least one neighbour would have to be found. The plan was to increase this parameter, but any tuning showed a steep decrease in the number of possible predictions. As such, the minimum neighbourhood size was set to 1.

Number of recommendations: This parameter just controlled how many recommendations to return after the prediction task was done. For the experiment, rows of four recommendations would be shown, and as such, the parameter was set to 4.

Statistics and experiment run time

To ensure that the results of the experiment would be statistically significant, it was vital to record enough observations. At the same time, tweaking the other parameters of equation 2.3 on the sample size calculator [29] yields dramatic changes in the number of observations needed. Here follows a small discussion of each parameter:

p_1 In truth, both variants of the experiment are new. As such, a large effort was made to ensure that the control variable of the experiment be as close as possible to the site pre-experiment. More on that in chapter 7. A best-effort analysis was made to calculate a possible baseline conversion rate based on the number of courses started and historical user activity. The analysis concluded a conversion rate of < 0.01%. As assurance the conversion rate was assumed to be 1%.

p_2 As part of p_2 the MDE had to be determined. Preliminary analysis of user activity suggested that p_2 could be reduced to at least 4% $(1\% + 3\%$. The parameter could be further reduced if the number of observations increased above the expected number.

α The significance level is usually set to 5% [52].

β The statistical power, $1 - \beta$, is usually set to 80% [52].

Using these parameters and the assumed 3% of p_2, $n = 236$, i.e. each variant needed that many observations. A preliminary analysis suggested a run time of about four weeks.

5.3 Experiment 2

5.3.1 Hypothesis

The second experiment was conceptualised to test the second research question and to follow up on the first experiment, i.e. how the individual components of the hybrid system of experiment 1 would compare in driving user engagement and time spent on the platform.

5.3.2 Variables

Independent variable: The independent variable of this experiment was a row of suggested courses for the user to complete. The list consisted of personal recommendations.

Dependent variable: The dependant variable of this experiment was the CTR of courses presented to the users and the ensuing dwell time on those courses.

Control variable: The control variables of this experiment was a row of suggested courses for the user to complete. The list consisted of personal recommendations.

5.3.3 Logging policy

The same logging policy was used here as was for the first experiment.

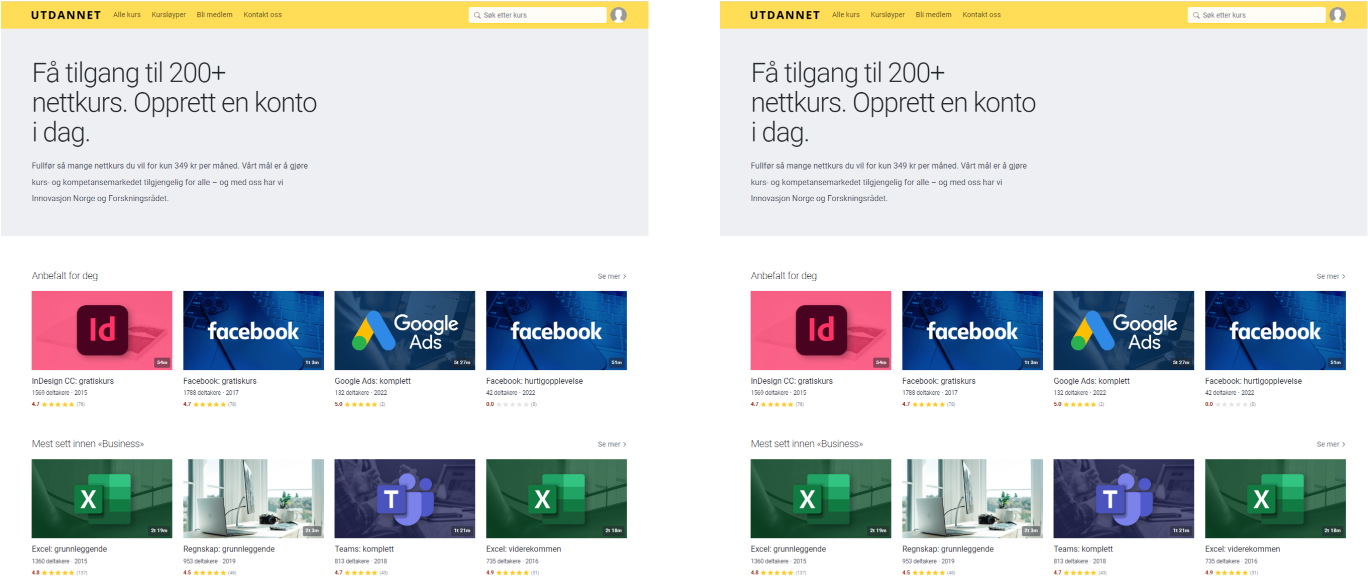
5.3.4 Variations

Control group

The control group, or group A, were shown a row of personal recommendations. The strategy used was one of the techniques used in the hybrid system of experiment 1; CF using explicit ratings.

Experimental group

The experimental group, or group B, were shown a row of personal recommendations. The strategy used was one of the techniques used in the hybrid system of experiment 1; CF using implicit ratings.



Få tilgang til 200+ nettkurs. Opprett en konto i dag.

Fullfør så mange nettkurs du vil for kun 349 kr per måned. Vårt mål er å gjøre kurs- og kompetansemarkedet tilgjengelig for alle – og med oss har vi Innovasjon Norge og Forskningsrådet.

Anbefalt for deg

InDesign CC: gratiskurs	Facebook: gratiskurs	Google Ads: komplett	Facebook: hurtigopplevelse
1569 deltakere · 2015	1788 deltakere · 2017	132 deltakere · 2022	42 deltakere · 2022
4.7 ★★★★★ (76)	4.7 ★★★★★ (78)	5.0 ★★★★★ (2)	0.0 ☆☆☆☆☆ (0)

Mest sett innen «Business»

Excel: grunnleggende	Regnskap: grunnleggende	Teams: komplett	Excel: viderekommen
1360 deltakere · 2015	953 deltakere · 2019	813 deltakere · 2018	735 deltakere · 2016
4.8 ★★★★★ (137)	4.5 ★★★★★ (46)	4.7 ★★★★★ (43)	4.9 ★★★★★ (51)

Få tilgang til 200+ nettkurs. Opprett en konto i dag.

Fullfør så mange nettkurs du vil for kun 349 kr per måned. Vårt mål er å gjøre kurs- og kompetansemarkedet tilgjengelig for alle – og med oss har vi Innovasjon Norge og Forskningsrådet.

Anbefalt for deg

InDesign CC: gratiskurs	Facebook: gratiskurs	Google Ads: komplett	Facebook: hurtigopplevelse
1569 deltakere · 2015	1788 deltakere · 2017	132 deltakere · 2022	42 deltakere · 2022
4.7 ★★★★★ (76)	4.7 ★★★★★ (78)	5.0 ★★★★★ (2)	0.0 ☆☆☆☆☆ (0)

Mest sett innen «Business»

Excel: grunnleggende	Regnskap: grunnleggende	Teams: komplett	Excel: viderekommen
1360 deltakere · 2015	953 deltakere · 2019	813 deltakere · 2018	735 deltakere · 2016
4.8 ★★★★★ (137)	4.5 ★★★★★ (46)	4.7 ★★★★★ (43)	4.9 ★★★★★ (51)

(a) Variant A

(b) Variant B

Figure 5.3: The control variable and the independent variable

Model parameters

The model parameters were chosen to be the same as in the first experiment to mimic the environment as closely as possible.

Statistics and experiment run time

Some of the parameters were the same. A discussion on p_1 and p_2 follows.

p_1 As with the first experiment, both variants were new to the user. A best-effort analysis was made to calculate a possible baseline conversion rate based on historical user activity and the analysis of ratings done in 4.5. The analysis concluded a conversion rate of $< 0.001\%$. As in the first experiment, the number was increased as an assurance to 1%.

p_2 Part of p_2 requires determining the MDE. Preliminary analysis of user activity suggested that p_2 could be reduced to at least 6% $(1\% + 5\%$. This parameter could be further reduced if the number of observations increased above the expected number.

Using these parameters and the assumed 6% of p_2, $n = 97$, i.e. each variant needed that many observations. A preliminary analysis suggested a run time of about three weeks.

6 Results

This chapter presents the results of the experiments conducted.

6.1 Experiment 1

6.1.1 General statistics

The first experiment started the 11th of June and ran until the 15th of July. In this experiment, the total number of users was 421, however not all of them were active during the experiment. There are 409 users from which we collected observations and the total number of observations collected is 1558. Table 6.1 shows the general statistics from this experiment.

Run-time of experiment	34 days
Total users	421
Total users with observations	409
Total observations	1558

Table 6.1: General statistics about the experiment

There are differences between the variants. While the total number of users for variant A was 216, variant B had 205 participants. Not all of the users had recorded observations. 209 users in variant A had recorded observations while 200 users did in variant B. In total, 917 observations were recorded for variant A, while 641 observations were recorded for variant B. This can be observed below in table 6.2.

	A	B
Total users	216	205
Total users with observations	209	200
Total observations	917	641

Table 6.2: General statistics about the variants

6.1.2 Metrics

CTR

As described previously, the CTR is defined as the rate at which a document is clicked on relative to its total number of views. Below in table 6.3 are statistics concerning the CTR for this experiment.

	A	B
Total observations	917	641
Total clicks	16	9
Clickthrough rate	1.74%	1.40%

Table 6.3: Statistics supporting the CTR

Dwell time

As replacement for the traditional definition, a more nuanced approach to dwell time has been recorded, wherein the progress of the subject on the course related to the clicked page has been recorded. The results were recorded on the last day of the experiment. Below in table 6.4 are statistics about the dwell time for the this experiment.

	A	B
Total showed course pages	16	9
Total started courses	4	0
Course start rate	25%	0%
Mean course completion rate	11.56%	0%
Median course completion rate	3.46%	0%

Table 6.4: Statistics supporting the dwell time

6.2 Experiment 2

6.2.1 General statistics

The second experiment started the 15th of July and ran until the 21st of August. In this experiment, the total number of users was 476, however not all of them were active during the experiment. There are 193 users from which we collected observations and the total number of observations collected is 853. Table 6.5 shows the general statistics from this experiment.

Run-time of experiment	37 days
Total users	476
Total users with observations	193
Total observations	853

Table 6.5: General statistics about the experiment

There are differences between the variants. While the total number of users for variant A was 216, variant B had 205 participants. Not all of the users had recorded observations. 209 users in variant A had recorded observations while 200 users did in variant B. In total, 917 observations were recorded for variant A, while 641 observations were recorded for variant B. This can be observed below in table 6.6.

	A	B
Total users	217	259
Total users with observations	5	188
Total observations	24	829

Table 6.6: General statistics about the variants

6.2.2 Metrics

CTR

As described previously, the CTR is defined as the rate at which a document is clicked on relative to its total number of views. Below in table 6.7 are statistics concerning the CTR for this experiment.

	A	B
Total observations	24	829
Total clicks	1	20
Clickthrough rate	4.17%	2.41%

Table 6.7: Statistics supporting the CTR

Dwell time

As replacement for the traditional definition, a more nuanced approach to dwell time has been recorded, wherein the progress of the subject on the course related to the clicked page has been recorded. The results were recorded on the last day of the experiment. Below in table 6.8 are statistics about the dwell time for the this experiment.

	A	B
Total showed course pages	1	20
Total started courses	1	3
Course start rate	100%	15%
Mean course completion rate	15.38%	25.37%
Median course completion rate	15.38%	15.69%

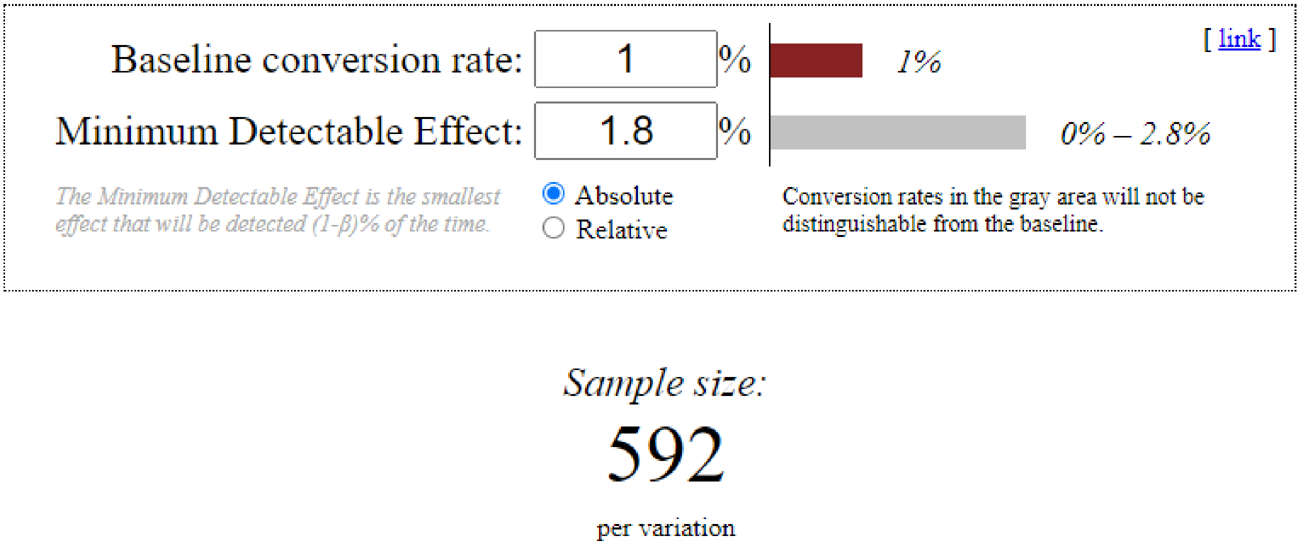
Table 6.8: Statistics supporting the dwell time

7 Discussion

This chapter discusses the results of the experiments conducted, the choice of models used for the experiments, some of the limitations of the work, and makes suggestions for future research.

7.1 Experiment 1

Using the number of observations registered during the first experiment on the sample size calculator [29], and in conjunction with the previously discussed 1% baseline conversion rate, we can narrow the MDE to 1.8%. This yields a window of 0% – 2.8% where the effect will be indistinguishable from the baseline version, as can be seen below in figure 7.1.



[link]

Baseline conversion rate: 1 % 1%

Minimum Detectable Effect: 1.8 % 0% – 2.8%

The Minimum Detectable Effect is the smallest effect that will be detected (1-β)% of the time.

● Absolute
○ Relative

Conversion rates in the gray area will not be distinguishable from the baseline.

Sample size:

592

per variation

Figure 7.1: The different parameters of equation 2.3 for the first experiment.

Looking at the results of experiment 1, both the CTR and the dwell time falls within this range, and as such the results of the experiment are inconclusive. Changing the baseline conversion rate for the one observed in the control variable yields similar results.

This experiment was designed to challenge the first research question, i.e. can the use of personalised recommendations aid in driving user engagement and time spent on the platform. Several studies show that the use of CF improves the effectiveness of the RS in different scenarios [53]. In looking at the CTR and dwell time of variants A and B, and with with the 2.8% window allotted us, it is not possible to verify this presupposition, and as such more research is needed to make any conclusions.

Below follows a list of possible reasons for why the results are as they are:

The effectiveness of the RS: As discussed previously, preliminary analysis of the historical data suggested that a subset of users would not receive personalised recommendations due to their limited interaction with the system, as can be seen in tables 4.3 and 4.5. While a hybrid model was used, collaborative filtering using implicit ratings can yield unsatisfying results. Implicit ratings do not necessarily infer interest, as they are calculated based on user activity rather than on user interest [14]. The inclusion of a CB approach would likely have improved the effectiveness of the system [54]. More about this in section 7.3.

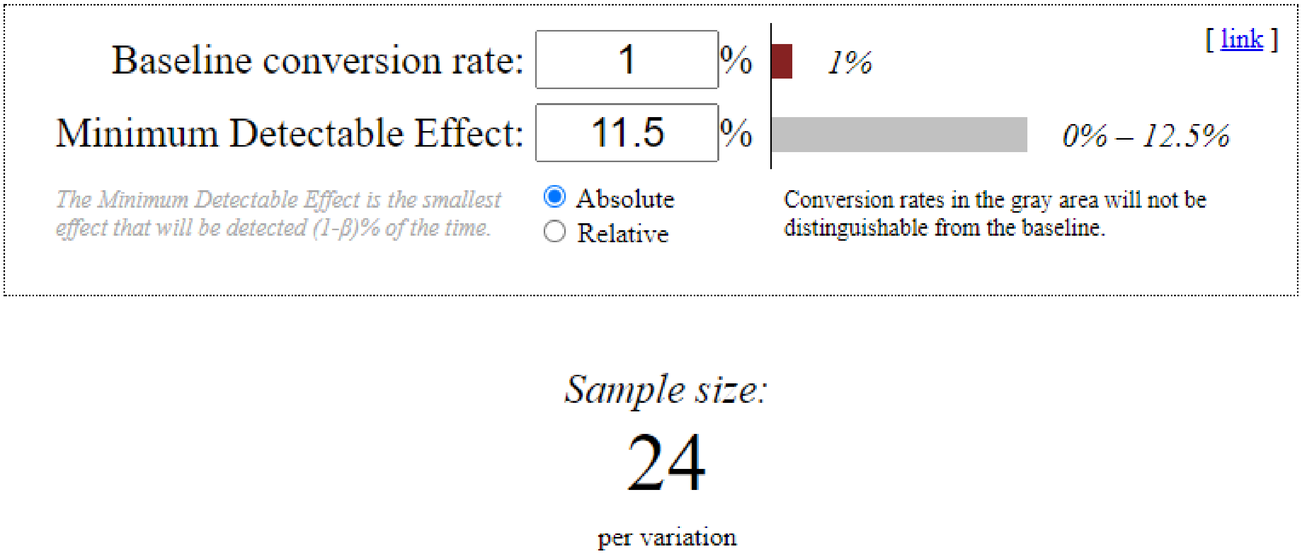
Adoption rate: Adopting new features takes time [33]. Depending on the size of the feature, its utility to the user, how disruptive it is to existing practice, and how familiar users are with it in theory are all variables that will affect how fast people will adopt it. In the case of this experiment, the feature was neither big nor disruptive in nature, but it was unfamiliar to the user and may have been incompatible with the way users browsed for new courses. Receiving personalised recommendations was new to the test subjects, and might have been glanced over in the same way the adjacent rows have been. Another explanation might be the utility of the personalised recommendations to the users. Utdannet is a subscription-based service, i.e. users pay for access to the catalogue. This suggests that users might have sought out the service for a specific course or for a specific category of courses. With the strategies implemented such a pattern could be difficult to discern, and with that the utility of the recommendations drop. It could also tell us something about the utility of Utdannet as a service to the user after the completion of the courses that attracted them to the service.

The effectiveness of the test: Some research indicates that in certain cases, the efficacy of A/B testing can be as low as 10%-12% [45] [55] [56]. Another issue might lie with presentation bias, where the presentation of the feature might have affected how attentive users would be to its presence and utility.

Modified dwell time: The modified version of the dwell time metric can be problematic. While the course start rate can tell us something about the effectiveness of the RS, users were served these recommendations at different times in the experiment. As such, using progress to compare the variants can be problematic. That was taken into account and is one of the reasons for why several metrics were chosen to evaluate the independent variable.

7.2 Experiment 2

Similar to the first experiment, we alter the MDE. Looking at table 6.6 and consulting the sample size calculator, the small number of observations in variant A only allows us to set the MDE to 11.5%. This yields a window of $0\% - 12.5\%$ where the effect will be indistinguishable from the baseline version, as can be seen below in figure 7.2.



[link]

Baseline conversion rate: 1 % — 1%

Minimum Detectable Effect: 11.5 % — 0% – 12.5%

The Minimum Detectable Effect is the smallest effect that will be detected (1-β)% of the time.

● Absolute
○ Relative

Conversion rates in the gray area will not be distinguishable from the baseline.

Sample size:

24

per variation

Figure 7.2: The different parameters of equation 2.3 for the second experiment.

Looking at the results of experiment 2, both the CTR and the dwell time falls within this range, and as such the results of the experiment are inconclusive. Changing the baseline conversion rate for the one observed in the control variable yields similar results.

This experiment was designed to challenge the second research question, i.e. how the use of implicit ratings compares to the use of explicit ratings with CF in driving user engagement and time spent on the platform. Looking at table 6.7, there is a large discrepancy in the number of registered observations and the total number of clicks. Because of the small number of observations made for the control group, the window allotted us for the MDE is 11.5%. As such, we can not conclusively say anything about the CTR and dwell time. What can be commented on is table 6.2. Looking at the number of observations, and furthermore, the number of people with observations can tell us something about the effectiveness of single-strategy models in a platform like Utdannet.

Below follows a list of possible reasons for why the results are as they are:

The effectiveness of the RS: As can be seen in table 6.5, there is a large discrepancy in the number of observations made between the control variable and the independent variable. This can likely be attributed to the large discrepancy in the number of implicit and explicit ratings, as can be surmised from tables 4.3 and 4.5. As discussed above, users may have joined Utdannet because of a specific interest in a course or a category of courses. The use of implicit ratings, while providing for a far larger number of observations, might not extend to an increase in preference among the users.

Adoption rate: The argument is the same one presented for the first experiment.

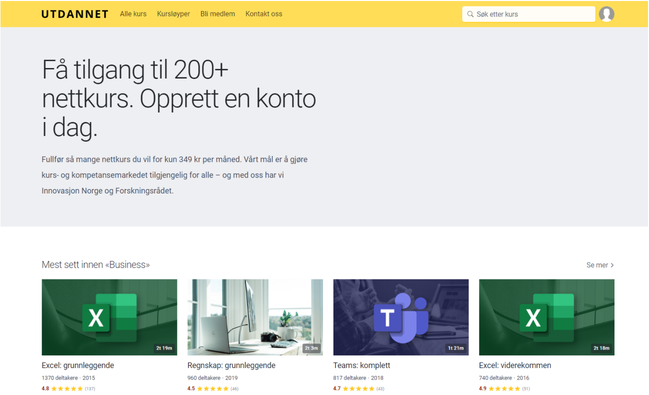
The effectiveness of the test: The argument is the same one presented for the first experiment.

Modified dwell time: The argument is the same one presented for the first experiment.

7.3 Choice of models

7.3.1 Popularity-based model

The control variable of experiment 1 used a popularity-based model. In truth, both the control variable and the independent variable were new to the user, so it was important that the control variable mirrored the pre-existing state of the system as much as possible. Before the experiment, rows of suggestions were offered to the user, based on the popularity of courses within pre-made sub-categories.



Få tilgang til 200+ nettkurs. Opprett en konto i dag.

Fullfør så mange nettkurs du vil for kun 349 kr per måned. Vårt mål er å gjøre kurs- og kompetansemarkedet tilgjengelig for alle – og med oss har vi Innovasjon Norge og Forskningsrådet.

Mest sett innen «Business»

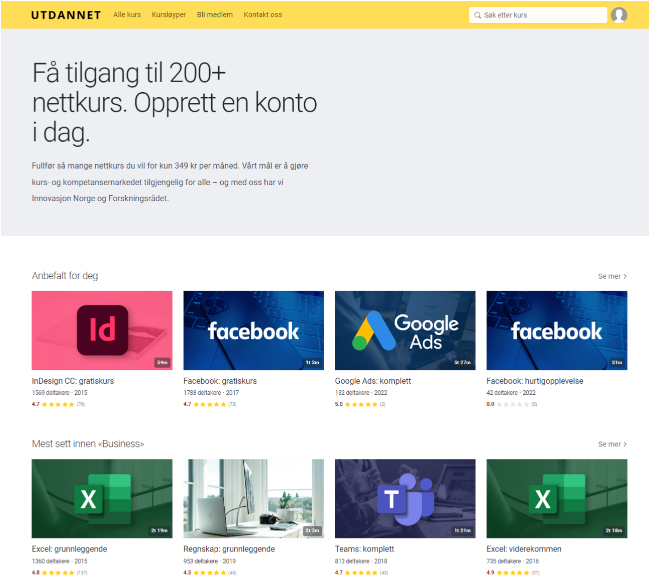
Excel: grunnleggende
1370 deltakere · 2015
4.8 ★★★★★ (137)

Regnskap: grunnleggende
960 deltakere · 2019
4.5 ★★★★★ (46)

Teams: komplett
817 deltakere · 2018
4.7 ★★★★★ (43)

Excel: viderekommen
740 deltakere · 2016
4.9 ★★★★★ (51)

Figure 7.3: The site pre-experiment



Få tilgang til 200+ nettkurs. Opprett en konto i dag.

Fullfør så mange nettkurs du vil for kun 349 kr per måned. Vårt mål er å gjøre kurs- og kompetansemarkedet tilgjengelig for alle – og med oss har vi Innovasjon Norge og Forskningsrådet.

Anbefalt for deg

InDesign CC: gratiskurs
1569 deltakere · 2015
4.7 ★★★★★ (76)

Facebook: gratiskurs
1788 deltakere · 2017
4.7 ★★★★★ (78)

Google Ads: komplett
132 deltakere · 2022
5.0 ★★★★★ (2)

Facebook: hurtigopplevelse
42 deltakere · 2022
0.0 ☆☆☆☆☆ (0)

Mest sett innen «Business»

Excel: grunnleggende
1360 deltakere · 2015
4.8 ★★★★★ (137)

Regnskap: grunnleggende
953 deltakere · 2019
4.5 ★★★★★ (46)

Teams: komplett
813 deltakere · 2018
4.7 ★★★★★ (43)

Excel: viderekommen
735 deltakere · 2016
4.9 ★★★★★ (51)

Figure 7.4: The site during the experiment

In essence the control variable functioned as an aggregation of the rows below it. More popular as a baseline is perhaps the random function, which yields a random assortment of items without bias. While this baseline variant was considered, it was paramount that the control variable was as close as possible to a known system and could work as a proper baseline. That is also why a simple version of the popularity metric was used, even though studies suggest that more nuanced popularity metrics can improve performance by as much as 70% [57].

7.3.2 Hybrid model

More approaches to RS have been developed since the inception of CF in the nineties [58]. Modern, more complex models can provide far better results than CF, depending on the available data and the prediction task. Regardless, the decision was made to implement a hybrid model consisting of variations of CF, for different reasons:

Premier research question: While the premier research question regards the effect of RS, it was deliberately left vague to allow for choice in the selection of which model to employ. As such the focus of the work was not necessarily on which model to use, but rather on exploiting the opportunity of working with live data to conduct a live experiment.

Related work: As discussed in chapter 3, previous work mimicked a lot of Utdannet's circumstances. Work on video recommenders has shown that novel approaches such as CF and CB provide the best results on large-scale datasets [39] [59]. Urdaneta-Ponte et al. show similarly that a lot of work on the domain still uses novel approaches, rather than complex, context-specific models [34]. As my work is in the intersection of several domains, it felt appropriate to choose a model that was widely reflected in all domains.

Dataset properties: As seen in figures 4.3 and 4.5, both the explicit and implicit ratings matrices were sparse, where the median user had started two courses, and had produced a single rating. This preliminary analysis of the data suggested that a single strategy could prove ineffective, and as such it was determined that a hybrid system was needed to ensure a degree of personalisation. The inclusion of a model-based strategy such as SVDs was considered as it is an effective method on sparse data, but as the accuracy of SVDs improves with the size of the dataset and mine was relatively small, the idea was dropped [39]. A CB approach was conceptualised and partly developed, but a lack of necessary metadata about the courses and the users obstructed such an implementation, which is often seen used in hybrid systems alongside CF [54].

Time to develop: As the RS had to be developed from the ground up to work in a live environment, it was essential to limit the number of strategies to implement. As using variations of CF implied re-use between the strategies and a lower time-to-develop, it was a natural choice.

Minimising adverse affects: Preliminary analysis and offline evaluation of historical data suggested that a lot of users would suffer from the cold start problem, as discussed in chapter 2. Looking at table 4.5, the median user had provided one rating, ensuring that a single-strategy approach would suffer in its effectiveness. Another known problem, the long tail problem, can also be seen in tables 4.5 and 4.6, where a small number of courses had many participants and ratings, while the majority had few. Using a hybrid strategy alleviated some of the problem, though more work should have gone into combating these effects.

7.3.3 Single-strategy CF

The single-strategy approaches were applied in the second experiment. Building on the first experiment and the second research question, it was of interest to evaluate the performance of each individual strategy that composed the backbone of the hybrid strategy of the first experiment. This is discussed in more detail in the results section and in chapter 5.

7.4 Limitations

7.4.1 Technical challenges

As previously discussed, I had to implement both the recommender systems and the A/B testing necessities, which took a considerable time to conceptualise, develop, implement, and test. Below is some discussion of what was done, why, and how it affected the end work:

Recommender system

The original plan was to use LensKit more actively in the experiments. LensKit provides a set of complete, tested algorithms that span CF, CB, and model-based strategies that would have made the implementation and setup of the experiment much faster. After initial discussions with Utdannet, it was made clear that a Python-based framework would require too much work on their end to facilitate integration, and as such part of the work consisted of porting algorithm implementations from LensKit's Python framework to JavaScript. While it was almost an identical port, it still slowed work down considerably.

A/B testing

As with the RS, the A/B testing suite also needed to be manually implemented. Several A/B testing frameworks exist, such as Google Optimize [60], but bridging them with Utdannet's systems required too many resources on their part. The scheme that was devised instead can be found in chapter 4.2, where different sub-routines were charged with aspects of updating the necessary structures to maintain, run, and update the A/B test and its necessary components.

7.4.2 Working around existing systems

All of the systems were developed to work with Utdannet's existing systems. Part of that work forced me to create ad hoc solutions that were sub-optimal for further use. In particular, because I did not have write access to Utdannet's databases, a lot of the intermittent results and necessary data were stored on JSON files. Pre-experiment testing showed that as the experiments would have progressed, the size of these files would have increased drastically. Extended work would likely have degraded the user-experience. As such, the method responsible for updating the implicit ratings could only be run a two times per day (as opposed to the expected 50), and was renamed *onHalfDay*. While not detrimental to the experiments, considerations like this slowed down work and may have degraded the effectiveness of the system slightly. As shown in a recent survey on RS in MOOCs [24], other systems struggle with space complexity too and it is an active field of research.

7.5 Future work

7.5.1 Different model

Data sparsity and the cold start problem are known limitations when using CF-based strategies [23]. The sparsity of the data proved to have a bigger effect on the data in the live test than it showed in offline evaluations. As mentioned in chapter 7.2, the original plan was to include a CB strategy for the hybrid model. As CB works very well independently of user-provided ratings and is commonly used with CF in hybrid systems [53], it would have likely increased the effectiveness of the system [54]. A different approach, using SVDs, has been shown to work on sparse data and aid with the cold start problem [19]. Investigating both these approaches and their effect on CTR and dwell time could prove useful.

7.5.2 Running several, longer experiments

Research shows that in certain cases, the efficacy of A/B testing can be as low as 10%-12% [45] [55] [56]. Suggested practice is running several experiments over longer time periods to see if the results remain the same or if they change [7]. There are numerous reasons for why one might see a difference. In Utdannet's circumstances, early communications with them suggested a big difference in user activity during the summer holidays (in Norwegian: "fellesferie") where most people are away. As the experiments were running during that period, the difference in user activity, and more importantly, user diversity, might affect results, and running more experiments could yield different results. Running more experiments would also improve adoption rate as the presence of the independent variable would be normalised.

8 Conclusion

8.1 Summary of contributions

This thesis aimed to explore the research questions as laid out in chapter 1. More specifically, through the use of A/B testing, online evaluation was performed to investigate whether personal recommendations would drive user engagement and time spent on the Utdannet platform. Furthermore, a second experiment was conducted to look at a comparison of collaborative filtering techniques using explicit and implicit ratings, as they were the principal components of the hybrid system of the first experiment, and how they individually contributed to user engagement and time spent on the platform. In this section I will summarise the experiments and their findings.

RQ1 *Does the use of personalised recommendations affect user engagement and time spent on the platform in non-formal education?*

An A/B test was conducted to test the first research question. As metrics, the CTR and a modified version of the dwell time metric were chosen. The results of the experiment were inconclusive. As such this thesis can not say anything decisively about the inclusion of personalised recommendations on a platform like Utdannet. There are many possible reasons for why the test was indecisive, including the effectiveness of the model and problems with the adoption rate.

RQ2 *When using personalised recommendations, how does the use of implicit ratings compare to the use of explicit ratings with CF in affecting user engagement and time spent on the platform?*

An A/B test was conducted to test the second research question. As metrics, the CTR and a modified version of the dwell time metric were chosen. The results of the experiment were inconclusive. As such this thesis can not say anything decisively about the differences of using explicit and implicit ratings combined with CF on a platform like Utdannet. Of interest is the large discrepancy in the number of observations between the variants, as seen in table 6.6. This reinforces results that others have found regarding problems caused by using sparse data [19]. There are many possible reasons for why the test was indecisive, including the effectiveness of the strategies and problems with the adoption rate.

To combat some of the drawbacks of the system, several solutions are proposed for future research. Using a model that is more lenient with sparse data would have likely increased the effectiveness of the system and yielded conclusive results about the effects of personal recommendations. Another suggestion is to run more experiments, for longer periods of time, as research shows that the success rate of A/B testing can diminish considerably in certain circumstances [45].

8.2 Lessons learned

I definitely learned a lot working on this thesis which I will bring with me. Working with, and around, third party systems is a challenge. Some of the assumptions I made early on in the work turned out to be untrue and it caused me a lot of extra work later. I also underestimated the amount of work that needed to be done (re-)implementing some of the recommendation strategies and enabling them to run on Utdannet's systems. All in all this thesis has been a lot of hard work, but I am really appreciative of the opportunities I have been given and the lessons I have learned.

A Appendix

References

[1] F. Alyari and N. J. Navimipour, Recommender systems: A systematic review of the state of the art literature and suggestions for future research, *Kybernetes* 2018, 2018.

[2] S. Kumar Basak, M. Wotto, and P. Belanger, E-learning, m-learning and d-learning: Conceptual definition and comparative analysis, *E-learning and Digital Media*, vol. 15, no. 4 2018, pp. 191–216, 2018.

[3] *The merlot system provides access to curated online learning and support materials and content creation tools, led by an international community of educators, learners and researchers.* available from: `https://merlot.org/merlot/index.htm`.

[4] *Explore. create. collaborate.* available from: `https://www.oercommons.org/`.

[5] S. Dhawan, Online learning: A panacea in the time of covid-19 crisis, *Journal of Educational Technology Systems* [online], vol. 49, no. 1 2020, pp. 5–22, 2020. DOI: `10.1177/0047239520934018`. eprint: `https://doi.org/10.1177/0047239520934018`. available from: `https://doi.org/10.1177/0047239520934018`.

[6] Mooc.org, *Massive open online courses: An edx site*. available from: `https://www.mooc.org/`.

[7] K. Hofmann, *Onlineevaluationforinformationretrieval - microsoft.com*. available from: `https://www.microsoft.com/en-us/research/wp-content/uploads/2016/06/ftir-online-evaluation-final-journal.pdf`.

[8] *Få tilgang til nettkurs*. available from: `https://www.utdannet.no/`.

[9] Available from: `https://www.youtube.com/`.

[10] *Listening is everything*. available from: `https://www.spotify.com/`.

[11] Available from: `https://www.google.com/`.

[12] E. Rich, User modeling via stereotypes, *Cognitive Science* [online], vol. 3, no. 4 1979, pp. 329–354, 1979, ISSN: 0364-0213. DOI: https://doi.org/10.1016/S0364-0213(79)80012-9. available from: https://www.sciencedirect.com/science/article/pii/S0364021379800129.

[13] A. Darmawan and I. B. K. Manuaba, A comparative study between collaborative filtering techniques and generate personalized story recommendations for the vixio application,

[14] K. Falk, *Practical recommender systems*. Manning Publications Company, 2019.

[15] X. Su and T. Khoshgoftaar, A survey of collaborative filtering techniques, *Adv. Artificial Intellegence* [online], vol. 2009 Jan. 2009, Jan. 2009. DOI: 10.1155/2009/421425.

[16] S. Bag, S. K. Kumar, and M. K. Tiwari, An efficient recommendation generation using relevant jaccard similarity, *Information Sciences* [online], vol. 483 2019, pp. 53–64, 2019, ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2019.01.023. available from: https://www.sciencedirect.com/science/article/pii/S0020025519300325.

[17] S. D. Jadhav and H. Channe, Comparative study of k-nn, naive bayes and decision tree classification techniques, *International Journal of Science and Research (IJSR)*, vol. 5, no. 1 2016, pp. 1842–1845, 2016.

[18] R. Cañamares, P. Castells, and A. Moffat, Offline evaluation options for recommender systems, *Information Retrieval Journal* [online], vol. 23, no. 4 Aug. 2020, pp. 387–410, Aug. 2020, ISSN: 1573-7659. DOI: 10.1007/s10791-020-09371-3. available from: https://doi.org/10.1007/s10791-020-09371-3.

[19] K. Vahidy Rodpysh, S. J. Mirabedini, and T. Banirostam, Resolving cold start and sparse data challenge in recommender systems using multi-level singular value decomposition, *Computers Electrical Engineering* [online], vol. 94 2021, p. 107 361, 2021, ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2021.107361. available from: https://www.sciencedirect.com/science/article/pii/S0045790621003311.

[20] C.-S. M. Wu, D. Garg, and U. Bhandary, "Movie recommendation system using collaborative filtering," in *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, 2018, pp. 11–15. DOI: 10.1109/ICSESS.2018.8663822.

[21] Cui, Bei-Bei, Design and implementation of movie recommendation system based on knn collaborative filtering algorithm, *ITM Web Conf.* [online], vol. 12 2017, p. 04 008, 2017. DOI: 10.1051/itmconf/20171204008. available from: https://doi.org/10.1051/itmconf/20171204008.

[22] V. Subramaniyaswamy, R. Logesh, M. Chandrashekhar, A. Challa, and V. Vijayakumar, A personalised movie recommendation system based on collaborative filtering, *International Journal of High Performance Computing and Networking* [online], vol. 10, no. 1-2 2017, pp. 54–63, 2017. DOI: 10.1504/IJHPCN.2017.083199. eprint: https://www.inderscienceonline.com/doi/pdf/10.1504/IJHPCN.2017.083199. available from: https://www.inderscienceonline.com/doi/abs/10.1504/IJHPCN.2017.083199.

[23] A. Klašnja-Milićević, M. Ivanović, and A. Nanopoulos, Recommender systems in e-learning environments: A survey of the state-of-the-art and possible extensions, *Artificial Intelligence Review* [online], vol. 44, no. 4 Dec. 2015, pp. 571–604, Dec. 2015, ISSN: 1573-7462. DOI: 10.1007/s10462-015-9440-z. available from: https://doi.org/10.1007/s10462-015-9440-z.

[24] A. Khalid, K. Lundqvist, and A. Yates, *Recommender systems for moocs: A systematic literature survey (january 1, 2012 – july 12, 2019) – international review of research in open and distributed learning*, Jan. 2021. available from: https://www.erudit.org/en/journals/irrodl/2020-v21-n4-irrodl05778/1074622ar/.

[25] A. Turpin and F. Scholer, "User performance versus precision measures for simple search tasks," Jan. 2006, pp. 11–18. DOI: 10.1145/1148170.1148176.

[26] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay, Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search, *ACM Trans. Inf. Syst.* [online], vol. 25 Apr. 2007, Apr. 2007. DOI: 10.1145/1229179.1229181.

[27] Y. Yue, R. Patel, and H. Roehrig, "Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data," Apr. 2010, pp. 1011–1018. DOI: `10.1145/1772690.1772793`.

[28] E. Yilmaz, M. Verma, N. Craswell, F. Radlinski, and P. Bailey, Relevance and effort: An analysis of document utility, *CIKM 2014 - Proceedings of the 2014 ACM International Conference on Information and Knowledge Management* [online] Nov. 2014, pp. 91–100, Nov. 2014. DOI: `10.1145/2661829.2661953`.

[29] E. Miller, *Sample size calculator*. available from: `https://www.evanmiller.org/ab-testing/sample-size.html`.

[30] N. M. Follow, *Wsdm 2011 - nicolaas matthijs and filip radlinski*. available from: `https://www.slideshare.net/nicolaasmatthijs/wsdm-2011-nicolaas-matthijs-and-filip-radlinski`.

[31] R. Kohavi and R. Longbotham, Online controlled experiments and a/b testing, in. Jan. 2017, pp. 922–929. DOI: `10.1007/978-1-4899-7687-1_891`.

[32] H. Gui, *Network a/b testing: From sampling to estimation*. available from: `https://hanj.cs.illinois.edu/pdf/www15_hgui.pdf`.

[33] R. W. Olshavsky, Time and the rate of adoption of innovations, *Journal of Consumer Research* [online], vol. 6, no. 4 1980, pp. 425–428, 1980, ISSN: 00935301, 15375277. available from: `http://www.jstor.org/stable/2488744` [Accessed 08/30/2022].

[34] M. C. Urdaneta-Ponte, A. Mendez-Zorrilla, and I. Oleagordia-Ruiz, Recommendation systems for education: Systematic review, *Electronics* [online], vol. 10, no. 14 2021, 2021, ISSN: 2079-9292. DOI: `10.3390/electronics10141611`. available from: `https://www.mdpi.com/2079-9292/10/14/1611`.

[35] A. U A and S. Biradar, *Recommender systems: A survey*, Apr. 2016. available from: `https://www.academia.edu/21284287/Recommender_Systems_A_Survey`.

[36] M. Ö. Karakaya and T. Aytekin, Effective methods for increasing aggregate diversity in recommender systems, *Knowledge and Information Systems* [online], vol. 56, no. 2 Aug. 2018, pp. 355–372, Aug. 2018, ISSN: 0219-3116. DOI: `10.1007/s10115-017-1135-0`. available from: `https://doi.org/10.1007/s10115-017-1135-0`.

[37] D. Onah and J. Sinclair, "Collaborative filtering recommendation system: A framework in massive open online courses," Mar. 2015. DOI: `10.13140/RG.2.1.5023.4409`.

[38] *Movielens*, Dec. 2021. available from: `https://grouplens.org/datasets/movielens/`.

[39] L. E. Molina and S. Bhulai, Recommendation system for netflix, *Retrieved June*, vol. 6 2018, p. 2021, 2018.

[40] A. Fabijan, P. Dmitriev, H. Holmstrom Olsson, and J. Bosch, "Online controlled experimentation at scale: An empirical survey on the current state of a/b testing," in *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 2018, pp. 68–72. DOI: `10.1109/SEAA.2018.00021`.

[41] R. Rahutomo, Y. Lie, A. S. Perbangsa, and B. Pardamean, "Improving conversion rates for fashion e-commerce with a/b testing," in *2020 International Conference on Information Management and Technology (ICIMTech)*, 2020, pp. 266–270. DOI: `10.1109/ICIMTech50083.2020.9210947`.

[42] X. Amatriain and J. Basilico, Recommender systems in industry: A netflix case study, in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 385–419, ISBN: 978-1-4899-7637-6. DOI: `10.1007/978-1-4899-7637-6_11`. available from: `https://doi.org/10.1007/978-1-4899-7637-6_11`.

[43] X. Wang, Y. Zhang, S. Yu, X. Liu, Y. Yuan, and F.-Y. Wang, "E-learning recommendation framework based on deep learning," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017, pp. 455–460. DOI: `10.1109/SMC.2017.8122647`.

[44] D. Gamage, I. Perera, and S. Fernando, Improving e-learning in mooc to meet challenges in 21 st century,

[45] J. Renz, D. Hoffmann, T. Staubitz, and C. Meinel, "Using a/b testing in mooc environments," in *Proceedings of the Sixth International Conference on Learning Analytics amp; Knowledge*, ser. LAK '16, Edinburgh, United Kingdom: Association for Computing Machinery, 2016, pp. 304–313, ISBN: 9781450341905. DOI: 10.1145/2883851.2883876. available from: https://doi.org/10.1145/2883851.2883876.

[46] *Lenskit*. available from: https://lkpy.readthedocs.io/en/stable/index.html.

[47] M. D. Ekstrand, "Lenskit for python: Next-generation software for recommender systems experiments," in *Proceedings of the 29th ACM International Conference on Information amp; Knowledge Management*, ser. CIKM '20, Virtual Event, Ireland: Association for Computing Machinery, 2020, pp. 2999–3006, ISBN: 9781450368599. DOI: 10.1145/3340531.3412778. available from: https://doi.org/10.1145/3340531.3412778.

[48] *S3*, 2002. available from: https://aws.amazon.com/s3/.

[49] N. Florian, G. Pauletto, and D. Duay, *Iot: L'émancipation des objets*, 2017. available from: https://aws.amazon.com/iot/.

[50] D. Rangel, *Dynamodb: Everything you need to know about amazon web service's nosql database*, 2015. available from: https://aws.amazon.com/dynamodb/.

[51] N. Florian, G. Pauletto, and D. Duay, *Iot: L'émancipation des objets*, 2017. available from: https://aws.amazon.com/iot/.

[52] G. Belle, Statistical rules of thumb, *Wiley series in probability and statistics* [online], vol. 699 Jan. 2008, Jan. 2008. DOI: 10.1002/9780470377963.

[53] P. B. Thorat, R. M. Goudar, and S. Barve, Survey on collaborative filtering, content-based filtering and hybrid recommendation system, *International Journal of Computer Applications*, vol. 110, no. 4 2015, pp. 31–36, 2015.

[54] K. Bhareti, S. Perera, S. Jamal, M. H. Pallege, V. Akash, and S. Wiieweera, "A literature review of recommendation systems," in *2020 IEEE International Conference for Innovation in Technology (INOCON)*, IEEE, 2020, pp. 1–7.

[55] P. Chopra, *Learn the a/b testing secret revealed by appsumo: Vwo*, Feb. 2022. available from: https://vwo.com/blog/a-b-testing-tips/.

[56] M. A. Farakh, *Most of your ab-tests will fail*, Jun. 2013. available from: https://www.jitbit.com/news/185-most-of-your-ab-tests-will-fail/.

[57] Y. J. N. T. University, Y. Ji, N. T. University, *et al.*, *A re-visit of the popularity baseline in recommender systems: Proceedings of the 43rd international acm sigir conference on research and development in information retrieval*, Jul. 2020. available from: https://dl.acm.org/doi/pdf/10.1145/3397271.3401233.

[58] N. Z. Tsaku and S. Kosaraju, "Boosting recommendation systems through an offline machine learning evaluation approach," in *Proceedings of the 2019 ACM Southeast Conference*, ser. ACM SE '19, Kennesaw, GA, USA: Association for Computing Machinery, 2019, pp. 182–185, ISBN: 9781450362511. DOI: 10.1145/3299815.3314454. available from: https://doi.org/10.1145/3299815.3314454.

[59] M. Gupta, A. Thakkar, V. Gupta, D. P. S. Rathore, *et al.*, "Movie recommender system using collaborative filtering," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, IEEE, 2020, pp. 415–420.

[60] *Website, a/b testing amp; optimization tools*. available from: https://marketingplatform.google.com/about/optimize/.

NTNU

Norwegian University of
Science and Technology