University of Glasgow | School of Computing Science

Honours Individual Project Dissertation

Explaining Recommendations for Collaborative Filtering Recommender Systems

Quentin Deligny
March 27, 2020

Abstract

With the rising popularity of recommender systems and their application in a wide spectrum of fields, the role played by explanation techniques in the context of recommendations has become more recognised. More recently, many studies have shifted their focus from optimising the design of standalone recommender systems to the conception of impactful explanation methods. This project seeks to build on existing work to provide a set of three explanation techniques alongside a collection of quantifiable metrics for evaluation. Our results indicate that two out of the three implemented explanation methods have improved user experience by enhancing effectiveness, scrutability, trust, satisfaction, transparency, persuasiveness and helpfulness. Furthermore, we have found this enhancement to be significantly different from the baseline in terms of transparency and effectiveness for explanation methods 1 and 2. On the other hand, explanation method 3 has degraded overall user experience; This corroborates previous findings that no explanation is better than poorly designed explanations. We conclude by stating that, in the context of this study, participants favoured well-structured, visual or informational explanations over less-personalised, technical explanations.

Acknowledgements

I would like to express my special thanks of gratitude to Dr. Craig Macdonald and Dr. Amir Jadidinejad for their valuable guidance and continued support throughout this project.

I would also like to thank my family and friends for their support as well as their precious insight and feedback regarding this project.

Education Use Consent

Contents

1 Introduction

The use of recommender systems (RS) has evolved over the years to reach a point where they have become omnipresent in the lives of many people. Nowadays, they are used by a multitude of large corporations such as Netflix, Spotify or Amazon to tailor their services to the needs and preferences of target users. The advantage such systems have over traditional search engines is that they allow for discovery rather than search: users are not required to explicitly state what they are searching for as the system will directly highlight items that may be of interest to them. RS rely on large information spaces in order to support the decision-making process.

1.1 The Paradox of Choice

It has been shown that having too many options to choose from may be a source of anxiety rather than happiness for some user (see graph in *Appendix E.1*). This notion is further described in *The Paradox of Choice*, Schwartz (2004). The main idea is that the impact of an additional choice on the subjective state of the consumer becomes negative past a certain point sometimes referred to as the sweet spot. RS help alleviate this impact by preselecting a set of relevant items for the consumer. Ideally, the size of the preselection will correspond to the number of items required to maximise the consumer's subjective state. In practice, the size is influenced by many other factors such as the importance of the choice, the type of item and other contextual information.

1.2 Motivation

The concepts of efficiency and effectiveness are often at the center of the attention when it comes to designing RS for large-scale businesses and companies. This is important for any business seeking to optimise their profit through the use of customised recommendations: an effective system will recommend relevant items to the customer who will in turn be more likely to purchase the recommended item. The constant search for the best-performing system often overlooks another essential aspect of RS: explanations. Herlocker et al. (2000) have shown that providing explanations may benefit user satisfaction and trust towards the system. The logical reason is that explanations provide justification for the recommendations that are being made which make users feel more involved in the process and more aware of the inner-workings of the system. This yields a better overall acceptance of the RS, and a certain sense of forgiveness for incorrect recommendations. Indeed, users are generally more likely to forgive the system if they understand the cause of the mistake. Thus, businesses should seek to provide explanations as it is a key factor in user satisfaction and retention. Theoretically, every company or individual who aims to provide effective recommendations should also implement adequate justifications. Even if the ultimate goal of the RS is not to increase engagement or maximise profit, it may be a desirable outcome to educate the user on the recommendation process by providing an explanation. The presentation (graph, chart, text) and the type (past performance, ratings of peers, opinions of experts...) of the explanation method may depend on the item that is being recommended as well as the RS that is being used. This project seeks to provide a better understanding of different explanation techniques and their impact on users in the case of collaborative filtering-based systems.

The focus is set on improving the transparency of the algorithms behind RS by providing a set of explanation techniques to remedy the fact that recommendations are often perceived as the result of an obscure process by many users.

1.3 Goals of the Project

The aim of this project is to increase the overall effectiveness of recommendations made by a system through explanation methods rather than using a more common approach which usually seeks to directly improve the performance of the recommendation algorithm. There are benefits to improving explanation techniques that are frequently overlooked in the design of RS: they are often implemented as a black-box which is given an input and outputs a recommendation without any details on the intermediate steps. This project will implement different visual explanation techniques to accompany recommendations made by a collaborative filtering (CF) based system. The aim of these explanations is primarily to deepen the understanding that the user has of the recommendation process. We will also devise a set of commonly-used metrics which will be assessed through participant feedback for evaluation purposes.

1.4 Summary

This chapter's role was to introduce the various motivations and goals behind this project in order to explicit the importance of providing high-quality explanation methods to improve the RS as a whole. This paper will respect the following structure:

- *Chapter 2* will review existing work in the field of RS and explanation methods while explaining some of the key concepts behind this project.
- *Chapter 3* will focus on defining the requirements as well as their potential evolution over the course of this project.
- *Chapter 4* will highlight the various design decisions which impacted the different components of this project; from the dataset to the model, ending with the user interfaces.
- *Chapter 5* will revisit the previously established design choices in order to further detail the technical aspects of their implementation.
- *Chapter 6* will analyse the findings of the conducted user study through the list of predefined metrics. The analysis will be supported by a statistical test to determine the significance of our results.
- *Chapter 7* will provide a summary of the entire project focusing on the achieved results. This chapter will also outline potential avenues for future work.

2 Background and Related Work

This section will discuss some of the key underlying concepts of RS. Two common types of RS will be briefly described. Explanation methods as a whole will also be examined in this chapter as they have an impact on user satisfaction.

2.1 Recommender Systems

Recommender Systems come in many shapes and forms. However, most agree on four main categories of RS: content-based, CF-based, knowledge-based and hybrid RS. The first two are more relevant to this project and are therefore further detailed below. Although they function differently, the end goal is generally the same: to provide effective and efficient recommendations (based on user ratings or preferences) in order to maximise user satisfaction.

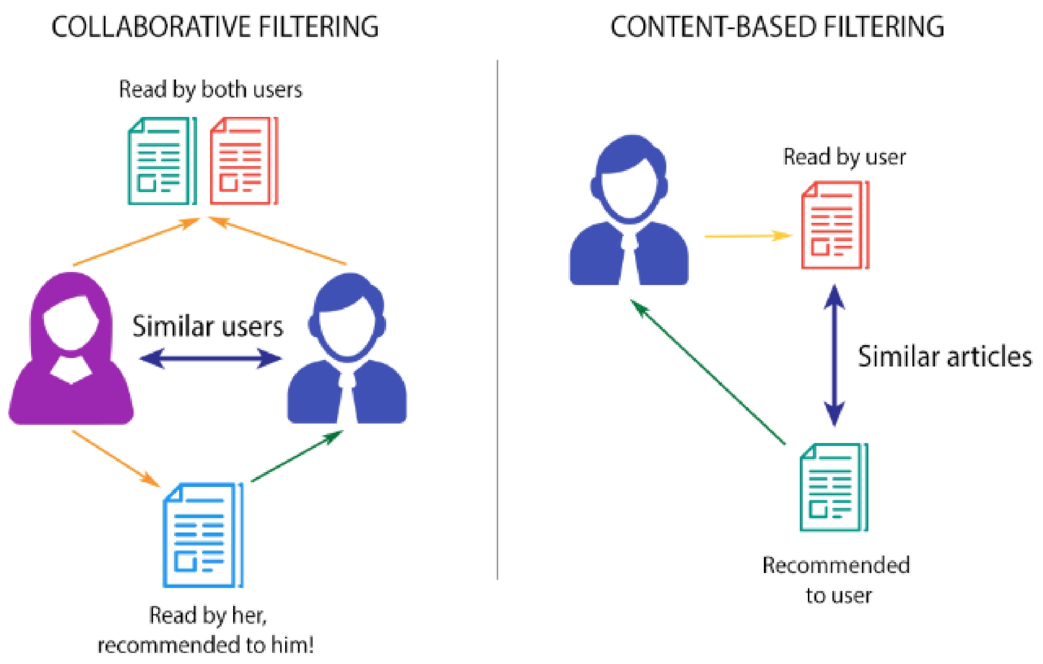


Figure 2.1: A comparison of CF-based RS and content-based RS. Figure from ***Towards Data Science****.*

2.1.1 Collaborative-based approach

Collaborative-based RS are one of the most widely used RS. One of the reasons behind this success is that this approach has many different applications. The main idea behind this type of RS is to make recommendations for a specific user based on the preferences of similar users rather than examining similar items.

In some cases, the system may compute item similarity instead of focusing on users. Regardless of the specific case, the system relies on a similarity function to make predictions. The most commonly used similarity functions are briefly introduced below.

The Pearson correlation similarity between two users:

$$\text{simil}(x, y) = \frac{\sum_{i \in I_{xy}} \left(r_{x,i} - \overline{r_x}\right)\left(r_{y,i} - \overline{r_y}\right)}{\sqrt{\sum_{i \in I_{xy}} \left(r_{x,i} - \overline{r_x}\right)^2}\sqrt{\sum_{i \in I_{xy}} \left(r_{y,i} - \overline{r_y}\right)^2}} \tag{2.1}$$

where I_{xy} is the item set rated by both user x and y.

The cosine-based approach is another method to compute the similarity between two users. It is defined as:

$$\text{simil}(x, y) = \cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{||\vec{x}|| \times ||\vec{y}||} = \frac{\sum_{i \in I_{xy}} r_{x,i} r_{y,i}}{\sqrt{\sum_{i \in I_x} r_{x,i}^2}\sqrt{\sum_{i \in I_y} r_{y,i}^2}} \tag{2.2}$$

This approach is thoroughly explored by Hameed et al. (2012) in their work on CF-based RS. The article discusses a framework for categorising and evaluating collaborative based RS. Aside from providing a very precise categorisation method, Hameed et al. (2012) also identify a number of interesting challenges to consider when working with this kind of RS:

- **Data sparsity** is one of the greatest challenges for CF-based systems. Generally, the greater the m by n user/item matrix, the sparser it becomes: many users will not interact with the majority of items in the database. The sparsity of user/item interactions makes it more challenging for the RS to provide effective recommendations. Hameed et al. (2012) further expand on this issue by using the example of "cold start" situations; These are situations in which a new user (or item) has been added to the database. As a result, the user has little to no interaction with other items in the database (see *Appendix 5.1*).
- **Increase in the number of users and items** may also lead to issues regarding scalability in cases where the increase is significant.
- **Similar items** are not always categorised as such by the system.
- **Graysheep** is a term used to describe users which "will not consistently agree or disagree with any group of people", Hameed et al. (2012). The system is likely to struggle to provide accurate recommendations for this kind of user.
- **Blacksheep** characterises users who are against the concept of recommendations. Users in this group constitute a known and accepted failure case for CF-based systems.
- **False recommendations** are often used as an attempt to discredit the competition in commercial applications of RS. For example, a product seller may be tempted to leave negative reviews to other sellers of the same product to increase his personal profit. RS with a commercial application need to account for false interactions between users and items to avoid a decrease in the effectiveness of their recommendations.
- **Privacy**: this is the last concern listed by Hameed et al. (2012). The challenge resides in the fact that users have become increasingly concerned with the way their personal information is treated. This implies that any modern RS needs to guarantee certain levels of safety and privacy towards users.

A collaborative approach for research paper Recommender Systems by Haruna et al. (2017) provides an implementation of this type of system, applied in a specific field. This work is interesting as it combines the advantages of a standard CF approach with available contextual metadata in order to improve their research papers recommendations. This is one of the many examples of the applications of CF-based RS.

2.1.2 Content-based approach

Content-based filtering is another common method used by RS to provide recommendations to users of the system. Antaris et al. (2008) provide an overview of techniques used by this type of system by comparing their advantages and drawbacks. Specifically, Antaris et al. (2008) state that the "continuously expanding volume and increasing complexity of information on the Web has therefore made such systems essential tools for users in a variety of applications". Similar claims are found across a range of different research papers.

This is the case of Basu et al. (1998), who apply content-based filtering methods to social recommendations, pointing out that they often fail to consider all the information, such as explicit user reviews. This paper identifies a new approach (inductive learning) that takes into account more information regarding the users in order to compare the effectiveness of this system with the old design. The proposed method achieves better precision and recall than the original social filtering approach: these two measures are often a trade-off between the inclusiveness and selectiveness of the model. Basu et al. (1998) do not claim to have invented the content-based filtering approach, which was first detailed by Lang (1995) as well as Pazzani, Muramatsu, and Billsus (1996).

Content-based RS rely on precise descriptions of all users and items contained in the database. This description often takes the form of a set of tags that characterise the item or the user; For example, possible tags used by a system where the recommended items are movies could be the movie's date, title, producer, actors, plot line or genres. This type of system performs generally better when there is a large collection of information attributed to each item.

Content-based RS attempt to solve the recommendation problem as a classification task: the goal is to recommend items which are similar to other items liked by a specific user in terms of content (or tags). Therefore, the possible features of the items need to be abstracted using an item representation algorithm.
A common example of such kind of algorithm is the *Term Frequency Inverse Document Frequency* (*tf-idf*) representation. The exact formulae is provided in *Appendix E.1*.

This vector representation of item features is used to create a content-based profile for each user. Weights associated with each user profile represent the importance of each feature to the user. There are many techniques to compute these weights from user/item interactions such as decision trees, Bayesian Classifiers, cluster analysis or neural networks.

Much like CF, the content-based filtering approach faces many challenges. The degree of variety in recommendations made to a specific user (or even across users) is a very important concern. A strong correlation has been observed between this degree of variety and the satisfaction of users (understandably). As such, an indirect goal of RS is to ensure that the set of items they recommend are diversified. This implies that the system should be able to learn user preferences from users' actions regarding one content source and use them across other content types. This can prove challenging for pure content-based RS. For this reason, hybrid systems tend to perform better regarding the degree of variety in their recommendations.

2.2 Explanations Methods in Recommendation

Explanation methods are often overlooked when trying to improve the performance of a RS; Providing good explanations is as important as implementing an effective RS for the users. The recent surge in the popularity of RS has brought more attention to the ways in which they are designed: the role played by explanations has significantly increased.

2.2.1 Concepts and metrics

The new interest of companies in the development of high-quality explanation methods has lead to the elaboration of various metrics and concepts. Although there is no norm or set process to define how explanation methods should be evaluated, a set of metrics are frequently used for this purpose. These new explanation evaluation methods provide a good overview of the performance of a given explanation in recommendations.

As discussed in a lecture on *Recommender Systems Explanations* by Macdonald (2020), transparency has become an important aspect of a system's design: compared to the traditional black-box implementation, a transparent system will facilitate the user's decision process (justification, trust, satisfaction). This lecture highlights how providing explanations aims to "influence the user's decision" (persuasiveness), "help the user make better decisions" (effectiveness) as well as "help the user make faster decisions" (efficiency). These concepts are fundamental for RS with business applications. Furthermore, providing explanations for a set of recommendations may educate the user and lead him towards a "sense of forgiveness" Daher et al. (2018) for the system (which stems from a better understanding of the system's recommendation process). Being able to understand why or how the system has provided a bad recommendation is beneficial for both the user and the business involved. This project will focus on reusing these metrics in a quantifiable way in an attempt to evaluate different explanation methods.

Bilgic and Mooney (2005) introduce a new pair of interesting concepts in *Explaining Recommendations: Satisfaction vs. Promotion*. Their work highlights the nuance between the two metrics, pointing out that they are often overlooked or confused with one another. The novel ideas found in this work is the categorisation of explanations methods in two distinct classes, based on the aim of the explanation:

- **Promotion**: this metric concerns explanations where the main objective is to convince the user to follow the provided recommendations. This is more likely to be the case in a business scenario where the seller seeks to maximise profit through his sales.
- **Satisfaction**: on the other hand, this category includes explanation methods aiming to satisfy the user. The end goal is not to push the user towards consumption, but rather to ensure that they make the best possible choice (higher satisfaction). This notion is also very important for commercial applications, as satisfied users are more likely to use the same RS again. For this reason, this metric may be more relevant for businesses aiming to build a loyal clientele, rather than encouraging a one-time purchase.

Bilgic and Mooney (2005) argue that satisfaction is more important than promotion in the design of an explanation method. They define a good explanation as "one which accurately illuminates the reasons behind a recommendation and allows users to correctly differentiate between sounds proposals and inadequately justified selections". Their claims are backed by a study comparing the effectiveness of three different explanation user interfaces. Each approach has been evaluated based on how "accurately they allow users to predict a more in-depth evaluation of a recommendation". Interestingly, their work has shown that some explanation methods are more effective at promoting recommendations (such as "neighbourhood style" explanations), while others enable a more accurate assessment of recommendations and yield a higher user satisfaction (such as "key-word style" explanations). Our project will measure promotion (or persuasiveness) rather than satisfaction as it would not be feasible to ask the user to watch all the recommended movies.

2.2.2 Practical applications

The large majority of innovations in the field or RS have a commercial goal. In itself, the main goal of a RS is to make effective item recommendations to a set of users. These items are often material or virtual products such as food, clothes, hotels, songs, movies... As a result, they aim to maximise profit by optimising the number of product sales through good recommendations.

A Review on Explanations in Recommender Systems by Daher et al. (2018) provides an evaluation of explanation methods through a specific case study. After having analysed existing work regarding explanation methods, the paper introduces its own approach through its e-learning project entitled METAL. They use this project to provide two different kinds of explanations for students regarding their academic level and external resources provided to them. Daher et al. (2018) identify key principles applied in the design of these explanation methods: simplicity, clarity, coherency and minimalism. Some of the explanation techniques implemented in this project will respect these principles, while others will not.

Similar studies have been conducted in an attempt to classify and differentiate a common set of visual and textual explanation methods. *How should I explain? A comparison of different explanation types for Recommender Systems* by Gediklia et al. (2014) is a good example of such work. The study focuses on providing users with 10 different explanation methods in order to assess their performance. The study demonstrates that "the content-based tag cloud explanations are particularly helpful to increase the user-perceived level of transparency" which, as discussed, is one of the key pillars of a good RS. Overall, this work implements a wide range of explanation methods to highlight a correlation between transparency and satisfaction. The current project will use some of the concepts described by Gediklia et al. (2014) using only three explanation methods: this will allow for an in-depth analysis of each implemented design.

2.3 Summary

This section discussed existing work in the fields of RS and explanation techniques. Existing research shows the importance of providing a benchmark for evaluating explanation techniques. Although major progress has been made in the field of RS and more recently their explanation methods, this review indicates that further work needs to be done in order to provide a systematic assessment of explanation methods. This is highly complex as it relies on many factors such as the type of system for which explanations are provided, the recommended items or the user population. This project aims to participate to this research effort by providing more insight into the evaluation of different explanation methods in the context of CF-based recommendations. More specifically, this project will implement and test 3 novel explanation techniques for CF-based RS using tools such as tSNE, Matplotlib and TkInter for visualisations. Aside from assessing new explanation techniques, this project introduces a new set of quantifiable metrics to evaluate and compare these explanations. This list of evaluation metrics is heavily inspired from Daher et al. (2018) who have introduced some of them but have not attempted to quantify these metrics during the evaluation stage.

The next chapter will define the requirements gathering process.

3 | Requirements Analysis

This chapter will focus on detailing the initial requirements which guided this project. It will also discuss the reasoning behind the requirement selection process, and examine the potential evolution of the requirements. Many requirements have been redefined and improved on a weekly basis at the start of the project. These iterative improvements allowed for better flexibility. The following requirements have been established based on existing background work. They have been designed with popular RS such as Netflix or Amazon Prime in mind to enhance user familiarity with the interfaces. Furthermore, a research paper literature survey has been conducted in order to identify the core requirements, among other objectives.

3.1 Functional Requirements

Functional requirements have been implemented following the standard MoSCoW framework to ensure adequate optimisation of tasks over time. The main concern behind the selection of functional requirements was to make sure the final product was sufficiently complex and developed to offer interesting information in the evaluation section (*Chapter 6*). Following this format, the initial requirements were chosen as follows:

1. **Must have**: one or more explanation methods to explain a set of recommendations to the user. This is the core of the project and it is therefore an essential requirement to measure the performance of explanation methods.
2. **Must have**: at least one user interface to visualise the explanation method (graphs, tables, charts...). The interface should be clear and concise while providing a personalised explanation. Although a visual explanation is not essential to obtain results for this project, it greatly enhances the complexity of the obtained results, yielding greater analysis potential.
3. **Should have**: the possibility to make predictions for unseen users by adding the relevant data to the system's existing database. This requirement has evolved along the course of the project; specific details will be provided in *Section 3.3*.
4. **Should have**: an interface designed specifically for preferences gathering. This would play a key role in the user study by allowing the system to learn about new users. More details on the elaboration of this requirement will be provided later.
5. **Could have**: a tool for movie metadata extraction. This is not entirely necessary, but would provide an improvement to the user interface by allowing for more information to be displayed concerning each item.
6. **Could have**: provide explanation techniques for CF-based systems as well as other types of RS. This would be useful to compare the performance of the chosen explanation techniques based on the type of RS (and find any potential correlations).
7. **Could have**: a set of metrics such as the novelty, relevance, serendipity and diversity of recommendations as a way to explain why a particular set of recommendations has been made to the user.
8. **Won't have**: an interface to compare the resulting pros and cons of how different types of RS explain their recommendations. Or, a comparison of how the same explanation method may perform differently based on the system's type.

3.2 Non-Functional Requirements

In this part, we discuss a broader set of requirements which may not be directly linked to a product feature. Since non-functional requirements constitute a set of general ideas or concepts they have not been structured following the same rigorous pattern as the functional requirements. The resulting requirements are listed below:

- A process, possibly using a set of defined metrics, to evaluate explanation methods. This is fundamental to assess the impact of implemented explanation methods on various factors. The limitations and alternative solutions proposed to fulfill this requirement are discussed below.
- Be accessible to any potential user in the case of a user study. By its nature, this project should be open to everyone in order to account for different types of users. This notion was included to provide grounds for a more complex analysis (by considering user factors such as age or familiarity with RS in our final analysis).
- The algorithm behind this project should be fast, maintainable, robust and possibly scalable. These are all general notions or concepts which cannot be embodied by a single feature; As such, they have been defined as non-functional requirements. Some notions (maintainability or scalability) are not fundamental to this project, as it will not be deployed or used for further testing. However, it is always good practice to keep these notions in mind when writing new code. Furthermore, a fast implementation will allow for more extensive testing in the evaluation phase.
- If a user study is performed, the main concerns should be centered around ease of use, intuitiveness and efficiency to promote a good user experience.

3.3 Identified Limitations and Evolution of Requirements

Some of the requirements listed above are the result of an iterative improvement process. Their evolution, as well as how it was affected by potential limitations and design decisions is brought to light in this section.

The main challenge in the establishment of requirements was to capture how the explanation methods would be evaluated. This was difficult as there is no conventional process typically used to assess the effectiveness of explanation methods in RS. Furthermore, it was challenging to find a proper evaluation process at this stage of the project considering it would largely depend on the chosen explanation techniques: some forms of evaluation may not be compatible will all types of explanations. In the end, the decision was to perform the final evaluation through the user study, which would assess various metrics for each explanation method (the specific metrics will be defined in *Chapter 6*). This option was chosen as it is represents a first-hand equivalent of user impressions for the given explanation. This would ensure that the results are not distorted (or too far from reality) by a poorly designed evaluation process. This option was also selected as user satisfaction is the ultimate goal of any RS; It can therefore be used as a way to directly measure the impact of an explanation method. Alternative evaluation metrics and approaches were also considered; for example, evaluating the explanations by providing a series of non-personalised explanations for each user (removing the need to gather information regarding the user's tastes), or even evaluating the explanations in a systematic fashion (removing the need for a user study all together).

This design decision had a great impact on the predefined requirements; Many were revisited and modified to fit this new purpose. For example, the chosen RS had to be able to train on a new data set (which contained ratings provided by the user in our study). This implies that the original data set had to be accessible for modification purposes. Gathering user preferences during the study also sparked the need for a dedicated user interface. This new interface has to rely on the existing database, to ensure that new users in the study would rate movies that already existed within the old database (otherwise it would have greatly handicapped the system's performance, as new users would be unlikely to rate movies found in this database). This posed further compatibility issues between the model, the database and the gathering of the user's tastes.

Some requirements were deliberately vague and redefined later as the project's steps became clearer (such as the type of explanations or number of different methods). Lastly, all features relating to the comparison of different explanation methods with the type of RS as determining factor have not been implemented. This is due to time constraints as well as compatibility issues between the model or type of system, the data set (not always possible to add new users to it, not compatible with all models), the acquiring of preferences and the type of explanation. Therefore, the same model has been used for all results: future work may compare and contrast the obtained results with different types of systems and/or explanations. The scope of the study was also limited by the size of the Movie Lens data set; testing was done on the data set with one million interactions rather than one hundred thousand. The results suggested that, although the new data set was compatible with the current implementation, it had a strong negative impact on the efficiency of the algorithm. The user study would have taken too long to complete under such circumstances. We concluded that gaining effectiveness at the cost of efficiency was not desirable for this project (further details given in *Chapter 4*).

3.4 Summary

This chapter discussed the initial selection of requirements, as well as the various decisions and limitations which affected them as the project unfolded. Some alternative options have been briefly mentioned, but the majority of the listed requirements are derived from the final design decision. The main challenges faced throughout the process of defining requirements have also been stated.

The following section will be covering the general design of the project and the reasons behind some of the choices that have been made.

4 Design

This chapter will examine the design decisions made during this project and the reasoning behind them; The technical implementation of these decisions will be detailed in the next section. The chapter is structured following the project's core organisation, from the back-end to the front-end.

4.1 Overview

The current implementation of the project focuses on providing explanations for recommendations made by a RS relying on CF. A visual representation of the project's architecture is shown in *Figure 4.1*.

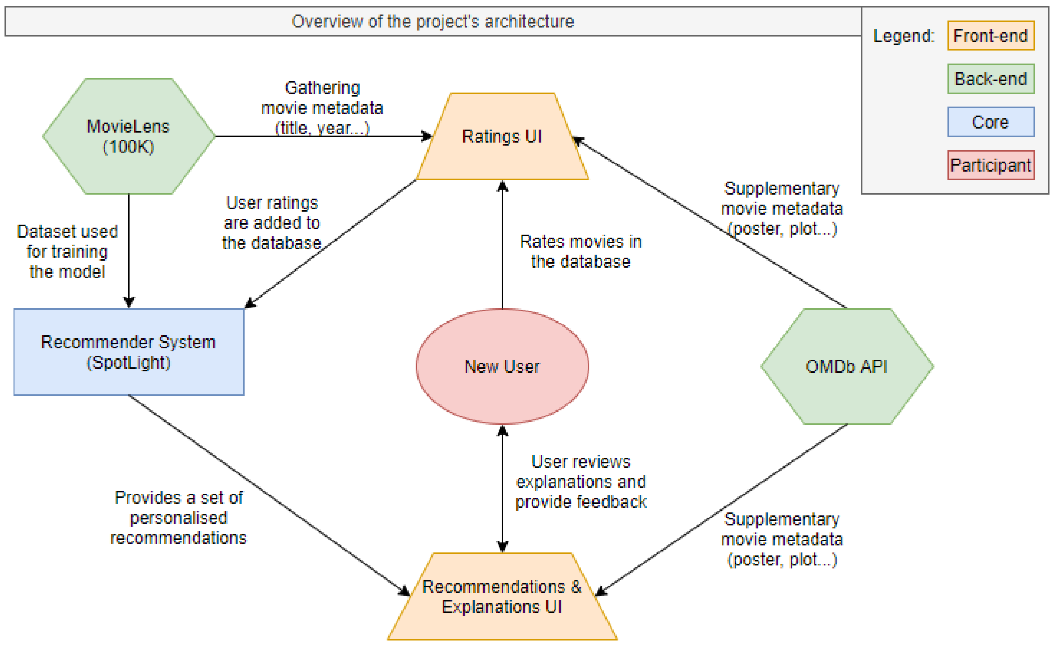

Overview of the project's architecture
Legend:
Front-end
Back-end
Core
Participant
MovieLens (100K)
Gathering movie metadata (title, year...)
Ratings UI
Dataset used for training the model
User ratings are added to the database
Rates movies in the database
Supplementary movie metadata (poster, plot...)
Recommender System (SpotLight)
New User
OMDb API
Provides a set of personalised recommendations
User reviews explanations and provide feedback
Supplementary movie metadata (poster, plot...)
Recommendations & Explanations UI

Figure 4.1: *Diagram detailing the organisation of the project by showing the relations between various components.*

The design choices behind each component of the project's structure are further discussed in the following sections.

4.2 Chosen dataset

The database is a fundamental component of any RS. High performing systems often rely on large/high-quality data to make accurate predictions. The RS implemented for this project relies on the MovieLens 100k dataset which is a freely available dataset provided by the GroupLens Research team. This dataset contains 100 000 interactions (or ratings) given by around 1000 users to approximately 1700 movies.

One of the reasons behind the selection of this popular dataset is that movie recommendation is one of the most widespread forms of recommendation; This helps the user study stage by guaranteeing a certain level of familiarity between users and the form of recommendation that is being provided. Indeed, providing unfamiliar recommendations (ie. items that users are not used to interact with) may have been detrimental to this project as the goal is to gauge the impact of explanation methods. It would have been a contradiction to provide explanations for a set of recommended items which are unfamiliar to the user (although such data might still be exploited). Once the type of recommended item had been chosen, we focused on selecting the right dataset: MovieLens has several options varying in size and date of publication. We considered using the 1M dataset instead of the smaller one as it was more recent on top of being larger. Detailed below are the various elements which came into consideration before making the final design decision:

- A larger dataset would provide a broader range of choices for users to rate as well as for the recommendation stage; However, the need for a wide movie selection was not deemed necessary for this project considering that users taking part in the study would only be required to rate a small number of movies regardless.
- Another argument in favour of the larger dataset is that it is likely to produce more accurate recommendations: the model's performance is likely to increase with the size of the dataset. Once again, this was not categorised as a significant advantage for our situation given that the project was not interested in optimising the model's performance, but rather in understanding the impact of explanation techniques regardless of whether the provided recommendations are good or simply average. Of course, the recommendations should be coherent and personalised (it would be of no use to explain random recommendations) for the explanations to function properly.
- Furthermore, some testing was done using both datasets (details provided in implementation): the model running on the larger dataset proved to be much less efficient (taking on average 7 times as long to run). This would have had a significant impact on the wait time of users during the user study. Following this trade-off, we concluded that the smaller dataset would better suit the goal of this project.
- Another factor which has not been thoroughly taken into account at the time of this decision was the publication date of the dataset. Indeed, the majority of users who have taken part in this study are between 18 and 25 years old. As a result, some of them struggled to find and rate 10 movies they had seen within the provided database. This issue may have been slightly alleviated by choosing the 1M dataset (published in 2003 rather than 1998).
- Lastly, we considered various characteristics of the dataset following the categorisation discussed by Adomavicius and Zhang (2012) in their work *Impact of data characteristics on recommender system performance*. Among the six characteristics defined and evaluated in this article, the user-item ratio is the most relevant for our situation. The measured item ratio for the MovieLens 100K, 1M and 10M datasets are respectively 0.588, 1,5 and 7,2. The findings made by Adomavicius and Zhang (2012) indicate that "higher user-item ratio leads to larger recommendation errors for item-based CF techniques". While this margin was not significant in the case of user-based collaborative filtering techniques, we opted to minimise this risk by choosing the dataset with the lowest user-item ratio.

These arguments have played a role in the final decision regarding the chosen database for the implementation of this project.

4.3 The type of Recommender System

4.3.1 Chosen Design

One of the main components of this project is the RS; Many design decisions came into play during the selection of this system considering the wide variety of models and implementations available. The implementation of our system is conducted through Spotlight, a tool tailored for "rapid exploration and prototyping of new recommender models" developed by Kula (2017). This suits the purpose of the project as the main interest lies in the explanation techniques rather than the implementation of the RS. Furthermore, Spotlight is compatible with explicit data such as MovieLens ratings.

4.3.2 Benefits of Collaborative Filtering

User-based CF is one of the many ways by which RS can make predictions. Although hybrid systems generally achieve a better performance in terms of the effectiveness of the provided recommendations, they tend to be more complex to implement. Each type of system has advantages and disadvantages which should be accounted for before making the final decision. This section will examine the benefits of collaborative filtering over other methods. These benefits contributed to the final model decision during the early stages of this project. The advantages are resumed below in a non-exhaustive list:

- The first advantage tied to collaborative filtering based systems is that they promote serendipity. Serendipity is often described as accidental yet beneficial discoveries; In other words, providing recommendations for a user which he may not expect, yet be pleasantly surprised. Kotkov et al. (2016) have accurately stated that "novelty and unexpectedness require serendipitous items to be relatively unpopular and significantly different from a user's profile". They conclude that, overall, serendipity greatly enhances the quality of the provided recommendations. This is beneficial in the context of explanations as unexpected or novel items are generally the ones which require the most explanations. Indeed, it is more challenging and interesting to provide explanations for a novel and unexpected item over a highly rated popular item. Therefore, we concluded that implementing a RS which promoted this notion would be beneficial for this project.
- Secondly, this type of system does not rely on contextual features to function properly. It may be trained solely using the feedback matrix containing user ratings, which is a plus considering the chosen dataset.
- Similarly, the model requires no prior domain knowledge to perform the recommendation task. Although providing information on the relation of concepts within a certain domain may enhance the system's performance, it was not deemed to be directly beneficial to the explanation process.

On the other hand, the chosen model also contains some disadvantages when compared with other types of RS. Most of these downsides did not have a significant impact on our situation; Those who did, such as the cold start problem, are further reviewed in the following chapter. More specifically, *Chapter 5* will indicate how these issues were addressed to mitigate their impact.

4.4 Gathering User Ratings and Providing Recommendations

The user interfaces constitute an essential element of this project. They are utilised to gather movie ratings from the participant, to display recommendations from the system and also to provide adequate explanations for various sets of recommendations. More details regarding the partitioning of these sets will be given in *Chapter 6*.

4.4.1 Wireframes and Consistency

The design process regarding the user interfaces relied on wireframes built during the early stages of the project. The main goal behind the design of these initial wireframes was to ensure a sense of familiarity and intuitiveness from the user.

RS have become omnipresent in recent years. As a result, many people are already familiar with the designs of the most popular RS such as Amazon Prime, Netflix or Hulu. The aim was to use this existing advantage in the design of the interfaces to minimise the amount of effort required from the user having to learn an entirely new interface.

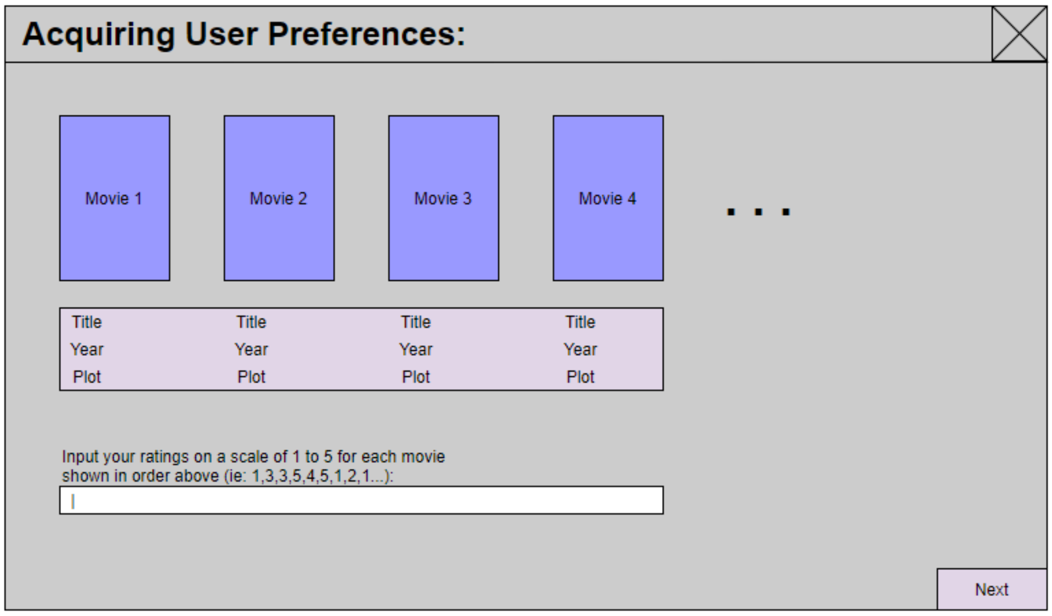

Acquiring User Preferences:
Movie 1
Movie 2
Movie 3
Movie 4
. . .
Title
Year
Plot
Title
Year
Plot
Title
Year
Plot
Title
Year
Plot
Input your ratings on a scale of 1 to 5 for each movie shown in order above (ie: 1,3,3,5,4,5,1,2,1...):
Next

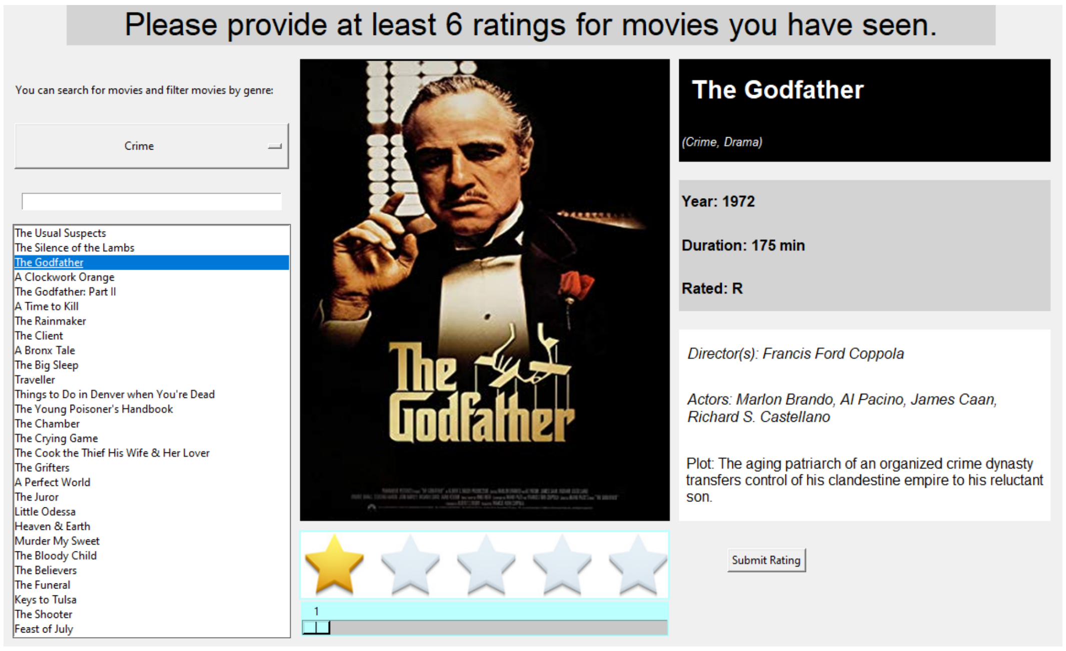
Figure 4.2: *First wireframe representing the user interface designed to gather participant ratings for movies within our database.*

Figure 4.2 represents a wireframe for the interface dedicated to user preferences gathering. This design focused on mimicking popular streaming/movie recommendation platforms: the movies are presented in a list, with additional metadata concerning the movie's title, plot or year of publication. Even though participants will be required to rate movies they have already seen, it is important to provide some form of visual data such as movie posters to enhance user engagement with the interface. Lastly, a field is provided to allow participants to enter a rating for the selected movie.

As it was designed during the early stages of the project, this wireframe lacks various elements which have been added to the actual user interface at a date. For example, it did not specify how the displayed list of movies would be selected for each user (since they had to be movies that the user had seen). The final design of the user interfaces will be reviewed in the following sections.

4.4.2 Acquiring Preferences

This section will detail the decisions behind the design of the user interface aiming to gather participant ratings. The final interface is shown below:


Please provide at least 6 ratings for movies you have seen.

You can search for movies and filter movies by genre:

Crime

The Usual Suspects
The Silence of the Lambs
The Godfather
A Clockwork Orange
The Godfather: Part II
A Time to Kill
The Rainmaker
The Client
A Bronx Tale
The Big Sleep
Traveller
Things to Do in Denver when You're Dead
The Young Poisoner's Handbook
The Chamber
The Crying Game
The Cook the Thief His Wife & Her Lover
The Grifters
A Perfect World
The Juror
Little Odessa
Heaven & Earth
Murder My Sweet
The Bloody Child
The Believers
The Funeral
Keys to Tulsa
The Shooter
Feast of July

The Godfather

(Crime, Drama)

Year: 1972

Duration: 175 min

Rated: R

Director(s): Francis Ford Coppola

Actors: Marlon Brando, Al Pacino, James Caan, Richard S. Castellano

Plot: The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.

1

Submit Rating

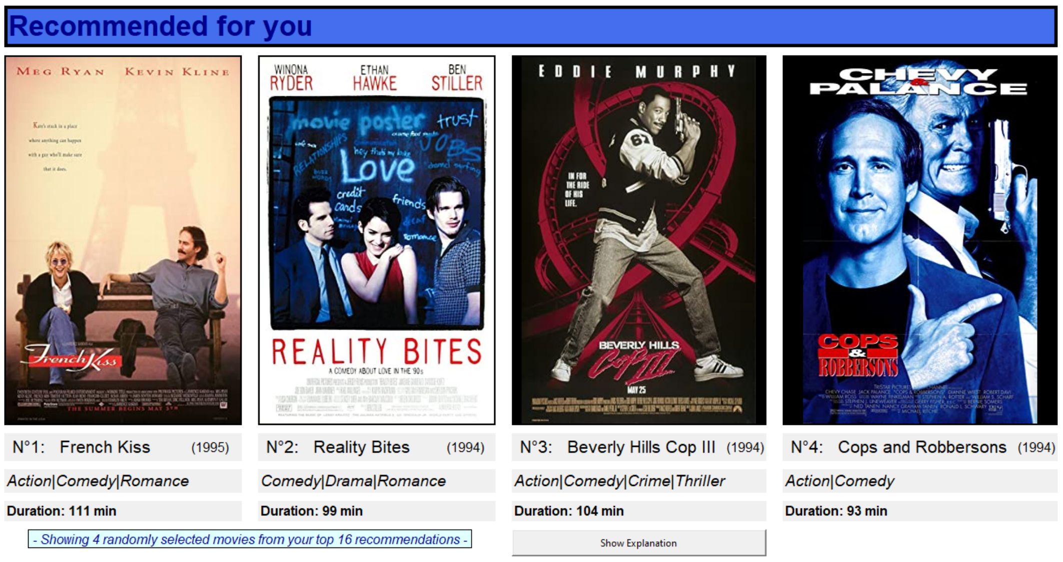
Figure 4.3: This is the first interface presented to the user during the user study. It serves as a mean to gather participant ratings for various movies.

The interface shown in *Figure 4.3* is designed to gather user input for movies they have seen. The rating scale has evolved since the original wireframe design, from a simple entry field to a more popular 5-star rating. This standard was adopted to facilitate the ease of use of this interface as it is already widely used by popular RS. The user may provide ratings for movies after he has selected them using a slider. The original goal was to allow the user to rate items by clicking on a star directly, but this was not implemented due to technical issues which will be discussed in the following chapter. Additionally, the user may search for specific movies within the database as well as filter the movies by genre. Second to enhancing user familiarity, the main goal of this interface was to make it as easy as possible for users to find and rate movies they had seen.

Finally, some further information is provided for each selected movie on the right-hand side column of the interface.

4.4.3 Recommendations User Interface

Once the user's ratings have been incorporated into the existing database and used to make personalised recommendations, the participant is shown the following figure:


Recommended for you

N°1: French Kiss (1995)	N°2: Reality Bites (1994)	N°3: Beverly Hills Cop III (1994)	N°4: Cops and Robbersons (1994)
Action|Comedy|Romance	*Comedy|Drama|Romance*	*Action|Comedy|Crime|Thriller*	*Action|Comedy*
Duration: 111 min	**Duration: 99 min**	**Duration: 104 min**	**Duration: 93 min**

- Showing 4 randomly selected movies from your top 16 recommendations -

Figure 4.4: *Figure showing the basic user interface used to display movie recommendations made for a particular user.*

Recommendations are presented to the user by sets of 4 items following the format shown in *Figure 4.4*. Although the notion of familiarity with the interface is not as important considering that the user will not be required to interact extensively with this display, the layout is inspired by popular movie streaming platforms. The predominant concern of this layout is clarity and simplicity. Some metadata is provided to help the user decide how likely he may be to follow these recommendations (which will be measured in the evaluation).

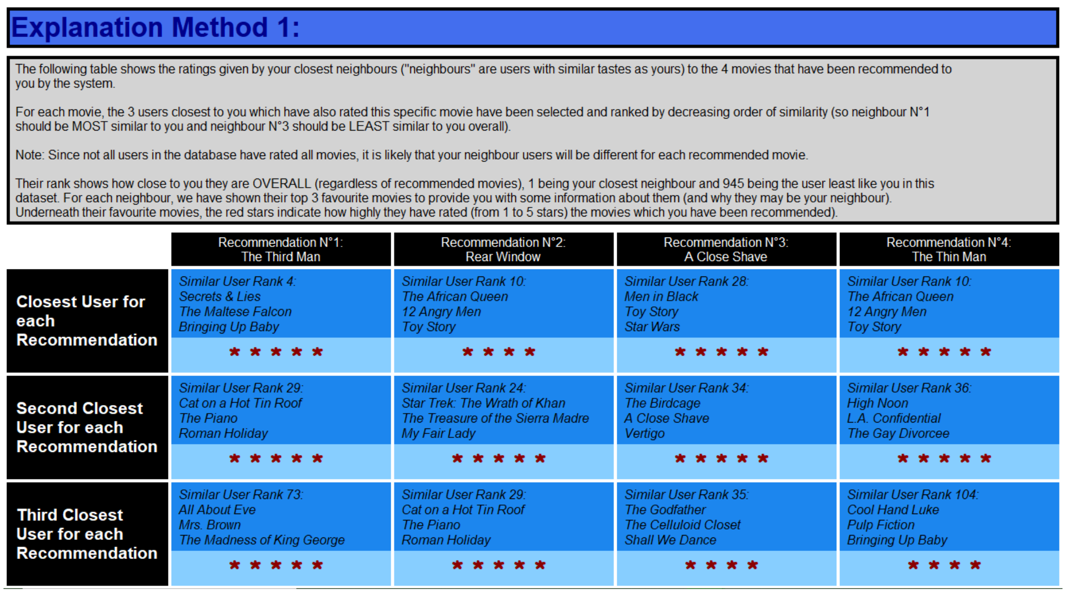
4.5 Providing Explanations

This section will provide visual representations of the finalised explanation designs while detailing the process behind the development of said design. The following three interfaces have been chosen among of wide spectrum of possibilities to accompany recommendations made by the system. Many other graphical displays have been considered during this stage, such as histograms, radar charts, diagrams or tag clouds. Ultimately, the designs have been chosen for their variety: while some are more compact, others are more visual or more scientific.

The explanation methods were presented in randomised order for each participant. Each explanation interface is accompanied by a textual explanation providing guidance on how to read the interface.

4.5.1 First explanation method: Table

This section concerns the first explanation method, which is the most compact of the three. It is more textual than the other two and offers a substantial amount of information to facilitate neighbour identification (this is the second stage of collaborative explanations as defined in *Recommender Systems Explanations* by Macdonald (2020)). As a result, participants are likely to spend more time on this interface compared to the other explanations.



Explanation Method 1:

The following table shows the ratings given by your closest neighbours ("neighbours" are users with similar tastes as yours) to the 4 movies that have been recommended to you by the system.

For each movie, the 3 users closest to you which have also rated this specific movie have been selected and ranked by decreasing order of similarity (so neighbour N°1 should be MOST similar to you and neighbour N°3 should be LEAST similar to you overall).

Note: Since not all users in the database have rated all movies, it is likely that your neighbour users will be different for each recommended movie.

Their rank shows how close to you they are OVERALL (regardless of recommended movies), 1 being your closest neighbour and 945 being the user least like you in this dataset. For each neighbour, we have shown their top 3 favourite movies to provide you with some information about them (and why they may be your neighbour). Underneath their favourite movies, the red stars indicate how highly they have rated (from 1 to 5 stars) the movies which you have been recommended).

	Recommendation N°1: The Third Man	Recommendation N°2: Rear Window	Recommendation N°3: A Close Shave	Recommendation N°4: The Thin Man
Closest User for each Recommendation	*Similar User Rank 4: Secrets & Lies The Maltese Falcon Bringing Up Baby*	*Similar User Rank 10: The African Queen 12 Angry Men Toy Story*	*Similar User Rank 28: Men in Black Toy Story Star Wars*	*Similar User Rank 10: The African Queen 12 Angry Men Toy Story*
	★ ★ ★ ★ ★	★ ★ ★ ★	★ ★ ★ ★ ★	★ ★ ★ ★ ★
Second Closest User for each Recommendation	*Similar User Rank 29: Cat on a Hot Tin Roof The Piano Roman Holiday*	*Similar User Rank 24: Star Trek: The Wrath of Khan The Treasure of the Sierra Madre My Fair Lady*	*Similar User Rank 34: The Birdcage A Close Shave Vertigo*	*Similar User Rank 36: High Noon L.A. Confidential The Gay Divorcee*
	★ ★ ★ ★ ★	★ ★ ★ ★ ★	★ ★ ★ ★ ★	★ ★ ★ ★ ★
Third Closest User for each Recommendation	*Similar User Rank 73: All About Eve Mrs. Brown The Madness of King George*	*Similar User Rank 29: Cat on a Hot Tin Roof The Piano Roman Holiday*	*Similar User Rank 35: The Godfather The Celluloid Closet Shall We Dance*	*Similar User Rank 104: Cool Hand Luke Pulp Fiction Bringing Up Baby*
	★ ★ ★ ★ ★	★ ★ ★ ★ ★	★ ★ ★ ★	★ ★ ★ ★

Figure 4.5: *First explanation technique: table showing the user how his closest neighbours have rated his recommended movies. The table also provides some information regarding each neighbour.*

Figure 4.5 constitutes the first explanation method, which takes the form of a table. Each column corresponds to a movie that has been recommended to the participant. Each row contains the participant's top three closest neighbours (from top to bottom) which have also rated the movie in the column. The neighbour's rating for the recommended movie corresponds to the red stars displayed in each box. Additionally, further information is given to the participant regarding each neighbour by providing the neighbour's top three favourite movies just under his rank. This additional information was implemented to allow to user to identify and connect with his neighbours (which would have been more difficult given only the neighbour's closeness rank).

4.5.2 Second explanation method: Interactive Graph

This representation is the most visual explanation technique implemented in this project. It features a graph depicting the similarity between the participant and every other movie in the data set. The distance between each point is an indicator of this similarity (where closer points imply they are more similar).

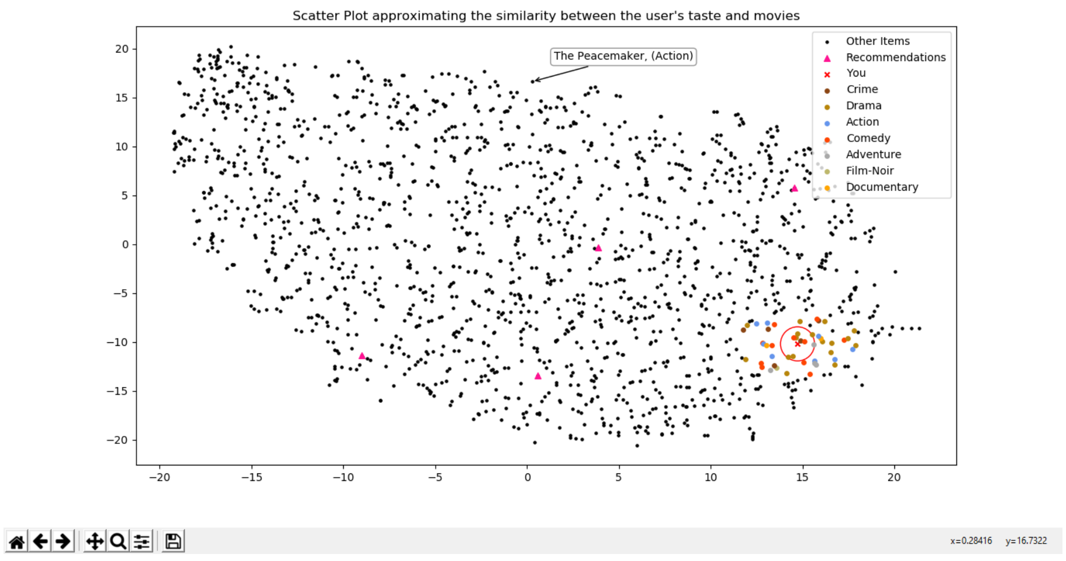


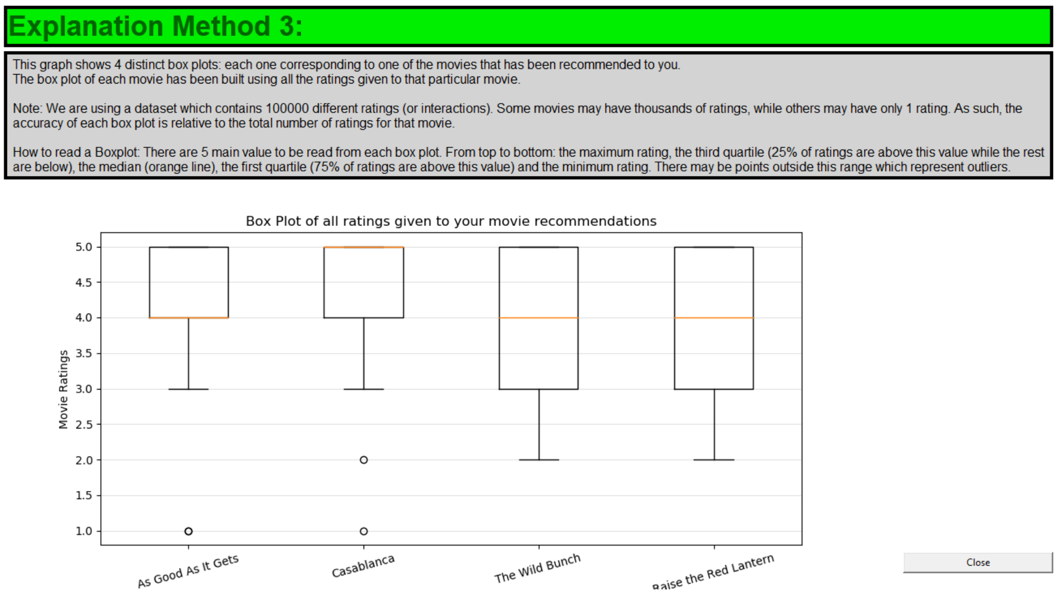
Figure 4.6: *Second explanation technique: scatter plot providing a visualisation of the similarity between the user and other movies in the dataset.*

As mentioned, *Figure 4.6* is a scatter plot which maps all the movies contained in the dataset alongside the user. Additionally, the top 50 closest movies to the user are coloured by genre. In an attempt to reduce visual clutter, the remaining movies have not been coloured by genre. However, the user may hover over each point to display the movie's title and genre. Furthermore, the participant has the possibility to interact with the graph. These interactions include resizing the axes (using a set of corresponding sliders), moving around the graph and zooming in/out on specific regions. Lastly, the triangles represented the movies that had been recommended to the user. This explanation method is different from the other two in the sense that it uses content (movie genres) in an attempt to explain recommendations made by a CF-based system. This is one of the reasons why the recommended movies did were not necessarily the closest items on the graph. If it is well received among users, it could hint that the type of system and the type of information shown in the explanation do not necessarily need to match.

It is interesting to point out that, although rather simplistic, this visualisation caused some confusion among some users (despite the textual/oral explanation given during the study). The source of this confusion is the distance observed between the user and the recommended movies: in many cases, the recommended movies were not the closest points to the user. This is a potential design flaw which was not accounted for when the explanation was conceived; As discussed, this explanation relies on the items' content (the movies' genres) and may have been better suited to explain recommendations made by a content-based system rather than a CF-based one. Nevertheless, the details at the root of this misunderstanding and their impact will be further discussed in *Chapter 6*, during the evaluation of the explanation method. The reasons behind the observed distance will be evoked in *Chapter 5* regarding the implementation.

4.5.3 Third explanation method: Box Plot

The final explanation takes the form of a box plot, where each recommended movie is represented by a distinct box. To some extent, this is the most scientific explanation method implemented for this project. The fact that users without a scientific background may need more explanations alongside these box plots was accounted for in the design decision (and additional textual explanations on how to read a box plot were provided).



Explanation Method 3:

This graph shows 4 distinct box plots: each one corresponding to one of the movies that has been recommended to you.
The box plot of each movie has been built using all the ratings given to that particular movie.

Note: We are using a dataset which contains 100000 different ratings (or interactions). Some movies may have thousands of ratings, while others may have only 1 rating. As such, the accuracy of each box plot is relative to the total number of ratings for that movie.

How to read a Boxplot: There are 5 main value to be read from each box plot. From top to bottom: the maximum rating, the third quartile (25% of ratings are above this value while the rest are below), the median (orange line), the first quartile (75% of ratings are above this value) and the minimum rating. There may be points outside this range which represent outliers.

Figure 4.7: *Third explanation technique: a box plot illustrating recommended movies. All the ratings given to a specific movie have been taken into account to produce the resulting box plots.*

The box plots presented in *Figure 4.7* are used to give the user a general idea of how well the movie has been rated by other users in the database. In this sense, this is the least personalised explanation of the three implemented methods (as all ratings are taken into account for each specific movie), although the recommendations remain personalised. The idea behind this design was to enable the participant to identify the overall distribution of ratings for each of the recommended movies: by doing so, the user would be able to tell if a movie had been recommended to him because it was very popular in general (this would be the case of *Casablanca* in *Figure 4.7*) or because it was more popular among neighbours. If a movie's ratings are more nuanced (such as *The Wild Bunch* in *Figure 4.7*), the underlying implications are that it is popular among the user's neighbours. This may be a good estimation of the previously mentioned notion of serendipity, as it would indicate that the RS is not simply recommending popular items.

4.6 Summary

To summarise, this chapter provided an in-depth reasoning behind the design choices made throughout this project; From the overall architecture of the project, to the dataset, the type of model and finally the various user interfaces implemented in order to assess the impact of explanation techniques.

In the next chapter, we describe the technicalities behind the implementation of the presented functionalities.

5 Implementation

This chapter covers the technical implementation and the associated challenges faced throughout this project. The tools and technologies used for this project will be detailed in their respective subsection. The core of this project has been developed in Python 3.7.

5.1 The Data

The data used in this project can be categorised in two distinct classes based on its purpose. The first and main use of this data is to provide a training and testing set for the RS. The second utilisation of external data stems from the need to provide movie metadata in all implemented user interfaces. Depending on the general purpose of this data, a different tool has been used to provide it. As mentioned in *Chapter 4*, the MovieLens 100K dataset has been selected to provide the training data. On the other hand, metadata was directly queried through the OMDb API.

5.1.1 Training and Testing data

This section will provide an overview of the data manipulation required for training and testing the model. Potential issues and limitations will also be briefly mentioned.

This type of data originated from MovieLens. However, the data format was not always consistent; MovieLens' latest dataset (which was not used for testing purposes but not selected in the end) contained *.csv* files while other datasets such as the 100K dataset contained non-standard file types. To ensure compatibility with the model's training and testing input formats, all the data was first converted to a text file (for future use). Distinct files were created for unique movie identifiers, titles and genres.

We encountered some issues during this step due to the irregular format of rows found in the MovieLens *.csv* files. The expected format of each row in the file is *ID, Title (Year),Genre(s)*. However, many rows were not formatted properly which complicated the file parsing process. The code shown in *Appendix A.1* is a simple example illustrating how the data was parsed to account for such irregularities. Aside from certain rows spanning multiple lines in the *.csv* files, other types of irregularities where encountered and specifically addressed:

- Many titles had the starting article at the end rather than at the start, for example *City of the Lost Children, The* rather than *The City of the Lost Children.* This occurred for English titles as well as titles in any other language; It was challenging to detect such occurrences given their inconsistency. The final solution was to implement an exhaustive case by case function checking if the title ended with a predefined list of articles. If it was the case, the article would be placed at the start of the title. A potential issue with this implementation is that some titles may truly end with an article (although very uncommon as it is not grammatically correct).
- Another special irregularity which had to be accounted for was tied to some titles having their translated equivalent in another language, such as *Postman, the (Postino, Il).* Much like the previous case, an easy fix to this situation would be to remove any text in brackets

following the title (provided it is not the year) but the drawback of this method would be that it risks removing potential parts of the same title (if the title contains brackets).

- Some titles contained special characters such as quotation marks or non-alphanumerical characters which should not be part of the original title.

The fact the the database was relatively small worked in our favour as the drawbacks of implementing the previously mentioned solutions to correct format issues were close to non-existent. Had the database been very large, some of the proposed solutions would likely have affected certain movie titles in an unwanted way.

The correct parsing of row information was essential to the project. Indeed, the MovieLens data files do not provide a unique identifier which can be used to cross-reference a movie from various datasets on the internet (such as a unique IMDb ID). The ID given to each movie within the MovieLens dataset is only relevant for tracking movies within this particular dataset. Being able to cross-reference movies between datasets is critical in order to fetch metadata regarding a specific movie from another source (such as the OMDb API).

Therefore, the only option was to rely on the movie's title to fetch metadata from sources other than MovieLens. This was not ideal as many movies share the same title; our best option to guarantee uniqueness given the MovieLens datasets was to check the correctness of other metadata when fetching a movie by its title. For example, there are at least two movies entitled *Anna* (one was published in 1987 and the other more recently, in 2019). In order to fetch the correct movie metadata (such as poster and plot) from OMDb (given that MovieLens did not provide a unique identifier), further checks were performed to ensure that the genres and year of publication matched between the MovieLens and the OMDb dataset.

5.1.2 Metadata

This section concerns the fetching and handling of data destined to provide additional information for the user. This type of data will not be used to train and test the model. As previously discussed, this type of data was fetched directly from OMDb. This was necessary considering that the MovieLens dataset did not have information regarding the movie's plot, poster, actor or director(s) which was required in the user interfaces.

```python
# Fetch movie metadata from the OMDb website
def get_metadata(movieTitles, details, wipeFile):
   """
   movieTitles: Array of formatted movie titles (no dates or genres).
   wipeFile: if True, will empty the contents of the metadata folder.
   """

   url = 'http://www.omdbapi.com/?t={}&apikey=19cc0456'
   path = 'movie_metadata/{}.h5py'
   folder = 'movie_metadata'
   allMetadata = []
   for title in movieTitles:
      currentMovie = []
      filePath = path.format(title)

      #Download OMDb metadata to a local file
      download_url(url.format(title),filePath)
      #Read the file to extract metadata
      metadata = extract_metadata(filePath)
```

```
        #Data not found
        if (metadata['Response'] == 'False') and (title!=""):
           print('Did not find "{}"'.format(title))
           currentMovie.append(None)
        elif(title != ""):
           currentMovie.append(metadata['Year'])
           currentMovie.append(metadata['Runtime'])
           currentMovie.append(metadata['Poster'])
           #Boolean used to extract more information
           if(details):
              currentMovie.append(metadata['Genre'])
              currentMovie.append(metadata['Rated'])
              currentMovie.append(metadata['Director'])
              currentMovie.append(metadata['Actors'])
              currentMovie.append(metadata['Plot'])


        allMetadata.append(currentMovie)

     #Wipes the local copy of the metadata once it has been used by the program
     if(wipeFile):
        for filename in os.listdir(folder):
           file_path = os.path.join(folder, filename)
           try:
              if os.path.isfile(file_path) or os.path.islink(file_path):
                 os.unlink(file_path)
           except Exception as e:
              print('Failed to delete %s. Reason: %s' % (file_path, e))

     return(allMetadata)
```

Listing 5.1: *Code implementation used for metadata extraction from the OMDb API. A key is required to have access to the metadata.*

The code in *Listing 5.1* details how movie information was queried through the OMDb API. A unique key was requested before-hand in order to gain access to the website's database. An example of the metadata provided by OMDb is given in *Appendix A.2*. A limitation regarding this key is that is was restrained to 1000 requests every 24 hours, after which the access to OMDb's database would be denied. The number of maximum request was reached relatively quickly during the testing phase; A request for a second key was submitted to OMDb. Data extraction failure was handled through the request's response status. Such failure may occur due to poor internet connection, incorrect url or request timeout.

Finally, once the metadata for a specific movie had been used, the local copy of the file was cleared from memory to avoid unnecessary storage allocation.

5.2 The Recommender System

There are a wide variety of types of RS available on the market; Each type may be designed and implemented in a number of ways. In this section, we will examine the specific implementation details of the collaborative filtering based system implemented for this task.

5.2.1 The Cold Start Problem

The cold start issue is a widely known issue in the domain of RS. The situation commonly referred to a cold start can concern both users and items for which the system has little to no information.

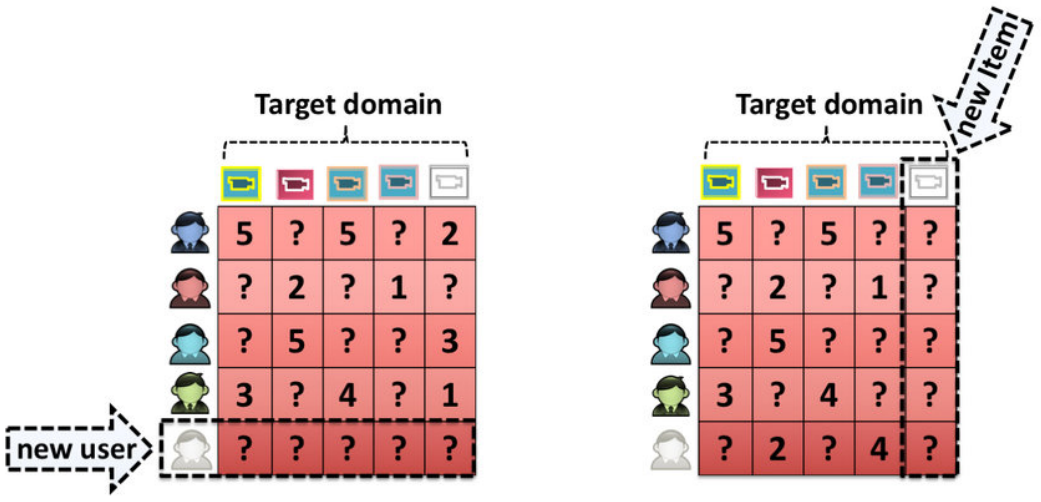


Figure 5.1: *Example of the cold start problem to highlight how making predictions for an unseen user or item can be challenging for some types of RS. Figure from Moghaddam and Elahi (2019).*

As shown in *Figure 5.1* by Moghaddam and Elahi (2019), it often characterises novel items or users, as these are more likely to have few given or received ratings. This issue affects certain types of RS more than others, depending on the information needed by the system to make predictions. For example, a content-based RS may still produce accurate predictions for a new item (with no ratings) given content information (generally in the form of tags describing the item).

CF-based systems are inherently more affected by this issue, as they rely on ratings to make recommendations. More specifically, participants in this project's study can be considered as concrete examples of the cold start problem: they were asked to provide at least 6 ratings to offset this issue. While this number remains relatively low compared to the average number of ratings provided by users in the MovieLens database, it is enough to help the system make predictions for the participant. As a general rule, the more ratings are provided by the participant during the user study, the more accurate the recommendations will be (participants were encouraged to rate at least 10 movies).

5.2.2 Model Implementation

The recommender model chosen for this project was implemented by the Spotlight tool (Kula (2017)). This tool provides a comprehensive explicit feedback model which is perfectly suited to MovieLens data, as highlighted in the previous chapter. Other advantages of this implementation is that various helper functions are provided alongside the main model to assist with data manipulation or model performance evaluation.

Before training the model, we add the participant's rating data to the existing MovieLens dataset. Following this step, the result is split incrementally into training and testing sets. We use a provided helper function called `random_train_test_split` to randomly split the user/item interactions according to the given split values. The splitting of the interactions is executed in two parts.

- First, we split the data following a common split ratio: 80% for the training set and 20% for the testing set.
- Secondly, the training set is split into further subsets using the same Spotlight helper function (we have found the optimal number to be 10 subsets for the final implementation). Each subset is used incrementally to train the model. The model's performance is measured and recorded at each training step (or incremental subset). The reasons behind this incremental training of the model will be further detailed in the next section.

The model's training is stopped as soon as overfitting is detected. overfitting is a common issue among RS: it describes the case where the model becomes overly reliant on the training data. This implies that the model will have a very high performance for the training data (as it has been specifically tailored to fit this data) but on the other hand will perform very poorly on unseen data as the learned parameters fail to adapt to new data. By definition, a common indicator that a model is overfitting is when the model's error on the testing test becomes significantly greater than its error on the training set.

Luckily, Spotlight provides another helper function to assist with detecting overfitting: `rmse_score`. RMSE stands for Root Mean Squared Error and measures the standard deviation of prediction errors for the given dataset. Thus, the higher the difference between the predicted ratings and the actual ratings, the higher the RMSE score. The RMSE score is calculated as follows:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (P_i - O_i)^2}{n}} \tag{5.1}$$

where P_i is the predicted rating, O_i is the actual rating and n is the total number of predictions. The output of this equation was directly generated through Spotlight's RMSE function. Further details regarding how this output was used will be provided in the section on model tuning (*Section 5.2.3*).

Once the model has finished training, predictions are made for the participant for every item in the dataset. Model predictions for each movie are directly obtained from Spotlight's `predict` function. Considering that our first explanation relied on the user's neighbours, we had to make a function to find the closest neighbours of the participant:

```
def assignClosestNeighbours(model, dataset, fileNeighUsers, embedding_dim,
    perplexity):

   #FIRST STEP: Assign neighbour users
   numUsers = dataset.num_users
   allUserFactors = np.empty((numUsers,embedding_dim))

   for i in range (numUsers):
      allUserFactors[i,:] = model._net.user_embeddings.weight[i].detach()

   # We are interested in the last user that has been added to the dataset.
   participantPoints = allUserFactors[numUsers-1,:]
   distances = []

   #Compute the Euclidean distance for high dimensions (more accurate).
   for index in range (numUsers):
        userXPoints = allUserFactors[index,:]
        intermediateSum = 0
        for k in range(embedding_dim):
            intermediateSum += (participantPoints[k]-userXPoints[k])**2

        dist = math.sqrt(intermediateSum)
        distances += [dist]

    distIndexes = np.argsort(distances)

    with open(fileNeighUsers, 'w') as f:
        for item in distIndexes:
            f.write("%s\n" % item)
    f.close()
```

Listing 5.2: *The following function finds the closest neighbours of the current user by measuring the Euclidean Distance between their associated feature-vectors.*

The function outlined in *Listing 5.2* shows how the neighbours of each user were computed through the 32-dimensional feature vector implementation provided by Spotlight. These neighbours were then saved to a specific file for future access. Saving large data structure to various files is an approach which was used frequently in this project as it proved more convenient and more efficient than passing these large data objects back and forth between various functions.

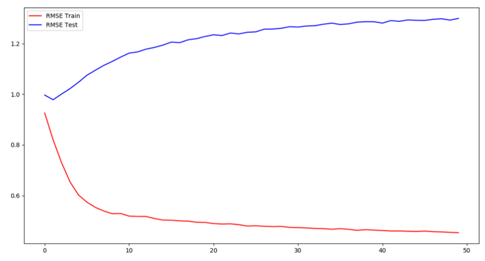
```
def shufflePredictions(recommendedTitles, recommendedIds):
    #first we use dictionaries to remember id, title and rank associations
    idToRank = {}
    idToTitle = {}
    rankCount=16
    for i in range(len(recommendedIds)):
        idToRank[recommendedIds[i]]=rankCount
        idToTitle[recommendedIds[i]]=recommendedTitles[i]
        rankCount -= 1

    random.shuffle(recommendedIds)
    shuffledRanks = []
    shuffledTitles = []
    for shuffledId in recommendedIds:
        shuffledRanks += [idToRank[shuffledId]]
        shuffledTitles += [idToTitle[shuffledId]]

    return(recommendedIds,shuffledTitles,shuffledRanks)
```

Listing 5.3: *This function is responsible for randomly shuffling the top 16 predictions made by the system for the user.*

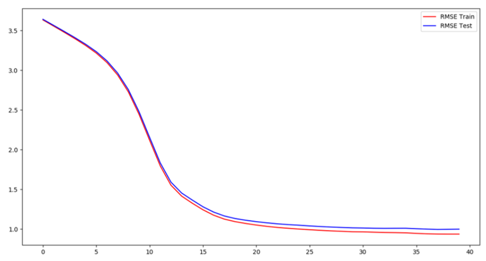
The code provided above illustrates the shuffling process. Before the start of the shuffling phase, a mapping is created between movie prediction ranks (from 1 to 16), titles and IDs using dictionaries of key-value pairs. Only the top 16 predictions are retained and randomly shuffled before being passed on to the interface responsible for displaying those recommendations.

5.2.3 Model Tuning

The explicit factorisation model provided by Spotlight provides many parameters which may be adapted to fit a specific task. We used the RMSE score as performance indicator to tune the most important parameters for this task.



(a) The entire training dataset is used to train the model at each iteration.



(b) The training dataset has been split into 40 subsets and each iteration is performed on a different subset.

Figure 5.2: *Each graph shows the evolution of RMSE scores at each iteration. The models were trained on the MovieLens 100K dataset, with a default learning rate of* 1^{-2} *and a standard train/test split of 80/20. In (a), the entire training set is used at each iteration. In (b) on the other hand, the training set has been split into subsets, where each subset corresponds to an iteration.*

The above figure shows records of the measurements of the RMSE score at every model iteration. *Figure 5.2a* shows the initial recorded measurements for 50 iterations where the entire training data was used at every iteration. As we can see, after only 1 or 2 iterations, the RMSE test score begins to increase rapidly. This is a clear sign of overfitting, as the model fails to adapt to unseen data (while over-performing for the RMSE train score). Following these observations, we have decided to further split the training set into a number of subsets; This would allow us to train the model in incremental steps in order to stop the training process before the overfitting stage. Many combinations between the number of subsets and the number of iterations have been tested. For example, dividing the training data into 10 subsets and training the model in 30 steps would imply that each subset is used for training 3 times. Lowering the number of training steps from 30 to 5 would mean that the model will only train on 5 of the 10 subsets. For each tested combination, we observed any signs of overfitting. We also observed the speed of convergence to ensure that training would not take too long.

In the end, the total number of steps is somewhat irrelevant as the model will stop training as soon as signs of overfitting are detected (subsequent increases in RMSE test scores).

The model parameters used in this task and tuned when necessary are detailed below:

- **Number of iterations**: this is the number of iterations used to run the model at each training step (ie. for each incremental training data split). Spotlight has a verbose mode which displays the loss at each epoch during model training which was used to optimise the number of iterations.
- **Embedding Dimension**: this parameter describes the number of dimensions used by the model to encode each user and each item. This is essentially a feature vector (of n dimensions) representing a user or an item. These vectors provide an insight on how similar two items/users may be. We have used the default value of 32 for this parameter as it is enough to accurately represent each item and user.
- **Learning Rate**: this is an important parameter which essentially controls how fast the model adapts to the data based on the measure estimated error. If it is too small, the training process will take more time and the model will struggle to learn about the data. On the other hand, an excessively large learning rate may result in over-fitting and sub-optimal

learning. Tested values for the learning rate are 1^{-2}, 1^{-3}, 5^{-2} and 5^{-3}. The latter value was used in the final implementation as it performed the best for our dataset during the testing phase.

- **Random State:** the random state is used to provide a seed for the model to guarantee reproducibility. This ensures that we can compare results between various runs without having to account for variations on the model's side (which could impact results).

5.3 Graphical User Interfaces

5.3.1 TkInter

The user interfaces required for this project were implemented through the Python TkInter package. This package facilitates creation and interaction with visual elements (buttons, fields, lists, sliders...) also referred to as widgets. It comes with built in geometry managers to arrange the layout of the widgets on a page: we used the `grid()` over the `pack()` method as was better suited to build on interface resembling the design of popular platforms like Netflix. Each interface was implemented by a corresponding Python class or method.

```
def poll(self):
   #Get value of the entry box
   self.search = self.search_var.get()
   if self.search != self.search_mem:
      self.update_list(is_contact_search=True)

      #set switch and search memory (switch is False at the start)
      self.switch = True
      self.search_mem = self.search

   #Reset listbox if search is cleared
   if self.switch == True and self.search == '':
      self.update_list()
      self.switch = False

   #Update stars every 300 ms based on rating's value
   self.update_star_image()
   self.after(300, self.poll)
```

Listing 5.4: *This function belongs to the* Application *class which is used to implement the interface where users can view and rate movies*

The above code enabled the user to search for specific movies within the dataset. It was also responsible for updating the number of stars based on the user's ratings. The stars are represented as an image rather than an interactive element. An interactive element would have been preferred considering many users are familiar with this way of rating; This is one of the weaknesses of TkInter, although the interfaces are clear and simple to implement, it comes at the cost of fewer customisation options like custom widgets (such as interactive stars). For these reasons, we decide to keep the slider implementation: although this method of rating would seem less familiar to participants, we considered it would be simple enough.

5.3.2 Graph Plotting

The various types of graphs such as box plots, histograms or scatter plots were constructed with the help of Matplotlib, a Python library tailored to simple and interactive visualisations of data. A wide range of visualisation methods were tested before selecting the ones which would be needed for the explanation interfaces. Examples include graphs depicting all the users in the dataset, neighbours, items, the content of items or even a mix between users and items.

5.3.3 Dimension Reduction

In order to plot all users or items through a Matplotlib visualisation, we needed a 2-dimensional representation of each item and each user in the dataset. As discussed before, the model characterised each user or item through a 32-dimensional feature vector. We used a tool called tSNE developed by Laurens Van Der Maaten and Geoffrey Hinton: it is a dimensionality reduction tool which conserves relative distances between each object in high-dimensions. In other words, objects that are similar in high-dimensions (such as our user/item feature vectors in 32-dimensions) will be represented by relatively close points in 2-dimensions. This proves very convenient to visualise users and movies from the MovieLens dataset using a scatter plot.

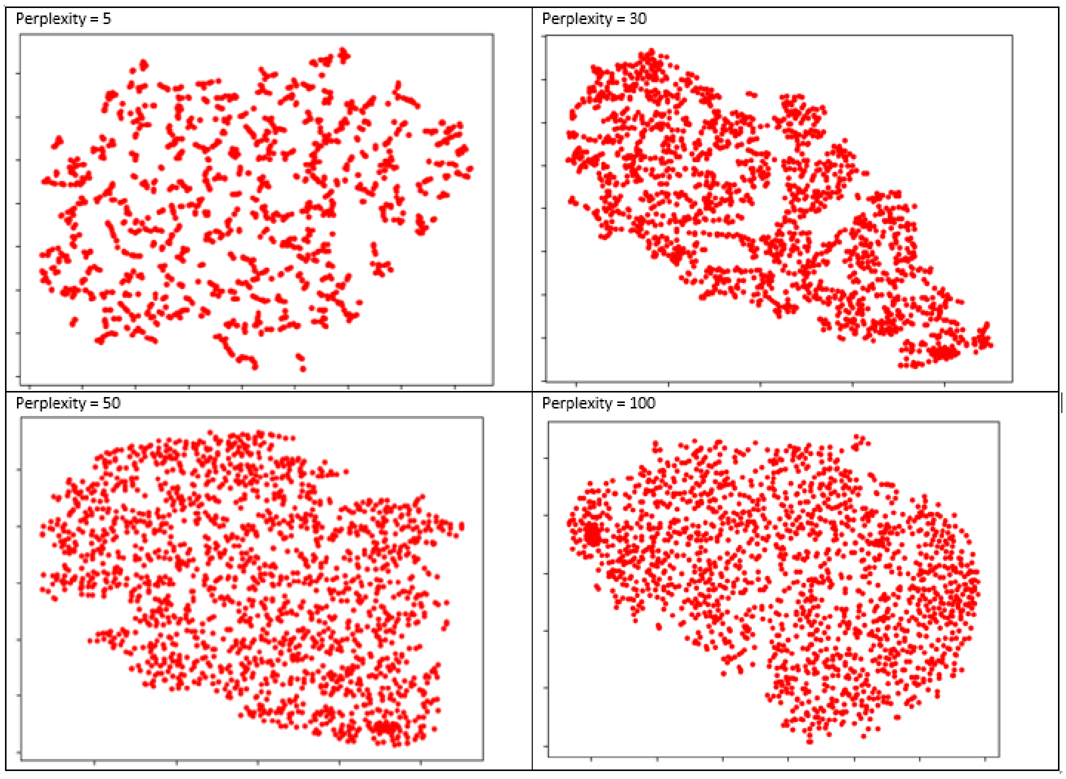


Figure 5.3: Scatter plot visualisations of all items contained in the MovieLens 100K dataset using tSNE with different perplexity settings.

The principle parameter used by tSNE during the dimensionality reduction process is known as perplexity; The developers behind tSNE advise that the perplexity's value lie between 5 and 50. Perplexity itself is defined as determining the number of nearest neighbours for each point in the graph: as observed in *Figure 5.3*, lower perplexity values will tend to favour point clusters while higher values will usually lead to more evenly-spaced points. The visualisations were tested alongside the model to find the best combination between the two. In the end, the final implementation contained a tSNE perplexity value of 40.

One of the issues encountered with tSNE was that some representations appeared extremely stretched or distorted. Furthermore, the produced visualisations seemed very inconsistent in some cases: visualisations were produced during each step of the training phase and sometimes lacked continuity between one step and the next. This issue was mitigated using Principal Component Analysis (PCA) to reduce from 32 dimensions to 10 dimensions, before using tSNE to further reduce the dimensions from 10 to 2 (rather than 32 to 2). PCA is a deterministic transformation which can be used for dimensionality reduction tasks. Resulting visualisations obtained using this method appeared overall more consistent. Lastly, we noticed that longer training times had a positive effect on the consistency of these visualisations.

5.4 Code Maintainability

Considering that this project is research-based rather than product-based, the need for code maintainability was most imperative during the development stage of the project. Steps taken to ensure satisfaction of this need are discussed below.

5.4.1 Version Control

The need for version control was not a priority considering only one person was working on this project. Furthermore, the project's aim was not to design a product which could be further maintained and enhanced by future developers. Nevertheless, version control was achieved using GitHub to enforce good coding practices.

5.4.2 Documentation

The GitHub repository dedicated to this project contains some documentation found in the *README* file. The contents of this file details some of the tools used throughout this project. Initial set-up instructions are also provided to facilitate a potential deployment of this project on another machine.

6 | Evaluation

This chapter will review the main objectives of this project and attempt to answer them through the analysis of the results obtained from the user study.

6.1 Research Question

The central focus of this project is to assess the impact of explanation methods for a collaborative filtering based RS. The means of evaluation are quite flexible: they were more specifically defined during the later stages of the project to ensure they provided a good coverage of various aspects of explanation techniques. There is currently no established standard for evaluating such methods in industry. Our solution is to design a series of questions to ask the user regarding the provided recommendations. Each question measures a specific metric. This process is detailed in the next section.

The second goal of this research is to determine if there is a potential user bias when evaluating explanation methods. Due to the relatively small number of participants, we will not make any generalisations regarding this notion but rather examine how the user's characteristics may affect their feedback on a case-by-case basis.

6.2 Explanation Methods Evaluation

6.2.1 User Study Outline

The general structure of the interview process is summarised in this subsection while the specific questions may be found in *Appendices D.3 to D.9*.

Following the guidelines defined by the Ethics Committee, the study begins by outlining the general aims of the project and detailing the interview's structure to the participant (see *Appendix D.1*), before seeking consent. Following the main steps behind collaborative explanations as defined in a lecture on RS by Macdonald (2020), the second stage focuses on gathering some information regarding the user to analyse potential user bias (feedback acquisition stage). The following user information will be collected during this stage:

- **User's age**: this may impact their views and opinions of RS and explanation methods as a whole.
- **User's familiarity with the type of data**: our project provides movie recommendations to the user. It is important to assess the user's level of familiarity with the recommended items: participants who rarely interact with movies and TV-shows may have different opinions than users who watch them on a daily basis.
- **User's familiarity with movie recommendation systems**: this question aims to provide a general idea of how often the user interacts with popular RS. It is closely linked to the previous point but has the added benefit of measuring how the user's interacts with personalised recommendations (and not mainstream TV for example).

- **User's views on recommendation systems**: we ask users if they are generally satisfied by recommendations provided to them. We also seek to determine if they believe popular recommendation systems should provide more explanations as this may impact their feedback. We gather this data to determine how well the user understands the recommendation process as a whole.

Following the initial information gathering phase, we ask the user to provide ratings for a least 6 movies they have seen within the database. Although we initially demanded that they rate a minimum number of 10 movies to ensure sufficient accuracy in the recommendations, we later lowered this number to 6 for reasons given in *Subsection 6.2.3* on limitations.

Once we have collected information on the user's general taste in movies, we run our model and provide a set of different recommendations for the user. We further detail this step in the next section.

The user study ends after evaluation of all three explanation methods by the user. Since the answers are recorded using Google Forms, we use a key as mapping to associate the user's answers with the program's hidden output metadata. This metadata contains user's ratings and shuffling information regarding the rank of recommended items and the order of explanations (example of these logs are shown in *Appendix A.1*). Finally, a quick debrief resuming the interview is given (see *Appendix D.2*): the user may ask any final unanswered question before the interview ends.

6.2.2 Experimental Design

This section is dedicated to the feedback gathering stage of the user study.

First of all, we obtain a list of the top 16 recommendations for that user based on the ratings he has provided. We take some precautions to minimise the impact of unwanted effects on the study's results:

- **Impact of the quality of the recommendations**: in order to reduce the impact of the quality of the provided recommendations on the feedback provided by the user, we randomly shuffle the top 16 recommendations into 4 sets of 4 recommendations. Out of the 4 sets, 3 are accompanied by an explanation method and 1 is given without any form of explanation. This ensures that the user will not rate an explanation method more highly because is was associated with a better set of recommendations, but because the explanation itself was more effective.
- **Learning Effect**: to account for learning effects, the order in which the explanations are presented to the user is also randomised between different users. However, the recommendations with no explanations are always presented first (and serve as baseline comparison). This is essential to offset potential learning effects where the users may become more familiar with the explanation process towards the end of the study. This would be problematic as a higher understanding of this process may lead them to rate the last explanation methods differently from the first ones.

For each set of 4 recommendations, we ask the user to provide some feedback through a Likert scale. This design was chosen to provide quantitative data regarding each metric, which can be further analysed. The following metrics were used to evaluate our explanation techniques. The majority of these metrics have been defined by Daher et al. (2018) in their work entitled *A Review on Explanations in Recommender Systems*:

- **Persuasiveness** measures how likely the user is to follow the system's recommendations. In this case, we asked users how likely they were to watch each recommended movie. This metric is similar to the notion of promotion introduced by Bilgic and Mooney (2005) in *Chapter 2*.
- **Effectiveness**: we asked the participant if he felt that the system had provided effective recommendations overall.

- **Transparency** estimated the user's understanding of the recommendation process. He was asked to state if he understood why these items had been recommended to him based on the ratings he had provided.
- **Scrutability** measures the extent to which the user is able to tell that the system is mistaken. This can favour a sense of forgiveness, as the participant if more likely to forgive the system if he understand why the mistakes were made.
- **Trust** is used to represent the extent of the user's confidence in the system and its ability to produce highly effective recommendations. This is largely affected by the performance of the model and the familiarity of the user with a given system. Nevertheless, we believed it would be interesting to measure this factor.
- **Helpfulness** is a metric which we added to measure the usefulness of the explanation in the user's decision process. We asked the user how the explanation had impacted their review of the recommended items. This is the only metric which was not recorded for the baseline recommendations (with no explanations).

The obtained results for each metric are analysed in the next section.

6.2.3 Limitations

One of the major limitations which we had not considered during the early design stages of the project is the date of publication of movies within our dataset. The most recent movies in the MovieLens 100K dataset were released in 1998, which is relatively old by modern standards. Considering that the majority of the users taking part in this study were students, many of them were not familiar with movies released during this time period. As a result, it was difficult for some users to find and rate 10 movies they had seen within our database. In an attempt to facilitate this process, we reduced the number of required ratings from 10 to 6.

This gain comes at the cost of the effectiveness of the provided recommendations. However, we considered that this would not have a significant impact on our study as we are not directly assessing the effectiveness of the recommendations. As long as they are generally effective and displayed in random order, it should not have a significant impact on the assessment of explanation methods.

6.3 Analysis of study results

6.3.1 Review of Participant Characteristics

In total, this study has been performed by 11 participants. We had planned to interview at least 1 additional participant but under the exceptional circumstances resulting from the ongoing health crisis, these interviews were cancelled. Furthermore, the project's configuration only allowed for in-person interviews which significantly reduced the number of potential participants.

In total, 7 (63.6%) of participants were aged between 18 and 25 years old. The remaining users were equally distributed between the 26 to 40 range and the 41 to 59 range (see figure in *Appendix B.1*. A younger demographic is well-suited for this study as younger users tend to be more accustomed to modern-day RS.

Furthermore, 6 (54.5%) users interacted with RS on a daily basis. The rest of the participants used RS on a weekly basis (see *Appendix B.3*). Concerning the number of movies and TV-shows watched weekly, 4 (36.4%) users watch between 1 and 3 while 4 other users claimed to watch between 5 and 7 (see *Appendix B.2*). Overall, these results indicate that the participants were used to interacting with the type of recommended item and with RS in general.

The last questions regarding user characteristics asked about their opinion on existing popular RS. In general, 2 users were very satisfied by these services, 4 were satisfied, 2 were neutral and 3 were dissatisfied (see *Appendix B.4*).

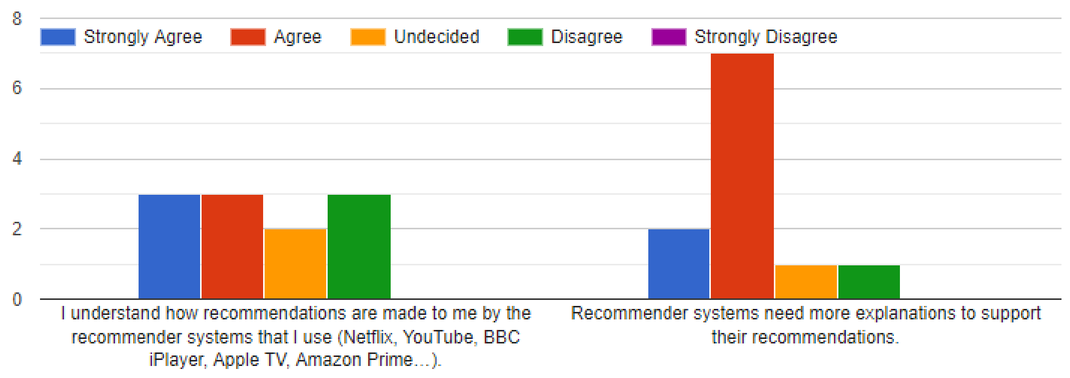


Figure 6.1: *Histograms containing user answers to the associated question (results gathered using a Likert Scale). The* y-axis *represents the number of users*

Figure 6.1 highlights interesting results. The participants' overall understanding of the recommendation process behind popular systems is somewhat evenly distributed for each represented category; more than half of them (6 users) understand this process (either agree or strongly agree with the statement). The second histogram in Figure 6.1 shows a greater imbalance in the answers. In total, 7 out of 11 users agree that RS in general need more explanations to support their recommendations.

Our results have not brought evidence of any correlation between the type of user (age, familiarity with RS, current opinion and understanding of RS) and the general feedback they have provided for each explanation technique. This is likely due to the relatively small number of selected participants, which was not representative of a wider population and not adequate to highlight this kind of pattern.

6.3.2 Interface for Preferences Acquiring

This section provides a brief review of the feedback given regarding the user interface designed to collect user ratings. This is also an important component of the project, as any major issue with this interface may affect the user's ability to provide ratings and therefore the results of the study.

In total, 9 users conceded (4 strongly agreed and 5 agreed) that the interface provided was easy to use. Despite the interface being easy to use, 3 users disagreed with the statement that it was easy to find movies to rate. This is largely due to the date of publication of the movies, as discussed previously.

The option to filter items by genre was more appreciated by user than the search feature: 9 users either agreed or strongly agreed that the former helped them find movies, while 7 said the same for the latter. This is understandable considering that the database only contained around 1700 movies: it was unlikely that the user would search and find a specific movie.

Lastly, 8 users either agreed or strongly agreed that the interface provided enough information about each movie. The remaining user were undecided. Providing more metadata in this interface was not a priority considering that the users were expected to rate movies they had already seen.

6.3.3 Statistical T-test Results

This section will analyse the quantitative results obtained for each of the metrics listed above in order to determine if there is a significant difference between the recommendations provided with and without explanations. We will perform a paired t-test to test for a significant difference in the means of our results between the baseline recommendations and recommendations with explanations. Before performing the statistical test, we check that the following assumptions hold:

- **Scale of measurement**: the gathered data follows an ordinal scale (from strongly disagree to strongly agree).
- **Simple Random Sampling**: implies that the data is collected from a representative portion of the population. We cannot assume that the 11 participants used in this study are representative of a wider population. Thus, our analysis will be limited to the context of this study and will not be generalised to a wider population.
- **Normal Distribution**: the plotted data follows a normal distribution. We have checked that the collected data approximately follows a bell-shaped distribution curve. Indeed, most participants tend to have average opinions (ie. they tend to *agree* or *disagree* more than they *strongly agree* or *strongly disagree*).
- **Sample Size**: the sample size should be large enough. The t-test is suited to compare the means of small samples (generally regarded as being less than 30).
- **Similar variance**: the standard deviation measured for each question are relatively similar.

We can now compare the obtained results for each explanation method for the different measured metrics. We make the following hypothesis with a significance level set at 0.05:

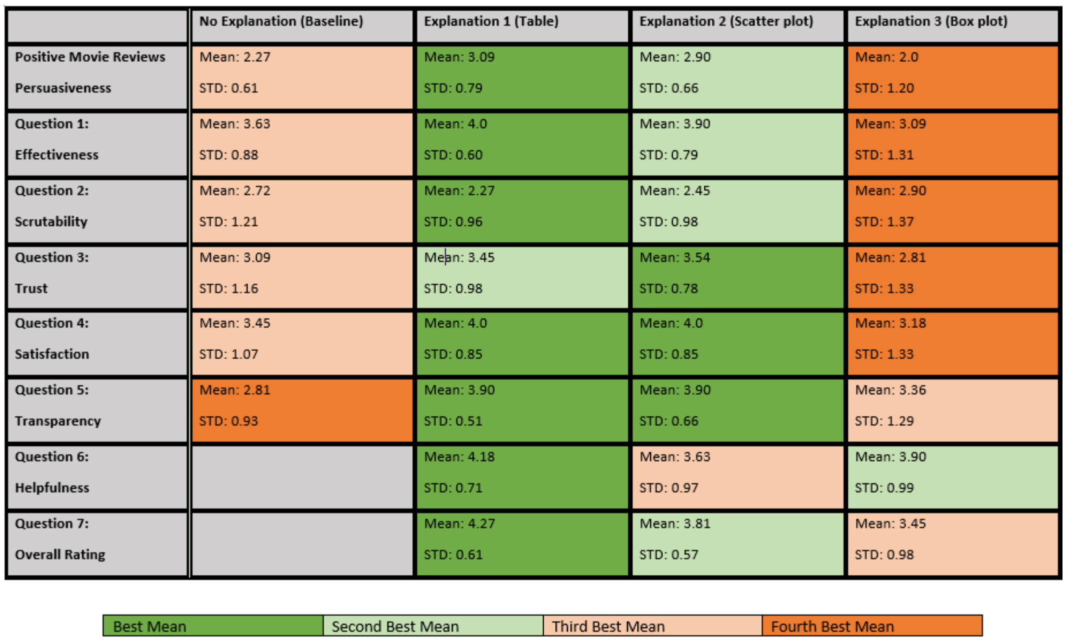
- **Null Hypothesis**: there is no significant difference between the mean value of results for the baseline recommendations and for recommendations with an explanation. Case where $p - value > 0.05$.
- **Alternate Hypothesis**: there is a significant difference between the mean value of results for the baseline recommendations and for recommendations with an explanation. Case where $p - value <= 0.05$

The means and standard deviations associated with different metrics and recommendation sets are listed below, where a score of 5 corresponds to users strongly agreeing with the question (see *Appendix D.8* and 1 represents users strongly disagreeing (3 being undecided):

Table 6.2 displays the main results gathered for this project regarding the evaluation of different explanation methods using a set of metrics. For each row, the results are highlighted in different colours to rank them from best to worst in terms of their mean (standard deviation is not considered in this ranking). Note that in the case of scrutability, a lower mean indicates better results. The helpfulness and overall explanation rating are not given for the baseline considering that it was not tied to any form of explanation. Regarding the row on *Positive Movie Reviews*, we asked user to state how likely they would be to watch recommended movies: answering either *likely* or *very likely* to this question is considered as positive review.

Judging from this figure (6.2), the first explanation method has performed the best for almost all metrics aside from trust. The second explanation method has also performed quite well compared to the baseline. On the other hand, the final explanation has done quite poorly for the vast majority of metrics (aside from helpfulness were it outperforms explanation 2) when compared with the baseline recommendations. This hints that this method most likely confused users and negatively impacted their interaction with the output provided by the RS. This indicates that no explanation may be better than a confusing or poorly designed explanation method.

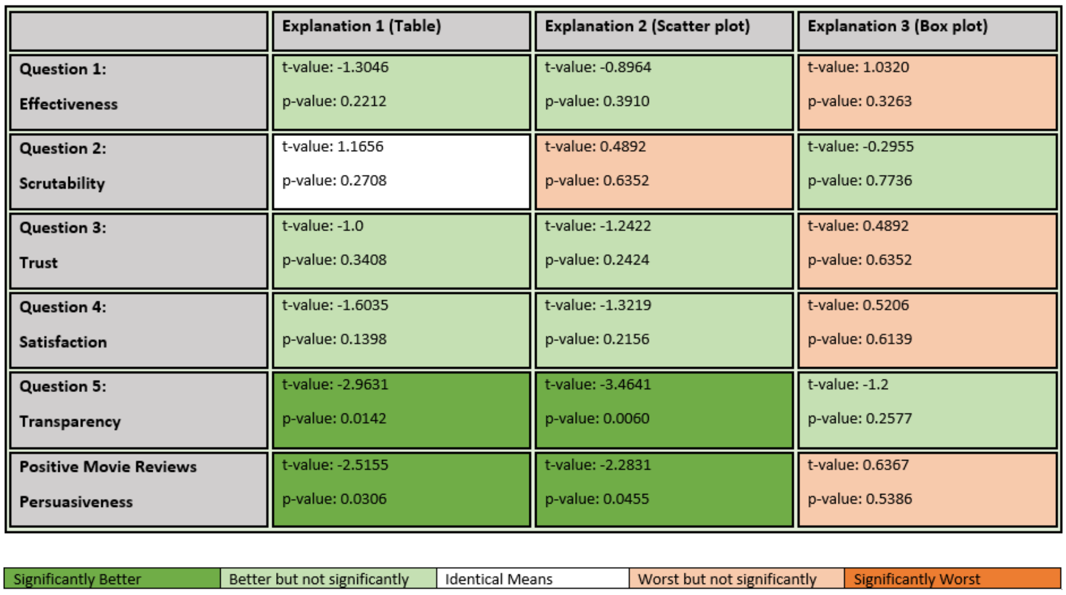
Although the observed means can be compared, *Table 6.2* does not indicate whether or not the observed differences are significant. We perform further analysis of the obtained results through a two sample t-test to provide more insight on this topic:



	No Explanation (Baseline)	Explanation 1 (Table)	Explanation 2 (Scatter plot)	Explanation 3 (Box plot)
Positive Movie Reviews Persuasiveness	Mean: 2.27 STD: 0.61	Mean: 3.09 STD: 0.79	Mean: 2.90 STD: 0.66	Mean: 2.0 STD: 1.20
Question 1: Effectiveness	Mean: 3.63 STD: 0.88	Mean: 4.0 STD: 0.60	Mean: 3.90 STD: 0.79	Mean: 3.09 STD: 1.31
Question 2: Scrutability	Mean: 2.72 STD: 1.21	Mean: 2.27 STD: 0.96	Mean: 2.45 STD: 0.98	Mean: 2.90 STD: 1.37
Question 3: Trust	Mean: 3.09 STD: 1.16	Mean: 3.45 STD: 0.98	Mean: 3.54 STD: 0.78	Mean: 2.81 STD: 1.33
Question 4: Satisfaction	Mean: 3.45 STD: 1.07	Mean: 4.0 STD: 0.85	Mean: 4.0 STD: 0.85	Mean: 3.18 STD: 1.33
Question 5: Transparency	Mean: 2.81 STD: 0.93	Mean: 3.90 STD: 0.51	Mean: 3.90 STD: 0.66	Mean: 3.36 STD: 1.29
Question 6: Helpfulness		Mean: 4.18 STD: 0.71	Mean: 3.63 STD: 0.97	Mean: 3.90 STD: 0.99
Question 7: Overall Rating		Mean: 4.27 STD: 0.61	Mean: 3.81 STD: 0.57	Mean: 3.45 STD: 0.98

Best Mean	Second Best Mean	Third Best Mean	Fourth Best Mean

Figure 6.2: *Collection of means and standard deviations resulting from the participants' answers on various metrics and for various recommendation sets (including the baseline). The values are truncated after the second decimal.*



	Explanation 1 (Table)	Explanation 2 (Scatter plot)	Explanation 3 (Box plot)
Question 1: Effectiveness	t-value: -1.3046 p-value: 0.2212	t-value: -0.8964 p-value: 0.3910	t-value: 1.0320 p-value: 0.3263
Question 2: Scrutability	t-value: 1.1656 p-value: 0.2708	t-value: 0.4892 p-value: 0.6352	t-value: -0.2955 p-value: 0.7736
Question 3: Trust	t-value: -1.0 p-value: 0.3408	t-value: -1.2422 p-value: 0.2424	t-value: 0.4892 p-value: 0.6352
Question 4: Satisfaction	t-value: -1.6035 p-value: 0.1398	t-value: -1.3219 p-value: 0.2156	t-value: 0.5206 p-value: 0.6139
Question 5: Transparency	t-value: -2.9631 p-value: 0.0142	t-value: -3.4641 p-value: 0.0060	t-value: -1.2 p-value: 0.2577
Positive Movie Reviews Persuasiveness	t-value: -2.5155 p-value: 0.0306	t-value: -2.2831 p-value: 0.0455	t-value: 0.6367 p-value: 0.5386

Significantly Better	Better but not significantly	Identical Means	Worst but not significantly	Significantly Worst

Figure 6.3: *Results obtained from the statistical T-test when comparing each set of recommendations with explanations to the set of recommendations with no explanation (baseline) using a significance level of* 0.05

.

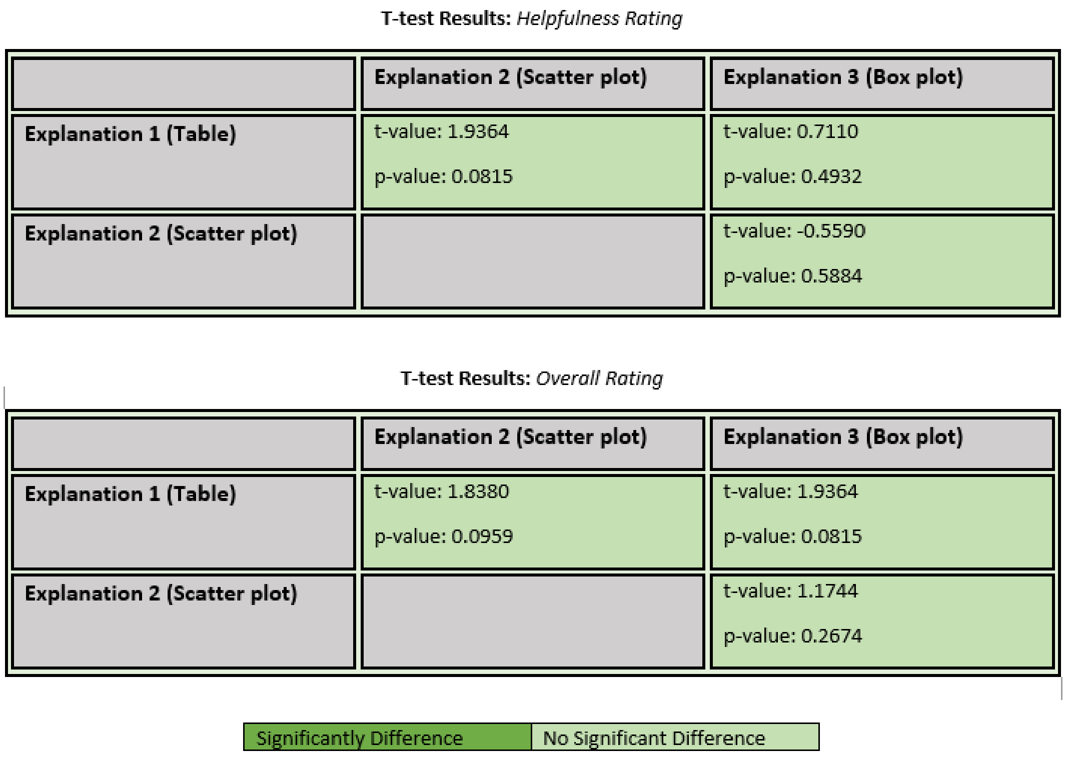
The obtained results are compiled in *Figure 6.3* where each row corresponds to a metric (aside from helpfulness which was only measured for recommendations with explanations). Note that row 2 (scrutability) is the only case where a higher mean is negative due to the way the question was formulated. A higher score for this question indicates that users think that the system was mistaken during the recommendation process. The *t-value* serves as an indication of the mean difference represented in units of standard error: the more extreme (either positive or negative), the more likely it is that the observed difference was not caused by chance (hence the significance). Greater *t-values* are associated with lower *p-values*, which measure the odds of observing such difference.

At first glance, explanation method 3 which relied on box plots to provide an overall assessment of each recommended item seems to have performed the worst. When compared to the baseline, the obtained mean was (not significantly) worst when measuring user perceived effectiveness, sense of trust, satisfaction and persuasiveness. As specified, a higher mean in row 2 also indicates that the scrutability was worst than it was for the baseline. This explanation seemed to confuse users more than it clarified the recommendations. Furthermore, the fact that most if not all of the provided box plots were very similar was pointed out as additional comment by 4 out of 11 users. This follows Herlocker et al. (2000)'s observations that poorly designed explanation interfaces may be worse than no explanation at all. This may be due to a number of reasons, some of which are provided in the final summary (7.1).

On the other hand, recommendations associated with explanation methods 1 and 2 have performed reasonably well. The results from each explanation are very similar and will be further compared through the helpfulness metric. Although the obtained results were not significantly better for the majority of metrics, the measured mean was nevertheless better for results measuring effectiveness, trust and satisfaction. The observed means were either identical or worst when measuring scrutability, which implies that scrutability of the system was either identical or better respectively. We can reject the null hypothesis in favour of the alternate hypothesis at $p <= 0.05$ in four different cases. We can assume with a 95% confidence interval that there is a significant difference in the means of the results measuring transparency and persuasiveness between the baseline and both explanation method 1 and 2 in our sample. However, we cannot the null hypothesis in favour of the alternate hypothesis for any other compared metrics.

We will now compare the results measuring the helpfulness of each explanation method in terms of how it aided the user decide whether he should follow a recommendation or not. On top of reviewing helpfulness, we have asked user to provide an overall rating of the explanation method.

These results indicate that the first explanation method has performed the best on average in terms of both helpfulness and overall rating. Surprisingly, explanation 3 was more helpful than explanation 2 on average, yet it was rated more poorly overall. We perform further statistical t-tests to determine if there is a significant difference between the observed means for both helpfulness and overall score using a significance level of 0.05.



T-test Results: *Helpfulness Rating*

	Explanation 2 (Scatter plot)	**Explanation 3 (Box plot)**
Explanation 1 (Table)	t-value: 1.9364 p-value: 0.0815	t-value: 0.7110 p-value: 0.4932
Explanation 2 (Scatter plot)		t-value: -0.5590 p-value: 0.5884

T-test Results: *Overall Rating*

	Explanation 2 (Scatter plot)	**Explanation 3 (Box plot)**
Explanation 1 (Table)	t-value: 1.8380 p-value: 0.0959	t-value: 1.9364 p-value: 0.0815
Explanation 2 (Scatter plot)		t-value: 1.1744 p-value: 0.2674

Significantly Difference	No Significant Difference

Figure 6.4: *Results obtained from the statistical T-test when comparing each explanation method with one another for both helpfulness and overall rating using a significance level of* 0.05
.

Figure 6.4 indicates that there is no significant difference (at $p <= 0.05$) between the measured results of all three explanation methods for both helpfulness and overall score. Despite explanation method 3 having the worst performance in terms of the previously mentioned metrics, this is not transcribed by a significantly worst performance overall.

6.4 Satisfaction of Requirements

This step focuses on reiterating the initial requirements to assess which have been met and which have not. As the requirements have evolved throughout the course of this project, only the final list of requirements will be examined.

Both *must have* (requirements 1 and 2) have been met. A total of three different explanation techniques have been implemented and tested using a distinct type of interface for each one.

Similarly, requirements 3 and 4 have been fulfilled. In order to provide explanations tailored to new users so that they may provide feedback (rather than users which already existed in the database), new user data had to be assimilated by the model. An interface was provided specifically for the collection of this new data.

Only two out of three *could have* requirements have been met. Concerning requirement 5, we used the OMDb API to extract additional metadata regarding each item such as the poster, the plot, the actors and the director of the movie (the rest was initially found in the MovieLens 100K dataset). However, we have only implemented a CF-based RS alongside relevant explanation techniques. Explanations methods which have performed well for this type of RS may not perform well for other types of RS as they may not be as accurate (for example our set of explanations tended to focus on the user's neighbours over the item's content). As a result, we have not compared the performance of explanation methods for different types of RS. As stated in requirement 7, evaluation of the explanation methods was performed through a series of metrics (although they differ from the leads given by the initial requirement).

As a direct consequence of failing to meet requirement 6, requirement 8 has not been met as well.

6.5 Limitations

This section will list potential flaws which have been identified in the interview process and which may have impacted the final results of this study.

First of all, although the online questionnaire provided extensive information to give participants more autonomy during this study, the interview was still conducted in person. As a result, many participants chose to ask questions directly rather than read some of the textual information provided. Participants asked similar questions in general and the answers they have received may have slightly differed (not in overall meaning but rather in smaller details such as the completeness or structure of the provided oral answer). These minor variations may have affected the participant's understanding relating to the question they initially asked, which in turn may have affected their review of the explanation methods.

Another identified limitation caused by the need to perform this evaluation in person is that participants had to provide their feedback in front of the interviewer. This may have impacted the honesty of their feedback, as people are generally less likely to be harsh or critical in person.

The most common source of confusion among users concerned the second explanation method (scatter plot) where they wondered why some of the recommended items did not appear closest to them on the graph. The provided answer was quite technical: We explained the concept dimensionality reduction, insisting that although tSNE preserved the overall distances between each point (while reducing the number of dimensions), it is a non-deterministic tool and thus did not guarantee the same output between different runs. This may be one of the reasons why this explanation method has not performed as well as the first one in *Table 6.2*.

6.6 Summary

This chapter was dedicated to the evaluation of the results observed during this project. As defined, the main goal was to assess the impact of various explanations on users for a CF-based system. Some novel metrics have been introduced alongside already existing ones to evaluate said methods. The results were further analysed through statistical tests to determine their significance, although they have not been generalised beyond the scope of this project due to the relatively small sample of participants. In the final sections of this chapter, we examined which of the established requirements had been met as well as the potential limitations originating from this study.

7 Conclusion

7.1 Summary of project

RS have become increasingly popular in the past decade and used as tools by many companies to provide more personalised services to their customers. This growth has lead to the development of many different types of systems. In more recent years, the importance of high-quality explanation methods to provide further context around recommendations has been acknowledged by the scientific community as a useful tool to enhance user experience (rather than solely focusing the the effectiveness of the RS).

In this project, we have implemented a basic CF-based RS using Spotlight in order to make recommendations to a user. The objective of this work was to provide various explanation techniques for different recommendations in order to evaluate and quantify their impact on the users. There is currently no standard to evaluate the effectiveness of explanation methods for CF-based systems; We have established a list of metrics (some of which originate from Daher et al. (2018)) to assess the performance of these methods through user feedback. The retained metrics used in the evaluation process are persuasiveness, effectiveness, scrutability, trust, satisfaction, transparency, helpfulness and overall impression of the user concerning the RS and the associate explanations.

We have observed that the sampled users favoured well-structured, visual and informational explanation methods over general, technical explanation methods. Although this difference was not significant, explanation 3 has performed worst on average than recommendations given without any explanations in terms of effectiveness, trust, satisfaction and persuasiveness. We suppose this may be linked to the scientific nature of the explanation (some users were not familiar with box plots), the lack of personalisation (as box plots estimated the general popularity of an item regardless of the user) and possibly the similarity between the box plots associated with different items (as the system had a tendency to recommend popular movies over less popular long-tail items). On the other hand, explanation 1 and 2 have performed better than the baseline for the vast majority of metrics. Additionally, we have found a significant difference in their performance regarding transparency and persuasiveness. This indicates that the explanation methods helped convince the user to follow the recommendations and provided more insight on how those items had been recommended. This suggests that users preferred visual explanations (such as a scatter plot depicting relative item similarity) and well-structured explanations (such as a table). The participants seemed to appreciate an explanation with a substantial amount of information provided it is clearly explained.

7.2 Reflections

Upon reflection, there are a few things we would have done differently given more time or a fresh start for this project.

First of all, we would have selected a more recent database to ensure that younger participants are not disadvantaged when required to rate the items. Furthermore, it would be beneficial to remove the obligation for participants to complete this study in person (for example by creating a website which provides and explains the recommendations). This would reduce the amount of effort required from participants to take this study, and may also lead to more honest feedback. Furthermore, this approach would have enabled the participation of more users in this study, which would have improved the accuracy of our results and potentially impacted their significance, allowing us to make inferences for a wider population.

7.3 Future work

This project has merely scraped the surface of this topic and there is still much work to be done in the field of RS and more specifically explanation techniques. We have identified some opportunities for potential future work associated with this project.

Future research could focus on developing a rigorous procedure for analysing recommendations and their associated explanation methods. This would be a great contribution to the field, as it would enable different works to be directly compared using this standard. This would allow for comparison of the pros and cons of different types of RS and their associated explanation methods. There is currently no common standard which can be used to compare the performance of various explanation methods in different works, which makes the analysis of explanation methods somewhat restricted to the scope of the project (heavily relies on conditions such as the data, the participants, the types of RS, the types of explanations...).

Another potential area to explore would be to compare and contrast a much wider range of explanation techniques (maybe more customised or professional-looking than what TkInter allowed). We have only implemented three distinct explanations in this project, yet there are many other types of potential explanations. These could be compared not only for CF-based RS, but also for other types of RS such as knowledge-based RS, content-based RS or even hybrid RS. It would be interesting to evaluate how well explanations perform depending on what kind of RS was used to recommend the items: the explanation techniques would need to be adapted to the type of RS.

A Technical Aspects

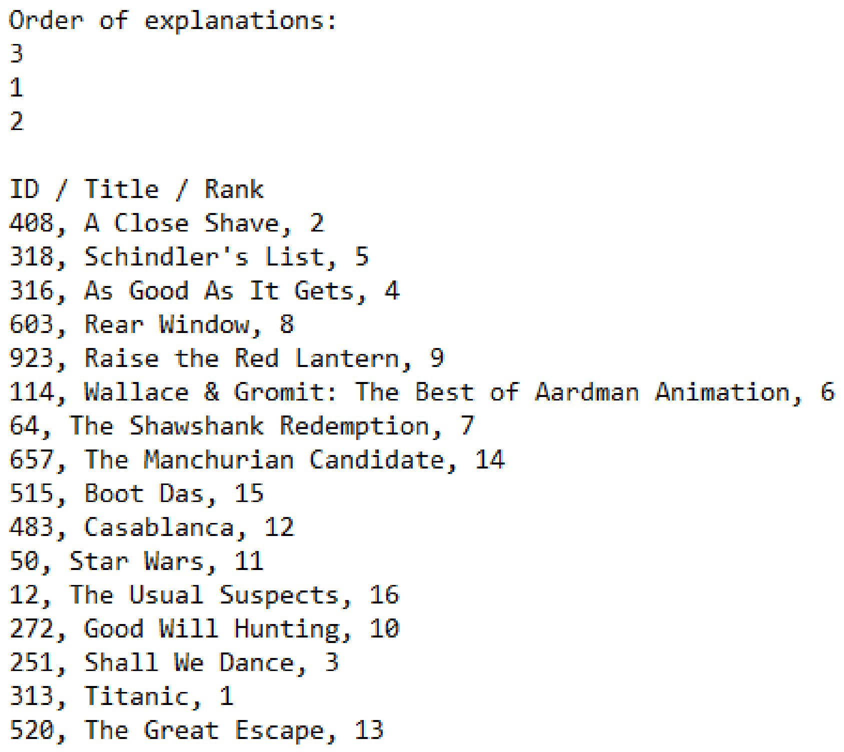
A.1 Code Extracts

```
#Some rows are poorly formatted an run over multiple lines
def concat_rows(rows):
   i = 0
   for row in rows:
      if(row[0].isnumeric()) is not True:
         rows[i-1] = rows[i-1] + rows[i]
         del rows[i]

      i += 1

return(rows)
```

Listing A.1: *Simple function performing row concatenation for cases where rows spanning multiple lines in the file rather than one.*

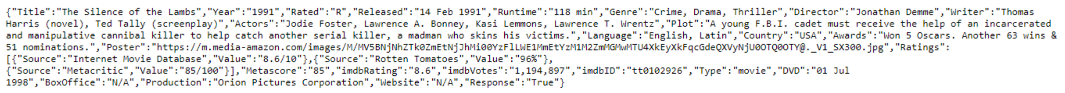
A.2 Other

The README contains basic information regarding the tools and set-up process for this project, and can be found here on GitHub.



```
Order of explanations:
3
1
2

ID / Title / Rank
408, A Close Shave, 2
318, Schindler's List, 5
316, As Good As It Gets, 4
603, Rear Window, 8
923, Raise the Red Lantern, 9
114, Wallace & Gromit: The Best of Aardman Animation, 6
64, The Shawshank Redemption, 7
657, The Manchurian Candidate, 14
515, Boot Das, 15
483, Casablanca, 12
50, Star Wars, 11
12, The Usual Suspects, 16
272, Good Will Hunting, 10
251, Shall We Dance, 3
313, Titanic, 1
520, The Great Escape, 13
```

Figure A.1: *A copy of logs saved by the program for each participant taking the study. This was used to keep track of which explanation method the participant was reviewing. The provided ratings (not shown here) were also part of the logs.*



```
{"Title":"The Silence of the Lambs","Year":"1991","Rated":"R","Released":"14 Feb 1991","Runtime":"118 min","Genre":"Crime, Drama, Thriller","Director":"Jonathan Demme","Writer":"Thomas
Harris (novel), Ted Tally (screenplay)","Actors":"Jodie Foster, Lawrence A. Bonney, Kasi Lemmons, Lawrence T. Wrentz","Plot":"A young F.B.I. cadet must receive the help of an incarcerated
and manipulative cannibal killer to help catch another serial killer, a madman who skins his victims.","Language":"English, Latin","Country":"USA","Awards":"Won 5 Oscars. Another 63 wins &
51 nominations.","Poster":"https://m.media-amazon.com/images/M/MV5BNjNhZTk0ZmEtNjJhMi00YzFlLWE1MmEtYzM1M2ZmMGMwMTU4XkEyXkFqcGdeQXVyNjU0OTQ0OTY@._V1_SX300.jpg","Ratings":
[{"Source":"Internet Movie Database","Value":"8.6/10"},{"Source":"Rotten Tomatoes","Value":"96%"},
{"Source":"Metacritic","Value":"85/100"}],"Metascore":"85","imdbRating":"8.6","imdbVotes":"1,194,897","imdbID":"tt0102926","Type":"movie","DVD":"01 Jul
1998","BoxOffice":"N/A","Production":"Orion Pictures Corporation","Website":"N/A","Response":"True"}
```

Figure A.2: *Copy of movie metadata information returned after performing a request using the OMDb API.*

B | Results: Additional Graphs

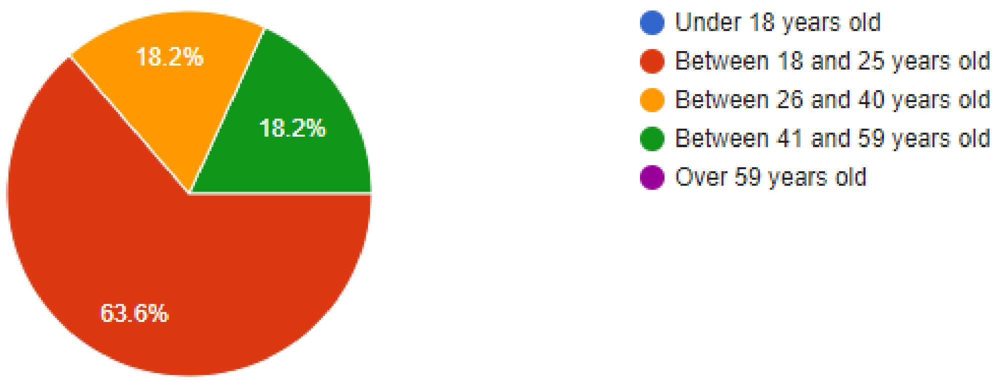


Figure B.1: *Results gathered from 11 participants regarding their age.*

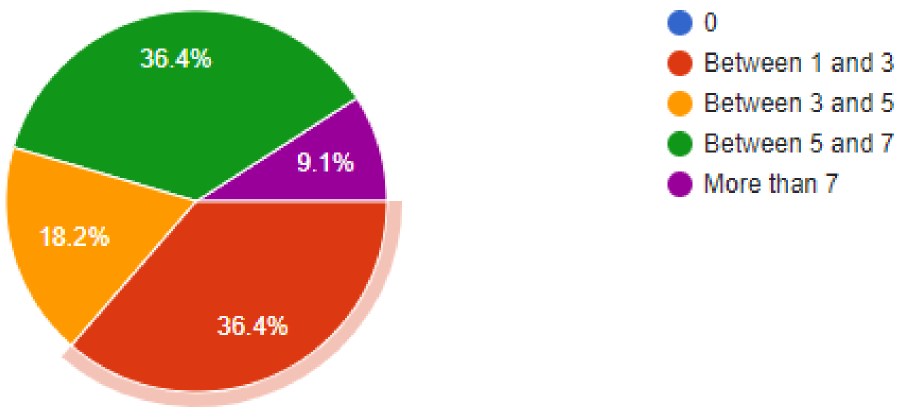


Figure B.2: *Results gathered from 11 participants regarding how many movies or TV-shows they watch every week.*

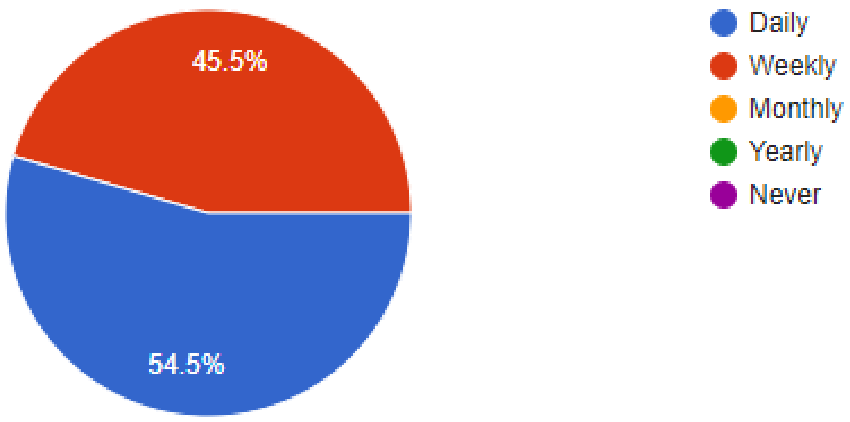


Figure B.3: *Results gathered from 11 participants regarding how frequently they use popular recommendation-based services.*

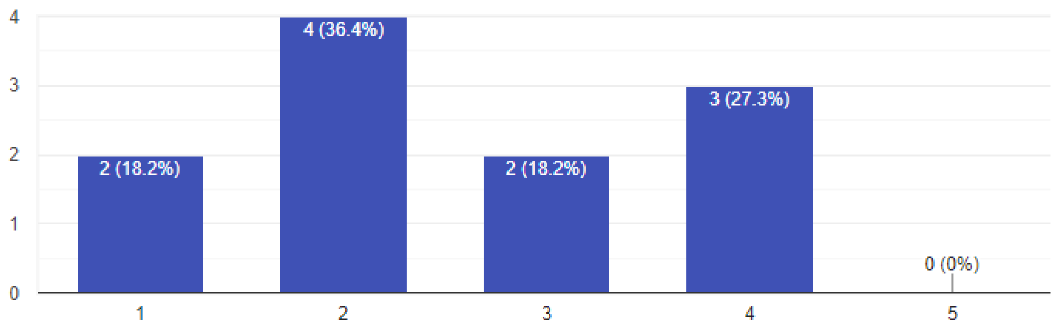
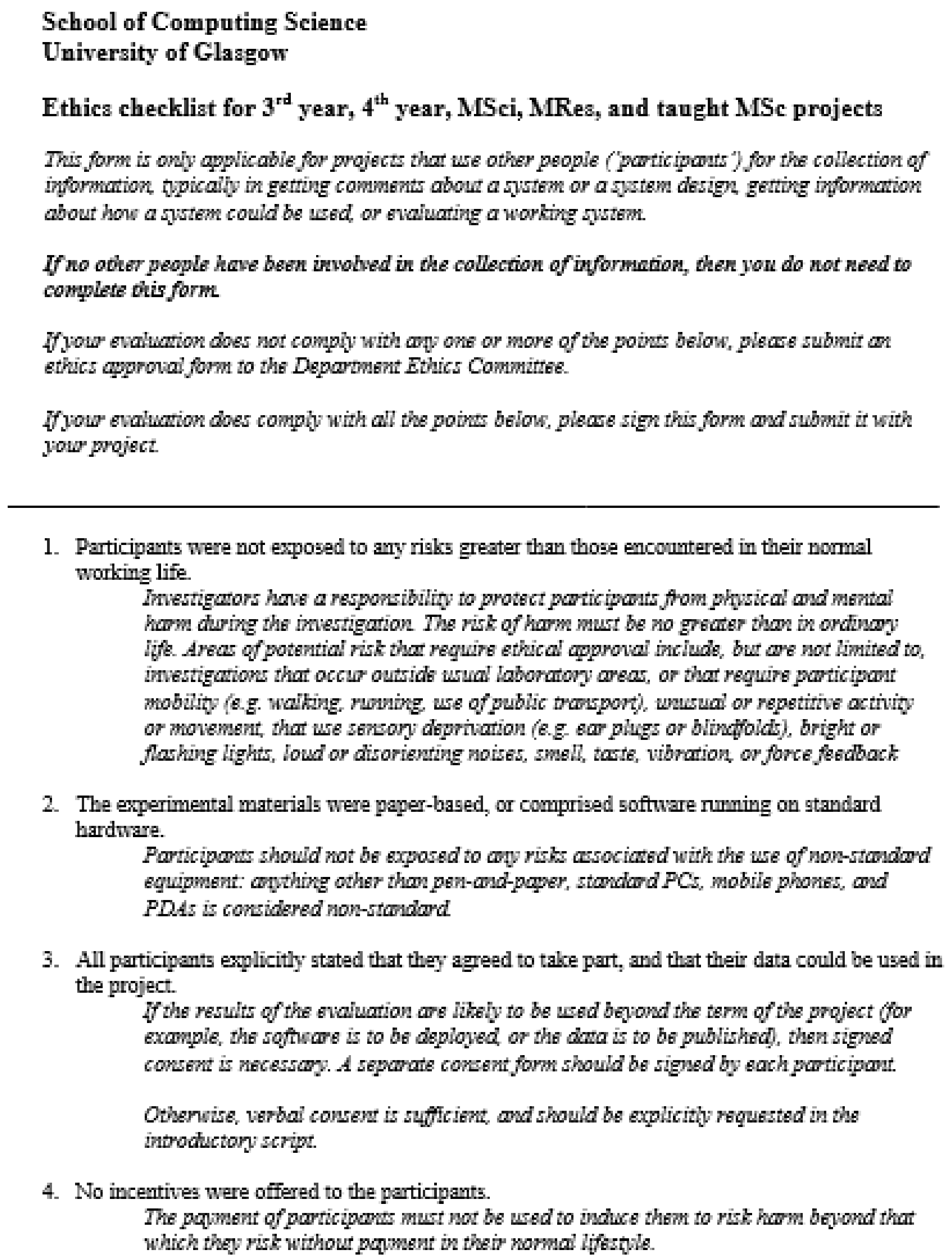


Figure B.4: *Results gathered from 11 participants regarding their level of satisfaction with RS that they use (1 being very satisfied to 5 being very dissatisfied).*

C | Ethics Approval



School of Computing Science
University of Glasgow

Ethics checklist for 3rd year, 4th year, MSci, MRes, and taught MSc projects

This form is only applicable for projects that use other people ('participants') for the collection of information, typically in getting comments about a system or a system design, getting information about how a system could be used, or evaluating a working system.

If no other people have been involved in the collection of information, then you do not need to complete this form.

If your evaluation does not comply with any one or more of the points below, please submit an ethics approval form to the Department Ethics Committee.

If your evaluation does comply with all the points below, please sign this form and submit it with your project.

---

1. Participants were not exposed to any risks greater than those encountered in their normal working life.
   Investigators have a responsibility to protect participants from physical and mental harm during the investigation. The risk of harm must be no greater than in ordinary life. Areas of potential risk that require ethical approval include, but are not limited to, investigations that occur outside usual laboratory areas, or that require participant mobility (e.g. walking, running, use of public transport), unusual or repetitive activity or movement, that use sensory deprivation (e.g. ear plugs or blindfolds), bright or flashing lights, loud or disorienting noises, smell, taste, vibration, or force feedback

2. The experimental materials were paper-based, or comprised software running on standard hardware.
   Participants should not be exposed to any risks associated with the use of non-standard equipment: anything other than pen-and-paper, standard PCs, mobile phones, and PDAs is considered non-standard.

3. All participants explicitly stated that they agreed to take part, and that their data could be used in the project.
   If the results of the evaluation are likely to be used beyond the term of the project (for example, the software is to be deployed, or the data is to be published), then signed consent is necessary. A separate consent form should be signed by each participant.

   Otherwise, verbal consent is sufficient, and should be explicitly requested in the introductory script.

4. No incentives were offered to the participants.
   The payment of participants must not be used to induce them to risk harm beyond that which they risk without payment in their normal lifestyle.

5. No information about the evaluation or materials was intentionally withheld from the participants.
   Withholding information or misleading participants is unacceptable if participants are likely to object or show unease when debriefed.

6. No participant was under the age of 16.
   Parental consent is required for participants under the age of 16.

7. No participant has an impairment that may limit their understanding or communication.
   Additional consent is required for participants with impairments.

8. Neither I nor my supervisor is in a position of authority or influence over any of the participants.
   A position of authority or influence over any participant must not be allowed to pressurise participants to take part in, or remain in, any experiment.

9. All participants were informed that they could withdraw at any time.
   All participants have the right to withdraw at any time during the investigation. They should be told this in the introductory script.

10. All participants have been informed of my contact details.
    All participants must be able to contact the investigator after the investigation. They should be given the details of both student and module co-ordinator or supervisor as part of the debriefing.

11. The evaluation was discussed with all the participants at the end of the session, and all participants had the opportunity to ask questions.
    The student must provide the participants with sufficient information in the debriefing to enable them to understand the nature of the investigation.

12. All the data collected from the participants is stored in an anonymous form.
    All participant data (hard-copy and soft-copy) should be stored securely, and in anonymous form.

Project title Explaining Recommender Systems

Student's Name Quentin Deligny

Student's Registration Number 2262655D

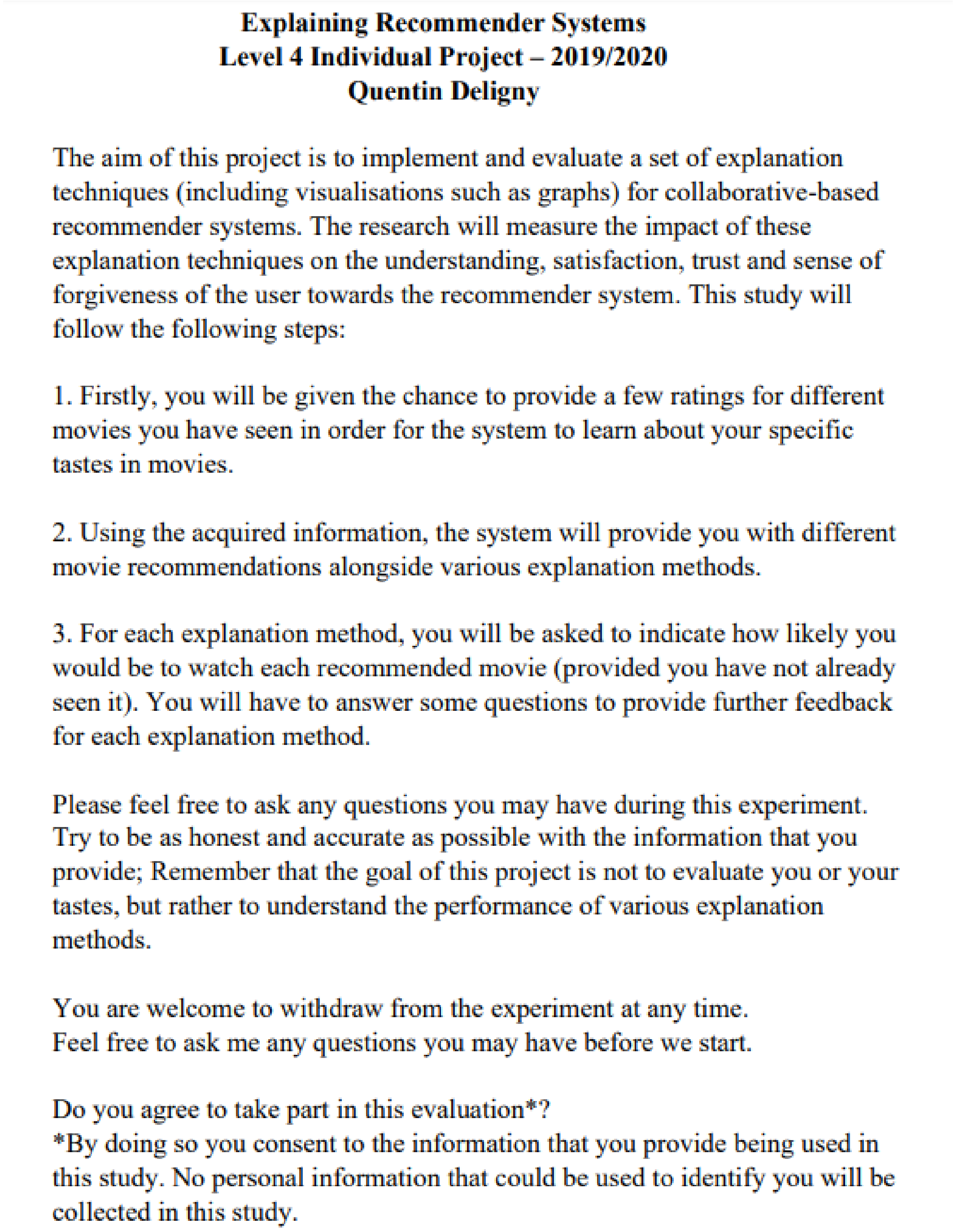
Student's Signature QD

Supervisor's Signature C. Macdonald

Date 01/04/2020

D Interview Structure

D.1 Introduction Script



Explaining Recommender Systems
Level 4 Individual Project – 2019/2020
Quentin Deligny

The aim of this project is to implement and evaluate a set of explanation techniques (including visualisations such as graphs) for collaborative-based recommender systems. The research will measure the impact of these explanation techniques on the understanding, satisfaction, trust and sense of forgiveness of the user towards the recommender system. This study will follow the following steps:

1. Firstly, you will be given the chance to provide a few ratings for different movies you have seen in order for the system to learn about your specific tastes in movies.

2. Using the acquired information, the system will provide you with different movie recommendations alongside various explanation methods.

3. For each explanation method, you will be asked to indicate how likely you would be to watch each recommended movie (provided you have not already seen it). You will have to answer some questions to provide further feedback for each explanation method.

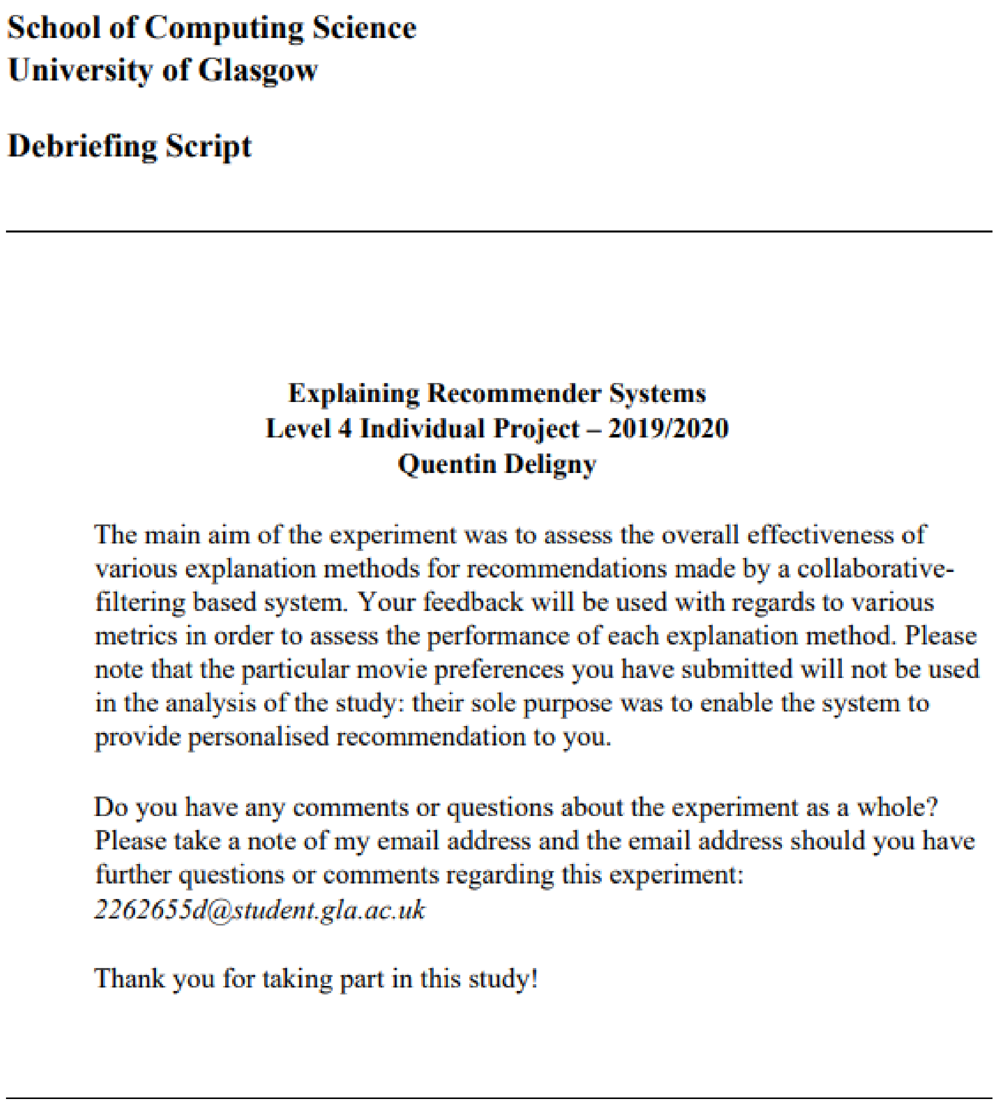
Please feel free to ask any questions you may have during this experiment. Try to be as honest and accurate as possible with the information that you provide; Remember that the goal of this project is not to evaluate you or your tastes, but rather to understand the performance of various explanation methods.

You are welcome to withdraw from the experiment at any time.
Feel free to ask me any questions you may have before we start.

Do you agree to take part in this evaluation*?
*By doing so you consent to the information that you provide being used in this study. No personal information that could be used to identify you will be collected in this study.

Figure D.1: A copy of the Introduction Script presented to the participants at the start of the interview.

D.2 Debrief Script



School of Computing Science
University of Glasgow

Debriefing Script

Explaining Recommender Systems
Level 4 Individual Project – 2019/2020
Quentin Deligny

The main aim of the experiment was to assess the overall effectiveness of various explanation methods for recommendations made by a collaborative-filtering based system. Your feedback will be used with regards to various metrics in order to assess the performance of each explanation method. Please note that the particular movie preferences you have submitted will not be used in the analysis of the study: their sole purpose was to enable the system to provide personalised recommendation to you.

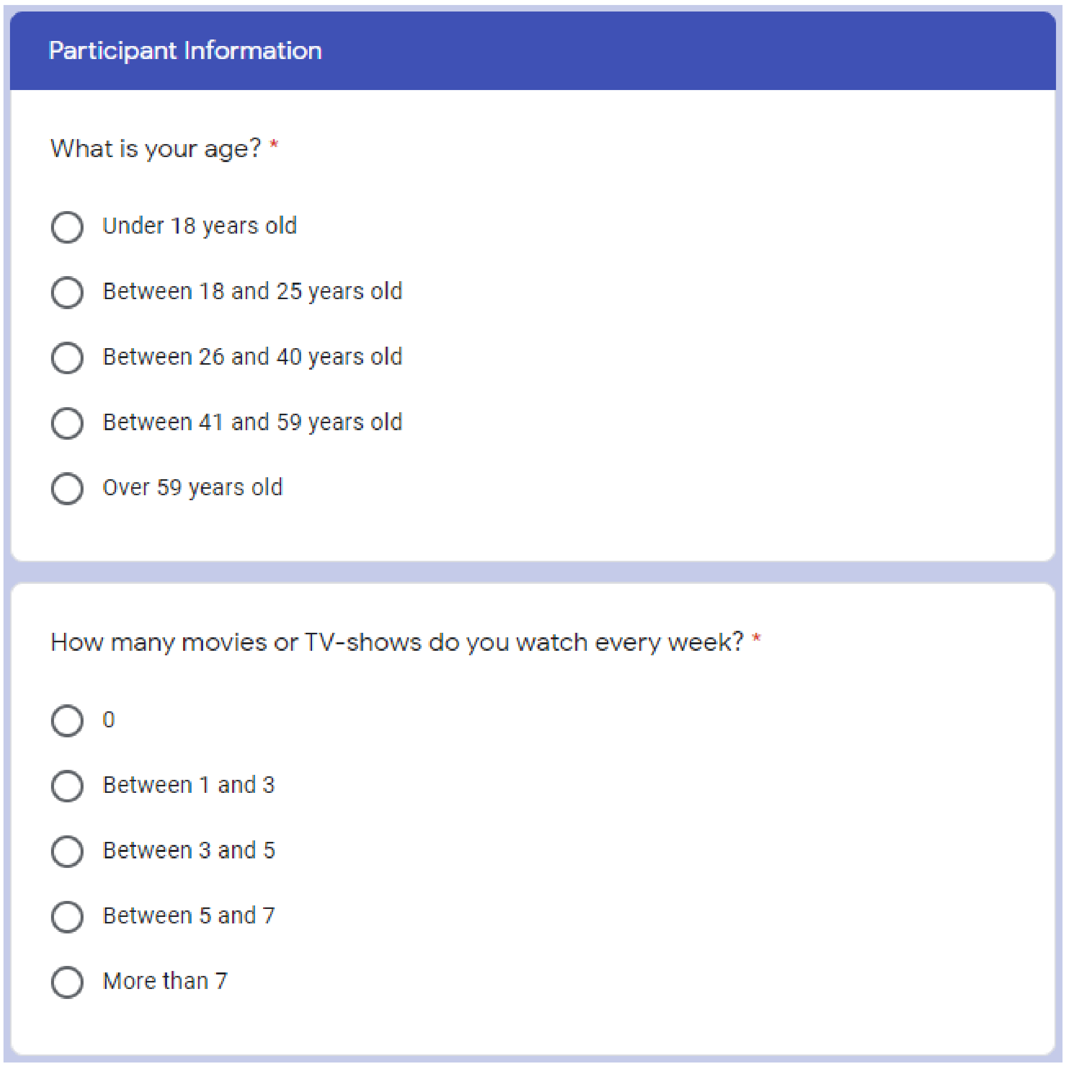
Do you have any comments or questions about the experiment as a whole? Please take a note of my email address and the email address should you have further questions or comments regarding this experiment: *2262655d@student.gla.ac.uk*

Thank you for taking part in this study!

Figure D.2: *A copy of the Debrief Script presented to the participants at the end of the interview.*

D.3 Questionnaire

The following figures show the questions which were asked to the participant during this user study.



Participant Information

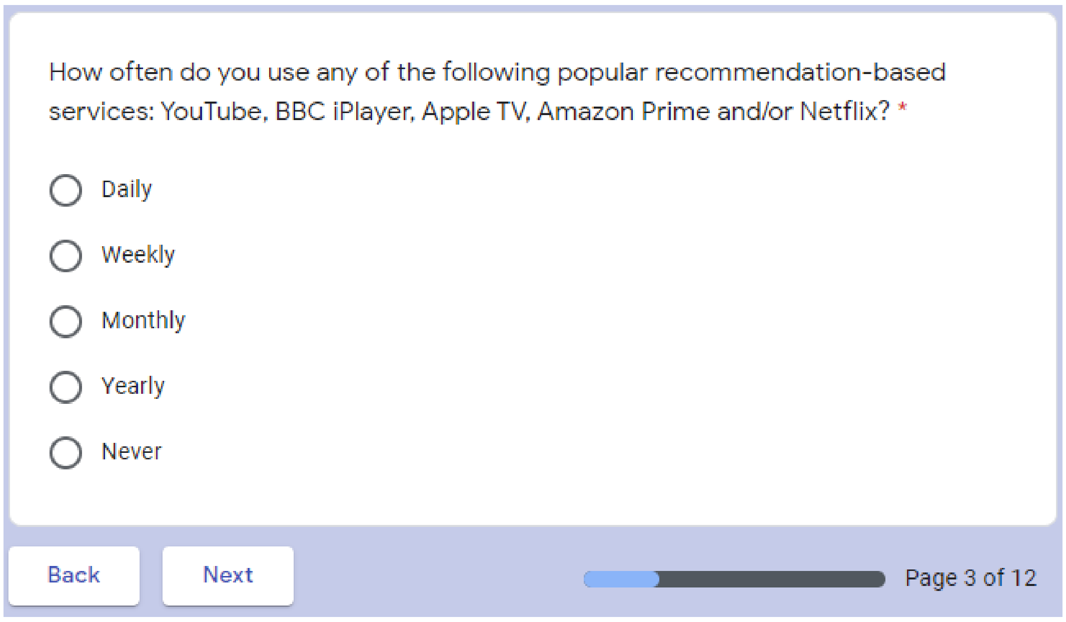
What is your age? *

- ○ Under 18 years old
- ○ Between 18 and 25 years old
- ○ Between 26 and 40 years old
- ○ Between 41 and 59 years old
- ○ Over 59 years old

How many movies or TV-shows do you watch every week? *

- ○ 0
- ○ Between 1 and 3
- ○ Between 3 and 5
- ○ Between 5 and 7
- ○ More than 7

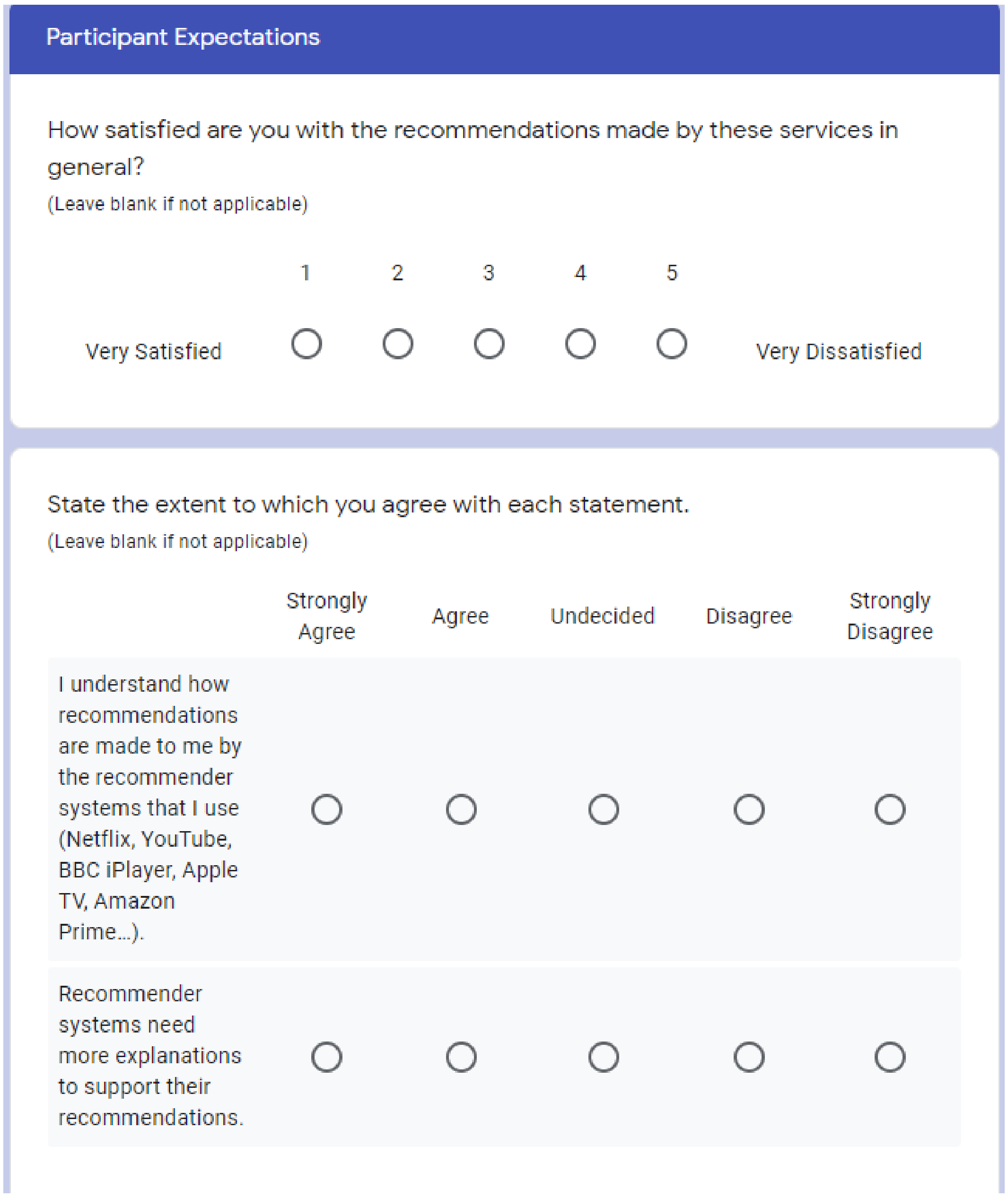
Figure D.3: *First set of questions to gather data regarding the participant and his level of familiarity with recommender systems.*



How often do you use any of the following popular recommendation-based services: YouTube, BBC iPlayer, Apple TV, Amazon Prime and/or Netflix? *

- ○ Daily
- ○ Weekly
- ○ Monthly
- ○ Yearly
- ○ Never

Figure D.4: *Second set of questions to gather data regarding the participant and his level of familiarity with recommender systems.*



Participant Expectations

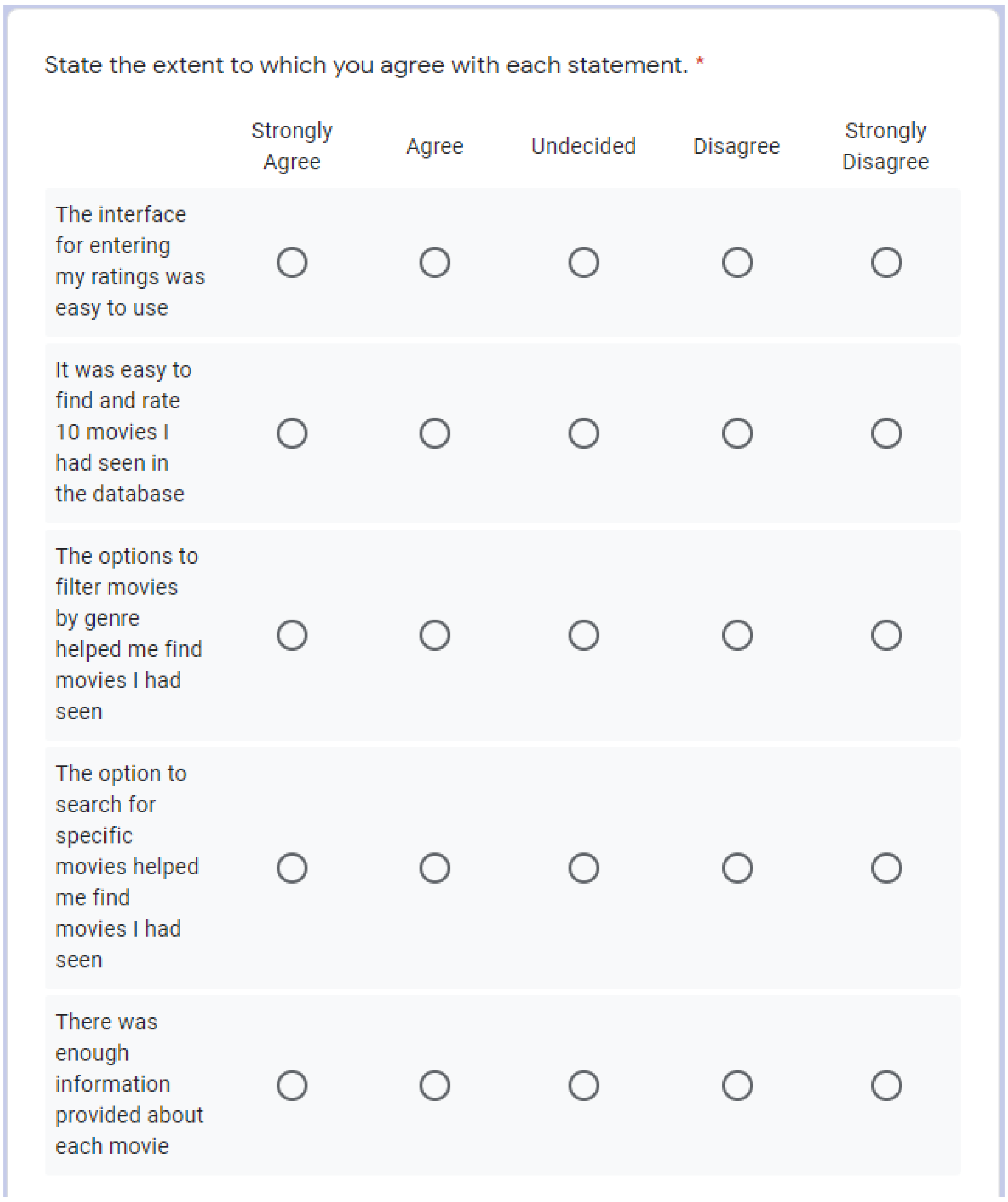
How satisfied are you with the recommendations made by these services in general?
(Leave blank if not applicable)

	1	2	3	4	5	
Very Satisfied	○	○	○	○	○	Very Dissatisfied

State the extent to which you agree with each statement.
(Leave blank if not applicable)

	Strongly Agree	Agree	Undecided	Disagree	Strongly Disagree
I understand how recommendations are made to me by the recommender systems that I use (Netflix, YouTube, BBC iPlayer, Apple TV, Amazon Prime...).	○	○	○	○	○
Recommender systems need more explanations to support their recommendations.	○	○	○	○	○

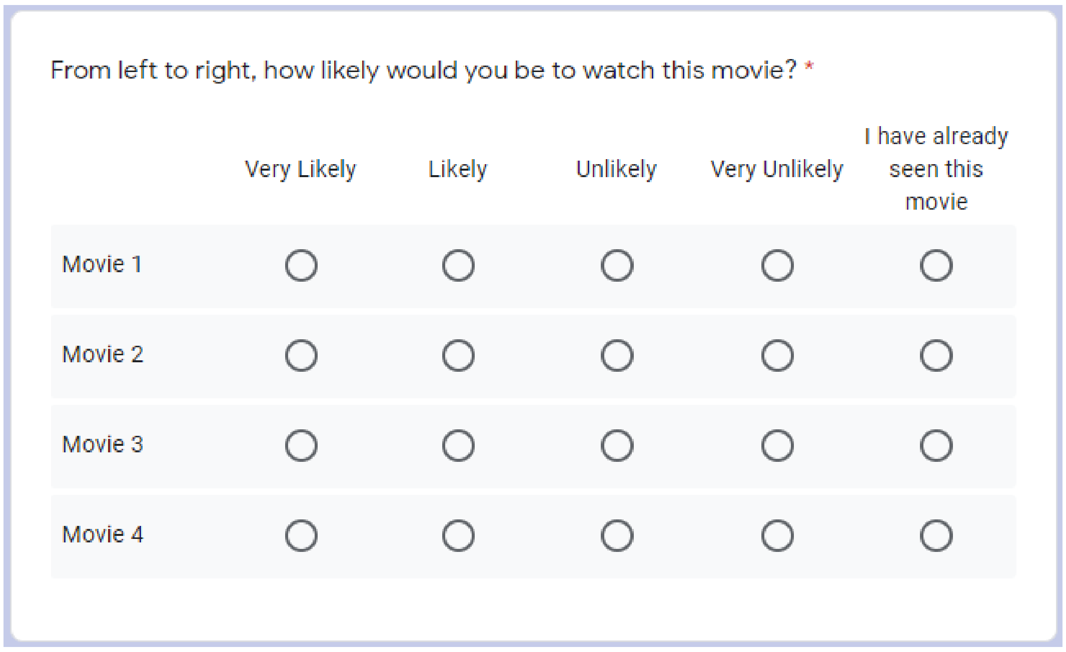
Figure D.5: *Third set of questions to gather data regarding the participant's general opinions of recommender systems.*



State the extent to which you agree with each statement. *

	Strongly Agree	Agree	Undecided	Disagree	Strongly Disagree
The interface for entering my ratings was easy to use	○	○	○	○	○
It was easy to find and rate 10 movies I had seen in the database	○	○	○	○	○
The options to filter movies by genre helped me find movies I had seen	○	○	○	○	○
The option to search for specific movies helped me find movies I had seen	○	○	○	○	○
There was enough information provided about each movie	○	○	○	○	○

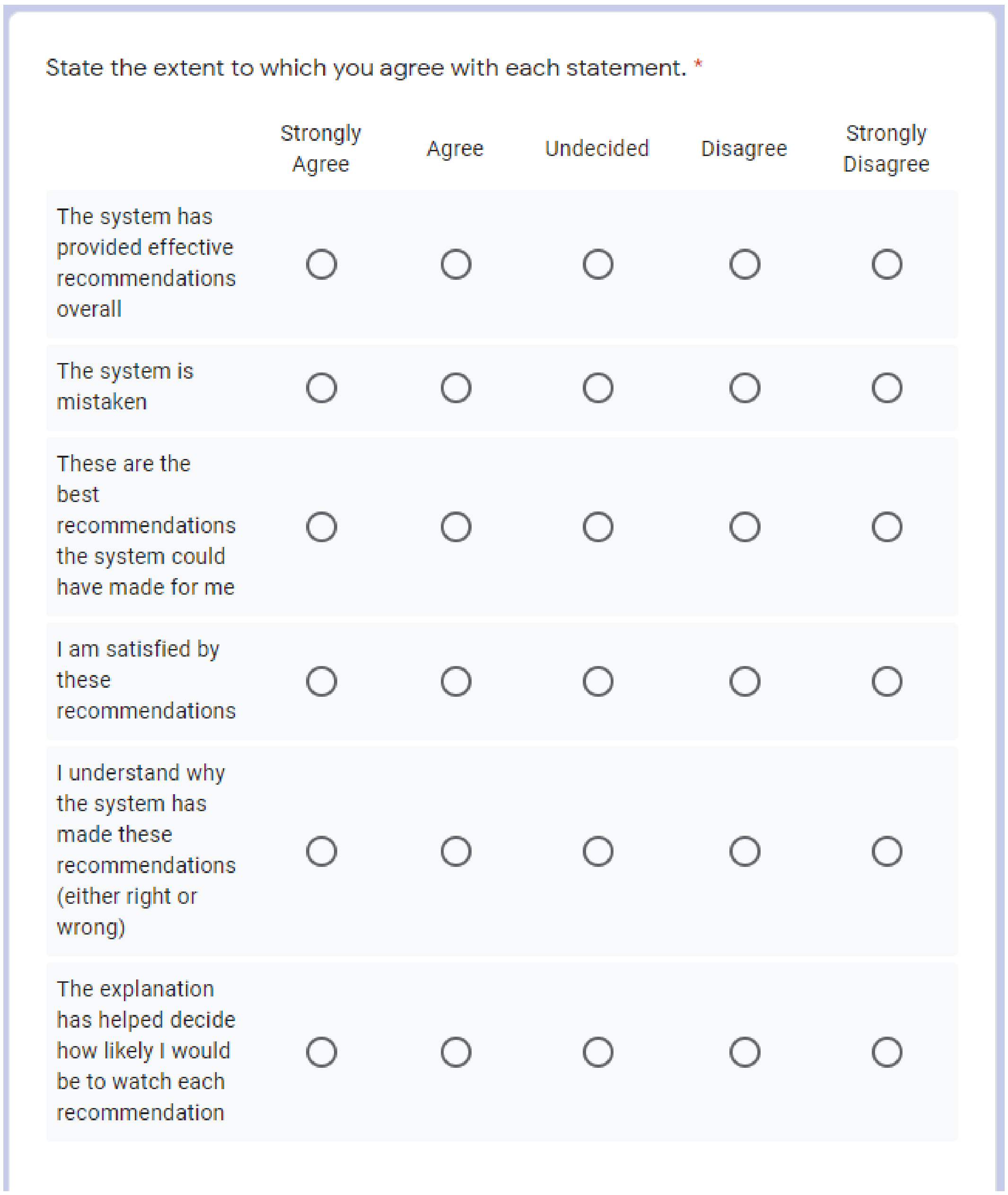
Figure D.6: *Fourth set of questions to allow the participant to provide some feedback regarding the interface used to provide movie ratings.*



From left to right, how likely would you be to watch this movie? *

	Very Likely	Likely	Unlikely	Very Unlikely	I have already seen this movie
Movie 1	○	○	○	○	○
Movie 2	○	○	○	○	○
Movie 3	○	○	○	○	○
Movie 4	○	○	○	○	○

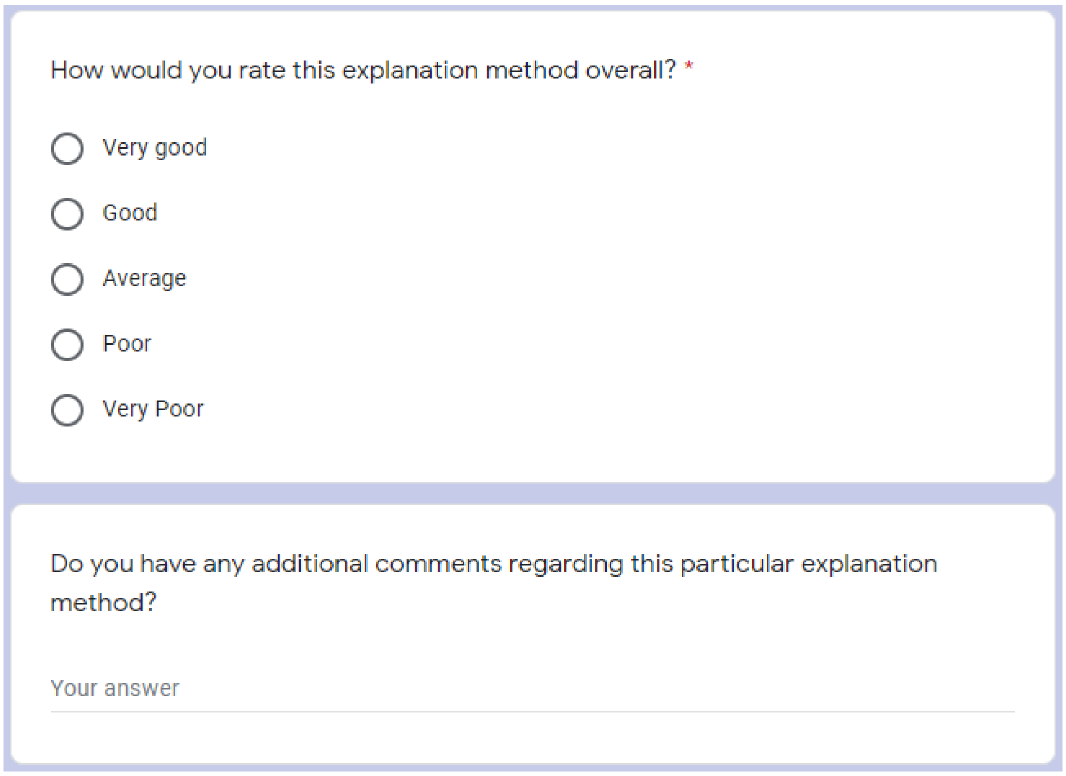
Figure D.7: *The fifth set of questions measured the persuasiveness of the recommendations. This was asked for each explanation method as well as for recommendations with no explanation.*



State the extent to which you agree with each statement. *

	Strongly Agree	Agree	Undecided	Disagree	Strongly Disagree
The system has provided effective recommendations overall	○	○	○	○	○
The system is mistaken	○	○	○	○	○
These are the best recommendations the system could have made for me	○	○	○	○	○
I am satisfied by these recommendations	○	○	○	○	○
I understand why the system has made these recommendations (either right or wrong)	○	○	○	○	○
The explanation has helped decide how likely I would be to watch each recommendation	○	○	○	○	○

Figure D.8: *The sixth set of questions measured the remaining metrics. This was asked for each explanation method as well as for recommendations with no explanation (aside from the last question measuring helpfulness which was not asked for the baseline).*


How would you rate this explanation method overall? *

- ○ Very good
- ○ Good
- ○ Average
- ○ Poor
- ○ Very Poor

Do you have any additional comments regarding this particular explanation method?

Your answer

Figure D.9: *The seventh set of questions measured the overall performance of each explanation method and allowed for further comments from the user. This was asked for each explanation method (not the baseline).*

E | Background Information

Appendix E.1 provides the required formulae to compute the *tf-idf* of a document.

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{E.1}$$

Where:

$$tf(t, d) = \log(1 + freq(t, d)) \tag{E.2}$$

$$idf(t, D) = \log\left(\frac{N}{\text{count}(d \in D : t \in d)}\right) \tag{E.3}$$

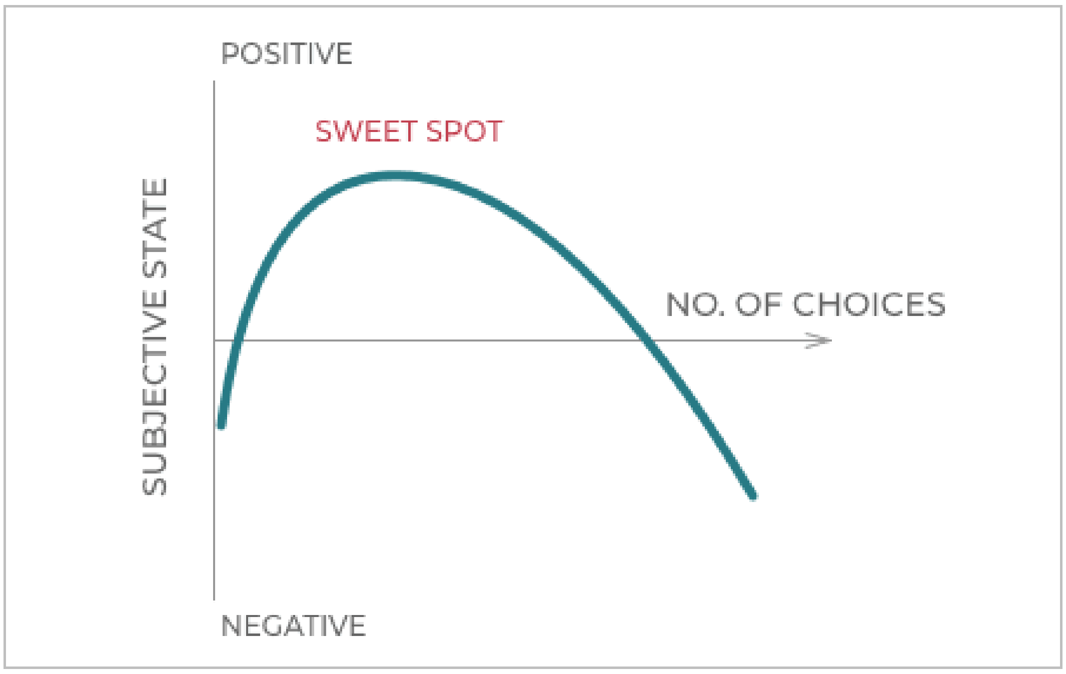


Figure E.1: *Example graph from Neurofied*
illustrating how having too many choices can be detrimental to the user.

Bibliography

Adomavicius, G. and Zhang, J. (2012), 'Impact of data characteristics on recommender systems performance', *ACM Transactions on Managements Information Systems* .

Antaris, S., Karagiannidis, S. and Zisopoulos, H. (2008), 'Content-based recommendation systems', *ResearchGate* .

Basu, C., Hirsh, H. and Cohen, W. (1998), Recommendation as classification: Using social and content-based information in recommendation, *in* 'AAAI-98 Proceedings'.

Bilgic, M. and Mooney, R. (2005), Explaining recommendations: Satisfaction vs. promotion, *in* 'International Conference on Intelligent User Interfaces'.

Daher, J., Brun, A. and Boyer, A. (2018), 'A review on explanations in recommender systems', *HAL* .

Gediklia, F., Jannacha, D. and Geb, M. (2014), 'How should i explain? a comparison of dierent explanation types for recommender systems', *IJHCS* .

Hameed, M. A., Jadaan, O. A. and S, R. (2012), 'Collaborative filtering based recommendation system: A survey', *IJCSE* .

Haruna, K., Akmar Ismail, M., Damiasih, D., Sutopo, J. and Herawan, T. (2017), 'A collaborative approach for research paper recommender system', *PLoS ONE* .

Herlocker, J. L., Konstan, J. A. and Riedl, J. (2000), Explaining collaborative filtering recommendations, *in* 'ACM Conference on Computer Supported Cooperative Work'.

Kotkov, D., Veijailenen, J. and Wang, S. (2016), 'A survey of serendipity in recommender systems', *Knowledge-Based Systems* .

Kula, M. (2017), 'Spotlight', `https://github.com/maciejkula/spotlight`.

Macdonald, C. (2020), 'Lecture notes in recommender systems h/m'.

Moghaddam, F. B. and Elahi, M. (2019), 'Cold start solutions for recommendation systems', *Research Gate* .

Schwartz, B. (2004), *The Paradox of Choice – Why More Is Less*, Harper Perennial.