UNIVERSITY OF PIRAEUS

A Serendipity Oriented Recommendation System

by

Maria Fritzela

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

In the
Area of Study: Big Data and Analytics
Postgraduate Program “Information Systems and Services”
Department of Digital Systems

January 2023

Supervised by Associate Prof. Maria Halkidi

Abstract

This thesis explores the concept of serendipity and the challenges associated with incorporating into recommendation systems. Through the study of related research in the field and the detailed analysis of the available dataset, three features were generated to be used for predicting a defined serendipity score based on real user feedback in movie recommendations. A Random Forest model was selected, trained, and evaluated on the generated dataset. The results offer insights into the potential of using predictive models for reranking recommendation lists to maximize serendipity for users. Through this work, it is hoped to provide a contribution towards a better understanding of serendipity and offer a starting point for future work in this area.

Η διπλωματική εργασία διερευνά την έννοια της «ευχάριστης έκπληξης» (serendipity) και τις προκλήσεις που σχετίζονται με την ενσωμάτωσή της σε συστήματα συστάσεων. Μέσω της μελέτης σχετικής ερευνητικής βιβλιογραφίας στον τομέα και της λεπτομερούς ανάλυσης του διαθέσιμου συνόλου δεδομένων, ορίστηκαν τρία χαρακτηριστικά ώστε να χρησιμοποιηθούν για την πρόβλεψη ενός δείκτη «ευχάριστης έκπληξης» ο οποίος βασίζεται σε πραγματικά σχόλια χρηστών σε προτάσεις ταινιών. Το μοντέλο Random Forest επιλέχθηκε, εκπαιδεύτηκε, και αξιολογήθηκε στο δημιουργηθέν σύνολο δεδομένων. Τα αποτελέσματα προσφέρουν πληροφορία σχετικά με την προοπτική χρήσης μοντέλων πρόβλεψης για την ανακατάταξη αποτελεσμάτων λιστών συστάσεων με στόχο τη μεγιστοποίηση της ευχάριστης έκπληξης των χρηστών ενός συστήματος. Μέσω αυτής της εργασίας, ελπίζεται να προσφερθεί μια συμβολή προς την καλύτερη κατανόηση της «ευχάριστης έκπληξης», και επιπλέον η εργασία να αποτελέσει σημείο εκκίνησης για μελλοντική έρευνα στο συγκεκριμένο πεδίο.

Acknowledgements

I would like to express my gratitude to Professor Maria Halkidi, for her guidance and support throughout the completion of this thesis, and for providing me with the resources and opportunities to investigate and work on this fascinating and complex subject. I would also like to thank my family and friends for their encouragement and motivation during the process of my research.

List of Figures

List of Tables

List of Equations

CONTENTS

Chapter 1. INTRODUCTION

1.1 PROBLEM STATEMENT AND RESEARCH GOAL

Serendipity is an elusive concept in the field of recommendation systems. It describes the occurrence of unexpected and surprising recommendations that lead to discovery, learning and broadening of preferences. Recent research has been attempting to define serendipity and incorporate it into recommendation algorithms, with the goal of providing a more engaging, personalized and satisfying user experience. This is essential in today's digital world, where users are faced with an overwhelming amount of information and choices. In this study, the possibility of creating a predictive model for serendipity in movie recommendations based on real user feedback is explored.

The contents of this thesis are organized into three chapters. In this first chapter, an introduction to recommendation systems and their different types is provided. In addition, the evaluation of recommendation systems via beyond accuracy objectives is discussed, along with an overview of examples of related academic work on such systems. Then, in the second chapter, the overall goal, architecture, and methodology is presented, and programming languages and tools used are listed. Chapter 3 focuses on the dataset analysis and feature engineering process. It offers a detailed description of the methods used to analyze the dataset and extract relevant features for the specific objective, as identified through the assessment of related research. Chapter 4 describes the implementation of the model, including the methodology, the training and testing techniques, and the results obtained. Finally, the conclusions are presented and possible future work for improvement is proposed. It is hoped that this thesis can provide further insights and future directions for this area of research.

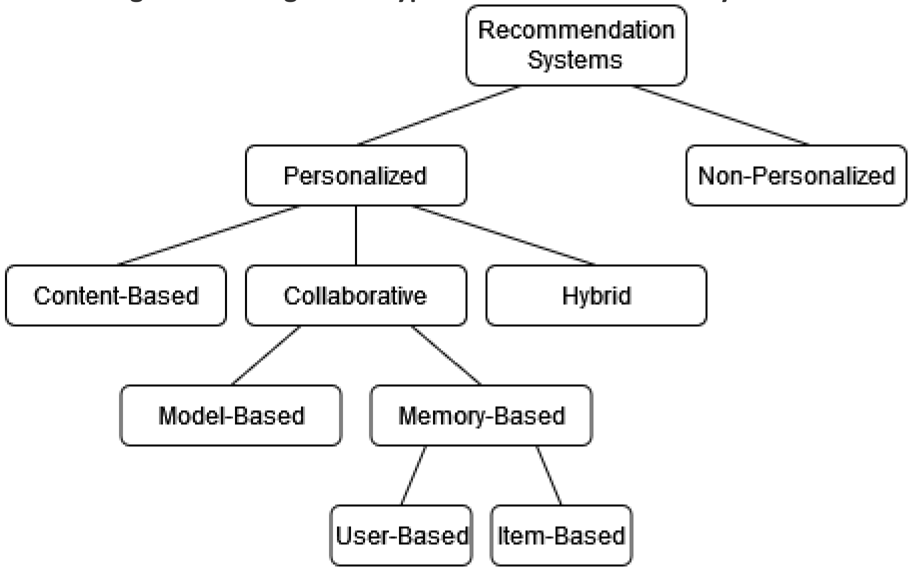
1.2 ABOUT RECOMMENDATION SYSTEMS

Recommendation systems are pieces of software that use algorithms to suggest products to users in accordance with their preferences, interests, and previous behavior. They can provide a more personalized experience for users and allow them to quickly find what they are looking for in the overwhelmingly large amounts of available information. Online product and media recommendation engines (like Netflix or Amazon), and social networking recommendation systems (like LinkedIn) are a few examples of recommendation systems that exist. They have become increasingly popular in recent years as more and more businesses notice their value in increasing user engagement and satisfaction.

1.2.1 Types of Recommendation Systems

Recommendation Systems can be non-personalized, meaning they suggest what is generally popular and relevant to the entire user base, or personalized to attempt to cater to the taste of each user separately. Personalized recommendation systems can be classified into categories based on the methods they employ to generate recommendations. The three main types are: content-based filtering, collaborative filtering, and hybrid methods [1].

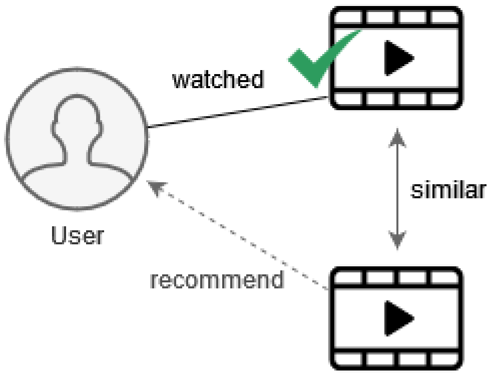
Figure 1.1: Diagram of types of recommendation systems



1.2.2 Content-Based Filtering

Recommendation systems based on Content-Based Filtering use the properties of the items to provide suggestions, considering the previous choices of the user. Essentially, their goal is to help the user find items with similar characteristics to the ones they enjoyed in the past. For example, a user is likely to receive a suggestion for a comedy movie if they have watched other comedy movies in the past. For readers interested in exploring further the topic of content-based filtering, see [2] and [3]. Although content-based filtering was initially applied as a standalone technique, it has recently been more commonly used in combination with the other methods, to capture user preferences beyond explicitly defined characteristics.

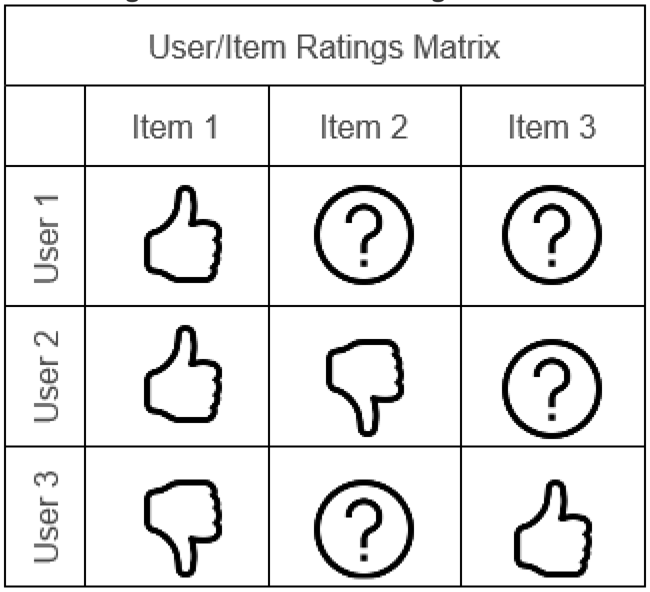
Figure 1.2: Content-based Filtering



1.2.3 Collaborative Filtering

Collaborative Filtering (CF) techniques rely exclusively on user ratings of items to produce recommendations. These ratings can be explicit, meaning the user has provided feedback for a specific product (e.g., in the case of a 5-star rating system), or implicit signals collected by researchers, such as click-through rates. This behavior is typically collected in a user-item ratings matrix which is then utilized by the CF algorithm to attempt to predict the unknown values.

Figure 1.3: User-Item Ratings Matrix



User/Item Ratings Matrix

	Item 1	Item 2	Item 3
User 1	👍	?	?
User 2	👍	👎	?
User 3	👎	?	👍

In the next subsections, a brief overview of the categories of CF techniques is presented. For a comprehensive survey on Collaborative Filtering, the interested reader may consult [4].

1.2.3.1 Model-Based

Model based techniques involve the design and development of models trained on the user-item ratings matrix to discover underlying patterns in the data and use that knowledge to generate predictions. One of the most common model-based approaches is Matrix factorization techniques, which were popularized by Simon Funk during the "Netflix Prize" challenge [5].

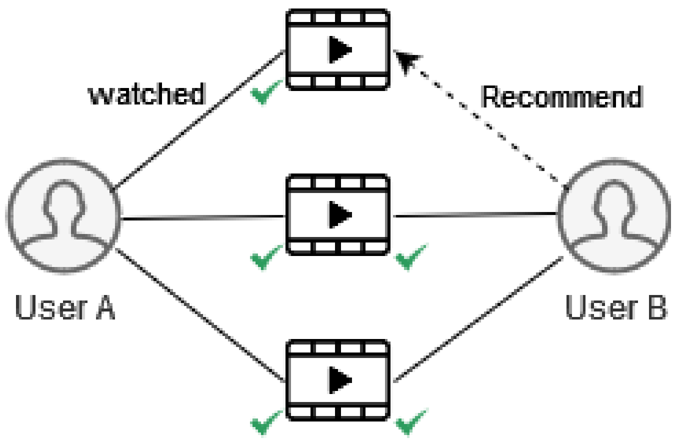
1.2.3.2 Memory-Based

Memory-based techniques use past user rating data to compute the similarity between users or items. User-Based and Item-Based are the two main kinds of memory-based collaborative filtering algorithms.

User-Based

The User-Based method divides users into groups (neighborhoods) based only on ratings, in such a way that users with matching tastes are placed in the same neighborhood. The core principle is that users who liked similar things in the past will continue to have similar preferences in the future. As a result, a user's suggestions are based on products that other users who are similar to them liked. The technique is illustrated in the image below: Because user A and user B are found to have many movies in common, they are considered "neighbors", and user B is recommended a movie that user A watched. In [6], Hamidreza Koohi and Kourosh Kiani use the fuzzy clustering technique to create a user-based collaborative filtering system, test it and compare it to other clustering methods on the MovieLens dataset.

Figure 1.4: Collaborative Filtering



Item-Based

Item-based collaborative filtering was first launched in 1998 by Amazon as described in their 2003 publication [7], and since then it has become one of the standard methods for collaborative filtering. The Item-Based technique creates neighborhoods of items with similar rating distributions. This is accomplished by creating an item-item matrix and calculating the similarity between items based on how they have been rated by users.

1.2.4 Hybrid Methods

Hybrid Systems combine two or more of the approaches mentioned, to improve performance and to overcome limitations of the individual methods such as the “Cold start problem” [8]. The idea is to leverage the strengths of each of the algorithms to mitigate their weaknesses, resulting in a better recommendation system when they are combined effectively.

There are several techniques used by researchers to combine recommendation algorithms when developing a hybrid system, which can be used standalone or in tandem with each other. According to [9], these are: weighted, switching, cascade, mixed, feature-combination, feature-augmentation, and meta-level hybridization. In weighted hybridization, results of different algorithms are combined by calculating a weighted linear formula of their individual scores. For switching hybridization, as the name implies, the system switches between different recommendation approaches based on a heuristic criterion. In cascade hybridization, results are iteratively refined by multiple algorithms, using the results of one as input to the next until the final list of recommendations is produced. While in mixed hybridization, different algorithms are run in parallel, and their outputs are combined to generate the list of recommendations. In feature-combination hybrid systems, one or more algorithms are used to create a single set of features which are then used by another algorithm to produce recommendations. Feature-augmentation systems work similarly, but features from one or more recommendation algorithms are used to augment the features used by another one. Finally, in meta-level hybrid recommenders, the internal model produced by an algorithm is used as input for another.

A short description of each technique was presented in this section; For a detailed analysis, as well as examples of relevant systems that employ each technique, see the relevant publication by F.O. Isinkaye et al. [9].

1.3 Evaluating Recommendation Systems: Metrics Beyond Accuracy

For a long time, academic research on recommendation systems has been centered around the objective of accuracy, specifically on how close the system's predicted ratings are to the users' actual ratings.

However, over-focusing on accuracy can create the effect of a "filter bubble", where recommendations are too similar to what the user has already seen, causing user dissatisfaction and inability to discover new items and broaden their preferences [10]. This phenomenon, also known as the over-specialization problem, has caused researchers to shift their focus to new metrics for producing recommendations [11]. In their article [12], Kaminskas and Bridge provide an overview of the most frequently publicized objectives in recommender systems research beyond accuracy: diversity, serendipity, novelty, and coverage.

In this study the focus is serendipity, so in the following section its definition and its link to other concepts are explored. It is beneficial for recommendation systems to aim to produce serendipitous recommendations, as it has been found that serendipity has a direct influence on user satisfaction and, in the context of commerce, purchase intention [13].

1.3.1 Defining Serendipity

The word "serendipity" originates from the Persian tale "The Three Princes of Serendip", a story about three princes who set out on a quest to discover the truth about their father's death. Throughout their journey, the three princes continuously find unexpected solutions to the challenges they face and gain a deeper understanding of the world and its workings. They eventually return home, where they are able to unravel the truth behind their father's passing and restore peace to their kingdom.

The research community in recommendation systems has been attempting to define serendipity in multiple ways, and there is no agreement on the definition of serendipity [14]. In their 2021 publication [15], Reza Jafari Ziarani & Reza Ravanmehr analyze the contextual convergence of definitions of serendipity in recommendation systems through a systematic literature overview, using a collection of research papers from 2013 to 2019. They conclude that the definition of a serendipitous recommendation should include the concepts of unexpectedness, novelty and relevance, and propose the below equation, where u is the target user in a U set of users and I represents a set of items:

Equation 1.1

$$\forall u \in U, I_{serendipity} = I_{unexcpected} \cap I_{novel} \cap I_{relevant}$$

This definition is selected to be used in this study: A serendipitous item is relevant, novel and unexpected to the user.

1.3.1.1 Relevance

A relevant item is defined as an item that is interesting, useful, or enjoyable to the user. Depending on the application domain, different explicit or implicit user signals can be used to determine whether an item is relevant for a certain user. In the case of movie recommendations, one might utilize the rating that a user has provided for a movie (the higher the rating, the more relevant the movie). According to [16] ratings can be a good predictor for serendipity: “In fact, the higher the rating the more likely an item to be perceived serendipitous.”

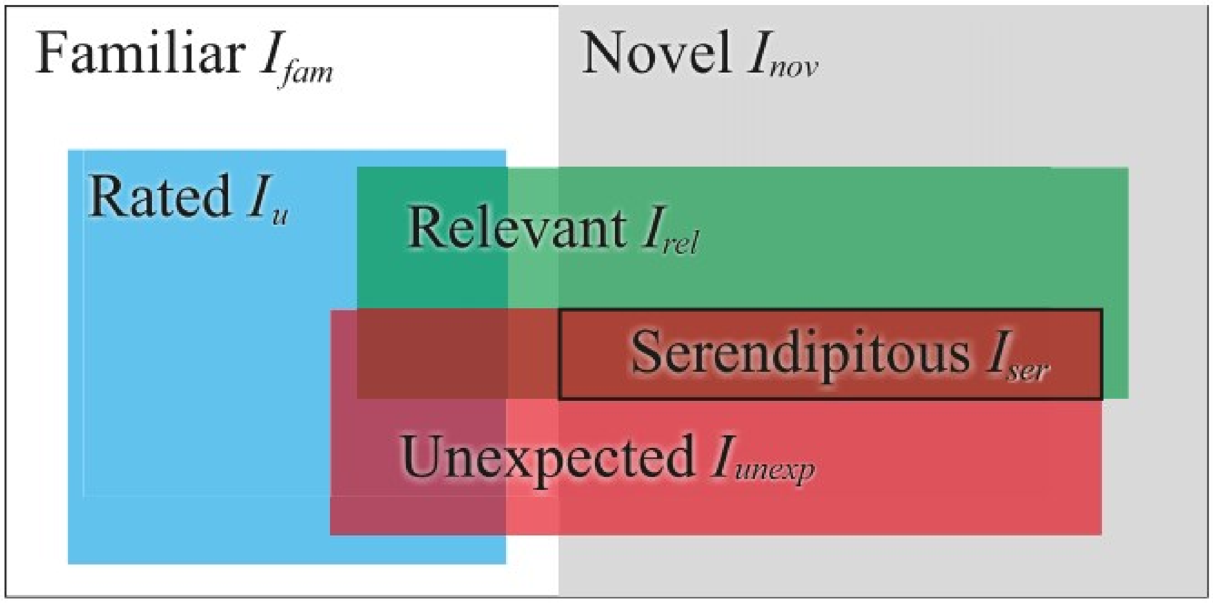
1.3.1.2 Novelty

A novel item is an item that the user is not aware of. Novelty is usually defined in a user-independent way, using the concept of popularity [12], since users are more likely to be familiar with an item that is popular. The novel item could be relevant or not relevant to the user’s interests, thus novelty is not always coupled with serendipity. However, a serendipitous item, as defined, is always novel to the user. It has been observed that novelty creates unexpectedness, which in turn promotes serendipity ([17], [13]).

1.3.1.3 Unexpectedness

Unexpectedness is another concept which has not been clearly defined and measured by researchers using a specific strategy ([12], [14], [18]). For the purpose of this study, an unexpected item is considered one that the user does not anticipate to be recommended to them, because it is different from the ones they usually choose.

Figure 1.5: Euler Diagram of items from a user’s point of view at a given moment of time (image source: [14])



1.4 Related Work: Serendipity-Oriented Recommendation Systems

In this section, other studies on serendipity-oriented recommendation systems will be explored. Serendipity oriented algorithms can be classified into three types: reranking, modifications, and novel algorithms [19].

1.4.1 Reranking Serendipity-Oriented Algorithms

Reranking algorithms attempt to improve serendipity by reordering the output of accuracy-oriented algorithms, to prioritize items that are more likely to be unexpected and/or novel to the user. In [19], Denis Kotkov et al. propose the Serendipity-Oriented Greedy (SOG) algorithm, a reranking algorithm based on the Topic Diversification (TD) algorithm [20]. The SOG is a kind of greedy reranking algorithm, it works by rearranging a recommendation list provided by another algorithm. In this case, the original list of candidates is generated by a typical accuracy-oriented algorithm. The candidates are assigned a score, which is calculated as a linear combination of parameters the authors defined. Each parameter is defined to correspond to an element of serendipity: relevance, diversity, dissimilarity to user profile, and unpopularity. Finally, the set items are sorted based on their score, and the top items are recommended to the user. SOG increases serendipity of suggestions through feature diversification and was found to outperform other serendipity-oriented algorithms of the time, as evaluated from experiments on the MovieLens "Serendipity2018" dataset [16].

1.4.2 Serendipity-Oriented Modifications

Serendipity-oriented modifications are changes made to existing accuracy-oriented recommendation algorithms to incorporate serendipity elements into their output. Anup Anand Deshmukh et al. [21] present such an algorithm in two parts: the Spherical K-Means Parallel (SPKM) algorithm for clustering users, an extension of K-means, and the Serendipitous Clustering for Collaborative Filtering (SC-CF), a modified clustering technique to recommend serendipitous items. They define that serendipitous items should be unexpected and relevant to the user, while also broadening their preferences. As a first step in their process, the SPKM algorithm is used to cluster the users into groups of similar taste, through the calculation of cosine similarity on the user-movie rating matrix, where each user is represented by a vector of movie ratings they have provided. Through Lloyd's iterations a final set of user clusters is produced, to be applied as an argument for the next step, the SC-CF algorithm. The SC-CF algorithm which is in charge of assigning each user to "serendipitous clusters". It considers the users preferences of genre of movie, as well as their preference in directors-cast, to assign the user in a cluster with users of dissimilar genres (to capture diversity), but of similar director-cast preferences (to maintain relevance). The importance of each of these two parameters (genre diversity vs. director-cast relevance) can be customized by assigning different weights to each of them. Finally, the ratings of movies are predicted utilizing the neighborhood of the user, which incorporates both the conventional and the "serendipitous cluster". Movies with a predicted rating higher than a defined threshold are recommended to the user.

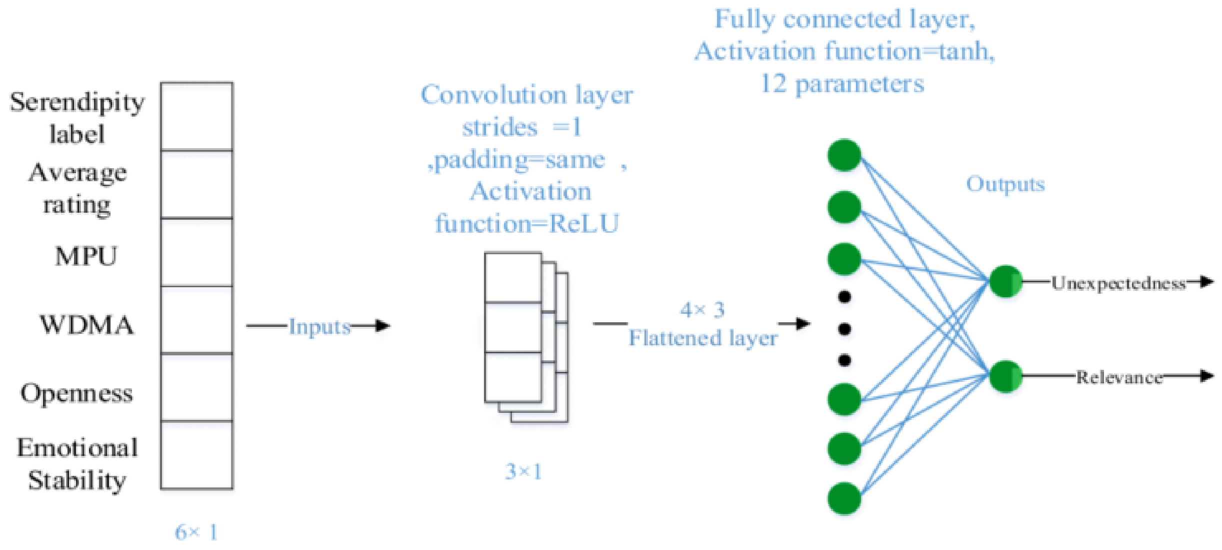
In another relevant study [22], Yongjian Yang et al. propose the Concise Satisfaction and Interest Injection (CSII) method, which can be used to remedy the sparsity issue and enhance the performance of classic CF recommendation algorithms. The authors define and detect serendipitous items using the concepts of pre-interest and post-satisfaction: A serendipitous item is one that the user was not initially interested in (low pre-interest) but was ultimately relevant and useful to them (high post-satisfaction). CSII uses the user-item rating matrix to find serendipitous items in the set of unrated items, and then enriches the user-item rating matrix with this information (creating an "intensified" matrix). This matrix can then be used by any type of CF recommendation algorithm to produce more accurate and serendipitous recommendations.

1.4.3 Novel Serendipity-Oriented Algorithms

Novel serendipity-oriented recommendation algorithms are innovative approaches that are not derived from any widely used accuracy-oriented algorithms and do not utilize their generated relevance scores. For example, in [23], Noa Tuval suggests a novel method based on graph theory to discover serendipitous items by their content. In their study, serendipitous items are unexpected or surprising items, defined using the minimum distance from the items the user expects to be recommended (not the ones they have been already exposed to as seen in [24]). The greater the distance between an item and the items the user is expecting to be recommended, the more surprising that item is considered to be for the user. The distance between items is based on features derived from their content (e.g. genre). The proposed "Resolving Sets Model" is a mathematical model for user model representation, which utilizes the concept of resolving sets in graph theory to produce serendipitous recommendations. The author provides an initial demonstration of the approach using the MovieLens "Serendipity2018" dataset [16].

Recent research has begun exploring the use of Deep Neural Networks to create recommendation systems aiming for serendipity. In a 2021 publication [25], Ziarani and Ravanmehr combine a Convolutional Neural Network (CNN) with the Particle Swarm Optimization (PSO) algorithm to produce serendipitous recommendations. They define serendipity as the intersection of 3 concepts: novelty, unexpectedness, and relevance. Their approach also utilizes the concept of focus shift points that consist of two parameters: unexpectedness and relevance: they consider these the "serendipity establishment factors" [26]. For this reason, as a first step in their process, a CNN with an input of 6 features extracted from the dataset is utilized to predict the focus shift points of a given user. The architecture of the network is shown in the figure below.

Figure 1.6: The CNN architecture for predicting Unexpectedness and Relevance (image source: [25])


Serendipity label
Average rating
MPU
WDMA
Openness
Emotional Stability
6× 1
Inputs
Convolution layer strides =1 ,padding=same , Activation function=ReLU
3×1
4× 3 Flattened layer
Fully connected layer, Activation function=tanh, 12 parameters
Outputs
Unexpectedness
Relevance

The next step is to use the PSO algorithm to create a list of recommendations near these focus shift points. The focus shift points are pre-filtered to speed up the PSO execution. Lastly, the list is reranked using the Serendipitous Personalized Ranking (SPR) method, generating the final list of recommendations. Through this study, it is evident that deep learning techniques are showing promising results in the area of serendipitous recommendations, however the system was not tested in an online environment to assess actual user satisfaction and amount of captured serendipity.

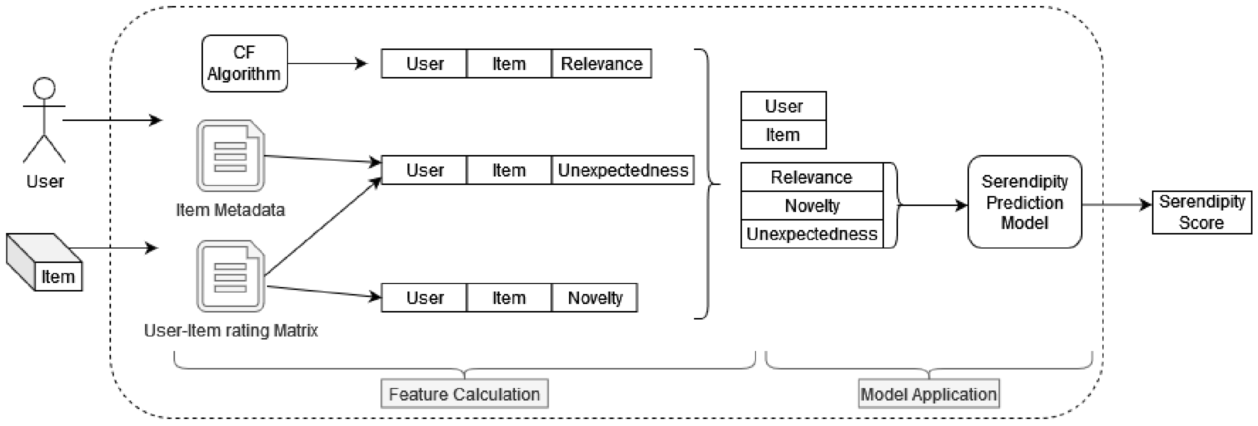
Chapter 2. SYSTEM ARCHITECTURE & METHODOLOGY

2.1 SYSTEM ARCHITECTURE

The focus of the thesis is to create a system that can improve the serendipity of recommendations. This aim is approached through a machine learning perspective, specifically though the development of a model that can predict a metric that quantifies serendipity based on real user feedback. The diagram of Figure 2.1 illustrates the envisioned architecture of the system. There are two inputs, the user and a candidate recommendation item, and one output, a serendipity score for the user-item pair, that quantifies how serendipitous the recommendation of that item is to the user. During the first step, labeled "Feature Calculation" the system calculates the necessary features for the prediction of serendipity, by employing mathematical methods on the user-item rating matrix and the item metadata, and using a pre-trained collaborative filtering algorithm. Once these features are calculated, the serendipity prediction model is used to provide the output serendipity score.

The system could be used to re-rank recommendations produced by an accuracy oriented algorithm: given a recommendation list and a user, the list items can be reordered based on their serendipity scores. Alternatively, another possible use would be to determine serendipitous items for a given user from a set of candidates, by running each candidate item through the system to calculate their serendipity scores and filtering them based on a selected threshold.

Figure 2.1: System Architecture



2.2 METHODOLOGY

In this section, the steps followed in this study to achieve the research goal are documented, as well as the tools (programming languages and libraries) used for their realization.

2.2.1 Dataset Analysis & Feature Engineering

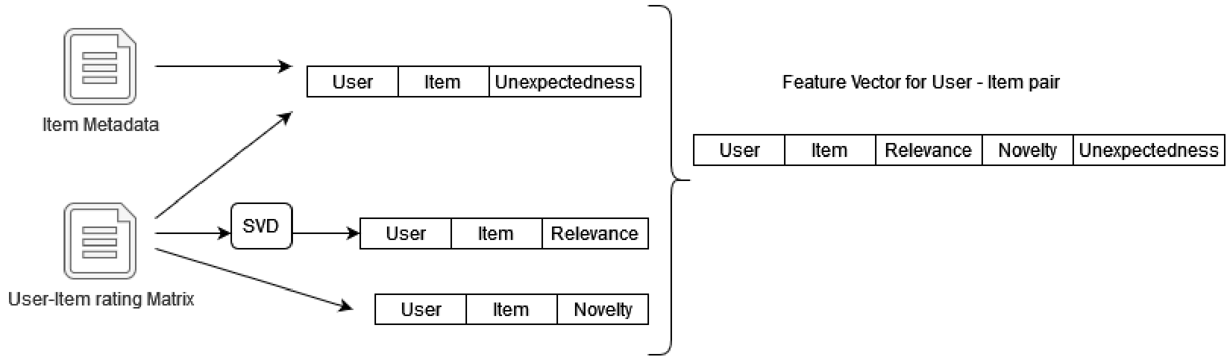
The first step to the creation of the model is to select a dataset that includes user feedback on serendipity related questions, as well as information about the users and the recommended items. The existence of these variables is necessary to create features that can be used for training the serendipity predicting model for recommendations. However, before feature engineering is performed, it is useful to analyze the selected dataset, to gain a deeper understanding of the data. This analysis aids in identifying the distribution of different variables, detection of outliers or missing values, and determining which parts of the dataset are relevant for achieving the research goal. In addition, through the analysis of user feedback the "Serendipity Score" can be defined, which is a metric of the serendipity of a recommendation of an item to a given user. The model will attempt to predict this Serendipity Score.

The next step is feature engineering. This is the process of choosing and constructing a set of features to be used by the serendipity predictive model. For the case of this study, this set of features should be designed so that each variable quantifies one of the factors that affect or elicit serendipity. Through the literature review done in the previous chapter (1.3.1 Defining Serendipity) three main concepts are key to serendipitous items: Relevance, Novelty, and Unexpectedness. Thus, three features are constructed for the prediction of serendipity.

- Relevance Feature – quantifies the recommended item's usefulness to a specific user.
- Novelty Feature – measures the unpopularity of the item (user independent); A user is more likely to not be aware of a less popular item.
- Unexpectedness Feature – evaluates the degree to which a recommendation differs from the users' expectations.

They are defined in the context of the selected dataset but are generic enough to be used with recommendation datasets of other domains.

Figure 2.2: Production of the feature vector


Item Metadata
User-Item rating Matrix
SVD
User | Item | Unexpectedness
User | Item | Relevance
User | Item | Novelty
Feature Vector for User - Item pair
User | Item | Relevance | Novelty | Unexpectedness

2.2.2 Implementation

The implementation phase includes the model selection, tuning, training and evaluation. Because of the limited number of user feedback available the training-testing dataset is small, and the K-Fold Cross Validation method is best suited for training and evaluating the model. Fine-tuning of model parameters can be performed using Grid Search hyperparameter tuning. Through experimentation with different machine learning regression models, the most performant is selected to be presented in the context of the study. The model's performance evaluation is based on Mean Absolute Error (MAE) and Root Mean

Squared Error (RMSE). The process of the implementation of the model and an analysis of the techniques employed is analyzed in detail in Chapter 4. The chapter also presents the results of the experiment.

2.2.3 Libraries and Programming Languages

All code during this study was developed using the open-source Python programming language (version 3.8.10). It is one of the most popular choices for programming in the data science and machine learning community, due to its simplicity, readability, and its large number of available libraries to facilitate development.

The study also used the following libraries, which are commonly used in similar projects: NumPy, Pandas, Scikit-learn, and Seaborn. NumPy provides support for complex numerical computations, while Pandas can be used to read, manipulate, and clean data files. Seaborn is a data visualization library that can create plots and charts, used extensively in the data analysis step. For the development of the model, the Scikit-learn (sklearn) library is used, a very popular machine learning library that offers tools for various tasks including classification, regression, and clustering.

Chapter 3. DATASET ANALYSIS & FEATURE ENGINEERING

3.1 DATASET SELECTION

For the purpose of creating a recommendation system that aims to serendipity, it is necessary to have a dataset where users have provided feedback on elements of serendipity for a given item that was recommended to them. There are not many publicly available datasets that include user feedback on serendipity. This is possibly because, as mentioned in section 1.3.1 Defining Serendipity, the scientific community has not reached a consensus on a formal definition for the term, combined with the fact that it is an emotional response that is not easy to quantify and evaluate. Also, serendipity in recommendation systems is a topic with limited public datasets available. After extensive research, by the time this text is written, only two datasets have been located: one by the GroupLens research lab of the University of Minnesota [16], and one by the Chinese mobile e-commerce platform Taobao and the Hong Kong Baptist University [13].

The MovieLens "Serendipity 2018" dataset by GroupLens is the product of a user survey conducted in April 2017 on the MovieLens website (movielens.org). It was used, in a slightly different version than the final publicly available version, in the publication "Investigating Serendipity in Recommender Systems Based on Real User Feedback" by Denis Kotkov et al in 2018 [16].

The Taobao Serendipity dataset was collected via user survey from Dec 21st, 2017, until Jan 11th 2018, and was first mentioned in the 2019 publication "How Serendipity Improves User Satisfaction with Recommendations? A Large-Scale User Evaluation" by Li Chen et al. [13] The data is currently hosted on a public GitHub repository since June 2020.

The dataset selected for this study is the "Serendipity 2018" dataset by GroupLens. This is because it is a dataset which includes information on the behavior of users and on characteristics of the items recommended (whereas in the Taobao dataset although more user answers are available, all information on users and items have been anonymized).

3.2 ABOUT THE DATASET COLLECTION

The MovieLens "Serendipity 2018" dataset was the first publicly available dataset which included user feedback on serendipitous movies [16]. The dataset was collected through a survey conducted in April 2017 on users of MovieLens, a research website run by the GroupLens Research team at the University of Minnesota. The website works as a movie recommendation service, users can create an account to rate movies they have watched on a scale of 0.5 to 5 stars (0.5-star granularity) and then receive personalized movie recommendations based on their provided ratings. Users can also perform other actions, like creating watchlists or assigning keywords (tags) to a specific movie.

3.2.1 Survey Description

In the publication released with the dataset [16], Denis Kotkov et al describe in detail how the survey was conducted: A selection of users were contacted in the online survey to answer questions on 5 movies they had rated recently (3 months or less). The researchers opted to ask about movies which were unpopular (i.e., they did not have many ratings in the system) and that the user had enjoyed watching (i.e. had rated with at least 3.5 stars). For each movie, the user had to answer one question and rate 8 statements using the 5-point Likert scale, a psychometric response scale ranging from 1 to 5 where: (1) Strongly disagree; (2) Disagree; (3) Neither agree nor disagree; (4) Agree; (5) Strongly agree. The option "NA" was also available for "don't remember". The statements are presented in the following table.

Table 3.1

#	Statement	Description
S1	The first time I heard of this movie was when MovieLens suggested it to me.	Strict Novelty
S2	MovieLens influenced my decision to watch this movie.	Motivational Novelty
S3	I expected to enjoy this movie before watching it for the first time.	Unexpectedness (relevance)
S4	'This is the type of movie I would not normally discover on my own; I need a recommender system like MovieLens to find movies like this one.	Unexpectedness (find)
S5	This movie is different (e.g., in style, genre, topic) from the movies I usually watch.	Unexpectedness (implicit)
S6	I was (or, would have been) surprised that MovieLens picked this movie to recommend to me.	Unexpectedness (recommend)
S7	I am glad I watched this movie.	Satisfaction
S8	Watching this movie broadened my preferences. Now I am interested in a wider selection of movies.	Preference Broadening

The final dataset has been enhanced to include some additional responses that were submitted after the dataset of the linked publication had been generated; it includes answers of 481 users on 1,678 movies. Note that not all users provided replies on all the questions for all movies.

3.3 FILE DESCRIPTIONS AND ANALYSIS

The "Serendipity 2018" dataset as it is hosted on the GroupLens website at the time this text is written includes a total of 6 files: `answers.csv`, `movies.csv`, `recommendations.csv`, `tag_genome.csv`, `tags.csv` and `training.csv`. In this section, for the sake of brevity, descriptions will be provided only for the files that were used for the purpose of this study. The three files selected for this study were:

`movies.csv`, `training.csv`, and `answers.csv`. They can be seen in the table below, along with a short description of their contents**.**

Table 3.2

File Name	**Description**
movies.csv	Movie metadata for 49,174 movies
training.csv	Ratings given by users for movies in the period of November 2009 to January 2018, excluding ratings from answers.csv
answers.csv	Answers of users to a survey conducted in April 2017 with questions about 5 unpopular movies they watched recently (3 months or less) and rated at least 3.5 starts

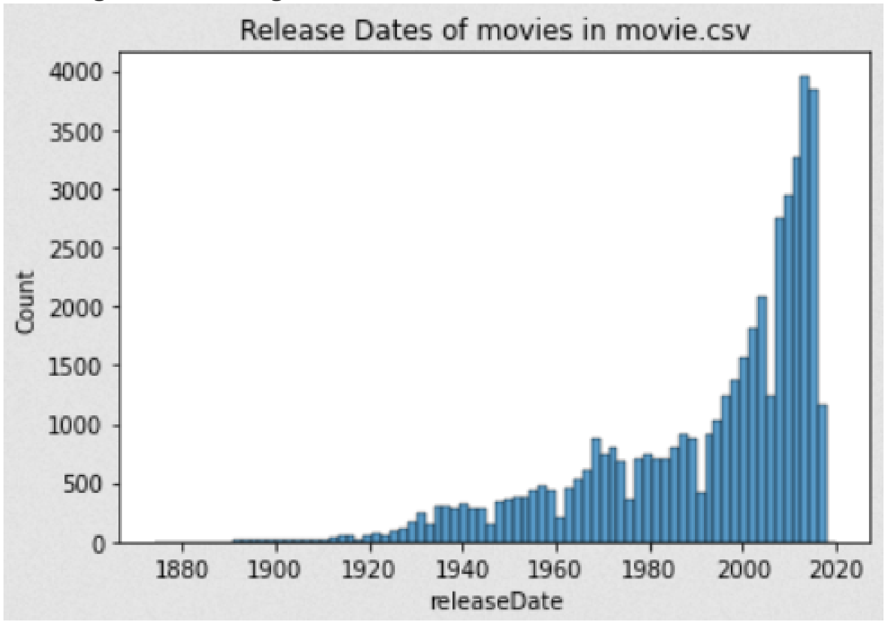
3.3.1 Movies.csv

The `movies.csv` file contains information about the movies in the database. There are a total of 49,174 movies in the database. The file contains the following fields:

- **movieId** – a unique identifier for that movie
- **title** – movie title
- **releaseDate** – the date when the movie was released
- **directedBy** – directors, separated by commas (`,`)
- **starring** – cast, separated by commas (`,`)
- **imdbId** – unique id from IMDB website
- **tmdbId** – unique id from The movie DB website
- **genres** – genres separated by commas (`,`)

The release date field is of the string format “YYYY-MM-DD”, and there are invalid dates (e.g. “0000-00-00”). For this study, only the release year is of interest, so a custom function is created to preprocess them, keeping only the year as an integer, and filtering out invalid dates (dates before the year 1880). There were 49,174 dates before cleaning. Using this parse function to filter them, the following histogram is created.

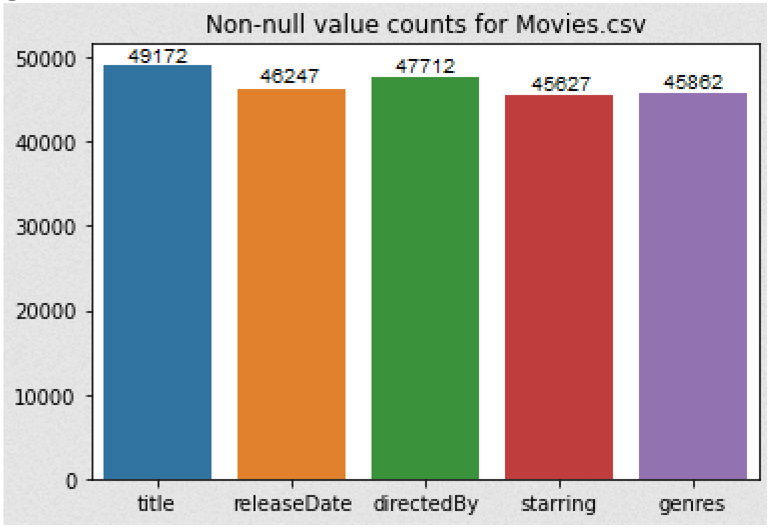
Figure 3.1: Histogram of release dates of movies in movie.csv



It is evident that the number of movies released per year has been rapidly increasing. After filtering out the invalid dates there were 46,247 movies with known release dates.

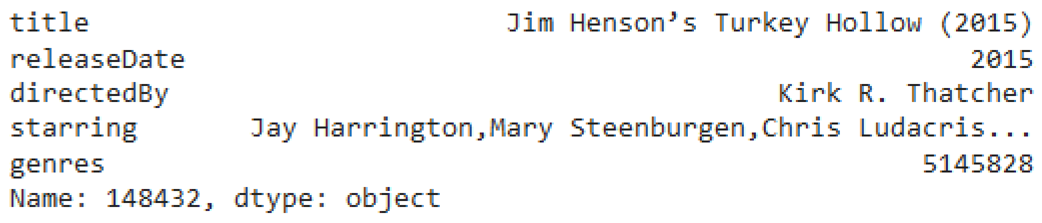
It is not just the releaseDate field that has missing information for some of the movies. In the bar chart below, the number of non-null values are summarized per column:

Figure 3.2: Non-null value counts of metadata fields of movies.csv



When exploring the 'genres' column, it is discovered that there are 3,312 movies with null values for their genres, as well as 6 movies with invalid genres (5 with a series of arbitrary numbers, shown in the image below, and one movie with the genre "Devon Libran''). Additionally, note that a movie can belong to multiple genres simultaneously.

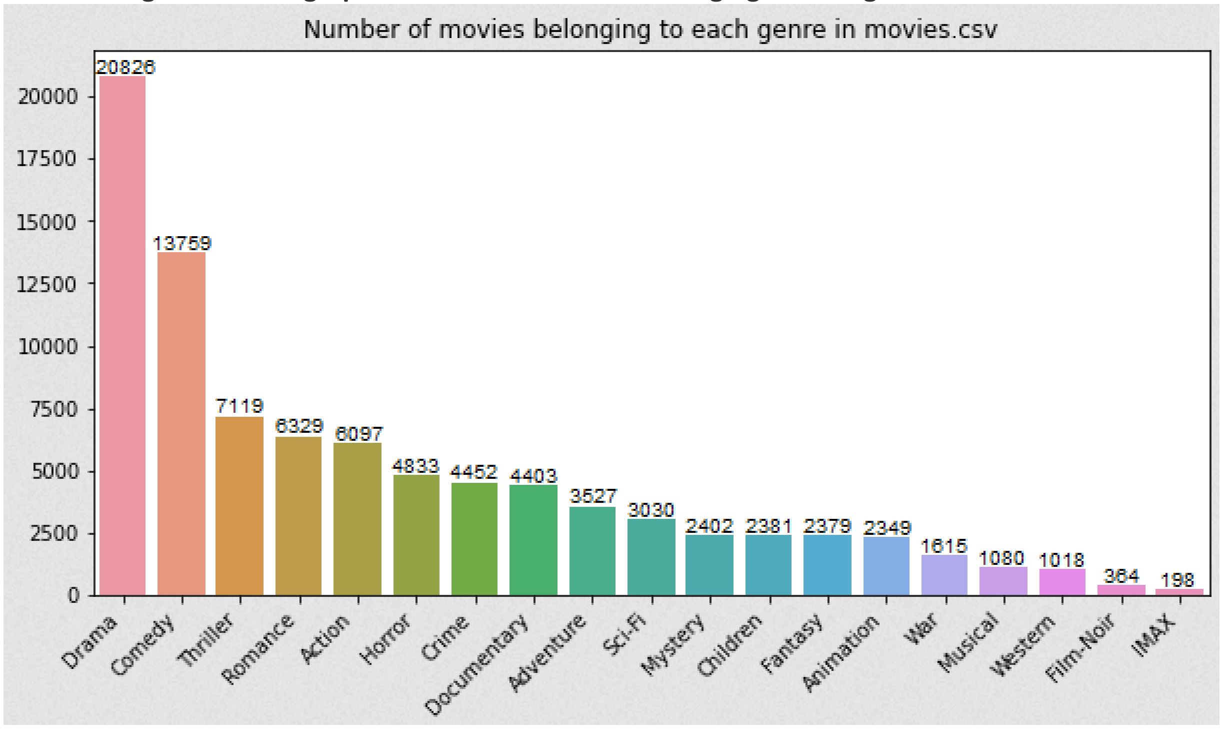
Figure 3.3: Example of movie metadata with an invalid genre



```
title                           Jim Henson's Turkey Hollow (2015)
releaseDate                                                  2015
directedBy                                       Kirk R. Thatcher
starring        Jay Harrington,Mary Steenburgen,Chris Ludacris...
genres                                                    5145828
Name: 148432, dtype: object
```

After removing these invalid values, the following chart can be created showing the most popular genres per movie. The keyword "IMAX" was used to describe the genre of a few movies in the dataset, and although it refers to the aspect ratio and type of cameras used when filming the movie, it was kept as one of the "genres" for the purpose of this study, because it is judged that this could be a deciding factor for a user's selection of a movie.

Figure 3.4: Bar graph of number of movies belonging to each genre in movies.csv



The majority of movies in the dataset belong to the "Drama", and/or "Comedy" genres.

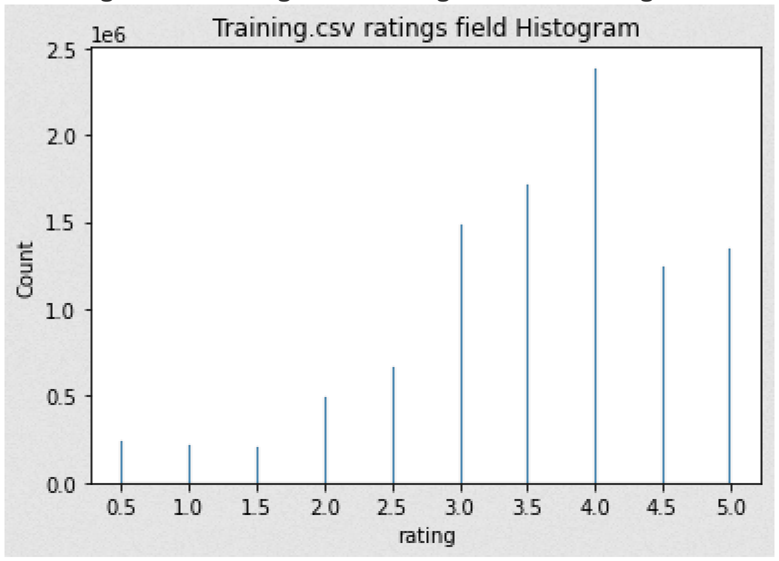
3.3.2 Training.csv

The `training.csv` file contains all user ratings for all movies in the database, from November 11th, 2009, to January 6th, 2018. The file contains almost 10 million ratings (9,997,850) of the following four fields:

- **userId** – a unique identifier for the user
- **movieId** – a unique identifier for the movie
- **rating** – the rating provided by the user
- **timestamp** – a timestamp which indicates when the user had rated the movie

Users can rate movies from 0.5 to 5 stars with the granularity of 0.5 star. The larger number of stars the user assigns a given movie, the more they enjoyed watching it.

Figure 3.5: Histogram of ratings field in training.csv



3.3.2.1 Number of ratings per user

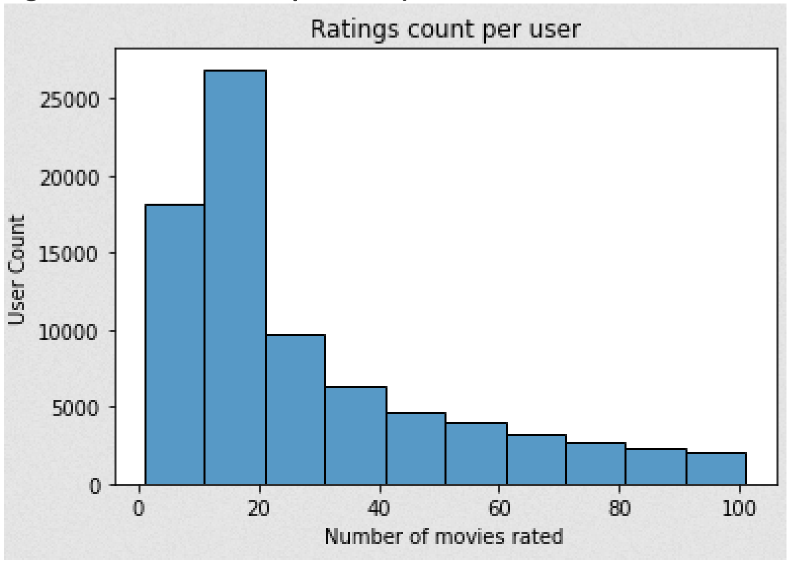
There are 104,661 unique users in this dataset that have rated 49,151 movies. The number of movies each user has rated varies. As shown in the table below, over ¾ of users have rated 100 movies or less.

Table 3.3

Number of ratings provided	**User count**
Over 1,000	974 (0.93%)
Over 500 and less than or equal to 1,000	2,708 (2.58%)
Over 100 and less than or equal to 500	20,713 (19.79%)
Less than or equal to 100	80,266 (76.69%)
TOTAL:	104,661

The histogram below presents the distribution of the number of movies rated for the users that rated less than 100 movies. The largest group is the one that rated between 10 and 20 movies.

Figure 3.6: Histogram of movies rated per user (for users that had rated less than 100 movies)



3.3.2.2 Number of ratings per movie

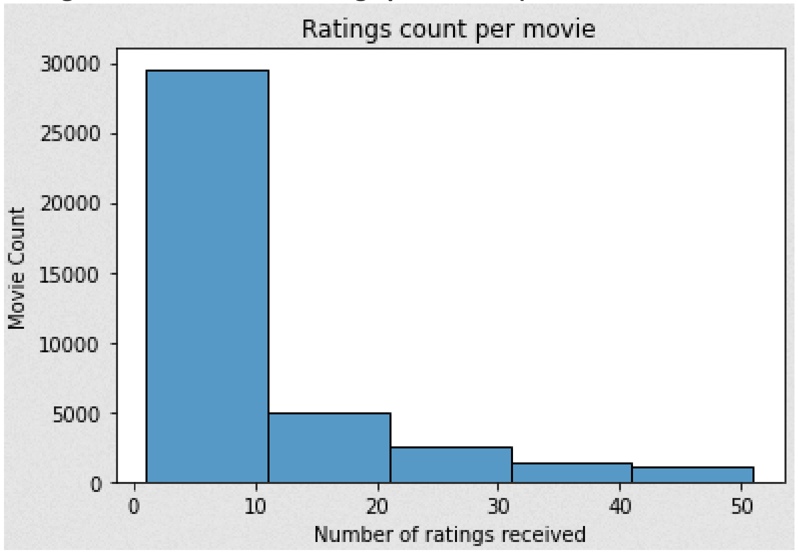
There are 49,151 movies in the dataset which have been rated at least once by users. This means that there remain 23 movies which do not have any ratings (since there are a total of 49.174 movies in the `movies.csv` file). The most rated movie is the 1999 Sci-Fi movie "The Matrix" which has been rated by 42,120 users. This is a very popular movie and is considered an **outlier since** the majority of the movies in the dataset (80%) have been rated 50 times or less.

Table 3.4

Number of ratings received	Movie count
Over 10,000	148 (0.3%)
Over 5,000 and less than or equal to 10,000	264 (0.5%)
Over 1,000 and less than or equal to 5,000	1.651 (3.35%)
Over 50 and less than or equal to 1,000	7,512 (15.2%)
Less than or equal to 50	39,576 (80.51%)
TOTAL:	49,151

After filtering the movies and only keeping the ones with 50 or less ratings, the following histogram is produced. The largest group is the first one (1 - 10 ratings) with almost 30 thousand movies.

Figure 3.7: Histogram of number of ratings per movie (for movies with 50 ratings or less)



The ratings of the user-movie pairs which were involved in the serendipity survey (i.e., the ones found in answers.csv) are excluded from this dataset. However, the users involved in the survey have rated at least one movie in the `training.csv` file.

3.3.3 Answers.csv

The `answers.csv` file contains the result of the serendipity survey conducted for the publication of Denis Kotkov et al. There are a total of 22 fields in the file, but for brevity only the ones relevant to this study are listed below:

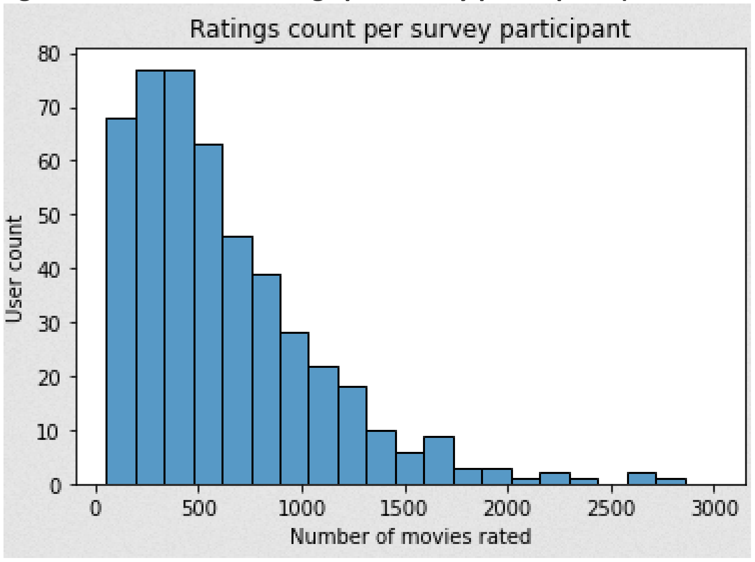
- userId – a unique identifier for the user
- movieId – a unique identifier for the movie
- Rating – the rating for the movie provided by the user
- Timestamp – a timestamp which indicates when the user had rated the movie
- predictedRating – the rating the MovieLens recommendation system had predicted for the user-movie pair
- s1 – user answer to survey statement 1
- s2 – user answer to survey statement 2
- …
- s8 – user answer to survey statement 8

There was a total of 481 users that responded to the survey about a total of 1,678 movies. Note that not all users provided replies on all the questions for all movies. The description of the statements s1 to s8 and how they were answered by users has been described in the section 3.2.1 Survey Description.

3.3.3.1 About the Survey Participants

By studying the `training.csv` file in combination with `answers.csv`, it is concluded that the users that participated in the serendipity survey (i.e., “the participants”) had all rated at least 53 movies prior to their participation. After removing 5 outliers which had rated more than 3,000 movies (top 1%), the following histogram is produced.

Figure 3.8: Histogram of number of ratings per survey participant (after removal of outliers)

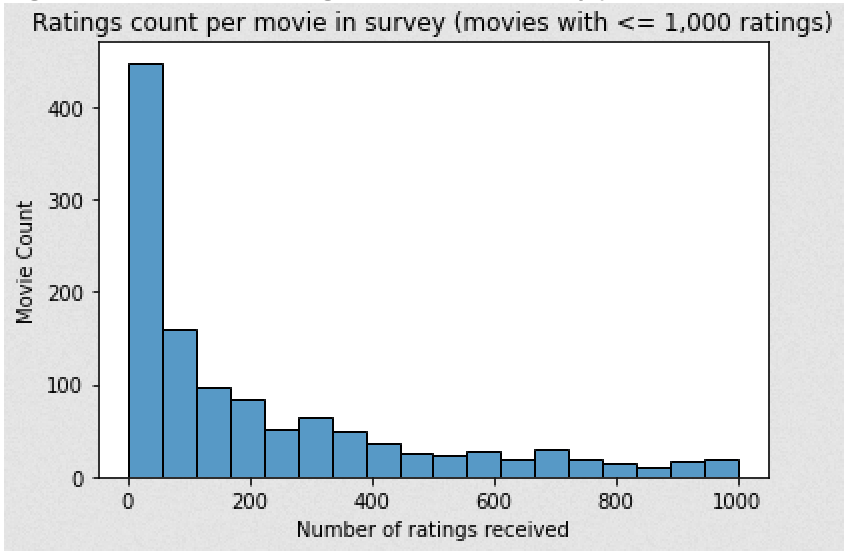


This distribution of ratings count is very different from the overall user ratings distribution presented in section 3.3.2.1 Number of ratings per user, where almost 80% of users had rated 100 movies or less. The survey participants are all active users, having rated at least 53 movies and on average about 500 movies.

3.3.3.2 About the Movies Selected for the Survey

By studying the movies in `training.csv` file in combination with the ones in `answers.csv`, it is revealed that out of the 1,678 movies that were included in the survey, 1,655 had been rated before by other users. In other words, there were 23 movies that had not been rated by any users prior to their participation in the survey. The most popular movie that participated in the survey had received 37,947 ratings (the 2010 movie Inception). However, 72% (1194 movies) had less than or equal to 1,000 ratings. Their distribution is presented in the histogram below:

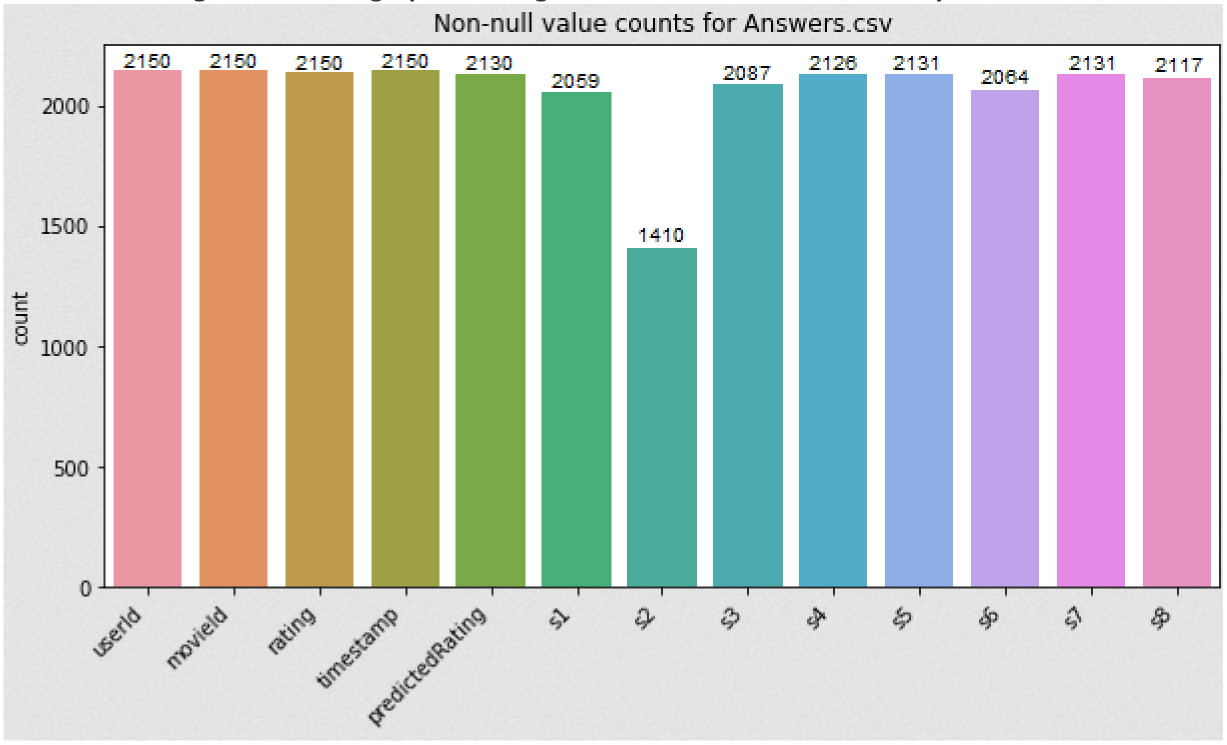
Figure 3.9: Histogram of number of ratings of movies in survey (for movies with 1K ratings or less)



3.3.3.3 Missing values in answers.csv

During the conduction of the survey, each of the 481 participating users were asked questions about 5 movies each, resulting in a total of 2,405 unique user-movie pairs for which answers were requested. However, as mentioned in the related publication [16], not all users answered all questions for all movies. There are a total of 2,150 user-movie pairs for which at least one answer was provided.

Figure 3.10: Bar graph showing non-null value counts for survey answers



3.4 Use of the Dataset in the Context of this Study

The base for this study will be the Answers.csv file. The goal is to create a model which is able to predict how serendipitous a movie will be for a given user if recommended to them from a list of candidate relevant movies. For this, it is necessary to quantify the concept of serendipity using a quantifiable metric: A “serendipity score" can be calculated for each user-movie pair using their answers to the statements in the survey. As mentioned in previous sections, Serendipity in this context is defined as “the occurrence of making pleasant and desirable discoveries by accident”, and ultimately broadening the user's tastes. Consider the statements s4, s5, and s8:

- S4 - “This is the type of movie I would not normally discover on my own; I need a recommender system like MovieLens to find movies like this one.”
- S5 - “This movie is different (e.g., in style, genre, topic) from the movies I usually watch.”
- S8 - “Watching this movie broadened my preferences. Now I am interested in a wider selection of movies.”

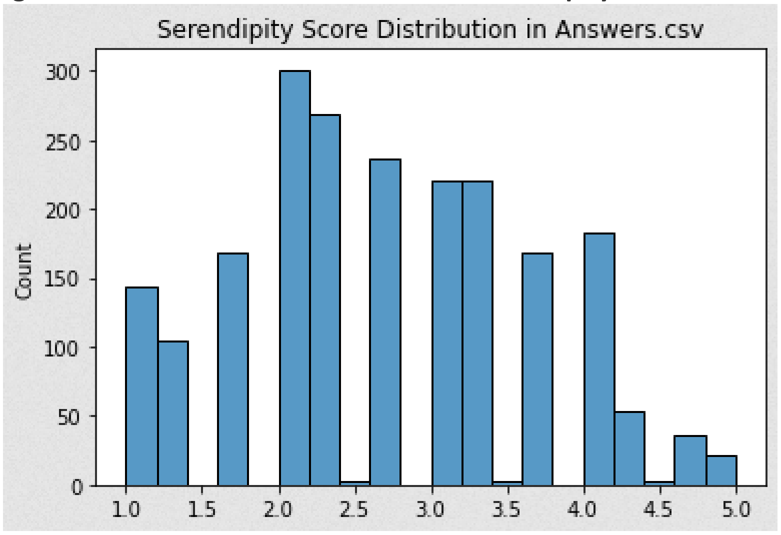
The Serendipity Score is defined by averaging the user answers to the statements s4, s5, s8, using the following formula:

Equation 3.1

$$SerSc = Avg(s4, s5, s8)$$

In the case where a statement answer is missing, the average is calculated for the remaining one(s). This value when calculated for the answers.csv file produces 2,133 non-null values, with their distribution being presented in the following histogram:

Figure 3.11: Distribution of the defined Serendipity Score variable



The Serendipity Score is the target variable that the model will be trained to predict. For a given user-movie pair, the higher the Serendipity Score value the more serendipitous the movie is considered for this user.

In the next chapter, a set of features will be engineered combining the information from the other files that should be appropriate to be used as predictors for the serendipity score.

3.5 Feature Engineering

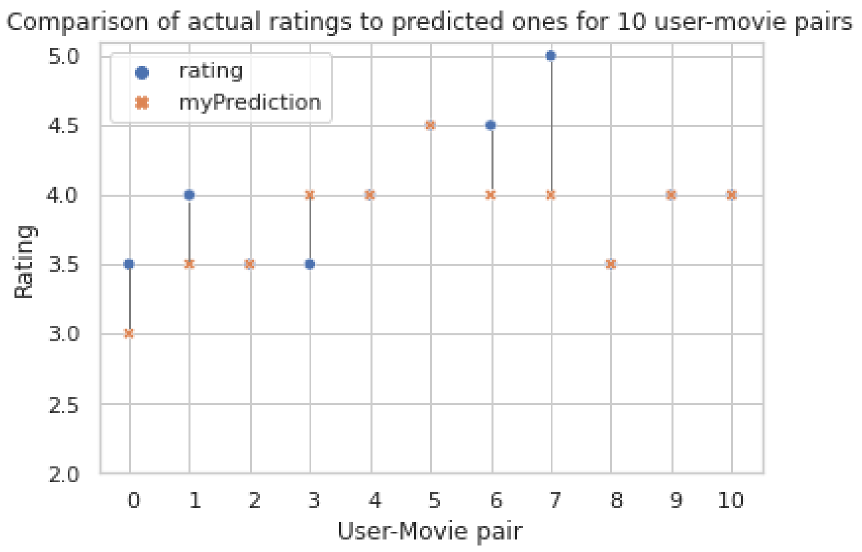
Feature engineering can be described as the process of selection, modification, and transformation of unprocessed data into features suitable to be used as inputs for supervised learning models. In this chapter, the calculation of features for relevance, unexpectedness, and novelty is discussed in detail. Each one attempts to quantify one of the factors that affect or elicit serendipity, and thus are anticipated be effective predictors for the "Serendipity Score" defined in section 3.4 Use of the Dataset in the Context of this Study.

3.5.1 Relevance Feature

A relevance feature should be a metric of a recommended item's usefulness to a specific user. In the context of this dataset, this translates to movies that the user will enjoy watching. The ratings that users provide to the system, which range from 0.5 to 5, are a quantifiable measure of their satisfaction. Consequently, to determine the relevance of an unknown movie to a certain user, a predicted rating value will be utilized.

A model is created for predicting user ratings using the SVD Collaborative Filtering algorithm, trained from the past ratings users had provided (in the "Training.csv" file). The model is then used to provide predicted rating values for each user-movie pair of the survey.

Figure 3.12: Scatter plot comparing actual ratings to predicted ones by SVD for a sample of 10 user-movie pairs



To evaluate the quality of this model, the Mean Absolute Error (MAE) can be calculated by dividing the total absolute errors by the sample size (n).

Equation 3.2

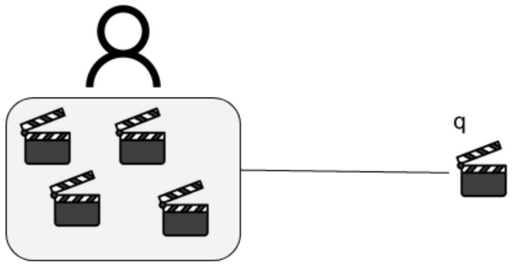
$$MAE = \frac{\sum_{i=1}^{n} |predictedRating - actualRating|}{n}$$

There is a Mean Absolute Error of 0.44 stars when comparing the predicted ratings to the actual ratings that the users reported after seeing the films. This means that on average, the predicted rating is inaccurate by about half a star.

3.5.2 Unexpectedness

In this section, the goal is to define an unexpectedness feature which could evaluate the degree of which a recommendation differs from the users' expectations. In [18], the authors define unexpectedness using the distance between a potentially recommended item and the set of items the user expects to be recommended. Similarly, for the purpose of this study unexpectedness is defined using a distance metric: the distance between a user and a candidate query movie they have not yet watched, based on the movies the user has already rated. The greater the distance, the more unexpected the query movie is to that user.

Figure 3.13: Graphical representation of distance between a user and a query movie q



The "genres" field in the movie information file ("movies.csv") has a list of genres which describe each movie, separated by commas (e.g., "Action, Sci-Fi, Mystery"). The unexpectedness of a movie to a given user can be defined by leveraging this information.

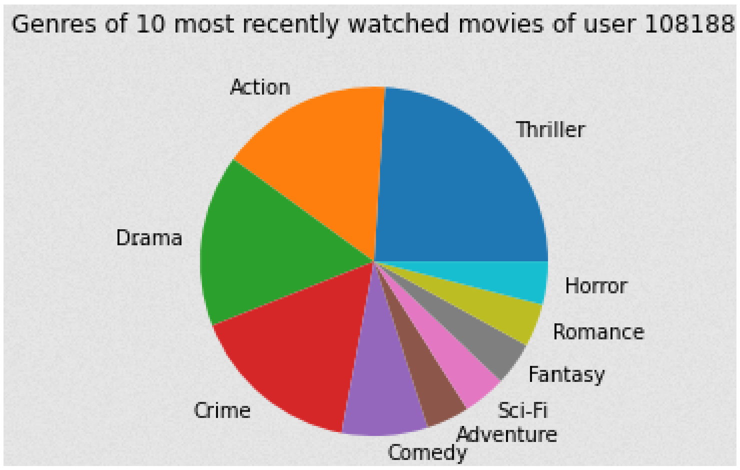
Hypothesizing that the user expects to be recommended movies similar to the ones they have watched in the past, an "Expectation vector" is assembled for the user, which includes the percentages of each genre of movies that the user has watched. However, a user that has watched many movies is likely not to remember or expect all the kinds of movies they have watched. For example, the user with userId "108188", had watched and rated 348 movies prior to their participation in the survey. Instead of using all the movies to calculate the expectation vector, only the 10 most recently watched movies are selected. For the user with userId "108188" the Expectation vector is the following:

Table 3.5

genre	Thriller	Action	Drama	Crime	Comedy	Adventure	Sci-Fi	Fantasy	Romance	Horror
counts	6	4	4	4	2	1	1	1	1	1
Expectation	24%	16%	16%	16%	8%	4%	4%	4%	4%	4%

These percentages describe how much the user expects to be recommended a movie from each genre, based on the movies they have recently watched. The values sum up to 100% and can be visualized in the pie chart below:

Figure 3.14: Pie chart of genres recently watched movies of sample user with id 108188



The Expectation vector is utilized to calculate an "Anticipation percentage" i.e., how much the user anticipates a specific movie recommendation. The percentage of user (u) anticipation for a given movie (m) is the sum of percentages of expectations of the user for each genre that this movie belongs to:

Equation 3.3

$$AnticipationPct(u, m) = \sum_{genre \in m}(Expectation_{u,genre})$$

The unexpectedness metric is considered complementary to the anticipation percentage and can be calculated by subtracting it from 100%.

Equation 3.4

$$Unexpectedness(u, m) = 100\% - AnticipationPct(u, m)$$

For example, the 2016 movie "Café Society" belongs to the genres Comedy, Drama, and Romance. The expectation of the user with userId "108188" to be recommended movies from each of these genres is 8%, 16%, and 4% respectively, summing up to a total of 28% of anticipation, and 72% unexpectedness.

The unexpectedness feature is calculated for all the user-movie pairs of the survey.

3.5.3 Novelty Feature

The definition of a feature related to novelty is established in this section. As mentioned in 1.3 Evaluating Recommendation Systems:, a novel item is one that the user does not know about. Kaminskas and Bridge [12] discuss the concept of novelty in recommendation systems: It is becoming more and more popular to define the novelty of an item in a user-independent fashion rather than the novelty of a recommended item against a target user. This is done to decouple the concept of a simply unknown item from the emotional response conveyed by a serendipitous one (which might not necessarily be unknown to the user). A user is more likely to be aware of a popular item, so novelty is usually defined by researchers using the inverse of popularity.

Following this trend, in this study the novelty feature is defined in a user-independent way, using the inverse of popularity. Popularity for an item pi is defined in the following way:

Equation 3.5

$$p_i = \frac{number\ of\ users\ that\ have\ rated\ item\ i}{number\ of\ users\ that\ have\ rated\ any\ item}$$

During the analysis of the dataset, it was noticed that 80% of the movies in the dataset have been rated by 50 users or less (see 3.3.2.2 Number of ratings per movie). The total number of unique users that have provided ratings is 104,661. This means that the value of p_i for the majority of movies will be very small (< 0.0004).

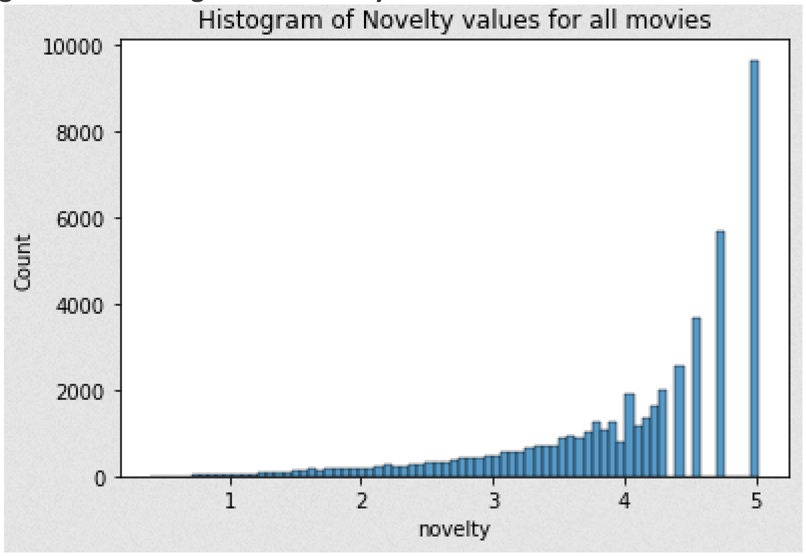
Thus, instead of the multiplicative inverse, to mitigate these extremely small values, the novelty for an item i will be measured as:

Equation 3.6

$$N_i = -\log_{10}(p_i)$$

After calculating the Novelty as defined in the above equation for all the movies in the dataset, the following histogram is produced:

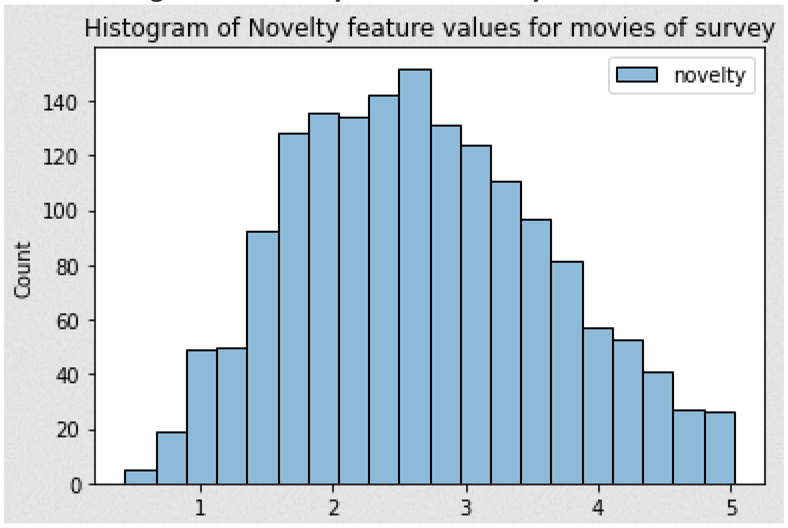
Figure 3.15: Histogram of Novelty feature for all movies in the dataset



To assess the validity of the formula, one can look at the movie with the highest and lowest Novelty: the least novel movie (with a value of $N = 0.39$) is the 1999 movie "The Matrix" which has been rated by 42,120 users, while the most novel movie (with $N = 5$) is the 1981 movie "The Mysterious Castle in the Carpathians" which has only been rated once.

When calculating the novelty feature for the movies of the survey, it is noticed that there are 23 movies that had not been rated by any users prior to their participation in the survey. These movies are assigned the maximum value of novelty ($N = 5$).

Figure 3.16: Histogram of Novelty feature for only the movies of the survey



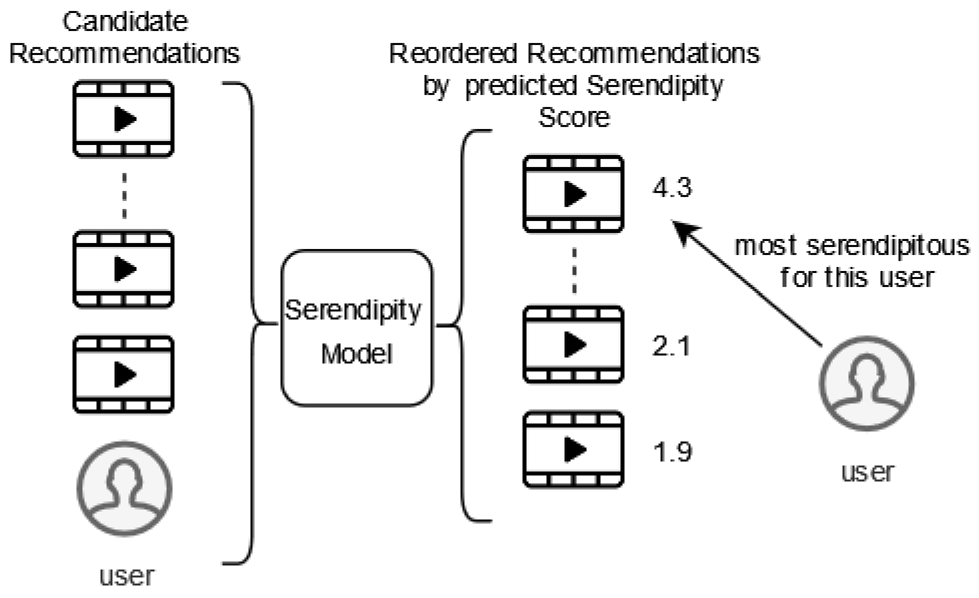
Chapter 4. IMPLEMENTATION

In this section the process of creating a model to predict the serendipity index is documented, and the results are presented.

4.1 METHODOLOGY

The idea is to create a model able to predict the serendipity index for a given user-movie pair. This model can be used to select the most serendipitous movies from a list of candidate recommendations for a given user. Out of the models tested, the Random Forest Regression achieved the best results and is described below.

Figure 4.1: Graphical representation of use of model for reranking a given recommendation list


Candidate Recommendations
user
Serendipity Model
Reordered Recommendations by predicted Serendipity Score
4.3
2.1
1.9
most serendipitous for this user
user

4.1.1 Random Forest Model for Regression

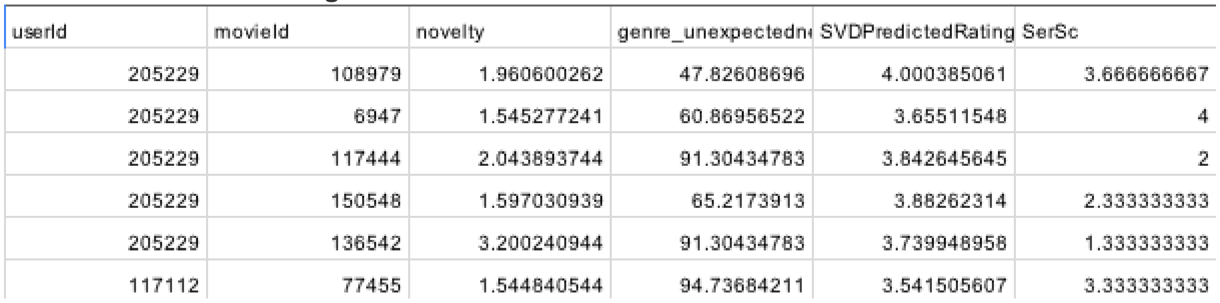
The random forest is an ensemble type of supervised learning model. Ensemble classifiers train a collection of “weak” classifiers on different sub-samples of the training dataset. Random forest uses a collection of Decision Tree regressors, and they are trained separately, without interacting with each other. Once trained, they can be used to perform a voting process to decide on a prediction for a given input. Random forest is useful for this case because it can handle missing values in data gracefully.

4.2 MODEL TRAINING AND EVALUATION

For the training of the model, the three features described in Chapter 3 (Relevance, Unexpectedness, Novelty) were calculated for each of the user-movie pairs that took part in the survey, as well as the

Serendipity Score (SerSc) which will be the target variable. A screenshot of the start of the final dataset file to be used for training and testing is shown below, there are a total of 1,566 samples.

Figure 4.2: Screenshot of final dataset's first 6 entries



userId	movieId	novelty	genre_unexpectedn	SVDPredictedRating	SerSc
205229	108979	1.960600262	47.82608696	4.000385061	3.666666667
205229	6947	1.545277241	60.86956522	3.65511548	4
205229	117444	2.043893744	91.30434783	3.842645645	2
205229	150548	1.597030939	65.2173913	3.88262314	2.333333333
205229	136542	3.200240944	91.30434783	3.739948958	1.333333333
117112	77455	1.544840544	94.73684211	3.541505607	3.333333333

Before initiating training, the model hyperparameters have to be configured. These parameters control the behavior of the model during the learning process. In the case of the Random Forest regressor, two important parameters that must be set are the number of Decision Trees that will be used and their maximal depth.

4.2.1 Grid Search Hyperparameter Optimization

Grid Search is a tuning method that can be used to determine the optimal hyperparameter values. It conducts an exhaustive search through a specified subset of the hyperparameter possible values. For the "Number of Trees" hyperparameter, the values in the range 1 to zero are selected with 10 steps (1, 10, 20, …, 100) and for the "Maximal Depth" the range 10 to 100 with 10 steps is explored as well as the "no bound" option, where the trees do not have a set maximal depth. The cartesian product of these ranges sets the bounds of the hyperparameter space to be explored by Grid Search: 11 different values for each of the two hyperparameters, resulting in a total of 121 different possible hyperparameter combinations to be evaluated.

Table 4.1

Parameter	Min	Max	Step
Number of Trees	1	100	10
Maximal Depth	10	100	10

Grid Search trains and evaluates a Random Forest regressor with each of these possible hyperparameters and selects the values which received the highest score during the validation process.

In this case the best performance was achieved using 50 trees of maximal depth 10.

4.2.2 K-Fold Cross Validation Training and Testing

Because the dataset is small, the K-Fold Cross Validation method will be used for training and evaluating the model. Cross Validation is a resampling technique that uses different subsets of the data to train and test a model across a k number of iterations (folds). First, the input dataset is divided into k equal-sized subsets. During each iteration, one subset is kept as the test dataset (to be used for evaluating the model), and the remaining k-1 subsets are used to train the model. The process is repeated k times, with each of the subsets used exactly once as the testing set. A value of 10 is selected for the k parameter because it has been determined through experimentation to typically produce a model with low bias and little variance [27].

4.2.3 Results

For evaluating the performance of a regression model, it is common to use error scores. The Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE) are commonly used. MAE has been defined in section, but the equations for calculation of both will be provided here:

Equation 4.1

$$MAE = \frac{\sum_{i=1}^{N} |Predicted_i - Actual_i|}{N}$$

Equation 4.2

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

Unlike MAE which considers the absolute of all errors, treating them equally, Root Mean Squared Error penalizes the large prediction errors. The final Random Forest model for predicting the serendipity score has an RMSE of 0.95 and a MAE of 0.79.

Table 4.2

Score	Value
RMSE	0.95
MAE	0.79

These scores do not seem very promising, however, although the model does not predict the serendipity score with high accuracy, it is theorized that it could be useful as an indicator of serendipity in the context of reranking produced recommendations. There were limitations considering the dataset used for training and evaluation is small and includes movies that the users had rated highly and that are considered unpopular. To the best of the writer's knowledge, there is no other public dataset with explicit user feedback which could be used to further train and evaluate the model.

Chapter 5. CONCLUSIONS AND FUTURE WORK

The nature of serendipity is unpredictable. Researchers in the recommendation field have not yet reached consensus on its definition, because it is a subjective concept related to an emotional response. It is dependent on the experience and individual preference of the user, and influenced by external factors, such as mood and context. Additionally, serendipity is associated with novelty, another concept that is difficult to predict. These challenges make it difficult to incorporate serendipity in a recommendation system.

In this study, the process of creating a model to predict a serendipity score for movie recommendations based on real user feedback was described. The dataset used was analyzed in detail, and through the study of related research, three features were generated to be used for predicting serendipity of a movie. The Random Forest model was selected for predicting the target variable and was trained and evaluated using 10-Fold Cross Validation. It is speculated that the model could be useful as an indicator of serendipity in the context of reranking a recommendations list for a specific user. It was not possible to evaluate the model in a real-world scenario.

As a possible extension of this work, the MovieLens Tag Genome (a collection of user-generated tags which associate to movies) could be used to calculate unexpectedness of a movie for a given user, as it provides more details on the style, context, and topic of the movie besides the genre. This could be added as an extra feature to the model to see the influence on the prediction of serendipity.

REFERENCES

[1] P. Kumar and R. S. Thakur, "Recommendation system techniques and related issues: a survey," *International Journal of Information Technology,* vol. 10, pp. 495--501, April 2018.

[2] M. J. Pazzani and D. Billsus, "Content-Based Recommendation Systems," in *The Adaptive Web*, Springer Berlin Heidelberg, 2007, pp. 325--341.

[3] P. Lops, M. de Gemmis and G. Semeraro, "Content-based Recommender Systems: State of the Art and Trends," in *Recommender Systems Handbook*, Boston, MA, Springer US, 2011, pp. 73--105.

[4] X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," *Advances in Artificial Intelligence,* vol. 2009, p. 1–19, October 2009.

[5] Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer,* vol. 42, p. 30–37, August 2009.

[6] H. Koohi and K. Kiani, "User based Collaborative Filtering using fuzzy C-means," *Measurement,* vol. 91, pp. 134-139, 2016.

[7] G. Linden, B. Smith and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing,* vol. 7, p. 76–80, January 2003.

[8] X. N. Lam, T. Vu, T. D. Le and A. D. Duong, "Addressing cold-start problem in recommendation systems," in *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, 2008.

[9] F. O. Isinkaye, Y. O. Folajimi and B. A. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal,* vol. 16, pp. 261-273, 2015.

[10] S. M. McNee, J. Riedl and J. A. Konstan, "Being accurate is not enough," in *'06 Extended Abstracts on Human Factors in Computing Systems*, 2006.

[11] P. Adamopoulos, "Beyond rating prediction accuracy," in *Proceedings of the 7th ACM conference on Recommender systems*, 2013.

[12] M. Kaminskas and D. Bridge, "Diversity, Serendipity, Novelty, and Coverage," *ACM Transactions on Interactive Intelligent Systems,* vol. 7, p. 1–42, December 2016.

[13] L. Chen, Y. Yang, N. Wang, K. Yang and Q. Yuan, "How Serendipity Improves User Satisfaction with Recommendations? A Large-Scale User Evaluation," in *The World Wide Web Conference on - WWW'19*, 2019.

[14] D. Kotkov, S. Wang and J. Veijalainen, "A survey of serendipity in recommender systems," *Knowledge-Based Systems,* vol. 111, p. 180–192, November 2016.

[15] R. J. Ziarani and R. Ravanmehr, "Serendipity in Recommender Systems: A Systematic Literature Review," *Journal of Computer Science and Technology,* vol. 36, p. 375–396, March 2021.

[16] D. Kotkov, J. A. Konstan, Q. Zhao and J. Veijalainen, "Investigating serendipity in recommender systems based on real user feedback," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, 2018.

[17] J. L. Herlocker, J. A. Konstan, L. G. Terveen and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems,* vol. 22, p. 5–53, January 2004.

[18] P. Adamopoulos and A. Tuzhilin, "On Unexpectedness in Recommender Systems," *ACM Transactions on Intelligent Systems and Technology,* vol. 5, p. 1–32, December 2014.

[19] D. Kotkov, J. Veijalainen and S. Wang, "How does serendipity affect diversity in recommender systems? A serendipity-oriented greedy algorithm," *Computing,* vol. 102, p. 393–411, December 2018.

[20] C.-N. Ziegler, S. M. McNee, J. A. Konstan and G. Lausen, "Improving recommendation lists through topic diversification," in *Proceedings of the 14th international conference on World Wide Web - WWW \textquotesingle05*, 2005.

[21] A. A. Deshmukh, P. Nair and S. Rao, "A Scalable Clustering Algorithm for Serendipity in Recommender Systems," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2018.

[22] Y. Yang, Y. Xu, E. Wang, J. Han and Z. Yu, "Improving Existing Collaborative Filtering Recommendations via Serendipity-Based Algorithm," *IEEE Transactions on Multimedia,* vol. 20, p. 1888–1900, July 2018.

[23] N. Tuval, "Exploring the Potential of the Resolving Sets Model for Introducing Serendipity to Recommender Systems," in *Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization*, 2019.

[24] M. Kaminskas and D. Bridge, "Measuring Surprise in Recommender Systems," in *Proceedings of the ACM RecSys Workshop on Recommender Systems Evaluation: Dimensions and Design (REDD'14)*, 2014.

[25] R. J. Ziarani and R. Ravanmehr, "Deep neural network approach for a serendipity-oriented recommendation system," *Expert Systems with Applications,* vol. 185, p. 115660, December 2021.

[26] J. Corneli, A. Jordanous, C. Guckelsberger, A. Pease and S. Colton, *Modelling serendipity in a computational context,* 2014.

[27] G. James, D. Witten and T. Hastie, An Introduction to Statistical Learning: With Applications in R, SPRINGER NATURE, 2017.