Towards a new generation of movie recommender systems: A mood based approach

Author: Niklas Wietreck
Supervisor: Görkem Pacaci

A thesis submitted in fulfillment of the requirements for the degree of Master of Social Science in the Department of Informatics and Media

June 3, 2018
Declaration of Authorship

I, Niklas WiETRECK, declare that this thesis titled, “Towards a new generation of movie recommender systems: A mood based approach” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: June 3, 2018
Towards a new generation of movie recommender systems: A mood based approach

by Niklas Wietreck

The emergence of the content overloaded internet creates a lot of new challenges for users and service providers alike. To minimize the displayed amount of content like movies, music, or other products service providers like Netflix or Amazon are using recommender systems which aim to guide the user through the available information. These systems collect knowledge about the user and try to deliver personalized experiences. Most of the state-of-the-art recommender systems are using a content focused approach but often fail to grasp the nature of users’ desires. Therefore, a mood-as-input model is developed which combines the existing research on human mood identification and the emotion classification of content in the domain of movies. In order to match these two components different machine learning models are evaluated and a Random Forest is selected as the main matching algorithm. The results of this study indicate that the mood of a user can be used to create personalized content recommendations and that it can perform better than an Arbitrary system.

Keywords: Recommendation System, Movie Recommendation, User Modeling, Mood Detection, Machine Learning, Random Forest
Acknowledgements

I would like to express my very great appreciation to my supervisor Görkem Pacaci for his valuable and constructive suggestions during the whole development of my master thesis. His useful feedback and answers to all my questions was highly appreciated and helped to improve this thesis paper in many different ways.

I would also like to extend my thanks to Mr. Sourabh Bikash, Mr. Nikola Katardjiev and Mr. Karl Ericsson who always had an open ear for a technology related discussion and gave valuable suggestions for solving crucial challenges.

Additionally, I would like to thank Ms. Raya Rizk for the careful and thorough comments on this thesis report. Her opposition helped to improve the descriptive as well as the logical aspects of this document, that results in an improved reading experience.

Last but not least I would like to thank Ms. Beate Renner for prove reading this document several times and for the resulting worthwhile feedback.
## Contents

Declaration of Authorship iii  
Abstract v  
Acknowledgements vii  

1 Introduction                   1  
1.1 Background                     1  
1.2 Recommender Systems           3  
  1.2.1 Definition                   3  
  1.2.2 Types of recommender systems 4  
  1.2.3 Knowledge sources of a recommender system 6  
  1.2.4 Knowledge sources in a mood based movie recommendation system 7  
  1.2.5 Existing movie recommender systems 9  
1.3 Research Problem             11  
  1.3.1 Research Questions         12  
  1.3.2 Hypotheses                 12  
1.4 Importance of the research   13  
1.5 Outline                      13  

2 Method                        15  
2.1 General Approach             15  
  2.1.1 Sample size                16  
2.2 The Arbitrary Recommender System 17  
  2.2.1 Input knowledge source     18  
  2.2.2 Output knowledge source    18  
  2.2.3 Matching logic             20  
  2.2.4 Data collection process    20  
2.3 Recommender data model       21  
  2.3.1 Random Forest              22  
  2.3.2 Artificial Neural Network  22  
  2.3.3 Support Vector Machine     23  
  2.3.4 k-Nearest Neighbor Algorithm 24  
  2.3.5 Naive Bayes Classifier     24  
  2.3.6 Data Model Comparison      25  
2.4 The Mood Based Recommender System 25  
  2.4.1 Input knowledge source     26  
  2.4.2 Output knowledge source    26  
  2.4.3 Matching logic             26  
  2.4.4 Data collection process    27  
2.5 Data Analysis and Coding     27  
  2.5.1 Frequency Analysis         27
## List of Figures

1. Development process of the system .................................................. 16
2. Schematic figure of the *Arbitrary* recommender system ...................... 18
3. Schematic figure of the *Mood Based* recommender system .................. 26

2.1 Frequency analysis *Arbitrary* Recommender System: Bar chart representation ................................................................. 32
2.2 Frequency analysis *Arbitrary* Recommender System: Pie chart representation ................................................................. 33
2.3 Frequency analysis *Mood Based* Recommender System: Bar chart representation ................................................................. 33
2.4 Frequency analysis *Mood Based* Recommender System: Pie chart representation ................................................................. 34
3.5 Frequency comparison between the two recommender systems ............. 35
3.6 t-Test Overview ............................................................................. 36
3.7 t-Test result ................................................................................... 36

A.1 Website PANAS-Questionnaire ......................................................... 51
A.2 Website Suggestion ......................................................................... 52
A.3 Website DataLayer .......................................................................... 53
A.4 Google Analytics Custom Reports ................................................... 53
List of Tables

1.1 The PANA-Scale ....................................................... 8
2.1 Example set of movies with their genre label configuration .......... 20
2.2 Accuracy comparison of data model .................................. 25
3.1 Success Rate in steps cumulative ..................................... 34
A.1 All sets of movies with their genre label configuration ................. 57
Chapter 1

Introduction

This chapter serves as the starting point for the thesis and establishes the key concepts and vocabulary for the rest of this master thesis. Followed by this quick introduction the background and a general introduction to recommender systems is presented. After the research problem and the research questions are identified, the hypotheses for the thesis are formulated. A section about the importance of this research as well as a quick outline for the rest of the paper will conclude the first chapter.

1.1 Background

Today's society faces a lot of new problems mainly due to the rapid advancements of information technology over the last decade. The new technologies gave almost everybody who is connected to the internet access to a mass of information and different types of content. Different types of news and media content is nowadays accessible to almost everybody. Online research libraries like google scholar, music streaming services like Spotify or companies who deliver movie on-demand services like Netflix are just a few examples of services which appeared since the end of the last century. All of these services have one major thing in common: they possess and share a vast amount of content.

Since new information and content is added nearly all the time, a huge amount of information and content is created for the user. This results in several challenges for the user like how to find the right information and content or how to navigate through the ever growing data jungle. To support the user with this challenge, a lot of services have implemented support systems which aim to guide the users through the process of finding the right content. Theses systems aim to identify what a user actually needs and transform this knowledge into the right content. Such systems are called recommender systems (Burke, 2002).

At the early times of the internet, information and content were just stored and accessed. Since there were only a couple of users, it was quite easy to share the location of a specific document. Also, the amount of the stored information was so low that a user might even browse through the text files to find the right documents (Burke, 2002). However, due to the rapid growth in popularity of the internet and with an increasing number of users new methods had to be developed. Search engines already helped users a lot by showing results based on search queries, a user typed into the system. This increased the browsing experience and usability of the internet in general immensely (Leiner et al., 2009). But the ever-increasing number of content soon had to deal with the fact that a search query might result in hundreds of different items. While search engines solved these issues by using different methods like sentiment analysis or authority scores, other more product focused services had to find different solutions. They added features that gave the user additional
information about a specific product or content. At the beginning, these included mainly reviews of other users in form of short texts or comments (Leiner et al., 1997; Ricci et al., 2011; Resnick and Varian, 1997).

But through the immense progress in the field of information technologies, new support systems have been developed. These systems can now perform more and more advanced tasks and analyses, and deliver personalized content recommendations for each user. The first generation of recommender systems targeted to analyze the interests of a unique user (Ricci et al., 2011) and matching them against the different labeled categories within the available content. This approach is nowadays used in a wide range of scenarios. The application cases range from recommending a specific movie on platforms like Netflix, delivering specific types of music to listen to on Spotify and to advise the user on what products to buy on Amazon. But do the current systems really meet the requirements of the user and are these systems actually able to deliver reasonable recommendations? Or can these recommender systems be designed in an even more user centered way?

The state-of-the-art of most recommender systems nowadays is that they are based on the consumption and search history of the user. The consumption history can be seen as the earlier accessed content. After analyzing this content, it can be linked to other types of content that share the same characteristics or labels (Tkalčič et al., 2016). Another widely used approach looks at the users history and compares it to the history of other users. Users with similar interests are grouped and content that one user accessed is suggested to the other users in that group. Therefore, the user will get a recommendation based on what another user with the same interests has consumed before. These two concepts, called content-based and collaborative filtering methods, are often combined and will be further explained later in this chapter (Burke, 2002). As an example for this state-of-the-art recommender system, I will take a look at how Netflix recommends movies to a viewer. On their main landing page they present an assemblage of movies and TV series mainly based on what the user has watched before or what related user groups are watching. For example, if I watch "Sherlock", a TV series about the sociopath, deduction mastermind, and criminal detective Sherlock Holmes, the system will automatically recommend series like "Elementary" or "The Mentalist" that both deal with the same kind of content, based on their description (Ampazis, 2010; Resnick and Varian, 1997).

These two approaches explained in the previous paragraph can be seen as a milestone in the way content is shared with the users and might even deliver personal recommendations to each user. But those recommendations are not necessarily personalized. They both follow the underlying concepts that interests and needs of users are a relative steady variable. A steady and only slowly changing variable that represents the needs and desires of the user. (Ullah, Sarwar, and Lee, 2014). But psychological research has shown that the emotional state of a human being can change on a monthly, daily or even hourly basis due to several influence factors. With the change of the emotional state also comes a change in needs and interests (Jarymowicz, 2012; Schwarz and Clore, 1983). Therefore, it can be argued if these current state-of-the-art systems are actually capable of giving accurate and personalized recommendations.

Considering these changes of the desires and interests of a human being, a recommender system should also include the current emotional state of a user to create better recommendations and to meet the users’ needs. If a user has watched a
lot of intellectually demanding drama movies in the past he or she \(^1\) will receive a recommendation based on this history. But he might currently be, for whatever reason, more interested in a lightweight comedy movie. Therefore, figuring out what a user really wants and needs in a specific moment might be the next step to a more advanced and really user centered recommender system which will finally lead to really personalized content (Ullah, Sarwar, and Lee, 2014).

Since this is a general problem for all types of recommendation systems, it seems especially crucial when it comes to the selection of a fitting movie. Today’s online on-demand video services offer several hundreds of movies which all can be accessed in a matter of just a few clicks. To browse through all available options and making a correct choice can be a quite excessive task. Also, it seems that a two-hour movie is quite of a commitment if compared against a three-minute audio song. Therefore, a suggestion should be a perfect fit in order to create a better user experience and higher user satisfaction. Due to these reasons, the context of movies was selected to showcase how a mood based recommender system can look like.

To understand how a mood based movie recommender system can be developed, it’s important to understand what types of recommender systems exist and how they are structured and implemented. All the different components (knowledge sources, input and output factors etc.) are explained in the following section.

### 1.2 Recommender Systems

In this section, a brief summary is given on what is known about recommender systems, what types exist and what components belong to it. I start this by giving a brief description and definition of a general recommender system and then present the known problems. In order to be able to evaluate these approaches properly it’s important to understand the different structures and taxonomies of a general recommender system. The focus lies hereby on the different types of recommender systems and the identification of a suitable solution for our problem. As well, I take a look at the different steps that must be deployed in order to implement a well working recommender system. Afterwards, examples of existing mood based recommender systems will be used to show what methods have been used already, and how they perform. By doing this, I would like to understand which analysis methods were used and if they can also be applied to our scenario.

To select the best fitting type and structure of the recommender system it is important to understand the theory behind the two different knowledge sources (mood and movie classification) that I am aiming to map. Namely, the theories behind the mood of an individual and the theories behind the classification of movies. When it comes to the mood of an individual I first take a look at the differences between mood, emotions and affect before a standardized method of mood analysis and extraction is presented. Regarding the classification of movies, traditional genre classifications as well as modern emotion optimized classifications are taken into consideration.

#### 1.2.1 Definition

At the very beginning it is necessary to understand what a recommender system is, what types of recommender systems exist, and what components are included.

\(^1\)In the following I will only use the male form to make this thesis easier to read. But it is important to note that I consider male and female persons in the upcoming scenarios.
Recommender systems emerged as an individual field of research in the mid-1990s and derived from different other research areas like cognitive science, approximation theory, information retrieval, forecasting theory, consumer modeling, and also management science (Adomavicius and Tuzhilin, 2005). Resnick and Varian (1997) describe recommender systems as:

“People provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients.” (Resnick and Varian, 1997)

While this seems to be a suitable definition for the early recommender system in the late 90s, recommender systems have developed a lot since then. Nowadays, the term can be described in a broader way as:

“Any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options.” (Burke, 2002)

Recommender systems are aiming to deliver these individualized outputs by learning about the distinct user preferences and then anticipating their unique needs (Campos et al., 2010). Due to different requirements there exist several types or techniques of recommender systems, that all differ in the way a specific recommendation is given. It depends on the scenario what type of system is selected and applied. In the following the most common types are explained in detail (Blanco-Fernández et al., 2008).

1.2.2 Types of recommender systems

There exist several different types of recommender systems that can mainly be divided into five different groups that are described in the following section (Burke and Ramezani, 2011; Ricci et al., 2011).

- **Collaborative recommender systems**

  Collaborative recommender systems are, next to content-based recommender systems, one of the most common used systems. In general, these systems create a user profile based on ratings of different objects and then compare these against a wider user group. The system recognizes similarities between users based on their ratings and then creates new recommendations based on this inter-user comparison. Collaborative recommender systems can differ in the way a rating is defined. They can be binary, focused on shift in opinion over time, model or memory-based (Burke, 2002). A common example are the additional product suggestions on amazon (People who bought .... also bought ...).

- **Content-based recommender systems**

  Different filtering methods are based on the attributes of the content in order to create an individualized recommendation. The content is classified through different tags or features, and can link these to the ratings of the users. By doing this, the systems learn from the user profile and displays appropriate recommendations (Burke, 2002). A common example for this recommender system are the movie suggestions on Netflix. If the user watched a movie of the type action once or even gave positive feedback (through votings or comments) for that movie, the user will get suggestions with the same label. How the content is actually labeled and matched against each other is often kept secret by the different service providers.
• **Demographic recommender systems**

  The classification of different demographic attributes builds the baseline of these types of recommender systems. These attributes can include age, gender, cultural background or other personal characteristics. The derived model of all combined personal attributes is then matched against a catalog of user stereotypes that was manually created. In contrast to, for example collaborative recommender systems, a history of different ratings is not required (Burke, 2002). Demographic recommender systems can be used in a wide range of applications but especially helps to solve the *new user problem*. This problem appears when a model of a user does not exist yet e.g. when the user utilizes a recommender system for the first time. Since the interests of users are unknown, suggestions can only be made based on the available demographic information of the existing users (Lika, Kolomvatsos, and Hadjieftymiades, 2014).

• **Utility-based recommender systems**

  These systems are trying to calculate the utility of each object in relation to the user. In the next step they attempt to increase this utility factor to meet the personal needs of each user. Therefore, a long-term evaluation of the user through ratings is not required since the system gets to know the user over time. The main problem hereby is to define this usability function for each user which is often done through different user satisfaction techniques (Burke, 2002).

• **Knowledge-based recommender systems**

  In a knowledge-based recommender system there exists underlying information about the relation between the needs of a user and a specific item. This inference between user needs and a specific item, that characterizes all recommendation types, can again then be stored in a user profile (Burke, 2002; Blanco-Fernández et al., 2008). Basically all kinds of expert systems can be considered a knowledge-based recommender systems, since the value of the system is created through the acquisition of the knowledge of the user (Carrer-Neto et al., 2012).

**Hybrid Techniques**

All the above mentioned types have their strengths and weaknesses. In order to overcome the weaknesses, hybrid techniques, that combine two or more types, have been developed in order to meet the requirements of the different scenarios and to increase performance. In general the hybrid techniques are especially used to solve the *new user problem* (Ricci et al., 2011).

• **Weighted recommender**

  The score of different recommender systems are combined into one single individual recommendation. Before combining them, the scores are weighted according to their influence on a specific item (Burke, 2002).

• **Switching recommender**

  Depending on the scenario, the recommender system switches between different types of recommender systems and uses the one most suitable for the given problem (Burke, 2002).
• **Mixed recommender**
  Different types of recommender systems are deployed in parallel and results are presented at the same time. This gives the user all possible suggestions that are derived based on the different methodologies (Burke, 2002).

• **Feature combined recommender**
  Different features from a variety of data sources are pulled together to create a single recommendation algorithm. This features can include for example demographic knowledge as well as the related and predefined utility factors (Burke, 2002).

• **Cascade recommender**
  The outcome of one recommendation system is refined by another recommendation system. This more sequential model aims to optimize the results of one recommendation technique with another. The techniques can be freely selected based on the domain (Burke, 2002).

• **Feature Augmentation recommender**
  This technique also operates sequentially. The output of one recommendation type is used as the input feature of another type of recommendation system. The difference is that in feature augmentation new features can be added in the second attached system (Burke, 2002).

• **Meta-level recommender**
  The meta level recommender uses the model learned by one recommender as an input for another type of recommendation system. Therefore, the focus is on understanding the meta structure of a problem before deploying it (Burke, 2002).

### 1.2.3 Knowledge sources of a recommender system

As explained in the previous paragraph a type of recommender system is selected based on the specific scenario. This scenario categorizes mainly through the different sources of knowledge, that are necessary as input variables for the recommender system. These knowledge sources, especially in a knowledge-based recommender, can be divided into three different groups (Burke, 2002):

• **Catalog knowledge**
  Deep knowledge about the item that is recommended. Depending on the domain this catalog can be quite complex and excessive, since this information is presented as features, attributes, labels, or tags (Burke, 2002).

• **Functional knowledge**
  The system must be able to match the user’s needs with a specific item. This knowledge about the matching process that includes the type of algorithms or methodologies has to be stored (Burke, 2002).

• **User knowledge**
  To create suitable recommendations the system requires detailed information about the user that can be seen as a user model (McTear, 1993). This can include the user’s demographic or preference information. This knowledge can be the
most crucial one since it can create a user-modeling problem (Jawaheer, Weller, and Kostkova, 2014).

The user-modeling problem appears in all kinds of recommender systems, when the tool tries to get a detailed understanding of the user. Since the human being behind the user has a variety of interests, desires and needs, it is difficult for the system to gather all this information. Therefore, there is always a high chance that the recommender system is not capable of representing the user completely accurate (Burke, 2002).

1.2.4 Knowledge sources in a mood based movie recommendation system

The research on recommender systems has shown that a deep understanding of the knowledge sources are required, to be able to select the right type of system or technique. Therefore, I take a more detailed look into two of the three knowledge sources which will be at the core of this thesis. On the one hand the theory behind the classification of human mood (user knowledge) is presented before I look at the classification of movies (catalog knowledge). The functional knowledge element will be described in detail in the method section.

Mood classification theory

When talking about moods it is necessary to understand the differences to emotions and affect. Psychological research differentiates them by defining emotions as intense feelings directed at someone or something, while moods are less intense feelings and do not require a specific stimulus. Affect is an even broader concept of perceived feelings which combines emotions and moods (Robbins and Judge, 2013). Several efforts were made in order to classify moods into different dimensions (Staw and Cummings, 1996). In general, it was identified that mood comes either in a positive affect or in a negative affect dimension. Positive affect mainly reflects on the level on “which a person feels enthusiastic, active, and alert” (Watson, Clark, and Tellegen, 1988) while “negative aspect is a general dimension of subjective distress and unpleasant engagement that subsumes a variety of aversive mood states” (Watson, Clark, and Tellegen, 1988).

Each of these two dimensions has been further categorized into a 10-items list that reflects on a specific mood state. This well studied and often reused scale is called PANAS (Positive Affect - Negative Affect Scale) and contains the items displayed in Table 1.1 (Watson, Clark, and Tellegen, 1988).

When analyzing the mood of a person using the PANAS each user rates each item on a scale from 1 (very slightly or not at all) to 5 (extremely) (Staw and Cummings, 1996). The given answers are combined to a score that represents the current mood of the user. Since this calculated score removes quite a number of insights (e.g. dependencies within the questionnaire) it can be discussed if all individual answers have to be taken into consideration (Watson, Clark, and Tellegen, 1988; Raghunathan and Pham, 1999).
Table 1.1: The PANA-Scale

<table>
<thead>
<tr>
<th>Positive Affect</th>
<th>Negative Affect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enthusiastic</td>
<td>Scared</td>
</tr>
<tr>
<td>Interested</td>
<td>Afraid</td>
</tr>
<tr>
<td>Determined</td>
<td>Upset</td>
</tr>
<tr>
<td>Excited</td>
<td>Distressed</td>
</tr>
<tr>
<td>Inspired</td>
<td>Jittery</td>
</tr>
<tr>
<td>Alert</td>
<td>Nervous</td>
</tr>
<tr>
<td>Active</td>
<td>Ashamed</td>
</tr>
<tr>
<td>Strong</td>
<td>Guilty</td>
</tr>
<tr>
<td>Proud</td>
<td>Irritable</td>
</tr>
<tr>
<td>Attentive</td>
<td>Hostile</td>
</tr>
</tbody>
</table>

Movie classification theory

A movie is generally labeled with different types of information. These information includes for example the title, year of release, description of the plot and actors. One of the most common labels used in online on-demand movie services is the genre which groups the movies into different categories depending on their content. The popular online movie database IMdB names 24 different unique genres (for example action, thriller, drama), but most of the movies are labeled with more than just one genre. One reason for this is that movies as type of media content are varying a lot from each other even within a genre or are dealing with themes from several genres. The film milestone Star Wars 4: A new hope from the year 1977 for example is listed in the IMdB database with three distinct genres (action, adventure, fantasy). Even though filmmakers and movie fans a-like try to give as much information about a movie as possible, there still exist a lack of detail in this community driven database. This shortcoming can be identified especially when a user wants to know about the tone, mood or atmosphere of a specific movie. Torben Grodal, a professor emeritus of film and media studies at the university of Copenhagen, tries to close this gap by categorizing movies along three dimensions of fiction reception (Grodal, 1999):

1. The ‘real-life’ contextualization of fiction consumption
2. The articulation of the viewer’s modes of perception and modes of cognitive and emphatic identification with fictive agents
3. The narrative structure of the world of fictions

With these three dimensions he aims to grasp the tone of a movie in a more natural and content focused way. This leads him to a new set of genres which are fundamentally different from the classical genres. The different genre categories according to Grodal are (Grodal, 1999):

- Associative lyricism
- Obsessional fictions
- Melodramas

IMdB.com
• Horror fictions
• Schizoid fictions
• Comic fictions
• Metafictions

Even though these more tone focused genres seem suitable for our mood based recommender system, it is questionable if there exist enough labeled data for this classification model. In the method section it will be discussed, if Grodal’s approach can actually be used for our given scenario.

1.2.5 Existing movie recommender systems

The increasing amount of movie content creates the need for recommender systems to support the users in their decision-making. This section aims to point out the main concepts and methodologies that are used in the current popular recommender systems. But I also present research papers about newer and more user focused approaches. This includes content and collaborative filtering that aim to deliver the right content to the right person as well as emotion based approaches from Wakil et al.(2015). Afterwards the advantages and disadvantages of these methods are discussed and the knowledge gap is identified.

Currently used methods

1. Content-based filtering

Since the concept of content-based filtering was already introduced in section 1.2.2, I will not specify the structure but instead focus on the process of this approach. Systems that are based on content-based filtering mainly compare the different items either against the user model or against other movies. Therefore, knowledge of the users’ perception about a movie is required which in popular systems is mainly done through rating systems. A user is asked to rate movies either right at the beginning by giving preferences or after watching the content in form of a rating scale. But not only ratings about a specific movie can be used as an input. Preferences about labeled movie information like genre, production year or actors can deliver valuable insights. By letting the user rate what genre he likes best the system will be able to give a suggestion (Burke, 2002; Wakil et al., 2015).

2. Collaborative filtering

The main concept of collaborative filtering was already introduced in section 1.2.2, so I now take a even deeper look into the process of this approach. Similar to a content-based filtering system it tries to grasp the users’ perception of a movie through different types of rating mechanisms. Once the rating is completed the ratings of the movies of different users are compared and like-minded users are identified. If for example User A and User B gave a five star rating on the movies Inception and Shutter Island they are considered as like-minded. If User A now also gave a five star rating to Interstellar the system considers this as a suggestion candidate and will prompt it to User B who is at the receiving end. Depending on who gave a rating for a movie, the position of who is at the receiving end changes and a dialog develops. The more users can
be classified into like-minded groups, the more suggestions will be available and the groups will be more accurate (Burke, 2002).

3. Methods including emotions or moods

First it has to be noted that including the emotions of a user is quite a new development in the field of movie recommender systems. During the literature review only two papers were identified that are trying to use an emotion or mood based approach and one is actually an advancement of the other. While Ho et al. (2006) mainly introduce the detection algorithm of the mood of a user through colors (Ho, Menezes, and Tagmouti, 2006), Wakil et al. (2015) implements this within a hybrid system.

In their paper Wakil et al. (2015) present a recommender system that includes the above mentioned methods but also the emotion of a user (Wakil et al., 2015). By combining these three approaches, and therefore building a hybrid recommender system, they try to solve the problem of a correct user model and how to link the fitting movie content to it. In their system the user passes through five different phases:

(a) Phase 1: User Registration
(b) Phase 2: User Rating
(c) Phase 3: Hybrid Recommendation approach
(d) Phase 4: Prediction
(e) Phase 5: List of recommendation movie

For me the most interesting aspect in this approach is happening during the user registration where the emotional state of the user is detected. The system prompts the user to choose three times between different colors, where every color is attributed with one emotional state (anger, fear, disgust, sadness, happiness). If two of the selected colors represent the same emotional state referenced to the color, this emotional state is attributed to this specific user.

The other mentioned phases in this system are a combined version of collaborative and content-based filtering.

Since the recommender system is a key element of the online user experience on a movie streaming website, companies are quite cautious when it comes to explaining their exact methodology. My literature research could not uncover any more details about specific algorithms except for the above mentioned methods. This lack of public available information about the structure and as well as about the performance of recommender systems limits this research in evaluating the performance of the developed recommender systems. Nevertheless, I will discuss advantages and disadvantages of the different methods when it comes to recommending movies in the next section.

Drawbacks of the current recommendation approaches

Even though the above mentioned methods are the current state-of-the-art in the domain of movie recommender systems, they are not undisputed. One discussed key element is the fact that most of the above mentioned methods mainly derived from a content perspective. The content is grouped into different categories (e.g. genres, actors, similar to..., etc.) and then prompted to the user. Once a user has watched a
movie from a specific category, the system will prompt more and more items from the same category. The underlying assumption is that users always have the same interests and that they would always like to watch the same category of movies. This creates a bubble of recommended content where only a couple of categories are promoted and the others are ‘hidden’. But interests of users might shift over time and a category that was liked before might not be that interesting anymore. In theory these systems do not really have a mechanism that catches these shifts in mood and therefore represses the curiosity for new topics that the user might have.

Another disadvantage of current recommender systems is the way the user model is attributed to a specific user. While in the most common solutions the model of the user is always linked to a specific account the system is only suitable for this one user. If a user for example only watches science-fiction (sci-fi) movies, the bubble of recommended movies is full of movies like Star Wars and Blade Runner. If now a user with a preference for romantic drama movies uses this ‘personalized’ recommender system on the same created account, he would not be satisfied with the given suggestions. Also, if this user would start searching for Titanic or other movies with this genre, this would interfere with the existing model of the original user.

These previously mentioned drawbacks can be led back to the new user problem. Since for content-based and collaborative filtering knowledge about the user in form of detailed ratings are required, it takes some time until the fitting content group is defined and the like-minded group is found. Depending on the user and the efficiency of the system this might take a specific amount of information exchange before the system suggests suitable candidates. To predict this amount is hard since a lot of different influence factors come into play (Burke, 2002).

Another drawback for the mentioned approaches is that the recommender system itself does not have a direct feedback mechanism. The only way these systems evaluate whether a suggestion is good or not, is by seeing if the user is consuming or rating the content. While there might be a lot of different reasons, other than the content, why a movie is not suitable in a certain situation, the system can not relate to these reasons and as a result can not evaluate the recommendation itself. This problem points back to the previous mentioned disadvantage that the recommendation is mainly derived from a content perspective.

Additionally, the current methods have a lack in confidence in their recommendations. When taking a look at the Netflix’s recommendations, the user is prompted with several movies at the same time. This gives the user on the one hand the freedom of choice but also makes it harder to decide (Ricci et al., 2011). Even though the number of possible movies for a user has been reduced through the mentioned methods, the user might still spend a lot of time browsing through the different suggestion pages. As mentioned before, there is only little information available on how successful recommendations based on these systems are. Therefore it is difficult to argue which of these approaches has the higher performance (Resnick and Varian, 1997).

1.3 Research Problem

The following section states the research problem that this paper will solve and will display the different research questions as well.

As explained in the previous section there are several disadvantages of the current existing approaches for recommender systems. They even seem to be intensified when it comes to movies due to the previously mentioned reasons. Even though
modern recommendations are already quite good when it comes to capturing the interests of a user, they still lack of knowledge about what the user actually wants in a certain situation. Therefore, the main research problem I am trying to solve is if knowledge about the mood of the user can help to close this knowledge gap. Since this is a quite new field of study these aspects, that are part of the general user modeling problem, have not been deeply investigated yet. Solving this problem might be the next step into the direction of a more user centered recommender system.

To find a proper solution to this problem it is divided into two different aspects and will be investigated in the form of the following research questions. With the research questions I aim to guide through the process of solving the research problem and to display the process in the most transparent way.

1.3.1 Research Questions

1. Is it possible to build a movie recommender system based on the mood of a user?
   At the beginning it is important to discover and identify the methods that allow us to build movie recommender system based on the mood of a user. This system is called Mood Based recommender system. As already explained in the previous sections it seems that all prerequisites, namely the knowledge sources, are available. The question therefore aims to answer if these different components can be combined to an actual recommender system which delivers high quality suggestions.

2. How does a mood based movie recommender system perform against a recommender system that suggests movies at random?
   To analyze if the mood based movie recommender system can actually deliver a more user centered recommendation, I need to compare its performance against the performance of other recommender systems. Other systems may include ones who follow different methodologies, structures, data models or types. As described in the previous sections there is no public information about the performance of e.g. Netflix’s recommender system what makes it difficult to compare. I am therefore only capable of comparing the performance of the mood based recommender system with the performance of a system that gives recommendations at random. In the following I will call this type of system Arbitrary recommender system.

This section presented the research problem and the associated research questions. To answer these questions several hypotheses are made and explained in detail in the next section.

1.3.2 Hypotheses

In this section the hypotheses for this master thesis are presented. They are derived from the research questions in the previous section.

Hypothesis 1: If I can match the mood of a user with a movie, then I can build a movie recommender system based on the mood of a user.
The first hypothesis tests whether the mood of a person can be used in the domain of movies to create a recommender system and if it fulfills all the requirements.
Hypothesis 2: If I create a Mood based movie recommender system, then it will perform better than an Arbitrary recommender system.

The second hypothesis tests whether a mood based movie recommender system has a higher success rate than a recommender system that suggests movies at random. The term Success Rate will be discussed in chapter 2 Methods.

In this section the two hypotheses were stated in order to be confirmed or rejected in this master thesis. The next section will state the importance of the research before a quick outline is presented.

1.4 Importance of the research

As explained in the background section the internet is already a huge part of our everyday lives without any signs that indicate an end of that trend. Therefore, in order to deal with the ever growing amount of data, the technology has to adapt in a way to serve their human counterparts in the most efficient way. If it can be proved that using a users mood improves the performance in the domain of movie recommender systems, it might in general lead towards a new generation of recommender systems. All sorts of online databases, from music streaming services up to online shops could harvest the additional knowledge to deliver more suitable products and to create a better digital user experience.

If this concept can be applied in an area like recommender systems, this can have an implication for quite a number of other research areas. Chatbots might use mood detection algorithms to be able to understand and react to their human counterparts in a better way. Other disciplines from the field of human computer interaction could follow. Also, the field of robotics could make use of this new approach when trying to improve the communication between a robot and e.g. a patient in a hospital (Breazeal, 2003).

1.5 Outline

This section aims to give a short recap about chapter one and a further outline of the thesis as a whole. Chapter one was mainly focused on introducing the most important concepts of recommender systems and their belonging components, namely the mood of a user and how to classify movies. By describing these concepts the knowledge gap and the underlying research problem were identified. The research problem was split up into two research questions that lead us to our two hypotheses that I am trying to prove in this thesis.

In the next chapter I will present the different methods that were applied to answer the research questions. Starting with the development and implementation of the Arbitrary recommender system with its different components and followed by the comparison of different data models. After the decision about the data model is made, it is implemented into the Mood Based recommender system. The whole process, as well as the different components, are described in chapter 3.4. At the end of the method chapter the analysis method is described before taking a critical perspective at the methodology itself.

Chapter three will explain the analysis of the two different recommender system, and how they can be compared against each other. The results of this analysis will be discussed in chapter four before a conclusion can be drawn in chapter five.
Chapter 2

Method

This chapter will introduce the main methods that are used for this master thesis. Starting with a description of the general quantitative framework I present the whole conducted process to be able to answer the stated research questions in chapter 1. Therefore, it is necessary to understand how the information is collected during the first iteration of the system which is the so called Arbitrary recommender system. The Arbitrary recommender system just randomly prompts suggestions to the user without any further logic behind the matching process. This data is not only used to calculate the success rate for the arbitrary scenario, but also serves as the main input for the data model which will be developed in the second step. The chosen recommender data model, a Random Forest, will be described in detail in chapter 2.2 and the different components are optimized in order to fit the case at hand. In a third step, the second iteration of data collection (the Mood Based recommender system) where the data model is included into the recommender system, will be conducted to get the success rate for the Mood Based recommender system. The results of both scenarios will be compared during a statistical analysis using several descriptive and significance measures. The chapter is concluded with a critical reflection on the chosen methods and on what kind of problems could appear.

2.1 General Approach

In this section the general methodological approach for this thesis, as well as the sample size of the collected data are explained. Since I am aiming to compare the performance of two different systems, a quantitative approach is used. The performance, that can be described as the number of steps until a suitable recommendation has been reached, is described in detail and coded in chapter 3. Before I am able to compare the two different systems both have to be described theoretically and implemented what is done in the next sections. Even though both systems differ in the way a user is matched to a recommendation object, they are using the same types of input and output knowledge base. The input knowledge base, which is based on the PANA-Scale, is described in chapter 3.2.1 and the output knowledge base, which is the outcome of an analysis of the IMdB movie database, is explained in chapter 3.2.2. I have chosen a quantitative approach since I would like to contrast the number of steps a user has to take until the system delivers a successful recommendation. The process of how this data is collected is described in the data collection methods section (chapter 3.2.4) and insights of the whole process are given.

While the first developed system (the Arbitrary recommender system) can be developed without fulfilling any prerequisites except for the availability of the input and output knowledge base, the case is different for the Mood Based recommender system. In that system, the collected data from the Arbitrary recommender system
Chapter 2. Method

![Diagram: Development process of the system]

**Figure 2.1:** Development process of the system

is required in order to train the data model that is the core of the *Mood Based* recommender system. Once the two systems are in place and sufficient data is collected, I can compare the performance of the two systems. An illustration of the whole development process can be seen in figure 2.1. Since the data is collected in form of numbers, and I would like to prove a hypothesis, it seems logical to chose a quantitative approach.

### 2.1.1 Sample size

When talking about the sample size I have to differentiate between the data required for the data model as well as the data which is required for the comparison of the two systems. The data required for the data model is collected within the *Arbitrary* recommender system while the data for the performance comparison is collected by both systems. For the *Arbitrary* recommender system, both data collection processes are the same.

#### Data model sample size

Answering the question of how many rated items are required for a data model in order to get an accurate result is pretty tough. Since I am in the domain of *machine learning* this number highly depends on the selected technique, domain and the data itself. The challenging part is that there exists no mathematical or statistical formula that can derive the perfect number of inputs e.g. for an artificial neural network. Therefore, the general approach is that the more data is used the more accurate are the results. There are only two things that can be done to make sure that enough that is collected.

On the one hand a search for research papers in the same domain where machine learning approach was conducted. By finding similar research papers I aim to reason with analogy. The premises for the selected papers was that they are trying to solve a classification problem, dealing with a similar amount of input as well as output factors and also deal with a similar field (emotion, moods, affect). It turned out that most of the conducted *machine learning* research in the field of mood and emotion detection was done in the context of sentiment analysis and facial expression recognition which is not suitable for my case (Alm, Roth, and Sproat, 2005; Bartlett et al., 2005). Since reasoning by analogy is not suitable for this scenario a different approach is followed.

Because on the other hand, the data model can be frequently tested to see how it converts. Conversion described the accuracy of the data model to predict the outcomes. A soon as this accuracy reaches an adequate value or is stagnating on the same level the data collection is terminated. For my data model I reached this adequate value with an accuracy rating of 0.743. This can be interpreted as 74.3% of
2.2. The Arbitrary Recommender System

In the first step of this research the Arbitrary recommender system is developed. This section will explain the structure of the system, it is knowledge sources, the matching logic and the data collection process. In this system the main idea is to take the input data of the user before I present a randomly selected output. After each output the user must decide if he likes the recommendation or not. The user is prompted with new recommendations until either a positive feedback is achieved or the maximum number of possible outputs is reached. A schematic structure of the Arbitrary recommender system can be seen in figure 2.2.

According to this description of a recommender system I can sort my system into the Utility-based typology as defined by Burke (Burke, 2002). As defined in the introduction chapter, the system is giving a specific recommendation based on its utility factor. Since in this scenario only one iteration is conducted, and I would like to get a random output I define the utility factors as equal for all reachable outputs for all participants. Once a positive feedback is achieved, I will be able to calculate the success rate of the recommender system that is described in detail in the method of the data analysis in chapter 4.
2.2.1 Input knowledge source

My Arbitrary recommender system uses mainly user knowledge as inputs in the way it is defined by Jawaheer, Weller, and Kostkova (2014). Almost all factors that have a scientifically proven strong influence on the mood of a person have been selected as inputs. This includes the age and the gender of the users as well as the current mood. The current mood is collected by using the PANA-Scale that models the mood by asking ten positive and ten negative affect related questions. Table 1.1 displays the several items from the PANA-Scale. Several other influence factors on the mood like the time, day of the week, season or physical condition were also considered as input factors but were rejected for several reasons. One reason was to keep the number of questions to a comfortable amount and also to lower the amount of inputs for the data model. Since for example a standardized test to evaluate the general physical condition of a user contains 100+ question which will ask way too much from the user. It will be argued at the end of this chapter in a critical reflection what additional inputs might be added in a future system.

The final user knowledge is collected through an online questionnaire that consists of 22 questions. Two of them are related to the age and gender of the user while the remaining 20 questions are a representation of the PANA scale test. An example picture of the online questionnaire is displayed in the appendix A or can be visited on http://iwillwatchthat.com/pages/evaluation.html. In general, the website was created using the Bootstrap open source toolkit in combination with an existing template. The goal was to create a simple responsive website with an easy to access questionnaire to create a high response rate. It also aims to intuitively guide the user through the questionnaire and to create an engaging and straightforward user experience.

2.2.2 Output knowledge source

The output knowledge source is based on catalog knowledge as defined again by Jawaheer, Weller, and Kostkova (2014). The IMdB\(^1\) movie database with its detailed labeling was used to select possible outputs. It was decided to use the genre as the

\(^{1}\text{IMdB.com}\)
main classification label. A quick analysis has shown that there exist 24 distinct genres. These genres range from adventure to historical and most movies labeled with more than one genre. This seems to be obvious since most of the labels are only partly capable of describing a whole movie plot. Western movies as an example often contain elements of action and historical movies. Modern sci-fi movies mostly have an adventures character and are also labeled with the adventure genre.

When looking into the labeled movies with only one genre in more detailed, it was shown that they have either a low number of ratings or no comments. This can be seen as an indicator for either completely new or unpopular movie. Since IMdB is a community driven database it is assumed that these movies might not be properly labeled due to their unpopularity. Therefore it was decided, to exclude all movies which are labeled with only one genre to meet the nature of a movie in a better way and to exclude possible false labeled movies.

Taking my analysis a bit further and looking into movies with more than two genre labels it has shown that there exist a lot of unique combinations of labels. This results in a massive amount of possible label configurations that will create an overload of information for our data model as well as for my recommender system. Since I am taking a machine learning approach, I am aiming to break down my output into its smallest reasonable components. Therefore it was decided to exclude all movies from the database that are labeled with more than two genres. The excluded movies (with only one or more than two labels) could be included in the second iteration of the recommender system. In the discussion chapter 4 there is more information available about how movies that are labeled with more than two genres could be easily added to the recommender system.

As a result of the analysis of the IMdB database, I receive a list of movies with only two genre labels. The analysis starts with the application of a SQL query (can be seen in the appendix A) where the occurrences of each genre configuration is counted. I consider a genre configuration as the combination of two distinct genres (e.g. drama and romance). Since this list still consists of more than 100 unique genre configurations, and I do not want to overwhelm the users, I decided to limit myself to the eight genre label combinations with the highest number of related movies. The different genre configurations are displayed in the Table 2.1. Additionally, for each genre configuration a movie that has exactly the same genre labels was selected and which was or will be released between 1894 and 2019. It was selected based on the rating given by the IMdB community. The only restriction was that a trailer is available on www.youtube.com. This is necessary since the trailer is the main format of the output which is prompted to the user. This combination of the eight genre configurations entailed with a specific movie is called one movie set. To avoid biases regarding the chosen movies and to make the results transferable to other movies, in total ten movie sets have been selected. Each set contains eight elements (one for each genre configuration) but with different fitting movies for each genre configuration. The complete list is displayed in the appendix A.

Once the user has answered all the questions from the questionnaire one of the movie sets is selected and the information about the movies is consecutively presented. This information consists out of the title and the trailer of the movie. It was decided not to present any further text descriptions since the main idea is, that a trailer displays the tone and mood of a movie in the most convenient way. The trailers were implemented by embedding the video files from www.youtube.com. The question, which trailer will be presented first, will be answered in the next section.
Table 2.1: Example set of movies with their genre label configuration

<table>
<thead>
<tr>
<th>Group</th>
<th>Genre Classification</th>
<th>Movie</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Drama-Romance</td>
<td>Forrest Gump</td>
</tr>
<tr>
<td>2</td>
<td>Drama-Comedy</td>
<td>Lady Bird</td>
</tr>
<tr>
<td>3</td>
<td>Drama-Crime</td>
<td>Three billboards outside Ebbing, Missouri</td>
</tr>
<tr>
<td>4</td>
<td>Comedy-Romance</td>
<td>Amélie</td>
</tr>
<tr>
<td>5</td>
<td>Drama-Action</td>
<td>Gladiator</td>
</tr>
<tr>
<td>6</td>
<td>Drama-Thriller</td>
<td>Sixth Sense</td>
</tr>
<tr>
<td>7</td>
<td>Documentary-Biography</td>
<td>Searching for sugar man</td>
</tr>
<tr>
<td>8</td>
<td>Action-Crime</td>
<td>Sherlock Holmes</td>
</tr>
</tbody>
</table>

2.2.3 Matching logic

As explained above the user is filling in the input knowledge questionnaire and answers all the prompted questions. After question number 22, the system conducts a random selection in two steps. In the first step it randomly selects one of the ten movie sets. In the following step the system conducts a second randomization by randomly selecting one element from the chosen movie set. The result of the randomization process is prompted to the user, in form of the information about the movie. The user decides whether he would like to watch a specific movie or not. The general randomization function is written in JavaScript and is defined the following way. Using this function basically allows to pick randomly one out of ten numbers starting with 0.

```javascript
ran = Math.floor((Math.random() * 10) + 0)
```

The second function then accesses the set and randomly selects a movie. A movie within a set is always displayed only once since it is expected that a user will not change his opinion regarding a movie within a couple of minutes.

```javascript
for (var i = 0; i < storesectiontrailer.length - 1; i++) {
  if (currentsection === storesectiontrailer[i]) {
    RandomSectionSelector();
    currentsection = null;
  }
}
```

2.2.4 Data collection process

All the different user inputs and output choices are collected using Google Analytics which entails also the usage of the Google Tagmanager. The two tools are widely popular website tracking programs, that can be easily implemented, and offer a variety of customization possibilities. Several so called data layer variables are defined and filled with the results of the different inputs and outputs of the user. When the user reaches the end of the system (either through selecting Yes as an answer choice or always selecting No) the data is pushed into the Google Tagmanager where the data is transformed and forwarded to Google Analytics. In the appendix A you can find a figure that displays the final DataLayer as well as a figure that displays the Google Analytics custom report which displays how the data appears in the tracking tool.

The tool was selected since it is the state-of-the-art tracking tool in web analytics and allows the developer to easily implement the tracking. Especially the concept of the data layer lets the developer decide what data and in which format should be
pushed to the tool. In the tool itself the data then can be accessed in real time what enhances the implementation and testing process.

2.3 Recommender data model

In this section the different investigated data model algorithms are presented and the selection process is documented. At the beginning a short introduction to machine learning is given, before the five different algorithms (Random Forest, Artificial Neural Network, Support Vector Machine, k-Nearest Neighbors Algorithm, and Naive Bayes Classifier) that have been applied are shortly explained and their performance is compared to each other. The algorithms have been selected since they are well-known in the domain of machine learning, and they are known to have their own strengths and weaknesses depending on the type of data. To receive the highest possible performance of the data model, it was decided to test several algorithms that are all described by Ethem Alpaydin in the book *Machine Learning* (Alpaydın, 2014). Additionally, the availability of the selected algorithms within the Scikit Python library is also a limiting factor (Pedregosa et al., 2011). It is important that they are all based on the same python library because then it can be assumed that the definition of accuracy do not deviate from each other. In this context the accuracy can be defined as the degree to which the algorithm is able to predict the outcomes (Alpaydın, 2014).

Machine learning is a concept within the field of computer science that aims to find dependencies, patterns or regularities in complex and data rich problem solving scenarios (Alpaydın, 2014). In order to solve my problem, a general supervised learning approach is selected. The main idea behind supervised learning is to link an input to an output by observing existing input-output pairs. In general (depending on the exact algorithm) the input is pushed through the algorithm and the output is observed. Afterwards, the deviation from this output to the actual output is calculated. The algorithm is now adjusted in a way that this deviation is minimized. This will result into a more accurate version of the algorithm than before. By consecutively continuing this process on the whole training data set, the algorithm will reach a level where it can predict the outcome of any given so far unknown input. Next to a training data set there is also a test set deployed where the size depends on the chosen algorithm and domain. The algorithm is now deployed with the test data and the two outcomes are compared.

The performance for each algorithm is calculated through the accuracy value that indicates to what degree the chosen algorithm is suitable for forecasting the results of the given problem. In general, it can be defined as the percentage of the cases the algorithm was able to predict the correct ranking of a movie set against a test data set. How to get this accuracy factor for each algorithm depends on the algorithm itself and is explained in detail for each algorithm in the following sections. In this chapter I also added the results of the comparison between the different algorithms, since the selection of an algorithm is required to build the Mood Based recommender system. This selection process is part of the method and the results are therefore presented in this chapter.

Before running the data through the algorithms, one major adjustment of the data had to be done. So far the result of the system could be one of the nine options: Genre1, Genre2, Genre3, Genre4, Genre5, Genre6, Genre7, Genre8, "No movie found". All users who were not able to find a movie are for now grouped into one category. But this can have a negative influence on the data model since it will try
to suggest a movie for specific and perhaps contradicting moods where actually no movie is available so far. To avoid this all the items with a "No movie found" were removed from the database. This leads to a decrease in the total number of entries in the database but will result in a higher data model accuracy.

2.3.1 Random Forest

Definition

A Random Forest is a sub concept of a decision forest which can be described as a combination of multiple learners. A decision forest consists of several trainable decision trees and helps to solve general classification problems especially with small data sets. A Random Forest differs from other statistical models by the fact that it does not care about the distribution of different features within the data set. The Random Forest also uses a discriminate-based approach while most statistical measures use a likelihood-based method. A discriminant method calculates the discriminant directly at the beginning without taking the density of each class into consideration (Alpaydın, 2014).

The developed Random Forest uses a couple of parameters that need to be documented to make the process comprehensive. In the Random Forest I have to define how many trees I would like to have in the forest. This parameter (n_estimators) was set to 25. This number derived from an experimental testing of different numbers and comparing the output. There exist several methods to approximate calculate the optimal number of trees. In the discussion, chapter 4 you can find an explanation why those methods were not conducted. Additionally, this algorithm uses a sigmoid function to calculate which side of the decision tree should be followed. The sigmoid function is recommended by the developers of the used scikit-learn library² (Pedregosa et al., 2011).

Accuracy

The accuracy is measured by dividing the data set into a training and test set. In my case the training set consist out of 400 entries while the test set consists out of the remaining entries. 400 entries were selected to have a clear cut within the data set. The training set is used to create the Random Forest (consisting out of several decision trees) while the testing set is used to validate the algorithm. Therefore the testing entries are pushed through the forest and their outcome is compared against the already collected data. The accuracy for the Random Forest was calculated with 74.3% which can be seen in table 2.2. The probabilities with which a specific movie for a given example are displayed to showcase how confident the system is in its decision. With a probability of 36.0% the system is confident that the user with the given input would like to watch a movie from Genre1. Combined with the second and third highest probabilities the system has a confidence level of 72.0% to find the right movie within the first three recommendations.

2.3.2 Artificial Neural Network

Definition

An Artificial Neural Network (ANN) is one of the most popular concepts in machine learning. While trying to reproduce the main concept of the human brain, ANN’s try

to solve the classification problem through adjusting the weights of the connections between different perceptrons. A perceptron can be described as the main processing element that gets an input, applies a number of calculations, and then creates an output. This output is either the input for a new perceptron or the final output at the end (Alpaydın, 2014). A calculation with different weights is used on top of the connections between the perceptrons in order to display the significance of each connection. During each iteration of the ANN these weights are adjusted and stored in order to minimize the difference between derived and actual output. Therefore, the ANN is following the general supervised machine learning approach. The stored values of the weights then represent the captured knowledge of the ANN.

From a structural perspective it was decided to select four hidden layers with 100 perceptrons. These four layers sorted the 22 input dimensions (the result of the questionnaire) into one of the available genre classifications. A ReLU (Rectified Linear Unit) activation function was used to decide whether a perceptron is fired or not. This was selected since it is one of the most popular activation functions and proven to be able to solve complex classification problems (Alpaydın, 2014). In that context an adam optimizer was deployed which was in charge of updating the weights and to reduce the loss function. The ANN was trained for 1000 epochs with a batch size of five. Similar to the Random Forest there exist quite a lot of room for improvement within the different algorithms. This and other possible improvements on the algorithm will be discussed in chapter 4.

Accuracy

The accuracy of an ANN is measured similarly as for the Random Forest. The collected data is divided into two distinct sets: training and testing set. While the training set is used to train the data model, the testing set is used to measure the accuracy and quality of the ANN. The testing and training set were composed in the same way as for the Random Forest. Again, the main concept of supervised learning is followed and the results from the ANN for a given input are compared with the actual results. The developed ANN achieved an accuracy score of 65.7% but because of the more complex set up of the ANN the probabilities were not calculated.

2.3.3 Support Vector Machine

Definition

One of the classical techniques in machine learning are support vector machines (SVM) that help to solve general classification problems in a big data environment. While there exist a couple of different types of SVM it was decided to select a linear SVM since it is the most convenient one to implement due to its availability in the respective python library and due to its proven record in solving general classification problems. A linear SVM uses a quite simple statistical equation to divide the different data points into domains. Since my problem consists of several domains a linear approach might not deliver the best results but as mentioned was selected based on the ease of implementation (Suthaharan, 2016). This potential of optimization will be discussed further in chapter 4.

Accuracy

In a SVM the accuracy can be defined and calculated by the mean certainty of the given test data and labels. This value is calculated directly by the respective method
in the selected python library. This is quite a harsh metric since it requires to correctly predict all labels (Pedregosa et al., 2011). The SVM received an accuracy score of 28.9% which is far behind the earlier presented methods. The probabilities displayed in figure 2.2 are the confidence scores calculated by the linear statistical function and are a bit more complex to interpret than the probability scores of the other algorithm. But since the accuracy score of the SVM is relatively low the confidence scores were only calculated for the sake of completeness.

### 2.3.4 k-Nearest Neighbor Algorithm

**Definition**

The *k*-Nearest Neighbor (*kNN*) algorithm interpolates a subset of neighboring instances and calculates the distance between the different subsets. The algorithm can also be seen as a ranking problem where the different items get different scores. These scores represent the distance between the different items and are grouped into the different features (Alpaydın, 2014). The *kNN* also belongs to the classical methods to solve complex classification problems.

**Accuracy**

The *k*-Nearest Neighbor algorithm is capable of returning the mean accuracy on the given test data and labels. Since I try to solve a multi-label classification, it is required to correctly predict all the available labels. The *kNN* received an accuracy score of 14.0% as displayed in figure 2.2. Due to the methodology the probabilities are calculated in steps of 11.1% with the highest probability for genre seven (33.3%). Followed by genre five and eight on the second rank I already receive a combined probability of (77.7%). This indicates that with a confidence of 77.7% the algorithm is able to find a fitting movie within the first three suggestions.

### 2.3.5 Naive Bayes Classifier

**Definition**

A Naive Bayes Classifier (*NBC*) uses a more statistical approach and is characterized by its underlying likelihood calculations. It can be used in supervised learning scenarios and can accurately predict the output of a test instance. The classifier is called naive since it follows two basic assumptions. First it is assumed that the different predictions are conditionally independent of each other. Secondly it is assumed that all information is available and that other influence factors do not exist. Especially the second assumption makes the classifier naive since it sees the data in a closed world scenario (Alpaydın, 2014).

For my data model I have chosen a single Gaussian distribution since it has been proven to solve several real world classification problems (John and Langley, 1995). Nevertheless, the authors of "Estimating Continuous Distributions in Bayesian Classifiers" (1995) already note that there might exist solutions that can grasp the real-world context in a more efficient way.

**Accuracy**

Similar to the SVM and the *kNN* algorithm the accuracy can be defined and calculated by the mean certainty on the given test data and labels. Once again the
2.4. The Mood Based Recommender System

Table 2.2: Accuracy comparison of data model

<table>
<thead>
<tr>
<th>Methodology</th>
<th>Accuracy</th>
<th>Prob1</th>
<th>Prob2</th>
<th>Prob3</th>
<th>Prob4</th>
<th>Prob5</th>
<th>Prob6</th>
<th>Prob7</th>
<th>Prob8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.743</td>
<td>0.36</td>
<td>0.2</td>
<td>0</td>
<td>0.08</td>
<td>0.12</td>
<td>0</td>
<td>0.08</td>
<td>0.16</td>
</tr>
<tr>
<td>ANN</td>
<td>0.657</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SVM</td>
<td>0.289</td>
<td>-0.756</td>
<td>-0.856</td>
<td>-0.768</td>
<td>-0.692</td>
<td>-0.368</td>
<td>-1.08</td>
<td>-0.747</td>
<td>-0.776</td>
</tr>
<tr>
<td>kNN</td>
<td>0.140</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.111</td>
<td>0.222</td>
<td>0.111</td>
<td>0.333</td>
<td>0.222</td>
</tr>
<tr>
<td>NBC</td>
<td>0.140</td>
<td>0.882</td>
<td>0.050</td>
<td>0.055</td>
<td>0</td>
<td>0.01</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The predefined method from the selected python library is used to get these accuracy factors. Also for the NBC it only counts as correctly predicted if every single label is predicted correctly. The NBC receives an accuracy rating of 14.0% with probabilities that have only a small variance. The genre one receives the probability of 88.2% which is way ahead genre two and three with approximately 5%.

2.3.6 Data Model Comparison

To decide what kind of data model should be used, all algorithms are compared with each other with the focus on the accuracy ratings. But also the format of the probabilities have to be taken into consideration to decide if the data model is a proper candidate for my system. In Table 2.2 you can see the results of this comparison with the measures for accuracy and an example for the probabilities of one set of inputs.

Since all the chosen algorithms (except ANN) are using the same python library classes and methods and share the same understanding of accuracy and probability, I can compare the outcomes directly. Due to the chosen implementation of the ANN it is very difficult to get the probabilities of every item. Since the ANN did not deliver the highest accuracy it was decided to not focus on this problem and continue with the promising Random Forest.

In Table 2.2 the results can be described as following. The Random Forest algorithm has the highest accuracy with 74.3% followed by the ANN with 65.7%. While the SVM achieves a score of 28.9% the kNN and the NBC only achieve 14.0%. After analyzing the five algorithms and comparing their results, one algorithm for my data model was selected. Based on the high accuracy scores as well as the promising probability distribution the Random Forest was selected as the main algorithm for the data model.

2.4 The Mood Based Recommender System

After the machine learning algorithm was selected and the data model was created, it can now be implemented. This will create the Mood Based recommender system with a more complex matching logic than before. In general the whole structure is similar to the Arbitrary recommender system where mainly the matching logic works differently. A schematic representation of the new process can be seen in figure 2.3. Nevertheless, also the type of recommender system varies a bit. Since there exist no information about the user preferences for a specific mood, I now can sort my developed system into the knowledge-based recommender system category. Even though I have no clear picture how a specific affect influences the desired outcome, I have this information stored within the data model based on the historical data and the
learned model. Since this slightly differs from the Arbitrary recommender system I can not use the utility-based recommender system typology anymore. The reason for this is that the focus of an utility-based recommender system lies in the optimization of the utility function which is achieved through several satisfaction techniques. In general this is a running process where the utility function is improved for every iteration. Since I use the data model as a kind of knowledge source, a utility score is not required in the new given scenario.

![Schematic figure of the Mood Based recommender system](image)

**FIGURE 2.3**: Schematic figure of the Mood Based recommender system

### 2.4.1 Input knowledge source

Similar to the Arbitrary recommender system the input knowledge source is collected through the questionnaire which is already described in section 2.2.1. Additionally, I take functional knowledge into account that comes mainly in form of the data model. In the data model the functional logic of how to match the user’s needs with a specific item is stored. Since this is stored in form of weights and vectors it is pretty hard to visualize this type of knowledge and refers basically to the Black Box Problem (Alpaydın, 2014).

### 2.4.2 Output knowledge source

The output knowledge source is exactly the same as described in section 3.2.2 and will be reused for the Mood Based recommender system. I divide the list of films again into sets, where each set consists of eight movies (one for each genre configuration). Also the way the output is displayed is the same as before. The displayed trailers, as the main source of information, aim to give the user a possible idea about the movie with the focus on the tone and mood of the movie. I use exactly the same set of output knowledge to be able to compare the two systems with each other during the analysis.

### 2.4.3 Matching logic

For the Mood Based recommender system I use the data model developed in section 2.3 in order to match a specific mood with a movie. A Random Forest based data
model was selected due to its good results during the selection process. In chapter 3.3 the different investigated methods are described and their performance is compared with each other. The data model takes the input from the user (derived through the questionnaire) and lets it run through the algorithm. The algorithm then calculates the probabilities for each single genre and the one with the highest probability is prompted to the user. If this suggestion is rejected the object with the second highest probability will be presented and so on. If the rare scenario appears and two objects have the same probability they will be presented in a chronological order since it is expected that the movies at the top of the movie set are slightly more popular than the ones at the bottom. The algorithm will be deployed in the back-end and the user therefore will not see any of the conducted calculations.

2.4.4 Data collection process

For the data collection process I am using the same approach as for the previous recommender system. Google Analytics will be used as the main platform for collecting data by connecting the within the website implemented dataLayer through the Google Tag Manager with the Google Analytics account. A picture of the dataLayer can be found in the appendix A.

2.5 Data Analysis and Coding

Once all data for the two recommender systems is acquired the data analysis can be conducted. What I am trying to achieve is to compare which of the two recommender system has the stronger performance. As an indicator for performance I define a variable called Steps to success which is calculated for both recommender systems. A recommendation is considered as successful only if the system delivered a suitable suggestion that the user accepted by answering the question whether he would like to watch it or not with Yes. By answering with Yes the user states that the given recommendation is of high quality and the system was able to deliver a fitting suggestion.

Rejects the user a presented recommendation by answering No it is considered as one step. So the variable could also be described as number of presented movies until the user selects Yes as an answer choice. The number of steps is then calculated for each recommendation and by using different statistical measurements the two recommender systems are compared with each other.

For the case that all suggestions have been rejected, which means that the presentation of all eight possible movies did not create a successful response, the Steps to success variable is labeled as No.

2.5.1 Frequency Analysis

This first set of analysis will mainly help us to prove the first hypothesis since it gives us insights if a recommendation can be given based on the mood of the user. Therefore, a frequency analysis is conducted to identify the outcomes and how the distribution of success looks within each system. It also assists to get a better understanding of the difference in the performance of the two recommender systems because the frequency distribution differs. This will be done on a second level where the aim is to showcase how the distribution of the required Steps to success varies between the two systems. This will build the baseline for the t-Test which is conducted as a second set of analysis method.
2.5.2 Student’s t-Test

Since I would like to compare the performance of both recommender systems with each other a Student’s t-Test with independent samples is deployed to prove the significance of the results. The goal of the t-test is to compare the responses in two different groups. In this case the users who used the Arbitrary and the ones who used the Mood Based system. This can be done if there are two sets of samples that are independent of each other and identically distributed. The two sets are the results of performance measurements for the two different recommender systems which are conducted independently of each other. For both systems I have the same measures for performance what makes it the dependent variable. Therefore, I take the number of steps as the dependent variable and the type of recommender system as the independent variable (Moore and McCabe, 2003). The t-test is conducted using the statistics tool SPSS and the results and analysis are presented in chapter 3.

2.6 Critical reflection

Even though the above mentioned methods were carefully selected in order to meet the unique demands of the presented problem, they will not be able to deal with all the potential difficulties. In this section I take a critical perspective on my chosen methods and will discuss scenarios where they might not fit properly.

- Quality of received data
  
  As described in section 3.2.1 and 3.2.2 I collect my data mainly using an online questionnaire in form of a website that selects a value by clicking Yes or No. While the method was selected in order to make data collection and the inputs and outputs for the data model as straightforward as possible, this can have some implications on the quality of the data. Since every user might have a different perception on what each element of the questionnaire means, answers could vary. For example the question “How proud do you feel?” highly depends on each users’ definition of “being proud” and how they perceive it. One user might be proud for getting up in the morning before 10 while another user only feels proud after completing a marathon. Even though a description for each of the question was given there might be some factors that influence the ability to compare the different data entries.

- Reasons for not finding a result
  
  In this research project I consider that the movie a user would like to watch then represents the genre configuration of the specific mood. In contrary if the user does not find a fitting movie his mood can not be linked to any genre configuration. But there might be several reasons why a user will not like a specific suggestion.

  One reason e.g. could be that he already watched the movie and might be looking for something new. Or a user might not like a specific actor for some reason who is staring in a proposed movie. This will result in a No to the system even though the user might be interested in the genre configuration in general. My current system will not be able to measure these kinds of failures due to the lack of background knowledge about why a user selected No. Was it really because he does not like the genre configuration or were other attributes the main driving forces behind the No? A qualitative analysis might help to
2.6. Critical reflection

put a little more context into the answers of a user but would fall a bit out of scope of this thesis.

• **Seasonality**

Another aspect that might influence the quality of the data is the factor of seasonality. Since psychological research identified a weak link between the time of the year and the way people experience their mood I have to take this factor into consideration. If e.g. the recommender system is tested during winter when it is cold and dark outside this might result in a different type of movie desire than on a warm sunny summer day would create. To remove this influence factor I aimed to conduct the data collection in the most narrow time frame as possible. To be more specific I conducted the data collection for the *Arbitrary* recommender system from mid March until end of April and the data collection for the *Mood Based* recommender system in the first two weeks of May.

Another way to solve this would be to use the season as an additional input factor for the data model. This approach will be discussed in detail in chapter 4.

• **Distribution of the Time Attribute**

As explained earlier the time of the day during that the user filled out the questionnaire might have an influence on the perception of mood and should therefore be taken into consideration. The assumption behind this is, as explained in the introduction chapter, that one’s mood can vary during the day as a result of stress, tiredness etc. This might result into the following problem for my system. Since most users filled in the questionnaire in the evening after they got home from a long day at work, the data is missing quite a lot of entries and might be therefore optimized for an *after work* scenario. It is hard to predict how this will influence the overall performance of the system, but I need to take this into consideration. This problem will be further discussed in chapter 4.

• **Selection of movies**

When I sorted the different movies into the different genre configurations I mainly relied on the top movies from IMdB. Since IMdB is a community driven website with their main users from Europe and especially the United States, there exist a bias within the selected movies. Most of the movies derive from a Hollywood production with only a few exceptions (e.g. *Amélie*) and other movie production areas seem to be underrepresented. For example movies which were created in India (also known as Bollywood) have their own database and will not show up directly on the IMdB page. This might indicate, that my system will be more optimized for the users who watch mainly productions from Europe and especially the United States.

• **Selection of genres**

In section 3.2.2 I described how I retrieved the genre configurations from IMdB in order to fit the purpose. Since it seems reasonable to select only a few genres that will be displayed to the user, some information and especially desires of the users might get lost. The way I extracted the most important types of genre (by counting the total number of appearance) might wrap the view on what is
actually a popular genre. E.g., it seems that *documentations* are highly over-
represented in comparison to *horror* movies. This problem might solve itself
over time when more and more genre configurations are included but this lack
of representation from specific genres still has to be taken into consideration.

- **Comparing only performance**

  As the main dependent variable for the t-Test the performance is used. Con-
sidering that the answer to the output is not completely black or white (*Yes* or
*No*) but might also be *maybe* the current setup will not be able to track that. In
general, it seems that measuring only the performance does not give a com-
plete picture of the users’ opinion about a specific suggestion. To get more
background information about the intentions of a user, more user behavior
centered measures must be taken into account. E.g. the time spent on the page
or how long a specific trailer was watched would make the results way more
specific. But since the focus of this research lies on the matching system in be-
tween and not the evaluation of the output, this problem falls out of the scope.
Nevertheless, this could be a good starting point for future research and will
be further discussed in chapter 4.
Chapter 3

Analysis

In this chapter the analysis of the previously collected data will be conducted. At the beginning two different analysis methods are operated by using the statistic analysis tool SPSS. In the next section I will first use a frequency analysis which aims to create insights on how the two systems convert the input into suitable suggestions. Afterwards a t-Test is applied to understand and measure the performance of the two implemented solutions with each other. Being able to do so requires a variable that was collected for both systems as performance indicator and as the baseline for the Null hypothesis which will be defined in the next paragraph. The variable that I call Steps to success was already defined and described in detail in the section 2.5 Data Analysis and Coding. To shortly sum this up I define Steps to success as the number of displayed suggestions before the user chooses a recommendation with Yes and therefore accepts the suggestion.

Additionally to the Steps to success which is the key indicator for the comparison of the performance of the two system, I also want to take a general look at the Success Rate of the two systems. As explained above a success is achieved as soon the user votes Yes on one of the recommendations. Therefore, the Success Rate is the number of successful recommendations in comparison to the unsuccessful ones where the user always selected No as the answer choice.

I then take my hypotheses from chapter 1 Introduction and try to verify them using two different statistical analysis methods. For the first hypothesis, whether I am able to build a recommender system based on the mood of a user, I am using descriptive statistics to showcase that the users’ mood can be used as an input factor for a recommender system.

The second hypothesis, that a Mood based recommender system can perform better than an Arbitrary recommender system, is verified by using a significance analysis approach. Therefore, I will create two Null hypotheses which will help us to understand how the two systems are related to each other. To compare the two systems I want to prove that $D$ (Steps to success for the system which uses the datamodel) is not equal to $R$ (Steps to success for the system which uses the Arbitrary approach). Therefore, my first Null hypothesis that we want to reject can be written as the following:

$$H_0 : D = R$$

(3.1)

The alternative hypothesis that we want to verify contains the information that $D$ and $R$ are not equal:

$$H_A : D \neq R$$

(3.2)
3.1 Hypotheses Testing

In this section the two hypotheses will be tested. This will be done by using descriptive statistics and significance tests.

3.1.1 Descriptive Statistics

The descriptive statistical tests aim to prove or reject the first hypothesis and to get a better understanding of the performance of the two recommender systems. In this section the observations from each of the graphs is stated and all findings are summed up in the respective chapter. The descriptive tests consist mainly of frequency tests that are displayed for each recommender system in two different ways (a bar- and a pie-chart).

I start with the descriptive statistics for the Arbitrary recommender system which can be seen in figure 3.1 and in figure 3.2. For the Arbitrary recommender system I had a total of 630 participants who completed the survey. As it can be seen in figure 3.1, 240 of these users found a fitting movie with the first suggestion. 101 users selected the second suggestion as a suitable candidate followed by 75 users for the third suggestion. Users who needed four to eight Steps to success are quite underrepresented with numbers between 38 and 4. At the end 108 users didn’t find any of the suggested movies as suitable.

To set this a little more into a context let us take a look at figure 3.2 where we can see a pie chart representation. 38.1% of the users found the first suggestion as fitting while 16.0% rejected the first suggestion and selected the second one. 11.9% of the users selected the third suggestion and the Steps to success four to eight deliver a combined 16.8%. The number of users who did not find any of the suggested movies suitable can be labeled with 17.1%.
3.1. Hypotheses Testing

**Figure 3.2:** Frequency analysis of the *Arbitrary* recommender system: Pie chart representation

The pie chart also delivers the values for the *Success Rate* of the recommender system which is the total number of suggestions minus the number of *No’s*. For the *Arbitrary* recommender system the *Success Rate* is 82.9%.

**Figure 3.3:** Frequency analysis of the *Mood Based* recommender system: Bar chart representation
I now take a look at the results of the Mood Based recommender system and I am therefore applying the same analysis as for the Arbitrary recommender system. In the second iteration of the data collection process I got responses from a total of 114 participants. 61 of these decided to select the first movie suggestion while 17 selected the second one. The third one was selected by 14 users and the remaining four to eight Steps to success make a combined 12 users. 10 users did not find any suggestion suitable and therefore always selected No as an answer for all presented suggestions.

The relative numbers for the Mood based recommender system can be described as following. 53.5% of the users selected the first movie that was presented to them. 14.9% decided to select the second presented choice while 12.3% have chosen the third presented suggestion. Four to eight Steps to success were selected by a combined 10.7% which is just a bit more than the 8.8% of users who could not find any suitable suggestion at all.

The Success Rate for the Mood Based recommender system, which is the total number of users minus the ones who always voted for No, is 91.2%.

<table>
<thead>
<tr>
<th>System</th>
<th>1 Step</th>
<th>2 Steps</th>
<th>3 Steps</th>
<th>4 Steps</th>
<th>5 Steps</th>
<th>6 Steps</th>
<th>7 Steps</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arbitrary</td>
<td>38.1%</td>
<td>54.1%</td>
<td>66.0%</td>
<td>72.0%</td>
<td>76.6%</td>
<td>79.3%</td>
<td>82.2%</td>
<td>82.8%</td>
</tr>
<tr>
<td>Mood Based</td>
<td>53.5%</td>
<td>68.4%</td>
<td>80.7%</td>
<td>86.0%</td>
<td>86.9%</td>
<td>88.7%</td>
<td>90.5%</td>
<td>91.4%</td>
</tr>
<tr>
<td>Difference</td>
<td>+15.4%</td>
<td>+14.3%</td>
<td>+14.7%</td>
<td>+14.0%</td>
<td>+10.3%</td>
<td>+9.4%</td>
<td>+8.3%</td>
<td>+8.6%</td>
</tr>
</tbody>
</table>
3.1. Hypotheses Testing

In figure 3.5 you can see a comparison of the frequencies between the two systems. It can be observed that for the data model driven Mood Based recommender system (D) the curve falls after the first step than for the Arbitrary recommender system (R). The curve for D proceeds in general about 8.0% to 15.0% below the curve of R. The exact values can be seen in table 3.1.

All the implications of the differences between the results and graphs are made in the findings section.

3.1.2 Significance Analysis

In this section I am going to describe the results of the significance analysis that was conducted in form of a Student’s t-test. The two figures 3.6 and 3.7 display these results. Figure 3.6 provides the general information about the two groups that I am comparing and that represent the two systems. Group D, which stands for the data model driven Mood Based system, counts 104 users. This number differs from the earlier stated sample size since for this t-Test all the users who were not able to find a suitable movie were removed. This was done since I am comparing only the performance of successful elaborated surveys. The Steps to success has an average of 1.92 with a standard mean error of 0.145. For the Arbitrary recommender system (R) the group size for the t-test is 522 and has a mean of 2.34 with a standard mean error of 0.075. The low standard mean error indicates that the values are well distributed (Moore and McCabe, 2003).
Figure 3.6: Variable overview t-Test

Figure 3.7 displays the significance of these numbers. At first, I analyze the results of the Levene's Test for Equality of Variances. Since the result for Sig. is 0.009 and therefore lower than 0.05 it is assumed that variances are not equal. Hence I therefore can select the second row of the figure to test my hypothesis. In the t-test for Equality of Means I can select the value for Sig. (2-tailed) which is 0.011. I additionally take a look at the Mean Difference which is -0.422.

All the implications of th results and graphs are made in the findings section.

![Group Statistics Table]

**Figure 3.6: Variable overview t-Test**

**Figure 3.7: t-Test results**

### 3.2 Findings

The findings are divided into three main categories. I start with the findings of the comparison of the success rates of the two recommender systems which will lead us to the analysis of the first hypothesis. Afterwards I take a deeper look into the second hypothesis so that I can prove or reject it.

As mentioned above and also displayed in table 3.1 the Success Rate differs for the two systems. With 91.2% of the sessions where the user was able to find a movie the Mood Based has a higher Success Rate than the Arbitrary recommender system (82.9%). This high number of successfully given recommendations can already be an indicator for the result of the first hypothesis. Additionally, the comparison of the two Success Rates can be used to prove or reject the first hypothesis.

**Hypothesis 1**

In order to prove or reject the first hypothesis I take the frequency analysis into account. Since there is no formal definition or value for when a recommendation system is considered as successful, I can only take the general definition into consideration. Since Burke describes a recommendation system "Any system that (...) has the effect of guiding the user in a personalized way to interesting or useful objects in a large
3.2. Findings

I have to prove that the user received a useful object.

The analysis of the frequency analysis showed that with a Success Rate of 91.2% the system was indeed guiding the user to “interesting or useful objects”. The frequency analysis even showed that in 53.5% of the cases the system displayed the right object as the first suggestion and in 80.7% of the cases within the first three suggestions (Figure 3.4). Therefore, my findings indicate that the Mood Based system works successfully because the answers from the users’ are positive (clicks on Yes).

Even though there is a lack of formal description of when a recommender system is successful, it has been proven that the presented Mood Based recommender system is able to deliver suitable recommendations for the user.

Therefore, my

**Hypothesis 1:** If I can match the mood of a user with a movie, then I can build a movie recommender system based on the mood of the user

- Matched the mood of user to a movie using the Random Forest machine learning approach
- Build a recommendation system which successfully delivers recommendations

**Hypothesis 2**

The second hypothesis was focused on comparing the outcomes of the two recommender systems. The hypothesis was stated as:

**Hypothesis 2:** If I create a Mood Based movie recommender system, then it will perform better than an Arbitrary movie recommender system.

In order to verify or reject the second hypothesis I take my earlier stated Null Hypotheses into consideration. The first Null Hypothesis ($H_0: D = R$) is rejected since the analysis of the t-Test in figure 3.7 show that the two system significantly different from each other (p-value = 0.011). Why they are significantly different from each other and how to interpret 3.7 can be read in section 3.1.2 Significance Analysis.

Since the first Null Hypothesis is rejected I can verify the alternative Null Hypothesis ($H_A: D \neq R$) which states that both systems perform significantly different from each other. This indicates that one of the two system needs significantly fewer Steps to success than the other one.

The Group Statistics in figure 3.6 can show us which of the two system requires fewer Steps to success. In the column labeled with Mean I can see that the Arbitrary recommender system requires on average 2.34 Steps to success while the Mood Based system only requires 1.92.

Also, when looking back at the frequency comparison in figure 3.5 it can be concluded that D performs better than R since the line of D is always below the line of R.

Therefore, the hypothesis two is verified since:

- I created a mood based movie recommender system
- The developed system performs better than the Arbitrary recommender system
Chapter 4

Discussion

In this chapter I will discuss the results of the data analysis and what implications this has for future research. Additionally, I will not only discuss the limitations of the research process but also of the system as a whole. The limitations also include some more general considerations about the proposed recommender system. A critical reflection on the PANAScale as well as on the data models will be presented and the comparison to the different other recommender systems will be challenged. In the end the ethical aspects of a mood based recommender system will be discussed.

4.1 Summary of the results

The observed results of the data analysis can be summed up as following. It is possible to build a recommender system that utilizes the mood of a user to create a model. This model of a user is developed by looking from the user’s perspective and tries to represent the user’s desires and needs. In contrast to the state-of-the-art systems where the suggestion is delivered from a content perspective, this new model delivers a more user centered recommender system.

The Mood Based recommender system delivers movie suggestions with a high Success Rate, that indicates that the suggestions are interesting and useful for the user. This personalized recommendation that is presented to the user is derived from a large space of possible items and therefore meets the requirements of a recommender system (Burke, 2002).

The analysis has shown that the Mood Based recommender system performs better than an Arbitrary recommender system. It also delivers high value recommendations in a more efficient way by reducing the Steps to success in a significant way.

4.2 Limitations

Possible reasons for not finding a fitting movie

The developed system was able to show its capability when it comes to suggesting the right movie at an early stage. Considering that in 80.3% of the cases the system was able to deliver a suitable suggestion within the first three displayed movies. But even more impressive is the fact that a suitable movie was found in 91.2% of the cases for the Mood Based recommender system. One reason for this might be that all the presented movies are quite popular and can be described as classics. But what are the other 8.9% of the users interested in? From my point of view I can consider two different scenarios:
1. **Not in the mood for a movie**

In the first scenario I would consider that the user is just not in the mood to watch a movie. If the user is for example at work or on his way home while doing the survey, he might not be in the mood to watch a movie at all. Different more personal reasons could also influence why a user does not want to watch any movie at the moment. All these different reasons are not captured by the current system and will therefore not be tracked.

2. **No suitable genre**

Even though my database covers the most common movie genres, there still exists a chance that a user is looking for something different. A genre that is a good example for this is the *horror* genre which is not part of any defined genre configurations. While my system contains up to eight different genre nuances (*Drama*, *Romance*, *Comedy*, *Crime*, *Action*, *Thriller*, *Documentary*, *Biography*) none of these go even close in the direction of the *horror* genre. So with my system I might be able to catch the appropriate mood for a *horror* movie, but I will not be able to deliver a suitable recommendation. This could be easily solved by extending the database, and I will talk about this in more detail in the next chapter.

Another interesting result of the analysis is the following. Since both data sets did not change for the two different systems it is interesting to see that the two **Success Rates** differ from each other. As shown in table 3.1 the **Success Rate** for the *Mood Based* system is 8.3% higher than the **Success Rate** of the *Arbitrary* recommender system. Since the data collection was conducted for both groups in the exact same way there might be other influence factors which lead to this deviation.

One factor could be that user get frustrated with the *Arbitrary* recommender system since no movie is presented at the beginning that they would like to watch. This frustration might lead to the scenario that a movie is rejected even though it might have been a suitable candidate if it would just have been displayed earlier. If a user is only slightly in the mood to watch *Forrest Gump* he might click Yes if it is displayed as the first or second recommendation. But if *Forrest Gump* shows up as the last option the user might be so annoyed and disappointed that his emotional state slightly changes in a way that *Forrest Gump* is no suitable candidate anymore. To answer this question might be difficult since these slightly changes in the mood of the user is difficult to measure without asking the user to answer the PANA-Scale every few clicks.

**Is the Mood Based recommender system just a different form of collaborative filtering?**

It could be argued that the *Mood Based* recommender system might be a form of a collaborative system since the ratings of the user are compared against the ratings of other users. In a typical collaborative system (as explained in section 1.2.2) a user gives a rating to a specific object which is then compared to the ratings of other users. In the developed *Mood Based* recommender system the user is also giving a rating for a specific object before it is compared with other users. Even though this might sound similar, the two processes are different in their core. In a collaborative recommender system the recommendation is seen from a recommendation object perspective. In a scenario where object X and Y were liked by user A while
user B only liked object X, the system will prompt object Y to user B. Therefore, the matching process is focused on what object user A has consumed and B does not.

For the Mood Based recommender system I use a more user centered perspective where I am trying to identify the model of the user. In a scenario where object X and Y were liked by user A (in two distinct iterations of the survey) while user B only liked object A, the system will still run the mood of user B through the data model. Therefore, the mood of the user lies at the core of the system what makes it more user and less object centered than a collaborative recommender system.

**Does the Mood Based recommender system actually solve the New User Problem?**

As explained in chapter 1 the New User Problem appears as soon as a user who is completely new to the system tries to get a recommendation. Since it is the first time the user uses the system, it has no information about the preferences of the user and mainly lacks an accurate user model. In the most commonly used methods (collaborative filtering and content based filtering) this problem is "solved" over time when the system gets to know the user. But there is no information available on how long this process takes until the user has rated enough objects so the system can deliver high quality suggestions.

For the Mood Based recommender system the time it takes to overcome the New User Problem can be easily calculated. Since a user is known to the system as soon as he answers the survey at the beginning (which takes approximately three to eight minutes), the user model is created quite quickly. This model is later used to deliver movie suggestions to the user and no additional ratings are required. The ratings that the user will deliver after he has completed the survey are collected and might be used to improve the data model, but will not influence the suggestions at the beginning.

4.3 Ethical Aspects

The Mood Based recommender system uses several highly personal information as the basis of the recommendation process. This raises a couple of ethical questions that will be discussed in the this section.

**Collecting knowledge about the user**

The Mood Based recommender system starts its task by asking the user several questions regarding his mood what can be considered as highly personal information. Personal information that can be used to manipulate the user in several different ways. Marketing or advertisement companies could use this new source of information to deliver their content based on the emotional state of the user. Political parties could misuse this data to manipulate voters during elections by fitting their political message to the well being of the user. This list could be extended with more examples where this data could be used against the will of the user.

Since the protection of personal information is not a new challenge for users and data protection activist, this new type of collected data might bring the discussion on a complete new level. Already today's social media companies like Facebook have a very detailed model about their users' desires, wishes and dreams. They gather all this knowledge about their users by observing their reaction to all the content that is displayed to them. Adding a new source of information to their already extensive user model will make the user even more transparent.
Since this discussion could and has filled already several books, I do not want to go too much into detail with this topic. Nevertheless, I would like to express my general concerns about being able to track the user model back to a real life person. This is also the reason why I decided to use only anonymous information without any user model stored within the system. But since I used a third party data collection tool to create the data model, it can be at least questioned if this data is not being misused.

The system creates a new type of filter bubble

Another ethical aspect that has to be taken into consideration is, that the system might create a new type of filter bubble that is based on the mood of the user. Since I used a machine learning algorithm as the basis of the data model, I am coming across the black box problem that is already described in the previous section of chapter 4. It might be the case that a specific type of mood will always be linked to the same movie genres. A user with a tendency to be nervous might always get prompted a documentation, which will set him into a new mood based filter bubble. It would need further investigations if such a bubble exist and how this might influence the user.

Should the data model be influenced by the developers?

The filter bubble in the previous paragraph leads us to another critical ethical consideration. Since the data model was created by me, I have generally the possibility to influence the outputs of the algorithms. Let’s take e.g. the scenario that a user with a general tendency to depression is looking for a suggestion in the presented system. Am I going to present him the original response of the system even though this could be a heavy drama movie which might enhance the depression? I would have the possibility to try to lift the mood with a lightweight comedy movie but am I actually in the position to influence a users emotions or even health? Because in the given scenario my influence might be beneficial for the user, but this could also be misused. In a worst case scenario purposely presenting heroic war movies to trigger a psychological disoriented user, could have serious implications for the life of a whole community.

4.4 Potential improvements and future research

Improving the mood detection algorithm

For the current system the PANA-Scale was used to detect the mood of the user and the results have shown that this is a suitable method. But to use it as a Real-World system the survey with its 20 questions is quite immense. At this point I would consider two different options on how this can be improved:

1. Change mood detection methodology

   As explained in the chapter 1 the research on the human emotions and moods is quite extensive. Therefore, other mood detection methods and theories exist, that not be as popular as the PANA-Scale but could be also applied for my scenario. Even within the PANA-Scale modified versions exist, that might be able to address my problem in an even better way.
2. **Remove redundant questions**

A second option would be to analyze the answers given in the survey. Since a few questions from the PANA-Scale are quite similar to each other, with non-native English speakers even having trouble to tell the difference, it might be possible that some questions can be removed due to redundancy. A user might for example always give the same values for *nervous* as well as for *jittery*. If it is proven through statistical measures that these are actually redundant, one can be removed without confounding the data model. Cronbach’s alpha as a statistical tool could be used for example to identify redundant questions.

**Improving the data model**

The data model used for the *Mood Based* recommender system was selected through comparing different machine learning techniques. While the *Random Forest* methodology was already quite successful there might still be some room for improvement. There are several optimization approaches for machine learning that could be applied in my scenario. Starting with the selection of the right mathematical functions and operations, optimizing the ratio of training and test data or looking for additional methods. Especially the optimization of the selected methods might be promising in order to receive even more accurate results.

**Recommender system comparison**

One of the flaws of this master thesis is the comparison to other recommender systems. As explained earlier it is difficult to get accurate data for other recommender systems either because this information is not public or the systems are too specific to a domain. Therefore, it is difficult to assess the validity of the *Success Rate* of the *Mood Based* recommender system since there are (except for the *Arbitrary recommender system*) no benchmarks. This could only be solved by new information about the performance of movie recommender systems being published.

**Change the use of the output knowledge**

Another way that might improve the performance of the system would be to change the way the output knowledge is used. So far this knowledge is grouped through genre configurations that always consist out of two genres. The reasons for this were explained in detail in the methods chapter. To get a little more variety into the output knowledge two additional activities could be executed:

1. **Elaborating the distribution within a set of output knowledge**

   In the *Mood Based* system the output knowledge is also grouped into a subset with eight different categories. While this makes sense in order to remove deviations of personal opinions regarding a specific movie, I could now use this knowledge to fine tune my system. By removing the barrier between the data sets I will be able to understand when a user would like to watch *Black Swan* or *Argo*, which both belong to the same genre configuration but differ quite a lot from each other.

2. **Considering only "pure" genres**

   As mentioned before the current system only uses genre configurations for several reasons. In a next step these could be separated in order to receive
two single genres ("drama - comedy" > "drama", "comedy"). After changing the structure of the data model the system will be able to give probabilities for each of the single genres which then can be composed back to a movie configuration with potentially even more than only two genres. Because as explained in the introduction there are a lot of movies with more than just one or two genres. By defining a threshold that tells the system whether to include a genre or not will give a lot of flexibility to the system and will also allow to easily scale up.

As a result of these two activities, I will change the whole structure of the output knowledge. This would move the results from a data bucket like configuration (each genre configuration can be considered as a bucket) towards a more spectral representation where all possible movies will find a place.

**Database size**

One of the biggest potential improvements could be the increase of the database size in both directions, depth and width. Depth connotes that movies from the same genres as I already use in my database are added. This includes movies with exactly the same genre configuration but might also (in dependence on the previously mentioned modifications) include single genres. When talking about the width of the database I talk about genres that are not represented at all in the current database (for example horror or western). By increasing the database size along these two dimensions I will be able to receive more precise results.

**Potential for a hybrid system**

Last but not least the Mood Based recommender system could be combined with other types of systems (like collaborative or content-based systems) to balance out the drawbacks of each approach. This hybrid system requires a complete new setup but could be promising, especially when it comes to overcoming the New User Problem or scaling up the project.
Chapter 5

Conclusion

As explained in section 1.3 Research Problem, it was aimed to close the knowledge gap to understand what a user actually wants and to deliver a real user centered recommendation system. The Mood Based recommender system was developed to close this gap and the results of the previous chapter 3 Analysis have shown that such a system is capable of delivering suitable recommendations. Since the high Success Rate (91.2%) indicates that the system actually converts, it is especially figure 3.5 that delivers us with insights about the performance. It shows that the system performs well on the first three steps and outscores the Arbitrary recommender system. Additionally, both hypotheses were verified which allows me to answer the two research questions:

1. Is it possible to build a movie recommender system based on the mood of a user?
   As shown above it is possible since from a theoretical perspective all required components can be built. The mood of a user can be served as an input knowledge base and the movie database represent an appropriate output knowledge. Also, for the matching process between input and output different methods were available. From a practical perspective it was shown that it can be implemented and the high Success Rate proves that the system is capable of delivering high quality recommendations.

2. How does a mood based movie recommender system perform against a recommender system that suggests movies at random?
   As shown through the frequency analysis as well as the t-Test in chapter 3, the Mood Based recommender system has a better performance than the Arbitrary recommender system. Since I do not have any information about the performance of other movie recommender systems, this question can only be answered in comparison to the Arbitrary scenario.

   As a result I have also verified that the mood of a user is influencing what kind of movie they would like to watch and that it can be used as an additional source for the modeling of the user. This can have some implications for the fields of recommender systems and human computer interaction since it will help to get a better understanding of the users desires and will deliver a clearer and more accurate model of the user. Since it was proven in the context of movies it now can also be applied to other domains which have the potential to show similar outcomes. It will be interesting to see how the chatbot or robotics domain will harvest the knowledge which is derived through mood detection in the future.

   Even though I have shown that the developed system is able to deliver successful recommendations it is still a long way to go to integrate these presented methods into the existing movie recommendation systems and to generate a better user experience. Further research first has to prove that this system also works in the context
of unfamiliar and less popular movies. Additionally, the suggested improvements from chapter 4 need to be tested in order to create a system that can grasp the interests and desires of the user accurately and then match it to the best fitting available object.


Appendix A

Method

A.1 The input of the online questionnaire

The figure displays the first question of the online questionnaire.

![Website PANAS-Questionnaire](image)

**Figure A.1**: Website PANAS-Questionnaire

A.2 The output of the online questionnaire

The first suggestion which is prompted to a user after completing the questionnaire.
A.3 Website Tracking

In the figure A.3 the DataLayer is displayed which is required for the web-tracking using Google Analytics. In the several UserProfile-variables you can see the values which are inserted into the online questionnaire.
Figure A.3: Website DataLayer

Figure A.4 displays the resulting Google Analytics custom report where you can see the values of the UserProfile-variables from the DataLayer as well as the presented outputs. In the third column it is displayed if the system was successful or not. The last column shows how many users received this exact combination of inputs and outputs.

Table A.4: Google Analytics Custom Reports

<table>
<thead>
<tr>
<th>Event Action</th>
<th>Event Label</th>
<th>Event Category</th>
<th>Test Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender1</td>
<td>enthusiastic3</td>
<td>interested</td>
<td>No</td>
</tr>
<tr>
<td>gender1</td>
<td>enthusiastic3</td>
<td>interested</td>
<td>No</td>
</tr>
<tr>
<td>action1</td>
<td>strong</td>
<td>profile1</td>
<td>No</td>
</tr>
</tbody>
</table>
A.4 SQL Query

The query which was used in order to get the number of appearances for each genre configuration. This is excessive since all possible scenarios have to be counted.

```
SELECT SUM(CASE WHEN (genres LIKE '%Drama%' AND genres LIKE '%Comedy%') THEN 1 END) AS 'Drama-Comedy',
        SUM(CASE WHEN (genres LIKE '%Drama%' AND genres LIKE '%Documentary%') THEN 1 END) AS 'Drama-Documentary',
        SUM(CASE WHEN (genres LIKE '%Romantic%' AND genres LIKE '%Drama%') THEN 1 END) AS 'Romantic-Drama',
        SUM(CASE WHEN (genres LIKE '%Action%' THEN 1 END) AS 'Drama-Action',
        SUM(CASE WHEN (genres LIKE '%Crime%' THEN 1 END) AS 'Drama-Crime',
        SUM(CASE WHEN (genres LIKE '%Thriller%' THEN 1 END) AS 'Drama-Thriller',
        SUM(CASE WHEN (genres LIKE '%Adventure%' THEN 1 END) AS 'Drama-Adventure',
        SUM(CASE WHEN (genres LIKE '%Horror%' THEN 1 END) AS 'Drama-Horror',
        SUM(CASE WHEN (genres LIKE '%Music%' THEN 1 END) AS 'Drama-Music',
        SUM(CASE WHEN (genres LIKE '%Family%' THEN 1 END) AS 'Drama-Family',
        SUM(CASE WHEN (genres LIKE '%Biography%' THEN 1 END) AS 'Drama-Biography',
        SUM(CASE WHEN (genres LIKE '%Mystery%' THEN 1 END) AS 'Drama-Mystery',
        SUM(CASE WHEN (genres LIKE '%Western%' THEN 1 END) AS 'Drama-Western',
        SUM(CASE WHEN (genres LIKE '%Sci-Fi%' THEN 1 END) AS 'Drama-Sci-Fi',
        SUM(CASE WHEN (genres LIKE '%Animation%' THEN 1 END) AS 'Drama-Animation',
        SUM(CASE WHEN (genres LIKE '%Sport%' THEN 1 END) AS 'Drama-Sport',
        SUM(CASE WHEN (genres LIKE '%News%' THEN 1 END) AS 'Drama-News',
        SUM(CASE WHEN (genres LIKE '%Reality-TV%' THEN 1 END) AS 'Drama-Reality-TV',
        SUM(CASE WHEN (genres LIKE '%Talk-Show%' THEN 1 END) AS 'Drama-Talk-Show',
        SUM(CASE WHEN (genres LIKE '%Game-Show%' THEN 1 END) AS 'Drama-Game-Show',
        SUM(CASE WHEN (genres LIKE '%Comedy%' THEN 1 END) AS 'Drama-Comedy',
        SUM(CASE WHEN (genres LIKE '%Drama%' THEN 1 END) AS 'Drama')
FROM genres

WHERE (genres LIKE '%Comedy%' AND genres LIKE '%Documentary%') THEN 1 END) AS 'Comedy-Documentary',
        SUM(CASE WHEN (genres LIKE '%Romantic%' THEN 1 END) AS 'Comedy-Romance',
        SUM(CASE WHEN (genres LIKE '%Action%' THEN 1 END) AS 'Comedy-Action',
        SUM(CASE WHEN (genres LIKE '%Crime%' THEN 1 END) AS 'Comedy-Crime',
        SUM(CASE WHEN (genres LIKE '%Thriller%' THEN 1 END) AS 'Comedy-Thriller',
        SUM(CASE WHEN (genres LIKE '%Adventure%' THEN 1 END) AS 'Comedy-Adventure',
        SUM(CASE WHEN (genres LIKE '%Horror%' THEN 1 END) AS 'Comedy-Horror',
        SUM(CASE WHEN (genres LIKE '%Music%' THEN 1 END) AS 'Comedy-Music',
        SUM(CASE WHEN (genres LIKE '%Family%' THEN 1 END) AS 'Comedy-Family',
        SUM(CASE WHEN (genres LIKE '%Biography%' THEN 1 END) AS 'Comedy-Biography',
        SUM(CASE WHEN (genres LIKE '%Mystery%' THEN 1 END) AS 'Comedy-Mystery',
        SUM(CASE WHEN (genres LIKE '%Western%' THEN 1 END) AS 'Comedy-Western',
        SUM(CASE WHEN (genres LIKE '%Sci-Fi%' THEN 1 END) AS 'Comedy-Sci-Fi',
        SUM(CASE WHEN (genres LIKE '%Animation%' THEN 1 END) AS 'Comedy-Animation',
        SUM(CASE WHEN (genres LIKE '%Sport%' THEN 1 END) AS 'Comedy-Sport',
        SUM(CASE WHEN (genres LIKE '%News%' THEN 1 END) AS 'Comedy-News',
        SUM(CASE WHEN (genres LIKE '%Reality-TV%' THEN 1 END) AS 'Comedy-Reality-TV',
        SUM(CASE WHEN (genres LIKE '%Talk-Show%' THEN 1 END) AS 'Comedy-Talk-Show',
        SUM(CASE WHEN (genres LIKE '%Game-Show%' THEN 1 END) AS 'Comedy-Game-Show',
        SUM(CASE WHEN (genres LIKE '%Comedy%' THEN 1 END) AS 'Comedy',
        SUM(CASE WHEN (genres LIKE '%Drama%' THEN 1 END) AS 'Drama')
FROM genres

WHERE (genres LIKE '%Drama%' AND genres LIKE '%Documentary%') THEN 1 END) AS 'Documentary-Drama',
        SUM(CASE WHEN (genres LIKE '%Romantic%' THEN 1 END) AS 'Documentary-Romance',
        SUM(CASE WHEN (genres LIKE '%Action%' THEN 1 END) AS 'Documentary-Action',
        SUM(CASE WHEN (genres LIKE '%Crime%' THEN 1 END) AS 'Documentary-Crime',
        SUM(CASE WHEN (genres LIKE '%Thriller%' THEN 1 END) AS 'Documentary-Thriller',
        SUM(CASE WHEN (genres LIKE '%Adventure%' THEN 1 END) AS 'Documentary-Adventure',
        SUM(CASE WHEN (genres LIKE '%Horror%' THEN 1 END) AS 'Documentary-Horror',
        SUM(CASE WHEN (genres LIKE '%Music%' THEN 1 END) AS 'Documentary-Music',
        SUM(CASE WHEN (genres LIKE '%Family%' THEN 1 END) AS 'Documentary-Family',
        SUM(CASE WHEN (genres LIKE '%Biography%' THEN 1 END) AS 'Documentary-Biography',
        SUM(CASE WHEN (genres LIKE '%Mystery%' THEN 1 END) AS 'Documentary-Mystery',
        SUM(CASE WHEN (genres LIKE '%Western%' THEN 1 END) AS 'Documentary-Western',
        SUM(CASE WHEN (genres LIKE '%Drama%' THEN 1 END) AS 'Drama')
FROM genres

WHERE (genres LIKE '%Documentary%' THEN 1 END) AS 'Documentary'\n```

SELECT CASE WHEN genres LIKE '%Romance%' THEN 1 END AS 'Romance-Western',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%War%' THEN 1 END AS 'Romance-War',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Sci-Fi%' THEN 1 END AS 'Romance-Sci-Fi',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Animation' THEN 1 END AS 'Romance-Animation',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Sports%' THEN 1 END AS 'Romance-Sport',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Noves%' THEN 1 END AS 'Romance-Novel',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Reality-TV%' THEN 1 END AS 'Romance-Reality-TV',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Talk-Show%' THEN 1 END AS 'Romance-Talk-Show',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Game-Show%' THEN 1 END AS 'Romance-Game-Show',
        CASE WHEN genres LIKE '%Romance%' AND genres LIKE '%Short%' THEN 1 END AS 'Romance-Short',
        CASE WHEN genres LIKE '%Action%' THEN 1 END AS 'Action-Crime',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Thriller%' THEN 1 END AS 'Action-Thriller',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Adventure%' THEN 1 END AS 'Action-Adventure',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Horror%' THEN 1 END AS 'Action-Horror',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Music%' THEN 1 END AS 'Action-Music',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Family%' THEN 1 END AS 'Action-Family',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Biography' THEN 1 END AS 'Action-Biography',
        CASE WHEN genres LIKE '%Action%' THEN 1 END AS 'Action-Mystery',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%History%' THEN 1 END AS 'Action-History',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Musical%' THEN 1 END AS 'Action-Musical',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Fantasy%' THEN 1 END AS 'Action-Fantasy',
        CASE WHEN genres LIKE '%Action%' AND genres LIKE '%Adult%' THEN 1 END AS 'Action-Adult',
        CASE WHEN genres LIKE '%Crime%' THEN 1 END AS 'Crime-Crime',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Adventure%' THEN 1 END AS 'Crime-Adventure',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Horror%' THEN 1 END AS 'Crime-Horror',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Music%' THEN 1 END AS 'Crime-Music',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Family%' THEN 1 END AS 'Crime-Family',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Biography' THEN 1 END AS 'Crime-Biography',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Mystery%' THEN 1 END AS 'Crime-Mystery',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%History%' THEN 1 END AS 'Crime-History',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Musical%' THEN 1 END AS 'Crime-Musical',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Fantasy%' THEN 1 END AS 'Crime-Fantasy',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Adult%' THEN 1 END AS 'Crime-Adult',
        CASE WHEN genres LIKE '%Crime%' AND genres LIKE '%Western%' THEN 1 END AS 'Crime-Western',
        CASE WHEN genres LIKE '%Crime%' THEN 1 END AS 'Crime',
        CASE WHEN genres LIKE '%Thriller%' THEN 1 END AS 'Thriller-Crime',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Adventure%' THEN 1 END AS 'Thriller-Adventure',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Horror%' THEN 1 END AS 'Thriller-Horror',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Music%' THEN 1 END AS 'Thriller-Music',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Family%' THEN 1 END AS 'Thriller-Family',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Biography' THEN 1 END AS 'Thriller-Biography',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Mystery%' THEN 1 END AS 'Thriller-Mystery',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%History%' THEN 1 END AS 'Thriller-History',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Musical%' THEN 1 END AS 'Thriller-Musical',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Fantasy%' THEN 1 END AS 'Thriller-Fantasy',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Adult%' THEN 1 END AS 'Thriller-Adult',
        CASE WHEN genres LIKE '%Thriller%' AND genres LIKE '%Western%' THEN 1 END AS 'Thriller-Western',
        CASE WHEN genres LIKE '%Thriller%' THEN 1 END AS 'Thriller',
        CASE WHEN genres LIKE '%Sci-Fi%' THEN 1 END AS 'Sci-Fi',
        CASE WHEN genres LIKE '%Sci-Fi%' AND genres LIKE '%Animation%' THEN 1 END AS 'Sci-Fi-Animation',
        CASE WHEN genres LIKE '%Sci-Fi%' AND genres LIKE '%Action%' THEN 1 END AS 'Sci-Fi-Action',
        CASE WHEN genres LIKE '%Sci-Fi%' THEN 1 END AS 'Sci-Fi',
        CASE WHEN genres LIKE '%Adventure%' THEN 1 END AS 'Adventure-Novel',
        CASE WHEN genres LIKE '%Adventure%' THEN 1 END AS 'Adventure-Game-Show',
        CASE WHEN genres LIKE '%Adult%' THEN 1 END AS 'Adult-Novel',
        CASE WHEN genres LIKE '%Adult%' THEN 1 END AS 'Adult-Game-Show',
        CASE WHEN genres LIKE '%Western%' THEN 1 END AS 'Western-Novel',
        CASE WHEN genres LIKE '%Western%' THEN 1 END AS 'Western-Game-Show',
        CASE WHEN genres LIKE '%Short%' THEN 1 END AS 'Short-Novel',
        CASE WHEN genres LIKE '%Short%' THEN 1 END AS 'Short-Game-Show',
        CASE WHEN genres LIKE '%Talk-Show%' THEN 1 END AS 'Talk-Show-Novel',
        CASE WHEN genres LIKE '%Talk-Show%' THEN 1 END AS 'Talk-Show-Game-Show',
        CASE WHEN genres LIKE '%Game-Show%' THEN 1 END AS 'Game-Show-Novel',
        CASE WHEN genres LIKE '%Game-Show%' THEN 1 END AS 'Game-Show-Game-Show',
        CASE WHEN genres LIKE '%Reality-TV%' THEN 1 END AS 'Reality-TV-Novel',
        CASE WHEN genres LIKE '%Reality-TV%' THEN 1 END AS 'Reality-TV-Game-Show',
        CASE WHEN genres LIKE '%War%' THEN 1 END AS 'War-Novel',
        CASE WHEN genres LIKE '%War%' THEN 1 END AS 'War-Game-Show',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Family%') THEN 1 END) AS 'Horror-Family',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Biography%') THEN 1 END) AS 'Horror-Biography',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Mystery%') THEN 1 END) AS 'Horror-Mystery',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%History%') THEN 1 END) AS 'Horror-History',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Musical%') THEN 1 END) AS 'Horror-Musical',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Fantasy%') THEN 1 END) AS 'Horror-Fantasy',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Adult%') THEN 1 END) AS 'Horror-Adult',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Western%') THEN 1 END) AS 'Horror-Western',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%War%') THEN 1 END) AS 'Horror-War',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Sci-Fi%') THEN 1 END) AS 'Horror-Sci-Fi',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Animation%') THEN 1 END) AS 'Horror-Animation',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Sports%') THEN 1 END) AS 'Horror-Sports',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%News%') THEN 1 END) AS 'Horror-News',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Reality-TV%') THEN 1 END) AS 'Horror-Reality-TV',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Talk-Show%') THEN 1 END) AS 'Horror-Talk-Show',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Game-Show%') THEN 1 END) AS 'Horror-Game-Show',
SUM (CASE WHEN (genres LIKE '%Horror%' AND genres LIKE '%Short%') THEN 1 END) AS 'Horror-Short',
SUM (CASE WHEN (genres LIKE 'Horror') THEN 1 END) AS 'Horror'
FROM 'basic'
WHERE titleType = 'movie' AND startYear < 2019 AND startYear >= 1894
## A.5 Table of all sets

### Table A.1: All sets of movies with their genre label configuration

<table>
<thead>
<tr>
<th>Set</th>
<th>Group</th>
<th>Genre Classification</th>
<th>Movie</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Drama-Romance</td>
<td>Forrest Gump</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>Lady Bird</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>Drama-Crime</td>
<td>Three billboards outside Ebbing, Missouri</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Amelie</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>Drama-Action</td>
<td>Gladiator</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Sixth Sense</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Searching for sugar man</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>Action-Crime</td>
<td>Sherlock Holmes</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Drama-Romance</td>
<td>Call me by your name</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>Birdman</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>Drama-Crime</td>
<td>The Godfather</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Aneurin Hall</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>Drama-Action</td>
<td>V for Vendetta</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Gravity</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Senna</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>Action-Crime</td>
<td>Speed</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>Drama-Romance</td>
<td>Before Sunset</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>3 idiots</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>Drama-Crime</td>
<td>The Shawshank redemption</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Some like it hot</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>Drama-Action</td>
<td>Rise of the planet of the apes</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Donnie Darko</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Man on wire</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Action-Crime</td>
<td>Baby Driver</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>Drama-Romance</td>
<td>Titanic</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>Dead poets society</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>Drama-Crime</td>
<td>Pulp Fiction</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Roman Holiday</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>Drama-Action</td>
<td>Watchmen</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Black Swan</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Grizzly Man</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>Action-Crime</td>
<td>Jack Reacher</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>Drama-Romance</td>
<td>The perks of being a wallflower</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>Downsizing</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>Drama-Crime</td>
<td>Taxi driver</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Vicky donor</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>Drama-Action</td>
<td>Fury</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Argo</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Citizen Four</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>Action-Crime</td>
<td>Deja vu</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>Drama-Romance</td>
<td>Slumdog millionaire</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>The square</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>Drama-Crime</td>
<td>American History X</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Knocked up</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>Drama-Action</td>
<td>Last Samurai</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Jaws</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>The impostor</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>Action-Crime</td>
<td>Batman return</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>Drama-Romance</td>
<td>Head-On</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>The Graduate</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>Drama-Crime</td>
<td>12 angry man</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Wedding crashers</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>Drama-Action</td>
<td>King Kong</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Blood diamond</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Cobain: Montage of Heck</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>Action-Crime</td>
<td>Dredd</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>Drama-Romance</td>
<td>The Notebook</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>Being there</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>Drama-Crime</td>
<td>Drive</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Meet the parents</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>Drama-Action</td>
<td>Dunkirk</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Children of men</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Capturing the friedmans</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>Action-Crime</td>
<td>Enemy of the state</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>Drama-Romance</td>
<td>Before Midnight</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>C.R.A.Z.Y.</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>Drama-Crime</td>
<td>Scarface</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>There's something about mary</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>Drama-Action</td>
<td>Cloud Atlas</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>The hurt locker</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>The salt of the earth</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>Action-Crime</td>
<td>Romon</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>Drama-Romance</td>
<td>The remains of the day</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>Drama-Comedy</td>
<td>Captain Fantastic</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>Drama-Crime</td>
<td>City of god</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>Comedy-Romance</td>
<td>Pretty woman</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>Drama-Action</td>
<td>Equilibrium</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>Drama-Thriller</td>
<td>Captain Philips</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>Documentary-Biography</td>
<td>Life itself</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>Action-Crime</td>
<td>Bronson</td>
</tr>
</tbody>
</table>